

# TCAN5102-Q1 Automotive Self-supplied CAN FD Light Responder to SPI, UART, or I<sup>2</sup>C Controller

## 1 Features

- AEC-Q100: Qualified for automotive applications
- CAN FD Light Responder meets ISO 11898-1:202x
- Supports CAN FD Light data rates up to 5Mbps when using TCAN4572-Q1 CAN FD Light Commander
- Ability to control external CAN transceiver such as: TCAN1162x-Q1, TCAN1043A, TCAN1463A
- Programmable via CAN bus
- 3.3V to 5V Supply voltage
- SPI controller support for up to 20MHz with up to 8 chip selects
- UART controller support for up to 2.5 Mbaud
- I<sup>2</sup>C controller supports fast mode plus (1MHz)
- 13 GPIOs which can be muxed with other functionality
- Programmable PWM output with trapezoidal ramp profile support for motor control
- 20-pin VSSOP (DGQ) package

## 2 Applications

- [Body electronics and lighting](#)
- [Hybrid, electric and powertrain systems](#)
- [Infotainment and cluster](#)
- [Appliances](#)

## 3 Description

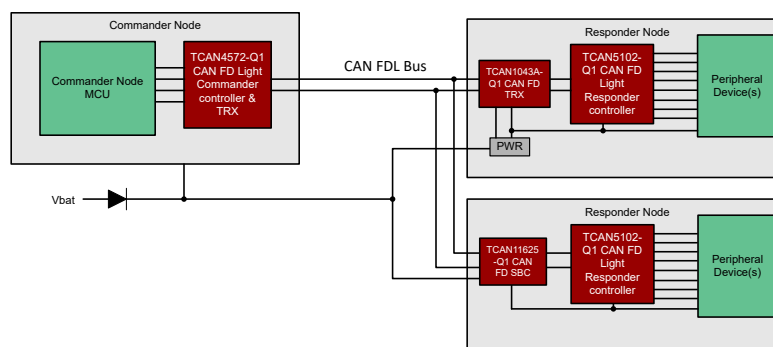
The TCAN5102-Q1 is a Control Area Network (CAN) Flexible Data (FD) Light responder device compatible with CiA 604-1 and ISO 11898-1:2024, capable of supporting up to 5Mbps data rate when a dedicated CAN FD light commander is used. The device is designed to support CAN FD light responder node applications in a commander - responder architecture which does not require a responder node processor. All control to the responder node is through the CAN bus from the commander node processor which eliminates the need for responder node processor and software.

The device receives data and/or commands from a CAN FD light commander node which translates these to either Serial Peripheral Interface (SPI) controller, UART controller, I<sup>2</sup>C controller, and/or GPIO pins communication to the device and/or peripheral devices the TCAN5102-Q1 is controlling. PWM output channels also support trapezoidal ramp profiles in hardware for controlling stepper motors. Ramping of duty cycle or frequency is possible. No external crystal or clock is required. The device controls the external TCAN1162x-Q1, TCAN1043A-Q1 or TCAN1463A-Q1 CAN FD (SIC) transceivers for system level flexibility. The device relies upon the CAN FD transceiver/SBC to control the node power and communicate a wake up signal to the TCAN5102-Q1 by latching the CAN RXD (CRXD) pin low.

### Package Information

| PART NUMBER | PACKAGE <sup>(1)</sup> | PACKAGE SIZE <sup>(2)</sup> |
|-------------|------------------------|-----------------------------|
| TCAN5102-Q1 | 20-Pin (VSSOP, DGQ)    | 5.1mm × 4.9mm               |

- (1) For more information, see [Section 11](#).  
 (2) The package size (length × width) is a nominal value and includes pins, where applicable.



**CAN FD Light Commander-Responder Application**

**ADVANCE INFORMATION**

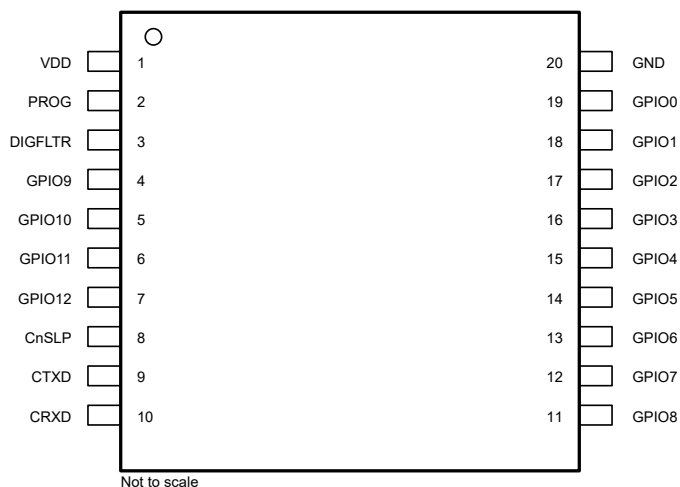


An IMPORTANT NOTICE at the end of this data sheet addresses availability, warranty, changes, use in safety-critical applications, intellectual property matters and other important disclaimers. ADVANCE INFORMATION for preproduction products; subject to change without notice.

## Table of Contents

|                                                   |           |                                                                  |            |
|---------------------------------------------------|-----------|------------------------------------------------------------------|------------|
| <b>1 Features</b> .....                           | <b>1</b>  | 7.4 Device Functional Modes.....                                 | <b>19</b>  |
| <b>2 Applications</b> .....                       | <b>1</b>  | 7.5 Programming.....                                             | <b>20</b>  |
| <b>3 Description</b> .....                        | <b>1</b>  | 7.6 Register Maps.....                                           | <b>55</b>  |
| <b>4 Pin Configuration and Functions</b> .....    | <b>3</b>  | <b>8 Application and Implementation</b> .....                    | <b>225</b> |
| <b>5 Specifications</b> .....                     | <b>4</b>  | 8.1 Application Information.....                                 | <b>225</b> |
| 5.1 Absolute Maximum Ratings.....                 | <b>4</b>  | 8.2 Typical Application.....                                     | <b>225</b> |
| 5.2 ESD Ratings.....                              | <b>4</b>  | 8.3 Power Supply Recommendations.....                            | <b>226</b> |
| 5.3 Recommended Operating Conditions.....         | <b>4</b>  | 8.4 Layout.....                                                  | <b>226</b> |
| 5.4 Thermal Information.....                      | <b>4</b>  | <b>9 Device and Documentation Support</b> .....                  | <b>228</b> |
| 5.5 Supply Characteristics.....                   | <b>5</b>  | 9.1 Documentation Support.....                                   | <b>228</b> |
| 5.6 Electrical Characteristics.....               | <b>5</b>  | 9.2 Receiving Notification of Documentation Updates.....         | <b>228</b> |
| 5.7 Timing Requirements.....                      | <b>6</b>  | 9.3 Support Resources.....                                       | <b>228</b> |
| 5.8 Switching Characteristics.....                | <b>7</b>  | 9.4 Trademarks.....                                              | <b>228</b> |
| 5.9 I <sup>2</sup> C Bus Timing Requirements..... | <b>9</b>  | 9.5 Electrostatic Discharge Caution.....                         | <b>228</b> |
| <b>6 Parameter Measurement Information</b> .....  | <b>11</b> | 9.6 Glossary.....                                                | <b>228</b> |
| <b>7 Detailed Description</b> .....               | <b>14</b> | <b>10 Revision History</b> .....                                 | <b>228</b> |
| 7.1 Overview.....                                 | <b>14</b> | <b>11 Mechanical, Packaging, and Orderable Information</b> ..... | <b>228</b> |
| 7.2 Functional Block Diagram.....                 | <b>14</b> | 11.1 Mechanical Data.....                                        | <b>229</b> |
| 7.3 Feature Description.....                      | <b>16</b> |                                                                  |            |

## 4 Pin Configuration and Functions



**Figure 4-1. DGQ 20-PIN (Top View)**

**Table 4-1. Pin Functions**

| PIN             |     | TYPE <sup>(1)</sup> | DESCRIPTION                                                                                          |
|-----------------|-----|---------------------|------------------------------------------------------------------------------------------------------|
| NAME            | NO. |                     |                                                                                                      |
| VDD             | 1   | P                   | Device power input, 3.3V or 5V                                                                       |
| PROG            | 2   | I                   | Input for device configuration and debug                                                             |
| DIGFLTR         | 3   | P                   | Digital power supply filter pin                                                                      |
| GPIO9/CS4/SCL   | 4   | I/O                 | General purpose input/output pin, SPI chip select channel 4 output, or I <sup>2</sup> C clock output |
| GPIO10/CS5/SDA  | 5   | I/O                 | General purpose input/output pin, SPI chip select channel 5 output, or I <sup>2</sup> C data pin     |
| GPIO11/CS6/PWM0 | 6   | I/O                 | General purpose input/output pin, SPI chip select channel 6 output, or PWM0 output                   |
| GPIO12/CS7/PWM1 | 7   | I/O                 | General purpose input/output pin, SPI chip select channel 6 output, or PWM1 output                   |
| CnSLP           | 8   | O                   | CAN transceiver control output pin (nSLP)                                                            |
| CTXD            | 9   | O                   | CAN transceiver TXD output pin (connects to transceiver's TXD input pin)                             |
| CRXD            | 10  | I                   | CAN transceiver RXD input pin (connects to transceiver's RXD output pin)                             |
| GPIO8/URXD      | 11  | I/O                 | General purpose input/output pin or UART RXD input                                                   |
| GPIO7/UTXD      | 12  | I/O                 | General purpose input/output pin or UART TXD output                                                  |
| GPIO6/PICO      | 13  | I/O                 | General purpose input/output pin or SPI peripheral in, controller out output                         |
| GPIO5/POCI      | 14  | I/O                 | General purpose input/output pin or SPI peripheral out, controller in input                          |
| GPIO4/SCK       | 15  | I/O                 | General purpose input/output pin or SPI clock output                                                 |
| GPIO3/CS0       | 16  | I/O                 | General purpose input/output pin or SPI chip select channel 0 output                                 |
| GPIO2/CS1       | 17  | I/O                 | General purpose input/output pin or SPI chip select channel 1 output                                 |
| GPIO1/CS2       | 18  | I/O                 | General purpose input/output pin or SPI chip select channel 2 output                                 |
| GPIO0/CS3       | 19  | I/O                 | General purpose input/output pin or SPI chip select channel 3 output                                 |
| GND             | 20  | P                   | Ground pin                                                                                           |

(1) I = Input, O = Output, I/O = Input or Output, G = Ground, P = Power.

### Note

For pins that have multiple functions see the IO\_CFG registers to configure the pins and see [Section 7.3.4](#) for more information about the pin muxing.

## 5 Specifications

### 5.1 Absolute Maximum Ratings

over operating free-air temperature range (unless otherwise noted)<sup>(1)</sup>

|                        |                                | MIN  | MAX | UNIT |
|------------------------|--------------------------------|------|-----|------|
| VDD                    | Supply input                   | −0.3 | 6   | V    |
| DIGFLTR                | Digital core filter pin        | −0.3 | 1.8 | V    |
| V <sub>LOGIC_IN</sub>  | Logic pin input voltage range  | −0.3 | 6   | V    |
| V <sub>LOGIC_OUT</sub> | Logic pin output voltage range | −0.5 | 6   | V    |
| I <sub>O(LOGIC)</sub>  | Logic pin output current       |      | 12  | mA   |
| T <sub>J</sub>         | Junction temperature           | −40  | 150 | °C   |
| T <sub>stg</sub>       | Storage temperature            | −65  | 150 | °C   |

- (1) Operation outside the Absolute Maximum Ratings may cause permanent device damage. Absolute Maximum Ratings do not imply functional operation of the device at these or any other conditions beyond those listed under Recommended Operating Conditions. If used outside the Recommended Operating Conditions but within the Absolute Maximum Ratings, the device may not be fully functional, and this may affect device reliability, functionality, performance, and shorten the device lifetime.

### 5.2 ESD Ratings

|                    |                         |                                                                                                 | VALUE | UNIT |
|--------------------|-------------------------|-------------------------------------------------------------------------------------------------|-------|------|
| V <sub>(ESD)</sub> | Electrostatic discharge | Human body model (HBM) Classification Level 3A, all other pins, per AEC Q100-002 <sup>(1)</sup> | ±2000 | V    |
|                    |                         | Charged device model (CDM) Classification Level C5, per AEC Q100-011                            | ±750  |      |
|                    |                         | Corner pins                                                                                     | ±750  |      |
|                    |                         | Other pins                                                                                      | ±750  |      |

- (1) AEC Q100-002 indicates that HBM stressing shall be in accordance with the ANSI/ESDA/JEDEC JS-001 specification.

### 5.3 Recommended Operating Conditions

over operating free-air temperature range (unless otherwise noted)

|                        |                                                    | MIN   | NOM | MAX | UNIT |
|------------------------|----------------------------------------------------|-------|-----|-----|------|
| VDD                    | Supply voltage                                     | 3.0   |     | 5.5 | V    |
| I <sub>OH(DO)</sub>    | Digital output high level current                  | −12   |     |     | mA   |
| I <sub>OL(DO)</sub>    | Digital output low level current                   |       |     | 12  | mA   |
| C <sub>(VDD)</sub>     | VDD supply capacitance                             | 1     |     |     | μF   |
| C <sub>(DIGFLTR)</sub> | Digital core filter pin capacitance                | 0.33  | 1   |     | μF   |
| ESR <sub>CO</sub>      | Supply and filter pin ESR capacitance requirements | 0.001 |     | 1   | Ω    |
| T <sub>J</sub>         | Operating junction temperature range               | −40   |     | 125 | °C   |

### 5.4 Thermal Information

| THERMAL METRIC <sup>(1)</sup> |                                              | TCAN5102-Q1 | UNIT |
|-------------------------------|----------------------------------------------|-------------|------|
|                               |                                              | RGY (QFN)   |      |
|                               |                                              | 24-PINS     |      |
| R <sub>θJA</sub>              | Junction-to-ambient thermal resistance       | 31.8        | °C/W |
| R <sub>θJB</sub>              | Junction-to-board thermal resistance         | 11.8        | °C/W |
| R <sub>θJC(top)</sub>         | Junction-to-case (top) thermal resistance    | 21.4        | °C/W |
| R <sub>θJC(bot)</sub>         | Junction-to-case (bottom) thermal resistance | 2.8         | °C/W |
| Ψ <sub>JT</sub>               | Junction-to-top characterization parameter   | 0.3         | °C/W |

## 5.4 Thermal Information (continued)

| THERMAL METRIC <sup>(1)</sup> |                                              | TCAN5102-Q1 | UNIT |
|-------------------------------|----------------------------------------------|-------------|------|
|                               |                                              | RGY (QFN)   |      |
|                               |                                              | 24-PINS     |      |
| $\Psi_{JB}$                   | Junction-to-board characterization parameter | 11.8        | °C/W |

(1) For more information about traditional and new thermal metrics, see the [Semiconductor and IC Package Thermal Metrics](#) application report.

## 5.5 Supply Characteristics

over operating free-air temperature range (unless otherwise noted)

| PARAMETER           |                                          | TEST CONDITIONS                                                                                                | MIN  | TYP | MAX  | UNIT |
|---------------------|------------------------------------------|----------------------------------------------------------------------------------------------------------------|------|-----|------|------|
| <b>VDD</b>          |                                          |                                                                                                                |      |     |      |      |
| IDD                 | VDD supply current device in normal mode | Normal mode; VDD = 5 V<br>$I_I = 0, 0 \text{ V or VDD}$<br>SPI, UART, I2C disabled                             |      | 1.4 | 1.8  | mA   |
|                     |                                          | Normal mode; VDD = 3.3 V<br>$I_I = 0, 0 \text{ V or VDD}$<br>SPI, UART, I2C disabled                           |      | 1.4 | 1.8  | mA   |
| IDD                 | VDD supply current device in sleep mode  | Sleep mode; VDD = 5 V<br>$I_I = 0, 0 \text{ V or VDD}$<br>$-40^\circ\text{C} \leq T_J \leq 85^\circ\text{C}$   |      | 10  | 20   | μA   |
|                     |                                          | Sleep mode; VDD = 5 V<br>$I_I = 0, 0 \text{ V or VDD}$<br>$T_J > 85^\circ\text{C}$                             |      | 20  | 60   | μA   |
|                     |                                          | Sleep mode; VDD = 3.3 V<br>$I_I = 0, 0 \text{ V or VDD}$<br>$-40^\circ\text{C} \leq T_J \leq 85^\circ\text{C}$ |      | 10  | 20   | μA   |
|                     |                                          | Sleep mode; VDD = 3.3 V<br>$I_I = 0, 0 \text{ V or VDD}$<br>$T_J > 85^\circ\text{C}$                           |      | 20  | 60   | μA   |
| UV <sub>DDR</sub>   | Supply undervoltage rising threshold     | VDD rising                                                                                                     | 2.35 |     | 2.95 | V    |
| UV <sub>DDF</sub>   | Supply undervoltage falling threshold    | VDD falling                                                                                                    | 2.1  |     | 2.7  | V    |
| UV <sub>DDHYS</sub> | Supply undervoltage detection hysteresis |                                                                                                                |      | 350 | 430  | mV   |

## 5.6 Electrical Characteristics

over operating free-air temperature range (unless otherwise noted)

| PARAMETER             |                          | TEST CONDITIONS | MIN | TYP | MAX | UNIT |
|-----------------------|--------------------------|-----------------|-----|-----|-----|------|
| <b>GPIOs (Inputs)</b> |                          |                 |     |     |     |      |
| V <sub>IH</sub>       | High-level input voltage | GPIOx pins      | 0.7 |     |     | VDD  |
|                       |                          | CRXD pin        | 0.7 |     |     |      |
|                       |                          | PROG pin        | 0.7 |     |     |      |
| V <sub>IL</sub>       | Low-level input voltage  | GPIOx pins      |     |     | 0.3 | VDD  |
|                       |                          | CRXD pin        |     |     | 0.3 |      |
|                       |                          | PROG pin        |     |     | 0.3 |      |

## 5.6 Electrical Characteristics (continued)

over operating free-air temperature range (unless otherwise noted)

| PARAMETER             |                                  | TEST CONDITIONS                                              | MIN  | TYP  | MAX | UNIT |
|-----------------------|----------------------------------|--------------------------------------------------------------|------|------|-----|------|
| I <sub>IH</sub>       | High-level input leakage current | GPIOx pins<br>Input = VDD<br>Pull-up/down resistors disabled | -1   |      | 1   | μA   |
|                       |                                  | CRXD pin<br>Input = VDD                                      | -1   |      | 1   |      |
|                       |                                  | PROG pin<br>Input = VDD                                      |      | 90   | 140 |      |
| I <sub>IL</sub>       | Low-level input leakage current  | GPIOx pins<br>Input = 0 V<br>Pull-up/down resistors disabled | -1   |      | 1   | μA   |
|                       |                                  | PROG pins<br>Input = 0 V                                     | -1   |      | 1   |      |
|                       |                                  | CRXD<br>Input = 0 V                                          | -140 | -90  |     |      |
| C <sub>IN</sub>       | Input Capacitance                | 20 MHz, GPIOx pins                                           |      | 4    | 10  | pF   |
|                       |                                  | 1 MHz, CRXD pin                                              |      | 4    | 10  |      |
|                       |                                  | 1 MHz, PROG pin                                              |      | 4    | 10  |      |
| I <sub>LKG(OFF)</sub> | Unpowered leakage current        | GPIOx pins<br>Inputs = 5.5 V, VDD = 0 V                      | -1   | 0    | 1   | μA   |
| R <sub>PD</sub>       | Pull-down Resistance             | GPIOx (if enabled)                                           | 40   | 60   | 80  | kΩ   |
|                       |                                  | PROG                                                         | 40   | 60   | 80  |      |
| R <sub>PU</sub>       | Pull-up Resistance               | GPIOx (if enabled)                                           | 40   | 60   | 80  | kΩ   |
|                       |                                  | CRXD                                                         | 40   | 60   | 80  |      |
| GPIOs (Outputs)       |                                  |                                                              |      |      |     |      |
| V <sub>OH</sub>       | High level output voltage        | GPIOx, CTXD, CnSLP pins I <sub>OH</sub> = 10 mA              | 0.7  | 0.85 |     | VDD  |
|                       |                                  | GPIOx, CTXD, CnSLP pins I <sub>OH</sub> = 2 mA               | 0.8  | 0.9  |     |      |
| V <sub>OL</sub>       | Low level output voltage         | GPIOx, CTXD, CnSLP pins I <sub>OL</sub> = -10 mA             |      | 0.15 | 0.3 | VDD  |
|                       |                                  | GPIOx, CTXD, CnSLP pins I <sub>OL</sub> = -2 mA              |      | 0.1  | 0.2 |      |
| I <sub>OH</sub>       | High level output current        | GPIOx, CTXD, CnSLP pins<br>V <sub>O</sub> = 0.7 VDD          | 10   | 12   |     | mA   |
| I <sub>OL</sub>       | Low Level output current         | GPIOx, CTXD, CnSLP pins<br>V <sub>O</sub> = 0.3 VDD          |      | -12  | -10 | mA   |
| I <sub>LKG(OFF)</sub> | Unpowered leakage current        | CTXD, CnSLP pins<br>VPIN = 5.5 V                             | -5   |      | 5   | μA   |
| Thermal Shutdown      |                                  |                                                              |      |      |     |      |
| TSD <sub>F</sub>      | Thermal shut down falling        |                                                              | 155  | 171  | 180 | °C   |
| TSD <sub>R</sub>      | Thermal shut down rising         |                                                              | 160  | 176  | 190 | °C   |
| TSD <sub>HYS</sub>    | Thermal shut down hysteresis     |                                                              | 2    | 5    | 10  | °C   |

## 5.7 Timing Requirements

|                      |                                                                      | MIN | TYP | MAX | UNIT    |
|----------------------|----------------------------------------------------------------------|-----|-----|-----|---------|
| <b>Supply</b>        |                                                                      |     |     |     |         |
| $t_{PWRUP}$          | Time after VDD exceeds POR for device to power up                    |     |     | 2   | ms      |
| <b>Mode Change</b>   |                                                                      |     |     |     |         |
| $t_{MODE\_NOM\_SLP}$ | Time from Go-to-sleep command where device turns off CAN transceiver |     | 100 | 600 | $\mu s$ |

## 5.7 Timing Requirements (continued)

|                                |                                                                                                             | MIN  | TYP | MAX  | UNIT |
|--------------------------------|-------------------------------------------------------------------------------------------------------------|------|-----|------|------|
| <b>Device Timing</b>           |                                                                                                             |      |     |      |      |
| t <sub>MODE_POR_NOM</sub>      | Time from POR to normal mode, ready to receive CAN messages                                                 |      | 150 | 400  | μs   |
| <b>SPI Timing Requirements</b> |                                                                                                             |      |     |      |      |
| t <sub>HD-DAT</sub>            | POCI input hold time requirement after active SCK edge<br>See <a href="#">SPI Timing Characteristics</a>    | 20   |     |      | ns   |
| t <sub>SU-DAT</sub>            | POCI input set up time requirement before active SCK edge<br>See <a href="#">SPI Timing Characteristics</a> | 5    |     |      | ns   |
| <b>UART</b>                    |                                                                                                             |      |     |      |      |
| t <sub>W(RX)</sub>             | Pulse width, receive start, stop, data bit <sup>(1)</sup>                                                   | 0.97 | 1   | 1.04 | U    |

(1) U = UART baud time = 1/programmed baud rate

## 5.8 Switching Characteristics

over operating free-air temperature range (unless otherwise noted)

| PARAMETER                     |                                                                                              | TEST CONDITIONS                                                                 | MIN  | TYP  | MAX  | UNIT             |
|-------------------------------|----------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------|------|------|------|------------------|
| SPI Switching Characteristics |                                                                                              |                                                                                 |      |      |      |                  |
| f <sub>SCK-MAX</sub>          | SCK, Maximum SPI clock frequency <sup>(1)</sup>                                              | Normal mode; C <sub>L</sub> = 20 pF                                             |      | 20   |      | MHz              |
| t <sub>SCK-MIN</sub>          | SCK, Minimum SPI clock period <sup>(1)</sup>                                                 | Normal mode; C <sub>L</sub> = 20 pF                                             |      | 50   |      | ns               |
| t <sub>SCKR</sub>             | SCK rise time <sup>(1)</sup>                                                                 | Normal mode; C <sub>L</sub> = 20 pF                                             |      | 4    | 10   | ns               |
| t <sub>SCKF</sub>             | SCK fall time <sup>(1)</sup>                                                                 | Normal mode; C <sub>L</sub> = 20 pF                                             |      | 4    | 8    | ns               |
| t <sub>SCKH</sub>             | SCK, SPI clock high <sup>(1)</sup>                                                           | Normal mode                                                                     | 0.47 | 0.49 | 0.5  | S <sup>(3)</sup> |
| t <sub>SCKL</sub>             | SCK, SPI clock low <sup>(1)</sup>                                                            | Normal mode                                                                     | 0.5  | 0.51 | 0.53 | S <sup>(3)</sup> |
| t <sub>SCK-CS</sub>           | Delay time from last SCK edge to CS inactive <sup>(1)</sup>                                  | Normal mode<br>See <a href="#">SPI Timing Characteristics</a>                   | 0.44 | 0.5  | 0.52 | S <sup>(3)</sup> |
| t <sub>CS-SCK</sub>           | Delay time from CS active to first SCK edge <sup>(1)</sup>                                   | Normal mode<br>See <a href="#">SPI Timing Characteristics</a>                   | 0.49 | 0.5  | 0.57 | S <sup>(3)</sup> |
| t <sub>HD-DAT</sub>           | Data hold time from capture SCK edge to PICO data changing <sup>(1)</sup>                    | Normal mode<br>See <a href="#">SPI Timing Characteristics</a>                   | 0.47 | 0.5  | 0.51 | S <sup>(3)</sup> |
| t <sub>CS-DV</sub>            | Delay time, chip select active edge to PICO data valid <sup>(1)</sup>                        | Normal mode<br>SPI Modes 0, 2<br>See <a href="#">SPI Timing Characteristics</a> |      | 0    | 5    | ns               |
|                               |                                                                                              | Normal mode<br>SPI Modes 1, 3<br>See <a href="#">SPI Timing Characteristics</a> |      | 0.5  | 0.57 | S <sup>(3)</sup> |
| I2C Switching Characteristics |                                                                                              |                                                                                 |      |      |      |                  |
| t <sub>STUCKBUS_I2C</sub>     | The time for a bus line to be "stuck" low until the device considers the bus stuck.          |                                                                                 | 25   | 40   | 65   | ms               |
| f <sub>SB_SCLOUT_I2C</sub>    | I2C stuck bus recovery pattern clock frequency generated while attempting to unstick the bus |                                                                                 | 5.5  | 8.5  | 14   | kHz              |
| UART                          |                                                                                              |                                                                                 |      |      |      |                  |
| t <sub>baud</sub>             | Maximum UART baud rate                                                                       |                                                                                 |      |      | 2.5  | MHz              |
| t <sub>UARTTXD</sub>          | Time from Complete of CAN command to start of UART transmit                                  |                                                                                 |      | 1    | 2    | U                |
| t <sub>UARTRXD</sub>          | Time from UART reception to CAN read ready <sup>(1)</sup>                                    | End of last data/parity bit to UART.RXN bit set<br>C <sub>L</sub> = 20 pF       |      | 0.5  | 2    | U                |

## 5.8 Switching Characteristics (continued)

over operating free-air temperature range (unless otherwise noted)

| PARAMETER   |                                                                           | TEST CONDITIONS       | MIN  | TYP | MAX  | UNIT |
|-------------|---------------------------------------------------------------------------|-----------------------|------|-----|------|------|
| $t_{W(TX)}$ | Pulse width, transmit start, stop, data bit <sup>(1)</sup> <sup>(2)</sup> | $C_L = 20 \text{ pF}$ | 0.98 |     | 1.02 | U    |

- (1) Specified by design
- (2) U = UART baud time = 1/programmed baud rate
- (3) S = SPI clock period = 1/SPI clock frequency

## 5.9 I<sup>2</sup>C Bus Timing Requirements

over operating free-air temperature range (unless otherwise noted)

|                                       |                                                          |                                          | MIN                            | MAX  | UNIT |
|---------------------------------------|----------------------------------------------------------|------------------------------------------|--------------------------------|------|------|
| I <sup>2</sup> C Bus - Standard Mode  |                                                          |                                          |                                |      |      |
| f <sub>scl</sub>                      | I <sup>2</sup> C clock frequency                         |                                          | 0                              | 100  | kHz  |
| t <sub>sch</sub>                      | I <sup>2</sup> C clock high time <sup>(1)</sup>          |                                          | 4                              |      | μs   |
| t <sub>scl</sub>                      | I <sup>2</sup> C clock low time <sup>(1)</sup>           |                                          | 4.7                            |      | μs   |
| t <sub>sp</sub>                       | I <sup>2</sup> C spike time                              |                                          |                                | 50   | ns   |
| t <sub>sds</sub>                      | I <sup>2</sup> C serial-data setup time                  |                                          | 250                            |      | ns   |
| t <sub>sdh</sub>                      | I <sup>2</sup> C serial-data hold time                   |                                          | 0                              |      | ns   |
| t <sub>icr</sub>                      | I <sup>2</sup> C input rise time <sup>(1)</sup>          |                                          |                                | 1000 | ns   |
| t <sub>icf</sub>                      | I <sup>2</sup> C input fall time                         |                                          |                                | 300  | ns   |
| t <sub>ocf</sub>                      | I <sup>2</sup> C output fall time                        | 10-pF to 400-pF bus                      |                                | 300  | ns   |
| t <sub>buf</sub>                      | I <sup>2</sup> C bus free time between stop and start    |                                          | 4.7                            |      | μs   |
| t <sub>sts</sub>                      | I <sup>2</sup> C start or repeated start condition setup |                                          | 4.7                            |      | μs   |
| t <sub>sth</sub>                      | I <sup>2</sup> C start or repeated start condition hold  |                                          | 4                              |      | μs   |
| t <sub>sps</sub>                      | I <sup>2</sup> C stop condition setup                    |                                          | 4                              |      | μs   |
| t <sub>vd(data)</sub>                 | Valid data time                                          | SCL low to SDA output valid              |                                | 3.45 | μs   |
| t <sub>vd(ack)</sub>                  | Valid data time of ACK condition                         | ACK signal from SCL low to SDA (out) low |                                | 3.45 | μs   |
| C <sub>b</sub>                        | I <sup>2</sup> C bus capacitive load                     |                                          |                                | 400  | pF   |
| I <sup>2</sup> C Bus - Fast Mode      |                                                          |                                          |                                |      |      |
| f <sub>scl</sub>                      | I <sup>2</sup> C clock frequency                         |                                          | 0                              | 400  | kHz  |
| t <sub>sch</sub>                      | I <sup>2</sup> C clock high time <sup>(1)</sup>          |                                          | 0.6                            |      | μs   |
| t <sub>scl</sub>                      | I <sup>2</sup> C clock low time <sup>(1)</sup>           |                                          | 1.3                            |      | μs   |
| t <sub>sp</sub>                       | I <sup>2</sup> C spike time                              |                                          |                                | 50   | ns   |
| t <sub>sds</sub>                      | I <sup>2</sup> C serial-data setup time                  |                                          | 100                            |      | ns   |
| t <sub>sdh</sub>                      | I <sup>2</sup> C serial-data hold time                   |                                          | 0                              |      | ns   |
| t <sub>icr</sub>                      | I <sup>2</sup> C input rise time <sup>(1)</sup>          |                                          | 20                             | 300  | ns   |
| t <sub>icf</sub>                      | I <sup>2</sup> C input fall time                         |                                          | 20 × (V <sub>CC</sub> / 5.5 V) | 300  | ns   |
| t <sub>ocf</sub>                      | I <sup>2</sup> C output fall time                        | 10-pF to 400-pF bus                      | 20 × (V <sub>CC</sub> / 5.5 V) | 300  | ns   |
| t <sub>buf</sub>                      | I <sup>2</sup> C bus free time between stop and start    |                                          | 1.3                            |      | μs   |
| t <sub>sts</sub>                      | I <sup>2</sup> C start or repeated start condition setup |                                          | 0.6                            |      | μs   |
| t <sub>sth</sub>                      | I <sup>2</sup> C start or repeated start condition hold  |                                          | 0.6                            |      | μs   |
| t <sub>sps</sub>                      | I <sup>2</sup> C stop condition setup                    |                                          | 0.6                            |      | μs   |
| t <sub>vd(data)</sub>                 | Valid data time                                          | SCL low to SDA output valid              |                                | 0.9  | μs   |
| t <sub>vd(ack)</sub>                  | Valid data time of ACK condition                         | ACK signal from SCL low to SDA (out) low |                                | 0.9  | μs   |
| C <sub>b</sub>                        | I <sup>2</sup> C bus capacitive load                     |                                          |                                | 400  | pF   |
| I <sup>2</sup> C Bus - Fast Mode Plus |                                                          |                                          |                                |      |      |
| f <sub>scl</sub>                      | I <sup>2</sup> C clock frequency                         |                                          | 0                              | 1000 | kHz  |
| t <sub>sch</sub>                      | I <sup>2</sup> C clock high time <sup>(1)</sup>          |                                          | 0.26                           |      | μs   |
| t <sub>scl</sub>                      | I <sup>2</sup> C clock low time <sup>(1)</sup>           |                                          | 0.5                            |      | μs   |
| t <sub>sp</sub>                       | I <sup>2</sup> C spike time                              |                                          |                                | 50   | ns   |
| t <sub>sds</sub>                      | I <sup>2</sup> C serial-data setup time                  |                                          | 50                             |      | ns   |
| t <sub>sdh</sub>                      | I <sup>2</sup> C serial-data hold time                   |                                          | 0                              |      | ns   |

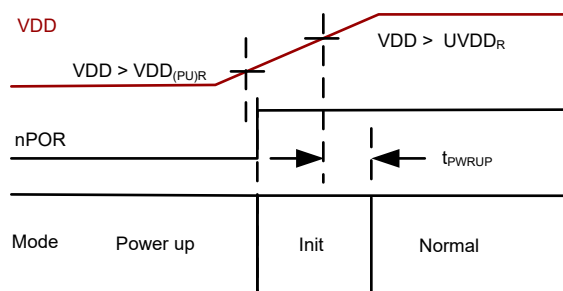
## 5.9 I<sup>2</sup>C Bus Timing Requirements (continued)

over operating free-air temperature range (unless otherwise noted)

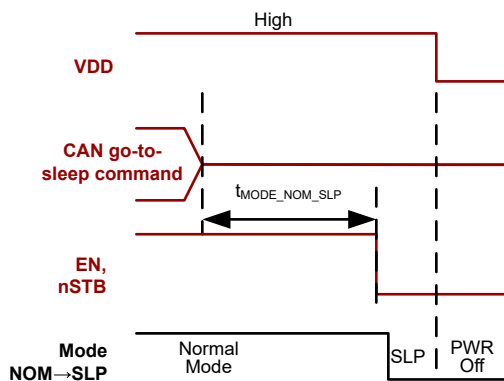
|                       |                                                          |                                          | MIN                            | MAX  | UNIT |
|-----------------------|----------------------------------------------------------|------------------------------------------|--------------------------------|------|------|
| t <sub>icr</sub>      | I <sup>2</sup> C input rise time <sup>(1)</sup>          |                                          |                                | 120  | ns   |
| t <sub>icf</sub>      | I <sup>2</sup> C input fall time                         |                                          | 20 × (V <sub>CC</sub> / 5.5 V) | 120  | ns   |
| t <sub>ocf</sub>      | I <sup>2</sup> C output fall time                        | 10-pF to 550-pF bus                      | 20 × (V <sub>CC</sub> / 5.5 V) | 120  | ns   |
| t <sub>buf</sub>      | I <sup>2</sup> C bus free time between stop and start    |                                          | 0.5                            |      | μs   |
| t <sub>sts</sub>      | I <sup>2</sup> C start or repeated start condition setup |                                          | 0.26                           |      | μs   |
| t <sub>sth</sub>      | I <sup>2</sup> C start or repeated start condition hold  |                                          | 0.26                           |      | μs   |
| t <sub>sps</sub>      | I <sup>2</sup> C stop condition setup                    |                                          | 0.26                           |      | μs   |
| t <sub>vd(data)</sub> | Valid data time                                          | SCL low to SDA output valid              |                                | 0.45 | μs   |
| t <sub>vd(ack)</sub>  | Valid data time of ACK condition                         | ACK signal from SCL low to SDA (out) low |                                | 0.45 | μs   |
| C <sub>b</sub>        | I <sup>2</sup> C bus capacitive load                     |                                          |                                | 550  | pF   |

(1) This parameter is up to the system designer to ensure is met. It will vary depending on bus load and pull up resistance.

## 6 Parameter Measurement Information



**Figure 6-1. Power Up Timing**



**Figure 6-2. Normal Mode to Sleep Mode Timing**

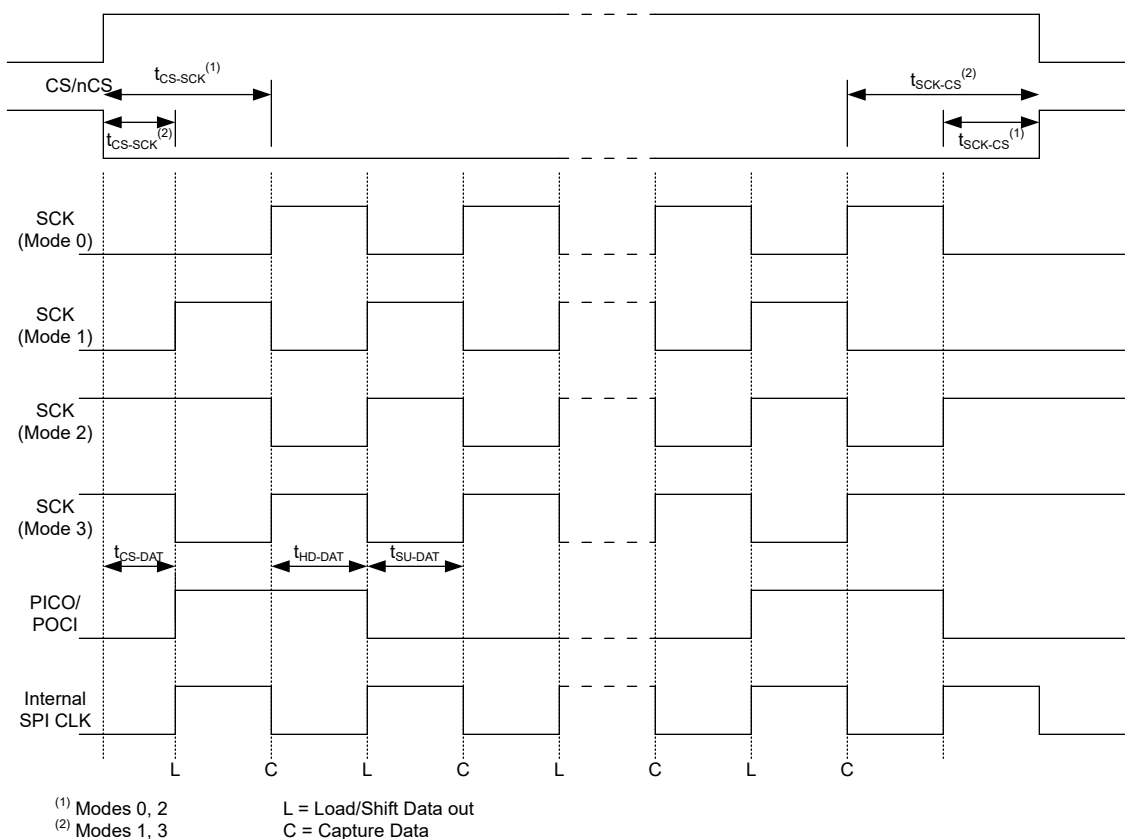


Figure 6-3. SPI Timing Characteristics

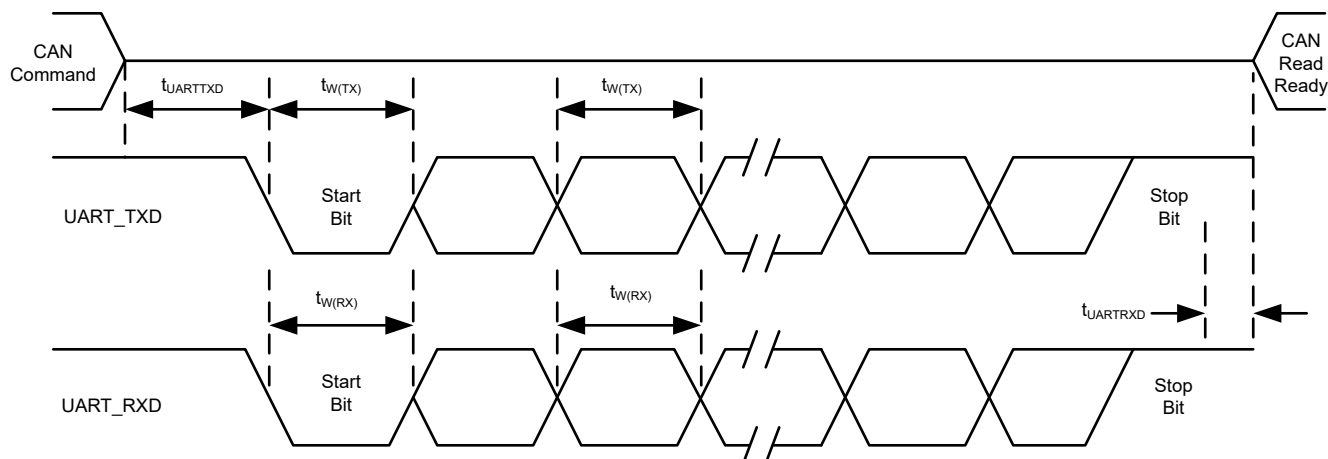
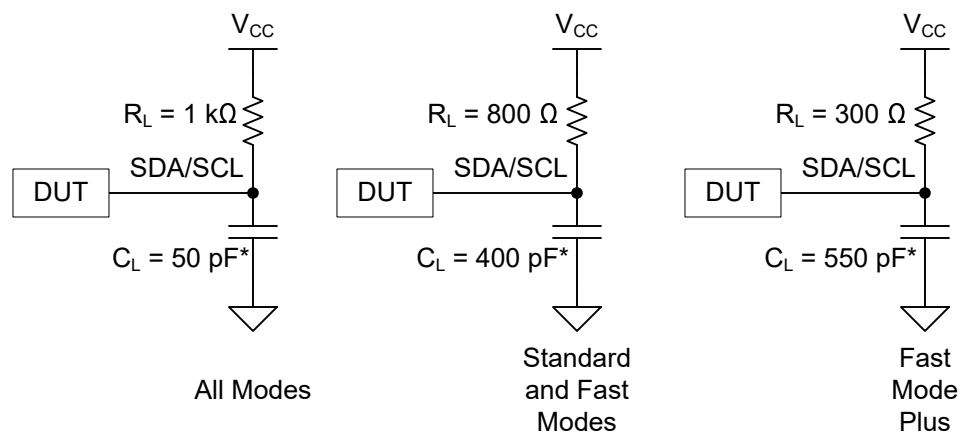


Figure 6-4. UART Timing Characteristics



\*C<sub>L</sub> includes probe and jig capacitance

**Figure 6-5. I2C Bus Loading**

## 7 Detailed Description

### 7.1 Overview

The TCAN5102-Q1 is a Control Area Network (CAN) Flexible Data (FD) Light responder device compatible with CiA 604-1 and ISO 11898-1:2024, capable of supporting up to 5Mbps data rate when a dedicated CAN FD light commander is used. The device is designed to support CAN FD light responder node applications in a commander - responder architecture which does not require a responder node processor. All control to the responder node is through the CAN bus from the commander node processor which eliminates the need for responder node processor and software.

The device receives data and/or commands from a CAN FD light commander node which translates these to either Serial Peripheral Interface (SPI) controller, UART controller, I<sup>2</sup>C controller, and/or GPIO pins communication to the device and/or peripheral devices the TCAN5102-Q1 is controlling. PWM output channels also support trapezoidal ramp profiles in hardware for controlling stepper motors. Ramping of duty cycle or frequency is possible. No external crystal or clock is required. The device controls the external TCAN1162x-Q1, TCAN1043A-Q1 or TCAN1463A-Q1 CAN FD (SIC) transceivers for system level flexibility. The device relies upon the CAN FD transceiver/SBC to control the node power and communicate a wake up signal to the TCAN5102-Q1 by latching the CAN RXD (CRXD) pin low.

### 7.2 Functional Block Diagram

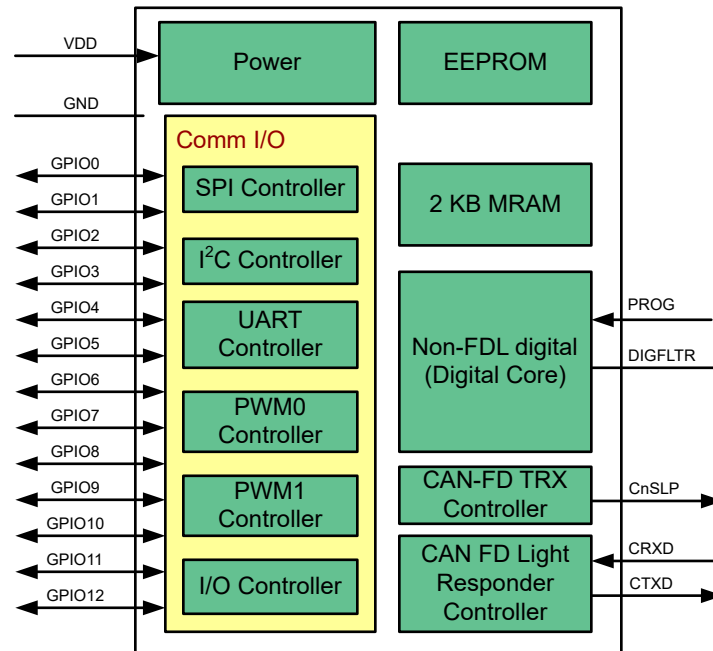
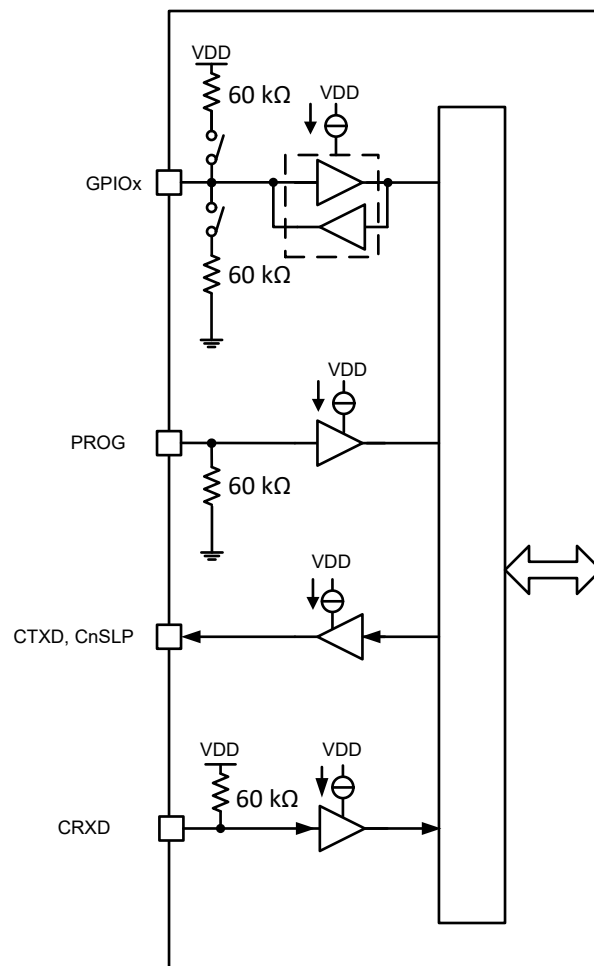


Figure 7-1. High-level block diagram



**Figure 7-2. Input and Output Block Diagram**

## 7.3 Feature Description

### 7.3.1 VDD

VDD is a power input for the device and sets the digital input/output voltage levels 3.3V, and 5V.

### 7.3.2 PROG Pin

The PROG pin is used during the ECU development phase. When setting this pin to GND at power up, the TCAN5102-Q1 can be programmed via end of line programming using the CAN bus. The CAN bus programming data rate is set at 1 Mbps for this process.

### 7.3.3 DIGFLTR Pin

This pin is used to provide filtering for the internal digital core regulator. Please refer to the Recommended Operating Conditions table to see the suggested filter capacitor value.

### 7.3.4 GPIOx and Pin Muxing Special Functions

The TCAN5102-Q1 has 12 GPIO pins which can be configured for multiple uses. As a general purpose I/O, the pin can be configured to be an input, a push-pull output, or an open-drain output. The pins also offer integrated user-enabled pull-up and pull-down resistors.

To support the different serial interfaces and IP blocks within the device, several pins have the option to be muxed with GPIOs. The extra functionality is referred to as "special function" in the register configuration, meaning that the pin no longer functions as a GPIO, but instead connects to a special IP block. When configured for special function, only the pull-up and pull down resistor configuration is used for the pin. When special function is selected, the output/input, push pull/open drain, etc mode is automatically configured for you. For example, if the GPIO corresponding to SPI POCI is selected to be a special function, then the settings for that GPIO are overwritten by the SPI IP to configure that GPIO to become an input. If desired, an internal pull up or pull down resistor can be enabled on that pin to provide a known pin state.

#### Note

Some pins have 2 special-functions assigned to them. In the event of an enable contention, the higher priority IP gets priority. For example, if the SPI IP block has enabled SPI chip select channel 4 (CS4) and the I<sup>2</sup>C IP has also been enabled, then GPIO9 is muxed to the I<sup>2</sup>C SCL pin. The default unprogrammed behavior of all pins is to be an input and high impedance.

The default power-on state for the GPIOs can be stored in EEPROM to have a power on response. The pins are in a high-impedance state while unpowered or below the POR threshold. Once the device powers up, the configuration is copied from EEPROM to the GPIO registers. Only the configuration registers which can affect the output behavior are are stored in EEPROM. The stored registers are

1. IO\_OE\_0 and IO\_OE\_1
2. IO\_RE\_0 and IO\_RE\_1
3. IO\_PU\_0 and IO\_PU\_1
4. IO\_OUTPUT\_0 and IO\_OUTPUT\_1

**Table 7-1. Pin Muxing for Special Functions**

| PIN | NAME   | LOW-PRIORITY | HIGH-PRIORITY        |
|-----|--------|--------------|----------------------|
| 4   | GPIO9  | SPI CS4      | I <sup>2</sup> C SCL |
| 5   | GPIO10 | SPI CS5      | I <sup>2</sup> C SDA |
| 6   | GPIO11 | SPI CS6      | PWM0                 |
| 7   | GPIO12 | SPI CS7      | PWM1                 |
| 11  | GPIO8  | UART RXD     | -                    |
| 12  | GPIO7  | UART TXD     | -                    |
| 13  | GPIO6  | SPI PICO     | -                    |
| 14  | GPIO5  | SPI POCI     | -                    |

**Table 7-1. Pin Muxing for Special Functions (continued)**

| PIN | NAME  | LOW-PRIORITY | HIGH-PRIORITY |
|-----|-------|--------------|---------------|
| 15  | GPIO4 | SPI SCK      | -             |
| 16  | GPIO3 | SPI CS0      | -             |
| 17  | GPIO2 | SPI CS1      | -             |
| 18  | GPIO1 | SPI CS2      | -             |
| 19  | GPIO0 | SPI CS3      | -             |

#### 7.3.4.1 GPIO Synchronization

The GPIOs configured as outputs support updates being done at the same time. There are 2 modes for this GPIO output updates to take place

1. Default: GPIOs are updated immediately when the IO\_OUTPUT\_x registers are updated (each byte)
2. Synchronized: Updates only occur after a write to IO\_OUTPUT\_1. To synchronize all, write to IO\_OUTPUT\_0 first, then once the write to IO\_OUTPUT\_1 is complete, all bits are updated at the same time

The feature is enabled in the [MRAM\\_IP\\_CFG](#) register by setting the GPIO\_OUT\_SYNC bit to 1.

#### 7.3.5 EEPROM

The TCAN5102-Q1 has an integrated EEPROM which is used to store pieces of information for power up configuration:

1. CAN bus data rate
2. CAN ID
3. Broadcast CAN ID and mask
4. Configuration locks (data rate and CAN IDs)
5. GPIO configuration required to set the power on reset pin state
  - a. IO\_OE\_0 and IO\_OE\_1
  - b. IO\_RE\_0 and IO\_RE\_1
  - c. IO\_PU\_0 and IO\_PU\_1
  - d. IO\_OUTPUT\_0 and IO\_OUTPUT\_1

#### Note

The EEPROM has a CRC field, and when a fault is detected with the CRC, the device disables the CAN interface and will hold the GPIOs in high impedance

#### 7.3.6 SPI Controller

The TCAN5102-Q1 features a SPI controller featuring 8 chip selects and different SPI settings for SPI channels 0 to 3. The last 4 channels share configuration with SPI channel 3. Each of the 4 first channels can have independent SPI speed, and SPI modes. Using the different chip selects allows communication to occur with a shared SPI POCI, PICO, and SCK lines. The chip selects can be configured for active high or active low. Messages are stored in the message RAM (MRAM), and the IP must be allocated space in the MRAM to function. See [Section 7.5.3](#) for more information.

See [Section 7.5.4](#) for more information about the configuration and usage of the SPI controller.

#### 7.3.7 UART Controller

The TCAN5102-Q1 features a UART controller. UART is a standard byte-oriented protocol that is used for many different interfaces (LIN, RS232, RS485, Flexwire, and so on). This controller has a fractional baud rate divider to support a wide range of baud rates for use with many interfaces.

Messages are stored in the message RAM (MRAM), and the IP must be allocated space in the MRAM to function. See [Section 7.5.3](#) for more information. It is possible to run the UART IP in single-byte mode, without using the MRAM. There is an enable bit for this mode in the UART configuration register.

See [Section 7.5.5](#) for more information about the configuration and usage of the UART controller.

### 7.3.8 I2C Controller

The TCAN5102-Q1 features an I<sup>2</sup>C controller supporting standard, fast, and fast mode plus. In addition to the standard controller, there is also a built-in stuck-bus recovery feature, which can detect when the SDA line is stuck low, and toggles the SCL line attempting to unstuck the bus. In addition, clock stretching is also supported, as long as the clock stretch time is less than the  $t_{\text{STUCKBUS\_I2C}}$  time.

Messages are stored in the message RAM (MRAM), and the IP must be allocated space in the MRAM to function. See [Section 7.5.3](#) for more information.

See [Section 7.5.6](#) for more information about the configuration and usage of the I<sup>2</sup>C controller.

#### 7.3.8.1 I2C Stuck Bus Recovery

The I2C controller has an optional feature called stuck bus recovery (SBR). Stuck bus recovery attempts to fix the bus when the SDA or SCL line is "stuck" low. This can occur if there is noise on the bus that causes an I2C peripheral to see the incorrect number of clocks. The stuck bus recovery toggles the SCL line up to 16 times to attempt to get offending device to release the bus. See [Section 7.5.6.2](#) for more information.

### 7.3.9 PWM Controller

The TCAN5102-Q1 features a PWM controller with integrated ramping features. The PWM controller supports static/typical PWM output (fixed duty cycle and frequency), but also supports advanced PWM features such as the ability to ramp either the duty cycle or the frequency in order to support basic motor control. The controller was developed with stepper motor control in mind, allowing the user to program an acceleration and deceleration profile. Once the controller is told to start the ramp, it does the math and performs the specified ramp profile on the PWM output.

The PWM controller even supports additional features such as PWM pulse counting, allowing the user to configure the PWM output to generate an acceleration and deceleration profile, and then stop after a certain amount of pulses/steps automatically. The sequence is programmed into the registers, and requires no intervention from the upstream controller other than being told to start the ramp.

For additional features, a GPIO input can be used to either start a deceleration ramp or to immediately turn off the PWM output. This is useful for a limit switch or some motor fault detection.

See [Section 7.5.7](#) for more information about the configuration and usage of the PWM controller.

### 7.3.10 CAN Transceiver Control Pins

The TCAN5102-Q1 does not have an integrated CAN transceiver, and requires an external one. This allows the user to select whichever CAN transceiver they want to support. This device has a CAN TXD, RXD and nSLP pin. The nSLP pin is used to control the sleep or standby mode of an external transceiver when going into sleep mode.

### 7.3.11 Under-Voltage Lockout and Unpowered Device

The TCAN5102-Q1 monitors the supply rail of the device (VDD). VDD is monitored for under-voltage. When a UVDD fault is detected, the device has a corresponding state change.

**Table 7-2. VDD Faults and Device Mode**

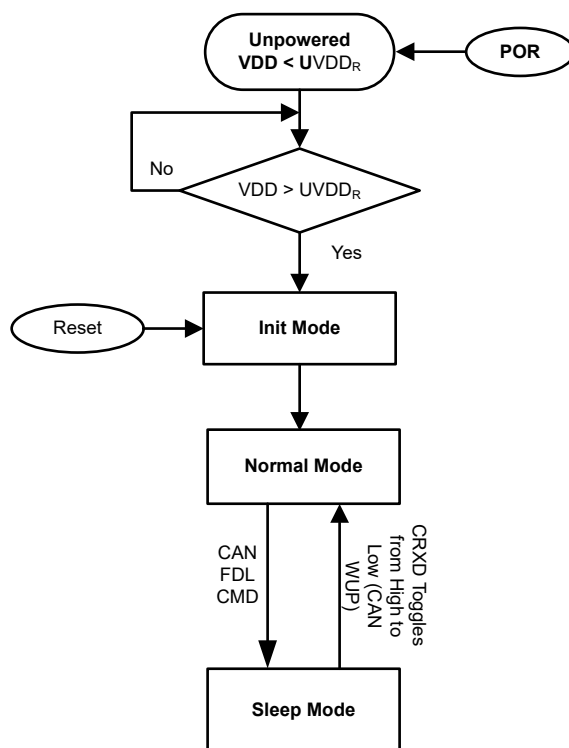
| VDD    | Device Mode    |
|--------|----------------|
| > UVDD | Normal         |
| < UVDD | Power on reset |

## 7.4 Device Functional Modes

The TCAN5102-Q1 has four modes of operation, unpowered, Init, Sleep, and Normal. See [Table 7-3](#) for how the different device blocks behave in each mode. See [Figure 7-3](#) for how the overall modes of operation interact.

**Table 7-3. Mode Overview**

| Block                   | Init       | Normal           | Sleep         |
|-------------------------|------------|------------------|---------------|
| CAN Transceiver Control | Off (Hi-Z) | On               | Off           |
| CAN nSLP                | Off (Hi-Z) | High (VDD)       | Low (GND)     |
| SPI                     | Off (Hi-Z) | On if programmed | Off           |
| I <sup>2</sup> C        | Off (Hi-Z) | On if programmed | Off           |
| UART                    | Off (Hi-Z) | On if programmed | Off           |
| GPIOx                   | Off (Hi-Z) | As programmed    | As programmed |



**Figure 7-3. TCAN5102-Q1 Modes of operation**

### 7.4.1 Init Mode

This is the initial mode of operation upon powering up. This is a transitional mode that is entered once VDD is greater than UV<sub>DD</sub> and the saved device configuration is loaded. Once the configuration is loaded, the device transitions to normal mode.

In init mode, the EEPROM is read, and once EEPROM is read, the configuration is copied to the registers and the device transitions to normal mode. See [Section 7.3.5](#) for more information about which configurations are saved.

### 7.4.2 Sleep Mode

Sleep mode is the lowest power state of the device. The TCAN5102-Q1 receives the go-to-sleep command from the CAN bus, and perform the following:

- Perform any internal house keeping actions
- Place the external CAN transceiver into sleep mode

Once the CAN transceiver is placed into sleep mode, no power is expected on VDD. The device is in the unpowered state. Keeping VDD powered while in sleep mode is possible. The device remains in a low power state until a WUP signal is received from the CAN transceiver to wake up the device.

### 7.4.3 Normal Mode

After leaving INIT mode, the device enters normal mode. Normal mode activates the transceiver and the device is ready to receive or transmit data.

## 7.5 Programming

The device uses a CAN FD Light interface for configuration. Configuration is done by writing to registers on the device. There is a specific CAN FD message structure used to access the device registers. There are multiple frame formats that this device supports for accessing device registers.

### 7.5.1 Device Programming via SPI Peripheral Mode

There are 2 options for configuration programming:

1. Program via CAN FD Light (using PROG pin to force baud rate to 1Mbps)
2. Using SPI (using a toggle sequence with the PROG and RXD pins to enable a local SPI peripheral to allow a programmer to write to the EEPROM)

The SPI peripheral mode method can be used if the CAN interface is not available to program EEPROM. This method requires a specific sequence of events in order to enable the local SPI peripheral. When the device is in SPI peripheral mode, the CAN interface is disabled and stays disabled until a power cycle.

### 7.5.2 CAN FD Light Protocol

The TCAN5102-Q1 supports the CiA 604-1 and CiA 604-3 communication standard through use of logical link control (LLC) and the FD Base Frame Format (FBFF),, see [Table 7-4](#).

**Table 7-4. Protocol CiA Format**

| Field                                                                                                                                                               | Size         | Function                                                    |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|-------------------------------------------------------------|
| Identifier                                                                                                                                                          | 11 bit       | Base identifier (either a target or broadcast ID)           |
| DLC                                                                                                                                                                 | 4 bit        | Data length code, as specified in ISO 11898-1:2015, Table 5 |
| LLC Data                                                                                                                                                            | 0 to 64 byte | Data content of the data frames                             |
| Since all transmitted and received frames are in FBFF and the bits ESI=0 and BRS=0, all bits in the format field are fixed. Therefore, the format field is omitted. |              |                                                             |

The device protocol is split into a request and a response. A device request is a request to this device to perform a read or a write to a specific register. A request always has at least 3 fields, which are required to decode any operation. These fields are always at the start of the CAN frame data section.

1. Operation Code (Op Code): This tells the device if a read or write to a register or FIFO is occurring
2. Length: Indicates the total size (in bytes) of data being transferred (after the device header)
3. Register Address: A 16-bit address of the register or FIFO being written to or read from

A device response is sent after a request to let the requesting device know that the request was received and is in progress or invalid. The response contains at least 3 fields

1. Operation Code (Op Code): Repeats back the received Op code

2. Length: Indicates the total size (in bytes) of data being returned after the header (if the request was a write, then this mirrors the number of bytes written, but no data is sent. The length field is used to confirm the number of bytes that were written)
3. Status Byte: Signals the status of the request. This can be OK/successful, or an error of some sort such as an invalid address or length.

There are 2 types of identifiers used by the device. A target identifier is the specific address that is unique to an individual responder. This allows a commander to address/configure an individual node. A broadcast identifier is also supported, which uses the broadcast ID and mask fields to match 1 or several IDs. Any requests from a broadcast ID has the request performed, but there is no response from the responder, since arbitration is not supported in CAN FD Light. Broadcast is useful if the commander wants to configure many nodes at once, or put multiple nodes to sleep. However, this drops the response frame, so it is not confirmed by the responder.

#### 7.5.2.1 CAN Frame Format

The standard CAN frame format uses 3 bytes of the CAN payload as a header, containing 3 fields: operation code (op code), data length and address.

| CAN ID          |  | Byte 0 | Byte 1               | Byte 2     | Byte 3    |          |
|-----------------|--|--------|----------------------|------------|-----------|----------|
| Device ID[10:0] |  | CTRL   | OP[7:6]/<br>LEN[5:0] | ADDR[15:8] | ADDR[7:0] | Data ... |

**Figure 7-4. Example Standard CAN Frame Format Request**

**Table 7-5. Standard CAN Frame Format Request Header Fields**

| Byte | Bits | Field      | Description                                                                                                               |
|------|------|------------|---------------------------------------------------------------------------------------------------------------------------|
| 0    | 7:6  | OPCODE     | Operation code of the device<br>2'b00 = Write<br>2'b01 = Read<br>2'b10 = Reserved<br>2'b11 = Reserved                     |
|      | 5:0  | LEN        | Number of data bytes being transferred after this header<br>Valid values (write): 0 to 61<br>Valid values (read): 0 to 62 |
| 1    | 7:0  | ADDR[15:8] | Upper byte of the address                                                                                                 |
| 2    | 7:0  | ADDR[7:0]  | Lower byte of the address                                                                                                 |

#### Note

For multi-byte reads, if the starting address is before the RX FIFO of one of the serial communication blocks (I2C, SPI, UART), the FIFO is NOT actually read. Instead, the byte corresponding to the RX FIFO returns 0 and then read continues on. This is to prevent a burst read from getting "stuck" in an RX FIFO. If the starting address of the read starts at the RX FIFO, then all bytes read are from the RX FIFO only.

| CAN ID          |  | Byte 0 | Byte 1               | Byte 2      |          |
|-----------------|--|--------|----------------------|-------------|----------|
| Device ID[10:0] |  | CTRL   | OP[7:6]/<br>LEN[5:0] | STATUS_BYTE | Data ... |

**Figure 7-5. Example Standard CAN Frame Format Response**

**Table 7-6. Standard CAN Frame Format Response Header Fields**

| Byte | Bits | Field  | Description                                                                                                                                                                                                                                                                                                                                                                |
|------|------|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0    | 7:6  | OPCODE | Operation code of the device<br>2'b00 = Write<br>2'b01 = Read<br>2'b10 = Reserved<br>2'b11 = Reserved                                                                                                                                                                                                                                                                      |
|      | 5:0  | LEN    | Echos the number of bytes of data that were requested being sent after the header if the requested amount of valid.<br><br><b>Note</b><br>If the response is to a write request, then the length field is the length of bytes that were requested, but no data is sent after the header. This is to allow for confirmation that the response frame is to the write request |
| 1    | 7:5  | RSVD   | Reserved                                                                                                                                                                                                                                                                                                                                                                   |
| 1    | 4    | FE     | Framing error on the previous request. Request ignored. Typically caused when the DLC is < 3 bytes since there is a minimum requirement for headers                                                                                                                                                                                                                        |
| 1    | 3    | IO     | Invalid operation. Request ignored                                                                                                                                                                                                                                                                                                                                         |
| 1    | 2    | IL     | Invalid length. Request ignored                                                                                                                                                                                                                                                                                                                                            |
| 1    | 1    | IA     | Invalid address. Request ignored                                                                                                                                                                                                                                                                                                                                           |
| 1    | 0    | OK     | Ok request, request complete or in progress                                                                                                                                                                                                                                                                                                                                |

### 7.5.3 Message RAM (MRAM) and IP Enable

The TCAN5102-Q1 has a 2kB message RAM (MRAM) that is used for storing the mailboxes for SPI, I2C, and UART communication. This RAM can split the memory allocated to any combination of the 2 IP blocks based upon the end-application's needs in 25% increments. The split between RX and TX is not adjustable and is always 50/50%. Handling of the RX and TX FIFOs is done by the TCAN5102-Q1, and requires no user setup outside of choosing how much of the MRAM to allocate to each IP block.

SPI and I2C have a straight forward split for memory use, where 50% of the allocated memory is used to store the TX data and the other 50% is used to store the RX data if specified.

For UART, there are additional bytes needed to store the status of each received byte. This means that for each byte of data received on UART, 1 additional byte is needed for storing the status of the byte. The result is that only 2/3 of the allocated memory for UART is used for storing actual data bytes.

Table 7-7 shows how many bytes of data are allocated to the TX or RX buffers for each IP block. The size shown is the same value for both the TX and RX buffers, and is NOT the value shared between both. For example, if 100% of the MRAM space is allocated to the SPI, then the SPI TX size is 1024 bytes, and the SPI RX buffer is 1024 bytes. Together, the TX and RX buffer sum up to 2048 bytes, which is all of the memory.

In the example where 50% of the MRAM is allocated to SPI and UART, the SPI RX and TX sizes are 512 each (1kB total). The UART IP is only 340 bytes for each buffer. Due to UART RX requiring an additional byte to store the status for each byte received, only 2/3 of the available 1024 bytes is used to store data. Internally, the number of bytes that can be stored by the TX and RX buffers are identical. For this reason, 4 bytes of data end up unused, because the 4 bytes remaining cannot be divided evenly across 3 functions (TX, RX, and RX status).

#### Note

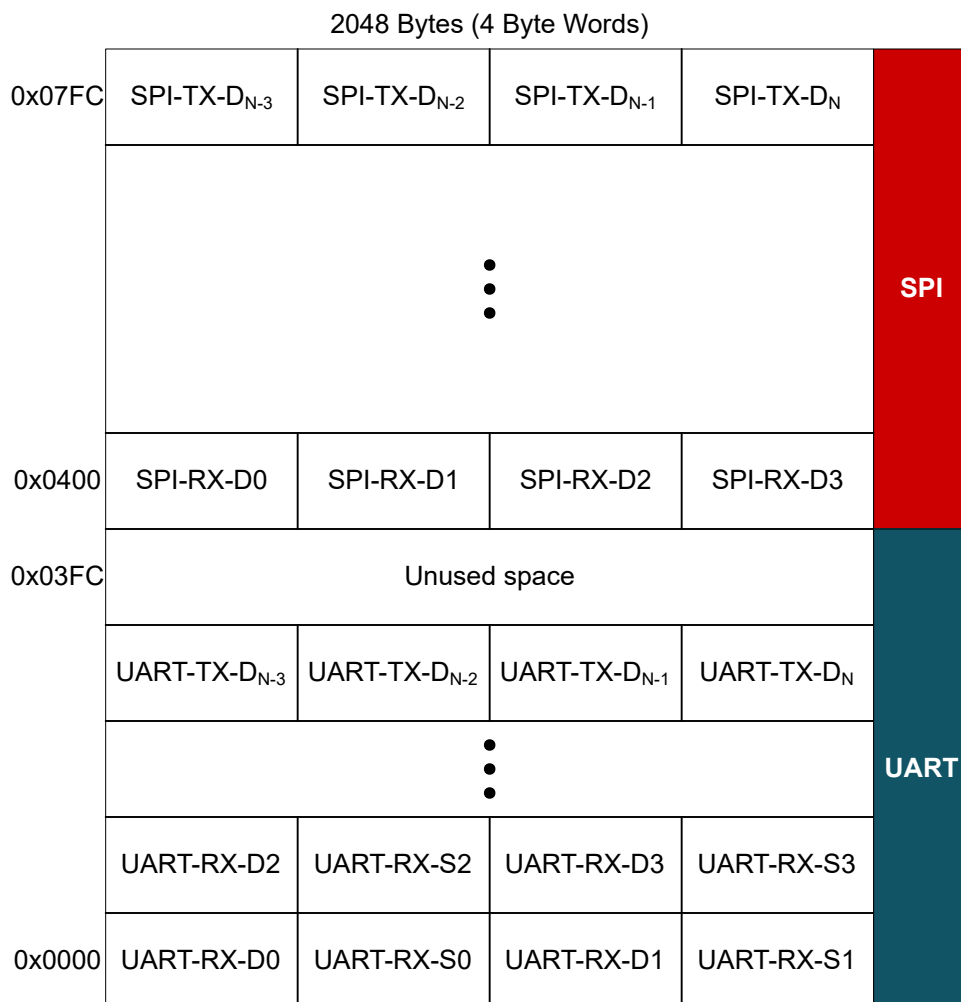
The MRAM allocation setting is used to enable the IP module for SPI and I2C. If any memory is allocated to the module, then the module is enabled, but the GPIOs in use for the module must still be manually set to special-function in the GPIO configuration registers.

**Table 7-7. MRAM Allocation**

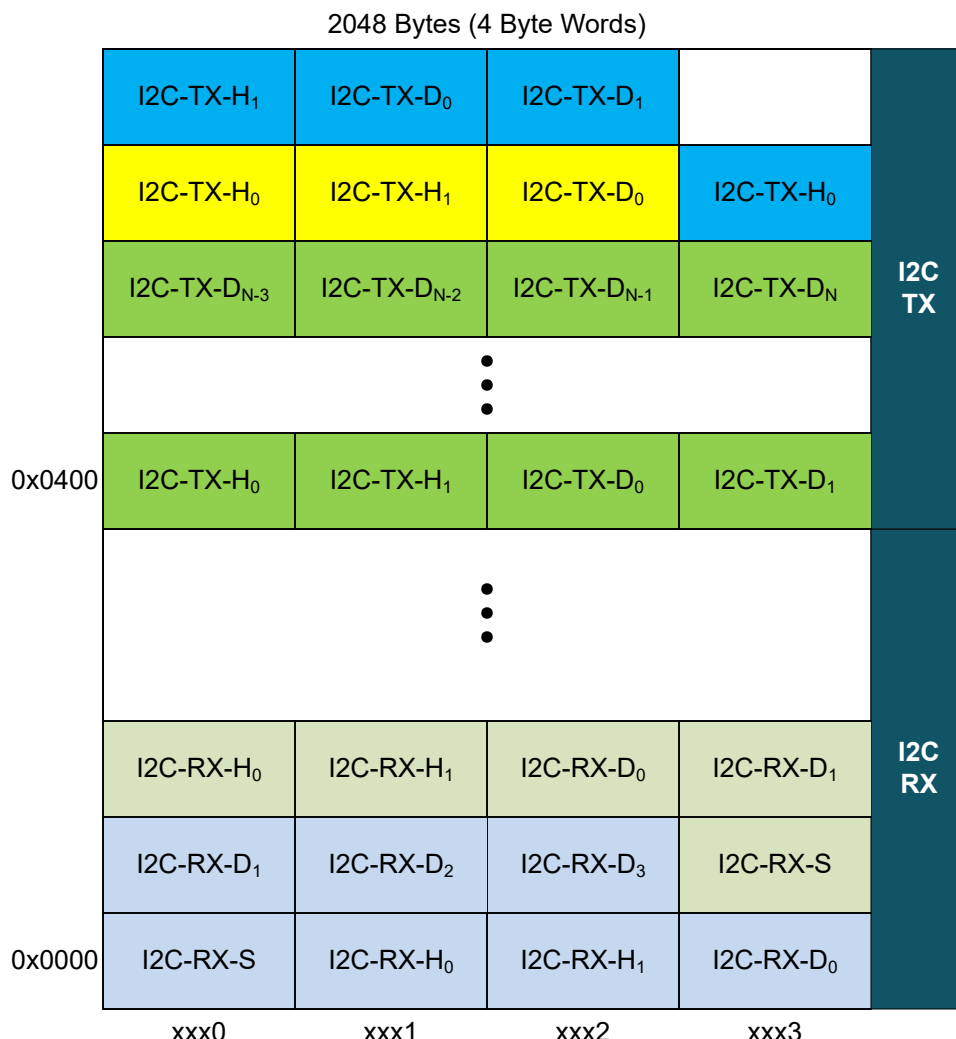
| % Allocated to SPI | SPI TX or RX Size (Bytes) | UART TX or RX Size (Bytes) | Total Bytes Used (% of Total) |
|--------------------|---------------------------|----------------------------|-------------------------------|
| 100%               | 1024                      | 0                          | 2048 (100%)                   |
| 75%                | 768                       | 168                        | 2040 (99.6%)                  |
| 50%                | 512                       | 340                        | 2044 (99.8%)                  |
| 25%                | 256                       | 512                        | 2048 (100%)                   |
| 0%                 | 0                         | 680                        | 2040 (99.6%)                  |

**Table 7-8. MRAM Allocation**

| MRAM_IP_EN (Hex) | % Allocated to SPI | % Allocated to UART | % Allocated to I2C | SPI TX and RX Size (Bytes) | UART TX and RX Size (Bytes) | I2C TX and RX Size (Bytes) | Total Bytes Used (% of Total) |
|------------------|--------------------|---------------------|--------------------|----------------------------|-----------------------------|----------------------------|-------------------------------|
| 0h               | 0%                 | 0%                  | 0%                 | 0                          | 0                           | 0                          | 0 (0%)                        |
| 1h               | 0%                 | 100%                | 0%                 | 0                          | 680                         | 0                          | 2040 (99.61%)                 |
| 2h               | 25%                | 75%                 | 0%                 | 256                        | 512                         | 0                          | 2048 (100%)                   |
| 3h               | 50%                | 50%                 | 0%                 | 512                        | 340                         | 0                          | 2044 (99.8%)                  |
| 4h               | 75%                | 25%                 | 0%                 | 768                        | 168                         | 0                          | 2040 (99.61%)                 |
| 5h               | 100%               | 0%                  | 0%                 | 1024                       | 0                           | 0                          | 2048 (100%)                   |
| 6h               | 0%                 | 0%                  | 100%               | 0                          | 0                           | 1024                       | 2048 (100%)                   |
| 7h               | 0%                 | 25%                 | 75%                | 0                          | 168                         | 768                        | 2040 (99.61%)                 |
| 8h               | 0%                 | 50%                 | 50%                | 0                          | 340                         | 512                        | 2044 (99.8%)                  |
| 9h               | 0%                 | 75%                 | 25%                | 0                          | 512                         | 256                        | 2048 (100%)                   |
| Ah               | 25%                | 0%                  | 75%                | 256                        | 0                           | 768                        | 2048 (100%)                   |
| Bh               | 50%                | 0%                  | 50%                | 512                        | 0                           | 512                        | 2048 (100%)                   |
| Ch               | 75%                | 0%                  | 25%                | 768                        | 0                           | 256                        | 2048 (100%)                   |



**Figure 7-6. Example MRAM Layout (SPI and UART enabled with a 50% split)**



**Figure 7-7. Example MRAM Layout with Example Data (100% of MRAM to I2C)**

In the above graphic, an example of how the MRAM holds data for I2C is shown. Each individual I2C frame/message is shown in a different color to show how the messages are grouped tightly in each RX or TX FIFO. The user does not need to be aware of how the data is stored in the FIFO, only the total size of the FIFO and the fact that any header/status bytes are also stored in memory.

#### 7.5.4 SPI Controller

##### Note

Before the module can be used, the appropriate GPIOs must be set to special function (see [Section 7.3.4](#) and [Section 7.6.1.18](#)) and the MRAM must enable the module by allocation memory (see [Section 7.5.3](#) and [Section 7.6.1.17](#)). Only once the GPIOs have been set to special function, and the module has been enabled by selecting a MRAM configuration that allocates memory, can the module be used. The appropriate SPI configuration registers can be seen in [Section 7.6.2](#).

##### 7.5.4.1 SPI Pins

SPI communication typically uses 4 pins for communication to another SPI device: clock (SCLK), data in (PICO), data out (POCI) and chip select (CS or nCS).

#### 7.5.4.1.1 SPI Clock (SCLK)

The SPI clock output pin is used to generate a clock that synchronizes data shifted in or out of the target peripheral. The pin can be configured for any of the 4 SPI modes (2 clock polarities) to give maximum configurability and compatibility with downstream peripherals.

#### 7.5.4.1.2 Peripheral In Controller Out (PICO)

This output pin is used to shift data into the peripheral device (downstream device). When the chip select signal is asserted (can be high or low depending on configuration, the PICO shifts data into the target/peripheral device). This pin would be connected to a serial data input (SDI) pin on a peripheral/target device.

#### 7.5.4.1.3 Peripheral Out Controller In (POCI)

This input pin is used to shift data out of a peripheral/downstream device and into this device/controller. The POCI pin is typically connected to the serial data output (SDO) pin on the peripheral/target device.

#### 7.5.4.1.4 Chip Select (CS or nCS)

The chip select output pin is used to tell a target peripheral when it is being communicated with. This device supports up to 4 separate chip select channels with individual polarity selects.

#### 7.5.4.2 SPI Clock Generator

The TCAN5102-Q1 SPI module contains a programmable clock generator that uses an 8-bit divisor. The SPI\_DR register is interpreted as 1 more than the value in the SPI\_DR register, and is used for the half-period. The formula for the divisor is:

$$\text{Divisor} = (\text{Clock Frequency} / (\text{Desired SPI clock frequency} \times 2)) - 1 \quad (1)$$

Where clock frequency is 40MHz. This formula shows that the maximum SPI frequency supported is 20MHz and the minimum frequency is 78,125Hz.

**Table 7-9. Common SPI Clocks**

| DESIRED SPI CLOCK | DIVISOR USED TO GENERATE CLOCK | SPI_DR VALUE (HEX) |
|-------------------|--------------------------------|--------------------|
| 80kHz             | 249                            | 0xF9               |
| 100kHz            | 199                            | 0xC7               |
| 200kHz            | 99                             | 0x63               |
| 250kHz            | 79                             | 0x4F               |
| 500kHz            | 39                             | 0x27               |
| 1MHz              | 19                             | 0x13               |
| 2MHz              | 9                              | 0x09               |
| 4MHz              | 4                              | 0x04               |
| 5MHz              | 3                              | 0x03               |
| 6.67MHz           | 2                              | 0x02               |
| 10MHz             | 1                              | 0x01               |
| 20MHz             | 0                              | 0x00               |

#### 7.5.4.3 SPI Control Protocol

When reading or writing to the SPI TX or RX FIFOs, there is an additional 2 byte header which provides additional robustness when reading or writing to a target peripheral. There are 2 formats, which are similar but differ depending on if the action is a read or a write to the SPI TX or RX FIFO.

When reading the RX FIFO, there is always the 2-byte header at the beginning of the data field which contains information on how many bytes are remaining and if the read is a continuation of a previous read (needed for SPI messages that are larger than can fit in a single CAN frame).

When writing to the TX FIFO, only the first write of a message contains the 2-byte header.

See [SPI Receive FIFO \(address = h1010\)](#) and [SPI Transmit FIFO \(address = h1010\)](#) for more information about the header formats.

#### 7.5.4.3.1 SPI Write Example 1

This is a basic example of sending a stream of bytes via SPI that fits into a single CAN frame. Once the device receives all bytes for the SPI frame (determined by the number of bytes given in the SPI header), the device begins transmitting the SPI frame.

#### Note

Before the module can be used, the appropriate GPIOs must be set to special function (see [Section 7.3.4](#) and [Section 7.6.1.18](#)) and the MRAM must enable the module by allocation memory (see [Section 7.5.3](#) and [Section 7.6.1.17](#)). Only once the GPIOs have been set to special function, and the module has been enabled by selecting a MRAM configuration that allocates memory, can the module be used. The appropriate SPI configuration registers can be seen in [Section 7.6.2](#).

**Table 7-10. Design Parameters and Assumptions**

| PARAMETER                     | VALUE                                                      |
|-------------------------------|------------------------------------------------------------|
| CAN Frame format              | Standard Frame Format                                      |
| Bytes to transfer             | 10'B                                                       |
| SPI channel                   | Channel 1                                                  |
| Save received data to RX FIFO | No                                                         |
| SPI byte stream               | 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88, 0x99, 0xAA |

The following table shows the steps and what data is transmitted on the CAN bus. The central gateway/controller of the bus is referred to as the ECU, and the TCAN device is referred to as "responder".

- Hexadecimal values marked as **BOLD** signify the bytes for CAN Frame header (standard frame format for this example)
- Hexadecimal values marked as *ITALIC* signify the SPI header bytes
- Hexadecimal values without any formatting are SPI data bytes

**Table 7-11. Example CAN Sequence (Single CAN Frame)**

| Step | Transmitter     | Data                                                                                                                                            | Description                                                                                                                                                                                                                                                                                                  |
|------|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | ECU (DLC = 16B) | <b>0x0C</b> , <b>0x10</b> , <b>0x10</b> , <i>0x01</i> ,<br><i>0x0A</i> , 0x11, 0x22, 0x33,<br>0x44, 0x55, 0x66, 0x77,<br>0x88, 0x99, 0xAA, 0x00 | The ECU requests a write of 10 bytes to register 0x1010 (SPI TX FIFO), with a SPI header with store = 0, SPI channel = 1, and a SPI frame of 10 bytes. The 10 bytes of data are then shifted into the device. Since the CAN DLC must be 16 bytes to fit these 15 bytes of data, a 0x00 is padded to the end. |
| 2    | Responder       | <b>0x0C</b> , <b>0x01</b>                                                                                                                       | The responder sends back an OK to acknowledge that the request was received.                                                                                                                                                                                                                                 |
| 3    | SPI             | PICO: 0x11, 0x22, 0x33, 0x44, 0x55, 0x66,<br>0x77, 0x88, 0x99, 0xAA                                                                             | At this point, the responder does the SPI transfer on SPI channel 1 of 10 bytes.                                                                                                                                                                                                                             |

**Table 7-12. Example CAN Sequence (Multiple CAN Frames to Transfer Payload)**

| Step | Transmitter    | Data                                                                                      | Description                                                                                                                                                                                                              |
|------|----------------|-------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | ECU (DLC = 8B) | <b>0x05</b> , <b>0x10</b> , <b>0x10</b> , <i>0x01</i> ,<br><i>0x0A</i> , 0x11, 0x22, 0x33 | The ECU requests a write of 5 bytes to register 0x1010 (SPI TX FIFO), with a SPI header with store = 0, SPI channel = 1, and a SPI frame of 10 bytes. Only the first 3 bytes of data for the SPI frame were transferred. |
| 2    | Responder      | <b>0x05</b> , <b>0x01</b>                                                                 | The responder sends back an OK to acknowledge that the request was received.                                                                                                                                             |

**Table 7-12. Example CAN Sequence (Multiple CAN Frames to Transfer Payload) (continued)**

| Step | Transmitter          | Data                                                                           | Description                                                                                                                                                                                                                                                                                     |
|------|----------------------|--------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 3    | ECU (DLC = 3B)       | <b>0x42, 0x10, 0x12</b>                                                        | The ECU requests a read of 2 bytes to read the TX FIFO status and TX Element Status Registers. This step is optional, but does show some useful information for partial writes.                                                                                                                 |
| 4    | Responder (DLC = 4B) | <b>0x42, 0x01</b> , 0x0C, 0x87                                                 | SPI_TXFS shows that there are 12 empty spaces (bytes) in the TX FIFO, so there is no risk of overflow. SPI_TXES shows the TXEIP (TX Element in progress) flag is set, as expected, signaling that the current SPI frame is incomplete in memory and requires 7 more bytes of data (TXEBP field) |
| 5    | ECU (DLC = 12B)      | <b>0x07, 0x10, 0x10</b> , 0x44, 0x55, 0x66, 0x77, 0x88, 0x99, 0xAA, 0x00, 0x00 | The ECU continues the write request to the same address of 7 bytes. Since the CAN DLC that can store this frame is 12 bytes, 2 bytes of padding are added to the end. The value of these padded bytes do not matter, and are ignored.                                                           |
| 6    | Responder            | <b>0x07, 0x01</b>                                                              | The responder sends back an OK to acknowledge that the request was received.                                                                                                                                                                                                                    |
| 7    | SPI                  | PICO: 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88, 0x99, 0xAA               | At this point, the responder does the SPI transfer on SPI channel 1 of 10 bytes. POCl is ignored.                                                                                                                                                                                               |

**7.5.4.3.2 SPI Read Example 1**

This is a basic example of sending and receiving a stream of bytes via SPI. Once the device receives all bytes for the SPI frame (determined by the number of bytes given in the SPI header), the device begins transmitting the SPI frame. This example stores the received POCl data for read back later.

**Note**

Before the module can be used, the appropriate GPIOs must be set to special function (see [Section 7.3.4](#) and [Section 7.6.1.18](#)) and the MRAM must enable the module by allocation memory (see [Section 7.5.3](#) and [Section 7.6.1.17](#)). Only once the GPIOs have been set to special function, and the module has been enabled by selecting a MRAM configuration that allocates memory, can the module be used. The appropriate SPI configuration registers can be seen in [Section 7.6.2](#).

**Table 7-13. Design Parameters and Assumptions**

| PARAMETER                     | VALUE                                                      |
|-------------------------------|------------------------------------------------------------|
| CAN Frame format              | Standard Frame Format                                      |
| Bytes to transfer             | 10B                                                        |
| SPI channel                   | Channel 2                                                  |
| Save received data to RX FIFO | Yes                                                        |
| SPI byte stream               | 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88, 0x99, 0xAA |

The following table shows the steps and what data is transmitted on the CAN bus. The central gateway/controller of the bus is referred to as the ECU, and the TCAN device is referred to as "responder".

- Hexadecimal values marked as **BOLD** signify the bytes for CAN Frame header (standard frame format for this example)
- Hexadecimal values marked as *ITALIC* signify the SPI header bytes
- Hexadecimal values without any formatting are SPI data bytes

**Table 7-14. Example CAN Sequence (Single CAN Frame)**

| Step | Transmitter           | Data                                                                                                                                 | Description                                                                                                                                                                                                                                                                                                                                         |
|------|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | ECU (DLC = 16B)       | <b>0x0C, 0x10, 0x10, 0x82, 0x0A, 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88, 0x99, 0xAA, 0x00</b>                                | The ECU requests a write of 10 bytes to register 0x1010 (SPI TX FIFO), with a SPI header with store = 1, SPI channel = 2, and a SPI frame of 10 bytes. The 10 bytes of data are then shifted into the device. Since the CAN DLC must be 16 bytes to fit these 15 bytes of data, a 0x00 is padded to the end.                                        |
| 2    | Responder             | <b>0x0C, 0x01</b>                                                                                                                    | The responder sends back an OK to acknowledge that the request was received.                                                                                                                                                                                                                                                                        |
| 3    | SPI                   | PICO: 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88, 0x99, 0xAA<br>POCI: 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0A | At this point, the responder does the SPI transfer on SPI channel 2 of 10 bytes. The POCI data is stored in the RX FIFO                                                                                                                                                                                                                             |
| 4    | ECU                   | <b>0x43, 0x10, 0x0F</b>                                                                                                              | The ECU requests a read of 3 bytes from registers 0x100F-0x1011 (SPI_STATUS and SPI_RXFS). Since the burst read was started NOT on 0x1010 (SPI RX FIFO), the byte corresponding to that register will be a 0 for padding. Burst reads that do not start on the FIFO, will return '0' for the byte corresponding to the FIFO.                        |
| 5    | Responder             | <b>0x43, 0x01, 0xCA, 0x00, 0x01</b>                                                                                                  | The responder returns the data from registers 0x100F, a 0 for 0x1010 and data for 0x1011. The resulting data states that the TX FIFO is empty, the RX FIFO contains 1 SPI frame and that the next RX FIFO element has 10 bytes of data. With this information, the ECU knows to request a read of 10 + 2 (SPI header) bytes from the device.        |
| 6    | ECU                   | <b>0x4C, 0x10, 0x10</b>                                                                                                              | The ECU requests a read of 12 bytes (10 bytes of data + 2 bytes of header) from the RX FIFO                                                                                                                                                                                                                                                         |
| 7    | Responder (DLC = 16B) | <b>0x4C, 0x01, 0x02, 0x0A, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0A, 0x00, 0x00</b>                                | The responder returns the requested 12 bytes of data. The SPI header bytes signify that this is the start of the frame readout, SPI channel 2 was used, and there are 10 bytes of SPI data to read (which fits into this entire CAN frame). Since the closest CAN DLC size to hold the data was 16 bytes, 2 bytes of padding were added to the end. |

**Table 7-15. Example CAN Sequence (Multiple CAN Frames to Transfer Payload)**

| Step | Transmitter           | Data                                                                                                                                 | Description                                                                                                                                                                                                                                                                                                                                  |
|------|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | ECU (DLC = 8B)        | <b>0x05, 0x10, 0x10, 0x02, 0x0A, 0x11, 0x22, 0x33</b>                                                                                | The ECU requests a write of 5 bytes to register 0x1010 (SPI TX FIFO), with a SPI header with store = 1, SPI channel = 2, and a SPI frame of 10 bytes. Only the first 3 bytes of data for the SPI frame were transferred.                                                                                                                     |
| 2    | Responder (DLC = 2 B) | <b>0x05, 0x01</b>                                                                                                                    | The responder sends back an OK to acknowledge that the request was received.                                                                                                                                                                                                                                                                 |
| 3    | ECU (DLC = 12B)       | <b>0x07, 0x10, 0x10, 0x44, 0x55, 0x66, 0x77, 0x88, 0x99, 0xAA, 0x00, 0x00</b>                                                        | The ECU continues the write request to the same address of 7 bytes. Since the CAN DLC that can store this frame is 12 bytes, 2 bytes of padding are added to the end. The value of these padded bytes do not matter, and are ignored.                                                                                                        |
| 4    | Responder (DLC = 2 B) | <b>0x07, 0x01</b>                                                                                                                    | The responder sends back an OK to acknowledge that the request was received.                                                                                                                                                                                                                                                                 |
| 3    | SPI                   | PICO: 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x77, 0x88, 0x99, 0xAA<br>POCI: 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09, 0x0A | At this point, the responder does the SPI transfer on SPI channel 2 of 10 bytes. The POCI data is stored in the RX FIFO                                                                                                                                                                                                                      |
| 4    | ECU (DLC = 3 B)       | <b>0x43, 0x10, 0x0F</b>                                                                                                              | The ECU requests a read of 3 bytes from registers 0x100F-0x1011 (SPI_STATUS and SPI_RXFS). Since the burst read was started NOT on 0x1010 (SPI RX FIFO), the byte corresponding to that register will be a 0 for padding. Burst reads that do not start on the FIFO, will return '0' for the byte corresponding to the FIFO.                 |
| 5    | Responder (DLC = 5 B) | <b>0x43, 0x01, 0xCA, 0x00, 0x01</b>                                                                                                  | The responder returns the data from registers 0x100F, a 0 for 0x1010 and data for 0x1011. The resulting data states that the TX FIFO is empty, the RX FIFO contains 1 SPI frame and that the next RX FIFO element has 10 bytes of data. With this information, the ECU knows to request a read of 10 + 2 (SPI header) bytes from the device. |
| 6    | ECU (DLC = 3 B)       | <b>0x48, 0x10, 0x10</b>                                                                                                              | The ECU requests a read of 8 bytes (which will contain 2 bytes of header + 6 bytes of data) from the RX FIFO                                                                                                                                                                                                                                 |

**Table 7-15. Example CAN Sequence (Multiple CAN Frames to Transfer Payload) (continued)**

| Step | Transmitter               | Data                                                                                | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|------|---------------------------|-------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7    | Responder<br>(DLC = 12 B) | <b>0x48, 0x01, 0x02, 0x0A,</b><br>0x01, 0x02, 0x03, 0x04,<br>0x05, 0x06, 0x00, 0x00 | The responder returns the requested 8 bytes of data (even though the DLC is 12, we know that only the 8 bytes of data after the CAN response header is valid, so we know the 0s are padding). The SPI header bytes signify that this is the start of the frame readout, SPI channel 2 was used, and there are 10 bytes of SPI data left to read. Since the closest CAN DLC size to hold the data was 12 bytes, 2 bytes of padding were added to the end. The ECU was told there are 10 bytes of SPI data to be read, but only 6 were requested. This means there are 4 bytes of SPI data left to read. |
| 8    | ECU (DLC = 3 B)           | <b>0x46, 0x10, 0x10</b>                                                             | The ECU requests a read of 6 bytes (4 bytes of data + 2 bytes of header) from the RX FIFO to finish the read.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 9    | Responder<br>(DLC = 8 B)  | <b>0x46, 0x01, 0x82, 0x04,</b><br>0x07, 0x08, 0x09, 0x0A                            | The responder returns the requested 6 bytes of data. The SPI header bytes signify that this is a continuation of a frame readout, SPI channel 2 was used, and there are 4 bytes of SPI data left to be read (which fits into this frame). Since the data to send lined up with a CAN frame size, no padding bytes were needed.                                                                                                                                                                                                                                                                         |

### 7.5.5 UART Controller

#### Note

Before the module can be used, the appropriate GPIOs must be set to special function (see [Section 7.3.4](#) and [Section 7.6.1.18](#)) and the MRAM must enable the module by allocation memory (see [Section 7.5.3](#) and [Section 7.6.1.17](#)). Only once the GPIOs have been set to special function, and the module has been enabled by selecting a MRAM configuration that allocates memory, can the module be used. The appropriate UART configuration registers can be seen in [Section 7.6.4](#).

#### 7.5.5.1 UART Baud Rate Generation and Fractional Divisor

The TCAN5102-Q1 UART contains a programmable baud generator that divides reference clock by a divisor in the range between 1 and  $(2^{16} - 1)$  and a decimal resolution of 1/64. The output frequency of the baud rate generator is 16× the baud rate. The input clock to the baud rate generator is always 40MHz. The formula for the divisor is:

$$\text{Divisor} = (40\text{E6} / (\text{Desired baud rate} \times 16)) \quad (2)$$

To calculate the values necessary to put into the registers, the following functions are needed:

- TRUNC(X): Truncate X, return just the integer portion of a real number. EX: TRUNC(3.14) = 3
- ROUND(X): Round X to the nearest integer. EX: ROUND(3.1) = 3 and ROUND(3.6) = 4
- >>: Bit shift towards the right operation. EX: 0x1000 >> 8 = 0x0010. Or 0b0001 0000 0000 0000 >> 8 = 0b0000 0000 0001 0000
- &: Bitwise AND function, used to mask bits. EX: 0x1234 & 0x00FF = 0x0034 and 0x8765 & 0xFF00 = 0x8700

Once the required divisor is found, then the register values can be calculated from:

$$\text{UART\_BR\_MSB} = \text{TRUNC}(\text{Divisor}) \gg 8 \quad (3)$$

$$\text{UART\_BR\_LSB} = \text{TRUNC}(\text{Divisor}) \& 0xFF \quad (4)$$

$$\text{UART\_BR\_FRAC} = \text{ROUND}([ \text{Divisor} - \text{TRUNC}(\text{Divisor}) ] \times 64) \quad (5)$$

**Table 7-16. Baud Rates Using a 16× Baud Divider (40MHz Clock)**

| DESIRED BAUD RATE | DIVISOR USED TO GENERATE 16× CLOCK | CLOSEST DIVISOR OBTAINABLE | UART_BR_MS_B VALUE (HEX) | UART_BR_LSB VALUE (HEX) | UART_BR_FRAC VALUE (HEX) | PERCENT ERROR DIFFERENCE BETWEEN DESIRED AND ACTUAL (%) |
|-------------------|------------------------------------|----------------------------|--------------------------|-------------------------|--------------------------|---------------------------------------------------------|
| 400               | 6250.0000                          | 6250                       | 0x18                     | 0x6A                    | 0x00                     | 0                                                       |
| 2400              | 1041.6667                          | 1041 43/64                 | 0x04                     | 0x11                    | 0x2B                     | 0                                                       |

**Table 7-16. Baud Rates Using a 16× Baud Divider (40MHz Clock) (continued)**

| DESIRED BAUD RATE | DIVISOR USED TO GENERATE 16× CLOCK | CLOSEST DIVISOR OBTAINABLE | UART_BR_MSB VALUE (HEX) | UART_BR_LSB VALUE (HEX) | UART_BR_FRAC VALUE (HEX) | PERCENT ERROR DIFFERENCE BETWEEN DESIRED AND ACTUAL (%) |
|-------------------|------------------------------------|----------------------------|-------------------------|-------------------------|--------------------------|---------------------------------------------------------|
| 4800              | 520.8333                           | 520 53/64                  | 0x02                    | 0x08                    | 0x35                     | 0.001                                                   |
| 9600              | 260.4167                           | 260 27/64                  | 0x01                    | 0x04                    | 0x1B                     | 0.002                                                   |
| 10000             | 250.0000                           | 250                        | 0x00                    | 0xFA                    | 0x00                     | 0                                                       |
| 19200             | 130.2083                           | 130 13/64                  | 0x00                    | 0x82                    | 0x0D                     | 0.004                                                   |
| 25000             | 100.0000                           | 100                        | 0x00                    | 0x64                    | 0x00                     | 0                                                       |
| 28800             | 86.8056                            | 86 52/64                   | 0x00                    | 0x56                    | 0x34                     | 0.008                                                   |
| 38400             | 65.1042                            | 65 7/64                    | 0x00                    | 0x41                    | 0x07                     | 0.008                                                   |
| 50000             | 50.0000                            | 50                         | 0x00                    | 0x32                    | 0x00                     | 0                                                       |
| 57600             | 43.4028                            | 43 26/64                   | 0x00                    | 0x2B                    | 0x1A                     | 0.008                                                   |
| 75000             | 33.3333                            | 33 21/64                   | 0x00                    | 0x21                    | 0x15                     | 0.016                                                   |
| 100000            | 25.0000                            | 25                         | 0x00                    | 0x19                    | 0x00                     | 0                                                       |
| 115200            | 21.7014                            | 21 45/64                   | 0x00                    | 0x15                    | 0x2D                     | 0.008                                                   |
| 128000            | 19.5313                            | 19 34/64                   | 0x00                    | 0x13                    | 0x22                     | 0                                                       |
| 153600            | 16.2760                            | 16 18/64                   | 0x00                    | 0x10                    | 0x12                     | 0.032                                                   |
| 200000            | 12.5000                            | 12 32/64                   | 0x00                    | 0x0C                    | 0x20                     | 0                                                       |
| 225000            | 11.1111                            | 11 7/64                    | 0x00                    | 0x0B                    | 0x07                     | 0.016                                                   |
| 230400            | 10.8507                            | 10 54/64                   | 0x00                    | 0x0A                    | 0x36                     | 0.064                                                   |
| 250000            | 10.0000                            | 10                         | 0x00                    | 0x0A                    | 0x00                     | 0                                                       |
| 256000            | 9.7656                             | 9 49/64                    | 0x00                    | 0x09                    | 0x31                     | 0                                                       |
| 300000            | 8.3333                             | 8 21/64                    | 0x00                    | 0x08                    | 0x15                     | 0.063                                                   |
| 400000            | 6.2500                             | 6 16/64                    | 0x00                    | 0x06                    | 0x10                     | 0                                                       |
| 460800            | 5.4253                             | 5 27/64                    | 0x00                    | 0x05                    | 0x1B                     | 0.064                                                   |
| 500000            | 5.0000                             | 5                          | 0x00                    | 0x05                    | 0x00                     | 0                                                       |
| 750000            | 3.3333                             | 3 21/64                    | 0x00                    | 0x03                    | 0x15                     | 0.156                                                   |
| 921600            | 2.7127                             | 2 46/64                    | 0x00                    | 0x02                    | 0x2E                     | 0.224                                                   |
| 1000000           | 2.5000                             | 2 32/64                    | 0x00                    | 0x02                    | 0x20                     | 0                                                       |

### 7.5.5.2 UART Control Protocol

Since UART is a byte based transfer with no higher level protocol, there is no concept of a frame. The UART module shifts bytes in and out doing a check on each byte for errors. For this reason, each byte received has a status byte associated that alerts the MCU to any errors. Since errors are not expected to be a common occurrence, there is no need to read the status byte for each byte unless there is an error. To save throughput, the end of each UART RX FIFO read has a global status byte appended. This global status byte is the logical or of all status bytes for the bytes that were read out. This allows the MCU to determine if there are any errors reported for the block of bytes that were transferred. If so, a read can be requested. A read to the RX Error Status register to determine which byte is at fault, and the error received.

There is no additional UART-specific header that is used when transmitting or receiving data, only the global status byte on a read of the RX FIFO.

See [UART Transmit FIFO \(address = h2010\)](#) , [UART Transmit FIFO \(address = h2010\)](#) and [UART Receive Error Status \(address = h2011\)](#) for more information about the register and data formats.

#### 7.5.5.2.1 UART Write Example 1

This is a basic example of sending a stream of bytes via UART. Once the device receives any bytes in the TX FIFO, the device begins transmitting the UART bytes.

**Note**

Before the module can be used, the appropriate GPIOs must be set to special function (see [Section 7.3.4](#) and [Section 7.6.1.18](#)) and the MRAM must enable the module by allocation memory (see [Section 7.5.3](#) and [Section 7.6.1.17](#)). Only once the GPIOs have been set to special function, and the module has been enabled by selecting a MRAM configuration that allocates memory, can the module be used. The appropriate UART configuration registers can be seen in [Section 7.6.4](#).

**Table 7-17. Design Parameters and Assumptions**

| PARAMETER              | VALUE                              |
|------------------------|------------------------------------|
| CAN Frame format       | Standard Frame Format              |
| Bytes to transfer      | 6B                                 |
| UART byte stream to TX | 0x11, 0x22, 0x33, 0x44, 0x55, 0x66 |

The following table shows the steps and what data is transmitted on the CAN bus. The central gateway/controller of the bus is referred to as the ECU, and the TCAN device is referred to as "responder".

- Hexadecimal values marked as **BOLD** signify the bytes for CAN Frame header (standard frame format for this example)
- Hexadecimal values without any formatting are UART data bytes

**Table 7-18. Example CAN Sequence (Single CAN Frame)**

| Step | Transmitter     | Data                                                                           | Description                                                                                                                                                          |
|------|-----------------|--------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | ECU (DLC = 12B) | <b>0x06, 0x20, 0x10</b> , 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x00, 0x00, 0x00 | The ECU requests a write of 6 bytes to register 0x2010 (UART TX FIFO), since the CAN DLC must be 12 bytes to fit these 6 bytes of data, a 0x00 is padded to the end. |
| 2    | Responder       | <b>0x06, 0x01</b>                                                              | The responder sends back an OK to acknowledge that the request was received.                                                                                         |
| 3    | UART            | TXD: 0x11, 0x22, 0x33, 0x44, 0x55, 0x66                                        | At this point, the responder does the UART transfer of 6 bytes.                                                                                                      |

**Table 7-19. Example CAN Sequence (Multiple CAN Frames to Transfer Payload)**

| Step | Transmitter    | Data                                       | Description                                                                                                                                                                                       |
|------|----------------|--------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | ECU (DLC = 6B) | <b>0x03, 0x20, 0x10</b> , 0x11, 0x22, 0x33 | The ECU requests a write of 3 bytes to register 0x2010 (UART TX FIFO).                                                                                                                            |
| 2    | Responder      | <b>0x03, 0x01</b>                          | The responder sends back an OK to acknowledge that the request was received. The responder also begins to transfer out the UART here, but for simplicity, this table shows all bytes going later. |
| 3    | ECU (DLC = 6B) | <b>0x03, 0x20, 0x10</b> , 0x44, 0x55, 0x66 | The ECU requests a write of 3 bytes to register 0x2010 (UART TX FIFO) to finish the rest of the data                                                                                              |
| 4    | Responder      | <b>0x03, 0x01</b>                          | The responder sends back an OK to acknowledge that the request was received.                                                                                                                      |
| 5    | UART           | TXD: 0x11, 0x22, 0x33, 0x44, 0x55, 0x66    | At this point, the responder does the UART transfer of 6 bytes.                                                                                                                                   |

**7.5.5.2.2 UART Read Example 1**

This example is a basic example of sending and receiving a stream of bytes via UART. Once the device receives any bytes in the TX FIFO, it will begin transmitting the UART bytes.

**Note**

Before the module can be used, the appropriate GPIOs must be set to special function (see [Section 7.3.4](#) and [Section 7.6.1.18](#)) and the MRAM must enable the module by allocation memory (see [Section 7.5.3](#) and [Section 7.6.1.17](#)). Only once the GPIOs have been set to special function, and the module has been enabled by selecting a MRAM configuration that allocates memory, can the module be used. The appropriate UART configuration registers can be seen in [Section 7.6.4](#).

**Table 7-20. Design Parameters and Assumptions**

| PARAMETER           | VALUE                                                 |
|---------------------|-------------------------------------------------------|
| CAN Frame format    | Standard Frame Format                                 |
| Bytes to transfer   | 6B                                                    |
| UART byte stream RX | 0x11, 0x22, 0x33, 0x44, 0x55, 0x66                    |
| Error               | Parity error on byte 0x33, framing error on byte 0x44 |

The table below shows the steps and what data is transmitted on the CAN bus. The central gateway/controller of the bus will be referred to as the ECU, and the TCAN device will be referred to as "responder".

- Hexadecimal values marked as **BOLD** signify the bytes for CAN Frame header (standard frame format for this example)
- Hexadecimal values without any formatting are UART data bytes

**Table 7-21. Example CAN Sequence (Single CAN Frame)**

| Step | Transmitter     | Data                                                                           | Description                                                                                                                                                                                                                                |
|------|-----------------|--------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | UART            | RXD: 0x11, 0x22, 0x33, 0x44, 0x55, 0x66                                        | The UART RXD receives 6 bytes                                                                                                                                                                                                              |
| 2    | ECU (DLC = 16B) | <b>0x41, 0x20, 0x0B</b>                                                        | The ECU requests a read to 0x200B (UART RX FIFO Status)                                                                                                                                                                                    |
| 3    | Responder       | <b>0x41, 0x01</b> , 0x06                                                       | The responder sends back an OK to acknowledge that the request was received and sends the register data, stating that there are 6 bytes of data in the RX FIFO currently                                                                   |
| 4    | ECU             | <b>0x47, 0x20, 0x10</b>                                                        | The ECU requests a read of 7 bytes (6 data bytes + global status byte) to 0x2010 (UART RX FIFO)                                                                                                                                            |
| 5    | Responder       | <b>0x47, 0x01</b> , 0x11, 0x22, 0x33, 0x44, 0x55, 0x66, 0x0C, 0x00, 0x00, 0x00 | The responder replies with 7 bytes of data (6 UART data bytes and a global status byte). The global status byte indicates that there is both a parity and a frame error on at least 1 byte in the 6 bytes of data that were just read out. |
| 6    | ECU             | <b>0x46, 0x20, 0x11</b>                                                        | Since the ECU saw a non-normal global status byte, a read to 0x2011 (UART RX Error Status) of 6 bytes (no global status byte) is requested.                                                                                                |
| 7    | Responder       | <b>0x46, 0x01</b> , 0x01, 0x01, 0x04, 0x08, 0x01, 0x01                         | The responder sends the 6 status bytes showing that 0x33 had a parity error and 0x44 had a framing error. The ECU can decide how to handle the error.                                                                                      |

**Table 7-22. Example CAN Sequence (Multiple CAN Frames to Transfer Payload)**

| Step | Transmitter | Data                                    | Description                                                                                                                                                                |
|------|-------------|-----------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | UART        | RXD: 0x11, 0x22, 0x33, 0x44, 0x55, 0x66 | The UART RXD receives 6 bytes                                                                                                                                              |
| 2    | ECU         | <b>0x41, 0x20, 0x0B</b>                 | The ECU requests a read to 0x200B (UART RX FIFO Status)                                                                                                                    |
| 3    | Responder   | <b>0x41, 0x01</b> , 0x06                | The responder sends back an OK to acknowledge that the request was received and sends the register data, stating that there are 6 bytes of data in the RX FIFO currently   |
| 4    | ECU         | <b>0x43, 0x20, 0x10</b>                 | The ECU requests a read of 3 bytes (2 data bytes + global status byte) to 0x2010 (UART RX FIFO)                                                                            |
| 5    | Responder   | <b>0x43, 0x01</b> , 0x11, 0x22, 0x01    | The responder replies with 3 bytes of data (2 UART data bytes and a global status byte). The global status byte indicates that there is no error reported (normal status). |
| 6    | ECU         | <b>0x41, 0x20, 0x0B</b>                 | The ECU requests a read to 0x200B (UART RX FIFO Status)                                                                                                                    |

**Table 7-22. Example CAN Sequence (Multiple CAN Frames to Transfer Payload) (continued)**

| Step | Transmitter | Data                                             | Description                                                                                                                                                                                                                                |
|------|-------------|--------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7    | Responder   | <b>0x41, 0x01</b> , 0x04                         | The responder sends back an OK to acknowledge that the request was received and sends the register data, stating that there are 4 bytes of data in the RX FIFO currently                                                                   |
| 8    | ECU         | <b>0x45, 0x20, 0x10</b>                          | The ECU requests a read of 5 bytes (4 data bytes + global status byte) to 0x2010 (UART RX FIFO)                                                                                                                                            |
| 9    | Responder   | <b>0x45, 0x01</b> , 0x33, 0x44, 0x55, 0x66, 0x0C | The responder replies with 3 bytes of data (2 UART data bytes and a global status byte). The global status byte indicates that there is both a parity and a frame error on at least 1 byte in the 4 bytes of data that were just read out. |
| 10   | ECU         | <b>0x44, 0x20, 0x11</b>                          | Since the ECU saw a non-normal global status byte, a read to 0x2011 (UART RX Error Status) of 4 bytes (no global status byte) is requested.                                                                                                |
| 11   | Responder   | <b>0x44, 0x01</b> , 0x04, 0x08, 0x01, 0x01       | The responder sends the 4 status bytes showing that 0x33 had a parity error and 0x44 had a framing error. The ECU can decide how to handle the error.                                                                                      |

### 7.5.6 I2C Controller

#### Note

Before the module can be used, the appropriate GPIOs must be set to special function (see [Section 7.3.4](#) and [Section 7.6.1.18](#)) and the MRAM must enable the module by allocation memory (see [Section 7.5.3](#) and [Section 7.6.1.17](#)). Only once the GPIOs have been set to special function, and the module has been enabled by selecting a MRAM configuration that allocates memory, can the module be used. The appropriate I2C configuration registers can be seen in [Section 7.6.6](#).

#### 7.5.6.1 I2C Baud Rate Generation

The TCAN5102-Q1 I2C module contains a programmable baud rate generator that divides reference clock by a divisor in the range between 1 and 255. The input clock to the divider can be either 10MHz or 2.5MHz, and is set by the I2C\_CTRL.LSM (low speed mode) bit. The formula for the divisor is:

$$\text{Divisor} = (\text{Input Clock} / \text{Desired I2C Speed}) - 1 \quad (6)$$

**Table 7-23. I2C Speed Using the High Speed Clock (10MHz Clock, I2C\_CTRL.LSM = 0)**

| DESIRED I2C SPEED (kHz) | CLOSEST DIVISOR OBTAINABLE | I2C_BR VALUE (HEX) | ACTUAL I2C SPEED (kHz) | PERCENT ERROR DIFFERENCE BETWEEN DESIRED AND ACTUAL (%) |
|-------------------------|----------------------------|--------------------|------------------------|---------------------------------------------------------|
| 39                      | 255                        | 0xFF               | 39                     | 0.16                                                    |
| 50                      | 199                        | 0xC7               | 50                     | 0                                                       |
| 60                      | 166                        | 0xA6               | 60                     | 0.2                                                     |
| 70                      | 142                        | 0x8E               | 70                     | 0.1                                                     |
| 80                      | 124                        | 0x7C               | 80                     | 0                                                       |
| 100                     | 99                         | 0x63               | 100                    | 0                                                       |
| 200                     | 49                         | 0x31               | 200                    | 0                                                       |
| 250                     | 39                         | 0x27               | 250                    | 0                                                       |
| 400                     | 24                         | 0x18               | 400                    | 0                                                       |
| 500                     | 19                         | 0x13               | 500                    | 0                                                       |
| 600                     | 16                         | 0x10               | 588                    | 2                                                       |
| 800                     | 12                         | 0x0C               | 769                    | 4                                                       |
| 1000                    | 9                          | 0x09               | 1000                   | 0                                                       |

**Table 7-24. I2C Speed Using the Low Speed Clock (2.5MHz Clock, I2C\_CTRL.LSM = 1)**

| DESIRED I2C SPEED (kHz) | CLOSEST DIVISOR OBTAINABLE | I2C_BR VALUE (HEX) | ACTUAL I2C SPEED (kHz) | PERCENT ERROR DIFFERENCE BETWEEN DESIRED AND ACTUAL (%) |
|-------------------------|----------------------------|--------------------|------------------------|---------------------------------------------------------|
| 10                      | 249                        | 0xF9               | 10                     | 0                                                       |
| 50                      | 49                         | 0x31               | 50                     | 0                                                       |
| 60                      | 41                         | 0x29               | 60                     | 0.8                                                     |
| 70                      | 35                         | 0x23               | 69                     | 0.8                                                     |
| 80                      | 30                         | 0x1E               | 81                     | 0.8                                                     |
| 100                     | 24                         | 0x18               | 100                    | 0                                                       |
| 200                     | 12                         | 0x0C               | 192                    | 4                                                       |
| 250                     | 9                          | 0x09               | 250                    | 0                                                       |
| 400                     | 5                          | 0x05               | 417                    | 4                                                       |
| 500                     | 4                          | 0x04               | 500                    | 0                                                       |
| 600                     | 3                          | 0x03               | 625                    | 4                                                       |
| 800                     | 2                          | 0x02               | 833                    | 4                                                       |

#### 7.5.6.2 I2C Stuck Bus Recovery

The I2C controller has an optional feature called stuck bus recovery (SBR). If enabled, the IP automatically attempts to unstuck a bus once a stuck condition is detected. The detection of a stuck bus is if either the SCL or SDA line is held low for  $t_{\text{STUCKBUS\_I2C}}$  time. Once a stuck condition is detected, and the automatic SBR is enabled, the I2C controller attempts to generate up to 16 clock pulses. While generating the clock pulses, the controller monitors to see if the stuck condition resolves. Once the condition resolves, the clock generation is immediately stopped and a STOP condition is generated to reset the bus. Once the stuck bus recovery sequence is complete, a SBRC (stuck bus recovery complete) interrupt is set. The state of the I2C bus can be polled via the I2C\_STATUS register. A stuck bus recovery sequence can be manually requested, by writing to the SBR\_START bit in the I2C\_CTRL register.

#### Note

Timers for the stuck bus detection starts once the I2C IP is enabled and connected to the GPIO. Take care to make sure that there is a pull up on the I2C bus prior to enabling the I2C IP, otherwise a false stuck bus condition can occur

#### 7.5.6.3 I2C Control Protocol

When writing to the I2C Transmit (TX) FIFO, there is a 2 byte header required for I2C address and number of data bytes. When reading from the I2C Receive (RX) FIFO, there are 3 additional header bytes which provide additional robustness when reading from a target peripheral. There are 2 header formats, which are similar but differ depending on if the action is a read or a write to the TX or RX FIFO.

When writing to the TX FIFO, the 2 byte header is the first 2 bytes written to the FIFO.

When reading the RX FIFO, there is always the 2-byte header at the beginning of the data field which contains information on how many bytes are remaining and if the read is a continuation of a previous read (needed for SPI messages that are larger than can fit in a single CAN frame).

If multiple I2C frames are queued in the TX FIFO, the IP will use a repeated start instead of a stop and a start.

See [Section 7.6.7.1](#) and [Section 7.6.7.2](#) for more information about the header formats.

##### 7.5.6.3.1 I2C Write Example 1

This example is a basic example of sending a stream of bytes via I2C.

**Note**

Before the module can be used, the appropriate GPIOs must be set to special function (see [Section 7.3.4](#) and [Section 7.6.1.18](#)) and the MRAM must enable the module by allocation memory (see [Section 7.5.3](#) and [Section 7.6.1.17](#)). Only once the GPIOs have been set to special function, and the module has been enabled by selecting a MRAM configuration that allocates memory, can the module be used. The appropriate I2C configuration registers can be seen in [Section 7.6.6](#).

**Table 7-25. Design Parameters and Assumptions**

| PARAMETER                  | VALUE                                          |
|----------------------------|------------------------------------------------|
| CAN Frame format           | Standard Frame Format                          |
| I2C Target Address (7-bit) | 0x12                                           |
| I2C Frame                  | Write                                          |
| Bytes to transfer          | 8B                                             |
| I2C data to write          | 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08 |

The following table shows the steps and what data is transmitted on the CAN bus. The central gateway/controller of the bus is referred to as the ECU, and the TCAN device is referred to as "responder".

- Hexadecimal values marked as **BOLD** signify the bytes for CAN Frame header (standard frame format for this example)
- Hexadecimal values marked as *ITALIC* signify the I2C header bytes
- Hexadecimal values without any formatting are I2C data bytes

**Table 7-26. Example CAN Sequence (Single CAN Frame)**

| Step | Transmitter     | Data                                                                                                                   | Description                                                                                                                                                                                                                                                                                                                              |
|------|-----------------|------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | ECU (DLC = 16B) | <b>0x0A</b> , <b>0x30</b> , <b>0x10</b> , 0x25, 0x88, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x00, 0x00, 0x00 | The ECU requests a write of 10 bytes to register 0x3010 (I2C TX FIFO), with an I2C header with store = 1, I2C target address = 0x12, write = 1, and I2C data size of 8 bytes. The 10 bytes of data are then shifted into the device. Since the CAN DLC must be 16 bytes to fit these 13 bytes of data, 0x00 bytes are padded to the end. |
| 2    | Responder       | <b>0x0A</b> , <b>0x01</b>                                                                                              | The responder sends back an OK to acknowledge that the request was received.                                                                                                                                                                                                                                                             |
| 3    | I2C             | I2C Frame: 0x25, 0x88, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08                                                  | At this point, the responder begins the I2C transfer of 10 bytes.                                                                                                                                                                                                                                                                        |

**Table 7-27. Example CAN Sequence (Multiple CAN Frames to Transfer Payload)**

| Step | Transmitter     | Data                                                                                           | Description                                                                                                                                                                                                                           |
|------|-----------------|------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | ECU (DLC = 7B)  | <b>0x04</b> , <b>0x30</b> , <b>0x10</b> , 0x25, 0x88, 0x01, 0x02                               | The ECU requests a write of 4 bytes to register 0x3010 (I2C TX FIFO), with an I2C header with store = 1, I2C target address = 0x12, write = 1, and I2C data size of 8 bytes. The 10 bytes of data are then shifted into the device.   |
| 2    | Responder       | <b>0x04</b> , <b>0x01</b>                                                                      | The responder sends back an OK to acknowledge that the request was received.                                                                                                                                                          |
| 3    | ECU (DLC = 12B) | <b>0x06</b> , <b>0x30</b> , <b>0x10</b> , 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x00, 0x00, 0x00 | The ECU continues the write request to the same address of 6 bytes. Since the CAN DLC that can store this frame is 12 bytes, 3 bytes of padding are added to the end. The value of these padded bytes do not matter, and are ignored. |
| 4    | Responder       | <b>0x06</b> , <b>0x01</b>                                                                      | The responder sends back an OK to acknowledge that the request was received.                                                                                                                                                          |

**Table 7-27. Example CAN Sequence (Multiple CAN Frames to Transfer Payload) (continued)**

| Step | Transmitter | Data                                                                  | Description                                                       |
|------|-------------|-----------------------------------------------------------------------|-------------------------------------------------------------------|
| 5    | I2C         | I2C Frame: 0x25, 0x88, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08 | At this point, the responder begins the I2C transfer of 10 bytes. |

#### 7.5.6.3.2 I2C Read Example 1

This is a basic example of sending and receiving a stream of bytes via I2C. Once the device receives any bytes in the TX FIFO, the device begins transmitting the I2C bytes.

#### Note

Before the module can be used, the appropriate GPIOs must be set to special function (see [Section 7.3.4](#) and [Section 7.6.1.18](#)) and the MRAM must enable the module by allocation memory (see [Section 7.5.3](#) and [Section 7.6.1.17](#)). Only once the GPIOs have been set to special function, and the module has been enabled by selecting a MRAM configuration that allocates memory, can the module be used. The appropriate I2C configuration registers can be seen in [Section 7.6.6](#).

**Table 7-28. Design Parameters and Assumptions**

| PARAMETER                   | VALUE                 |
|-----------------------------|-----------------------|
| CAN Frame format            | Standard Frame Format |
| I2C Target Address (7-bit)  | 0x12                  |
| I2C Frame                   | Read                  |
| I2C Target Register Address | 0x10                  |
| Bytes to transfer           | 8B                    |

The following table shows the steps and what data is transmitted on the CAN bus. The central gateway/controller of the bus is referred to as the ECU, and the TCAN device is referred to as "responder".

- Hexadecimal values marked as **BOLD** signify the bytes for CAN Frame header (standard frame format for this example)
- Hexadecimal values marked as *ITALIC* signify the I2C header bytes
- Hexadecimal values without any formatting are I2C data bytes

When doing an I2C read, a common practice is to write the register address before doing the read. This lets the target device know which register to read. This means that an I2C read is normally 2 separate I2C frames: an I2C write followed by an I2C read. This example is shown as follows.

**Table 7-29. Example CAN Sequence (Single CAN Frame)**

| Step | Transmitter    | Data                                                       | Description                                                                                                                                                                                        |
|------|----------------|------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | ECU (DLC = 6B) | <b>0x03</b> , <b>0x30</b> , <b>0x10</b> , 0x25, 0x81, 0x10 | The ECU requests a write of 3 bytes to register 0x3010 (I2C TX FIFO), with an I2C header with store = 1, I2C target address = 0x12, write = 1, and I2C data size of 1 byte (register address 0x10) |
| 2    | Responder      | <b>0x03</b> , <b>0x01</b>                                  | The responder sends back an OK to acknowledge that the request was received.                                                                                                                       |
| 3    | I2C            | I2C Frame: 0x25, 0x10                                      | At this point, the responder begins the I2C transfer of 1 data byte.                                                                                                                               |
| 4    | ECU (DLC = 3B) | <b>0x43</b> , <b>0x30</b> , <b>0x10</b>                    | The ECU requests a read to the RX FIFO to check the status of the initial I2C write to set the target address. This is done to verify that the target has acknowledged.                            |
| 5    | Responder      | <b>0x43</b> , <b>0x01</b> , 0x25, 0x01, 0x01               | The responder sends back an I2C status flag of 0x01, signaling 'SUCCESS' status, meaning the target acknowledged both the address and the data (register address).                                 |
| 6    | ECU (DLC = 5B) | <b>0x02</b> , <b>0x30</b> , <b>0x10</b> , 0x24, 0x08       | The ECU now requests a write of 2 bytes to register 0x3010 (I2C TX FIFO), with an I2C header with store = 1, I2C target address = 0x12, write = 1, and I2C read size of 8 bytes                    |

**Table 7-29. Example CAN Sequence (Single CAN Frame) (continued)**

| Step | Transmitter    | Data                                                                                                  | Description                                                                                                                                                                                                                                                                                                                          |
|------|----------------|-------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7    | Responder      | <b>0x02, 0x01</b>                                                                                     | The responder sends back an OK to acknowledge that the request was received.                                                                                                                                                                                                                                                         |
| 8    | I2C            | I2C Frame: 0x24, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08                                       | At this point, the responder begins the I2C read that's 8 bytes long                                                                                                                                                                                                                                                                 |
| 9    | ECU (DLC = 3B) | <b>0x4B, 0x30, 0x10</b>                                                                               | The ECU requests a read to the RX FIFO of 11 bytes (2 header bytes + 8 data bytes + 1 status byte) to get the data that was read from the I2C target                                                                                                                                                                                 |
| 10   | Responder      | <b>0x4B, 0x01, 0x24, 0x09, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x01, 0x00, 0x00, 0x00</b> | The responder sends back the I2C header information along with 8 bytes of data. The data is followed up with an I2C status flag of 0x01, signaling 'SUCCESS' status, meaning the target acknowledged the address. Since the closest CAN FD payload size that can hold the data is 16 bytes, 0x00 padding bytes are added to the end. |

**Table 7-30. Example CAN Sequence (Multiple CAN Frames to Transfer Payload)**

| Step | Transmitter    | Data                                                                          | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------|----------------|-------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | ECU (DLC = 6B) | <b>0x03, 0x30, 0x10, 0x25, 0x81, 0x10</b>                                     | The ECU requests a write of 3 bytes to register 0x3010 (I2C TX FIFO), with an I2C header with store = 1, I2C target address = 0x12, write = 1, and I2C data size of 1 byte (register address 0x10)                                                                                                                                                                                                                                                           |
| 2    | Responder      | <b>0x0A, 0x01</b>                                                             | The responder sends back an OK to acknowledge that the request was received.                                                                                                                                                                                                                                                                                                                                                                                 |
| 3    | I2C            | I2C Frame: 0x25, 0x10                                                         | At this point, the responder begins the I2C transfer of 1 data byte.                                                                                                                                                                                                                                                                                                                                                                                         |
| 4    | ECU (DLC = 3B) | <b>0x43, 0x30, 0x10</b>                                                       | The ECU requests a read to the RX FIFO to check the status of the initial I2C write to set the target address. This is done to verify that the target has acknowledged.                                                                                                                                                                                                                                                                                      |
| 5    | Responder      | <b>0x43, 0x01, 0x25, 0x01, 0x01</b>                                           | The responder sends back an I2C status flag of 0x01, signalling 'SUCCESS' status, meaning the target acknowledged both the address and the data (register address).                                                                                                                                                                                                                                                                                          |
| 6    | ECU (DLC = 5B) | <b>0x02, 0x30, 0x10, 0x24, 0x08</b>                                           | The ECU now requests a write of 2 bytes to register 0x3010 (I2C TX FIFO), with an I2C header with store = 1, I2C target address = 0x12, write = 1, and I2C read size of 8 bytes                                                                                                                                                                                                                                                                              |
| 7    | Responder      | <b>0x02, 0x01</b>                                                             | The responder sends back an OK to acknowledge that the request was received.                                                                                                                                                                                                                                                                                                                                                                                 |
| 8    | I2C            | I2C Frame: 0x24, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08               | At this point, the responder begins the I2C read that's 8 bytes long                                                                                                                                                                                                                                                                                                                                                                                         |
| 9    | ECU (DLC = 3B) | <b>0x46, 0x30, 0x10</b>                                                       | The ECU requests a read to the RX FIFO of 6 bytes to get the data that was read from the I2C target                                                                                                                                                                                                                                                                                                                                                          |
| 10   | Responder      | <b>0x46, 0x01, 0x24, 0x09, 0x01, 0x02, 0x03, 0x04</b>                         | The responder sends back the I2C header information along with 8 bytes of data.                                                                                                                                                                                                                                                                                                                                                                              |
| 11   | ECU (DLC = 3B) | <b>0x47, 0x30, 0x10</b>                                                       | The ECU requests a read to the RX FIFO of 7 bytes (2 header bytes + 8 data bytes + 1 status byte - the already 4 bytes read) to get the data that was read from the I2C target                                                                                                                                                                                                                                                                               |
| 12   | Responder      | <b>0x47, 0x01, 0x24, 0x85, 0x05, 0x06, 0x07, 0x08, 0x01, 0x00, 0x00, 0x00</b> | The responder sends back the I2C header information along with 8 bytes of data. The second I2C header byte has the CONT flag set, meaning that this is the continuation of the partially-read I2C frame. The data is followed up with an I2C status flag of 0x01, signaling 'SUCCESS' status, meaning the target acknowledged the address. Since the closest CAN FD payload size that can hold the data is 12 bytes, 0x00 padding bytes are added to the end |

### 7.5.7 PWM and Trapezoidal PWM Ramp Profiles

The TCAN5102-Q1 features 2 highly configurable PWM outputs (PWM0 and PWM1), supporting 8-bit or 10-bit output resolution with a wide frequency range of 20Hz to 100KHz. There are 2 main modes of operation: static PWM output and trapezoidal PWM ramp control. Ramp control allows for ramping 2 parameters: the duty cycle or the frequency of the PWM output.

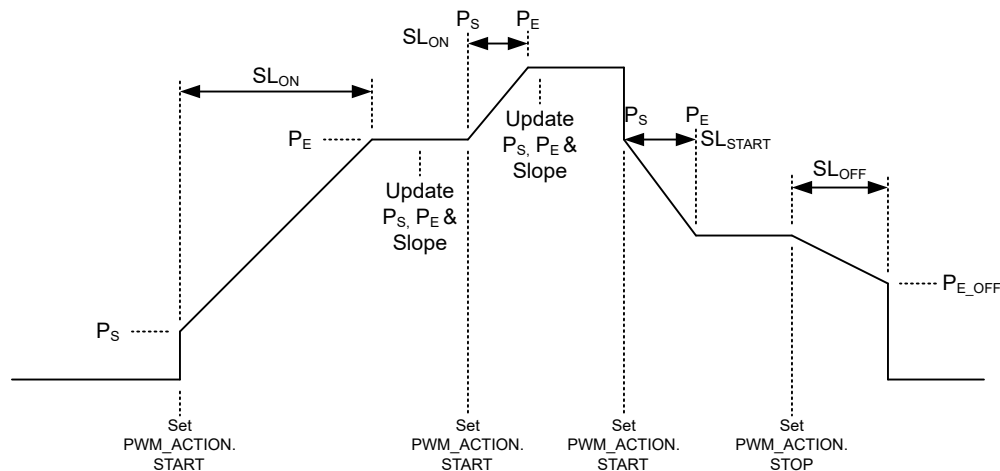
### Note

The PWM module must be muxed to a GPIO by setting the appropriate GPIO to the special function mode in the IO\_CFG\_1. If this is not done, the PWM signal is NOT muxed to a pin.

When the PWM mode is configured as static output, only a subset of registers are needed, since no ramping occurs.

- Frequency is controlled by the PWM\_END\_VAL registers (0x4019-0x401B and 0x4119-0x411B)
- Duty cycle is controlled by PWM\_CONST registers (0x400B-0x400C and 0x410B-0x410C)

There are 2 configurable ramps: An on-ramp, and an off-ramp. The on-ramp can be used to create different ramp profiles by updating the ramp values before requesting another on-ramp. The off-ramp is used to provide a predictable and controlled ramp down/off transition. A hard shut off is available to immediately disable the PWM output. Once an on-ramp completes, the PWM output remains at the end value until instructed to change. An off-ramp ramps to the specified ending point and then disables the PWM output.



**Figure 7-8. Example PWM Ramping**

Where the following points are defined by:

- $P_S$ : Starting point (for on-ramp), describes the starting point of an on-ramp.
- $P_E$ : Ending point (for on-ramp), describes the ending point of an on-ramp.
- $SL_{ON}$ : Ramp slope (for on-ramp), describes the rate of change during an on-ramp.
- $P_{E\_OFF}$ : Ending point (for off-ramp), describes the ending point of an off-ramp.
- $SL_{OFF}$ : Ramp slope (for off-ramp), describes the rate of change during an off-ramp.

There are 3 values that make up a PWM ramp: A starting point, an ending point, and the ramp slope. The starting point of an on-ramp can be either the current PWM value, or a specified starting point. This allows for flexibility to create a step-function type behavior, if desired, or allow for a smooth change to an existing ramp. The ramp slope is used to describe the rate of change of the varied-parameter over time. As an example, the ramp slope can be used to tell the device to change the output duty cycle at a rate of 5% per second.

Since there are 2 PWM parameters that can be varied (PWM frequency and duty cycle), the non-varied-parameter is a static value that is configurable by the user. For example, if the user is varying the PWM frequency for a ramp, the duty cycle can be set to any value (such as 30%), but cannot change during a ramp.

### Note

Updating the values of the on-ramp or off-ramp profile settings during the applicable ramp does NOT change the ramp values in real time. The ramp settings are loaded into a buffer when an action command is set (such as a start or stop action).

### 7.5.7.1 Trapezoidal Ramp Control

The TCAN5102-Q1 supports hardware-controlled trapezoidal ramps. There are 2 parameters supported for ramping:

- Duty cycle ramping
- Switching frequency ramping

There is one pair of starting point and ending point values for ramping the varied parameter (duty cycle or switching frequency). This pair describes the starting and ending points of either duty cycle or the switching frequency of the PWM output. The parameter that is not being varied (non-varied-parameter) has only 1 value which can be set, and this value does not change during a ramp.

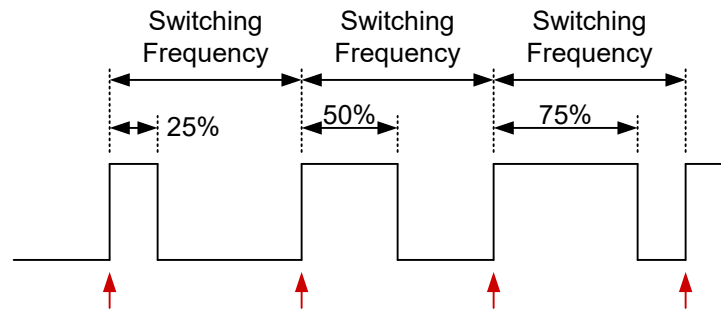
Off-ramps are a defined behavior when commanded to turn off. This consists of a defined stop value to ramp to, along with the specified rate of change. Once the off-ramp completes, the PWM output is disabled until re-enabled.

#### 7.5.7.1.1 Duty Cycle Ramping

The TCAN5102-Q1 supports ramping by varying the duty cycle. When the duty cycle is being ramped, the switching frequency is a fixed value.

During a ramp, the duty cycle is updated at the start of each period. Another way to say this is that the output duty cycle value is updated at the switching frequency rate.

In [Figure 7-9](#) the duty cycle value is updated at the start of each period as shown by the red arrows.



**Figure 7-9. Example Duty Cycle Ramping**

#### Note

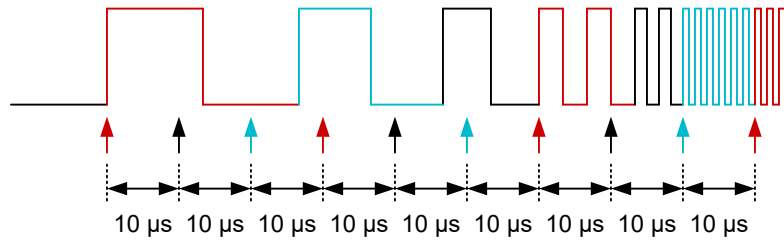
If the ramp is relatively slow, several periods can occur before a different value is loaded. This occurs due to the minimum step size of the 8 or 10-bit output.

#### 7.5.7.1.2 Switching Frequency Ramping

The TCAN5102-Q1 supports ramping by varying the switching frequency. When the switching frequency is being ramped, the duty cycle is fixed.

During a ramp, the switching frequency value is calculated every 10  $\mu$ s, but the output switching frequency is only updated at the start of a period. This is to make sure that at least 1 full period at a frequency takes place.

As shown in [Figure 7-10](#), the frequency value for the ramp is calculated every 10  $\mu$ s, but the last value generated at the start of a new period is used. For speeds above 100 kHz, multiple periods occur before the value is updated.



**Figure 7-10. Switching Frequency Ramp Updates**

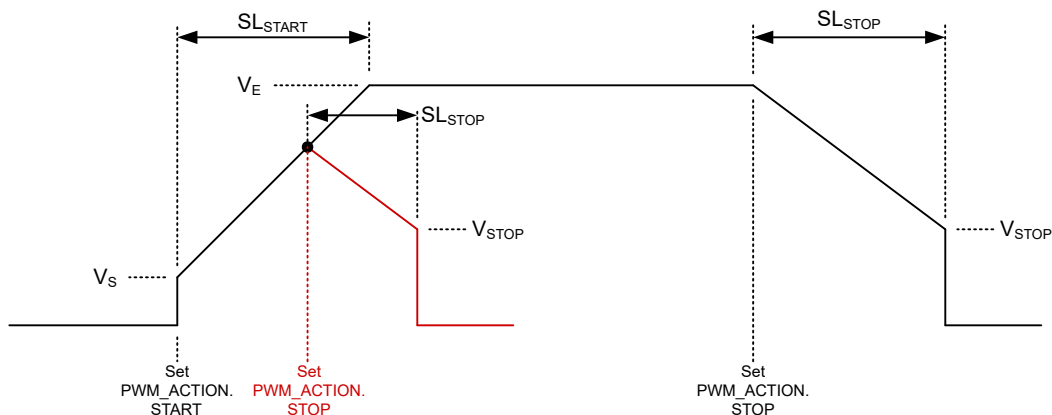
**Note**

If the ramp is slow, several periods can occur before a different switching frequency is loaded. This occurs due to the minimum step size.

### 7.5.7.1.3 Off-Ramp

The TCAN5102-Q1 has a separate set of values to be used for the off-ramp, allowing for a specified shut off behavior. These values are used when an off-ramp is requested by writing to the STOP bit in the PWM\_ACTION register. The timing of when the off-ramp is requested does not matter, and can occur in the middle of an existing on-ramp or when the PWM output is not currently ramping. The off-ramp always obeys the specified off-ramp slope and ending point of the off-ramp registers. If the ending point is non-zero, then the PWM output shuts off once the varied-parameter reaches the specified ending point. This allows for creating a step-function, useful for driving motors. The starting point of an off-ramp is always the current value of the varied-parameter.

In Figure 7-11, there are 2 examples shown for an off-ramp. The red line shows the output if the STOP bit is set during an on-ramp. The black line shows the STOP bit being set after the on-ramp finished and the PWM output is idle.



**Figure 7-11. Example PWM Off-Ramps**

### 7.5.7.2 PWM Clock Generator

The TCAN5102-Q1 features a PWM clock generator that supports fractional values. The frequency value is always an 11-bit integer divisor with a 7-bit fractional divisor. The input clock to the PWM clock generator is a 40MHz clock, and can be set up to generate an 8-bit (256 steps) or a 10-bit (1024 steps) output (duty cycle resolution). Possible divisors are in the range between 1 and  $(2^{11} - 1)$  with a decimal resolution of  $1/128$ .

The equation to calculate the divisor is:  $\text{divisor} = (40E6 / (\text{Desired Switching Frequency} \times 2^{\text{RES}}))$

Where 'RES' is the number of bits for the resolution of the PWM output (8 or 10).

To calculate the necessary register values, the following functions are needed:

- TRUNC(x): Truncate x. Return just the integer portion of the number. (Ex: TRUNC(3.14) = 3)
- ROUND(x): Round x to the nearest integer. (Ex: ROUND(3.1) = 3 and ROUND(3.6) = 4)

- >>: Bit shift towards the right operation. EX:  $0x1000 \gg 8 = 0x0010$ . Or  $0b0001\ 0000\ 0000\ 0000 \gg 8 = 0b0000\ 0000\ 0001\ 0000$
- &: Bitwise AND function, used to mask bits. EX:  $0x1234 \& 0x00FF = 0x0034$  and  $0x8765 \& 0xFF00 = 0x8700$

Once the divisor is calculated, the register values can be calculated with the following 3 equations:

$$\text{FREQ MSB} = \text{TRUNC}(\text{Divisor}) \gg 8 \quad (7)$$

$$\text{FREQ LSB} = \text{TRUNC}(\text{Divisor}) \& 0x00FF \quad (8)$$

$$\text{FREQ FRAC} = \text{ROUND}([ \text{Divisor} - \text{TRUNC}(\text{Divisor}) ] \times 128) \quad (9)$$

**Table 7-31. Switching Frequencies (8-Bit Output)**

| DESIRED SWITCHING FREQUENCY (Hz) | DIVISOR TO GENERATE FREQUENCY | CLOSEST DIVISOR OBTAINABLE | FREQ MSB VALUE (HEX) | FREQ LSB VALUE (HEX) | FREQ FRAC VALUE (HEX) | PERCENT ERROR DIFFERENCE BETWEEN DESIRED AND ACTUAL (%) |
|----------------------------------|-------------------------------|----------------------------|----------------------|----------------------|-----------------------|---------------------------------------------------------|
| 100                              | 1562.5000                     | $1562\ 64/128$             | 0x06                 | 0x1A                 | 0x40                  | 0                                                       |
| 250                              | 625.0000                      | 625                        | 0x02                 | 0x71                 | 0x00                  | 0                                                       |
| 500                              | 312.5000                      | $312\ 64/128$              | 0x01                 | 0x38                 | 0x40                  | 0                                                       |
| 1,000                            | 156.2500                      | $156\ 32/128$              | 0x00                 | 0x9C                 | 0x20                  | 0                                                       |
| 2,500                            | 62.5000                       | $62\ 64/128$               | 0x00                 | 0x3E                 | 0x40                  | 0                                                       |
| 5,000                            | 31.2500                       | $31\ 32/128$               | 0x00                 | 0x1F                 | 0x20                  | 0                                                       |
| 7,500                            | 20.8333                       | $20\ 107/128$              | 0x00                 | 0x14                 | 0x6B                  | 0.013                                                   |
| 10,000                           | 15.6250                       | $15\ 80/128$               | 0x00                 | 0x0F                 | 0x50                  | 0                                                       |
| 25,000                           | 6.2500                        | $6\ 32/128$                | 0x00                 | 0x06                 | 0x20                  | 0                                                       |
| 50,000                           | 3.1250                        | $3\ 16/128$                | 0x00                 | 0x03                 | 0x10                  | 0                                                       |
| 75,000                           | 2.0833                        | $2\ 11/128$                | 0x00                 | 0x02                 | 0x0B                  | 0.125                                                   |
| 100,000                          | 1.5625                        | $1\ 72/128$                | 0x00                 | 0x01                 | 0x48                  | 0                                                       |

**Table 7-32. Switching Frequencies (10-Bit Output)**

| DESIRED SWITCHING FREQUENCY (Hz) | DIVISOR TO GENERATE FREQUENCY | CLOSEST DIVISOR OBTAINABLE | FREQ MSB VALUE (HEX) | FREQ LSB VALUE (HEX) | FREQ FRAC VALUE (HEX) | PERCENT ERROR DIFFERENCE BETWEEN DESIRED AND ACTUAL (%) |
|----------------------------------|-------------------------------|----------------------------|----------------------|----------------------|-----------------------|---------------------------------------------------------|
| 50                               | 781.2500                      | $781\ 32/128$              | 0x03                 | 0x0D                 | 0x20                  | 0                                                       |
| 100                              | 390.6250                      | $390\ 80/128$              | 0x01                 | 0x86                 | 0x50                  | 0                                                       |
| 250                              | 156.2500                      | $156\ 32/128$              | 0x00                 | 0x9C                 | 0x20                  | 0                                                       |
| 500                              | 78.1250                       | $78\ 16/128$               | 0x00                 | 0x4E                 | 0x10                  | 0                                                       |
| 1,000                            | 39.0625                       | $39\ 8/128$                | 0x00                 | 0x27                 | 0x08                  | 0                                                       |
| 2,500                            | 15.6250                       | $15\ 80/128$               | 0x00                 | 0x0F                 | 0x50                  | 0                                                       |
| 5,000                            | 7.8125                        | $7\ 104/128$               | 0x00                 | 0x07                 | 0x68                  | 0                                                       |
| 7,500                            | 5.2083                        | $5\ 27/128$                | 0x00                 | 0x05                 | 0x1B                  | 0.05                                                    |
| 10,000                           | 3.9063                        | $3\ 116/128$               | 0x00                 | 0x03                 | 0x74                  | 0                                                       |
| 25,000                           | 1.5625                        | $1\ 72/128$                | 0x00                 | 0x01                 | 0x48                  | 0                                                       |

#### Note

Values greater than 100kHz are possible to generate while configured in 8-bit output mode. There are special considerations when running a frequency ramp. See [Switching Frequency Ramping](#) for more information.

The appropriate register to write the values to vary depending on the parameter that's being ramped. For example, if the ramp is a switching frequency ramp, then the register values go into the PWM\_START, PWM\_END, and PWM\_OFF register groups. If the ramp is a duty cycle ramp, then the frequency register values go into PWM\_CONST and PWM\_END\_VAL\_CONST\_FRAC\_F.

### 7.5.7.3 PWM Duty Cycle

The TCAN5102-Q1 features a PWM module that can be set up to generate an 8-bit (256 steps) or a 10-bit (1024 steps) output (duty cycle resolution).

The duty cycle count value is straight forward to calculate and changes based on if the PWM duty cycle is configured as 8-bit mode or 10-bit mode.

To calculate the necessary register values, the following function is needed:

- ROUND(x): Round x to the nearest integer. (Ex: ROUND(3.1) = 3 and ROUND(3.6) = 4)

DC<sub>TARGET</sub> represents the desired duty cycle percentage as a float (Ex: 40% = 0.40, 25% = 0.25)

$$DC\_COUNT_{8-BIT} = \text{ROUND}(DC_{TARGET} \times 256) \quad (10)$$

$$DC\_COUNT_{10-BIT} = \text{ROUND}(DC_{TARGET} \times 1024) \quad (11)$$

**Table 7-33. Duty Cycle**

| DESIRED<br>DUTY<br>CYCLE<br>(%) | PWM_CTRL.DC_8B = 1<br>8-BIT |                                |                                 |                                | PWM_CTRL.DC_8B = 0<br>10-BIT |                                |                                 |                                |
|---------------------------------|-----------------------------|--------------------------------|---------------------------------|--------------------------------|------------------------------|--------------------------------|---------------------------------|--------------------------------|
|                                 | DUTY<br>CYCLE<br>COUNT      | CLOSEST<br>COUNT<br>OBTAINABLE | DUTY<br>CYCLE<br>COUNT<br>(HEX) | ACTUAL<br>DUTY<br>CYCLE<br>(%) | DUTY<br>CYCLE<br>COUNT       | CLOSEST<br>COUNT<br>OBTAINABLE | DUTY<br>CYCLE<br>COUNT<br>(HEX) | ACTUAL<br>DUTY<br>CYCLE<br>(%) |
| 5                               | 12.800                      | 13                             | 0x0D                            | 5.08%                          | 51.200                       | 51                             | 0x033                           | 4.98%                          |
| 10                              | 25.600                      | 26                             | 0x1A                            | 10.16%                         | 102.400                      | 102                            | 0x066                           | 9.96%                          |
| 15                              | 38.400                      | 38                             | 0x26                            | 14.84%                         | 153.600                      | 154                            | 0x099                           | 15.04%                         |
| 20                              | 51.200                      | 51                             | 0x33                            | 19.92%                         | 204.800                      | 205                            | 0x0CC                           | 20.02%                         |
| 25                              | 64.000                      | 64                             | 0x40                            | 25.00%                         | 256.000                      | 256                            | 0x100                           | 25.00%                         |
| 30                              | 76.800                      | 77                             | 0x4D                            | 30.08%                         | 307.200                      | 307                            | 0x133                           | 29.98%                         |
| 35                              | 89.600                      | 90                             | 0x5A                            | 35.16%                         | 358.400                      | 358                            | 0x166                           | 34.96%                         |
| 40                              | 102.400                     | 102                            | 0x66                            | 39.84%                         | 409.600                      | 410                            | 0x199                           | 40.04%                         |
| 45                              | 115.200                     | 115                            | 0x73                            | 44.92%                         | 460.800                      | 461                            | 0x1CC                           | 45.02%                         |
| 50                              | 128.000                     | 128                            | 0x80                            | 50.00%                         | 512.000                      | 512                            | 0x200                           | 50.00%                         |
| 55                              | 140.800                     | 141                            | 0x8D                            | 55.08%                         | 563.200                      | 563                            | 0x233                           | 54.98%                         |
| 60                              | 153.600                     | 154                            | 0x9A                            | 60.16%                         | 614.400                      | 614                            | 0x266                           | 59.96%                         |
| 65                              | 166.400                     | 166                            | 0xA6                            | 64.84%                         | 665.600                      | 666                            | 0x299                           | 65.04%                         |
| 70                              | 179.200                     | 179                            | 0xB3                            | 69.92%                         | 716.800                      | 717                            | 0x2CC                           | 70.02%                         |
| 75                              | 192.000                     | 192                            | 0xC0                            | 75.00%                         | 768.000                      | 768                            | 0x300                           | 75.00%                         |
| 80                              | 204.800                     | 205                            | 0xCD                            | 80.08%                         | 819.200                      | 819                            | 0x333                           | 79.98%                         |
| 85                              | 217.600                     | 218                            | 0xDA                            | 85.16%                         | 870.400                      | 870                            | 0x366                           | 84.96%                         |
| 90                              | 230.400                     | 230                            | 0xE6                            | 89.84%                         | 921.600                      | 922                            | 0x399                           | 90.04%                         |
| 95                              | 243.200                     | 243                            | 0xF3                            | 94.92%                         | 972.800                      | 973                            | 0x3CC                           | 95.02%                         |

### 7.5.7.4 Ramp Slope and Scale Factor

Slope control is part of the trapezoidal PWM ramp, to have a linear change of the varied parameter (such as duty cycle or switching frequency) over time. This device specifies the slope in either Hz/10 μs (for switching frequency ramps) or %/period (for duty cycle ramps).

The slope values are made up of 21-bits, where the bits are split between integer and fractional fields. The split between integer and fractional is determined by the PWM\_CTRL.SLOPE\_SCALE bits. Being adjustable lets the user be able to choose between slow and fast ramps. See the applicable sections below to determine which range is needed.

#### Note

The split of integer and fractional bits is different for the same slope scale depending on if duty cycle or frequency is being ramped. See each section below for specific splits.

##### 7.5.7.4.1 Duty Cycle Ramp Slope

For duty cycle ramping, the slope control is done in units of '%/period'. In other words, the change in duty cycle per switching period (switching frequency).

Since the duty cycle value is updated at the start of each period, the slowest and fastest ramps are dependent upon multiple factors:

- The % of change. 0% to 100% takes longer than 0% to 50%.
- Output resolution: 10-bit is slower than 8-bit due to more steps for the same %.
- Switching Frequency: This determines how often the duty cycle update occurs, since an update occurs at the start of a new period.

The desired value for a slope can be calculated from a desired change % change over time with the following equations:

$$\text{SLOPE\_CHANGE\_STEPS} = \text{ABS}(\text{SLOPE\_CHANGE\_DEC}) \times (2^{\text{NUM\_DC\_BITS}}) \quad (12)$$

$$\text{SLOPE\_VAL} = \text{SLOPE\_CHANGE\_STEPS} / (t \times f) \quad (13)$$

$$\text{SLOPE\_INT} = \text{TRUNCATE}(\text{SLOPE\_VAL}) \quad (14)$$

$$\text{SLOPE\_FRAC} = \text{ROUND}([\text{SLOPE\_VAL} - \text{TRUNCATE}(\text{SLOPE\_VAL})] \times 2^{\text{NUM\_FRAC\_BITS}}) \quad (15)$$

Where 'SLOPE\_CHANGE\_DECIMAL' is the percentage change desired (Ex: Start at 100% and end at 40% = 60%, or 0.6 as a decimal), but this equation is only used to get to the SLOPE\_CHANGE\_STEPS. The SLOPE\_CHANGE\_STEPS is used to list the total number of steps changed. For example, for a starting duty cycle of 102/1024 (~10%) to 768/1024 (~75%), then the total change is 768-102 = 666 steps.

'NUM\_DC\_BITS' is the number of bits of the duty cycle resolution (8 or 10 bits). 't' is the amount of time the change occurs in seconds. 'f' is the switching frequency in Hz or pulses per second.

As a quick example, if the user desires to ramp from 25% to 75% in 2 seconds, and has a switching frequency of 5KHz in 10-bit resolution mode, then slope change = 75%-25% = 50% (or 0.5),  $t = 2$ ,  $f = 5000$ , and NUM\_DC\_BITS = 10 (10-bit duty cycle mode). This gives a SLOPE\_VAL of  $(0.50 \times 2^{10}) / (2 \times 5000) = 0.2048$ . Since this slope value is low the maximum value of SLOPE\_SCALE = 3b000 (1.999999), this is the best choice to get the most resolution for the ramp, so SLOPE\_SCALE is selected to be 3b000, which gives 1 bit for integer, and 20 bits for fractional. The SLOPE\_INT = 0, and SLOPE\_FRAC =  $0.2048 \times 2^{20} = 214,748.3648$ . Round this to 214,748 decimal, or 0x346DC.

The slope scale setting selects how to divide the 21-bits between the integer and the fractional fields. A larger fractional field is needed for slower ramps.

**Table 7-34. Slope Scale Options (Duty Cycle)**

| SLOPE_SCALE | NUM_INT_BITS | NUM_FRAC_BITS | MINIMUM SLOPE VALUE | MAXIMUM SLOPE VALUE |
|-------------|--------------|---------------|---------------------|---------------------|
| 3b000       | 1 - [20]     | 20 - [19:0]   | 1/1,048,576         | 1.999999            |
| 3b001       | 2 - [20:19]  | 19 - [18:0]   | 1/524,288           | 3.999998            |
| 3b010       | 3 - [20:18]  | 18 - [17:0]   | 1/262,144           | 7.999996            |
| 3b011       | 4 - [20:17]  | 17 - [16:0]   | 1/131,072           | 15.999992           |

**Table 7-34. Slope Scale Options (Duty Cycle) (continued)**

| SLOPE_SCALE | NUM_INT_BITS | NUM_FRAC_BITS | MINIMUM SLOPE VALUE | MAXIMUM SLOPE VALUE |
|-------------|--------------|---------------|---------------------|---------------------|
| 3b100       | 5 - [20:16]  | 16 - [15:0]   | 1/65536             | 31.999985           |

**Table 7-35. Duty Cycle Slope Speed Limits (0% to 100%, SLOPE\_SCALE = 3b000)**

| SWITCHING<br>FREQUENCY<br>(Hz) | 8-BIT DUTY CYCLE      |                       | 10-BIT DUTY CYCLE     |                       |
|--------------------------------|-----------------------|-----------------------|-----------------------|-----------------------|
|                                | SLOWEST RAMP<br>(sec) | FASTEST RAMP<br>(sec) | SLOWEST RAMP<br>(sec) | FASTEST RAMP<br>(sec) |
| 19 (10-bit mode only)          | 14,128,182            | 6.7E+0                | 56,512,728            | 26.9E+0               |
| 50 (10-bit mode only)          | 5,368,709             | 2.6E+0                | 21,474,836            | 10.2E+0               |
| 76                             | 3,532,045             | 1.7E+0                | 14,128,182            | 6.7E+0                |
| 100                            | 2,684,355             | 1.3E+0                | 10,737,418            | 5.1E+0                |
| 250                            | 1,073,742             | 512.0E-3              | 4,294,967             | 2.0E+0                |
| 500                            | 536,871               | 256.0E-3              | 2,147,484             | 1.0E+0                |
| 1,000                          | 268,435               | 128.0E-3              | 1,073,742             | 512.0E-3              |
| 2,500                          | 107,374               | 51.2E-3               | 429,497               | 204.8E-3              |
| 5,000                          | 53,687                | 25.6E-3               | 214,748               | 102.4E-3              |
| 7,500                          | 35,791                | 17.1E-3               | 143,166               | 68.3E-3               |
| 10,000                         | 26,844                | 12.8E-3               | 107,374               | 51.2E-3               |
| 25,000                         | 10,737                | 5.1E-3                | 42,950                | 20.5E-3               |
| 39,000                         | 6,883                 | 3.3E-3                | 27,532                | 13.1E-3               |
| 50,000 (8-bit mode only)       | 5,369                 | 2.6E-3                | 21,475                | 10.2E-3               |
| 75,000 (8-bit mode only)       | 3,579                 | 1.7E-3                | 14,317                | 6.8E-3                |
| 100,000 (8-bit mode only)      | 2,684                 | 1.3E-3                | 10,737                | 5.1E-3                |

**Table 7-36. Duty Cycle Slope Speed Limits (0% to 100%, SLOPE\_SCALE = 3b001)**

| SWITCHING<br>FREQUENCY<br>(Hz) | 8-BIT DUTY CYCLE      |                       | 10-BIT DUTY CYCLE     |                       |
|--------------------------------|-----------------------|-----------------------|-----------------------|-----------------------|
|                                | SLOWEST RAMP<br>(sec) | FASTEST RAMP<br>(sec) | SLOWEST RAMP<br>(sec) | FASTEST RAMP<br>(sec) |
| 19 (10-bit mode only)          | 7,064,091             | 3.4E+0                | 28,256,364            | 13.5E+0               |
| 50 (10-bit mode only)          | 2,684,355             | 1.3E+0                | 10,737,418            | 5.1E+0                |
| 76                             | 1,766,023             | 842.1E-3              | 7,064,091             | 3.4E+0                |
| 100                            | 1,342,177             | 640.0E-3              | 5,368,709             | 2.6E+0                |
| 250                            | 536,871               | 256.0E-3              | 2,147,484             | 1.0E+0                |
| 500                            | 268,435               | 128.0E-3              | 1,073,742             | 512.0E-3              |
| 1,000                          | 134,218               | 64.0E-3               | 536,871               | 256.0E-3              |
| 2,500                          | 53,687                | 25.6E-3               | 214,748               | 102.4E-3              |
| 5,000                          | 26,844                | 12.8E-3               | 107,374               | 51.2E-3               |
| 7,500                          | 17,896                | 8.5E-3                | 71,583                | 34.1E-3               |
| 10,000                         | 13,422                | 6.4E-3                | 53,687                | 25.6E-3               |
| 25,000                         | 5,369                 | 2.6E-3                | 21,475                | 10.2E-3               |
| 39,000                         | 3,441                 | 1.6E-3                | 13,766                | 6.6E-3                |
| 50,000 (8-bit mode only)       | 2,684                 | 1.3E-3                | 10,737                | 5.1E-3                |
| 75,000 (8-bit mode only)       | 1,790                 | 853.3E-6              | 7,158                 | 3.4E-3                |
| 100,000 (8-bit mode only)      | 1,342                 | 640.0E-6              | 5,369                 | 2.6E-3                |

**Table 7-37. Duty Cycle Slope Speed Limits (0% to 100%, SLOPE\_SCALE = 3b010)**

| SWITCHING<br>FREQUENCY<br>(Hz) | 8-BIT DUTY CYCLE      |                       | 10-BIT DUTY CYCLE     |                       |
|--------------------------------|-----------------------|-----------------------|-----------------------|-----------------------|
|                                | SLOWEST RAMP<br>(sec) | FASTEST RAMP<br>(sec) | SLOWEST RAMP<br>(sec) | FASTEST RAMP<br>(sec) |
| 19 (10-bit mode only)          | 3,532,045             | 1.7E+0                | 14,128,182            | 6.7E+0                |
| 50 (10-bit mode only)          | 1,342,177             | 640.0E-3              | 5,368,709             | 2.6E+0                |
| 76                             | 883,011               | 421.1E-3              | 3,532,045             | 1.7E+0                |
| 100                            | 671,089               | 320.0E-3              | 2,684,355             | 1.3E+0                |
| 250                            | 268,435               | 128.0E-3              | 1,073,742             | 512.0E-3              |
| 500                            | 134,218               | 64.0E-3               | 536,871               | 256.0E-3              |
| 1,000                          | 67,109                | 32.0E-3               | 268,435               | 128.0E-3              |
| 2,500                          | 26,844                | 12.8E-3               | 107,374               | 51.2E-3               |
| 5,000                          | 13,422                | 6.4E-3                | 53,687                | 25.6E-3               |
| 7,500                          | 8,948                 | 4.3E-3                | 35,791                | 17.1E-3               |
| 10,000                         | 6,711                 | 3.2E-3                | 26,844                | 12.8E-3               |
| 25,000                         | 2,684                 | 1.3E-3                | 10,737                | 5.1E-3                |
| 39,000                         | 1,721                 | 820.5E-6              | 6,883                 | 3.3E-3                |
| 50,000 (8-bit mode only)       | 1,342                 | 640.0E-6              | 5,369                 | 2.6E-3                |
| 75,000 (8-bit mode only)       | 895                   | 426.7E-6              | 3,579                 | 1.7E-3                |
| 100,000 (8-bit mode only)      | 671                   | 320.0E-6              | 2,684                 | 1.3E-3                |

**Table 7-38. Duty Cycle Slope Speed Limits (0% to 100%, SLOPE\_SCALE = 3b011)**

| SWITCHING<br>FREQUENCY<br>(Hz) | 8-BIT DUTY CYCLE      |                       | 10-BIT DUTY CYCLE     |                       |
|--------------------------------|-----------------------|-----------------------|-----------------------|-----------------------|
|                                | SLOWEST RAMP<br>(sec) | FASTEST RAMP<br>(sec) | SLOWEST RAMP<br>(sec) | FASTEST RAMP<br>(sec) |
| 19 (10-bit mode only)          | 1,766,023             | 842.1E-3              | 7,064,091             | 3.4E+0                |
| 50 (10-bit mode only)          | 671,089               | 320.0E-3              | 2,684,355             | 1.3E+0                |
| 76                             | 441,506               | 210.5E-3              | 1,766,023             | 842.1E-3              |
| 100                            | 335,544               | 160.0E-3              | 1,342,177             | 640.0E-3              |
| 250                            | 134,218               | 64.0E-3               | 536,871               | 256.0E-3              |
| 500                            | 67,109                | 32.0E-3               | 268,435               | 128.0E-3              |
| 1,000                          | 33,554                | 16.0E-3               | 134,218               | 64.0E-3               |
| 2,500                          | 13,422                | 6.4E-3                | 53,687                | 25.6E-3               |
| 5,000                          | 6,711                 | 3.2E-3                | 26,844                | 12.8E-3               |
| 7,500                          | 4,474                 | 2.1E-3                | 17,896                | 8.5E-3                |
| 10,000                         | 3,355                 | 1.6E-3                | 13,422                | 6.4E-3                |
| 25,000                         | 1,342                 | 640.0E-6              | 5,369                 | 2.6E-3                |
| 39,000                         | 860                   | 410.3E-6              | 3,441                 | 1.6E-3                |
| 50,000 (8-bit mode only)       | 671                   | 320.0E-6              | 2,684                 | 1.3E-3                |
| 75,000 (8-bit mode only)       | 447                   | 213.3E-6              | 1,790                 | 853.3E-6              |
| 100,000 (8-bit mode only)      | 336                   | 160.0E-6              | 1,342                 | 640.0E-6              |

**Table 7-39. Duty Cycle Slope Speed Limits (0% to 100%, SLOPE\_SCALE = 3b100)**

| SWITCHING<br>FREQUENCY<br>(Hz) | 8-BIT DUTY CYCLE      |                       | 10-BIT DUTY CYCLE     |                       |
|--------------------------------|-----------------------|-----------------------|-----------------------|-----------------------|
|                                | SLOWEST RAMP<br>(sec) | FASTEST RAMP<br>(sec) | SLOWEST RAMP<br>(sec) | FASTEST RAMP<br>(sec) |
| 19 (10-bit mode only)          | 883,011               | 421.1E-3              | 3,532,045             | 1.7E+0                |
| 50 (10-bit mode only)          | 335,544               | 160.0E-3              | 1,342,177             | 640.0E-3              |
| 76                             | 220,753               | 105.3E-3              | 883,011               | 421.1E-3              |

**Table 7-39. Duty Cycle Slope Speed Limits (0% to 100%, SLOPE\_SCALE = 3b100) (continued)**

| SWITCHING<br>FREQUENCY<br>(Hz) | 8-BIT DUTY CYCLE      |                       | 10-BIT DUTY CYCLE     |                       |
|--------------------------------|-----------------------|-----------------------|-----------------------|-----------------------|
|                                | SLOWEST RAMP<br>(sec) | FASTEST RAMP<br>(sec) | SLOWEST RAMP<br>(sec) | FASTEST RAMP<br>(sec) |
| 100                            | 167,772               | 80.0E-3               | 671,089               | 320.0E-3              |
| 250                            | 67,109                | 32.0E-3               | 268,435               | 128.0E-3              |
| 500                            | 33,554                | 16.0E-3               | 134,218               | 64.0E-3               |
| 1,000                          | 16,777                | 8.0E-3                | 67,109                | 32.0E-3               |
| 2,500                          | 6,711                 | 3.2E-3                | 26,844                | 12.8E-3               |
| 5,000                          | 3,355                 | 1.6E-3                | 13,422                | 6.4E-3                |
| 7,500                          | 2,237                 | 1.1E-3                | 8,948                 | 4.3E-3                |
| 10,000                         | 1,678                 | 800.0E-6              | 6,711                 | 3.2E-3                |
| 25,000                         | 671                   | 320.0E-6              | 2,684                 | 1.3E-3                |
| 39,000                         | 430                   | 205.1E-6              | 1,721                 | 820.5E-6              |
| 50,000 (8-bit mode only)       | 336                   | 160.0E-6              | 1,342                 | 640.0E-6              |
| 75,000 (8-bit mode only)       | 224                   | 106.7E-6              | 895                   | 426.7E-6              |
| 100,000 (8-bit mode only)      | 168                   | 80.0E-6               | 671                   | 320.0E-6              |

#### 7.5.7.4.2 Switching Frequency Ramp Slope

For switching frequency ramping, the slope control is done in units of 'Hz/10μs'.

Since the period at which the updates happen is a fixed 10 μs, the minimum and maximum time to ramp is dependent only upon how big the change in frequency is.

The desired value for a slope can be calculated from a desired Hz/sec change over time with the following equations:

$$\text{SLOPE\_VAL} = (\text{ABS}(\text{Freq Change}) \times 10\text{E-}6) / t$$

$$\text{SLOPE\_INT} = \text{TRUNC}(\text{SLOPE\_VAL})$$

$$\text{SLOPE\_FRAC} = \text{ROUND}([\text{SLOPE\_VAL} - \text{TRUNC}(\text{SLOPE\_VAL})] \times 2^{\text{NUM\_FRAC\_BITS}})$$

Where 'Freq Change' is the change in frequency (Ex: Start at 200 Hz and end at 2 kHz = 1,800 kHz). 't' is the amount of time the change occurs over in seconds.

As a quick example, if the user desires to ramp from 250 Hz to 2 KHz in 4 seconds, and is running with SLOPE\_SCALE = 0b000, then freq change = 2000-250 = 1750, t = 4, NUM\_FRAC\_BITS = 16. This gives a SLOPE\_VAL of (1750 × 10E-6) / 4 = 0.07. The SLOPE\_INT = 0, and SLOPE\_FRAC = ROUND(0.07 × 2<sup>16</sup>) = 4588.

**Table 7-40. Slope Scale Options (Switching Frequency)**

| SLOPE_SCALE | NUM_INT_BITS | NUM_FRAC_BITS | MINIMUM SLOPE<br>VALUE<br>(Hz/10μs) | MAXIMUM<br>SLOPE VALUE<br>(Hz/10μs) | MINIMUM SLOPE<br>VALUE<br>(Hz/s) | MAXIMUM<br>SLOPE VALUE<br>(Hz/s) |
|-------------|--------------|---------------|-------------------------------------|-------------------------------------|----------------------------------|----------------------------------|
| 3b000       | 5 - [20:16]  | 16 - [15:0]   | 1/65536                             | 31.99998                            | 1.526                            | 3.2E+6                           |
| 3b001       | 9 - [20:12]  | 12 - [11:0]   | 1/4096                              | 511.99976                           | 24.414                           | 51.2E+6                          |
| 3b010       | 13 - [20:8]  | 8 - [7:0]     | 1/256                               | 8,191.99609                         | 390.625                          | 819.2E+6                         |
| 3b011       | 17 - [20:4]  | 4 - [3:0]     | 1/16                                | 131,071.93750                       | 6250                             | 13.1E+9                          |
| 3b100       | 21 - [20:0]  | 0 - None      | 1/1                                 | 2,097,151.00000                     | 100000                           | 209.7E+9                         |

#### 7.5.7.5 Automatic Deceleration and Stop Conditions

In addition to supporting up or down ramps, the TCAN5102-Q1 also supports the ability to do an entire acceleration and deceleration ramp with only a "start" signal from the upstream controller. This is set up by setting a counter that counts the number of pulses, and is used to determine when to begin the "stop/off" ramp.

There is also another counter for the maximum number of pulses allowed. This acts as a hard cut off for stopping PWM output when a total number of pulses have been generated during a PWM ramp. The stop feature is supported in all PWM modes (duty cycle, frequency, or static).

A user-selectable GPIO can be used to trigger a stop ramp or an immediate off condition. This is useful for a limit switch, or some sort of fault to immediately disable the PWM output.

#### 7.5.7.5.1 Automatic Deceleration Off-Ramp with Pulse Counting

The TCAN5102-Q1 can use pulse counters to time when to begin an off-ramp. The PWM\_PULSE\_STOP\_RAMP registers are used to set the compare value of a 32-bit counter. When a new ramp is started, the internal pulse counter is reset (can be read by reading PWM\_CUR\_PULSE) and begins to count up at the start of each pulse. If the PWM.AUTO\_STOP bit was set to 1 when the START bit was set, then once the PWM\_CUR\_PULSE matches the value in PWM\_PULSE\_STOP\_RAMP, the PWM IP writes a 1 to the PWM\_ACTION.STOP bit to automatically begin the OFF\_RAMP.

The PWM\_PULSE\_MAX register is then used to determine when to turn off the PWM output. Similar to the PWM\_PULSE\_STOP\_RAMP register, the value is compared against the PWM\_CUR\_PULSE register. When the PWM\_PULSE\_MAX matches the PWM\_CUR\_PULSE register, the PWM output is then turned off. Note that the PWM\_PULSE\_MAX register takes priority over the PWM\_PULSE\_STOP\_RAMP, so if the value of PWM\_PULSE\_MAX is less than the value of PWM\_PULSE\_STOP\_RAMP, no off-ramp takes place.

In [Figure 7-12](#), a typical automatic off-ramp is shown. Once the  $P_{E\_OFF}$  value is reached, the output continues until the PWM\_PULSE\_MAX value is reached and the output is turned off. In [Figure 7-13](#), the PWM\_PULSE\_MAX value is too close to the PWM\_PULSE\_STOP\_RAMP and does not allow for enough pulses for the off-ramp to complete, causing the output to get shut off in the middle of the off-ramp.

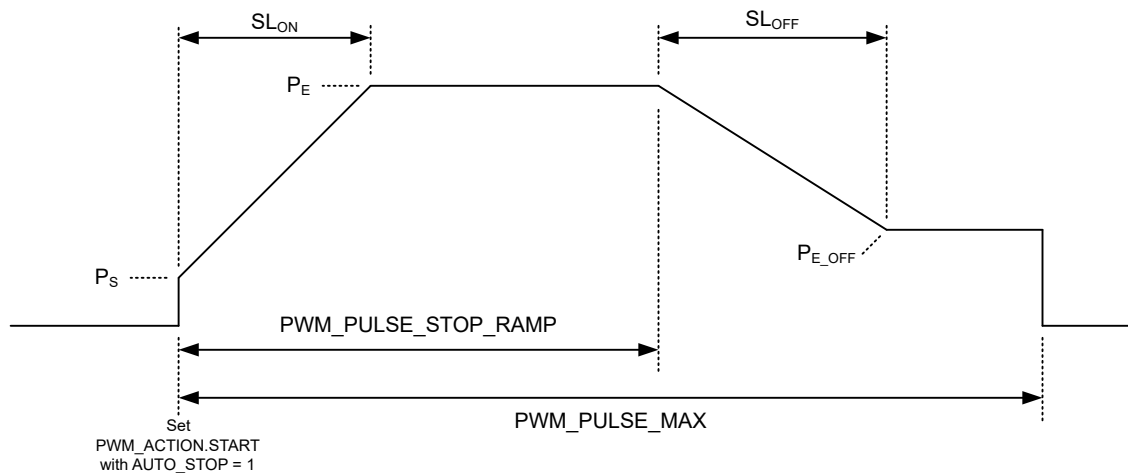
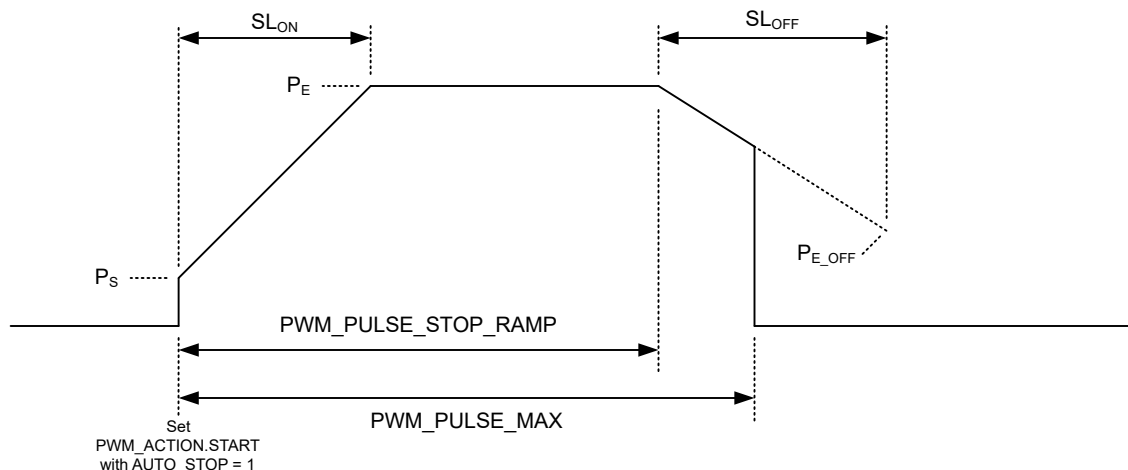


Figure 7-12. Auto-off Example 1

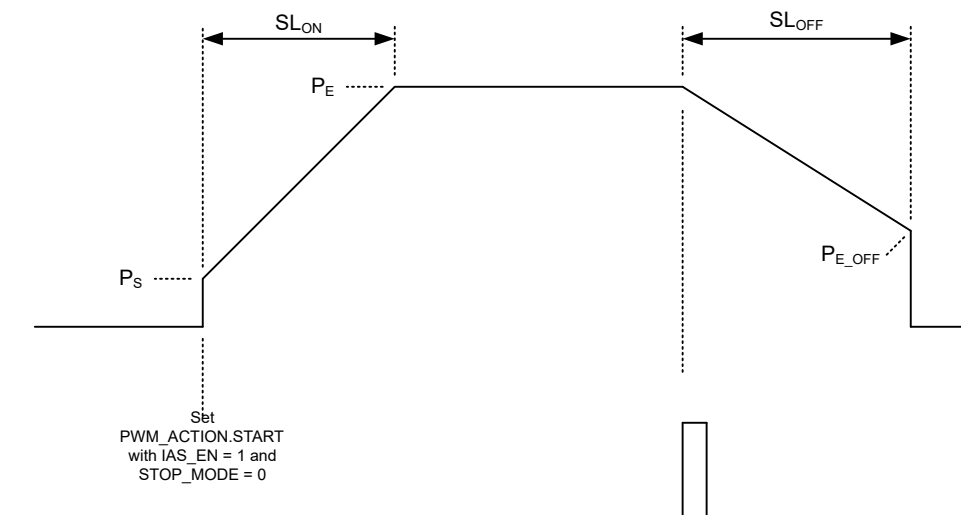


**Figure 7-13. Auto-off Example 2**

#### 7.5.7.5.2 Automatic Deceleration/Off-Ramp with GPIO Input

The TCAN5102-Q1 can use a GPIO input to trigger an off-ramp. The configuration for this feature is in the PWM\_IAS\_CTRL registers, allowing select of the GPIO to use as a trigger, along with polarity. Once the GPIO switches to the polarity selected, the behavior depends on the setting of PWM\_IAS\_CTRL.STOP\_MODE. If STOP\_MODE is set to 0, the PWM\_ACTION.STOP bit will be set, to start an off-ramp. If STOP\_MODE is set to 1, the PWM output is immediately stopped once the GPIO toggles to the active polarity.

In Figure 7-14, a typical automatic off-ramp is shown once the GPIO input toggles to the active high polarity. Once the  $P_{E\_OFF}$  value is reached, the output is turned off. In Figure 7-15 STOP\_MODE is set to 1, causing the output to be shut off immediately, without an off-ramp.



**Figure 7-14. GPIO-triggered off Example 1**

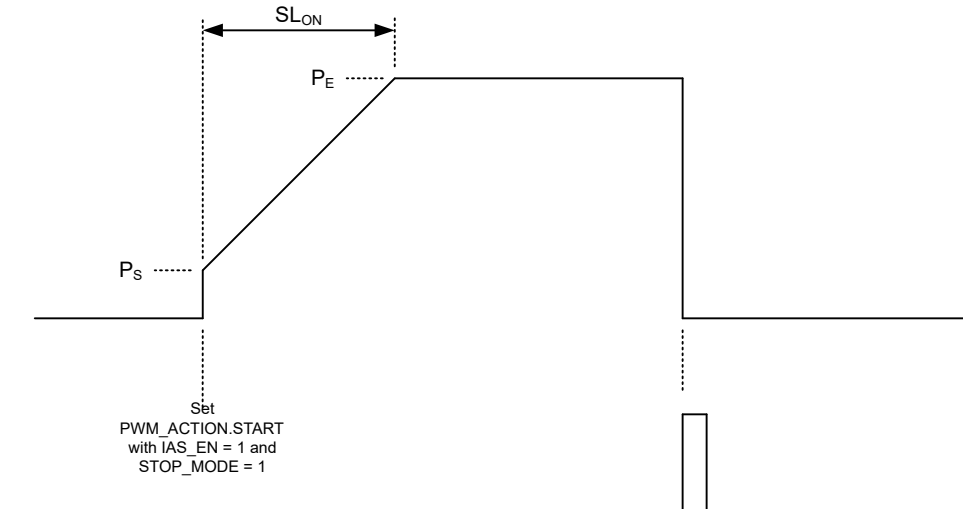


Figure 7-15. GPIO-triggered off Example 2

#### 7.5.7.6 Duty Cycle Ramp Example

This example is a basic example of configuring a duty cycle ramp of the following parameters.

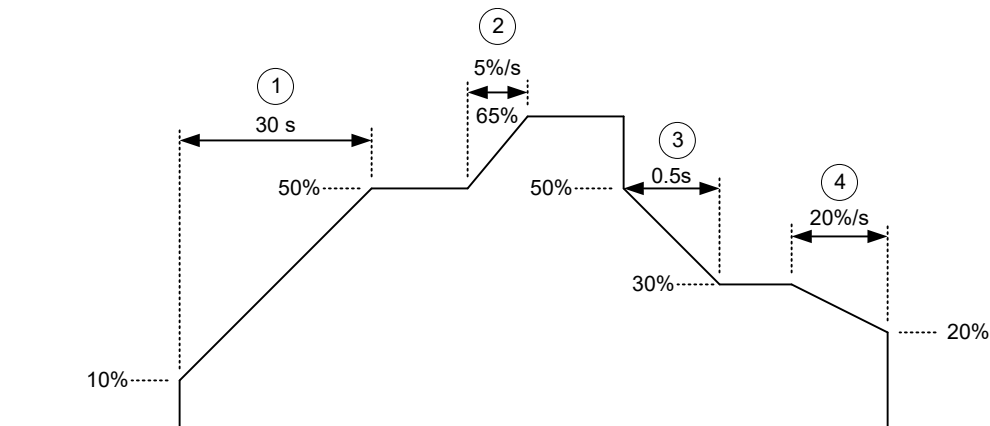


Figure 7-16. Example PWM DC Ramp

Table 7-41. Design Parameters and Assumptions

| PARAMETER           | VALUE               |
|---------------------|---------------------|
| Switching Frequency | 20 kHz              |
| Output Resolution   | 10-Bit              |
| Slope Scale         | 0b000 (1/1,048,576) |

The ramp profile shown above can be broken down into 4 individual trapezoidal ramps.

1. On-ramp: 10% to 50% in 30 s
2. On-ramp: 50% to 65% at a rate of 5%/second
3. On-ramp: A step function 50% to 30% in 500 ms
4. Off-ramp: Ramp to 20% at a rate of 20%/second then turn off the output.

The point of changing the slope values is to illustrate how the slope values can be calculated from either an absolute time, or a rate of change target. Since this example is made up of 4 individual ramps, each individual ramp is broken down. These 4 examples show a few different ways to arrive to the same result (different input units/combining some steps).

**Table 7-42. Ramp 1: 10 % to 50 % over 30 seconds**

| Step | Parameter         | Example                                                        | Description                                                                                            | Value         |
|------|-------------------|----------------------------------------------------------------|--------------------------------------------------------------------------------------------------------|---------------|
| 1    | Start Value       | $10\% \times 1024 = 102.4 \Rightarrow 102$                     | Convert the starting duty cycle to a 10-bit value                                                      | d102 or 0x066 |
| 2    | Stop Value        | $50\% \times 1024 = 512$                                       | Convert the ending duty cycle to a 10-bit value                                                        | d512 or 0x200 |
| 3    | Slope Calculation | $(512-102) = 410$                                              | Calculate the difference between end and start values                                                  |               |
|      |                   | $410 / 30 \text{ s} = 13.6666667$                              | Divide the difference over the time to ramp                                                            |               |
|      |                   | $13.6666667 / 20 \text{ kHz} = 0.00068333 \text{ count/cycle}$ | Calculate the change in duty cycle count value (10-bit value) per duty cycle (SLOPE_VAL)               |               |
|      |                   | $0.00068333 \times 1,048,576 = 716.52 \Rightarrow 717$         | Calculate the fractional value based on the current slope scale factor. Round to nearest integer value | d717 or 0x2CD |

**Table 7-43. Ramp 2: 50 % to 65 % with 5%/s**

| Step | Parameter         | Example                                                   | Description                                                                                            | Value          |
|------|-------------------|-----------------------------------------------------------|--------------------------------------------------------------------------------------------------------|----------------|
| 1    | Start Value       | -                                                         | Not needed since we'll set the 'use current PWM value' flag                                            | -              |
| 2    | Stop Value        | $65\% \times 1024 = 665.6 \Rightarrow 666$                | Convert the ending duty cycle to a 10-bit value                                                        | d666 or 0x29A  |
| 3    | Slope Calculation | $(5\% \times 1024) / 1 \text{ s} = 51.2 \text{ counts/s}$ | Convert a %/s slope to a count/second                                                                  |                |
|      |                   | $51.2 / 20 \text{ kHz} = 0.00256 \text{ counts/cycle}$    | Divide by switching frequency to get change of counts/cycle                                            |                |
|      |                   | $0.00256 \times 1,048,576 = 2684.35 \Rightarrow 2684$     | Calculate the fractional value based on the current slope scale factor. Round to nearest integer value | d2684 or 0xA7C |

**Table 7-44. Ramp 3: 50 % to 30 % over 0.5 seconds**

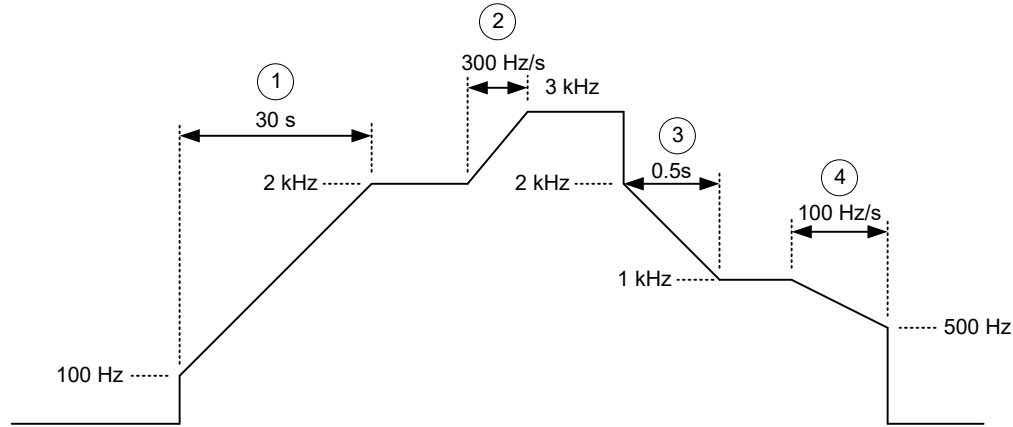
| Step | Parameter         | Example                                                 | Description                                                                                            | Value            |
|------|-------------------|---------------------------------------------------------|--------------------------------------------------------------------------------------------------------|------------------|
| 1    | Start Value       | $50\% \times 1024 = 512$                                | Convert the starting duty cycle to a 10-bit value                                                      | d512 or 0x200    |
| 2    | Stop Value        | $30\% \times 1024 = 307.2 \Rightarrow 307$              | Convert the ending duty cycle to a 10-bit value                                                        | d307 or 0x133    |
| 3    | Slope Calculation | $512 - 307 = 205$                                       | Calculate the difference between end and start values                                                  |                  |
|      |                   | $205 / 0.5 \text{ s} = 410 \text{ counts / s}$          | Divide the difference over the time to ramp                                                            |                  |
|      |                   | $410 / 20 \text{ kHz} = 0.0205 \text{ counts/cycle}$    | Convert counts / s to counts/cycle                                                                     |                  |
|      |                   | $0.0205 \times 1,048,576 = 21,498.8 \Rightarrow 21,499$ | Calculate the fractional value based on the current slope scale factor. Round to nearest integer value | d21499 or 0x53FB |

**Table 7-45. Ramp 4: x % to 20 % at 20 %/s**

| Step | Parameter         | Example                                                  | Description                                                                                            | Value            |
|------|-------------------|----------------------------------------------------------|--------------------------------------------------------------------------------------------------------|------------------|
| 1    | Start Value       | -                                                        | Not used on stop ramp. Current value is always used.                                                   | -                |
| 2    | Stop Value        | $20\% \times 1024 = 204.8$                               | Convert the ending duty cycle to a 10-bit value                                                        | d205 or 0x0CD    |
| 3    | Slope Calculation | $20 \text{ %/s} \times 1024 = 204.8 \text{ count/s}$     | Convert slope to 10-bit value                                                                          |                  |
|      |                   | $204.8 / 20 \text{ kHz} = 0.01024 \text{ count/cycle}$   | Convert slope to counts per cycle                                                                      |                  |
|      |                   | $0.01024 \times 1,048,576 = 10,737.4 \Rightarrow 10,737$ | Calculate the fractional value based on the current slope scale factor. Round to nearest integer value | d10737 or 0x29F1 |

### 7.5.7.7 Frequency Ramp Example

This example is a basic example of configuring a switching frequency ramp of the following parameters.



**Figure 7-17. Example PWM Switching Frequency Ramp**

**Table 7-46. Design Parameters and Assumptions**

| PARAMETER           | VALUE           |
|---------------------|-----------------|
| Switching Frequency | 20kHz           |
| Output Resolution   | 8-Bit           |
| Slope Scale         | 0b000 (1/65536) |

The ramp profile shown above can be broken down into 4 individual trapezoidal ramps.

1. On-ramp: 100Hz to 2kHz in 30 seconds
2. On-ramp: Ramp to 3kHz at a rate of 300Hz/s
3. On-ramp: A step function from 2kHz to 1kHz in 0.5 seconds
4. Off-ramp: Ramp to 500Hz at a rate of 100Hz/s

The point of changing the slope values is to illustrate how the slope values can be calculated from either an absolute time, or a rate of change target. Since this example is made up of 4 individual ramps, each individual ramp is broken down.

**Table 7-47. Ramp 1: 100Hz to 2kHz in 30 seconds**

| Step | Parameter         | Example                                                         | Description                                                     | Value          |
|------|-------------------|-----------------------------------------------------------------|-----------------------------------------------------------------|----------------|
| 1    | Start Value       | $40\text{MHz} / (100\text{Hz} \times 2^8) = 1562.5$             | Calculate the start frequency divisor                           |                |
|      |                   | $\text{TRUNC}(1562.5) = 1562$                                   | Calculate the integer portion of the divisor                    | d1562 or 0x61A |
|      |                   | $\text{ROUND}(0.5 \times 128) = 64$                             | Calculate the fractional portion of the divisor                 | d64 or 0x40    |
| 2    | Stop Value        | $40\text{ MHz} / (2\text{kHz} \times 2^8) = 78.125$             | Calculate the stop frequency divisor                            |                |
|      |                   | $\text{TRUNC}(78.125) = 78$                                     | Calculate the integer portion of the divisor                    | d78 or 0x4E    |
|      |                   | $\text{ROUND}(0.125 \times 128) = 16$                           | Calculate the fractional portion of the divisor                 | d16 or 0x10    |
| 3    | Slope Calculation | $(2\text{kHz} - 100\text{Hz}) / 30\text{ s} = 63.33\text{Hz/s}$ | Calculate the slope in Hz/s                                     |                |
|      |                   | $63.33 \times 10\mu\text{s} = 0.06333\text{Hz}/10\mu\text{s}$   | Convert the slope to Hz/10μs                                    |                |
|      |                   | $\text{TRUNC}(0.06333) = 0$                                     | Calculate the integer value of the slope                        | d0 or 0x00     |
|      |                   | $\text{ROUND}(0.06333 \times 65536) = 42$                       | Calculate the fractional value based on the slope scale factor. | d42 or 0x02A   |

**Table 7-48. Ramp 2: Ramp to 3kHz at a Rate of 300Hz/s**

| Step | Parameter         | Example                                                   | Description                                                     | Value         |
|------|-------------------|-----------------------------------------------------------|-----------------------------------------------------------------|---------------|
| 1    | Start Value       | -                                                         | Not needed since we'll set the 'use current PWM value' flag     | -             |
| 2    | Stop Value        | $40\text{MHz} / (3\text{kHz} \times 2^8) = 52.083$        | Calculate the stop frequency divisor                            |               |
|      |                   | $\text{TRUNC}(52.083) = 52$                               | Calculate the integer portion of the divisor                    | d52 or 0x34   |
|      |                   | $\text{ROUND}(0.083 \times 128) = 12$                     | Calculate the fractional portion of the divisor                 | d12 or 0x0C   |
| 3    | Slope Calculation | 300Hz/s (provided)                                        | Calculate the slope in Hz/s                                     |               |
|      |                   | $300 \times 10\mu\text{s} = 0.003\text{Hz}/10\mu\text{s}$ | Convert the slope to Hz/10μs                                    |               |
|      |                   | $\text{TRUNC}(0.003) = 0$                                 | Calculate the integer value of the slope                        | d0 or 0x00    |
|      |                   | $\text{ROUND}(0.003 \times 65536) = 197$                  | Calculate the fractional value based on the slope scale factor. | d197 or 0x0C5 |

**Table 7-49. Ramp 3: Ramp 2kHz to 1kHz over 0.5 seconds**

| Step | Parameter         | Example                                                              | Description                                                     | Value          |
|------|-------------------|----------------------------------------------------------------------|-----------------------------------------------------------------|----------------|
| 1    | Start Value       | $40\text{MHz} / (2\text{kHz} \times 2^8) = 78.125$                   | Calculate the start frequency divisor                           |                |
|      |                   | $\text{TRUNC}(78.125) = 78$                                          | Calculate the integer portion of the divisor                    | d78 or 0x4E    |
|      |                   | $\text{ROUND}(0.125 \times 128) = 16$                                | Calculate the fractional portion of the divisor                 | d16 or 0x10    |
| 2    | Stop Value        | $40\text{MHz} / (1\text{kHz} \times 2^8) = 156.25$                   | Calculate the stop frequency divisor                            |                |
|      |                   | $\text{TRUNC}(156.25) = 156$                                         | Calculate the integer portion of the divisor                    | d156 or 0x9C   |
|      |                   | $\text{ROUND}(0.25 \times 128) = 32$                                 | Calculate the fractional portion of the divisor                 | d32 or 0x20    |
| 3    | Slope Calculation | $(2\text{kHz} - 1\text{kHz}) / 0.5\text{s} = 2000\text{Hz}/\text{s}$ | Calculate the slope in Hz/s                                     |                |
|      |                   | $2000 \times 10\mu\text{s} = 0.02\text{Hz}/10\mu\text{s}$            | Convert the slope to Hz/10μs                                    |                |
|      |                   | $\text{TRUNC}(0.02) = 0$                                             | Calculate the integer value of the slope                        | d0 or 0x00     |
|      |                   | $\text{ROUND}(0.02 \times 65536) = 1311$                             | Calculate the fractional value based on the slope scale factor. | d1311 or 0x51F |

**Table 7-50. Ramp 4: Ramp to 500Hz at a rate of 100Hz/s**

| Step | Parameter         | Example                                                   | Description                                                     | Value         |
|------|-------------------|-----------------------------------------------------------|-----------------------------------------------------------------|---------------|
| 1    | Start Value       | -                                                         | Not used on stop ramp. Current value is always used.            | -             |
| 2    | Stop Value        | $40\text{MHz} / (500\text{Hz} \times 2^8) = 312.5$        | Calculate the stop frequency divisor                            |               |
|      |                   | $\text{TRUNC}(312.5) = 312$                               | Calculate the integer portion of the divisor                    | d312 or 0x138 |
|      |                   | $\text{ROUND}(0.5 \times 128) = 64$                       | Calculate the fractional portion of the divisor                 | d64 or 0x40   |
| 3    | Slope Calculation | 100Hz/s (provided)                                        | Calculate the slope in Hz/s                                     |               |
|      |                   | $100 \times 10\mu\text{s} = 0.001\text{Hz}/10\mu\text{s}$ | Convert the slope to Hz/10μs                                    |               |
|      |                   | $\text{TRUNC}(0.001) = 0$                                 | Calculate the integer value of the slope                        | d0 or 0x00    |
|      |                   | $\text{ROUND}(0.001 \times 65536) = 66$                   | Calculate the fractional value based on the slope scale factor. | d66 or 0x042  |

#### 7.5.7.8 Static On Example

This example is a basic example of configuring a simple static-on PWM waveform.

**Table 7-51. Design Parameters and Assumptions**

| PARAMETER           | VALUE |
|---------------------|-------|
| Switching Frequency | 10kHz |
| Output Resolution   | 8-Bit |
| Duty Cycle          | 20%   |

There are only a few registers that need to be configured for a simple static PWM waveform. The general process and registers needed are:

1. Configuration register must have mode set to static on, and the resolution set to 8-bit mode
2. For frequency, registers PWM\_END\_VAL\_CONST\_FRAC\_F, PWM\_END\_VAL\_MSB, and PWM\_END\_VAL\_LSB
3. For duty cycle, registers PWM\_CONST\_MSB, and PWM\_CONST\_LSB
4. Write to the start bit in the PWM\_ACTION register

**Table 7-52. Static PWM Calculations**

| Step | Parameter           | Example                                             | Description                                                   | Value        |
|------|---------------------|-----------------------------------------------------|---------------------------------------------------------------|--------------|
| 1    | Switching Frequency | $40\text{MHz} / (10\text{kHz} \times 2^8) = 15.625$ | Calculate the switching frequency divisor                     |              |
|      |                     | $\text{TRUNC}(15.625) = 15$                         | Calculate the integer portion of the divisor                  | d15 or 0x00F |
|      |                     | $\text{ROUND}(0.625 \times 128) = 80$               | Calculate the fractional portion of the divisor               | d80 or 0x50  |
| 2    | Duty Cycle          | $\text{ROUND}(20\% \times 256) = 51.2$              | Determine and get the nearest integer duty cycle count value. | d51 or 0x33  |

After the parameters are calculated, the parameters are written into the appropriate registers as shown in the example below.

**Table 7-53. Register Writes**

| Step | Register                 | Data (Hex) | Description                                                                                           |
|------|--------------------------|------------|-------------------------------------------------------------------------------------------------------|
| 1    | PWM_CTRL                 | 0x0F       | Sets INIT, the PWM mode to static on, and enables 8-bit duty cycle resolution                         |
| 2    | PWM_END_VAL_CONST_FRAC_F | 0x50       | Write the fractional divisor for switching frequency                                                  |
| 3    | PWM_END_VAL_MSB          | 0x00       | Write the MSB of the divisor for switching frequency                                                  |
| 4    | PWM_END_VAL_LSB          | 0x0F       | Write the LSB of the divisor for switching frequency                                                  |
| 5    | PWM_CONST_MSB            | 0x00       | Write the MSB for the duty cycle count. Since this example is 8-bit mode, this does not really matter |
| 6    | PWM_CONST_LSB            | 0x40       | Write the LSB for the duty cycle count                                                                |
| 7    | PWM_CTRL                 | 0x0E       | Disable the INIT bit to enable IP                                                                     |
| 8    | PWM_ACTION               | 0x02       | Turn on PWM output                                                                                    |

**Note**

The PWM module outputs must be muxed to a GPIO in the IO\_CFG\_0 register, otherwise PWM output are not on a pin.

## 7.6 Register Maps

The TCAN5102-Q1 has a comprehensive register set with 16-bit addressing. The register is broken down into several sections:

- Device Registers ([Section 7.6.1](#))
- SPI Registers ([Section 7.6.2](#) and [Section 7.6.3](#))
- UART Registers ([Section 7.6.4](#) and [Section 7.6.5](#))
- I2C Registers ([Section 7.6.6](#) and [Section 7.6.7](#))
- PWM0 Registers ([Section 7.6.8](#))
- PWM1 Registers ([Section 7.6.9](#))

### 7.6.1 DEVICE Registers

[Section 7.6.1](#) lists the memory-mapped registers for the Device registers. All register offset addresses not listed in [Section 7.6.1](#) should be considered as reserved locations and the register contents should not be modified.

**Table 7-54. DEVICE Registers**

| Offset       | Acronym                | Register Name                       | Section                          |
|--------------|------------------------|-------------------------------------|----------------------------------|
| 0h + formula | DEV_ID[y]              | Device ID                           | <a href="#">Section 7.6.1.1</a>  |
| 8h           | DEV_ID_REV             | Device Revision                     | <a href="#">Section 7.6.1.2</a>  |
| 9h           | DEV_CMD                | Device Commands                     | <a href="#">Section 7.6.1.3</a>  |
| Ah           | DEV_CFG_EN             | Device Configuration Enable         | <a href="#">Section 7.6.1.4</a>  |
| Bh           | DEV_CFG_BR             | Device CAN Baud rate                | <a href="#">Section 7.6.1.5</a>  |
| Ch           | DEV_CFG_ID_0           | Device CAN ID                       | <a href="#">Section 7.6.1.6</a>  |
| Dh           | DEV_CFG_ID_1           | Device CAN ID                       | <a href="#">Section 7.6.1.7</a>  |
| Eh           | DEV_CFG_ID_BCST_0      | Device CAN ID (Broadcast)           | <a href="#">Section 7.6.1.8</a>  |
| Fh           | DEV_CFG_ID_BCST_1      | Device CAN ID (Broadcast)           | <a href="#">Section 7.6.1.9</a>  |
| 10h          | DEV_CFG_ID_BCST_MASK_0 | Device CAN ID (Broadcast) Mask      | <a href="#">Section 7.6.1.10</a> |
| 11h          | DEV_CFG_ID_BCST_MASK_1 | Device CAN ID (Broadcast) Mask      | <a href="#">Section 7.6.1.11</a> |
| 12h          | DEV_CFG_NVM_PROG_CODE  | Device NVM Programming Key          | <a href="#">Section 7.6.1.12</a> |
| 13h          | DEV_CFG_NVM_PROG       | Device NVM Programming              | <a href="#">Section 7.6.1.13</a> |
| 1Dh          | INT_CFG                | GPIO Interrupt Output Configuration | <a href="#">Section 7.6.1.14</a> |
| 1Eh          | DEV_IE_0               | Device Interrupt Enable 0           | <a href="#">Section 7.6.1.15</a> |
| 1Fh          | DEV_IE_1               | Device Interrupt Enable 1           | <a href="#">Section 7.6.1.16</a> |
| 20h          | MRAM_IP_CFG            | MRAM and IP Configuration           | <a href="#">Section 7.6.1.17</a> |
| 21h          | IO_CFG_0               | GPIO Pin Mode Configuration         | <a href="#">Section 7.6.1.18</a> |
| 22h          | IO_CFG_1               | GPIO Pin Mode Configuration         | <a href="#">Section 7.6.1.19</a> |
| 23h          | IO_OE_0                | GPIO Output or Input Configuration  | <a href="#">Section 7.6.1.20</a> |
| 24h          | IO_OE_1                | GPIO Output or Input Configuration  | <a href="#">Section 7.6.1.21</a> |
| 25h          | IO_OD_0                | GPIO Open Drain Configuration       | <a href="#">Section 7.6.1.22</a> |
| 26h          | IO_OD_1                | GPIO Open Drain Configuration       | <a href="#">Section 7.6.1.23</a> |
| 27h          | IO_RE_0                | GPIO Resistor Enable                | <a href="#">Section 7.6.1.24</a> |
| 28h          | IO_RE_1                | GPIO Resistor Enable                | <a href="#">Section 7.6.1.25</a> |
| 29h          | IO_PU_0                | GPIO Pull-up Resistor Enable        | <a href="#">Section 7.6.1.26</a> |
| 2Ah          | IO_PU_1                | GPIO Pull-up Resistor Enable        | <a href="#">Section 7.6.1.27</a> |
| 2Bh          | IO_POL_0               | GPIO Polarity Inversion Enable      | <a href="#">Section 7.6.1.28</a> |
| 2Ch          | IO_POL_1               | GPIO Polarity Inversion Enable      | <a href="#">Section 7.6.1.29</a> |
| 2Dh          | IO_OUTPUT_0            | GPIO Output Values                  | <a href="#">Section 7.6.1.30</a> |
| 2Eh          | IO_OUTPUT_1            | GPIO Output Values                  | <a href="#">Section 7.6.1.31</a> |
| 2Fh          | IO_INPUT_0             | GPIO Input Values                   | <a href="#">Section 7.6.1.32</a> |
| 30h          | IO_INPUT_1             | GPIO Input Values                   | <a href="#">Section 7.6.1.33</a> |

**Table 7-54. DEVICE Registers (continued)**

| Offset | Acronym   | Register Name             | Section                          |
|--------|-----------|---------------------------|----------------------------------|
| 31h    | IR_STATUS | Interrupt Status          | <a href="#">Section 7.6.1.34</a> |
| 32h    | DEV_IR    | Device Interrupt Register | <a href="#">Section 7.6.1.35</a> |
| 33h    | SPI_IR    | SPI Interrupt Register    | <a href="#">Section 7.6.1.36</a> |
| 34h    | UART_IR   | UART Interrupt Register   | <a href="#">Section 7.6.1.37</a> |
| 35h    | I2C_IR    | I2C Interrupt Register    | <a href="#">Section 7.6.1.38</a> |
| 36h    | PWM0_IR   | PWM0 Interrupt Register   | <a href="#">Section 7.6.1.39</a> |
| 37h    | PWM1_IR   | PWM1 Interrupt Register   | <a href="#">Section 7.6.1.40</a> |

Complex bit access types are encoded to fit into small table cells. [Section 7.6.1](#) shows the codes that are used for access types in this section.

**Table 7-55. Device Access Type Codes**

| Access Type              | Code    | Description                                                                                                                                                                                                                                                                          |
|--------------------------|---------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Read Type                |         |                                                                                                                                                                                                                                                                                      |
| R                        | R       | Read                                                                                                                                                                                                                                                                                 |
| RH                       | R<br>H  | Read<br>Set or cleared by hardware                                                                                                                                                                                                                                                   |
| Write Type               |         |                                                                                                                                                                                                                                                                                      |
| W                        | W       | Write                                                                                                                                                                                                                                                                                |
| W1C                      | W<br>1C | Write<br>1 to clear                                                                                                                                                                                                                                                                  |
| W1S                      | W<br>1S | Write<br>1 to set                                                                                                                                                                                                                                                                    |
| WP                       | W<br>P  | Write<br>Requires privileged access                                                                                                                                                                                                                                                  |
| Reset or Default Value   |         |                                                                                                                                                                                                                                                                                      |
| -n                       |         | Value after reset or the default value                                                                                                                                                                                                                                               |
| Register Array Variables |         |                                                                                                                                                                                                                                                                                      |
| i,j,k,l,m,n              |         | When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula. |
| y                        |         | When this variable is used in a register name, an offset, or an address it refers to the value of a register array.                                                                                                                                                                  |

### 7.6.1.1 DEV\_ID[y] Register (Offset = 0h + formula) [Reset = 0032303135414354h]

DEV\_ID[y] is shown in [Figure 7-18](#) and described in [Table 7-56](#).

Return to the [Summary Table](#).

Device ID register containing part number in ASCII

Offset = 0h + (y \* 8h); where y = 0h to 7h

**Figure 7-18. DEV\_ID[y] Register**

|              |    |    |    |    |    |    |    |              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|--------------|----|----|----|----|----|----|----|
| 63           | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55           | 54 | 53 | 52 | 51 | 50 | 49 | 48 |
| DEV_ID_63:56 |    |    |    |    |    |    |    | DEV_ID_55:48 |    |    |    |    |    |    |    |
| R-0h         |    |    |    |    |    |    |    | R-32h        |    |    |    |    |    |    |    |
| 47           | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39           | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
| DEV_ID_40:47 |    |    |    |    |    |    |    | DEV_ID_39:32 |    |    |    |    |    |    |    |
| R-30h        |    |    |    |    |    |    |    | R-31h        |    |    |    |    |    |    |    |
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| DEV_ID_31:24 |    |    |    |    |    |    |    | DEV_ID_23:16 |    |    |    |    |    |    |    |
| R-35h        |    |    |    |    |    |    |    | R-41h        |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  | 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DEV_ID_15:8  |    |    |    |    |    |    |    | DEV_ID_7:0   |    |    |    |    |    |    |    |
| R-43h        |    |    |    |    |    |    |    | R-54h        |    |    |    |    |    |    |    |

**Table 7-56. DEV\_ID[y] Register Field Descriptions**

| Bit   | Field        | Type | Reset | Description    |
|-------|--------------|------|-------|----------------|
| 63-56 | DEV_ID_63:56 | R    | 0h    | Null character |
| 55-48 | DEV_ID_55:48 | R    | 32h   | ASCII for 2    |
| 47-40 | DEV_ID_40:47 | R    | 30h   | ASCII for 0    |
| 39-32 | DEV_ID_39:32 | R    | 31h   | ASCII for 1    |
| 31-24 | DEV_ID_31:24 | R    | 35h   | ASCII for 5    |
| 23-16 | DEV_ID_23:16 | R    | 41h   | ASCII for A    |
| 15-8  | DEV_ID_15:8  | R    | 43h   | ASCII for C    |
| 7-0   | DEV_ID_7:0   | R    | 54h   | ASCII for T    |

### 7.6.1.2 DEV\_ID\_REV Register (Offset = 8h) [Reset = 10h]

DEV\_ID\_REV is shown in [Figure 7-19](#) and described in [Table 7-57](#).

Return to the [Summary Table](#).

**Figure 7-19. DEV\_ID\_REV Register**

|              |   |   |   |              |   |   |   |
|--------------|---|---|---|--------------|---|---|---|
| 7            | 6 | 5 | 4 | 3            | 2 | 1 | 0 |
| DEV_ID_MAJOR |   |   |   | DEV_ID_MINOR |   |   |   |
| R-1h         |   |   |   | R-0h         |   |   |   |

**Table 7-57. DEV\_ID\_REV Register Field Descriptions**

| Bit | Field        | Type | Reset | Description              |
|-----|--------------|------|-------|--------------------------|
| 7-4 | DEV_ID_MAJOR | R    | 1h    | Major revision of device |
| 3-0 | DEV_ID_MINOR | R    | 0h    | Minor revision of device |

### 7.6.1.3 DEV\_CMD Register (Offset = 9h) [Reset = 00h]

DEV\_CMD is shown in [Figure 7-20](#) and described in [Table 7-58](#).

Return to the [Summary Table](#).

**Figure 7-20. DEV\_CMD Register**

| 7    | 6 | 5 | 4 | 3 | 2 | 1 | 0           |
|------|---|---|---|---|---|---|-------------|
| RSVD |   |   |   |   |   |   | GO_TO_SLEEP |
| R-0h |   |   |   |   |   |   | R/W1S-0h    |

**Table 7-58. DEV\_CMD Register Field Descriptions**

| Bit | Field       | Type  | Reset | Description                                                                      |
|-----|-------------|-------|-------|----------------------------------------------------------------------------------|
| 7-1 | RSVD        | R     | 0h    | Reserved                                                                         |
| 0   | GO_TO_SLEEP | R/W1S | 0h    | Set this bit to go to sleep mode<br>0h = Stay in Normal Mode<br>1h = Go to sleep |

#### 7.6.1.4 DEV\_CFG\_EN Register (Offset = Ah) [Reset = 00h]

DEV\_CFG\_EN is shown in [Figure 7-21](#) and described in [Table 7-59](#).

Return to the [Summary Table](#).

Unlocks the device configuration registers

**Figure 7-21. DEV\_CFG\_EN Register**

| 7    | 6 | 5 | 4 | 3 | 2               | 1          | 0          |
|------|---|---|---|---|-----------------|------------|------------|
| RSVD |   |   |   |   | CAN_ID_BCST_CCE | CAN_ID_CCE | CAN_BR_CCE |
| R-0h |   |   |   |   | R/W-0h          | R/W-0h     | R/W-0h     |

**Table 7-59. DEV\_CFG\_EN Register Field Descriptions**

| Bit | Field           | Type | Reset | Description                                                                                                                                   |
|-----|-----------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD            | R    | 0h    | Reserved                                                                                                                                      |
| 2   | CAN_ID_BCST_CCE | R/W  | 0h    | 0h = Changes to DEV_CFG_ID_BCST and DEV_CFG_ID_BCST_MASK are disabled<br>1h = Changes to DEV_CFG_ID_BCST and DEV_CFG_ID_BCST_MASK are enabled |
| 1   | CAN_ID_CCE      | R/W  | 0h    | 0h = Changes to DEV_CFG_ID are disabled<br>1h = Changes to DEV_CFG_ID are enabled                                                             |
| 0   | CAN_BR_CCE      | R/W  | 0h    | 0h = Changes to DEV_CFG_BR are disabled<br>1h = Changes to DEV_CFG_BR are enabled                                                             |

### 7.6.1.5 DEV\_CFG\_BR Register (Offset = Bh) [Reset = 10h]

DEV\_CFG\_BR is shown in [Figure 7-22](#) and described in [Table 7-60](#).

Return to the [Summary Table](#).

Device CAN Baud rate

#### Note

Change enable must be enabled in DEV\_CFG\_EN to make changes

**Figure 7-22. DEV\_CFG\_BR Register**

| 7    | 6 | 5 | 4          | 3    | 2       | 1 | 0 |
|------|---|---|------------|------|---------|---|---|
| RSVD |   |   | CAN_ACK_EN | RSVD | CAN_BR  |   |   |
| R-0h |   |   | R/WP-1h    | R-0h | R/WP-0h |   |   |

**Table 7-60. DEV\_CFG\_BR Register Field Descriptions**

| Bit | Field      | Type | Reset | Description                                                                                                                                                                                                                                      |
|-----|------------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD       | R    | 0h    | Reserved                                                                                                                                                                                                                                         |
| 4   | CAN_ACK_EN | R/WP | 1h    | Selects whether to enable the ACK bit when receiving a message (including broadcast messages). This is not recommended to be enabled for 2M or 5M, but is optional at speeds 1M or below.<br>0h = ACK bit is disabled<br>1h = ACK bit is enabled |
| 3   | RSVD       | R    | 0h    | Reserved                                                                                                                                                                                                                                         |
| 2-0 | CAN_BR     | R/WP | 0h    | Selects the CAN baud rate. The change occurs immediately after the responder sends the acknowledgement frame<br>0h = 250k<br>1h = 500k<br>2h = 1M<br>3h = 2M (ACK bit is disabled)<br>4h = 5M (ACK bit is disabled)                              |

#### 7.6.1.6 DEV\_CFG\_ID\_0 Register (Offset = Ch) [Reset = 00h]

DEV\_CFG\_ID\_0 is shown in [Figure 7-23](#) and described in [Table 7-61](#).

Return to the [Summary Table](#).

Device CAN ID that the responder will listen for

#### Note

Change enable must be enabled in DEV\_CFG\_EN to make changes

**Figure 7-23. DEV\_CFG\_ID\_0 Register**

| 7    | 6 | 5 | 4 | 3 | 2           | 1 | 0 |
|------|---|---|---|---|-------------|---|---|
| RSVD |   |   |   |   | CAN_ID_10:8 |   |   |
| R-0h |   |   |   |   | R/WP-0h     |   |   |

**Table 7-61. DEV\_CFG\_ID\_0 Register Field Descriptions**

| Bit | Field       | Type | Reset | Description                                                                                                                                                                                                                               |
|-----|-------------|------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD        | R    | 0h    | Reserved                                                                                                                                                                                                                                  |
| 2-0 | CAN_ID_10:8 | R/WP | 0h    | CAN ID[10:8] that is used for requests addressed to this specific responder.<br>It must match exactly for the responder to reply.<br>The LSB is ignored and always 0 for requests, and the responder will set the bit to 1 for responses. |

#### 7.6.1.7 DEV\_CFG\_ID\_1 Register (Offset = Dh) [Reset = 00h]

DEV\_CFG\_ID\_1 is shown in [Figure 7-24](#) and described in [Table 7-62](#).

Return to the [Summary Table](#).

Device CAN ID that the responder will listen for

#### Note

Change enable must be enabled in DEV\_CFG\_EN to make changes

**Figure 7-24. DEV\_CFG\_ID\_1 Register**

| 7          | 6 | 5 | 4 | 3 | 2 | 1 | 0    |
|------------|---|---|---|---|---|---|------|
| CAN_ID_7:1 |   |   |   |   |   |   | RSVD |
| R/WP-0h    |   |   |   |   |   |   | R-0h |

**Table 7-62. DEV\_CFG\_ID\_1 Register Field Descriptions**

| Bit | Field      | Type | Reset | Description                                                                                                                                                                                                                         |
|-----|------------|------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-1 | CAN_ID_7:1 | R/WP | 0h    | CAN ID that is used for requests addressed to this specific responder.<br>It must match exactly for the responder to reply.<br>The LSB is ignored and always 0 for requests, and the responder will set the bit to 1 for responses. |
| 0   | RSVD       | R    | 0h    | The LSB of the CAN ID is ignored, because 0 is used for requests, and 1 is used for response frames                                                                                                                                 |

### 7.6.1.8 DEV\_CFG\_ID\_BCST\_0 Register (Offset = Eh) [Reset = 00h]

DEV\_CFG\_ID\_BCST\_0 is shown in [Figure 7-25](#) and described in [Table 7-63](#).

Return to the [Summary Table](#).

Device broadcast CAN ID that the responder will listen for. The device will not send a response frame to broadcast messages

**Note**

Change enable must be enabled in DEV\_CFG\_EN to make changes

**Figure 7-25. DEV\_CFG\_ID\_BCST\_0 Register**

| 7    | 6 | 5 | 4 | 3 | 2                | 1 | 0 |
|------|---|---|---|---|------------------|---|---|
| RSVD |   |   |   |   | CAN_ID_BCST_10:8 |   |   |
| R-0h |   |   |   |   | R/WP-0h          |   |   |

**Table 7-63. DEV\_CFG\_ID\_BCST\_0 Register Field Descriptions**

| Bit | Field            | Type | Reset | Description                                                                                                                                                 |
|-----|------------------|------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD             | R    | 0h    | Reserved                                                                                                                                                    |
| 2-0 | CAN_ID_BCST_10:8 | R/WP | 0h    | CAN ID that is used for requests addressed to this specific responder.<br>This ID is used with the mask registers to determine which bits must match or not |

#### 7.6.1.9 DEV\_CFG\_ID\_BCST\_1 Register (Offset = Fh) [Reset = 00h]

DEV\_CFG\_ID\_BCST\_1 is shown in [Figure 7-26](#) and described in [Table 7-64](#).

Return to the [Summary Table](#).

Device broadcast CAN ID that the responder will listen for. The device will not send a response frame to broadcast messages

#### Note

Change enable must be enabled in DEV\_CFG\_EN to make changes

**Figure 7-26. DEV\_CFG\_ID\_BCST\_1 Register**

|                 |   |   |   |   |   |   |   |
|-----------------|---|---|---|---|---|---|---|
| 7               | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CAN_ID_BCST_7:0 |   |   |   |   |   |   |   |
| R/WP-0h         |   |   |   |   |   |   |   |

**Table 7-64. DEV\_CFG\_ID\_BCST\_1 Register Field Descriptions**

| Bit | Field           | Type | Reset | Description                                                                                                                                                 |
|-----|-----------------|------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | CAN_ID_BCST_7:0 | R/WP | 0h    | CAN ID that is used for requests addressed to this specific responder.<br>This ID is used with the mask registers to determine which bits must match or not |

#### 7.6.1.10 DEV\_CFG\_ID\_BCST\_MASK\_0 Register (Offset = 10h) [Reset = 00h]

DEV\_CFG\_ID\_BCST\_MASK\_0 is shown in [Figure 7-27](#) and described in [Table 7-65](#).

Return to the [Summary Table](#).

Device broadcast CAN ID mask that the responder will listen for. The device will not send a response frame to broadcast messages

#### Note

Change enable must be enabled in DEV\_CFG\_EN to make changes

**Figure 7-27. DEV\_CFG\_ID\_BCST\_MASK\_0 Register**

| 7    | 6 | 5 | 4 | 3                    | 2 | 1 | 0 |
|------|---|---|---|----------------------|---|---|---|
| RSVD |   |   |   | CAN_ID_BCST_MSK_10:8 |   |   |   |
| R-0h |   |   |   | R/WP-0h              |   |   |   |

**Table 7-65. DEV\_CFG\_ID\_BCST\_MASK\_0 Register Field Descriptions**

| Bit | Field                | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                         |
|-----|----------------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD                 | R    | 0h    | Reserved                                                                                                                                                                                                                                                                                                                                                                                            |
| 2-0 | CAN_ID_BCST_MSK_10:8 | R/WP | 0h    | The mask is used along with the CAN_ID_BCST to create several IDs that the broadcast ID can match.<br>A '0' means the bit must match the value provided in CAN_ID_BCST. A '1' value means that the bit is a 'do not care' and will accept either a 0 or a 1 for that bit in the CAN ID.<br>All '0's will make the broadcast ID require an exact match to what was entered in the CAN_ID_BCST field. |

### 7.6.1.11 DEV\_CFG\_ID\_BCST\_MASK\_1 Register (Offset = 11h) [Reset = 00h]

DEV\_CFG\_ID\_BCST\_MASK\_1 is shown in [Figure 7-28](#) and described in [Table 7-66](#).

Return to the [Summary Table](#).

Device broadcast CAN ID mask that the responder will listen for. The device will not send a response frame to broadcast messages

#### Note

Change enable must be enabled in DEV\_CFG\_EN to make changes

**Figure 7-28. DEV\_CFG\_ID\_BCST\_MASK\_1 Register**

|                     |   |   |   |   |   |   |   |
|---------------------|---|---|---|---|---|---|---|
| 7                   | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CAN_ID_BCST_MSK_7:0 |   |   |   |   |   |   |   |
| R/WP-0h             |   |   |   |   |   |   |   |

**Table 7-66. DEV\_CFG\_ID\_BCST\_MASK\_1 Register Field Descriptions**

| Bit | Field               | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                            |
|-----|---------------------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | CAN_ID_BCST_MSK_7:0 | R/WP | 0h    | The mask is used along with the CAN_ID_BCST to create several IDs that the broadcast ID can match.<br>A '0' means the bit must match the value provided in CAN_ID_BCST.<br>A '1' value means that the bit is a 'do not care' and will accept either a 0 or a 1 for that bit in the CAN ID.<br>All '0's will make the broadcast ID require an exact match to what was entered in the CAN_ID_BCST field. |

**7.6.1.12 DEV\_CFG\_NVM\_PROG\_CODE Register (Offset = 12h) [Reset = 00h]**

DEV\_CFG\_NVM\_PROG\_CODE is shown in [Figure 7-29](#) and described in [Table 7-67](#).

Return to the [Summary Table](#).

This register must be written to before attempting to program the EEPROM in order to unlock the process.

**Figure 7-29. DEV\_CFG\_NVM\_PROG\_CODE Register**

|             |   |   |   |   |   |   |   |
|-------------|---|---|---|---|---|---|---|
| 7           | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PROG_UNLOCK |   |   |   |   |   |   |   |
| R/W-0h      |   |   |   |   |   |   |   |

**Table 7-67. DEV\_CFG\_NVM\_PROG\_CODE Register Field Descriptions**

| Bit | Field       | Type | Reset | Description                                                                                                                 |
|-----|-------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------|
| 7-0 | PROG_UNLOCK | R/W  | 0h    | To flash the EEPROM, this field must be written as 0x78.<br>Any other value will result in the flash request being ignored. |

### 7.6.1.13 DEV\_CFG\_NVM\_PROG Register (Offset = 13h) [Reset = 00h]

DEV\_CFG\_NVM\_PROG is shown in [Table 7-68](#).

Return to the [Summary Table](#).

Save existing device configuration into the device's one time programmable memory.

**Table 7-68. DEV\_CFG\_NVM\_PROG Register Field Descriptions**

| Bit | Field          | Type   | Reset | Description                                                                                                                                                                                                                                                                                                                         |
|-----|----------------|--------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 6   | PROG_FAIL      | RH     | 0h    | Status flag to report if there was a fault with the EEPROM program process.<br>This flag should be checked after attempting to program the EEPROM.<br>If there is a failure of any sort, this bit will be set, and the user should attempt to program the EEPROM again.<br>0h = EEPROM is not faulted<br>1h = EEPROM program failed |
| 5   | RSVD           | R      | 0h    | Reserved                                                                                                                                                                                                                                                                                                                            |
| 4   | EEPROM_FLASHED | RH     | 0h    | Status flag to query if the EEPROM has been flashed already<br>0h = EEPROM has NOT been programmed<br>1h = EEPROM has been programmed                                                                                                                                                                                               |
| 3   | RSVD           | R      | 0h    | Reserved                                                                                                                                                                                                                                                                                                                            |
| 2   | LOCK_ID        | RH/W1S | 0h    | If this is set, the device will permanently disable the CAN_ID_CCE bit so that the CAN IDs and mask can not be changed.<br>0h = CAN IDs and mask registers can be updated (both in register and EEPROM)<br>1h = CAN IDs and mask registers can NOT be updated (both in register and EEPROM)                                         |
| 1   | LOCK_BR        | RH/W1S | 0h    | If this is set, the device will permanently disable the CAN_BR_CCE bit so that the CAN baud rate can not be changed.<br>0h = CAN data rate registers can be updated (both in register and EEPROM)<br>1h = CAN data rate registers can NOT be updated (both in register and EEPROM)                                                  |
| 0   | PROG           | RH/W1S | 0h    | If set with the flash code being set correctly, then the EEPROM will be programmed.<br>The bit will read back 1 until the flash procedure is complete.<br>0h = Not actively programming EEPROM/program is complete<br>1h = Actively programming EEPROM                                                                              |

#### 7.6.1.14 INT\_CFG Register (Offset = 1Dh) [Reset = 00h]

INT\_CFG is shown in [Figure 7-30](#) and described in [Table 7-69](#).

Return to the [Summary Table](#).

GPIO interrupt output

**Figure 7-30. INT\_CFG Register**

| 7    | 6        | 5 | 4       | 3    | 2        | 1 | 0       |
|------|----------|---|---------|------|----------|---|---------|
| RSVD | INT1_SEL |   | INT1_EN | RSVD | INT0_SEL |   | INT0_EN |
| R-0h | R/W-0h   |   | R/W-0h  | R-0h | R/W-0h   |   | R/W-0h  |

**Table 7-69. INT\_CFG Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                  |
|-----|----------|------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | RSVD     | R    | 0h    | Reserved                                                                                                                                                                                     |
| 6-5 | INT1_SEL | R/W  | 0h    | Selects the GPIO pin to be used as an interrupt output<br>0h = GPIO1 as active-low INT1<br>1h = GPIO8 as active-low INT1<br>2h = GPIO1 as active-high INT1<br>3h = GPIO8 as active-high INT1 |
| 4   | INT1_EN  | R/W  | 0h    | Used to enable the selected GPIO as an interrupt output.<br>0h = Normal function (as configured)<br>1h = Interrupt function (IO_CFG must be set to GPIO function)                            |
| 3   | RSVD     | R    | 0h    | Reserved                                                                                                                                                                                     |
| 2-1 | INT0_SEL | R/W  | 0h    | Selects the GPIO pin to be used as an interrupt output<br>0h = GPIO0 as active-low INT0<br>1h = GPIO7 as active-low INT0<br>2h = GPIO0 as active-high INT0<br>3h = GPIO7 as active-high INT0 |
| 0   | INT0_EN  | R/W  | 0h    | Used to enable the selected GPIO as an interrupt output.<br>0h = Normal function (as configured)<br>1h = Interrupt function (IO_CFG must be set to GPIO function)                            |

### 7.6.1.15 DEV\_IE\_0 Register (Offset = 1Eh) [Reset = 00h]

DEV\_IE\_0 is shown in [Figure 7-31](#) and described in [Table 7-70](#).

Return to the [Summary Table](#).

Interrupt enable bits for enabling certain interrupts for the INT0 pin. Interrupts that are enabled with be signaled on the INT0 pin. Note that INT0 functionality must be enabled in the INT\_CFG register to see output on the pin.

**Figure 7-31. DEV\_IE\_0 Register**

| 7    | 6 | 5 | 4 | 3 | 2 | 1      | 0    |
|------|---|---|---|---|---|--------|------|
| RSVD |   |   |   |   |   | CANWK  | RSVD |
| R-0h |   |   |   |   |   | R/W-0h | R-0h |

**Table 7-70. DEV\_IE\_0 Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                                                                                                                                                        |
|-----|-------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-2 | RSVD  | R    | 0h    | Reserved                                                                                                                                                                                                                           |
| 1   | CANWK | R/W  | 0h    | CAN Wakeup<br>The device has come out of sleep mode due to the CAN bus<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output. |
| 0   | RSVD  | R    | 0h    | Reserved                                                                                                                                                                                                                           |

#### 7.6.1.16 DEV\_IE\_1 Register (Offset = 1Fh) [Reset = 00h]

DEV\_IE\_1 is shown in [Figure 7-32](#) and described in [Table 7-71](#).

Return to the [Summary Table](#).

Interrupt enable bits for enabling certain interrupts for the INT0 pin. Interrupts that are enabled with be signaled on the INT1 pin. Note that INT0 functionality must be enabled in the INT\_CFG register to see output on the pin.

**Figure 7-32. DEV\_IE\_1 Register**

| 7    | 6 | 5 | 4 | 3 | 2 | 1      | 0    |
|------|---|---|---|---|---|--------|------|
| RSVD |   |   |   |   |   | CANWK  | RSVD |
| R-0h |   |   |   |   |   | R/W-0h | R-0h |

**Table 7-71. DEV\_IE\_1 Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                                                                                                                                                        |
|-----|-------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-2 | RSVD  | R    | 0h    | Reserved                                                                                                                                                                                                                           |
| 1   | CANWK | R/W  | 0h    | CAN Wakeup<br>The device has come out of sleep mode due to the CAN bus<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output. |
| 0   | RSVD  | R    | 0h    | Reserved                                                                                                                                                                                                                           |

### 7.6.1.17 MRAM\_IP\_CFG Register (Offset = 20h) [Reset = 00h]

MRAM\_IP\_CFG is shown in [Figure 7-33](#) and described in [Table 7-72](#).

Return to the [Summary Table](#).

MRAM and IP configuration. This register is used to enable the various serial interfaces as well as allocate a portion of the MRAM to them.

**Figure 7-33. MRAM\_IP\_CFG Register**

| 7    | 6 | 5 | 4             | 3          | 2 | 1 | 0 |
|------|---|---|---------------|------------|---|---|---|
| RSVD |   |   | GPIO_OUT_SYNC | MRAM_IP_EN |   |   |   |
| R-0h |   |   | R/W-0h        | R/W-0h     |   |   |   |

**Table 7-72. MRAM\_IP\_CFG Register Field Descriptions**

| Bit | Field         | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-----|---------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD          | R    | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 4   | GPIO_OUT_SYNC | R/W  | 0h    | Synchronizes the GPIO output value updates by holding off on updates until the 2nd GPIO register is written.<br>This requires ALL output GPIO changes to write to 0x002D and 0x002E, and once the 0x002E write is completed, then all GPIOs will update.<br>0h = GPIOs are updated as the bytes are written in<br>1h = GPIOs are updated once 0x2E is written to                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 3-0 | MRAM_IP_EN    | R/W  | 0h    | MRAM and IP enable.<br>These bits will enable the selected IP if the serial interface is allocated any memory (>0%).<br>Do not change this value while the enabled IP is active, undefined behavior can occur (will be changed in a future version)<br><br><div style="text-align: center;"><b>Note</b></div> <div style="text-align: center;">The GPIOs for the intended IP needs to be set as special function or the IP will not be enabled</div><br>0h = All disabled/0%<br>1h = SPI 0%, UART 100%, I2C 0%<br>2h = SPI 25%, UART 75%, I2C 0%<br>3h = SPI 50%, UART 50%, I2C 0%<br>4h = SPI 75%, UART 25%, I2C 0%<br>5h = SPI 100%, UART 0%, I2C 0%<br>6h = SPI 0%, UART 0%, I2C 100%<br>7h = SPI 0%, UART 25%, I2C 75%<br>8h = SPI 0%, UART 50%, I2C 50%<br>9h = SPI 0%, UART 75%, I2C 25%<br>Ah = SPI 25%, UART 0%, I2C 75%<br>Bh = SPI 50%, UART 0%, I2C 50%<br>Ch = SPI 75%, UART 0%, I2C 25% |

### 7.6.1.18 IO\_CFG\_0 Register (Offset = 21h) [Reset = 00h]

IO\_CFG\_0 is shown in [Figure 7-34](#) and described in [Table 7-73](#).

Return to the [Summary Table](#).

GPIO pin mode configuration

**Figure 7-34. IO\_CFG\_0 Register**

| 7         | 6         | 5         | 4         | 3         | 2         | 1         | 0         |
|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| GPIO7_CFG | GPIO6_CFG | GPIO5_CFG | GPIO4_CFG | GPIO3_CFG | GPIO2_CFG | GPIO1_CFG | GPIO0_CFG |
| R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h    |

**Table 7-73. IO\_CFG\_0 Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                            |
|-----|-----------|------|-------|--------------------------------------------------------|
| 7   | GPIO7_CFG | R/W  | 0h    | 0h = GPIO function<br>1h = Special function (UART TXD) |
| 6   | GPIO6_CFG | R/W  | 0h    | 0h = GPIO function<br>1h = Special function (SPI PICO) |
| 5   | GPIO5_CFG | R/W  | 0h    | 0h = GPIO function<br>1h = Special function (SPI POCI) |
| 4   | GPIO4_CFG | R/W  | 0h    | 0h = GPIO function<br>1h = Special function (SPI SCK)  |
| 3   | GPIO3_CFG | R/W  | 0h    | 0h = GPIO function<br>1h = Special function (SPI CS0)  |
| 2   | GPIO2_CFG | R/W  | 0h    | 0h = GPIO function<br>1h = Special function (SPI CS1)  |
| 1   | GPIO1_CFG | R/W  | 0h    | 0h = GPIO function<br>1h = Special function (SPI CS2)  |
| 0   | GPIO0_CFG | R/W  | 0h    | 0h = GPIO function<br>1h = Special function (SPI CS3)  |

### 7.6.1.19 IO\_CFG\_1 Register (Offset = 22h) [Reset = 00h]

IO\_CFG\_1 is shown in [Figure 7-35](#) and described in [Table 7-74](#).

Return to the [Summary Table](#).

GPIO pin mode configuration

**Figure 7-35. IO\_CFG\_1 Register**

| 7    | 6 | 5 | 4          | 3          | 2          | 1         | 0         |
|------|---|---|------------|------------|------------|-----------|-----------|
| RSVD |   |   | GPIO12_CFG | GPIO11_CFG | GPIO10_CFG | GPIO9_CFG | GPIO8_CFG |
| R-0h |   |   | R/W-0h     | R/W-0h     | R/W-0h     | R/W-0h    | R/W-0h    |

**Table 7-74. IO\_CFG\_1 Register Field Descriptions**

| Bit | Field      | Type | Reset | Description                                                                        |
|-----|------------|------|-------|------------------------------------------------------------------------------------|
| 7-5 | RSVD       | R    | 0h    |                                                                                    |
| 4   | GPIO12_CFG | R/W  | 0h    | 0h = GPIO function<br>1h = Special function (SPI CS7 or PWM1 if PWM1 is enabled)   |
| 3   | GPIO11_CFG | R/W  | 0h    | 0h = GPIO function<br>1h = Special function (SPI CS6 or PWM0 if PWM0 is enabled)   |
| 2   | GPIO10_CFG | R/W  | 0h    | 0h = GPIO function<br>1h = Special function (SPI CS5 or I2C SDA if I2C is enabled) |
| 1   | GPIO9_CFG  | R/W  | 0h    | 0h = GPIO function<br>1h = Special function (SPI CS4 or I2C SCL if I2C is enabled) |
| 0   | GPIO8_CFG  | R/W  | 0h    | 0h = GPIO function<br>1h = Special function (UART RXD)                             |

### 7.6.1.20 IO\_OE\_0 Register (Offset = 23h) [Reset = 00h]

IO\_OE\_0 is shown in [Figure 7-36](#) and described in [Table 7-75](#).

Return to the [Summary Table](#).

GPIO output or input configuration

**Figure 7-36. IO\_OE\_0 Register**

| 7        | 6        | 5        | 4        | 3        | 2        | 1        | 0        |
|----------|----------|----------|----------|----------|----------|----------|----------|
| GPIO7_OE | GPIO6_OE | GPIO5_OE | GPIO4_OE | GPIO3_OE | GPIO2_OE | GPIO1_OE | GPIO0_OE |
| R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   |

**Table 7-75. IO\_OE\_0 Register Field Descriptions**

| Bit | Field    | Type | Reset | Description               |
|-----|----------|------|-------|---------------------------|
| 7   | GPIO7_OE | R/W  | 0h    | 0h = Input<br>1h = Output |
| 6   | GPIO6_OE | R/W  | 0h    | 0h = Input<br>1h = Output |
| 5   | GPIO5_OE | R/W  | 0h    | 0h = Input<br>1h = Output |
| 4   | GPIO4_OE | R/W  | 0h    | 0h = Input<br>1h = Output |
| 3   | GPIO3_OE | R/W  | 0h    | 0h = Input<br>1h = Output |
| 2   | GPIO2_OE | R/W  | 0h    | 0h = Input<br>1h = Output |
| 1   | GPIO1_OE | R/W  | 0h    | 0h = Input<br>1h = Output |
| 0   | GPIO0_OE | R/W  | 0h    | 0h = Input<br>1h = Output |

### 7.6.1.21 IO\_OE\_1 Register (Offset = 24h) [Reset = 00h]

IO\_OE\_1 is shown in [Figure 7-37](#) and described in [Table 7-76](#).

Return to the [Summary Table](#).

GPIO output or input configuration

**Figure 7-37. IO\_OE\_1 Register**

| 7    | 6 | 5 | 4         | 3         | 2         | 1        | 0        |
|------|---|---|-----------|-----------|-----------|----------|----------|
| RSVD |   |   | GPIO12_OE | GPIO11_OE | GPIO10_OE | GPIO9_OE | GPIO8_OE |
| R-0h |   |   | R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h   | R/W-0h   |

**Table 7-76. IO\_OE\_1 Register Field Descriptions**

| Bit | Field     | Type | Reset | Description               |
|-----|-----------|------|-------|---------------------------|
| 7-5 | RSVD      | R    | 0h    |                           |
| 4   | GPIO12_OE | R/W  | 0h    | 0h = Input<br>1h = Output |
| 3   | GPIO11_OE | R/W  | 0h    | 0h = Input<br>1h = Output |
| 2   | GPIO10_OE | R/W  | 0h    | 0h = Input<br>1h = Output |
| 1   | GPIO9_OE  | R/W  | 0h    | 0h = Input<br>1h = Output |
| 0   | GPIO8_OE  | R/W  | 0h    | 0h = Input<br>1h = Output |

### 7.6.1.22 IO\_OD\_0 Register (Offset = 25h) [Reset = 00h]

IO\_OD\_0 is shown in [Figure 7-38](#) and described in [Table 7-77](#).

Return to the [Summary Table](#).

GPIO open drain configuration

**Figure 7-38. IO\_OD\_0 Register**

| 7        | 6        | 5        | 4        | 3        | 2        | 1        | 0        |
|----------|----------|----------|----------|----------|----------|----------|----------|
| GPIO7_OD | GPIO6_OD | GPIO5_OD | GPIO4_OD | GPIO3_OD | GPIO2_OD | GPIO1_OD | GPIO0_OD |
| R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   |

**Table 7-77. IO\_OD\_0 Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                 |
|-----|----------|------|-------|---------------------------------------------|
| 7   | GPIO7_OD | R/W  | 0h    | 0h = Push-pull mode<br>1h = Open-drain mode |
| 6   | GPIO6_OD | R/W  | 0h    | 0h = Push-pull mode<br>1h = Open-drain mode |
| 5   | GPIO5_OD | R/W  | 0h    | 0h = Push-pull mode<br>1h = Open-drain mode |
| 4   | GPIO4_OD | R/W  | 0h    | 0h = Push-pull mode<br>1h = Open-drain mode |
| 3   | GPIO3_OD | R/W  | 0h    | 0h = Push-pull mode<br>1h = Open-drain mode |
| 2   | GPIO2_OD | R/W  | 0h    | 0h = Push-pull mode<br>1h = Open-drain mode |
| 1   | GPIO1_OD | R/W  | 0h    | 0h = Push-pull mode<br>1h = Open-drain mode |
| 0   | GPIO0_OD | R/W  | 0h    | 0h = Push-pull mode<br>1h = Open-drain mode |

### 7.6.1.23 IO\_OD\_1 Register (Offset = 26h) [Reset = 00h]

IO\_OD\_1 is shown in [Figure 7-39](#) and described in [Table 7-78](#).

Return to the [Summary Table](#).

GPIO open drain configuration

**Figure 7-39. IO\_OD\_1 Register**

| 7    | 6 | 5 | 4         | 3         | 2         | 1        | 0        |
|------|---|---|-----------|-----------|-----------|----------|----------|
| RSVD |   |   | GPIO12_OD | GPIO11_OD | GPIO10_OD | GPIO9_OD | GPIO8_OD |
| R-0h |   |   | R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h   | R/W-0h   |

**Table 7-78. IO\_OD\_1 Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                 |
|-----|-----------|------|-------|---------------------------------------------|
| 7-5 | RSVD      | R    | 0h    |                                             |
| 4   | GPIO12_OD | R/W  | 0h    | 0h = Push-pull mode<br>1h = Open-drain mode |
| 3   | GPIO11_OD | R/W  | 0h    | 0h = Push-pull mode<br>1h = Open-drain mode |
| 2   | GPIO10_OD | R/W  | 0h    | 0h = Push-pull mode<br>1h = Open-drain mode |
| 1   | GPIO9_OD  | R/W  | 0h    | 0h = Push-pull mode<br>1h = Open-drain mode |
| 0   | GPIO8_OD  | R/W  | 0h    | 0h = Push-pull mode<br>1h = Open-drain mode |

### 7.6.1.24 IO\_RE\_0 Register (Offset = 27h) [Reset = 00h]

IO\_RE\_0 is shown in [Figure 7-40](#) and described in [Table 7-79](#).

Return to the [Summary Table](#).

GPIO resistor enable

**Figure 7-40. IO\_RE\_0 Register**

| 7        | 6        | 5        | 4        | 3        | 2        | 1        | 0        |
|----------|----------|----------|----------|----------|----------|----------|----------|
| GPIO7_RE | GPIO6_RE | GPIO5_RE | GPIO4_RE | GPIO3_RE | GPIO2_RE | GPIO1_RE | GPIO0_RE |
| R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   |

**Table 7-79. IO\_RE\_0 Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                         |
|-----|----------|------|-------|-----------------------------------------------------------------------------------------------------|
| 7   | GPIO7_RE | R/W  | 0h    | 0h = No bias resistor is connected to pin<br>1h = Bias resistor is connected to pin according IO_PU |
| 6   | GPIO6_RE | R/W  | 0h    | 0h = No bias resistor is connected to pin<br>1h = Bias resistor is connected to pin according IO_PU |
| 5   | GPIO5_RE | R/W  | 0h    | 0h = No bias resistor is connected to pin<br>1h = Bias resistor is connected to pin according IO_PU |
| 4   | GPIO4_RE | R/W  | 0h    | 0h = No bias resistor is connected to pin<br>1h = Bias resistor is connected to pin according IO_PU |
| 3   | GPIO3_RE | R/W  | 0h    | 0h = No bias resistor is connected to pin<br>1h = Bias resistor is connected to pin according IO_PU |
| 2   | GPIO2_RE | R/W  | 0h    | 0h = No bias resistor is connected to pin<br>1h = Bias resistor is connected to pin according IO_PU |
| 1   | GPIO1_RE | R/W  | 0h    | 0h = No bias resistor is connected to pin<br>1h = Bias resistor is connected to pin according IO_PU |
| 0   | GPIO0_RE | R/W  | 0h    | 0h = No bias resistor is connected to pin<br>1h = Bias resistor is connected to pin according IO_PU |

### 7.6.1.25 IO\_RE\_1 Register (Offset = 28h) [Reset = 00h]

IO\_RE\_1 is shown in [Figure 7-41](#) and described in [Table 7-80](#).

Return to the [Summary Table](#).

GPIO resistor enable

**Figure 7-41. IO\_RE\_1 Register**

| 7    | 6 | 5 | 4         | 3         | 2         | 1        | 0        |
|------|---|---|-----------|-----------|-----------|----------|----------|
| RSVD |   |   | GPIO12_RE | GPIO11_RE | GPIO10_RE | GPIO9_RE | GPIO8_RE |
| R-0h |   |   | R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h   | R/W-0h   |

**Table 7-80. IO\_RE\_1 Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                         |
|-----|-----------|------|-------|-----------------------------------------------------------------------------------------------------|
| 7-5 | RSVD      | R    | 0h    |                                                                                                     |
| 4   | GPIO12_RE | R/W  | 0h    | 0h = No bias resistor is connected to pin<br>1h = Bias resistor is connected to pin according IO_PU |
| 3   | GPIO11_RE | R/W  | 0h    | 0h = No bias resistor is connected to pin<br>1h = Bias resistor is connected to pin according IO_PU |
| 2   | GPIO10_RE | R/W  | 0h    | 0h = No bias resistor is connected to pin<br>1h = Bias resistor is connected to pin according IO_PU |
| 1   | GPIO9_RE  | R/W  | 0h    | 0h = No bias resistor is connected to pin<br>1h = Bias resistor is connected to pin according IO_PU |
| 0   | GPIO8_RE  | R/W  | 0h    | 0h = No bias resistor is connected to pin<br>1h = Bias resistor is connected to pin according IO_PU |

### 7.6.1.26 IO\_PU\_0 Register (Offset = 29h) [Reset = 00h]

IO\_PU\_0 is shown in [Figure 7-42](#) and described in [Table 7-81](#).

Return to the [Summary Table](#).

GPIO pull-up resistor enable

**Figure 7-42. IO\_PU\_0 Register**

| 7        | 6        | 5        | 4        | 3        | 2        | 1        | 0        |
|----------|----------|----------|----------|----------|----------|----------|----------|
| GPIO7_PU | GPIO6_PU | GPIO5_PU | GPIO4_PU | GPIO3_PU | GPIO2_PU | GPIO1_PU | GPIO0_PU |
| R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   | R/W-0h   |

**Table 7-81. IO\_PU\_0 Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                    |
|-----|----------|------|-------|------------------------------------------------------------------------------------------------|
| 7   | GPIO7_PU | R/W  | 0h    | 0h = Pull-down resistor to pin if IO_RE is set<br>1h = Pull-up resistor to pin if IO_RE is set |
| 6   | GPIO6_PU | R/W  | 0h    | 0h = Pull-down resistor to pin if IO_RE is set<br>1h = Pull-up resistor to pin if IO_RE is set |
| 5   | GPIO5_PU | R/W  | 0h    | 0h = Pull-down resistor to pin if IO_RE is set<br>1h = Pull-up resistor to pin if IO_RE is set |
| 4   | GPIO4_PU | R/W  | 0h    | 0h = Pull-down resistor to pin if IO_RE is set<br>1h = Pull-up resistor to pin if IO_RE is set |
| 3   | GPIO3_PU | R/W  | 0h    | 0h = Pull-down resistor to pin if IO_RE is set<br>1h = Pull-up resistor to pin if IO_RE is set |
| 2   | GPIO2_PU | R/W  | 0h    | 0h = Pull-down resistor to pin if IO_RE is set<br>1h = Pull-up resistor to pin if IO_RE is set |
| 1   | GPIO1_PU | R/W  | 0h    | 0h = Pull-down resistor to pin if IO_RE is set<br>1h = Pull-up resistor to pin if IO_RE is set |
| 0   | GPIO0_PU | R/W  | 0h    | 0h = Pull-down resistor to pin if IO_RE is set<br>1h = Pull-up resistor to pin if IO_RE is set |

### 7.6.1.27 IO\_PU\_1 Register (Offset = 2Ah) [Reset = 00h]

IO\_PU\_1 is shown in [Figure 7-43](#) and described in [Table 7-82](#).

Return to the [Summary Table](#).

GPIO pull-up resistor enable

**Figure 7-43. IO\_PU\_1 Register**

| 7    | 6 | 5 | 4         | 3         | 2         | 1        | 0        |
|------|---|---|-----------|-----------|-----------|----------|----------|
| RSVD |   |   | GPIO12_PU | GPIO11_PU | GPIO10_PU | GPIO9_PU | GPIO8_PU |
| R-0h |   |   | R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h   | R/W-0h   |

**Table 7-82. IO\_PU\_1 Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                    |
|-----|-----------|------|-------|------------------------------------------------------------------------------------------------|
| 7-5 | RSVD      | R    | 0h    |                                                                                                |
| 4   | GPIO12_PU | R/W  | 0h    | 0h = Pull-down resistor to pin if IO_RE is set<br>1h = Pull-up resistor to pin if IO_RE is set |
| 3   | GPIO11_PU | R/W  | 0h    | 0h = Pull-down resistor to pin if IO_RE is set<br>1h = Pull-up resistor to pin if IO_RE is set |
| 2   | GPIO10_PU | R/W  | 0h    | 0h = Pull-down resistor to pin if IO_RE is set<br>1h = Pull-up resistor to pin if IO_RE is set |
| 1   | GPIO9_PU  | R/W  | 0h    | 0h = Pull-down resistor to pin if IO_RE is set<br>1h = Pull-up resistor to pin if IO_RE is set |
| 0   | GPIO8_PU  | R/W  | 0h    | 0h = Pull-down resistor to pin if IO_RE is set<br>1h = Pull-up resistor to pin if IO_RE is set |

### 7.6.1.28 IO\_POL\_0 Register (Offset = 2Bh) [Reset = 00h]

IO\_POL\_0 is shown in [Figure 7-44](#) and described in [Table 7-83](#).

Return to the [Summary Table](#).

GPIO polarity inversion enable. These bits only apply when the pin is configured as an input. It will invert the signal input shown shown in the IO\_INPUT register

**Figure 7-44. IO\_POL\_0 Register**

| 7         | 6         | 5         | 4         | 3         | 2         | 1         | 0         |
|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| GPIO7_PIE | GPIO6_PIE | GPIO5_PIE | GPIO4_PIE | GPIO3_PIE | GPIO2_PIE | GPIO1_PIE | GPIO0_PIE |
| R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h    | R/W-0h    |

**Table 7-83. IO\_POL\_0 Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                           |
|-----|-----------|------|-------|---------------------------------------------------------------------------------------|
| 7   | GPIO7_PIE | R/W  | 0h    | 0h = Input polarity inversion is disabled<br>1h = Input polarity inversion is enabled |
| 6   | GPIO6_PIE | R/W  | 0h    | 0h = Input polarity inversion is disabled<br>1h = Input polarity inversion is enabled |
| 5   | GPIO5_PIE | R/W  | 0h    | 0h = Input polarity inversion is disabled<br>1h = Input polarity inversion is enabled |
| 4   | GPIO4_PIE | R/W  | 0h    | 0h = Input polarity inversion is disabled<br>1h = Input polarity inversion is enabled |
| 3   | GPIO3_PIE | R/W  | 0h    | 0h = Input polarity inversion is disabled<br>1h = Input polarity inversion is enabled |
| 2   | GPIO2_PIE | R/W  | 0h    | 0h = Input polarity inversion is disabled<br>1h = Input polarity inversion is enabled |
| 1   | GPIO1_PIE | R/W  | 0h    | 0h = Input polarity inversion is disabled<br>1h = Input polarity inversion is enabled |
| 0   | GPIO0_PIE | R/W  | 0h    | 0h = Input polarity inversion is disabled<br>1h = Input polarity inversion is enabled |

### 7.6.1.29 IO\_POL\_1 Register (Offset = 2Ch) [Reset = 00h]

IO\_POL\_1 is shown in [Figure 7-45](#) and described in [Table 7-84](#).

Return to the [Summary Table](#).

GPIO polarity inversion enable. These bits only apply when the pin is configured as an input. It will invert the signal input shown shown in the IO\_INPUT register

**Figure 7-45. IO\_POL\_1 Register**

| 7    | 6 | 5 | 4          | 3          | 2          | 1         | 0         |
|------|---|---|------------|------------|------------|-----------|-----------|
| RSVD |   |   | GPIO12_PIE | GPIO11_PIE | GPIO10_PIE | GPIO9_PIE | GPIO8_PIE |
| R-0h |   |   | R/W-0h     | R/W-0h     | R/W-0h     | R/W-0h    | R/W-0h    |

**Table 7-84. IO\_POL\_1 Register Field Descriptions**

| Bit | Field      | Type | Reset | Description                                                                           |
|-----|------------|------|-------|---------------------------------------------------------------------------------------|
| 7-5 | RSVD       | R    | 0h    |                                                                                       |
| 4   | GPIO12_PIE | R/W  | 0h    | 0h = Input polarity inversion is disabled<br>1h = Input polarity inversion is enabled |
| 3   | GPIO11_PIE | R/W  | 0h    | 0h = Input polarity inversion is disabled<br>1h = Input polarity inversion is enabled |
| 2   | GPIO10_PIE | R/W  | 0h    | 0h = Input polarity inversion is disabled<br>1h = Input polarity inversion is enabled |
| 1   | GPIO9_PIE  | R/W  | 0h    | 0h = Input polarity inversion is disabled<br>1h = Input polarity inversion is enabled |
| 0   | GPIO8_PIE  | R/W  | 0h    | 0h = Input polarity inversion is disabled<br>1h = Input polarity inversion is enabled |

### 7.6.1.30 IO\_OUTPUT\_0 Register (Offset = 2Dh) [Reset = 00h]

IO\_OUTPUT\_0 is shown in [Figure 7-46](#) and described in [Table 7-85](#).

Return to the [Summary Table](#).

**Figure 7-46. IO\_OUTPUT\_0 Register**

| 7                 | 6                 | 5                 | 4                 | 3                 | 2                 | 1                 | 0                 |
|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|-------------------|
| GPIO7_OUTPUT<br>T | GPIO6_OUTPUT<br>T | GPIO5_OUTPUT<br>T | GPIO4_OUTPUT<br>T | GPIO3_OUTPUT<br>T | GPIO2_OUTPUT<br>T | GPIO1_OUTPUT<br>T | GPIO0_OUTPUT<br>T |
| R/W-0h            | R/W-0h            | R/W-0h            | R/W-0h            | R/W-0h            | R/W-0h            | R/W-0h            | R/W-0h            |

**Table 7-85. IO\_OUTPUT\_0 Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                                                                                                                       |
|-----|--------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | GPIO7_OUTPUT | R/W  | 0h    | 0h = If pin is configured as an output, then the output is driven low.<br>1h = If the pin is configured as an output, then the output is driven high or high-z (depends on IO_OD) |
| 6   | GPIO6_OUTPUT | R/W  | 0h    | 0h = If pin is configured as an output, then the output is driven low.<br>1h = If the pin is configured as an output, then the output is driven high or high-z (depends on IO_OD) |
| 5   | GPIO5_OUTPUT | R/W  | 0h    | 0h = If pin is configured as an output, then the output is driven low.<br>1h = If the pin is configured as an output, then the output is driven high or high-z (depends on IO_OD) |
| 4   | GPIO4_OUTPUT | R/W  | 0h    | 0h = If pin is configured as an output, then the output is driven low.<br>1h = If the pin is configured as an output, then the output is driven high or high-z (depends on IO_OD) |
| 3   | GPIO3_OUTPUT | R/W  | 0h    | 0h = If pin is configured as an output, then the output is driven low.<br>1h = If the pin is configured as an output, then the output is driven high or high-z (depends on IO_OD) |
| 2   | GPIO2_OUTPUT | R/W  | 0h    | 0h = If pin is configured as an output, then the output is driven low.<br>1h = If the pin is configured as an output, then the output is driven high or high-z (depends on IO_OD) |
| 1   | GPIO1_OUTPUT | R/W  | 0h    | 0h = If pin is configured as an output, then the output is driven low.<br>1h = If the pin is configured as an output, then the output is driven high or high-z (depends on IO_OD) |
| 0   | GPIO0_OUTPUT | R/W  | 0h    | 0h = If pin is configured as an output, then the output is driven low.<br>1h = If the pin is configured as an output, then the output is driven high or high-z (depends on IO_OD) |

### 7.6.1.31 IO\_OUTPUT\_1 Register (Offset = 2Eh) [Reset = 00h]

IO\_OUTPUT\_1 is shown in [Figure 7-47](#) and described in [Table 7-86](#).

Return to the [Summary Table](#).

**Figure 7-47. IO\_OUTPUT\_1 Register**

| 7    | 6 | 5 | 4                 | 3                 | 2                 | 1                | 0                |
|------|---|---|-------------------|-------------------|-------------------|------------------|------------------|
| RSVD |   |   | GPIO12_OUTP<br>UT | GPIO11_OUTP<br>UT | GPIO10_OUTP<br>UT | GPIO9_OUTPU<br>T | GPIO8_OUTPU<br>T |
| R-0h |   |   | R/W-0h            | R/W-0h            | R/W-0h            | R/W-0h           | R/W-0h           |

**Table 7-86. IO\_OUTPUT\_1 Register Field Descriptions**

| Bit | Field         | Type | Reset | Description                                                                                                                                                                       |
|-----|---------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD          | R    | 0h    |                                                                                                                                                                                   |
| 4   | GPIO12_OUTPUT | R/W  | 0h    | 0h = If pin is configured as an output, then the output is driven low.<br>1h = If the pin is configured as an output, then the output is driven high or high-z (depends on IO_OD) |
| 3   | GPIO11_OUTPUT | R/W  | 0h    | 0h = If pin is configured as an output, then the output is driven low.<br>1h = If the pin is configured as an output, then the output is driven high or high-z (depends on IO_OD) |
| 2   | GPIO10_OUTPUT | R/W  | 0h    | 0h = If pin is configured as an output, then the output is driven low.<br>1h = If the pin is configured as an output, then the output is driven high or high-z (depends on IO_OD) |
| 1   | GPIO9_OUTPUT  | R/W  | 0h    | 0h = If pin is configured as an output, then the output is driven low.<br>1h = If the pin is configured as an output, then the output is driven high or high-z (depends on IO_OD) |
| 0   | GPIO8_OUTPUT  | R/W  | 0h    | 0h = If pin is configured as an output, then the output is driven low.<br>1h = If the pin is configured as an output, then the output is driven high or high-z (depends on IO_OD) |

### 7.6.1.32 IO\_INPUT\_0 Register (Offset = 2Fh) [Reset = 00h]

IO\_INPUT\_0 is shown in [Figure 7-48](#) and described in [Table 7-87](#).

Return to the [Summary Table](#).

**Figure 7-48. IO\_INPUT\_0 Register**

| 7           | 6           | 5           | 4           | 3           | 2           | 1           | 0           |
|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| GPIO7_INPUT | GPIO6_INPUT | GPIO5_INPUT | GPIO4_INPUT | GPIO3_INPUT | GPIO2_INPUT | GPIO1_INPUT | GPIO0_INPUT |
| RH-0h       | RH-0h       | RH-0h       | RH-0h       | RH-0h       | RH-0h       | RH-0h       | RH-0h       |

**Table 7-87. IO\_INPUT\_0 Register Field Descriptions**

| Bit | Field       | Type | Reset | Description                                                                           |
|-----|-------------|------|-------|---------------------------------------------------------------------------------------|
| 7   | GPIO7_INPUT | RH   | 0h    | 0h = The current state of the pin is low<br>1h = The current state of the pin is high |
| 6   | GPIO6_INPUT | RH   | 0h    | 0h = The current state of the pin is low<br>1h = The current state of the pin is high |
| 5   | GPIO5_INPUT | RH   | 0h    | 0h = The current state of the pin is low<br>1h = The current state of the pin is high |
| 4   | GPIO4_INPUT | RH   | 0h    | 0h = The current state of the pin is low<br>1h = The current state of the pin is high |
| 3   | GPIO3_INPUT | RH   | 0h    | 0h = The current state of the pin is low<br>1h = The current state of the pin is high |
| 2   | GPIO2_INPUT | RH   | 0h    | 0h = The current state of the pin is low<br>1h = The current state of the pin is high |
| 1   | GPIO1_INPUT | RH   | 0h    | 0h = The current state of the pin is low<br>1h = The current state of the pin is high |
| 0   | GPIO0_INPUT | RH   | 0h    | 0h = The current state of the pin is low<br>1h = The current state of the pin is high |

### 7.6.1.33 IO\_INPUT\_1 Register (Offset = 30h) [Reset = 00h]

IO\_INPUT\_1 is shown in [Figure 7-49](#) and described in [Table 7-88](#).

Return to the [Summary Table](#).

**Figure 7-49. IO\_INPUT\_1 Register**

| 7    | 6 | 5 | 4            | 3            | 2            | 1           | 0           |
|------|---|---|--------------|--------------|--------------|-------------|-------------|
| RSVD |   |   | GPIO12_INPUT | GPIO11_INPUT | GPIO10_INPUT | GPIO9_INPUT | GPIO8_INPUT |
| R-0h |   |   | RH-0h        | RH-0h        | RH-0h        | RH-0h       | RH-0h       |

**Table 7-88. IO\_INPUT\_1 Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                           |
|-----|--------------|------|-------|---------------------------------------------------------------------------------------|
| 7-5 | RSVD         | R    | 0h    |                                                                                       |
| 4   | GPIO12_INPUT | RH   | 0h    | 0h = The current state of the pin is low<br>1h = The current state of the pin is high |
| 3   | GPIO11_INPUT | RH   | 0h    | 0h = The current state of the pin is low<br>1h = The current state of the pin is high |
| 2   | GPIO10_INPUT | RH   | 0h    | 0h = The current state of the pin is low<br>1h = The current state of the pin is high |
| 1   | GPIO9_INPUT  | RH   | 0h    | 0h = The current state of the pin is low<br>1h = The current state of the pin is high |
| 0   | GPIO8_INPUT  | RH   | 0h    | 0h = The current state of the pin is low<br>1h = The current state of the pin is high |

### 7.6.1.34 IR\_STATUS Register (Offset = 31h) [Reset = 00h]

IR\_STATUS is shown in [Table 7-89](#).

Return to the [Summary Table](#).

Interrupt status register shows which IP block has a pending interrupt is enabled on either interrupt line and is also set. If an interrupt is not enabled, the interrupt will not be shown

**Table 7-89. IR\_STATUS Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                                                                                                 |
|-----|-----------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD      | R    | 0h    | Reserved                                                                                                                                                                    |
| 5   | PWM1_IR   | R    | 0h    | Interrupt is set in the PWM1 interrupt register<br>0h = No interrupt is set in the PWM1 interrupt register<br>1h = An interrupt is set in the PWM1 interrupt register       |
| 4   | PWM0_IR   | R    | 0h    | Interrupt is set in the PWM0 interrupt register<br>0h = No interrupt is set in the PWM0 interrupt register<br>1h = An interrupt is set in the PWM0 interrupt register       |
| 3   | I2C_IR    | R    | 0h    | Interrupt is set in the I2C interrupt register<br>0h = No interrupt is set in the I2C interrupt register<br>1h = An interrupt is set in the I2C interrupt register          |
| 2   | UART_IR   | R    | 0h    | Interrupt is set in the UART interrupt register<br>0h = No interrupt is set in the UART interrupt register<br>1h = An interrupt is set in the UART interrupt register       |
| 1   | SPI_IR    | R    | 0h    | Interrupt is set in the SPI interrupt register<br>0h = No interrupt is set in the SPI interrupt register<br>1h = An interrupt is set in the SPI interrupt register          |
| 0   | DEVICE_IR | R    | 0h    | Interrupt is set in the device interrupt register<br>0h = No interrupt is set in the device interrupt register<br>1h = An interrupt is set in the device interrupt register |

### 7.6.1.35 DEV\_IR Register (Offset = 32h) [Reset = 00h]

DEV\_IR is shown in [Figure 7-50](#) and described in [Table 7-90](#).

Return to the [Summary Table](#).

Device interrupt status register

**Figure 7-50. DEV\_IR Register**

| 7    | 6 | 5 | 4 | 3 | 2 | 1        | 0    |
|------|---|---|---|---|---|----------|------|
| RSVD |   |   |   |   |   | CANWK    | RSVD |
| R-0h |   |   |   |   |   | R/W1C-0h | R-0h |

**Table 7-90. DEV\_IR Register Field Descriptions**

| Bit | Field | Type  | Reset | Description                                                                                                                              |
|-----|-------|-------|-------|------------------------------------------------------------------------------------------------------------------------------------------|
| 7-2 | RSVD  | R     | 0h    | Reserved                                                                                                                                 |
| 1   | CANWK | R/W1C | 0h    | CAN Wakeup<br>The device has come out of sleep mode due to the CAN bus<br>0h = Interrupt has not occurred<br>1h = Interrupt has occurred |
| 0   | RSVD  | R     | 0h    | Reserved                                                                                                                                 |

### 7.6.1.36 SPI\_IR Register (Offset = 33h) [Reset = 00h]

SPI\_IR is shown in [Figure 7-51](#) and described in [Table 7-91](#).

Return to the [Summary Table](#).

This is a writable mirror of the SPI\_IR register for easy reading and clearing of interrupts

**Figure 7-51. SPI\_IR Register**

| 7        | 6        | 5    | 4 | 3 | 2        | 1        | 0        |
|----------|----------|------|---|---|----------|----------|----------|
| TXL      | TXA      | RSVD |   |   | RXL      | RXFL     | RXN      |
| R/W1C-0h | R/W1C-0h | R-0h |   |   | R/W1C-0h | R/W1C-0h | R/W1C-0h |

**Table 7-91. SPI\_IR Register Field Descriptions**

| Bit | Field | Type  | Reset | Description                                                                                                                                                                                                                                  |
|-----|-------|-------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXL   | R/W1C | 0h    | TX lost message due to full FIFO interrupt<br>0h = No TX FIFO message was lost<br>1h = TX FIFO lost a message (overrun) due to FIFO being full when a write was attempted                                                                    |
| 6   | TXA   | R/W1C | 0h    | TX space available interrupt<br><br><b>Note</b><br>When the SPI IP is enabled, this interrupt will set since the FIFO will be empty<br><br>0h = No TX FIFO space available interrupt<br>1h = The TX FIFO has hit the trigger level threshold |
| 5-3 | RSVD  | R     | 0h    |                                                                                                                                                                                                                                              |
| 2   | RXL   | R/W1C | 0h    | RX overrun/lost message interrupt<br>0h = No RX FIFO messages lost (no overrun) interrupt<br>1h = At least 1 message has been lost due to full RX FIFO (overrun interrupt)                                                                   |
| 1   | RXFL  | R/W1C | 0h    | RX fill level interrupt.<br>Level is set by SPI_CTRL.RX_LVL_INT<br>0h = No RX FIFO fill level interrupt<br>1h = The RX FIFO reached the fill level                                                                                           |
| 0   | RXN   | R/W1C | 0h    | RX new message interrupt<br>0h = No new message received interrupt<br>1h = A new message has been received                                                                                                                                   |

### 7.6.1.37 UART\_IR Register (Offset = 34h) [Reset = 00h]

UART\_IR is shown in [Figure 7-52](#) and described in [Table 7-92](#).

Return to the [Summary Table](#).

This is a writable mirror of the UART\_IR register for easy reading and clearing of interrupts

**Figure 7-52. UART\_IR Register**

| 7        | 6        | 5        | 4        | 3        | 2        | 1        | 0        |
|----------|----------|----------|----------|----------|----------|----------|----------|
| TXL      | TXA      | RXBR     | RXFE     | RXPE     | RXL      | RXFL     | RXN      |
| R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h |

**Table 7-92. UART\_IR Register Field Descriptions**

| Bit | Field | Type  | Reset | Description                                                                                                                                                                                                                                                             |
|-----|-------|-------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXL   | R/W1C | 0h    | TX lost byte due to being full<br>0h = No TX FIFO byte was lost<br>1h = TX FIFO lost a byte (overrun) due to FIFO being full when a write was attempted                                                                                                                 |
| 6   | TXA   | R/W1C | 0h    | TX space available<br><br><div style="text-align: center;"><b>Note</b></div> When the UART IP is enabled, this interrupt will set since the FIFO will be empty<br><br>0h = No TX FIFO space available interrupt<br>1h = The TX FIFO has hit the trigger level threshold |
| 5   | RXBR  | R/W1C | 0h    | RX break received<br>0h = No break was received<br>1h = A break was received                                                                                                                                                                                            |
| 4   | RXFE  | R/W1C | 0h    | RX framing error<br>0h = No received framing error has occurred<br>1h = RX framing error has occurred                                                                                                                                                                   |
| 3   | RXPE  | R/W1C | 0h    | RX parity error<br>0h = No received parity error has occurred<br>1h = RX parity error has occurred                                                                                                                                                                      |
| 2   | RXL   | R/W1C | 0h    | RX overrun/lost byte<br>0h = No RX FIFO bytes lost (no overrun) interrupt<br>1h = At least 1 byte has been lost due to full RX FIFO (overrun interrupt)                                                                                                                 |
| 1   | RXFL  | R/W1C | 0h    | RX fill level<br>0h = No RX FIFO fill level interrupt<br>1h = The RX FIFO reached the fill level                                                                                                                                                                        |
| 0   | RXN   | R/W1C | 0h    | RX new byte<br>0h = No new byte received interrupt<br>1h = A new byte has been received                                                                                                                                                                                 |

### 7.6.1.38 I2C\_IR Register (Offset = 35h) [Reset = 00h]

I2C\_IR is shown in [Figure 7-53](#) and described in [Table 7-93](#).

Return to the [Summary Table](#).

This is a writable mirror of the I2C\_IR register for easy reading and clearing of interrupts

**Figure 7-53. I2C\_IR Register**

| 7        | 6        | 5        | 4        | 3        | 2        | 1        | 0        |
|----------|----------|----------|----------|----------|----------|----------|----------|
| TXL      | TXA      | DNACK    | ANACK    | SBRC     | RXL      | RXFL     | RXN      |
| R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h |

**Table 7-93. I2C\_IR Register Field Descriptions**

| Bit | Field | Type  | Reset | Description                                                                                                                                                                                                                                                                                                   |
|-----|-------|-------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXL   | R/W1C | 0h    | TX lost message due to full FIFO interrupt<br>0h = No TX FIFO byte was lost<br>1h = TX FIFO lost a byte (overrun) due to FIFO being full when a write was attempted                                                                                                                                           |
| 6   | TXA   | R/W1C | 0h    | TX space available interrupt.<br>The trigger level is set by the I2C_FIFO_CTRL.TX_TRG setting<br><br><b>Note</b><br>When the I2C IP is enabled, this interrupt will set since the FIFO will be empty<br><br>0h = No TX FIFO space available interrupt<br>1h = The TX FIFO has hit the trigger level threshold |
| 5   | DNACK | R/W1C | 0h    | Data NACK interrupt<br>0h = No NACK was detected<br>1h = A NACK was detected during the data phase                                                                                                                                                                                                            |
| 4   | ANACK | R/W1C | 0h    | Address NACK interrupt<br>0h = No NACK was detected<br>1h = A NACK was detected during the address phase                                                                                                                                                                                                      |
| 3   | SBRC  | R/W1C | 0h    | Stuck bus recovery completed interrupt<br>0h = No stuck bus recovery has occurred<br>1h = Stuck bus recovery has completed. Check the I2C_STATUS register to see if the bus is stuck or still.                                                                                                                |
| 2   | RXL   | R/W1C | 0h    | RX overrun/lost message interrupt<br>0h = No RX FIFO bytes lost (no overrun) interrupt<br>1h = At least 1 byte has been lost due to full RX FIFO (overrun interrupt)                                                                                                                                          |
| 1   | RXFL  | R/W1C | 0h    | RX fill level interrupt<br>0h = No RX FIFO fill level interrupt<br>1h = The RX FIFO reached the fill level                                                                                                                                                                                                    |
| 0   | RXN   | R/W1C | 0h    | RX new message interrupt<br>0h = No new byte received interrupt<br>1h = A new byte has been received                                                                                                                                                                                                          |

### 7.6.1.39 PWM0\_IR Register (Offset = 36h) [Reset = 00h]

PWM0\_IR is shown in [Figure 7-54](#) and described in [Table 7-94](#).

Return to the [Summary Table](#).

This is a writable mirror of the PWM0\_IR register for easy reading and clearing of interrupts

**Figure 7-54. PWM0\_IR Register**

| 7    | 6 | 5 | 4 | 3         | 2 | 1        | 0        |
|------|---|---|---|-----------|---|----------|----------|
| RSVD |   |   |   | PULSE_OVF |   | IAS      | RC       |
| R-0h |   |   |   | R/W1C-0h  |   | R/W1C-0h | R/W1C-0h |

**Table 7-94. PWM0\_IR Register Field Descriptions**

| Bit | Field     | Type  | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----|-----------|-------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD      | R     | 0h    | Reserved                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 2   | PULSE_OVF | R/W1C | 0h    | Current Pulse Overflow<br>Set when the CUR_PULSE counter overflows<br>0h = Current pulse counter has not overflowed<br>1h = Current pulse counter has overflowed                                                                                                                                                                                                                                                                                                             |
| 1   | IAS       | R/W1C | 0h    | Input Auto Stop Interrupt<br>Sets when the input auto stop condition has occurred, if the PWM is configured to immediately stop, then the output will also be stopped (reflected by the RC bit).<br>If PWM is configured to begin the stop ramp on an IAS trigger, then this interrupt is set immediately when the input condition happens and the RC bit will be set when the output turns off.<br>0h = Input for IAS has not triggered<br>1h = Input for IAS has triggered |
| 0   | RC        | R/W1C | 0h    | PWM Ramp Complete<br>Sets when the ramp has finished. If AS_EN = 1 (auto stop), then this happens when the PWM channel turns off.<br>If AS_EN = 0, then this gets set when the PWM has reached the end value.<br>If IAS_EN = 1 (input auto stop), then this will get set once the PWM output is stopped<br>0h = PWM channel ramp is not complete<br>1h = PWM channel ramp is complete                                                                                        |

### 7.6.1.40 PWM1\_IR Register (Offset = 37h) [Reset = 00h]

PWM1\_IR is shown in [Figure 7-55](#) and described in [Table 7-95](#).

Return to the [Summary Table](#).

This is a writable mirror of the PWM1\_IR register for easy reading and clearing of interrupts

**Figure 7-55. PWM1\_IR Register**

| 7    | 6 | 5 | 4 | 3         | 2 | 1        | 0        |
|------|---|---|---|-----------|---|----------|----------|
| RSVD |   |   |   | PULSE_OVF |   | IAS      | RC       |
| R-0h |   |   |   | R/W1C-0h  |   | R/W1C-0h | R/W1C-0h |

**Table 7-95. PWM1\_IR Register Field Descriptions**

| Bit | Field     | Type  | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----|-----------|-------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD      | R     | 0h    | Reserved                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 2   | PULSE_OVF | R/W1C | 0h    | Current Pulse Overflow<br>Set when the CUR_PULSE counter overflows<br>0h = Current pulse counter has not overflowed<br>1h = Current pulse counter has overflowed                                                                                                                                                                                                                                                                                                             |
| 1   | IAS       | R/W1C | 0h    | Input Auto Stop Interrupt<br>Sets when the input auto stop condition has occurred, if the PWM is configured to immediately stop, then the output will also be stopped (reflected by the RC bit).<br>If PWM is configured to begin the stop ramp on an IAS trigger, then this interrupt is set immediately when the input condition happens and the RC bit will be set when the output turns off.<br>0h = Input for IAS has not triggered<br>1h = Input for IAS has triggered |
| 0   | RC        | R/W1C | 0h    | PWM Ramp Complete<br>Sets when the ramp has finished. If AS_EN = 1 (auto stop), then this happens when the PWM channel turns off.<br>If AS_EN = 0, then this gets set when the PWM has reached the end value.<br>If IAS_EN = 1 (input auto stop), then this will get set once the PWM output is stopped<br>0h = PWM channel ramp is not complete<br>1h = PWM channel ramp is complete                                                                                        |

### 7.6.2 SPI Registers

[Section 7.6.2](#) lists the memory-mapped registers for the SPI registers. All register offset addresses not listed in [Section 7.6.2](#) should be considered as reserved locations and the register contents should not be modified.

**Table 7-96. SPI Registers**

| Offset | Acronym       | Register Name               | Section                          |
|--------|---------------|-----------------------------|----------------------------------|
| 1000h  | SPI_CREL      | SPI Core Release            | <a href="#">Section 7.6.2.1</a>  |
| 1001h  | SPI_SCRATCH   | SPI Scratchpad              | <a href="#">Section 7.6.2.2</a>  |
| 1002h  | SPI_CTRL      | SPI Control                 | <a href="#">Section 7.6.2.3</a>  |
| 1003h  | SPI_CFG_0     | SPI Channel Configuration 0 | <a href="#">Section 7.6.2.4</a>  |
| 1004h  | SPI_CFG_1     | SPI Channel Configuration 1 | <a href="#">Section 7.6.2.5</a>  |
| 1005h  | SPI_DR_0      | SPI Data Rate 0             | <a href="#">Section 7.6.2.6</a>  |
| 1006h  | SPI_DR_1      | SPI Data Rate 1             | <a href="#">Section 7.6.2.7</a>  |
| 1007h  | SPI_DR_2      | SPI Data Rate 2             | <a href="#">Section 7.6.2.8</a>  |
| 1008h  | SPI_DR_3      | SPI Data Rate 3             | <a href="#">Section 7.6.2.9</a>  |
| 1009h  | SPI_CHAN_EN   | SPI Channel Enable          | <a href="#">Section 7.6.2.10</a> |
| 100Ah  | SPI_CS_POL    | SPI Chip Select Polarity    | <a href="#">Section 7.6.2.11</a> |
| 100Bh  | SPI_FIFO_CTRL | SPI FIFO Control            | <a href="#">Section 7.6.2.12</a> |

**Table 7-96. SPI Registers (continued)**

| Offset | Acronym     | Register Name          | Section                          |
|--------|-------------|------------------------|----------------------------------|
| 100Ch  | SPI_IE_0    | SPI Interrupt Enable 0 | <a href="#">Section 7.6.2.13</a> |
| 100Dh  | SPI_IE_1    | SPI Interrupt Enable 1 | <a href="#">Section 7.6.2.14</a> |
| 100Eh  | SPI_IR      | SPI Interrupt Register | <a href="#">Section 7.6.2.15</a> |
| 100Fh  | SPI_FS      | SPI FIFO Status        | <a href="#">Section 7.6.2.16</a> |
| 1010h  | SPI_RX_FIFO | SPI Receive FIFO       | <a href="#">Section 7.6.2.18</a> |
| 1010h  | SPI_TX_FIFO | SPI Transmit FIFO      | <a href="#">Section 7.6.2.17</a> |
| 1011h  | SPI_RXFS    | SPI RX FIFO Status     | <a href="#">Section 7.6.2.19</a> |
| 1012h  | SPI_TXFS    | SPI TX FIFO Status     | <a href="#">Section 7.6.2.20</a> |
| 1013h  | SPI_TXES    | SPI TX Element Status  | <a href="#">Section 7.6.2.21</a> |

Complex bit access types are encoded to fit into small table cells. [Section 7.6.2](#) shows the codes that are used for access types in this section.

**Table 7-97. SPI Access Type Codes**

| Access Type            | Code    | Description                            |
|------------------------|---------|----------------------------------------|
| Read Type              |         |                                        |
| R                      | R       | Read                                   |
| RH                     | R<br>H  | Read<br>Set or cleared by hardware     |
| Write Type             |         |                                        |
| W                      | W       | Write                                  |
| W1C                    | W<br>1C | Write<br>1 to clear                    |
| W1S                    | W<br>1S | Write<br>1 to set                      |
| WP                     | W<br>P  | Write<br>Requires privileged access    |
| Reset or Default Value |         |                                        |
| -n                     |         | Value after reset or the default value |

### 7.6.2.1 SPI\_CREL Register (Offset = 1000h) [Reset = 87h]

SPI\_CREL is shown in [Figure 7-56](#) and described in [Table 7-98](#).

Return to the [Summary Table](#).

SPI IP release version

**Figure 7-56. SPI\_CREL Register**

| 7     | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|---|---|---|
| CREL  |   |   |   |   |   |   |   |
| R-87h |   |   |   |   |   |   |   |

**Table 7-98. SPI\_CREL Register Field Descriptions**

| Bit | Field | Type | Reset | Description              |
|-----|-------|------|-------|--------------------------|
| 7-0 | CREL  | R    | 87h   | SPI Core Release Version |

### 7.6.2.2 SPI\_SCRATCH Register (Offset = 1001h) [Reset = 00h]

SPI\_SCRATCH is shown in [Figure 7-57](#) and described in [Table 7-99](#).

Return to the [Summary Table](#).

Customer Scratchpad register

**Figure 7-57. SPI\_SCRATCH Register**

|         |   |   |   |   |   |   |   |
|---------|---|---|---|---|---|---|---|
| 7       | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SCRATCH |   |   |   |   |   |   |   |
| R/W-0h  |   |   |   |   |   |   |   |

**Table 7-99. SPI\_SCRATCH Register Field Descriptions**

| Bit | Field   | Type | Reset | Description                                                                                                                                |
|-----|---------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | SCRATCH | R/W  | 0h    | A customer writable register for testing reads and writes. This register does not have any functionality outside of a read and write test. |

### 7.6.2.3 SPI\_CTRL Register (Offset = 1002h) [Reset = 00h]

SPI\_CTRL is shown in [Figure 7-58](#) and described in [Table 7-100](#).

Return to the [Summary Table](#).

This register controls the SPI IP

**Figure 7-58. SPI\_CTRL Register**

| 7      | 6 | 5      | 4 | 3        | 2 | 1      | 0       |
|--------|---|--------|---|----------|---|--------|---------|
| TX_TRG |   | RX_TRG |   | RESERVED |   | SPI_EN | SPI_CCE |
| R/W-0h |   | R/W-0h |   | R-0h     |   | RH-0h  | R/W-0h  |

**Table 7-100. SPI\_CTRL Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----|----------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-6 | TX_TRG   | R/W  | 0h    | <p>Sets the trigger level for when the number of free bytes in the TX FIFO increases to the specified amount. Used to interrupt when there's a certain number of free spaces in the FIFO so the processor can load multiple bytes into the FIFO instead of just 1 at a time.</p> <hr/> <p style="text-align: center;"><b>Note</b></p> <p style="text-align: center;">This will trigger like an edge-based interrupt.<br/>If the user never falls below 8 free spaces</p> <hr/> <p>0h = 16 bytes free<br/>1h = 32 bytes free<br/>2h = [(max size) - 128] bytes free<br/>3h = [(max size) - 64] bytes free</p> |
| 5-4 | RX_TRG   | R/W  | 0h    | <p>Sets the trigger level for the RX FIFO when the RX FIFO reaches the specified number of bytes stored in the FIFO. This allows the CPU to shift bytes out in bulk instead of 1 byte at a time to save the number of interrupts</p> <p>0h = 4 bytes in RX FIFO<br/>1h = 8 bytes in RX FIFO<br/>2h = [(max size) - 128] bytes in RX FIFO<br/>3h = [(max size) - 64] bytes in RX FIFO</p>                                                                                                                                                                                                                     |
| 3-2 | RESERVED | R    | 0h    | Reserved                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 1   | SPI_EN   | RH   | 0h    | <p>SPI IP enable status flag. This flag is not writable, but is set if the SPI IP is enabled by allocating memory from MRAM to the IP. This is done with the <a href="#">MRAM_IP_CFG</a> register</p> <p>0h = SPI IP is disabled, all SPI functionality is disabled<br/>1h = SPI IP is enabled</p>                                                                                                                                                                                                                                                                                                           |
| 0   | SPI_CCE  | R/W  | 0h    | <p>SPI IP change control enable bit. Can be set only if SPI_EN is 0</p> <p>0h = SPI configuration registers are write protected<br/>1h = SPI configuration registers can be changed</p>                                                                                                                                                                                                                                                                                                                                                                                                                      |

#### 7.6.2.4 SPI\_CFG\_0 Register (Offset = 1003h) [Reset = 00h]

SPI\_CFG\_0 is shown in [Figure 7-59](#) and described in [Table 7-101](#).

Return to the [Summary Table](#).

SPI channel configuration. Sets SPI mode, and other channel-specific settings.

**Figure 7-59. SPI\_CFG\_0 Register**

| 7    | 6       | 5       | 4       | 3    | 2       | 1       | 0       |
|------|---------|---------|---------|------|---------|---------|---------|
| RSVD | LSB_1   | CLKPL_1 | CLKPH_1 | RSVD | LSB_0   | CLKPL_0 | CLKPH_0 |
| R-0h | R/WP-0h | R/WP-0h | R/WP-0h | R-0h | R/WP-0h | R/WP-0h | R/WP-0h |

**Table 7-101. SPI\_CFG\_0 Register Field Descriptions**

| Bit | Field   | Type | Reset | Description                                                                                                                                                                     |
|-----|---------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | RSVD    | R    | 0h    | Reserved                                                                                                                                                                        |
| 6   | LSB_1   | R/WP | 0h    | LSB first select.<br>Controls the direction of the receive and transmit shift register<br>0h = MSB first<br>1h = LSB first                                                      |
| 5   | CLKPL_1 | R/WP | 0h    | Clock polarity select<br>0h = The inactive state of SCK is low<br>1h = The inactive state of SCK is high                                                                        |
| 4   | CLKPH_1 | R/WP | 0h    | Clock phase select<br>0h = Data is changed on the first CLK edge and captured on the following edge<br>1h = Data is captured on the first CLK edge and changed on the following |
| 3   | RSVD    | R    | 0h    | Reserved                                                                                                                                                                        |
| 2   | LSB_0   | R/WP | 0h    | LSB first select.<br>Controls the direction of the receive and transmit shift register<br>0h = MSB first<br>1h = LSB first                                                      |
| 1   | CLKPL_0 | R/WP | 0h    | Clock polarity select<br>0h = The inactive state of SCK is low<br>1h = The inactive state of SCK is high                                                                        |
| 0   | CLKPH_0 | R/WP | 0h    | Clock phase select<br>0h = Data is changed on the first CLK edge and captured on the following edge<br>1h = Data is captured on the first CLK edge and changed on the following |

### 7.6.2.5 SPI\_CFG\_1 Register (Offset = 1004h) [Reset = 00h]

SPI\_CFG\_1 is shown in [Figure 7-60](#) and described in [Table 7-102](#).

Return to the [Summary Table](#).

SPI channel configuration. Sets SPI mode, and other channel-specific settings.

**Figure 7-60. SPI\_CFG\_1 Register**

| 7    | 6       | 5       | 4       | 3    | 2       | 1       | 0       |
|------|---------|---------|---------|------|---------|---------|---------|
| RSVD | LSB_3   | CLKPL_3 | CLKPH_3 | RSVD | LSB_2   | CLKPL_2 | CLKPH_2 |
| R-0h | R/WP-0h | R/WP-0h | R/WP-0h | R-0h | R/WP-0h | R/WP-0h | R/WP-0h |

**Table 7-102. SPI\_CFG\_1 Register Field Descriptions**

| Bit | Field   | Type | Reset | Description                                                                                                                                                                     |
|-----|---------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | RSVD    | R    | 0h    | Reserved                                                                                                                                                                        |
| 6   | LSB_3   | R/WP | 0h    | LSB first select.<br>Controls the direction of the receive and transmit shift register<br>0h = MSB first<br>1h = LSB first                                                      |
| 5   | CLKPL_3 | R/WP | 0h    | Clock polarity select<br>0h = The inactive state of SCK is low<br>1h = The inactive state of SCK is high                                                                        |
| 4   | CLKPH_3 | R/WP | 0h    | Clock phase select<br>0h = Data is changed on the first CLK edge and captured on the following edge<br>1h = Data is captured on the first CLK edge and changed on the following |
| 3   | RSVD    | R    | 0h    | Reserved                                                                                                                                                                        |
| 2   | LSB_2   | R/WP | 0h    | LSB first select.<br>Controls the direction of the receive and transmit shift register<br>0h = MSB first<br>1h = LSB first                                                      |
| 1   | CLKPL_2 | R/WP | 0h    | Clock polarity select<br>0h = The inactive state of SCK is low<br>1h = The inactive state of SCK is high                                                                        |
| 0   | CLKPH_2 | R/WP | 0h    | Clock polarity select<br>0h = The inactive state of SCK is low<br>1h = The inactive state of SCK is high                                                                        |

### 7.6.2.6 SPI\_DR\_0 Register (Offset = 1005h) [Reset = 00h]

SPI\_DR\_0 is shown in [Figure 7-61](#) and described in [Table 7-103](#).

Return to the [Summary Table](#).

The SPI data rate register for SPI channel 0. Values in this register are interpreted as 1 more than the entered value. The value is interpreted as the half-period of the SPI clock. The frequency can be found with the equation  $f = 20\text{MHz} / (x+1)$ . Where x is the value in SPI\_DR. The closest register value can be found with  $x = (20\text{MHz} / f) - 1$

**Figure 7-61. SPI\_DR\_0 Register**

| 7         | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----------|---|---|---|---|---|---|---|
| PRESCALER |   |   |   |   |   |   |   |
| R/WP-0h   |   |   |   |   |   |   |   |

**Table 7-103. SPI\_DR\_0 Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                                                                                      |
|-----|-----------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | PRESCALER | R/WP | 0h    | Lower byte of SPI clock prescaler.<br>The prescaler value is interpreted as 1 more than entered value.<br>This value defaults to the value programmed in EEPROM. |

### 7.6.2.7 SPI\_DR\_1 Register (Offset = 1006h) [Reset = 00h]

SPI\_DR\_1 is shown in [Figure 7-62](#) and described in [Table 7-104](#).

Return to the [Summary Table](#).

The SPI data rate register for SPI channel 1. Values in this register are interpreted as 1 more than the entered value. The value is interpreted as the half-period of the SPI clock. The frequency can be found with the equation  $f = 20\text{MHz} / (x+1)$ . Where x is the value in SPI\_DR. The closest register value can be found with  $x = (20\text{MHz} / f) - 1$

**Figure 7-62. SPI\_DR\_1 Register**

| 7         | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----------|---|---|---|---|---|---|---|
| PRESCALER |   |   |   |   |   |   |   |
| R/WP-0h   |   |   |   |   |   |   |   |

**Table 7-104. SPI\_DR\_1 Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                                                                                      |
|-----|-----------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | PRESCALER | R/WP | 0h    | Lower byte of SPI clock prescaler.<br>The prescaler value is interpreted as 1 more than entered value.<br>This value defaults to the value programmed in EEPROM. |

### 7.6.2.8 SPI\_DR\_2 Register (Offset = 1007h) [Reset = 00h]

SPI\_DR\_2 is shown in [Figure 7-63](#) and described in [Table 7-105](#).

Return to the [Summary Table](#).

The SPI data rate register for SPI channel 2. Values in this register are interpreted as 1 more than the entered value. The value is interpreted as the half-period of the SPI clock. The frequency can be found with the equation  $f = 20\text{MHz} / (x+1)$ . Where x is the value in SPI\_DR. The closest register value can be found with  $x = (20\text{MHz} / f) - 1$

**Figure 7-63. SPI\_DR\_2 Register**

| 7         | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----------|---|---|---|---|---|---|---|
| PRESCALER |   |   |   |   |   |   |   |
| R/WP-0h   |   |   |   |   |   |   |   |

**Table 7-105. SPI\_DR\_2 Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                                                                                      |
|-----|-----------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | PRESCALER | R/WP | 0h    | Lower byte of SPI clock prescaler.<br>The prescaler value is interpreted as 1 more than entered value.<br>This value defaults to the value programmed in EEPROM. |

### 7.6.2.9 SPI\_DR\_3 Register (Offset = 1008h) [Reset = 00h]

SPI\_DR\_3 is shown in [Figure 7-64](#) and described in [Table 7-106](#).

Return to the [Summary Table](#).

The SPI data rate register for SPI channel 3. Values in this register are interpreted as 1 more than the entered value. The value is interpreted as the half-period of the SPI clock. The frequency can be found with the equation  $f = 20\text{MHz} / (x+1)$ . Where x is the value in SPI\_DR. The closest register value can be found with  $x = (20\text{MHz} / f) - 1$

**Figure 7-64. SPI\_DR\_3 Register**

| 7         | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----------|---|---|---|---|---|---|---|
| PRESCALER |   |   |   |   |   |   |   |
| R/WP-0h   |   |   |   |   |   |   |   |

**Table 7-106. SPI\_DR\_3 Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                                                                                                                                                                                    |
|-----|-----------|------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | PRESCALER | R/WP | 0h    | Lower byte of SPI clock prescaler.<br>The prescaler value is interpreted as 1 more than entered value.<br>This value defaults to the value programmed in EEPROM. <div> <b>Note</b><br/>           SPI configuration is shared with channels 4-7         </div> |

### 7.6.2.10 SPI\_CHAN\_EN Register (Offset = 1009h) [Reset = 00h]

SPI\_CHAN\_EN is shown in [Figure 7-65](#) and described in [Table 7-107](#).

Return to the [Summary Table](#).

SPI channel enable register. A SPI channel maps to an individual chip select channel.

#### Note

The GPIO of the chip select channel needs to be set to special function in the IO\_CFG\_0 and IO\_CFG\_1 registers

**Figure 7-65. SPI\_CHAN\_EN Register**

| 7       | 6       | 5       | 4       | 3       | 2       | 1       | 0       |
|---------|---------|---------|---------|---------|---------|---------|---------|
| C7_EN   | C6_EN   | C5_EN   | C4_EN   | C3_EN   | C2_EN   | C1_EN   | C0_EN   |
| R/WP-0h | R/WP-0h | R/WP-0h | R/WP-0h | R/WP-0h | R/WP-0h | R/WP-0h | R/WP-0h |

**Table 7-107. SPI\_CHAN\_EN Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                         |
|-----|-------|------|-------|-----------------------------------------------------------------------------------------------------|
| 7   | C7_EN | R/WP | 0h    | CS Channel 7 enable<br>0h = The SPI channel is disabled/not used<br>1h = The SPI channel is enabled |
| 6   | C6_EN | R/WP | 0h    | CS Channel 6 enable<br>0h = The SPI channel is disabled/not used<br>1h = The SPI channel is enabled |
| 5   | C5_EN | R/WP | 0h    | CS Channel 5 enable<br>0h = The SPI channel is disabled/not used<br>1h = The SPI channel is enabled |
| 4   | C4_EN | R/WP | 0h    | CS Channel 4 enable<br>0h = The SPI channel is disabled/not used<br>1h = The SPI channel is enabled |
| 3   | C3_EN | R/WP | 0h    | CS Channel 3 enable<br>0h = The SPI channel is disabled/not used<br>1h = The SPI channel is enabled |
| 2   | C2_EN | R/WP | 0h    | CS Channel 2 enable<br>0h = The SPI channel is disabled/not used<br>1h = The SPI channel is enabled |
| 1   | C1_EN | R/WP | 0h    | CS Channel 1 enable<br>0h = The SPI channel is disabled/not used<br>1h = The SPI channel is enabled |
| 0   | C0_EN | R/WP | 0h    | CS Channel 0 enable<br>0h = The SPI channel is disabled/not used<br>1h = The SPI channel is enabled |

### 7.6.2.11 SPI\_CS\_POL Register (Offset = 100Ah) [Reset = 00h]

SPI\_CS\_POL is shown in [Figure 7-66](#) and described in [Table 7-108](#).

Return to the [Summary Table](#).

**Figure 7-66. SPI\_CS\_POL Register**

| 7    | 6 | 5 | 4 | 3       | 2       | 1       | 0       |
|------|---|---|---|---------|---------|---------|---------|
| RSVD |   |   |   | CS3_AH  | CS2_AH  | CS1_AH  | CS0_AH  |
| R-0h |   |   |   | R/WP-0h | R/WP-0h | R/WP-0h | R/WP-0h |

**Table 7-108. SPI\_CS\_POL Register Field Descriptions**

| Bit | Field  | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                         |
|-----|--------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-4 | RSVD   | R    | 0h    |                                                                                                                                                                                                                                                                                                                                                     |
| 3   | CS3_AH | R/WP | 0h    | Configures the active state of the CS pin for the channel<br><br><div style="text-align: center;"> <b>Note</b><br/>           CS3 chip select configuration is shared with CS4-7         </div> 0h = The chip select pin functionality is active-low (nCS or $\overline{\text{CS}}$ )<br>1h = The chip select pin functionality is active-high (CS) |
| 2   | CS2_AH | R/WP | 0h    | Configures the active state of the CS pin for the channel<br>0h = The chip select pin functionality is active-low (nCS or $\overline{\text{CS}}$ )<br>1h = The chip select pin functionality is active-high (CS)                                                                                                                                    |
| 1   | CS1_AH | R/WP | 0h    | Configures the active state of the CS pin for the channel<br>0h = The chip select pin functionality is active-low (nCS or $\overline{\text{CS}}$ )<br>1h = The chip select pin functionality is active-high (CS)                                                                                                                                    |
| 0   | CS0_AH | R/WP | 0h    | Configures the active state of the CS pin for the channel<br>0h = The chip select pin functionality is active-low (nCS or $\overline{\text{CS}}$ )<br>1h = The chip select pin functionality is active-high (CS)                                                                                                                                    |

### 7.6.2.12 SPI\_FIFO\_CTRL Register (Offset = 100Bh) [Reset = 0h]

SPI\_FIFO\_CTRL is shown in [Table 7-109](#).

Return to the [Summary Table](#).

**Table 7-109. SPI\_FIFO\_CTRL Register Field Descriptions**

| Bit | Field  | Type   | Reset | Description                                                                                                                             |
|-----|--------|--------|-------|-----------------------------------------------------------------------------------------------------------------------------------------|
| 2   | CLR_TX | RH/W1S | 0h    | Clear the transmit FIFO contents.<br>Hardware sets bit back to 0 when clear is complete.<br>0h = Do nothing<br>1h = Clear transmit FIFO |
| 1   | CLR_RX | RH/W1S | 0h    | Clear the receive FIFO contents.<br>Hardware sets bit back to 0 when clear is complete.<br>0h = Do nothing<br>1h = Clear receive FIFO   |
| 0   | RSVD   | R      | 0h    |                                                                                                                                         |

### 7.6.2.13 SPI\_IE\_0 Register (Offset = 100Ch) [Reset = 00h]

SPI\_IE\_0 is shown in [Figure 7-67](#) and described in [Table 7-110](#).

Return to the [Summary Table](#).

Interrupt enable bits for enabling certain interrupts for the INT0 pin. Interrupts that are enabled will be signaled on the INT0 pin. Note that INT0 functionality must be enabled in the INT\_CFG register to see output on the pin.

**Figure 7-67. SPI\_IE\_0 Register**

| 7      | 6      | 5    | 4 | 3 | 2      | 1       | 0      |
|--------|--------|------|---|---|--------|---------|--------|
| TXLIE0 | TXAIE0 | RSVD |   |   | RXLIE0 | RXFLIE0 | RXNIE0 |
| R/W-0h | R/W-0h | R-0h |   |   | R/W-0h | R/W-0h  | R/W-0h |

**Table 7-110. SPI\_IE\_0 Register Field Descriptions**

| Bit | Field   | Type | Reset | Description                                                                                                                                                                                                            |
|-----|---------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXLIE0  | R/W  | 0h    | TX lost message due to full FIFO interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output. |
| 6   | TXAIE0  | R/W  | 0h    | TX space available interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.               |
| 5-3 | RSVD    | R    | 0h    |                                                                                                                                                                                                                        |
| 2   | RXLIE0  | R/W  | 0h    | RX overrun/lost message interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.          |
| 1   | RXFLIE0 | R/W  | 0h    | RX fill level interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                    |
| 0   | RXNIE0  | R/W  | 0h    | RX new message interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                   |

#### 7.6.2.14 SPI\_IE\_1 Register (Offset = 100Dh) [Reset = 00h]

SPI\_IE\_1 is shown in [Figure 7-68](#) and described in [Table 7-111](#).

Return to the [Summary Table](#).

Interrupt enable bits for enabling certain interrupts for the INT0 pin. Interrupts that are enabled with be signaled on the INT1 pin. Note that INT0 functionality must be enabled in the INT\_CFG register to see output on the pin.

**Figure 7-68. SPI\_IE\_1 Register**

| 7      | 6      | 5    | 4 | 3 | 2      | 1       | 0      |
|--------|--------|------|---|---|--------|---------|--------|
| TXLIE1 | TXAIE1 | RSVD |   |   | RXLIE1 | RXFLIE1 | RXNIE1 |
| R/W-0h | R/W-0h | R-0h |   |   | R/W-0h | R/W-0h  | R/W-0h |

**Table 7-111. SPI\_IE\_1 Register Field Descriptions**

| Bit | Field   | Type | Reset | Description                                                                                                                                                                                                            |
|-----|---------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXLIE1  | R/W  | 0h    | TX lost message due to full FIFO interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output. |
| 6   | TXAIE1  | R/W  | 0h    | TX space available interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.               |
| 5-3 | RSVD    | R    | 0h    |                                                                                                                                                                                                                        |
| 2   | RXLIE1  | R/W  | 0h    | RX overrun/lost message interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.          |
| 1   | RXFLIE1 | R/W  | 0h    | RX fill level interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                    |
| 0   | RXNIE1  | R/W  | 0h    | RX new message interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                   |

### 7.6.2.15 SPI\_IR Register (Offset = 100Eh) [Reset = 00h]

SPI\_IR is shown in [Figure 7-69](#) and described in [Table 7-112](#).

Return to the [Summary Table](#).

Current interrupt register, which contains the bits for any interrupts that are currently set. Write 1 to clear.

**Figure 7-69. SPI\_IR Register**

| 7        | 6        | 5    | 4 | 3 | 2        | 1        | 0        |
|----------|----------|------|---|---|----------|----------|----------|
| TXL      | TXA      | RSVD |   |   | RXL      | RXFL     | RXN      |
| R/W1C-0h | R/W1C-0h | R-0h |   |   | R/W1C-0h | R/W1C-0h | R/W1C-0h |

**Table 7-112. SPI\_IR Register Field Descriptions**

| Bit | Field | Type  | Reset | Description                                                                                                                                                                                                                                  |
|-----|-------|-------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXL   | R/W1C | 0h    | TX lost message due to full FIFO interrupt<br>0h = No TX FIFO message was lost<br>1h = TX FIFO lost a message (overrun) due to FIFO being full when a write was attempted                                                                    |
| 6   | TXA   | R/W1C | 0h    | TX space available interrupt<br><br><b>Note</b><br>When the SPI IP is enabled, this interrupt will set since the FIFO will be empty<br><br>0h = No TX FIFO space available interrupt<br>1h = The TX FIFO has hit the trigger level threshold |
| 5-3 | RSVD  | R     | 0h    |                                                                                                                                                                                                                                              |
| 2   | RXL   | R/W1C | 0h    | RX overrun/lost message interrupt<br>0h = No RX FIFO messages lost (no overrun) interrupt<br>1h = At least 1 message has been lost due to full RX FIFO (overrun interrupt)                                                                   |
| 1   | RXFL  | R/W1C | 0h    | RX fill level interrupt.<br>Level is set by SPI_CTRL.RX_LVL_INT<br>0h = No RX FIFO fill level interrupt<br>1h = The RX FIFO reached the fill level                                                                                           |
| 0   | RXN   | R/W1C | 0h    | RX new message interrupt<br>0h = No new message received interrupt<br>1h = A new message has been received                                                                                                                                   |

### 7.6.2.16 SPI\_FS Register (Offset = 100Fh) [Reset = 80h]

SPI\_FS is shown in [Figure 7-70](#) and described in [Table 7-113](#).

Return to the [Summary Table](#).

SPI FIFO status

**Figure 7-70. SPI\_FS Register**

| 7      | 6     | 5 | 4 | 3 | 2 | 1 | 0     |
|--------|-------|---|---|---|---|---|-------|
| TXFSRE | RXFD  |   |   |   |   |   | RXFB  |
| RH-1h  | RH-0h |   |   |   |   |   | RH-0h |

**Table 7-113. SPI\_FS Register Field Descriptions**

| Bit | Field  | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-----|--------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXFSRE | RH   | 1h    | <p>Whether the TX FIFO and shift register is empty or not. This means that all queued data has been transmitted and the controller is idle. Nothing is in the FIFO, and nothing is in the shift register. If this is set, it is safe to configure or disable the SPI controller</p> <hr/> <p style="text-align: center;"><b>Note</b></p> <p>This does NOT take into consideration a TX FIFO element which is not been fully written to. If there is a partially written TX FIFO element, that untransmitted element does not clear the idle flag until the element has been fully written</p> <hr/> <p>0h = There is at least 1 ongoing and/or pending transmissions.<br/>1h = The TX FIFO and shift register are empty and IP is idle. It is safe to configure or turn off the controller</p> |
| 6   | RXFD   | RH   | 0h    | <p>Whether the RX FIFO contains 1 or more unread element (SPI transfer)<br/>0h = The RX FIFO is empty<br/>1h = At least 1 element in the RX FIFO</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 5-0 | RXFB   | RH   | 0h    | <p>The size of the SPI frame stored at the next RX FIFO element. Valid values are 0 to 63. Any value greater than 63 will be reflected as 63, and the remaining will be updated as the processor reads the data out.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

**7.6.2.17 SPI\_TX\_FIFO Register (Offset = 1010h) [Reset = 0000h]**

SPI\_TX\_FIFO is shown in [Figure 7-71](#) and described in [Table 7-114](#).

Return to the [Summary Table](#).

The SPI transmit FIFO. When performing a write to this address, the data is written into the SPI transmit FIFO. Please see [SPI Transmit FIFO](#) for more information.; This only shows the header for the SPI frame

**Figure 7-71. SPI\_TX\_FIFO Register**

|           |      |    |    |    |      |   |   |
|-----------|------|----|----|----|------|---|---|
| 15        | 14   | 13 | 12 | 11 | 10   | 9 | 8 |
| STORE_RX  | RSVD |    |    |    | CHAN |   |   |
| W-0h      | R-0h |    |    |    | W-0h |   |   |
| 7         | 6    | 5  | 4  | 3  | 2    | 1 | 0 |
| NUM_BYTES |      |    |    |    |      |   |   |
| W-0h      |      |    |    |    |      |   |   |

**Table 7-114. SPI\_TX\_FIFO Register Field Descriptions**

| Bit   | Field     | Type | Reset | Description                                                                                                                                              |
|-------|-----------|------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15    | STORE_RX  | W    | 0h    | Store the transfer in the RX FIFO.<br>Can be used to ignore incoming data if it is not needed<br><a href="#">SPI Transmit FIFO</a> for more information. |
| 14-11 | RSVD      | R    | 0h    | Reserved                                                                                                                                                 |
| 10-8  | CHAN      | W    | 0h    | SPI Channel to transmit on<br>The SPI Channel controls the chip select pin<br><a href="#">SPI Transmit FIFO</a> for more information.                    |
| 7-0   | NUM_BYTES | W    | 0h    | Number of bytes in the SPI transfer<br><a href="#">SPI Transmit FIFO</a> for more information.                                                           |

### 7.6.2.18 SPI\_RX\_FIFO Register (Offset = 1010h) [Reset = 0000h]

SPI\_RX\_FIFO is shown in [Figure 7-72](#) and described in [Table 7-115](#).

Return to the [Summary Table](#).

When performing a read to this address, the data is read from the SPI receive FIFO. Please see [SPI Receive FIFO](#) for more information.; This only shows the header for the SPI frame

**Figure 7-72. SPI\_RX\_FIFO Register**

|                     |      |    |    |    |      |   |   |
|---------------------|------|----|----|----|------|---|---|
| 15                  | 14   | 13 | 12 | 11 | 10   | 9 | 8 |
| CONT                | RSVD |    |    |    | CHAN |   |   |
| R-0h                | R-0h |    |    |    | R-0h |   |   |
| 7                   | 6    | 5  | 4  | 3  | 2    | 1 | 0 |
| NUM_BYTES_REMAINING |      |    |    |    |      |   |   |
| R-0h                |      |    |    |    |      |   |   |

**Table 7-115. SPI\_RX\_FIFO Register Field Descriptions**

| Bit   | Field               | Type | Reset | Description                                                                                                                                                                       |
|-------|---------------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15    | CONT                | R    | 0h    | Flag to indicate whether this is a partial RX FIFO element read. If 0, then this is the first read to this FIFO element<br><a href="#">SPI Receive FIFO</a> for more information. |
| 14-11 | RSVD                | R    | 0h    | Reserved                                                                                                                                                                          |
| 10-8  | CHAN                | R    | 0h    | SPI Channel the transfer was on<br><a href="#">SPI Receive FIFO</a> for more information.                                                                                         |
| 7-0   | NUM_BYTES_REMAINING | R    | 0h    | Number of bytes remaining to read in the FIFO element (including this transaction)<br><a href="#">SPI Receive FIFO</a> for more information.                                      |

### 7.6.2.19 SPI\_RXFS Register (Offset = 1011h) [Reset = 00h]

SPI\_RXFS is shown in [Figure 7-73](#) and described in [Table 7-116](#).

Return to the [Summary Table](#).

Contains the counter of how many unread messages are stored in the RX FIFO.

**Figure 7-73. SPI\_RXFS Register**

|       |   |   |   |   |   |   |   |
|-------|---|---|---|---|---|---|---|
| 7     | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RX_FL |   |   |   |   |   |   |   |
| R-0h  |   |   |   |   |   |   |   |

**Table 7-116. SPI\_RXFS Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                                                                                                                                                                     |
|-----|-------|------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | RX_FL | R    | 0h    | RX FIFO fill level states how many messages are currently in the RX FIFO.<br>Values range between 1 to 255.<br>Maximum value listed depends on how device memory is configured.<br>If 255 is reported, then 255 or more frames are in the FIFO. |

#### 7.6.2.20 SPI\_TXFS Register (Offset = 1012h) [Reset = 00h]

SPI\_TXFS is shown in [Figure 7-74](#) and described in [Table 7-117](#).

Return to the [Summary Table](#).

Contains the counter of how many available spaces are in the TX FIFO.

**Figure 7-74. SPI\_TXFS Register**

|       |   |   |   |   |   |   |   |
|-------|---|---|---|---|---|---|---|
| 7     | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TX_SA |   |   |   |   |   |   |   |
| R-0h  |   |   |   |   |   |   |   |

**Table 7-117. SPI\_TXFS Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                                                                                                                                           |
|-----|-------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | TX_SA | R    | 0h    | TX FIFO fill level states how many free bytes are currently in the TX FIFO.<br><br><div> <b>Note</b><br/>           If there are more than 255 bytes available, the space available will say 255 bytes         </div> |

**7.6.2.21 SPI\_TXES Register (Offset = 1013h) [Reset = 00h]**

SPI\_TXES is shown in [Figure 7-75](#) and described in [Table 7-118](#).

Return to the [Summary Table](#).

Contains information about the current/next transmit FIFO element

**Figure 7-75. SPI\_TXES Register**

| 7     | 6     | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-------|---|---|---|---|---|---|
| TXEIP | TXEBP |   |   |   |   |   |   |
| R-0h  | R-0h  |   |   |   |   |   |   |

**Table 7-118. SPI\_TXES Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-----|-------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXEIP | R    | 0h    | <p>TX FIFO element write in progress</p> <p>Used to signal if a TX FIFO element is partially written. If there are not enough data bytes that were written into the FIFO (according to the header), then this flag is set. If it is cleared, then the next write to the TX FIFO will be a new element and should start with the SPI header fields</p> <p>0h = TX element is not in progress (next write to TX FIFO is a new element)</p> <p>1h = TX element is in progress, and has an expected number of bytes left</p>                                                                                                                                                                                                                                                                                  |
| 6-0 | TXEBP | R    | 0h    | <p>TX Element Bytes Pending</p> <p>If a TX element write has started, this is how many bytes are still expected in order to complete the FIFO element. If the number of expected bytes remaining is more than 127 bytes, 127 bytes remaining is still shown (due to maximum number that can be shown)</p> <hr/> <p style="text-align: center;"><b>Note</b></p> <p>This includes header bytes and can change values if the number of bytes in the transfer has not been written yet.</p> <p>For example, if only the SPI channel header byte was written, this field will read 1, since we do not know how many bytes are pending until told how many bytes of data are to be transferred.</p> <p>Once the number of bytes in the transfer is known, this will recalculate as data is written in</p> <hr/> |

### 7.6.3 SPI Data FIFOs

First in first out data structures (FIFOs) are used to shift data out and into the device. There are two main FIFOs: the receive (RX) FIFO and the transmit (TX) FIFO. The address is the same to access either, but when a write is performed, the data is copied into the TX FIFO and when a read is performed, data is read out from the RX FIFO.

#### 7.6.3.1 SPI Transmit FIFO (address = h1010)

The transmit FIFO is write only and is written to when a write command is issued to its address. Transmission will begin when the entire frame has been written into the buffer. All data bytes written after the end of the frame is ignored until a new transmit FIFO write is started. All writes to a transmit FIFO should be directed at the main address only, as each byte that is written is automatically shifted into the FIFO. For example, writing larger SPI messages than can be transferred in a single CAN message occurs over multiple CAN writes to the same address, since the SPI header bytes will tell the device how many data bytes to expect.

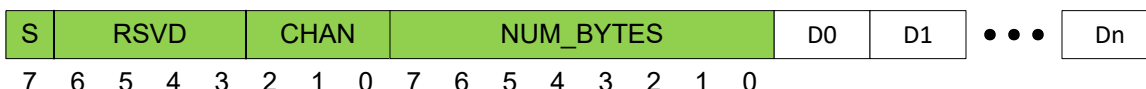
The size of each frame in the TX buffer is determined by the message length. There is a 2-byte header required at the beginning of each SPI FIFO element. The total length of the TX frame in the buffer is the sum of

- 2 bytes for the SPI header
- Message length of data bytes

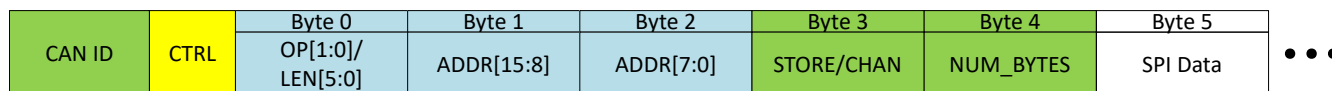
See [SPI Control Protocol](#) for more information and examples.

#### Note

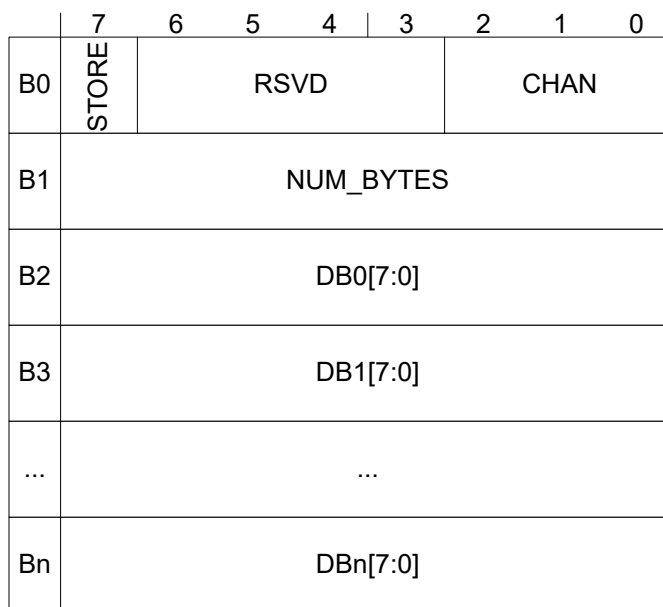
If the transmit FIFO is full, any new messages written to it are discarded and an interrupt is set.



**Figure 7-76. SPI TX FIFO Write Header (Header and Data Only)**



**Figure 7-77. Example SPI TX FIFO Header (Standard CAN Frame Format)**



**Figure 7-78. SPI Transmit FIFO**

**Table 7-119. SPI Transmit FIFO Descriptions**

| Byte | Bit | Field     | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                           |
|------|-----|-----------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0    | 7   | STORE_RX  | W    | 0     | Whether to store the POCI data for this message in the receive FIFO<br>0 = Do not store any data in the receive FIFO<br>1 = Save the received POCI data in the receive FIFO for this message                                                                                                                                                                          |
|      | 6:3 | RSVD      | R    | 0     | Reserved                                                                                                                                                                                                                                                                                                                                                              |
|      | 2:0 | CHAN      | W    | 0x0   | SPI Channel that was used for this message<br>Valid values 0-3<br>0 = SPI channel 0<br>1 = SPI channel 1<br>2 = SPI channel 2<br>3 = SPI channel 3<br>4 = SPI channel 4<br>5 = SPI channel 5<br>6 = SPI channel 6<br>7 = SPI channel 7<br><br><b>Note</b><br>A SPI channel corresponds to a specific GPIO and requires that GPIO to be assigned as a special function |
| 1    | 7:0 | NUM_BYTES | W    | 0x0   | Number of bytes in the SPI frame (does NOT include the above header bytes)<br>Valid values 1-255. 0 is invalid.                                                                                                                                                                                                                                                       |

### 7.6.3.2 SPI Receive FIFO (address = h1010)

The receive FIFO is read only.

The size of each frame in the RX buffer is determined by the message's data length. The total length of the RX frame in the buffer is the sum of

- 2 bytes for SPI header
- Message length of data bytes

A read from the RX FIFO is considered completed and freed when all data bytes have been read from the FIFO element. Any bytes beyond the number of bytes remaining for a read returns 0x00 for the invalid bytes.

A partial read of a frame from the RX FIFO will automatically be continued for the next read to the SPI FIFO, and is signaled as such with the CONT bit being set, as the NUM\_BYTES\_REMAINING reflecting the number of data bytes left in this message to read.

A read from an empty RX buffer returns 0 NUM\_BYTES\_REMAINING for channel 0. Frames will always be read in the order that they were received.

The RX buffer can be cleared (all frames discarded) by a write of 1 to the SPI Clear RX FIFO bit (SPI\_FIFO\_CTRL[1]).

#### Note

If the receive FIFO is full, all new incoming messages is discarded and an interrupt set to alert the processor that a message was lost.

See [SPI Control Protocol](#) for more information and examples.



**Figure 7-79. SPI RX FIFO Read Header (Header and Data Only)**



**Figure 7-80. Example SPI RX FIFO Header (Standard CAN Frame Format)**

|     |                     |      |   |   |   |      |   |   |
|-----|---------------------|------|---|---|---|------|---|---|
|     | 7                   | 6    | 5 | 4 | 3 | 2    | 1 | 0 |
| B0  | CONT                | RSVD |   |   |   | CHAN |   |   |
| B1  | NUM_BYTES_REMAINING |      |   |   |   |      |   |   |
| B2  | DB0[7:0]            |      |   |   |   |      |   |   |
| B3  | DB1[7:0]            |      |   |   |   |      |   |   |
| ... | ...                 |      |   |   |   |      |   |   |
| Bn  | DBn[7:0]            |      |   |   |   |      |   |   |

Figure 7-81. SPI Receive FIFO

Table 7-120. SPI Receive FIFO Descriptions (Start of SPI Header)

| Byte | Bit | Field               | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                           |
|------|-----|---------------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0    | 7   | CONT                | R    | 0     | A flag to indicate whether this is the first part of a message, or a continuation (used for reading messages that are split into multiple CAN reads)<br>0 = First part of a new message<br>1 = Continuation of a previous message read                                                                                                                                |
|      | 6:3 | RSVD                | R    | 0     | Reserved                                                                                                                                                                                                                                                                                                                                                              |
|      | 2:0 | CHAN                | R    | 0x0   | SPI Channel that was used for this message<br>Valid values 0-3<br>0 = SPI channel 0<br>1 = SPI channel 1<br>2 = SPI channel 2<br>3 = SPI channel 3<br>4 = SPI channel 4<br>5 = SPI channel 5<br>6 = SPI channel 6<br>7 = SPI channel 7<br><br><b>Note</b><br>A SPI channel corresponds to a specific GPIO and requires that GPIO to be assigned as a special function |
| 1    | 7:0 | NUM_BYTES_REMAINING | R    | 0x0   | Number of data bytes remaining to be read in this message (including the data bytes transmitted in the current frame). This does NOT include the header bytes. This lets the processor know how many bytes in the current CAN frame are valid, and if there another read is needed to get more data from the FIFO for this message.                                   |

#### 7.6.4 UART Registers

Section 7.6.4 lists the memory-mapped registers for the UART registers. All register offset addresses not listed in Section 7.6.4 should be considered as reserved locations and the register contents should not be modified.

**Table 7-121. UART Registers**

| Offset | Acronym            | Register Name               | Section                          |
|--------|--------------------|-----------------------------|----------------------------------|
| 2000h  | UART_CREL          | UART Core Release           | <a href="#">Section 7.6.4.1</a>  |
| 2001h  | UART_SCRATCH       | UART Scratchpad             | <a href="#">Section 7.6.4.2</a>  |
| 2002h  | UART_CTRL          | UART Control                | <a href="#">Section 7.6.4.3</a>  |
| 2003h  | UART_BR_LSB        | UART Baud Rate (LSB)        | <a href="#">Section 7.6.4.4</a>  |
| 2004h  | UART_BR_MSB        | UART Baud Rate (MSB)        | <a href="#">Section 7.6.4.5</a>  |
| 2005h  | UART_BR_FRAC       | UART Baud Rate (Fractional) | <a href="#">Section 7.6.4.6</a>  |
| 2006h  | UART_FIFO_CTRL     | UART FIFO Control           | <a href="#">Section 7.6.4.7</a>  |
| 2007h  | UART_IE_0          | UART Interrupt Enable 0     | <a href="#">Section 7.6.4.8</a>  |
| 2008h  | UART_IE_1          | UART Interrupt Enable 1     | <a href="#">Section 7.6.4.9</a>  |
| 2009h  | UART_IR            | UART Interrupt Register     | <a href="#">Section 7.6.4.10</a> |
| 200Ah  | UART_STATUS        | UART Status                 | <a href="#">Section 7.6.4.11</a> |
| 200Bh  | UART_RXFS          | UART RX FIFO Status         | <a href="#">Section 7.6.4.12</a> |
| 200Ch  | UART_TXFS          | UART TX FIFO Status         | <a href="#">Section 7.6.4.13</a> |
| 2010h  | UART_RX_FIFO       | UART RX FIFO                | <a href="#">Section 7.6.4.14</a> |
| 2010h  | UART_TX_FIFO       | UART TX FIFO                | <a href="#">Section 7.6.4.15</a> |
| 2011h  | UART_RX_ERR_STATUS | UART RX FIFO Error Status   | <a href="#">Section 7.6.4.16</a> |

Complex bit access types are encoded to fit into small table cells. [Section 7.6.4](#) shows the codes that are used for access types in this section.

**Table 7-122. UART Access Type Codes**

| Access Type            | Code    | Description                            |
|------------------------|---------|----------------------------------------|
| Read Type              |         |                                        |
| R                      | R       | Read                                   |
| RH                     | R<br>H  | Read<br>Set or cleared by hardware     |
| Write Type             |         |                                        |
| W                      | W       | Write                                  |
| W1C                    | W<br>1C | Write<br>1 to clear                    |
| W1S                    | W<br>1S | Write<br>1 to set                      |
| WP                     | W<br>P  | Write<br>Requires privileged access    |
| Reset or Default Value |         |                                        |
| -n                     |         | Value after reset or the default value |

#### 7.6.4.1 UART\_CREL Register (Offset = 2000h) [Reset = 10h]

UART\_CREL is shown in [Figure 7-82](#) and described in [Table 7-123](#).

Return to the [Summary Table](#).

UART IP release version

**Figure 7-82. UART\_CREL Register**

| 7     | 6 | 5 | 4 | 3     | 2 | 1 | 0 |
|-------|---|---|---|-------|---|---|---|
| MAJOR |   |   |   | MINOR |   |   |   |
| R-1h  |   |   |   | R-0h  |   |   |   |

**Table 7-123. UART\_CREL Register Field Descriptions**

| Bit | Field | Type | Reset | Description                               |
|-----|-------|------|-------|-------------------------------------------|
| 7-4 | MAJOR | R    | 1h    | The major version number of this revision |
| 3-0 | MINOR | R    | 0h    | The minor version number of this revision |

#### 7.6.4.2 UART\_SCRATCH Register (Offset = 2001h) [Reset = 00h]

UART\_SCRATCH is shown in [Figure 7-83](#) and described in [Table 7-124](#).

Return to the [Summary Table](#).

Customer Scratchpad register

**Figure 7-83. UART\_SCRATCH Register**

|            |   |   |   |   |   |   |   |
|------------|---|---|---|---|---|---|---|
| 7          | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SCRATCHPAD |   |   |   |   |   |   |   |
| R/W-0h     |   |   |   |   |   |   |   |

**Table 7-124. UART\_SCRATCH Register Field Descriptions**

| Bit | Field      | Type | Reset | Description                                                                                                                                                                                                   |
|-----|------------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | SCRATCHPAD | R/W  | 0h    | A customer scratchpad register.<br>This can be used to validate the communication by writing and reading the value or can be used to store some data.<br>This is NOT retained through resets/power on resets. |

#### 7.6.4.3 UART\_CTRL Register (Offset = 2002h) [Reset = 04h]

UART\_CTRL is shown in [Figure 7-84](#) and described in [Table 7-125](#).

Return to the [Summary Table](#).

This register controls the data communication format. The word length, number of stop bits, and parity type, as well as enabling changes to UART configuration.

**Figure 7-84. UART\_CTRL Register**

| 7      | 6       | 5       | 4       | 3       | 2       | 1       | 0      |
|--------|---------|---------|---------|---------|---------|---------|--------|
| BRKGEN | FPAREN  | PAR     | PEN     | 2BSTOP  | 8BIT    | UART_EN | CCE    |
| R/W-0h | R/WP-0h | R/WP-0h | R/WP-0h | R/WP-0h | R/WP-1h | RH-0h   | R/W-0h |

**Table 7-125. UART\_CTRL Register Field Descriptions**

| Bit | Field   | Type | Reset | Description                                                                                                                                                                                                                                                                                       |
|-----|---------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | BRKGEN  | R/W  | 0h    | Break control bit<br>0h = Normal UART operation<br>1h = Force TXD low to generate a break                                                                                                                                                                                                         |
| 6   | FPAREN  | R/WP | 0h    | Force parity bit (if enabled) to be a specific value<br>0h = Generate parity as configured<br>1h = Force parity bit as defined in parity bit (PAR) configuration for transmitted and received data                                                                                                |
| 5   | PAR     | R/WP | 0h    | Odd or even parity configuration<br>0h = Odd parity (1 bit value)<br>1h = Even parity (0 bit value)                                                                                                                                                                                               |
| 4   | PEN     | R/WP | 0h    | Enables if parity is used for each character<br>0h = No parity bit<br>1h = Parity bit enabled                                                                                                                                                                                                     |
| 3   | 2BSTOP  | R/WP | 0h    | Defines the number of stop bits used per character<br>0h = 1 stop bit<br>1h = 2 stop bits                                                                                                                                                                                                         |
| 2   | 8BIT    | R/WP | 1h    | Defines the size of a character<br>0h = 7-bit data<br>1h = 8-bit data                                                                                                                                                                                                                             |
| 1   | UART_EN | RH   | 0h    | UART IP enable status flag.<br>This flag is not writable, but is set if the UART IP is enabled by allocating memory from MRAM to the IP.<br>This is done with the <a href="#">MRAM_IP_CFG</a> register<br>0h = UART IP is disabled, all UART functionality is disabled<br>1h = UART IP is enabled |
| 0   | CCE     | R/W  | 0h    | UART IP change control enable bit.<br>Can be set only if UART_EN is 0<br>0h = UART configuration registers are write protected<br>1h = UART configuration registers can be changed                                                                                                                |

#### 7.6.4.4 UART\_BR\_LSB Register (Offset = 2003h) [Reset = 00h]

UART\_BR\_LSB is shown in [Figure 7-85](#) and described in [Table 7-126](#).

Return to the [Summary Table](#).

The lower byte of the 16-bit divisor used for the baud clock

**Figure 7-85. UART\_BR\_LSB Register**

|         |   |   |   |   |   |   |   |
|---------|---|---|---|---|---|---|---|
| 7       | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| DLL     |   |   |   |   |   |   |   |
| R/WP-0h |   |   |   |   |   |   |   |

**Table 7-126. UART\_BR\_LSB Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                     |
|-----|-------|------|-------|---------------------------------------------------------------------------------|
| 7-0 | DLL   | R/WP | 0h    | The least significant byte of the 16-bit divisor used for baud rate generation. |

#### 7.6.4.5 UART\_BR\_MSB Register (Offset = 2004h) [Reset = 00h]

UART\_BR\_MSB is shown in [Figure 7-86](#) and described in [Table 7-127](#).

Return to the [Summary Table](#).

The upper byte of the 16-bit divisor used for the baud clock

**Figure 7-86. UART\_BR\_MSB Register**

| 7       | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---------|---|---|---|---|---|---|---|
| DLH     |   |   |   |   |   |   |   |
| R/WP-0h |   |   |   |   |   |   |   |

**Table 7-127. UART\_BR\_MSB Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                    |
|-----|-------|------|-------|--------------------------------------------------------------------------------|
| 7-0 | DLH   | R/WP | 0h    | The most significant byte of the 16-bit divisor used for baud rate generation. |

#### 7.6.4.6 UART\_BR\_FRAC Register (Offset = 2005h) [Reset = 00h]

UART\_BR\_FRAC is shown in [Figure 7-87](#) and described in [Table 7-128](#).

Return to the [Summary Table](#).

The 6-bit fractional divisor used for the baud clock (which is a x/64 value)

**Figure 7-87. UART\_BR\_FRAC Register**

|         |      |         |   |   |   |   |   |
|---------|------|---------|---|---|---|---|---|
| 7       | 6    | 5       | 4 | 3 | 2 | 1 | 0 |
| BD      | RSVD | DLF     |   |   |   |   |   |
| R/WP-0h | R-0h | R/WP-0h |   |   |   |   |   |

**Table 7-128. UART\_BR\_FRAC Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                                                                                      |
|-----|-------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | BD    | R/WP | 0h    | Baud divider bit for controlling the over-sampling used by the baud generation.<br>0h = Enable divide-by-16 baud divider<br>1h = Enable divide-by-8 baud divider |
| 6   | RSVD  | R    | 0h    |                                                                                                                                                                  |
| 5-0 | DLF   | R/WP | 0h    | The fractional divisor used for baud rate generation (x / 64).                                                                                                   |

### 7.6.4.7 UART\_FIFO\_CTRL Register (Offset = 2006h) [Reset = 01h]

UART\_FIFO\_CTRL is shown in [Figure 7-88](#) and described in [Table 7-129](#).

Return to the [Summary Table](#).

FIFO control fields to enable the FIFOs or clear the contents of the FIFOs.

**Figure 7-88. UART\_FIFO\_CTRL Register**

| 7       | 6 | 5       | 4 | 3    | 2         | 1         | 0       |
|---------|---|---------|---|------|-----------|-----------|---------|
| RX_TRG  |   | TX_TRG  |   | RSVD | CLR_TX    | CLR_RX    | FIFO_EN |
| R/WP-0h |   | R/WP-0h |   | R-0h | RH/W1S-0h | RH/W1S-0h | R/WP-1h |

**Table 7-129. UART\_FIFO\_CTRL Register Field Descriptions**

| Bit | Field   | Type   | Reset | Description                                                                                                                                                                          |
|-----|---------|--------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-6 | RX_TRG  | R/WP   | 0h    | Sets the trigger level for the RX FIFO<br>0h = 1 byte in RX FIFO<br>1h = 4 bytes in RX FIFO<br>2h = (max size) - 16 bytes in RX FIFO<br>3h = (max size) - 8 bytes in RX FIFO         |
| 5-4 | TX_TRG  | R/WP   | 0h    | Sets the trigger level for the number of free spaces in the TX FIFO<br>0h = 16 bytes free<br>1h = 32 bytes free<br>2h = (max size) - 16 bytes free<br>3h = (max size) - 8 bytes free |
| 3   | RSVD    | R      | 0h    |                                                                                                                                                                                      |
| 2   | CLR_TX  | RH/W1S | 0h    | Clear the transmit FIFO contents.<br>Hardware sets bit back to 0 when clear is complete.<br>0h = Do nothing<br>1h = Clear transmit FIFO                                              |
| 1   | CLR_RX  | RH/W1S | 0h    | Clear the receive FIFO contents.<br>Hardware sets bit back to 0 when clear is complete.<br>0h = Do nothing<br>1h = Clear receive FIFO                                                |
| 0   | FIFO_EN | R/WP   | 1h    | Enables the TX and RX FIFOs<br>0h = Disable the transmit and receive FIFOs<br>1h = Enable the transmit and receive FIFOs                                                             |

#### 7.6.4.8 UART\_IE\_0 Register (Offset = 2007h) [Reset = 00h]

UART\_IE\_0 is shown in [Figure 7-89](#) and described in [Table 7-130](#).

Return to the [Summary Table](#).

Interrupt enable bits for enabling certain interrupts for the INT0 pin. Interrupts that are enabled here will be signaled on the INT0 pin. Note that the INT0 functionality must be enabled.

**Figure 7-89. UART\_IE\_0 Register**

| 7      | 6      | 5       | 4       | 3       | 2      | 1       | 0      |
|--------|--------|---------|---------|---------|--------|---------|--------|
| TXLIE0 | TXAIE0 | RXBRIE0 | RXFEIE0 | RXPEIE0 | RXLIE0 | RXFLIE0 | RXNIE0 |
| R/W-0h | R/W-0h | R/W-0h  | R/W-0h  | R/W-0h  | R/W-0h | R/W-0h  | R/W-0h |

**Table 7-130. UART\_IE\_0 Register Field Descriptions**

| Bit | Field   | Type | Reset | Description                                                                                                                                                                                                         |
|-----|---------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXLIE0  | R/W  | 0h    | TX lost byte due to being full interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output. |
| 6   | TXAIE0  | R/W  | 0h    | TX space available interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.             |
| 5   | RXBRIE0 | R/W  | 0h    | RX break received interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.              |
| 4   | RXFEIE0 | R/W  | 0h    | RX framing error interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.               |
| 3   | RXPEIE0 | R/W  | 0h    | RX parity error interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.                |
| 2   | RXLIE0  | R/W  | 0h    | RX overrun/lost byte interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.           |
| 1   | RXFLIE0 | R/W  | 0h    | RX fill level interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.                  |
| 0   | RXNIE0  | R/W  | 0h    | RX new byte interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.                    |

#### 7.6.4.9 UART\_IE\_1 Register (Offset = 2008h) [Reset = 00h]

UART\_IE\_1 is shown in [Figure 7-90](#) and described in [Table 7-131](#).

Return to the [Summary Table](#).

Interrupt enable bits for enabling certain interrupts for the INT1 pin. Interrupts that are enabled here will be signaled on the INT1 pin. Note that the INT1 functionality must be enabled.

**Figure 7-90. UART\_IE\_1 Register**

| 7      | 6      | 5       | 4       | 3       | 2      | 1       | 0      |
|--------|--------|---------|---------|---------|--------|---------|--------|
| TXLIE1 | TXAIE1 | RXBRIE1 | RXFEIE1 | RXPEIE1 | RXLIE1 | RXFLIE1 | RXNIE0 |
| R/W-0h | R/W-0h | R/W-0h  | R/W-0h  | R/W-0h  | R/W-0h | R/W-0h  | R/W-0h |

**Table 7-131. UART\_IE\_1 Register Field Descriptions**

| Bit | Field   | Type | Reset | Description                                                                                                                                                                                                         |
|-----|---------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXLIE1  | R/W  | 0h    | TX lost byte due to being full interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output. |
| 6   | TXAIE1  | R/W  | 0h    | TX space available interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.             |
| 5   | RXBRIE1 | R/W  | 0h    | RX break received interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.              |
| 4   | RXFEIE1 | R/W  | 0h    | RX framing error interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.               |
| 3   | RXPEIE1 | R/W  | 0h    | RX parity error interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.                |
| 2   | RXLIE1  | R/W  | 0h    | RX overrun/lost byte interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.           |
| 1   | RXFLIE1 | R/W  | 0h    | RX fill level interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.                  |
| 0   | RXNIE0  | R/W  | 0h    | RX new byte interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output.                    |

#### 7.6.4.10 UART\_IR Register (Offset = 2009h) [Reset = 00h]

UART\_IR is shown in [Figure 7-91](#) and described in [Table 7-132](#).

Return to the [Summary Table](#).

Pending interrupt register, which contains the bits for any interrupts that are currently set. Write 1 to clear.

**Figure 7-91. UART\_IR Register**

| 7        | 6        | 5        | 4        | 3        | 2        | 1        | 0        |
|----------|----------|----------|----------|----------|----------|----------|----------|
| TXL      | TXA      | RXBR     | RXFE     | RXPE     | RXL      | RXFL     | RXN      |
| R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h |

**Table 7-132. UART\_IR Register Field Descriptions**

| Bit | Field | Type  | Reset | Description                                                                                                                                                                                                                                                                                                |
|-----|-------|-------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXL   | R/W1C | 0h    | TX lost byte due to being full<br>0h = No TX FIFO byte was lost<br>1h = TX FIFO lost a byte (overrun) due to FIFO being full when a write was attempted                                                                                                                                                    |
| 6   | TXA   | R/W1C | 0h    | TX space available<br><br><div style="text-align: center;"><b>Note</b></div> <div style="text-align: center;">When the UART IP is enabled, this interrupt will set since the FIFO will be empty</div><br>0h = No TX FIFO space available interrupt<br>1h = The TX FIFO has hit the trigger level threshold |
| 5   | RXBR  | R/W1C | 0h    | RX break received<br>0h = No break was received<br>1h = A break was received                                                                                                                                                                                                                               |
| 4   | RXFE  | R/W1C | 0h    | RX framing error<br>0h = No received framing error has occurred<br>1h = RX framing error has occurred                                                                                                                                                                                                      |
| 3   | RXPE  | R/W1C | 0h    | RX parity error<br>0h = No received parity error has occurred<br>1h = RX parity error has occurred                                                                                                                                                                                                         |
| 2   | RXL   | R/W1C | 0h    | RX overrun/lost byte<br>0h = No RX FIFO bytes lost (no overrun) interrupt<br>1h = At least 1 byte has been lost due to full RX FIFO (overrun interrupt)                                                                                                                                                    |
| 1   | RXFL  | R/W1C | 0h    | RX fill level<br>0h = No RX FIFO fill level interrupt<br>1h = The RX FIFO reached the fill level                                                                                                                                                                                                           |
| 0   | RXN   | R/W1C | 0h    | RX new byte<br>0h = No new byte received interrupt<br>1h = A new byte has been received                                                                                                                                                                                                                    |

#### 7.6.4.11 UART\_STATUS Register (Offset = 200Ah) [Reset = 60h]

UART\_STATUS is shown in [Figure 7-92](#) and described in [Table 7-133](#).

Return to the [Summary Table](#).

**Figure 7-92. UART\_STATUS Register**

| 7     | 6      | 5     | 4    | 3 | 2 | 1 | 0     |
|-------|--------|-------|------|---|---|---|-------|
| RXBS  | TXFSRE | TXFE  | RSVD |   |   |   | RXFB  |
| RH-0h | RH-1h  | RH-1h | R-0h |   |   |   | RH-0h |

**Table 7-133. UART\_STATUS Register Field Descriptions**

| Bit | Field  | Type | Reset | Description                                                                                                                                                                                                                                                                             |
|-----|--------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | RXBS   | RH   | 0h    | There is a parity error, framing error, or break for at least 1 byte in the RX FIFO<br>0h = No parity error, framing error, or break on any bytes in RX FIFO<br>1h = At least 1 byte with a parity error, framing error, or break in the RX FIFO                                        |
| 6   | TXFSRE | RH   | 1h    | Whether the TX FIFO and shift register is empty or not. This means that all queued data has been shifted out. Nothing is in the FIFO, and nothing is in the shift register<br>0h = At least 1 element in the TX FIFO or shift register<br>1h = The TX FIFO and shift register are empty |
| 5   | TXFE   | RH   | 1h    | Whether the TX FIFO is empty or not, but not necessarily the TX shift register<br>0h = At least 1 element in the TX FIFO<br>1h = The TX FIFO is empty                                                                                                                                   |
| 4-1 | RSVD   | R    | 0h    |                                                                                                                                                                                                                                                                                         |
| 0   | RXFB   | RH   | 0h    | Whether the RX FIFO contains 1 or more unread bytes<br>0h = The RX FIFO is empty<br>1h = At least 1 element in the RX FIFO                                                                                                                                                              |

#### 7.6.4.12 UART\_RXFS Register (Offset = 200Bh) [Reset = 00h]

UART\_RXFS is shown in [Figure 7-93](#) and described in [Table 7-134](#).

Return to the [Summary Table](#).

Contains the counter of how many unread messages are stored in the RX FIFO.

**Figure 7-93. UART\_RXFS Register**

|       |   |   |   |   |   |   |   |
|-------|---|---|---|---|---|---|---|
| 7     | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RX_FL |   |   |   |   |   |   |   |
| R-0h  |   |   |   |   |   |   |   |

**Table 7-134. UART\_RXFS Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                                                                                                                                                             |
|-----|-------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | RX_FL | R    | 0h    | RX FIFO fill level states how many bytes are currently in the RX FIFO.<br>Valid values are 0 to 255.<br>Maximum value listed depends on how device memory is configured.<br>If 255 is reported, then 255 or more bytes are in the FIFO. |

#### 7.6.4.13 UART\_TXFS Register (Offset = 200Ch) [Reset = 00h]

UART\_TXFS is shown in [Figure 7-94](#) and described in [Table 7-135](#).

Return to the [Summary Table](#).

Contains the counter of how many available spaces are in the TX FIFO.

**Figure 7-94. UART\_TXFS Register**

|       |   |   |   |   |   |   |   |
|-------|---|---|---|---|---|---|---|
| 7     | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TX_SA |   |   |   |   |   |   |   |
| R-0h  |   |   |   |   |   |   |   |

**Table 7-135. UART\_TXFS Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                 |
|-----|-------|------|-------|-----------------------------------------------------------------------------|
| 7-0 | TX_SA | R    | 0h    | TX FIFO fill level states how many free spaces are currently in the TX FIFO |

#### 7.6.4.14 UART\_RX\_FIFO Register (Offset = 2010h) [Reset = 00h]

UART\_RX\_FIFO is shown in [Figure 7-95](#) and described in [Table 7-136](#).

Return to the [Summary Table](#).

Reads the next FIFO element and returns up to the number of requested bytes of data that will fit in the CAN message. A status byte is appended to the end of the data which shows if any of the bytes in the read data section have a non-normal status.

**Figure 7-95. UART\_RX\_FIFO Register**

| 7    | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|---|---|---|
| DATA |   |   |   |   |   |   |   |
| R-0h |   |   |   |   |   |   |   |

**Table 7-136. UART\_RX\_FIFO Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-----|-------|------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | DATA  | R    | 0h    | <p>Reads out the next byte of data in the RX FIFO.</p> <p>It's recommended reading the UART_RXFS register to see how many bytes are in the FIFO before performing a burst read.</p> <p>A burst read to this register will return multiple bytes of data up to the number of requested bytes.</p> <p>At the end of the data bytes, a global status byte is appended, signaling if there were any errors in recorded in the bytes of data read out.</p> <p>If the status byte returns a non-normal status, a read of same length to the UART_RX_ERR_STATUS register will give a status byte for each corresponding data byte that was read out, allowing the user to determine which byte had a non-normal status.</p> |

**7.6.4.15 UART\_TX\_FIFO Register (Offset = 2010h) [Reset = 00h]**

UART\_TX\_FIFO is shown in [Figure 7-96](#) and described in [Table 7-137](#).

Return to the [Summary Table](#).

Writes to the UART transmit FIFO will queue bytes of data to send as soon as they are written into the buffer.

**Figure 7-96. UART\_TX\_FIFO Register**

| 7    | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|------|---|---|---|---|---|---|---|
| DATA |   |   |   |   |   |   |   |
| W-0h |   |   |   |   |   |   |   |

**Table 7-137. UART\_TX\_FIFO Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                                                                                                                |
|-----|-------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | DATA  | W    | 0h    | Writing data to this FIFO will begin to shift data out to the UART bus. It's suggested that the user read the UART_TXFS register to see how many spots are available before doing a write. |

#### 7.6.4.16 UART\_RX\_ERR\_STATUS Register (Offset = 2011h) [Reset = 00h]

UART\_RX\_ERR\_STATUS is shown in [Figure 7-97](#) and described in [Table 7-138](#).

Return to the [Summary Table](#).

This is to be read after reading the UART\_RX\_FIFO element. If the status byte returns that there was a non-normal status for at least 1 byte, reading the same length of data from this register will return the status bytes for each byte that was last read from the RX FIFO.

**Figure 7-97. UART\_RX\_ERR\_STATUS Register**

| 7    | 6 | 5 | 4    | 3    | 2    | 1     | 0      |
|------|---|---|------|------|------|-------|--------|
| RSVD |   |   | BRK  | FE   | PAR  | NO_RX | NO_ERR |
| R-0h |   |   | R-0h | R-0h | R-0h | R-0h  | R-0h   |

**Table 7-138. UART\_RX\_ERR\_STATUS Register Field Descriptions**

| Bit | Field  | Type | Reset | Description                                                                                                                                                                                                                                        |
|-----|--------|------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD   | R    | 0h    |                                                                                                                                                                                                                                                    |
| 4   | BRK    | R    | 0h    | Set when a break condition was detected.<br>The data byte is returned as 0x00 for a break<br>0h = Not a break condition<br>1h = Is a break                                                                                                         |
| 3   | FE     | R    | 0h    | When set, the byte was received with a framing error (invalid stop bit count)<br>0h = Byte framing was valid<br>1h = Byte framing was invalid                                                                                                      |
| 2   | PAR    | R    | 0h    | When set, the byte was received with an invalid parity bit<br>0h = Parity bit is valid<br>1h = Parity bit is invalid                                                                                                                               |
| 1   | NO_RX  | R    | 0h    | Reads of an empty RX FIFO will return 0x00, and this bit signals that the data is invalid/not received.<br>This signals the absence of reception<br>0h = Not an empty/not received byte<br>1h = This byte is invalid/was not received and is empty |
| 0   | NO_ERR | R    | 0h    | When set, signals that the byte was received without any errors and is a valid byte.<br>This bit will be set if no other bits are set<br>0h = There is some non-normal status<br>1h = Successful reception of byte                                 |

#### 7.6.5 UART Data FIFOs

FIFOs are used to transfer data for UART. There are a few FIFOs used for the interface

- UART Transmit FIFO (h2010): Used to queue bytes of data to transmit onto the UART interface
- UART Receive FIFO (h2010): Used to read bytes received on UART, with a global status byte.
- UART Receive Error Status (h2011): Used to check the status of each byte received in the previous read

##### 7.6.5.1 UART Transmit FIFO (address = h2010)

The transmit FIFO is write only and is written to when a write command is issued to its address. Transmission will begin as soon as a byte is written into the FIFO if the module is idle. The UART interface deals with data on a per-byte basis.

See [UART Control Protocol](#) for more information and examples.

#### Note

If the transmit FIFO is full, any new bytes written to it are discarded and an interrupt is set.

**Table 7-139. UART Transmit FIFO Descriptions**

| Bit | Field | Type | Reset | Description                                                                                                                                                                                |
|-----|-------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7:0 | DATA  | W    | 0     | Writing data to this FIFO will begin to shift data out to the UART bus. It's suggested that the user read the UART_TXFS register to see how many spots are available before doing a write. |

### 7.6.5.2 UART Receive FIFO (address = h2010)

The receive FIFO is read only and reads out the next byte of data in the RX FIFO. It's recommended reading the UART\_RXFS register to see how many bytes are in the FIFO before performing a burst read. A burst read to this register returns multiple bytes of data up to the number of requested bytes.

At the end of the data bytes, a global status byte is appended, which is the logical bitwise OR of all bytes statuses. This is used to signal if any of the bytes that were read have any non-normal statuses. If the status byte returns a non-normal status, a read to the UART\_RX\_ERR\_STATUS register gives a status byte for each corresponding data byte that was read most recently, allowing the user to determine which byte had a non-normal status.

A normal global status byte returns 0x01. A 0x00 denotes that invalid/read overrun has occurred.

A read from an empty RX buffer will return 0 and a status byte of 0. Frames are always read in the order that received.

The RX buffer can be cleared (all frames discarded) by a write of 1 to the UART Clear RX FIFO bit (UART\_FIFO\_CTRL[1]).

See [UART Control Protocol](#) for more information and examples.

#### Note

If the receive FIFO is full, all new incoming messages is discarded and an interrupt set to alert the processor that a message was lost.

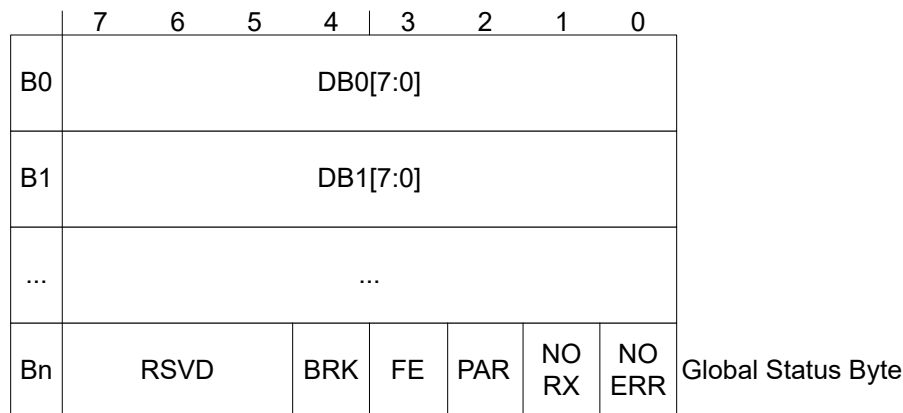


Figure 7-98. UART Receive FIFO Burst Read

Table 7-140. UART Receive FIFO Descriptions (with Global Status Byte)

| Byte     | Bit | Field | Type | Reset | Description           |
|----------|-----|-------|------|-------|-----------------------|
| 0 to N-1 | 7:0 | DATA  | R    | 0     | Byte of data received |

**Table 7-140. UART Receive FIFO Descriptions (with Global Status Byte) (continued)**

| Byte     | Bit | Field  | Type | Reset | Description                                                                                                                                                        |
|----------|-----|--------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| N (Last) | 7:5 | RSVD   | R    | 0x0   | Reserved                                                                                                                                                           |
|          | 4   | BRK    | R    | 0     | Set when a break condition was detected. The data byte is returned as 0x00 for a break<br>0 = No break detected in bytes<br>1 = At least one byte is a break       |
|          | 3   | FRAME  | R    | 0     | When set, a byte was received with a framing error (invalid stop bit count)<br>0 = No framing error detected in bytes<br>1 = At least one byte had a framing error |
|          | 2   | PAR    | R    | 0     | Parity Error. When set, at least 1 byte was received with an invalid parity bit.<br>0 = All parity bits were valid<br>1 = At least one parity bit was incorrect    |
|          | 1   | NO_RX  | R    | 0     | Empty Byte/No RX. When set, at least 1 byte was an invalid byte and a read overrun has occurred.<br>0 = All bytes are valid<br>1 = At least one byte is invalid    |
|          | 0   | NO_ERR | R    | 0     | No Error. When set, all bytes were received successfully.<br>0 = At least one byte has a non-normal status<br>1 = All bytes were received successfully.            |

### 7.6.5.3 UART Receive Error Status (address = h2011)

The UART receive error status register operates similar to the UART receive FIFO, except that it returns the individual status bytes for each UART byte that was read most recently.

When the processor reads from the UART RX FIFO, the status bytes for the corresponding bytes are loaded into the UART receive error status [FIFO]. The general procedure is to read from the UART RX FIFO, checking the global status register to see if any bytes reported a non-normal status. If a non-normal status byte is returned, then the processor should read the UART receive error status for the number of UART data bytes that were read from the UART RX FIFO. This returns the status byte of each UART data byte that was read most recently from the UART RX FIFO.

The contents of this register are cleared once the UART RX FIFO is read, updating the contents of this register with the status bytes of the latest read UART data bytes.

See [UART Control Protocol](#) for more information and examples. The status byte values are shown below.

**Figure 7-99. UART RX FIFO Error Status Register**

| 7    | 6 | 5 | 4    | 3    | 2    | 1     | 0      |
|------|---|---|------|------|------|-------|--------|
| RSVD |   |   | BRK  | FE   | PAR  | NO_RX | NO_ERR |
| R-0h |   |   | R-0h | R-0h | R-0h | R-0h  | R-0h   |

**Table 7-141. UART RX FIFO Error Status Register Field Descriptions**

| Bit | Field  | Type | Default | Description                                                                                                                                                                                                                                        |
|-----|--------|------|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD   | R    | 0h      |                                                                                                                                                                                                                                                    |
| 4   | BRK    | R    | 0h      | Set when a break condition was detected.<br>The data byte is returned as 0x00 for a break<br>0h = Not a break condition<br>1h = Is a break                                                                                                         |
| 3   | FE     | R    | 0h      | When set, the byte was received with a framing error (invalid stop bit count)<br>0h = Byte framing was valid<br>1h = Byte framing was invalid                                                                                                      |
| 2   | PAR    | R    | 0h      | When set, the byte was received with an invalid parity bit<br>0h = Parity bit is valid<br>1h = Parity bit is invalid                                                                                                                               |
| 1   | NO_RX  | R    | 0h      | Reads of an empty RX FIFO will return 0x00, and this bit signals that the data is invalid/not received.<br>This signals the absence of reception<br>0h = Not an empty/not received byte<br>1h = This byte is invalid/was not received and is empty |
| 0   | NO_ERR | R    | 0h      | When set, signals that the byte was received without any errors and is a valid byte.<br>This bit is set if no other bits are set<br>0h = There is some non-normal status<br>1h = Successful reception of byte                                      |

### 7.6.6 I2C Registers

[Section 7.6.6](#) lists the memory-mapped registers for the I2C registers. All register offset addresses not listed in [Section 7.6.6](#) should be considered as reserved locations and the register contents should not be modified.

**Table 7-142. I2C Registers**

| Offset | Acronym       | Register Name    | Section                         |
|--------|---------------|------------------|---------------------------------|
| 3000h  | I2C_CREL      | I2C Core Release | <a href="#">Section 7.6.6.1</a> |
| 3001h  | I2C_SCRATCH   | I2C Scratchpad   | <a href="#">Section 7.6.6.2</a> |
| 3002h  | I2C_CTRL      | I2C Control      | <a href="#">Section 7.6.6.3</a> |
| 3003h  | I2C_BR        | I2C Baud Rate    | <a href="#">Section 7.6.6.4</a> |
| 3004h  | I2C_FIFO_CTRL | I2C FIFO Control | <a href="#">Section 7.6.6.5</a> |

**Table 7-142. I2C Registers (continued)**

| Offset | Acronym    | Register Name          | Section                          |
|--------|------------|------------------------|----------------------------------|
| 3005h  | I2C_IE_0   | I2C Interrupt Enable 0 | <a href="#">Section 7.6.6.6</a>  |
| 3006h  | I2C_IE_1   | I2C Interrupt Enable 1 | <a href="#">Section 7.6.6.7</a>  |
| 3007h  | I2C_IR     | I2C Interrupt Status   | <a href="#">Section 7.6.6.8</a>  |
| 3008h  | I2C_STATUS | I2C Status             | <a href="#">Section 7.6.6.9</a>  |
| 3009h  | I2C_FS     | I2C FIFO Status        | <a href="#">Section 7.6.6.10</a> |
| 300Ah  | I2C_RXFS   | I2C RX FIFO Status     | <a href="#">Section 7.6.6.11</a> |
| 300Bh  | I2C_TXFS   | I2C TX FIFO Status     | <a href="#">Section 7.6.6.12</a> |
| 300Ch  | I2C_TXES   | I2C TX Element Status  | <a href="#">Section 7.6.6.13</a> |

Complex bit access types are encoded to fit into small table cells. [Section 7.6.6](#) shows the codes that are used for access types in this section.

**Table 7-143. I2C Access Type Codes**

| Access Type            | Code    | Description                            |
|------------------------|---------|----------------------------------------|
| Read Type              |         |                                        |
| R                      | R       | Read                                   |
| RH                     | R<br>H  | Read<br>Set or cleared by hardware     |
| Write Type             |         |                                        |
| W                      | W       | Write                                  |
| W1C                    | W<br>1C | Write<br>1 to clear                    |
| W1S                    | W<br>1S | Write<br>1 to set                      |
| WP                     | W<br>P  | Write<br>Requires privileged access    |
| Reset or Default Value |         |                                        |
| -n                     |         | Value after reset or the default value |

### 7.6.6.1 I2C\_CREL Register (Offset = 3000h) [Reset = 10h]

I2C\_CREL is shown in [Figure 7-100](#) and described in [Table 7-144](#).

Return to the [Summary Table](#).

I2C IP release version

**Figure 7-100. I2C\_CREL Register**

| 7     | 6 | 5 | 4 | 3     | 2 | 1 | 0 |
|-------|---|---|---|-------|---|---|---|
| MAJOR |   |   |   | MINOR |   |   |   |
| R-1h  |   |   |   | R-0h  |   |   |   |

**Table 7-144. I2C\_CREL Register Field Descriptions**

| Bit | Field | Type | Reset | Description                               |
|-----|-------|------|-------|-------------------------------------------|
| 7-4 | MAJOR | R    | 1h    | The major version number of this revision |
| 3-0 | MINOR | R    | 0h    | The minor version number of this revision |

### 7.6.6.2 I2C\_SCRATCH Register (Offset = 3001h) [Reset = 00h]

I2C\_SCRATCH is shown in [Figure 7-101](#) and described in [Table 7-145](#).

Return to the [Summary Table](#).

Customer Scratchpad register

**Figure 7-101. I2C\_SCRATCH Register**

|            |   |   |   |   |   |   |   |
|------------|---|---|---|---|---|---|---|
| 7          | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SCRATCHPAD |   |   |   |   |   |   |   |
| R/W-0h     |   |   |   |   |   |   |   |

**Table 7-145. I2C\_SCRATCH Register Field Descriptions**

| Bit | Field      | Type | Reset | Description                                                                                                                                                                                                   |
|-----|------------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | SCRATCHPAD | R/W  | 0h    | A customer scratchpad register.<br>This can be used to validate the communication by writing and reading the value or can be used to store some data.<br>This is NOT retained through resets/power on resets. |

### 7.6.6.3 I2C\_CTRL Register (Offset = 3002h) [Reset = 18h]

I2C\_CTRL is shown in [Figure 7-102](#) and described in [Table 7-146](#).

Return to the [Summary Table](#).

This register controls the I2C IP

**Figure 7-102. I2C\_CTRL Register**

| 7         | 6           | 5    | 4           | 3           | 2       | 1      | 0      |
|-----------|-------------|------|-------------|-------------|---------|--------|--------|
| SBR_START | AUTO_SBR_EN | RSVD | NACK_FAIL_D | NACK_FAIL_A | LSM     | I2C_EN | CCE    |
| RH/W1S-0h | R/WP-0h     | R-0h | R/WP-1h     | R/WP-1h     | R/WP-0h | RH-0h  | R/W-0h |

**Table 7-146. I2C\_CTRL Register Field Descriptions**

| Bit | Field       | Type   | Reset | Description                                                                                                                                                                                                                                                                                                |
|-----|-------------|--------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | SBR_START   | RH/W1S | 0h    | Generate I2C Stuck Bus Recovery Sequence Manually. Cleared automatically by IP when reset is complete<br>0h = No reset pattern is generated<br>1h = Begin I2C stuck bus recovery sequence                                                                                                                  |
| 6   | AUTO_SBR_EN | R/WP   | 0h    | Enable automatic stuck bus recovery sequence if the device detects that the bus data line is stuck.<br>0h = Do nothing if a stuck bus is detected<br>1h = Automatically generate the stuck bus recovery sequence if a stuck bus is detected                                                                |
| 5   | RSVD        | R      | 0h    |                                                                                                                                                                                                                                                                                                            |
| 4   | NACK_FAIL_D | R/WP   | 1h    | Consider the message failed if a NACK is received in the data field. A failed message will terminate termination and signal a stop.<br>0h = Note NACK flag and continue transmitting (message still considered successful)<br>1h = Terminate transaction if a NACK is received when unexpected             |
| 3   | NACK_FAIL_A | R/WP   | 1h    | Consider the message failed if a NACK is received after an address (first byte). A failed message will terminate termination and signal a stop.<br>0h = Note NACK flag and continue transmitting (message still considered successful)<br>1h = Terminate transaction if a NACK is received when unexpected |
| 2   | LSM         | R/WP   | 0h    | Changes the input clock to a 5 MHz clock instead of 20 MHz in order to get slower I2C speeds<br>0h = High speed clock, 10 MHz<br>1h = Low speed clock, 2.5 MHz (For less than 100 KHz)                                                                                                                     |
| 1   | I2C_EN      | RH     | 0h    | I2C IP enable status flag.<br>This flag is not writable, but is set if the I2C IP is enabled by allocating memory from MRAM to the IP.<br>This is done with the <a href="#">MRAM_IP_CFG</a> register<br>0h = I2C IP is disabled, all I2C functionality is disabled<br>1h = I2C IP is enabled               |
| 0   | CCE         | R/W    | 0h    | I2C IP change control enable bit.<br>If this bit is set, the I2C IP is held in reset<br>0h = I2C configuration registers are write protected<br>1h = I2C configuration registers can be changed                                                                                                            |

**7.6.6.4 I2C\_BR Register (Offset = 3003h) [Reset = 18h]**

I2C\_BR is shown in [Figure 7-103](#) and described in [Table 7-147](#).

Return to the [Summary Table](#).

**Figure 7-103. I2C\_BR Register**

|           |   |   |   |   |   |   |   |
|-----------|---|---|---|---|---|---|---|
| 7         | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BAUD_RATE |   |   |   |   |   |   |   |
| R/WP-18h  |   |   |   |   |   |   |   |

**Table 7-147. I2C\_BR Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                  |
|-----|-----------|------|-------|------------------------------------------------------------------------------|
| 7-0 | BAUD_RATE | R/WP | 18h   | The 8-bit divider value for I2C.<br>Interpreted as 1 more than entered value |

### 7.6.6.5 I2C\_FIFO\_CTRL Register (Offset = 3004h) [Reset = 00h]

I2C\_FIFO\_CTRL is shown in [Figure 7-104](#) and described in [Table 7-148](#).

Return to the [Summary Table](#).

**Figure 7-104. I2C\_FIFO\_CTRL Register**

| 7       | 6 | 5       | 4 | 3    | 2         | 1         | 0    |
|---------|---|---------|---|------|-----------|-----------|------|
| RX_TRG  |   | TX_TRG  |   | RSVD | CLR_TX    | CLR_RX    | RSVD |
| R/WP-0h |   | R/WP-0h |   | R-0h | RH/W1S-0h | RH/W1S-0h | R-0h |

**Table 7-148. I2C\_FIFO\_CTRL Register Field Descriptions**

| Bit | Field  | Type   | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-----|--------|--------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-6 | RX_TRG | R/WP   | 0h    | <p>Sets the trigger level for the RX FIFO when the RX FIFO reaches the specified number of bytes stored in the FIFO.</p> <p>This allows the CPU to shift bytes out in bulk instead of 1 byte at a time to save the number of interrupts</p> <p>0h = 4 bytes in RX FIFO<br/> 1h = 8 bytes in RX FIFO<br/> 2h = [(max size) - 128] bytes in RX FIFO<br/> 3h = [(max size) - 64] bytes in RX FIFO</p>                                                                                                                                                                                                                      |
| 5-4 | TX_TRG | R/WP   | 0h    | <p>Sets the trigger level for when the number of free bytes in the TX FIFO increases to the specified amount.</p> <p>Used to interrupt when there's a certain number of free spaces in the FIFO so the processor can load multiple bytes into the FIFO instead of just 1 at a time.</p> <hr/> <p style="text-align: center;"><b>Note</b></p> <p style="text-align: center;">This will trigger like an edge-based interrupt.<br/> If the user never falls below 8 free spaces</p> <hr/> <p>0h = 16 bytes free<br/> 1h = 32 bytes free<br/> 2h = [(max size) - 128] bytes free<br/> 3h = [(max size) - 64] bytes free</p> |
| 3   | RSVD   | R      | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 2   | CLR_TX | RH/W1S | 0h    | <p>Clear the transmit FIFO contents.</p> <p>Hardware sets bit back to 0 when clear is complete.</p> <p>0h = Do nothing<br/> 1h = Clear transmit FIFO</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 1   | CLR_RX | RH/W1S | 0h    | <p>Clear the receive FIFO contents.</p> <p>Hardware sets bit back to 0 when clear is complete.</p> <p>0h = Do nothing<br/> 1h = Clear receive FIFO</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 0   | RSVD   | R      | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

### 7.6.6.6 I2C\_IE\_0 Register (Offset = 3005h) [Reset = 00h]

I2C\_IE\_0 is shown in [Figure 7-105](#) and described in [Table 7-149](#).

Return to the [Summary Table](#).

**Figure 7-105. I2C\_IE\_0 Register**

| 7      | 6      | 5        | 4        | 3       | 2      | 1       | 0      |
|--------|--------|----------|----------|---------|--------|---------|--------|
| TXLIE0 | TXAIE0 | DNACKIE0 | ANACKIE0 | SBRCIE0 | RXLIE0 | RXFLIE0 | RXNIE0 |
| R/W-0h | R/W-0h | R/W-0h   | R/W-0h   | R/W-0h  | R/W-0h | R/W-0h  | R/W-0h |

**Table 7-149. I2C\_IE\_0 Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                                                                                              |
|-----|----------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXLIE0   | R/W  | 0h    | TX lost message due to full FIFO interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                                                   |
| 6   | TXAIE0   | R/W  | 0h    | TX space available interrupt enable for INT0.<br>The trigger level is set by the I2C_FIFO_CTRL.TX_TRG setting<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output. |
| 5   | DNACKIE0 | R/W  | 0h    | Data NACK interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                                                                          |
| 4   | ANACKIE0 | R/W  | 0h    | Address NACK interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                                                                       |
| 3   | SBRCIE0  | R/W  | 0h    | Stuck bus recovery completed interrupt<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                                                                       |
| 2   | RXLIE0   | R/W  | 0h    | RX overrun/lost message interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                                                            |
| 1   | RXFLIE0  | R/W  | 0h    | RX fill level interrupt enable for INT0.<br>The trigger level is set by the I2C_FIFO_CTRL.RX_TRG setting<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.     |
| 0   | RXNIE0   | R/W  | 0h    | RX new message interrupt enable for INT0<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                                                                     |

### 7.6.6.7 I2C\_IE\_1 Register (Offset = 3006h) [Reset = 00h]

I2C\_IE\_1 is shown in [Figure 7-106](#) and described in [Table 7-150](#).

Return to the [Summary Table](#).

**Figure 7-106. I2C\_IE\_1 Register**

| 7      | 6      | 5        | 4        | 3       | 2      | 1       | 0      |
|--------|--------|----------|----------|---------|--------|---------|--------|
| TXLIE1 | TXAIE1 | DNACKIE1 | ANACKIE1 | SBRCIE1 | RXLIE1 | RXFLIE1 | RXNIE1 |
| R/W-0h | R/W-0h | R/W-0h   | R/W-0h   | R/W-0h  | R/W-0h | R/W-0h  | R/W-0h |

**Table 7-150. I2C\_IE\_1 Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                                                                                              |
|-----|----------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXLIE1   | R/W  | 0h    | TX lost message due to full FIFO interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                                                   |
| 6   | TXAIE1   | R/W  | 0h    | TX space available interrupt enable for INT1.<br>The trigger level is set by the I2C_FIFO_CTRL.TX_TRG setting<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt enabled. Flag being set WILL assert an interrupt output. |
| 5   | DNACKIE1 | R/W  | 0h    | Data NACK interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                                                                          |
| 4   | ANACKIE1 | R/W  | 0h    | Address NACK interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                                                                       |
| 3   | SBRCIE1  | R/W  | 0h    | Stuck bus recovery completed interrupt<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                                                                       |
| 2   | RXLIE1   | R/W  | 0h    | RX overrun/lost message interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                                                            |
| 1   | RXFLIE1  | R/W  | 0h    | RX fill level interrupt enable for INT1.<br>The trigger level is set by the I2C_FIFO_CTRL.RX_TRG setting<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.     |
| 0   | RXNIE1   | R/W  | 0h    | RX new message interrupt enable for INT1<br>0h = Interrupt disabled. Flag being set will NOT assert an interrupt output.<br>1h = Interrupt disabled. Flag being set WILL assert an interrupt output.                                                                     |

### 7.6.6.8 I2C\_IR Register (Offset = 3007h) [Reset = 00h]

I2C\_IR is shown in [Figure 7-107](#) and described in [Table 7-151](#).

Return to the [Summary Table](#).

**Figure 7-107. I2C\_IR Register**

| 7        | 6        | 5        | 4        | 3        | 2        | 1        | 0        |
|----------|----------|----------|----------|----------|----------|----------|----------|
| TXL      | TXA      | DNACK    | ANACK    | SBRC     | RXL      | RXFL     | RXN      |
| R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h | R/W1C-0h |

**Table 7-151. I2C\_IR Register Field Descriptions**

| Bit | Field | Type  | Reset | Description                                                                                                                                                                                                                                                                                                   |
|-----|-------|-------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXL   | R/W1C | 0h    | TX lost message due to full FIFO interrupt<br>0h = No TX FIFO byte was lost<br>1h = TX FIFO lost at least byte (overrun) due to FIFO being full when a write was attempted                                                                                                                                    |
| 6   | TXA   | R/W1C | 0h    | TX space available interrupt.<br>The trigger level is set by the I2C_FIFO_CTRL.TX_TRG setting<br><br><b>Note</b><br>When the I2C IP is enabled, this interrupt will set since the FIFO will be empty<br><br>0h = No TX FIFO space available interrupt<br>1h = The TX FIFO has hit the trigger level threshold |
| 5   | DNACK | R/W1C | 0h    | Data NACK interrupt<br>0h = No NACK was detected<br>1h = A NACK was detected during the data phase                                                                                                                                                                                                            |
| 4   | ANACK | R/W1C | 0h    | Address NACK interrupt<br>0h = No NACK was detected<br>1h = A NACK was detected during the address phase                                                                                                                                                                                                      |
| 3   | SBRC  | R/W1C | 0h    | Stuck bus recovery completed interrupt<br>0h = No stuck bus recovery has occurred<br>1h = Stuck bus recovery has completed. Check the I2C_STATUS register to see if the bus is still stuck.                                                                                                                   |
| 2   | RXL   | R/W1C | 0h    | RX overrun/lost message interrupt<br>0h = No RX FIFO bytes lost (no overrun) interrupt<br>1h = At least 1 byte has been lost due to full RX FIFO (overrun interrupt)                                                                                                                                          |
| 1   | RXFL  | R/W1C | 0h    | RX fill level interrupt.<br>The trigger level is set by the I2C_FIFO_CTRL.RX_TRG setting<br>0h = No RX FIFO fill level interrupt<br>1h = The RX FIFO reached the fill level                                                                                                                                   |
| 0   | RXN   | R/W1C | 0h    | RX new message interrupt<br>0h = No new byte received interrupt<br>1h = A new byte has been received                                                                                                                                                                                                          |

### 7.6.6.9 I2C\_STATUS Register (Offset = 3008h) [Reset = XXh]

I2C\_STATUS is shown in [Figure 7-108](#) and described in [Table 7-152](#).

Return to the [Summary Table](#).

**Figure 7-108. I2C\_STATUS Register**

| 7     | 6     | 5     | 4     | 3    | 2 | 1     | 0 |
|-------|-------|-------|-------|------|---|-------|---|
| SDA_S | SCL_S | SDA_V | SCL_V | RSVD |   | IDLE  |   |
| RH-0h | RH-0h | RH-Xh | RH-Xh | R-0h |   | RH-Xh |   |

**Table 7-152. I2C\_STATUS Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                        |
|-----|-------|------|-------|----------------------------------------------------------------------------------------------------|
| 7   | SDA_S | RH   | 0h    | SDA Stuck<br>0h = SDA is not stuck<br>1h = SDA is currently stuck low                              |
| 6   | SCL_S | RH   | 0h    | SCL Stuck<br>0h = SCL is not stuck<br>1h = SCL is currently stuck low                              |
| 5   | SDA_V | RH   | X     | SDA Value<br>0h = SDA is currently low<br>1h = SDA is currently high                               |
| 4   | SCL_V | RH   | X     | SCL Value<br>0h = SCL is currently low<br>1h = SCL is currently high                               |
| 3-1 | RSVD  | R    | 0h    |                                                                                                    |
| 0   | IDLE  | RH   | X     | Whether the I2C IP is idle<br>0h = I2C IP is busy, a transaction is ongoing<br>1h = I2C IP is idle |

**7.6.6.10 I2C\_FS Register (Offset = 3009h) [Reset = 00h]**

I2C\_FS is shown in [Figure 7-109](#) and described in [Table 7-153](#).

Return to the [Summary Table](#).

**Figure 7-109. I2C\_FS Register**

| 7      | 6     | 5     | 4 | 3 | 2 | 1 | 0 |
|--------|-------|-------|---|---|---|---|---|
| TXFSRE | RXFD  | RXFB  |   |   |   |   |   |
| RH-0h  | RH-0h | RH-0h |   |   |   |   |   |

**Table 7-153. I2C\_FS Register Field Descriptions**

| Bit | Field  | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-----|--------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXFSRE | RH   | 0h    | <p>Whether the TX FIFO and shift register is empty or not. This means that all queued data has been transmitted and the controller is idle. Nothing is in the FIFO, and nothing is in the shift register. If this is set, it is safe to configure or disable the I2C controller</p> <hr/> <p><b>Note</b></p> <p>This does NOT take into consideration a TX FIFO element which is not been fully written to. If there is a partially written TX FIFO element, that untransmitted element does not clear the idle flag until the element has been fully written</p> <hr/> <p>0h = There is at least 1 ongoing and/or pending transmissions.<br/>1h = The TX FIFO and shift register are empty and IP is idle. It is safe to configure or turn off the controller</p> |
| 6   | RXFD   | RH   | 0h    | <p>Whether the RX FIFO contains 1 or more unread bytes<br/>0h = The RX FIFO is empty<br/>1h = At least 1 element in the RX FIFO</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 5-0 | RXFB   | RH   | 0h    | <p>The number of data bytes remaining to be read from the I2C frame stored at the next RX FIFO element. Valid values are 0 to 63. Any value greater than 63 will be reflected as 63, and the remaining will be updated as the processor reads the data out.</p> <hr/> <p><b>Note</b></p> <p>This refers to only the number of data bytes in the I2C frame. Note that each read to the RX FIFO will need to add 3 bytes of the I2C frame header at each read to the RX FIFO</p> <hr/>                                                                                                                                                                                                                                                                               |

#### 7.6.6.11 I2C\_RXFS Register (Offset = 300Ah) [Reset = 00h]

I2C\_RXFS is shown in [Figure 7-110](#) and described in [Table 7-154](#).

Return to the [Summary Table](#).

Contains the counter of how many unread I2C messages are stored in the RX FIFO.

**Figure 7-110. I2C\_RXFS Register**

|       |   |   |   |   |   |   |   |
|-------|---|---|---|---|---|---|---|
| 7     | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| RX_FL |   |   |   |   |   |   |   |
| R-0h  |   |   |   |   |   |   |   |

**Table 7-154. I2C\_RXFS Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                                                                                                                                                                   |
|-----|-------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | RX_FL | R    | 0h    | RX FIFO fill level states how many messages are currently in the RX FIFO.<br>Valid values are 0 to 255.<br>Maximum value listed depends on how device memory is configured.<br>If 255 is reported, then 255 or more messages are in the FIFO. |

#### 7.6.6.12 I2C\_TXFS Register (Offset = 300Bh) [Reset = 00h]

I2C\_TXFS is shown in [Figure 7-111](#) and described in [Table 7-155](#).

Return to the [Summary Table](#).

Contains the counter of how many available bytes are in the TX FIFO. Note that you need the 2 bytes of I2C packet header data are to be included

**Figure 7-111. I2C\_TXFS Register**

| 7     | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|---|---|---|---|---|---|---|
| TX_SA |   |   |   |   |   |   |   |
| R-0h  |   |   |   |   |   |   |   |

**Table 7-155. I2C\_TXFS Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                                                                                                                                                                                                               |
|-----|-------|------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | TX_SA | R    | 0h    | <p>TX FIFO fill level states how many free bytes are currently in the TX FIFO.<br/>Numbers greater than 255 will show as 255.</p> <hr/> <p><b>Note</b></p> <p>The number of bytes needed for an I2C frame must include the 2 bytes of I2C header, in addition to the data bytes</p> <hr/> |

### 7.6.6.13 I2C\_TXES Register (Offset = 300Ch) [Reset = 00h]

I2C\_TXES is shown in [Figure 7-112](#) and described in [Table 7-156](#).

Return to the [Summary Table](#).

Contains information about the current/next transmit FIFO element

**Figure 7-112. I2C\_TXES Register**

| 7     | 6     | 5 | 4 | 3 | 2 | 1 | 0 |
|-------|-------|---|---|---|---|---|---|
| TXEIP | TXEBP |   |   |   |   |   |   |
| R-0h  | R-0h  |   |   |   |   |   |   |

**Table 7-156. I2C\_TXES Register Field Descriptions**

| Bit | Field | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-----|-------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | TXEIP | R    | 0h    | TX FIFO element write in progress<br>Used to signal if a TX FIFO element is partially written. If there are not enough data bytes that were written into the FIFO (according to the header), then this flag is set. If it is cleared, then the next write to the TX FIFO will be a new element and should start with the I2C header fields<br>0h = TX element is not in progress (next write to TX FIFO is a new element)<br>1h = TX element is in progress, and has an expected number of bytes left                                                                                                                                 |
| 6-0 | TXEBP | R    | 0h    | TX Element Bytes Pending<br>If a TX element write has started, this is how many bytes are still expected in order to complete the FIFO element.<br><br><div style="text-align: center;"><b>Note</b></div> <div>This includes header bytes and can change values if the number of bytes in the transfer has not been written yet.<br/>For example, if only the I2C address byte was written, this field will read 1, since we do not know how many bytes are pending until told how many bytes of data are to be transferred.<br/>Once the number of bytes in the transfer is known, this will recalculate as data is written in</div> |

### 7.6.7 I2C Data FIFOs

FIFOs are used to transfer data for I2C. There are 2 FIFOs used for the interface

- I2C Transmit FIFO (h3010): Used to queue bytes of data to transmit onto the I2C interface
- I2C Receive FIFO (h3010): Used to read bytes received from an I2C read

#### 7.6.7.1 I2C Transmit FIFO (address = h3010)

The transmit FIFO is write only and is written to when a write command is issued to its address. Transmission will begin when the entire frame has been written into the buffer. All data bytes written after the end of the frame (specified by the initial header) are ignored until a new transmit FIFO write is started. All writes to a transmit FIFO should be directed at the specified I2C Transmit FIFO address only, as each byte that is written is automatically shifted into the FIFO. For example, writing larger I2C frames than can be transferred in a single CAN message occurs over multiple CAN writes to the same address, since the SPI header bytes will tell the device how many data bytes to expect.

The size of each frame in the TX buffer is determined by the desired I2C frame length. There is a 2-byte header required at the beginning of each I2C FIFO element. The total length of the frame in the TX FIFO is the sum of

- 2 bytes for the I2C header
- Message length of data bytes (if the frame is an I2C write. An I2C read only requires the number of bytes to read in the header, and any additional bytes are ignored)

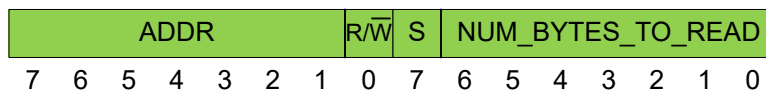
See [Section 7.5.6.3](#) for more information and examples.

**Note**

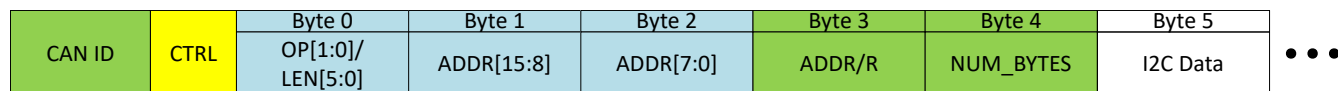
If the transmit FIFO is full, any new messages written to it are discarded and an interrupt is set.



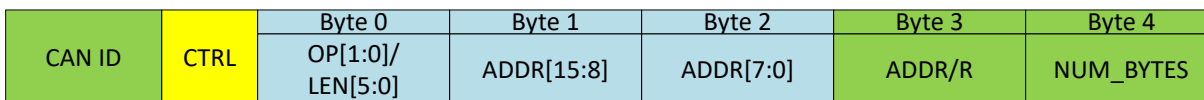
**Figure 7-113. I2C Write TX FIFO Header (Header and Data Only)**



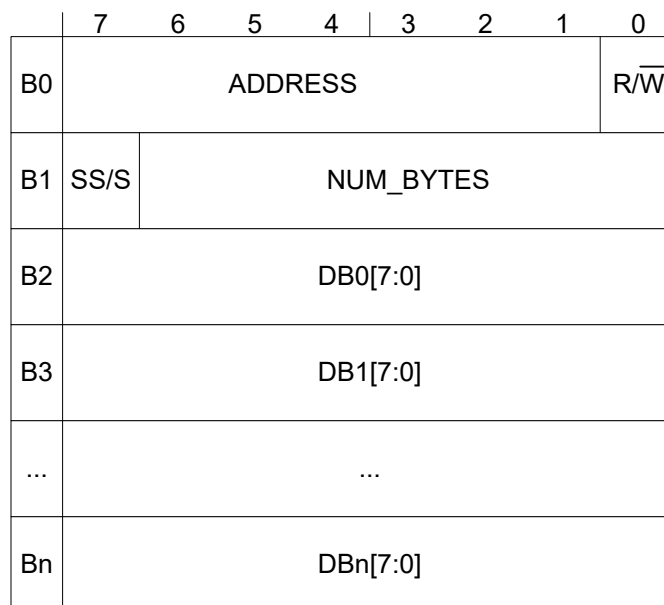
**Figure 7-114. I2C Read TX FIFO Header (Header Only)**



**Figure 7-115. Example I2C Write TX FIFO Header (Standard CAN Frame Format)**



**Figure 7-116. Example I2C Read TX FIFO Header (Standard CAN Frame Format)**



**Figure 7-117. I2C Transmit FIFO**

**Table 7-157. I2C Transmit FIFO Descriptions**

| Byte | Bit | Field                          | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|------|-----|--------------------------------|------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0    | 7:1 | ADDRESS                        | W    | 0x0   | I2C target device address                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|      | 0   | R/ $\overline{W}$              | W    | 0     | Whether I2C message is an I2C read or an I2C write<br>0 = I2C Write<br>1 = I2C Read                                                                                                                                                                                                                                                                                                                                                                                        |
| 1    | 7   | S (I2C Read)<br>SS (I2C Write) | W    | 0     | Store (I2C Read) or Store Status (I2C Write)<br>For an I2C read, stores the received data into the receive FIFO. For an I2C write, since no data is received from the target device, the user can choose whether to store a status frame for the I2C write or assume it was successful (do not store any status in the receive FIFO)<br>0 = Do not store any data/status of this transaction to the receive FIFO<br>1 = Store the data or write status to the receive FIFO |
|      | 6:0 | NUM_BYTES_TO_WRITE/READ        | W    | 0x0   | I2C Write (B0.R/ $\overline{W}$ = 0): Number of bytes in the I2C frame after the address (does NOT include the above header bytes)<br>Valid values 0-127. 0 will not send any data, but only transmit the device address (useful for checking for an ACK)<br>I2C Read (B0.R/ $\overline{W}$ = 1): Number of bytes to read after the address byte (does NOT include the above header bytes)                                                                                 |

### 7.6.7.2 I2C Receive FIFO (address = h3010)

The receive FIFO is read only.

The size of each frame in the RX buffer is determined by the I2C frame's data length. The total length of the RX frame in the buffer is the sum of

- 2 bytes for I2C header
- 1 byte for I2C status
- Message length of data bytes

A read from the RX FIFO is considered completed and freed when all data bytes have been read from the FIFO element. Any bytes beyond the number of bytes remaining for a read returns 0x00 for the invalid bytes.

A partial read of a frame from the RX FIFO will automatically be continued for the next read to the I2C Recieve FIFO, and is signaled as such with the CONT bit being set to 1, as the NUM\_BYTES\_REMAINING reflecting the number of data bytes left in this I2C frame to read.

A read from an empty RX buffer returns 0 NUM\_BYTES\_REMAINING for channel 0. Frames will always be read in the order that they were received.

The RX buffer can be cleared (all frames discarded) by a write of 1 to the I2C Clear RX FIFO bit (I2C\_FIFO\_CTRL[1]).

#### Note

If the receive FIFO is full, all new incoming messages is discarded and an interrupt set to alert the processor that a message was lost.

See [Section 7.5.6.3](#) for more information and examples.



Figure 7-118. I2C RX FIFO Read Header (Header and Data Only)

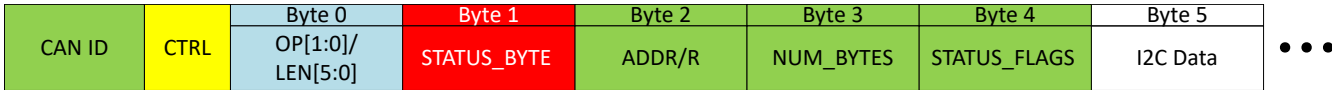
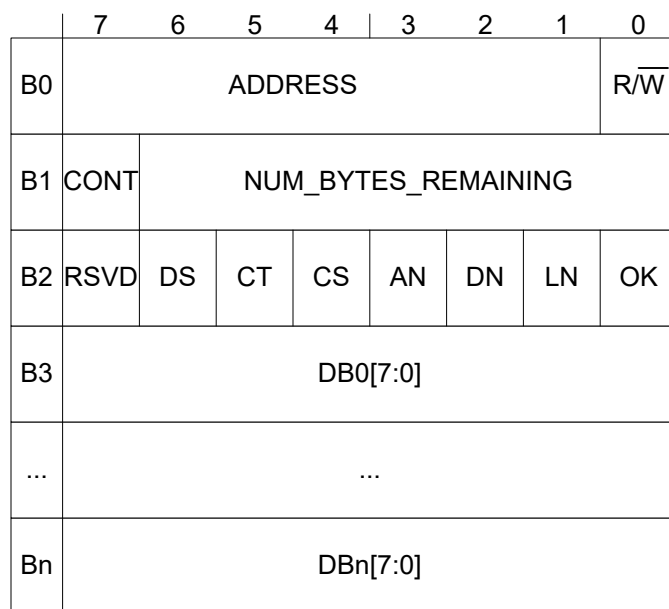


Figure 7-119. Example I2C RX FIFO Header (Standard CAN Frame Format)



**Figure 7-120. I2C Receive FIFO Frame Format**

**Table 7-158. I2C Receive FIFO Byte Descriptions**

| Byte | Bit | Field               | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                       |
|------|-----|---------------------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0    | 7:1 | ADDRESS             | R    | 0x0   | I2C target device address                                                                                                                                                                                                                                                                                                                                                                         |
|      | 0   | W/ $\overline{R}$   | R    | 0     | Whether I2C message is an I2C read or an I2C write<br>0 = I2C Read<br>1 = I2C Write                                                                                                                                                                                                                                                                                                               |
| 1    | 7   | CONT                | R    | 0     | Continued Read<br>0 = This read is from the beginning of a frame<br>1 = This read is continued from a previous read                                                                                                                                                                                                                                                                               |
|      | 6:0 | NUM_BYTES_REMAINING | R    | 0x0   | I2C Read (B0.W/ $\overline{R}$ = 0): Number of bytes to read after the address byte (does NOT include any of the above header bytes)<br>I2C Write (B0.W/ $\overline{R}$ = 1): Number of bytes in the I2C frame after the address (does NOT include the above header bytes)<br>Valid values 0-127. 0 will not send any data, but only transmit the device address (useful for checking for an ACK) |
| 2    | 7   | RSVD                | R    | 0     | Reserved                                                                                                                                                                                                                                                                                                                                                                                          |
|      | 6   | DS                  | R    | 0     | Data Stuck<br>Bus is stuck (clock or data is not being released)                                                                                                                                                                                                                                                                                                                                  |
|      | 5   | CT                  | R    | 0     | Clock Timeout<br>SMBus clock timeout was detected                                                                                                                                                                                                                                                                                                                                                 |
|      | 4   | CS                  | R    | 0     | Clock Stretched<br>Clock stretching occurred during this frame                                                                                                                                                                                                                                                                                                                                    |
|      | 3   | AN                  | R    | 0     | Address NACK<br>The I2C target device NACKed on the address byte                                                                                                                                                                                                                                                                                                                                  |
|      | 2   | DN                  | R    | 0     | Data NACK<br>The I2C target device NACKed on at least 1 byte during the write.                                                                                                                                                                                                                                                                                                                    |
|      | 1   | LN                  | R    | 0     | Last Byte NACK<br>The I2C target device NACKed the last byte of the I2C frame                                                                                                                                                                                                                                                                                                                     |
|      | 0   | OK                  | R    | 0     | Success<br>I2C frame completed successfully                                                                                                                                                                                                                                                                                                                                                       |

### 7.6.8 PWM0 Registers

Section 7.6.8 lists the memory-mapped registers for the PWM0 registers. All register offset addresses not listed in Section 7.6.8 should be considered as reserved locations and the register contents should not be modified.

**Table 7-159. PWM0 Registers**

| Offset             | Acronym                   | Register Name                             | Section                          |
|--------------------|---------------------------|-------------------------------------------|----------------------------------|
| 4000h              | PWM1_ACTION               |                                           | <a href="#">Section 7.6.8.1</a>  |
| 4000h              | PWM0_CTRL                 | PWM0 Control                              | <a href="#">Section 7.6.8.2</a>  |
| 4001h              | PWM0_IE0                  | Interrupt Enable                          | <a href="#">Section 7.6.8.3</a>  |
| 4002h              | PWM0_IE1                  | Interrupt Enable                          | <a href="#">Section 7.6.8.4</a>  |
| 4003h              | PWM0_IR                   | Interrupt Register                        | <a href="#">Section 7.6.8.5</a>  |
| 4004h              | PWM0_STATUS               | Status                                    | <a href="#">Section 7.6.8.6</a>  |
| 4005h +<br>formula | PWM0_CUR_PULSE[y]         | Current Pulse Count                       | <a href="#">Section 7.6.8.7</a>  |
| 4009h              | PWM0_CUR_VAL_MSB          | Current Value MSB                         | <a href="#">Section 7.6.8.8</a>  |
| 400Ah              | PWM0_CUR_VAL_LSB          | Current Value LSB                         | <a href="#">Section 7.6.8.9</a>  |
| 400Bh              | PWM0_CONST_MSB            | Constant Value Integer MSB                | <a href="#">Section 7.6.8.10</a> |
| 400Ch              | PWM0_CONST_LSB            | Constant Value Integer LSB                | <a href="#">Section 7.6.8.11</a> |
| 400Dh              | PWM0_STOP_VAL_FRAC_F      | Stop Fractional Value (Frequency Only)    | <a href="#">Section 7.6.8.12</a> |
| 400Eh              | PWM0_STOP_VAL_MSB         | Stop Value Integer MSB                    | <a href="#">Section 7.6.8.13</a> |
| 400Fh              | PWM0_STOP_VAL_LSB         | Stop Value Integer LSB                    | <a href="#">Section 7.6.8.14</a> |
| 4010h              | PWM0_STOP_SL_MSB          | Stop Slope MSB                            | <a href="#">Section 7.6.8.15</a> |
| 4011h              | PWM0_STOP_SL_MID          | Stop Slope Mid                            | <a href="#">Section 7.6.8.16</a> |
| 4012h              | PWM0_STOP_SL_LSB          | Stop Slope LSB                            | <a href="#">Section 7.6.8.17</a> |
| 4013h              | PWM0_START_VAL_FRAC_F     | Start Fractional Value (Frequency Only)   | <a href="#">Section 7.6.8.18</a> |
| 4014h              | PWM0_START_VAL_MSB        | Start Value MSB                           | <a href="#">Section 7.6.8.19</a> |
| 4015h              | PWM0_START_VAL_LSB        | Start Value LSB                           | <a href="#">Section 7.6.8.20</a> |
| 4016h              | PWM0_START_SL_MSB         | Start Slope MSB                           | <a href="#">Section 7.6.8.21</a> |
| 4017h              | PWM0_START_SL_MID         | Start Slope Mid Byte                      | <a href="#">Section 7.6.8.22</a> |
| 4018h              | PWM0_START_SL_LSB         | Start Slope LSB                           | <a href="#">Section 7.6.8.23</a> |
| 4019h              | PWM0_END_VAL_CONST_FRAC_F | End (Freq) or Const (DC) Fractional Value | <a href="#">Section 7.6.8.24</a> |
| 401Ah              | PWM0_END_VAL_MSB          | End Value MSB                             | <a href="#">Section 7.6.8.25</a> |
| 401Bh              | PWM0_END_VAL_LSB          | End Value LSB                             | <a href="#">Section 7.6.8.26</a> |
| 401Ch +<br>formula | PWM0_PULSE_STOP_RAMP[y]   | Pulse Count to Begin Stop Ramp            | <a href="#">Section 7.6.8.27</a> |
| 4020h +<br>formula | PWM0_PULSE_MAX[y]         | Pulse Count Maximum                       | <a href="#">Section 7.6.8.28</a> |
| 4024h              | PWM0_ACTION               | Start Action Register                     | <a href="#">Section 7.6.8.29</a> |
| 4030h              | PWM0_IAS_CTRL             | Input Auto Stop Control                   | <a href="#">Section 7.6.8.30</a> |

Complex bit access types are encoded to fit into small table cells. Section 7.6.8 shows the codes that are used for access types in this section.

**Table 7-160. PWM0 Access Type Codes**

| Access Type | Code   | Description                        |
|-------------|--------|------------------------------------|
| Read Type   |        |                                    |
| R           | R      | Read                               |
| RH          | R<br>H | Read<br>Set or cleared by hardware |
| Write Type  |        |                                    |

**Table 7-160. PWM0 Access Type Codes (continued)**

| Access Type              | Code    | Description                                                                                                                                                                                                                                                                          |
|--------------------------|---------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| W                        | W       | Write                                                                                                                                                                                                                                                                                |
| W1C                      | W<br>1C | Write<br>1 to clear                                                                                                                                                                                                                                                                  |
| W1S                      | W<br>1S | Write<br>1 to set                                                                                                                                                                                                                                                                    |
| WP                       | W<br>P  | Write<br>Requires privileged access                                                                                                                                                                                                                                                  |
| Reset or Default Value   |         |                                                                                                                                                                                                                                                                                      |
| -n                       |         | Value after reset or the default value                                                                                                                                                                                                                                               |
| Register Array Variables |         |                                                                                                                                                                                                                                                                                      |
| i,j,k,l,m,n              |         | When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula. |
| y                        |         | When this variable is used in a register name, an offset, or an address it refers to the value of a register array.                                                                                                                                                                  |

### 7.6.8.1 PWM1\_ACTION Register (Offset = 4000h) [Reset = 00h]

PWM1\_ACTION is shown in [Table 7-161](#).

Return to the [Summary Table](#).

**Table 7-161. PWM1\_ACTION Register Field Descriptions**

| Bit | Field     | Type  | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-----|-----------|-------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 4   | COUNT_RST | R/W1S | 0h    | <p>Reset the pulse counter to 0<br/>If this bit is set, the pulse counter is reset.<br/>This is typically performed at the start of a ramp, in order to keep track of the total number of pulses.<br/>It gets cleared by hardware once the reset is complete.</p> <hr/> <p><b>Note</b></p> <p>Pay attention to the stop and maximum pulse count values if using AUTO_STOP.<br/>Those pulse counters are compared against the current pulse counter, and if the maximum pulse count is less than the current pulse counter value, no PWM output will be generated</p> <hr/> |
| 3-0 | RESERVED  | R     | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |

### 7.6.8.2 PWM0\_CTRL Register (Offset = 4000h) [Reset = 11h]

PWM0\_CTRL is shown in [Figure 7-121](#) and described in [Table 7-162](#).

Return to the [Summary Table](#).

Control fields to configure the PWM0 module

**Figure 7-121. PWM0\_CTRL Register**

| 7    | 6           | 5 | 4 | 3       | 2 | 1       | 0      |
|------|-------------|---|---|---------|---|---------|--------|
| RSVD | SLOPE_SCALE |   |   | MODE    |   | DC_8B   | INIT   |
| R-0h | R/WP-1h     |   |   | R/WP-0h |   | R/WP-0h | R/W-1h |

**Table 7-162. PWM0\_CTRL Register Field Descriptions**

| Bit | Field       | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-----|-------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | RSVD        | R    | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 6-4 | SLOPE_SCALE | R/WP | 1h    | <p>Slope Scale</p> <p>Adjusts the number of bits that are used for integer vs fractional portions of the slope register.</p> <p>The meaning of the enumerations change depending on if frequency or duty cycle is being ramped.</p> <p>See <a href="#">Section 7.5.7.4</a> for more information</p> <p>0h = Int [20-16], Frac [15-0] (Frequency) - Int [20], Frac [19-0] (Duty Cycle)</p> <p>1h = Int [20-12], Frac [11-0] (Frequency) - Int [20-19], Frac [18-0] (Duty Cycle)</p> <p>2h = Int [20-8], Frac [7-0] (Frequency) - Int [20-18], Frac [17-0] (Duty Cycle)</p> <p>3h = Int [20-4], Frac [3-0] (Frequency) - Int [20-17], Frac [16-0] (Duty Cycle)</p> <p>4h = Int [20-0], No Frac (Frequency) - Int [20-16], Frac [15-0] (Duty Cycle)</p> |
| 3-2 | MODE        | R/WP | 0h    | <p>Adjusts the number of bits that are used for integer vs fractional portions of the slope register.</p> <p>Note that setting the mode to anything but off will take priority over SPI chip select if the GPIO is configured for special functionality.</p> <p>0h = Off</p> <p>1h = Duty Cycle Ramping</p> <p>2h = Frequency Ramping</p> <p>3h = Static On</p>                                                                                                                                                                                                                                                                                                                                                                                      |
| 1   | DC_8B       | R/WP | 0h    | <p>8 Bit Duty Cycle Mode</p> <p>0h = 10-bit duty cycle resolution</p> <p>1h = 8-Bit duty cycle resolution</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 0   | INIT        | R/W  | 1h    | <p>Initialization Mode</p> <p>Holds the PWM IP in reset, used for configuring settings in this PWM0_CTRL register</p> <p>0h = Initialization mode disabled. Writes to protected PWM bits are not allowed</p> <p>1h = Initialization mode enabled. Writes to protected PWM bits are allowed</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

### 7.6.8.3 PWM0\_IE0 Register (Offset = 4001h) [Reset = 00h]

PWM0\_IE0 is shown in [Figure 7-122](#) and described in [Table 7-163](#).

Return to the [Summary Table](#).

**Figure 7-122. PWM0\_IE0 Register**

| 7    | 6 | 5 | 4 | 3 | 2                | 1      | 0      |
|------|---|---|---|---|------------------|--------|--------|
| RSVD |   |   |   |   | PULSE_OVF_I<br>E | IAS_IE | RC_IE  |
| R-0h |   |   |   |   | R/W-0h           | R/W-0h | R/W-0h |

**Table 7-163. PWM0\_IE0 Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-----|--------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD         | R    | 0h    | Reserved                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 2   | PULSE_OVF_IE | R/W  | 0h    | Current Pulse Overflow<br>Set when the CUR_PULSE counter overflows<br>0h = Current pulse counter has not overflowed<br>1h = Current pulse counter has overflowed                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 1   | IAS_IE       | R/W  | 0h    | Input Auto Stop Interrupt<br>Sets when the input auto stop condition has occurred, if the PWM is configured to immediately stop, then the output will also be stopped (reflected by the RC bit).<br>If PWM is configured to begin the stop ramp on an IAS trigger, then then this interrupt is set immediately when the input condition happens and the RC bit will be set when the output turns off.<br>0h = Input for IAS has not triggered<br>1h = Input for IAS has triggered                                                                                                                                                                                                                         |
| 0   | RC_IE        | R/W  | 0h    | PWM Ramp Complete<br>Sets when the ramp has finished. If AS_EN = 1 (auto stop), then this happens when the PWM channel turns off.<br>If AS_EN = 0, then this gets set when the PWM has reached the end value.<br>If IAS_EN = 1 (input auto stop), then this will get set once the PWM output is stopped.<br>If a stop ramp is manually activated, the RC bit will be set once the ramp completes and the output is turned off.<br><br><b>Note</b><br><br>If an on-ramp is in progress and a stop condition occurs, the RC bit will NOT be set for the on-ramp, but only once the output stops due to the stop condition<br><br>0h = PWM channel ramp is not complete<br>1h = PWM channel ramp is complete |

#### 7.6.8.4 PWM0\_IE1 Register (Offset = 4002h) [Reset = 00h]

PWM0\_IE1 is shown in [Figure 7-123](#) and described in [Table 7-164](#).

Return to the [Summary Table](#).

**Figure 7-123. PWM0\_IE1 Register**

| 7    | 6 | 5 | 4 | 3 | 2                | 1      | 0      |
|------|---|---|---|---|------------------|--------|--------|
| RSVD |   |   |   |   | PULSE_OVF_I<br>E | IAS_IE | RC_IE  |
| R-0h |   |   |   |   | R/W-0h           | R/W-0h | R/W-0h |

**Table 7-164. PWM0\_IE1 Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-----|--------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD         | R    | 0h    | Reserved                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 2   | PULSE_OVF_IE | R/W  | 0h    | Current Pulse Overflow<br>Set when the CUR_PULSE counter overflows<br>0h = Current pulse counter has not overflowed<br>1h = Current pulse counter has overflowed                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 1   | IAS_IE       | R/W  | 0h    | Input Auto Stop Interrupt<br>Sets when the input auto stop condition has occurred, if the PWM is configured to immediately stop, then the output will also be stopped (reflected by the RC bit).<br>If PWM is configured to begin the stop ramp on an IAS trigger, then this interrupt is set immediately when the input condition happens and the RC bit will be set when the output turns off.<br>If a stop ramp is manually activated, the RC bit will be set once the ramp completes and the output is turned off.<br>0h = Input for IAS has not triggered<br>1h = Input for IAS has triggered                                                                                                                                                                          |
| 0   | RC_IE        | R/W  | 0h    | PWM Ramp Complete<br>Sets when the ramp has finished. If AS_EN = 1 (auto stop), then this happens when the PWM channel turns off.<br>If AS_EN = 0, then this gets set when the PWM has reached the end value.<br>If IAS_EN = 1 (input auto stop), then this will get set once the PWM output is stopped<br>If a stop ramp is manually activated, the RC bit will be set once the ramp completes and the output is turned off.<br><br><div style="text-align: center;"><b>Note</b></div> <div style="text-align: center;">If an on-ramp is in progress and a stop condition occurs, the RC bit will NOT be set for the on-ramp, but only once the output stops due to the stop condition</div><br>0h = PWM channel ramp is not complete<br>1h = PWM channel ramp is complete |

### 7.6.8.5 PWM0\_IR Register (Offset = 4003h) [Reset = 00h]

PWM0\_IR is shown in [Figure 7-124](#) and described in [Table 7-165](#).

Return to the [Summary Table](#).

**Figure 7-124. PWM0\_IR Register**

| 7    | 6 | 5 | 4 | 3         | 2 | 1        | 0        |
|------|---|---|---|-----------|---|----------|----------|
| RSVD |   |   |   | PULSE_OVF |   | IAS      | RC       |
| R-0h |   |   |   | R/W1C-0h  |   | R/W1C-0h | R/W1C-0h |

**Table 7-165. PWM0\_IR Register Field Descriptions**

| Bit | Field     | Type  | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-----|-----------|-------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD      | R     | 0h    | Reserved                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 2   | PULSE_OVF | R/W1C | 0h    | Current Pulse Overflow<br>Set when the CUR_PULSE counter overflows<br>0h = Current pulse counter has not overflowed<br>1h = Current pulse counter has overflowed                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 1   | IAS       | R/W1C | 0h    | Input Auto Stop Interrupt<br>Sets when the input auto stop condition has occurred, if the PWM is configured to immediately stop, then the output will also be stopped (reflected by the RC bit).<br>If PWM is configured to begin the stop ramp on an IAS trigger, then then this interrupt is set immediately when the input condition happens and the RC bit will be set when the output turns off.<br>0h = Input for IAS has not triggered<br>1h = Input for IAS has triggered                                                                                                                                                                                                                                                                             |
| 0   | RC        | R/W1C | 0h    | PWM Ramp Complete<br>Sets when the ramp has finished. If AS_EN = 1 (auto stop), then this happens when the PWM channel turns off.<br>If AS_EN = 0, then this gets set when the PWM has reached the end value.<br>If IAS_EN = 1 (input auto stop), then this will get set once the PWM output is stopped<br>If a stop ramp is manually activated, the RC bit will be set once the ramp completes and the output is turned off.<br><br><div style="text-align: center;"> <b>Note</b><br/>           If an on-ramp is in progress and a stop condition occurs, the RC bit will NOT be set for the on-ramp, but only once the output stops due to the stop condition         </div><br>0h = PWM channel ramp is not complete<br>1h = PWM channel ramp is complete |

### 7.6.8.6 PWM0\_STATUS Register (Offset = 4004h) [Reset = 00h]

PWM0\_STATUS is shown in [Figure 7-125](#) and described in [Table 7-166](#).

Return to the [Summary Table](#).

**Figure 7-125. PWM0\_STATUS Register**

| 7    | 6 | 5 | 4 | 3       | 2 | 1     | 0     |
|------|---|---|---|---------|---|-------|-------|
| RSVD |   |   |   | IAS_ACT |   | BUSY  | OEN   |
| R-0h |   |   |   | RH-0h   |   | RH-0h | RH-0h |

**Table 7-166. PWM0\_STATUS Register Field Descriptions**

| Bit | Field   | Type | Reset | Description                                                                                                                                                                                                                                                                |
|-----|---------|------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD    | R    | 0h    |                                                                                                                                                                                                                                                                            |
| 2   | IAS_ACT | RH   | 0h    | Input Auto Stop Active<br>Gets set when the GPIO input toggles to the asserted state and will self-clear when the PWM output stops.<br>An RC interrupt bit should be set once the ramp is complete<br>0h = Input auto stop is not active<br>1h = Input auto stop is active |
| 1   | BUSY    | RH   | 0h    | Ramp IP Busy<br>0h = PWM ramp is complete<br>1h = PWM ramp is currently on-going                                                                                                                                                                                           |
| 0   | OEN     | RH   | 0h    | Output Enabled<br>0h = PWM output is disabled<br>1h = PWM output is enabled and generating output                                                                                                                                                                          |

**7.6.8.7 PWM0\_CUR\_PULSE[y] Register (Offset = 4005h + formula) [Reset = 00000000h]**

PWM0\_CUR\_PULSE[y] is shown in [Figure 7-126](#) and described in [Table 7-167](#).

Return to the [Summary Table](#).

An unsigned 32 bit counter value of the current pulse count. The lowest address is corresponds to the MSB of the counter.

Offset = 4005h + (y \* 4h); where y = 0h to 3h

**Figure 7-126. PWM0\_CUR\_PULSE[y] Register**

|                |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
|----------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PWM0_CUR_PULSE |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| R-0h           |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

**Table 7-167. PWM0\_CUR\_PULSE[y] Register Field Descriptions**

| Bit  | Field          | Type | Reset | Description                                                                                                                                                                                                                                    |
|------|----------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 31-0 | PWM0_CUR_PULSE | R    | 0h    | The current pulse count of the current ramp.<br>If the ramp has finished, then this value is the total number of pulses generated for the last ramp.<br>This value is reset each time a new ramp is started.<br>The lowest address is the MSB. |

#### 7.6.8.8 PWM0\_CUR\_VAL\_MSB Register (Offset = 4009h) [Reset = 00h]

PWM0\_CUR\_VAL\_MSB is shown in [Figure 7-127](#) and described in [Table 7-168](#).

Return to the [Summary Table](#).

**Figure 7-127. PWM0\_CUR\_VAL\_MSB Register**

|      |   |   |   |   |               |   |   |
|------|---|---|---|---|---------------|---|---|
| 7    | 6 | 5 | 4 | 3 | 2             | 1 | 0 |
| RSVD |   |   |   |   | CUR_VAL[10:8] |   |   |
| R-0h |   |   |   |   | R-0h          |   |   |

**Table 7-168. PWM0\_CUR\_VAL\_MSB Register Field Descriptions**

| Bit | Field         | Type | Reset | Description |
|-----|---------------|------|-------|-------------|
| 7-3 | RSVD          | R    | 0h    |             |
| 2-0 | CUR_VAL[10:8] | R    | 0h    |             |

### 7.6.8.9 PWM0\_CUR\_VAL\_LSB Register (Offset = 400Ah) [Reset = 00h]

PWM0\_CUR\_VAL\_LSB is shown in [Figure 7-128](#) and described in [Table 7-169](#).

Return to the [Summary Table](#).

**Figure 7-128. PWM0\_CUR\_VAL\_LSB Register**

|              |   |   |   |   |   |   |   |
|--------------|---|---|---|---|---|---|---|
| 7            | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CUR_VAL[7:0] |   |   |   |   |   |   |   |
| R-0h         |   |   |   |   |   |   |   |

**Table 7-169. PWM0\_CUR\_VAL\_LSB Register Field Descriptions**

| Bit | Field        | Type | Reset | Description |
|-----|--------------|------|-------|-------------|
| 7-0 | CUR_VAL[7:0] | R    | 0h    |             |

### 7.6.8.10 PWM0\_CONST\_MSB Register (Offset = 400Bh) [Reset = 00h]

PWM0\_CONST\_MSB is shown in [Figure 7-129](#) and described in [Table 7-170](#).

Return to the [Summary Table](#).

If configured to vary/ramp frequency, this is used for the static duty cycle MSB (if 10-bit duty cycle resolution is enabled). If configured to vary/ramp duty cycle, this is used for the constant integer divisor MSB of the switching frequency. If configured for static PWM output, this is used for the duty cycle MSB (if 10-bit duty cycle resolution is enabled).

**Figure 7-129. PWM0\_CONST\_MSB Register**

| 7    | 6 | 5 | 4 | 3 | 2        | 1       | 0 |
|------|---|---|---|---|----------|---------|---|
| RSVD |   |   |   |   | CV_F[10] | CV[9:8] |   |
| R-0h |   |   |   |   | R/W-0h   | R/W-0h  |   |

**Table 7-170. PWM0\_CONST\_MSB Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                            |
|-----|----------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD     | R    | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                        |
| 2   | CV_F[10] | R/W  | 0h    | Constant Value (Frequency Only)<br>This is a multi-purpose register.<br>PWM0_CTRL.MODE = Duty Cycle, then this is the integer value of the switching frequency divisor CONST_FREQ[10].<br>If PWM0_CTRL.MODE = Frequency or Static, then this bit is unused                                                                                                                                             |
| 1-0 | CV[9:8]  | R/W  | 0h    | Constant Value<br>This is a multi-purpose register<br>PWM<br>0_CTRL.MODE = Duty Cycle, then this is the integer value of the switching frequency divisor CONST_FREQ[9:8]<br>If PWM<br>0_CTRL.MODE = (Frequency OR Static) AND PWM<br>0_CTRL.<br>8B =<br>0 (<br>10-bit), then this is the duty cycle value CONST_DC[9:8] (CONST_DC/1024 = duty cycle).<br>For<br>8-bit resolution mode, this is unused. |

#### 7.6.8.11 PWM0\_CONST\_LSB Register (Offset = 400Ch) [Reset = 00h]

PWM0\_CONST\_LSB is shown in [Figure 7-130](#) and described in [Table 7-171](#).

Return to the [Summary Table](#).

If configured to vary/ramp frequency, this is used for the static duty cycle LSB. If configured to vary/ramp duty cycle, this is used for the constant integer divisor LSB of the switching frequency. If configured for static PWM output, this is used for the duty cycle LSB.

**Figure 7-130. PWM0\_CONST\_LSB Register**

| 7       | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---------|---|---|---|---|---|---|---|
| CV[7:0] |   |   |   |   |   |   |   |
| R/W-0h  |   |   |   |   |   |   |   |

**Table 7-171. PWM0\_CONST\_LSB Register Field Descriptions**

| Bit | Field   | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                             |
|-----|---------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | CV[7:0] | R/W  | 0h    | Constant Value<br>This is a multi-purpose register<br>PWM<br>0_CTRL.MODE = Duty Cycle, then this is the integer value of the switching frequency divisor CONST_FREQ[7:0].<br>If PWM<br>0_CTRL.MODE = Frequency or Static, then this is the duty cycle divisor CONST_DC[7:0].<br>If<br>8-bit mode, then CONST_DC/256 = duty cycle.<br>If<br>10-bit mode, then CONST_DC/1024 = duty cycle |

### 7.6.8.12 PWM0\_STOP\_VAL\_FRAC\_F Register (Offset = 400Dh) [Reset = 00h]

PWM0\_STOP\_VAL\_FRAC\_F is shown in [Figure 7-131](#) and described in [Table 7-172](#).

Return to the [Summary Table](#).

If configured to vary/ramp frequency, this is used for the stop/off-target fractional divisor of the frequency. This is not used if configured to ramp duty cycle or for static PWM.

**Figure 7-131. PWM0\_STOP\_VAL\_FRAC\_F Register**

|      |          |   |   |   |   |   |   |
|------|----------|---|---|---|---|---|---|
| 7    | 6        | 5 | 4 | 3 | 2 | 1 | 0 |
| RSVD | SPV_FR_F |   |   |   |   |   |   |
| R-0h | R/W-0h   |   |   |   |   |   |   |

**Table 7-172. PWM0\_STOP\_VAL\_FRAC\_F Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                                                      |
|-----|----------|------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | RSVD     | R    | 0h    |                                                                                                                                                                                                                                  |
| 6-0 | SPV_FR_F | R/W  | 0h    | Stop Value Fractional Value (Frequency Only)<br>Only used when PWM0_CTRL.MODE = Frequency.<br>Interpreted as the x/128 fractional portion of the frequency divisor.<br>If PWM0_CTRL.MODE = Duty Cycle or Static, this is unused. |

### 7.6.8.13 PWM0\_STOP\_VAL\_MSB Register (Offset = 400Eh) [Reset = 00h]

PWM0\_STOP\_VAL\_MSB is shown in [Figure 7-132](#) and described in [Table 7-173](#).

Return to the [Summary Table](#).

If configured to vary/ramp duty cycle, this is used for the duty cycle MSB (only used if 10-bit duty cycle resolution is enabled). If configured to vary/ramp frequency, this is used for the stop/off-target integer divisor MSB of the frequency. This is not used if configured for static PWM

**Figure 7-132. PWM0\_STOP\_VAL\_MSB Register**

| 7    | 6 | 5 | 4 | 3         | 2 | 1        | 0 |
|------|---|---|---|-----------|---|----------|---|
| RSVD |   |   |   | SPV_F[10] |   | SPV[9:8] |   |
| R-0h |   |   |   | R/W-0h    |   | R/W-0h   |   |

**Table 7-173. PWM0\_STOP\_VAL\_MSB Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                                                                                                                                                                                                                                  |
|-----|-----------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD      | R    | 0h    |                                                                                                                                                                                                                                                                                                              |
| 2   | SPV_F[10] | R/W  | 0h    | Stop Value (Frequency Only)<br>This is a multi-purpose register.<br>If PWM0_CTRL.MODE = Frequency, then this is the integer value of the switching frequency divisor STOP_VAL[10].<br>If PWM0_CTRL.MODE = Duty Cycle, then this bit is unused.<br>If PWM0_CTRL.MODE = Static, this is unused.                |
| 1-0 | SPV[9:8]  | R/W  | 0h    | Stop Value<br>This is a multi-purpose register.<br>If PWM0_CTRL.MODE = Duty Cycle AND PWM0_CTRL.MODE = Frequency, then this is the integer value of the switching frequency divisor STOP_VAL[10].<br>If PWM0_CTRL.MODE = Duty Cycle, then this bit is unused.<br>If PWM0_CTRL.MODE = Static, this is unused. |

#### 7.6.8.14 PWM0\_STOP\_VAL\_LSB Register (Offset = 400Fh) [Reset = 00h]

PWM0\_STOP\_VAL\_LSB is shown in [Figure 7-133](#) and described in [Table 7-174](#).

Return to the [Summary Table](#).

If configured to vary/ramp duty cycle, this is used for the duty cycle LSB. If configured to vary/ramp frequency, this is used for the stop/off-target integer divisor LSB of the frequency. This is not used if configured for static PWM

**Figure 7-133. PWM0\_STOP\_VAL\_LSB Register**

|          |   |   |   |   |   |   |   |
|----------|---|---|---|---|---|---|---|
| 7        | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SPV[7:0] |   |   |   |   |   |   |   |
| R/W-0h   |   |   |   |   |   |   |   |

**Table 7-174. PWM0\_STOP\_VAL\_LSB Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-----|----------|------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | SPV[7:0] | R/W  | 0h    | <p>Stop Value</p> <p>This is a multi-purpose register.</p> <p>PWM</p> <p>0_CTRL.MODE = Duty Cycle, then this is the duty cycle value SPV[7:0].</p> <p>If</p> <p>8-bit mode, then SPV/256 = duty cycle.</p> <p>If</p> <p>10-bit mode, then SPV/1024 = duty cycle.</p> <p>If PWM</p> <p>0_CTRL.MODE = Frequency, then this is the integer value of the switching frequency divisor STOP_VAL[7:0].</p> <p>If PWM</p> <p>0_CTRL.MODE = Static, this is unused.</p> |

#### 7.6.8.15 PWM0\_STOP\_SL\_MSB Register (Offset = 4010h) [Reset = 00h]

PWM0\_STOP\_SL\_MSB is shown in [Figure 7-134](#) and described in [Table 7-175](#).

Return to the [Summary Table](#).

Used to configure the slope scale of the stop ramp

**Figure 7-134. PWM0\_STOP\_SL\_MSB Register**

| 7    | 6 | 5 | 4            | 3 | 2 | 1 | 0 |
|------|---|---|--------------|---|---|---|---|
| RSVD |   |   | SP_SL[20:16] |   |   |   |   |
| R-0h |   |   | R/W-0h       |   |   |   |   |

**Table 7-175. PWM0\_STOP\_SL\_MSB Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                                                                                                                                                                  |
|-----|--------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD         | R    | 0h    |                                                                                                                                                                                                                              |
| 4-0 | SP_SL[20:16] | R/W  | 0h    | Stop Slope<br>The slope for the stop ramp.<br>The fields vary depending on the value of<br>PWM0_CTRL.SLOPE_SCALE.<br>See <a href="#">Section 7.5.7.4</a> for more information<br>If PWM0_CTRL.MODE = Static, this is unused. |

#### 7.6.8.16 PWM0\_STOP\_SL\_MID Register (Offset = 4011h) [Reset = 00h]

PWM0\_STOP\_SL\_MID is shown in [Figure 7-135](#) and described in [Table 7-176](#).

Return to the [Summary Table](#).

Used to configure the slope scale of the stop ramp

**Figure 7-135. PWM0\_STOP\_SL\_MID Register**

|             |   |   |   |   |   |   |   |
|-------------|---|---|---|---|---|---|---|
| 7           | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SP_SL[15:8] |   |   |   |   |   |   |   |
| R/W-0h      |   |   |   |   |   |   |   |

**Table 7-176. PWM0\_STOP\_SL\_MID Register Field Descriptions**

| Bit | Field       | Type | Reset | Description                                                                                                                                                                                                               |
|-----|-------------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | SP_SL[15:8] | R/W  | 0h    | Stop Slope<br>The slope for the stop ramp.<br>The fields vary depending on the value of PWM0_CTRL.SLOPE_SCALE.<br>See <a href="#">Section 7.5.7.4</a> for more information<br>If PWM0_CTRL.MODE = Static, this is unused. |

### 7.6.8.17 PWM0\_STOP\_SL\_LSB Register (Offset = 4012h) [Reset = 00h]

PWM0\_STOP\_SL\_LSB is shown in [Figure 7-136](#) and described in [Table 7-177](#).

Return to the [Summary Table](#).

Used to configure the slope scale of the stop ramp

**Figure 7-136. PWM0\_STOP\_SL\_LSB Register**

|            |   |   |   |   |   |   |   |
|------------|---|---|---|---|---|---|---|
| 7          | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SP_SL[7:0] |   |   |   |   |   |   |   |
| R/W-0h     |   |   |   |   |   |   |   |

**Table 7-177. PWM0\_STOP\_SL\_LSB Register Field Descriptions**

| Bit | Field      | Type | Reset | Description                                                                                                                                                                                                               |
|-----|------------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | SP_SL[7:0] | R/W  | 0h    | Stop Slope<br>The slope for the stop ramp.<br>The fields vary depending on the value of PWM0_CTRL.SLOPE_SCALE.<br>See <a href="#">Section 7.5.7.4</a> for more information<br>If PWM0_CTRL.MODE = Static, this is unused. |

#### 7.6.8.18 PWM0\_START\_VAL\_FRAC\_F Register (Offset = 4013h) [Reset = 00h]

PWM0\_START\_VAL\_FRAC\_F is shown in [Figure 7-137](#) and described in [Table 7-178](#).

Return to the [Summary Table](#).

If configured to vary/ramp frequency, this is used for the start fractional divisor of the frequency. This is not used if configured to ramp duty cycle or for static PWM.

**Figure 7-137. PWM0\_START\_VAL\_FRAC\_F Register**

| 7    | 6        | 5 | 4 | 3 | 2 | 1 | 0 |
|------|----------|---|---|---|---|---|---|
| RSVD | SPV_FR_F |   |   |   |   |   |   |
| R-0h | R/W-0h   |   |   |   |   |   |   |

**Table 7-178. PWM0\_START\_VAL\_FRAC\_F Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                                                       |
|-----|----------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | RSVD     | R    | 0h    |                                                                                                                                                                                                                                   |
| 6-0 | SPV_FR_F | R/W  | 0h    | Start Value Fractional Value (Frequency Only)<br>Only used when PWM0_CTRL.MODE = Frequency.<br>Interpreted as the x/128 fractional portion of the frequency divisor.<br>If PWM0_CTRL.MODE = Duty Cycle or Static, this is unused. |

### 7.6.8.19 PWM0\_START\_VAL\_MSB Register (Offset = 4014h) [Reset = 00h]

PWM0\_START\_VAL\_MSB is shown in [Figure 7-138](#) and described in [Table 7-179](#).

Return to the [Summary Table](#).

If configured to vary/ramp duty cycle, this is used for the duty cycle MSB (only used if 10-bit duty cycle resolution is enabled). If configured to vary/ramp frequency, this is used for the start integer divisor MSB of the frequency. This is not used if configured for static PWM

**Figure 7-138. PWM0\_START\_VAL\_MSB Register**

| 7    | 6 | 5 | 4 | 3         | 2 | 1        | 0 |
|------|---|---|---|-----------|---|----------|---|
| RSVD |   |   |   | STV_F[10] |   | STV[9:8] |   |
| R-0h |   |   |   | R/W-0h    |   | R/W-0h   |   |

**Table 7-179. PWM0\_START\_VAL\_MSB Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-----|-----------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD      | R    | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 2   | STV_F[10] | R/W  | 0h    | Start Value (Frequency Only)<br>This is a multi-purpose register.<br>If PWM0_CTRL.MODE = Frequency , then this is the integer value of the switching frequency divisor START_VAL[10].<br>If PWM0_CTRL.MODE = Duty Cycle, then this bit is unused.<br>If PWM0_CTRL.MODE = Static, this is unused..                                                                                                                                                                |
| 1-0 | STV[9:8]  | R/W  | 0h    | Start Value<br>This is a multi-purpose register.<br>PWM<br>0_CTRL.MODE = Duty Cycle AND PWM<br>0_CTRL.<br>8B =<br>0 (<br>10-bit), then this is the duty cycle value STV[<br>9:<br>8], otherwise this is unused.<br>STV/<br>1024 = duty cycle, and in<br>8-bit mode is unused.<br>If PWM<br>0_CTRL.MODE = Frequency, then this is the integer value of the switching frequency divisor START_VAL[<br>9:<br>8].<br>If PWM<br>0_CTRL.MODE = Static, this is unused. |

#### 7.6.8.20 PWM0\_START\_VAL\_LSB Register (Offset = 4015h) [Reset = 00h]

PWM0\_START\_VAL\_LSB is shown in [Figure 7-139](#) and described in [Table 7-180](#).

Return to the [Summary Table](#).

If configured to vary/ramp duty cycle, this is used for the duty cycle LSB. If configured to vary/ramp frequency, this is used for the start integer divisor LSB of the frequency. This is not used if configured for static PWM

**Figure 7-139. PWM0\_START\_VAL\_LSB Register**

| 7        | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|---|---|---|---|---|---|---|
| STV[7:0] |   |   |   |   |   |   |   |
| R/W-0h   |   |   |   |   |   |   |   |

**Table 7-180. PWM0\_START\_VAL\_LSB Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-----|----------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | STV[7:0] | R/W  | 0h    | <p>Start Value</p> <p>This is a multi-purpose register.</p> <p>If PWM</p> <p>0_CTRL.MODE = Duty Cycle, then this is the duty cycle value STV[7:0].</p> <p>If</p> <p>8-bit mode, then STV/256 = duty cycle.</p> <p>If</p> <p>10-bit mode, then STV/1024 = duty cycle.</p> <p>If PWM</p> <p>0_CTRL.MODE = Frequency, then this is the integer value of the switching frequency divisor START_VAL[7:0].</p> <p>If PWM</p> <p>0_CTRL.MODE = Static, this is unused.</p> |

### 7.6.8.21 PWM0\_START\_SL\_MSB Register (Offset = 4016h) [Reset = 00h]

PWM0\_START\_SL\_MSB is shown in [Figure 7-140](#) and described in [Table 7-181](#).

Return to the [Summary Table](#).

Used to configure the slope scale of the start ramp

**Figure 7-140. PWM0\_START\_SL\_MSB Register**

| 7    | 6 | 5 | 4            | 3 | 2 | 1 | 0 |
|------|---|---|--------------|---|---|---|---|
| RSVD |   |   | ST_SL[20:16] |   |   |   |   |
| R-0h |   |   | R/W-0h       |   |   |   |   |

**Table 7-181. PWM0\_START\_SL\_MSB Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                                                                                                                                                                 |
|-----|--------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD         | R    | 0h    |                                                                                                                                                                                                                             |
| 4-0 | ST_SL[20:16] | R/W  | 0h    | Start Slope<br>The slope for the start ramp.<br>The fields vary depending on the value of PWM0_CTRL.SLOPE_SCALE.<br>See <a href="#">Section 7.5.7.4</a> for more information<br>If PWM0_CTRL.MODE = Static, this is unused. |

### 7.6.8.22 PWM0\_START\_SL\_MID Register (Offset = 4017h) [Reset = 00h]

PWM0\_START\_SL\_MID is shown in [Figure 7-141](#) and described in [Table 7-182](#).

Return to the [Summary Table](#).

Used to configure the slope scale of the start ramp

**Figure 7-141. PWM0\_START\_SL\_MID Register**

|             |   |   |   |   |   |   |   |
|-------------|---|---|---|---|---|---|---|
| 7           | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ST_SL[15:8] |   |   |   |   |   |   |   |
| R/W-0h      |   |   |   |   |   |   |   |

**Table 7-182. PWM0\_START\_SL\_MID Register Field Descriptions**

| Bit | Field       | Type | Reset | Description                                                                                                                                                                                                                 |
|-----|-------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | ST_SL[15:8] | R/W  | 0h    | Start Slope<br>The slope for the start ramp.<br>The fields vary depending on the value of PWM0_CTRL.SLOPE_SCALE.<br>See <a href="#">Section 7.5.7.4</a> for more information<br>If PWM0_CTRL.MODE = Static, this is unused. |

### 7.6.8.23 PWM0\_START\_SL\_LSB Register (Offset = 4018h) [Reset = 00h]

PWM0\_START\_SL\_LSB is shown in [Figure 7-142](#) and described in [Table 7-183](#).

Return to the [Summary Table](#).

Used to configure the slope scale of the start ramp

**Figure 7-142. PWM0\_START\_SL\_LSB Register**

|            |   |   |   |   |   |   |   |
|------------|---|---|---|---|---|---|---|
| 7          | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ST_SL[7:0] |   |   |   |   |   |   |   |
| R/W-0h     |   |   |   |   |   |   |   |

**Table 7-183. PWM0\_START\_SL\_LSB Register Field Descriptions**

| Bit | Field      | Type | Reset | Description                                                                                                                                                                                                                    |
|-----|------------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | ST_SL[7:0] | R/W  | 0h    | Start Slope<br>The slope for the start ramp.<br>The fields vary depending on the value of<br>PWM0_CTRL.SLOPE_SCALE.<br>See <a href="#">Section 7.5.7.4</a> for more information<br>If PWM0_CTRL.MODE = Static, this is unused. |

#### 7.6.8.24 PWM0\_END\_VAL\_CONST\_FRAC\_F Register (Offset = 4019h) [Reset = 00h]

PWM0\_END\_VAL\_CONST\_FRAC\_F is shown in [Figure 7-143](#) and described in [Table 7-184](#).

Return to the [Summary Table](#).

If configured to vary frequency, this is the ending value (target value) fractional divisor for frequency. If configured to vary/ramp duty cycle, then this is the fractional divisor of the switching frequency. If configured as static PWM output, this register is used for the fractional portion of the frequency divisor.

**Figure 7-143. PWM0\_END\_VAL\_CONST\_FRAC\_F Register**

|      |           |   |   |   |   |   |   |
|------|-----------|---|---|---|---|---|---|
| 7    | 6         | 5 | 4 | 3 | 2 | 1 | 0 |
| RSVD | EVCV_FR_F |   |   |   |   |   |   |
| R-0h | R/W-0h    |   |   |   |   |   |   |

**Table 7-184. PWM0\_END\_VAL\_CONST\_FRAC\_F Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-----|-----------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | RSVD      | R    | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 6-0 | EVCV_FR_F | R/W  | 0h    | End Value Fractional (Frequency Mode) or Const Value Fractional (DC Mode)<br>End value fractional portion.<br>This is always used, but the field changes based upon the mode currently in use.<br>Interpreted as the x/128 fractional portion of the frequency divisor.<br>PWM0_CTRL.MODE = Duty Cycle, then this is the integer value of the switching frequency divisor CONST_FREQ[10].<br>If PWM0_CTRL.MODE = Frequency or Static, then this bit is unused |

### 7.6.8.25 PWM0\_END\_VAL\_MSB Register (Offset = 401Ah) [Reset = 00h]

PWM0\_END\_VAL\_MSB is shown in [Figure 7-144](#) and described in [Table 7-185](#).

Return to the [Summary Table](#).

Settings for the varied PWM ending value (target value) if configured to ramp. If configured as static PWM output, this register is used for the integer MSB portion of the frequency divisor.

**Figure 7-144. PWM0\_END\_VAL\_MSB Register**

| 7    | 6 | 5 | 4 | 3         | 2 | 1            | 0 |
|------|---|---|---|-----------|---|--------------|---|
| RSVD |   |   |   | EV_FR[10] |   | END_VAL[9:8] |   |
| R-0h |   |   |   | R/W-0h    |   | R/W-0h       |   |

**Table 7-185. PWM0\_END\_VAL\_MSB Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----|--------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD         | R    | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 2   | EV_FR[10]    | R/W  | 0h    | End Value (Frequency Only)<br>This is a multi-purpose register.<br>PWM0_CTRL.MODE = Duty Cycle or Static, then this is the integer value of the switching frequency value START_VAL[10].<br>If PWM0_CTRL.MODE = Frequency, then this bit is unused                                                                                                                                                                           |
| 1-0 | END_VAL[9:8] | R/W  | 0h    | End Value<br>This is a multi-purpose register.<br>PWM<br>0_CTRL.MODE = Duty Cycle or Static, then this is the integer value of the switching frequency divisor END_VAL[9:8].<br>If PWM<br>0_CTRL.MODE = Frequency AND PWM<br>0_CTRL.MODE = 8B = 0 (10-bit), then this is the duty cycle value END_VAL[9:8], otherwise this is unused.<br>If<br>10-bit mode, then END_VAL/1024 = duty cycle, but in 8-bit mode this is unused |

#### 7.6.8.26 PWM0\_END\_VAL\_LSB Register (Offset = 401Bh) [Reset = 00h]

PWM0\_END\_VAL\_LSB is shown in [Figure 7-145](#) and described in [Table 7-186](#).

Return to the [Summary Table](#).

Settings for the varied PWM ending value (target value) if configured to ramp. If configured as static PWM output, this register is used for the integer LSB portion of the frequency divisor.

**Figure 7-145. PWM0\_END\_VAL\_LSB Register**

| 7            | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------------|---|---|---|---|---|---|---|
| END_VAL[7:0] |   |   |   |   |   |   |   |
| R/W-0h       |   |   |   |   |   |   |   |

**Table 7-186. PWM0\_END\_VAL\_LSB Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                    |
|-----|--------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | END_VAL[7:0] | R/W  | 0h    | <p>End Value<br/>This is a multi-purpose register.</p> <p>PWM<br/>0_CTRL.MODE = Duty Cycle or Static, then this is the integer value of the switching frequency divisor END_VAL[7:0].<br/>If PWM<br/>0_CTRL.MODE = Frequency, then this is the duty cycle value END_VAL[7:0].<br/>If<br/>8-bit mode, then END_VAL/256 = duty cycle.<br/>If<br/>10-bit mode, then END_VAL/1024 = duty cycle</p> |

#### 7.6.8.27 PWM0\_PULSE\_STOP\_RAMP[y] Register (Offset = 401Ch + formula) [Reset = 00000000h]

PWM0\_PULSE\_STOP\_RAMP[y] is shown in [Figure 7-146](#) and described in [Table 7-187](#).

Return to the [Summary Table](#).

If configured for automatic stop-ramp, this is the number of generated PWM pulses from the starting pulse that the module will automatically set the stop ramp command. The lowest address is corresponds to the MSB of the counter.

Offset = 401Ch + (y \* 4h); where y = 0h to 3h

**Figure 7-146. PWM0\_PULSE\_STOP\_RAMP[y] Register**

| 31                   | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| PWM0_PULSE_STOP_RAMP |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| R/W-0h               |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

**Table 7-187. PWM0\_PULSE\_STOP\_RAMP[y] Register Field Descriptions**

| Bit  | Field                | Type | Reset | Description                                                                                                                                                                                    |
|------|----------------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 31-0 | PWM0_PULSE_STOP_RAMP | R/W  | 0h    | Once the specified number of PWM pulses are generated, the stop-ramp is automatically started.<br>This is only used if the PWM0_ACTION.AUTO_STOP bit is set.<br>The lowest address is the MSB. |

### 7.6.8.28 PWM0\_PULSE\_MAX[y] Register (Offset = 4020h + formula) [Reset = 00000000h]

PWM0\_PULSE\_MAX[y] is shown in [Figure 7-147](#) and described in [Table 7-188](#).

Return to the [Summary Table](#).

If configured for the automatic stop-ramp, this is the maximum number of PWM pulses that will be allowed. If the number of PWM pulses hits this limit, then the output will immediately be stopped. The lowest address is corresponds to the MSB of the counter.

Offset = 4020h + (y \* 4h); where y = 0h to 3h

**Figure 7-147. PWM0\_PULSE\_MAX[y] Register**

|                |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
|----------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PWM0_PULSE_MAX |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| R/W-0h         |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

**Table 7-188. PWM0\_PULSE\_MAX[y] Register Field Descriptions**

| Bit  | Field          | Type | Reset | Description                                                                                                                                                                        |
|------|----------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 31-0 | PWM0_PULSE_MAX | R/W  | 0h    | Once the specified number of PWM pulses are generated, the PWM output is disabled.<br>This is only used if the PWM0_ACTION.AUTO_STOP bit is set.<br>The lowest address is the MSB. |

**7.6.8.29 PWM0\_ACTION Register (Offset = 4024h) [Reset = 00h]**

PWM0\_ACTION is shown in [Figure 7-148](#) and described in [Table 7-189](#).

Return to the [Summary Table](#).

**Figure 7-148. PWM0\_ACTION Register**

| 7    | 6 | 5 | 4         | 3         | 2      | 1         | 0         |
|------|---|---|-----------|-----------|--------|-----------|-----------|
| RSVD |   |   | COUNT_RST | AUTO_STOP | UDS    | START     | STOP      |
| R-0h |   |   | R/W1S-0h  | R/W-0h    | R/W-0h | RH/W1S-0h | RH/W1S-0h |

**Table 7-189. PWM0\_ACTION Register Field Descriptions**

| Bit | Field     | Type   | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-----|-----------|--------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD      | R      | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 4   | COUNT_RST | R/W1S  | 0h    | <p>Reset the pulse counter to 0<br/>If this bit is set, the pulse counter is reset.<br/>This is typically performed at the start of a ramp, in order to keep track of the total number of pulses.<br/>It gets cleared by hardware once the reset is complete.</p> <hr/> <p><b>Note</b></p> <p>Pay attention to the stop and maximum pulse count values if using AUTO_STOP.<br/>Those pulse counters are compared against the current pulse counter, and if the maximum pulse count is less than the current pulse counter value, no PWM output will be generated</p> <hr/> |
| 3   | AUTO_STOP | R/W    | 0h    | <p>Use automatic stop-ramp<br/>If enabled, once the pulse count hits the specified<br/>0h = Do not use automatic-stop-ramp behavior<br/>1h = Automatically begin the stop-ramp based on the pulse count registers</p>                                                                                                                                                                                                                                                                                                                                                      |
| 2   | UDS       | R/W    | 0h    | <p>Use Defined Start<br/>Use the start point defined in the start value register.<br/>Used to ensure the start point is known, or if you want the ramp to take place from the current value (mid ramp or existing end point)<br/>0h = Use the existing current value as the starting point<br/>1h = Start from the defined start point</p>                                                                                                                                                                                                                                 |
| 1   | START     | RH/W1S | 0h    | <p>Start<br/>Start the defined ramp profile based on configured settings.<br/>If STOP is also set at same time, the START will be ignored.<br/>Bit is cleared by hardware.</p>                                                                                                                                                                                                                                                                                                                                                                                             |
| 0   | STOP      | RH/W1S | 0h    | <p>Stop<br/>Will ramp to the specified stop point and then turn off.<br/>If START and STOP are set together, STOP will occur.<br/>Bit is cleared by hardware</p>                                                                                                                                                                                                                                                                                                                                                                                                           |

### 7.6.8.30 PWM0\_IAS\_CTRL Register (Offset = 4030h) [Reset = 00h]

PWM0\_IAS\_CTRL is shown in [Figure 7-149](#) and described in [Table 7-190](#).

Return to the [Summary Table](#).

**Figure 7-149. PWM0\_IAS\_CTRL Register**

| 7    | 6 | 5 | 4      | 3        | 2 | 1         | 0      |
|------|---|---|--------|----------|---|-----------|--------|
| RSVD |   |   | IN_POL | GPIO_SEL |   | STOP_MODE | AS_EN  |
| R-0h |   |   | R/W-0h | R/W-0h   |   | R/W-0h    | R/W-0h |

**Table 7-190. PWM0\_IAS\_CTRL Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                   |
|-----|-----------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD      | R    | 0h    | Reserved                                                                                                                                                                                                                                                                                                                                                      |
| 4   | IN_POL    | R/W  | 0h    | Input Polarity<br>Selects which level is used to trigger the stop behavior<br>0h = Active low input signal<br>1h = Active high input signal                                                                                                                                                                                                                   |
| 3-2 | GPIO_SEL  | R/W  | 0h    | GPIO Select<br>Specifies which GPIO is used.<br>GPIO must be configured as a GPIO function instead of special function in order to work.<br>0h = GPIO0<br>1h = GPIO1<br>2h = GPIO7<br>3h = GPIO8                                                                                                                                                              |
| 1   | STOP_MODE | R/W  | 0h    | Stop Mode<br>When a GPIO triggered stop event occurs, select between immediately disabling PWM output, or requesting the stop ramp.<br>0h = GPIO triggered event requests a stop ramp<br>1h = GPIO triggered event immediately disables PWM output                                                                                                            |
| 0   | AS_EN     | R/W  | 0h    | Input Auto Stop Enable<br><br><div style="text-align: center;"><b>Note</b></div> <div style="text-align: center;">This does NOT require the AUTO_STOP bit to be set,<br/>and is a separate function to stop the output based on a<br/>GPIO input</div><br>0h = No auto stop is enabled from GPIO signal<br>1h = Enable GPIO triggered auto stop functionality |

### 7.6.9 PWM1 Registers

[Section 7.6.9](#) lists the memory-mapped registers for the PWM1 registers. All register offset addresses not listed in [Section 7.6.9](#) should be considered as reserved locations and the register contents should not be modified.

**Table 7-191. PWM1 Registers**

| Offset          | Acronym           | Register Name              | Section                         |
|-----------------|-------------------|----------------------------|---------------------------------|
| 4100h           | PWM1_CTRL         | PWM1 Control               | <a href="#">Section 7.6.9.1</a> |
| 4101h           | PWM1_IE0          | Interrupt Enable           | <a href="#">Section 7.6.9.2</a> |
| 4102h           | PWM1_IE1          | Interrupt Enable           | <a href="#">Section 7.6.9.3</a> |
| 4103h           | PWM1_IR           | Interrupt Register         | <a href="#">Section 7.6.9.4</a> |
| 4104h           | PWM1_STATUS       | Status                     | <a href="#">Section 7.6.9.5</a> |
| 4105h + formula | PWM1_CUR_PULSE[y] | Current Pulse Count        | <a href="#">Section 7.6.9.6</a> |
| 4109h           | PWM1_CUR_VAL_MSB  | Current Value MSB          | <a href="#">Section 7.6.9.7</a> |
| 410Ah           | PWM1_CUR_VAL_LSB  | Current Value LSB          | <a href="#">Section 7.6.9.8</a> |
| 410Bh           | PWM1_CONST_MSB    | Constant Value Integer MSB | <a href="#">Section 7.6.9.9</a> |

**Table 7-191. PWM1 Registers (continued)**

| Offset          | Acronym                   | Register Name                             | Section                          |
|-----------------|---------------------------|-------------------------------------------|----------------------------------|
| 410Ch           | PWM1_CONST_LSB            | Constant Value Integer LSB                | <a href="#">Section 7.6.9.10</a> |
| 410Dh           | PWM1_STOP_VAL_FRAC_F      | Stop Fractional Value (Frequency Only)    | <a href="#">Section 7.6.9.11</a> |
| 410Eh           | PWM1_STOP_VAL_MSB         | Stop Value Integer MSB                    | <a href="#">Section 7.6.9.12</a> |
| 410Fh           | PWM1_STOP_VAL_LSB         | Stop Value Integer LSB                    | <a href="#">Section 7.6.9.13</a> |
| 4110h           | PWM1_STOP_SL_MSB          | Stop Slope MSB                            | <a href="#">Section 7.6.9.14</a> |
| 4111h           | PWM1_STOP_SL_MID          | Stop Slope Mid                            | <a href="#">Section 7.6.9.15</a> |
| 4112h           | PWM1_STOP_SL_LSB          | Stop Slope LSB                            | <a href="#">Section 7.6.9.16</a> |
| 4113h           | PWM1_START_VAL_FRAC_F     | Start Fractional Value (Frequency Only)   | <a href="#">Section 7.6.9.17</a> |
| 4114h           | PWM1_START_VAL_MSB        | Start Value MSB                           | <a href="#">Section 7.6.9.18</a> |
| 4115h           | PWM1_START_VAL_LSB        | Start Value LSB                           | <a href="#">Section 7.6.9.19</a> |
| 4116h           | PWM1_START_SL_MSB         | Start Slope MSB                           | <a href="#">Section 7.6.9.20</a> |
| 4117h           | PWM1_START_SL_MID         | Start Slope Mid Byte                      | <a href="#">Section 7.6.9.21</a> |
| 4118h           | PWM1_START_SL_LSB         | Start Slope LSB                           | <a href="#">Section 7.6.9.22</a> |
| 4119h           | PWM1_END_VAL_CONST_FRAC_F | End (Freq) or Const (DC) Fractional Value | <a href="#">Section 7.6.9.23</a> |
| 411Ah           | PWM1_END_VAL_MSB          | End Value MSB                             | <a href="#">Section 7.6.9.24</a> |
| 411Bh           | PWM1_END_VAL_LSB          | End Value LSB                             | <a href="#">Section 7.6.9.25</a> |
| 411Ch + formula | PWM1_PULSE_STOP_RAMP[y]   | Pulse Count to Begin Stop Ramp            | <a href="#">Section 7.6.9.26</a> |
| 4120h + formula | PWM1_PULSE_MAX[y]         | Pulse Count Maximum                       | <a href="#">Section 7.6.9.27</a> |
| 4124h           | PWM1_ACTION               | Start Action Register.                    | <a href="#">Section 7.6.9.28</a> |
| 4130h           | PWM1_IAS_CTRL             | Input Auto Stop Control                   | <a href="#">Section 7.6.9.29</a> |

Complex bit access types are encoded to fit into small table cells. [Section 7.6.9](#) shows the codes that are used for access types in this section.

**Table 7-192. PWM1 Access Type Codes**

| Access Type              | Code    | Description                                                                                                                                                                                                                                                                          |
|--------------------------|---------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Read Type                |         |                                                                                                                                                                                                                                                                                      |
| R                        | R       | Read                                                                                                                                                                                                                                                                                 |
| RH                       | R<br>H  | Read<br>Set or cleared by hardware                                                                                                                                                                                                                                                   |
| Write Type               |         |                                                                                                                                                                                                                                                                                      |
| W                        | W       | Write                                                                                                                                                                                                                                                                                |
| W1C                      | W<br>1C | Write<br>1 to clear                                                                                                                                                                                                                                                                  |
| W1S                      | W<br>1S | Write<br>1 to set                                                                                                                                                                                                                                                                    |
| WP                       | W<br>P  | Write<br>Requires privileged access                                                                                                                                                                                                                                                  |
| Reset or Default Value   |         |                                                                                                                                                                                                                                                                                      |
| -n                       |         | Value after reset or the default value                                                                                                                                                                                                                                               |
| Register Array Variables |         |                                                                                                                                                                                                                                                                                      |
| i,j,k,l,m,n              |         | When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula. |

**Table 7-192. PWM1 Access Type Codes (continued)**

| Access Type | Code | Description                                                                                                         |
|-------------|------|---------------------------------------------------------------------------------------------------------------------|
| y           |      | When this variable is used in a register name, an offset, or an address it refers to the value of a register array. |

### 7.6.9.1 PWM1\_CTRL Register (Offset = 4100h) [Reset = 11h]

PWM1\_CTRL is shown in [Figure 7-150](#) and described in [Table 7-193](#).

Return to the [Summary Table](#).

Control fields to configure the PWM1 module

**Figure 7-150. PWM1\_CTRL Register**

| 7    | 6           | 5 | 4 | 3       | 2 | 1       | 0      |
|------|-------------|---|---|---------|---|---------|--------|
| RSVD | SLOPE_SCALE |   |   | MODE    |   | DC_8B   | INIT   |
| R-0h | R/WP-1h     |   |   | R/WP-0h |   | R/WP-0h | R/W-1h |

**Table 7-193. PWM1\_CTRL Register Field Descriptions**

| Bit | Field       | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-----|-------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | RSVD        | R    | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 6-4 | SLOPE_SCALE | R/WP | 1h    | <p>Slope Scale</p> <p>Adjusts the number of bits that are used for integer vs fractional portions of the slope register.</p> <p>The meaning of the enumerations change depending on if frequency or duty cycle is being ramped.</p> <p>See <a href="#">Section 7.5.7.4</a> for more information</p> <p>0h = Int [20-16], Frac [15-0] (Frequency) - Int [20], Frac [19-0] (Duty Cycle)</p> <p>1h = Int [20-12], Frac [11-0] (Frequency) - Int [20-19], Frac [18-0] (Duty Cycle)</p> <p>2h = Int [20-8], Frac [7-0] (Frequency) - Int [20-18], Frac [17-0] (Duty Cycle)</p> <p>3h = Int [20-4], Frac [3-0] (Frequency) - Int [20-17], Frac [16-0] (Duty Cycle)</p> <p>4h = Int [20-0], No Frac (Frequency) - Int [20-16], Frac [15-0] (Duty Cycle)</p> |
| 3-2 | MODE        | R/WP | 0h    | <p>Adjusts the number of bits that are used for integer vs fractional portions of the slope register.</p> <p>Note that setting the mode to anything but off will take priority over SPI chip select if the GPIO is configured for special functionality.</p> <p>0h = Off</p> <p>1h = Duty Cycle Ramping</p> <p>2h = Frequency Ramping</p> <p>3h = Static On</p>                                                                                                                                                                                                                                                                                                                                                                                      |
| 1   | DC_8B       | R/WP | 0h    | <p>8 Bit Duty Cycle Mode</p> <p>0h = 10-bit duty cycle resolution</p> <p>1h = 8-Bit duty cycle resolution</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 0   | INIT        | R/W  | 1h    | <p>Initialization Mode</p> <p>Holds the PWM IP in reset, used for configuring settings in this PWM1_CTRL register</p> <p>0h = Initialization mode disabled. Writes to protected PWM bits are not allowed</p> <p>1h = Initialization mode enabled. Writes to protected PWM bits are allowed</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                       |

### 7.6.9.2 PWM1\_IE0 Register (Offset = 4101h) [Reset = 00h]

PWM1\_IE0 is shown in [Figure 7-151](#) and described in [Table 7-194](#).

Return to the [Summary Table](#).

**Figure 7-151. PWM1\_IE0 Register**

| 7    | 6 | 5 | 4 | 3 | 2                | 1      | 0        |
|------|---|---|---|---|------------------|--------|----------|
| RSVD |   |   |   |   | PULSE_OVF_I<br>E | IAS_IE | RESERVED |
| R-0h |   |   |   |   | R/W-0h           | R/W-0h | R-0h     |

**Table 7-194. PWM1\_IE0 Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-----|--------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD         | R    | 0h    | Reserved                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 2   | PULSE_OVF_IE | R/W  | 0h    | Current Pulse Overflow<br>Set when the CUR_PULSE counter overflows<br>0h = Current pulse counter has not overflowed<br>1h = Current pulse counter has overflowed                                                                                                                                                                                                                                                                                                                  |
| 1   | IAS_IE       | R/W  | 0h    | Input Auto Stop Interrupt<br>Sets when the input auto stop condition has occurred, if the PWM is configured to immediately stop, then the output will also be stopped (reflected by the RC bit).<br>If PWM is configured to begin the stop ramp on an IAS trigger, then then this interrupt is set immediately when the input condition happens and the RC bit will be set when the output turns off.<br>0h = Input for IAS has not triggered<br>1h = Input for IAS has triggered |
| 0   | RESERVED     | R    | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

### 7.6.9.3 PWM1\_IE1 Register (Offset = 4102h) [Reset = 00h]

PWM1\_IE1 is shown in [Figure 7-152](#) and described in [Table 7-195](#).

Return to the [Summary Table](#).

**Figure 7-152. PWM1\_IE1 Register**

| 7    | 6 | 5 | 4 | 3 | 2                | 1      | 0      |
|------|---|---|---|---|------------------|--------|--------|
| RSVD |   |   |   |   | PULSE_OVF_I<br>E | IAS_IE | RC_IE  |
| R-0h |   |   |   |   | R/W-0h           | R/W-0h | R/W-0h |

**Table 7-195. PWM1\_IE1 Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-----|--------------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD         | R    | 0h    | Reserved                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 2   | PULSE_OVF_IE | R/W  | 0h    | Current Pulse Overflow<br>Set when the CUR_PULSE counter overflows<br>0h = Current pulse counter has not overflowed<br>1h = Current pulse counter has overflowed                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 1   | IAS_IE       | R/W  | 0h    | Input Auto Stop Interrupt<br>Sets when the input auto stop condition has occurred, if the PWM is configured to immediately stop, then the output will also be stopped (reflected by the RC bit).<br>If PWM is configured to begin the stop ramp on an IAS trigger, then this interrupt is set immediately when the input condition happens and the RC bit will be set when the output turns off.<br><br><div style="text-align: center;"> <b>Note</b><br/>           If an on-ramp is in progress and a stop condition occurs, the RC bit will NOT be set for the on-ramp, but only once the output stops due to the stop condition         </div><br>0h = Input for IAS has not triggered<br>1h = Input for IAS has triggered                                |
| 0   | RC_IE        | R/W  | 0h    | PWM Ramp Complete<br>Sets when the ramp has finished. If AS_EN = 1 (auto stop), then this happens when the PWM channel turns off.<br>If AS_EN = 0, then this gets set when the PWM has reached the end value.<br>If IAS_EN = 1 (input auto stop), then this will get set once the PWM output is stopped<br>If a stop ramp is manually activated, the RC bit will be set once the ramp completes and the output is turned off.<br><br><div style="text-align: center;"> <b>Note</b><br/>           If an on-ramp is in progress and a stop condition occurs, the RC bit will NOT be set for the on-ramp, but only once the output stops due to the stop condition         </div><br>0h = PWM channel ramp is not complete<br>1h = PWM channel ramp is complete |

#### 7.6.9.4 PWM1\_IR Register (Offset = 4103h) [Reset = 00h]

PWM1\_IR is shown in [Figure 7-153](#) and described in [Table 7-196](#).

Return to the [Summary Table](#).

**Figure 7-153. PWM1\_IR Register**

| 7    | 6 | 5 | 4 | 3         | 2 | 1        | 0        |
|------|---|---|---|-----------|---|----------|----------|
| RSVD |   |   |   | PULSE_OVF |   | IAS      | RC       |
| R-0h |   |   |   | R/W1C-0h  |   | R/W1C-0h | R/W1C-0h |

**Table 7-196. PWM1\_IR Register Field Descriptions**

| Bit | Field     | Type  | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----|-----------|-------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD      | R     | 0h    | Reserved                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 2   | PULSE_OVF | R/W1C | 0h    | Current Pulse Overflow<br>Set when the CUR_PULSE counter overflows<br>0h = Current pulse counter has not overflowed<br>1h = Current pulse counter has overflowed                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 1   | IAS       | R/W1C | 0h    | Input Auto Stop Interrupt<br>Sets when the input auto stop condition has occurred, if the PWM is configured to immediately stop, then the output will also be stopped (reflected by the RC bit).<br>If PWM is configured to begin the stop ramp on an IAS trigger, then then this interrupt is set immediately when the input condition happens and the RC bit will be set when the output turns off.<br>0h = Input for IAS has not triggered<br>1h = Input for IAS has triggered                                                                                                                                                                                                                                                                                            |
| 0   | RC        | R/W1C | 0h    | PWM Ramp Complete<br>Sets when the ramp has finished. If AS_EN = 1 (auto stop), then this happens when the PWM channel turns off.<br>If AS_EN = 0, then this gets set when the PWM has reached the end value.<br>If IAS_EN = 1 (input auto stop), then this will get set once the PWM output is stopped.<br>If a stop ramp is manually activated, the RC bit will be set once the ramp completes and the output is turned off.<br><br><div style="text-align: center;"><b>Note</b></div> <div style="text-align: center;">If an on-ramp is in progress and a stop condition occurs, the RC bit will NOT be set for the on-ramp, but only once the output stops due to the stop condition</div><br>0h = PWM channel ramp is not complete<br>1h = PWM channel ramp is complete |

### 7.6.9.5 PWM1\_STATUS Register (Offset = 4104h) [Reset = 00h]

PWM1\_STATUS is shown in [Figure 7-154](#) and described in [Table 7-197](#).

Return to the [Summary Table](#).

**Figure 7-154. PWM1\_STATUS Register**

| 7    | 6 | 5 | 4 | 3       | 2 | 1     | 0     |
|------|---|---|---|---------|---|-------|-------|
| RSVD |   |   |   | IAS_ACT |   | BUSY  | OEN   |
| R-0h |   |   |   | RH-0h   |   | RH-0h | RH-0h |

**Table 7-197. PWM1\_STATUS Register Field Descriptions**

| Bit | Field   | Type | Reset | Description                                                                                                                                                                                                                                                                |
|-----|---------|------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD    | R    | 0h    |                                                                                                                                                                                                                                                                            |
| 2   | IAS_ACT | RH   | 0h    | Input Auto Stop Active<br>Gets set when the GPIO input toggles to the asserted state and will self-clear when the PWM output stops.<br>An RC interrupt bit should be set once the ramp is complete<br>0h = Input auto stop is not active<br>1h = Input auto stop is active |
| 1   | BUSY    | RH   | 0h    | Ramp IP Busy<br>0h = PWM ramp is complete<br>1h = PWM ramp is currently on-going                                                                                                                                                                                           |
| 0   | OEN     | RH   | 0h    | Output Enabled<br>0h = PWM output is disabled<br>1h = PWM output is enabled and generating output                                                                                                                                                                          |

### 7.6.9.6 PWM1\_CUR\_PULSE[y] Register (Offset = 4105h + formula) [Reset = 00000000h]

PWM1\_CUR\_PULSE[y] is shown in [Figure 7-155](#) and described in [Table 7-198](#).

Return to the [Summary Table](#).

An unsigned 32 bit counter value of the current pulse count. The lowest address is corresponds to the MSB of the counter.

Offset = 4105h + (y \* 4h); where y = 0h to 3h

**Figure 7-155. PWM1\_CUR\_PULSE[y] Register**

|                |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
|----------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PWM0_CUR_PULSE |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| R-0h           |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

**Table 7-198. PWM1\_CUR\_PULSE[y] Register Field Descriptions**

| Bit  | Field          | Type | Reset | Description                                                                                                                                                                                                                                    |
|------|----------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 31-0 | PWM0_CUR_PULSE | R    | 0h    | The current pulse count of the current ramp.<br>If the ramp has finished, then this value is the total number of pulses generated for the last ramp.<br>This value is reset each time a new ramp is started.<br>The lowest address is the MSB. |

**7.6.9.7 PWM1\_CUR\_VAL\_MSB Register (Offset = 4109h) [Reset = 00h]**

PWM1\_CUR\_VAL\_MSB is shown in [Figure 7-156](#) and described in [Table 7-199](#).

Return to the [Summary Table](#).

**Figure 7-156. PWM1\_CUR\_VAL\_MSB Register**

|      |   |   |   |   |               |   |   |
|------|---|---|---|---|---------------|---|---|
| 7    | 6 | 5 | 4 | 3 | 2             | 1 | 0 |
| RSVD |   |   |   |   | CUR_VAL[10:8] |   |   |
| R-0h |   |   |   |   | R-0h          |   |   |

**Table 7-199. PWM1\_CUR\_VAL\_MSB Register Field Descriptions**

| Bit | Field         | Type | Reset | Description |
|-----|---------------|------|-------|-------------|
| 7-3 | RSVD          | R    | 0h    |             |
| 2-0 | CUR_VAL[10:8] | R    | 0h    |             |

#### 7.6.9.8 PWM1\_CUR\_VAL\_LSB Register (Offset = 410Ah) [Reset = 00h]

PWM1\_CUR\_VAL\_LSB is shown in [Figure 7-157](#) and described in [Table 7-200](#).

Return to the [Summary Table](#).

**Figure 7-157. PWM1\_CUR\_VAL\_LSB Register**

|              |   |   |   |   |   |   |   |
|--------------|---|---|---|---|---|---|---|
| 7            | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CUR_VAL[7:0] |   |   |   |   |   |   |   |
| R-0h         |   |   |   |   |   |   |   |

**Table 7-200. PWM1\_CUR\_VAL\_LSB Register Field Descriptions**

| Bit | Field        | Type | Reset | Description |
|-----|--------------|------|-------|-------------|
| 7-0 | CUR_VAL[7:0] | R    | 0h    |             |

#### 7.6.9.9 PWM1\_CONST\_MSB Register (Offset = 410Bh) [Reset = 00h]

PWM1\_CONST\_MSB is shown in [Figure 7-158](#) and described in [Table 7-201](#).

Return to the [Summary Table](#).

If configured to vary/ramp frequency, this is used for the static duty cycle MSB (if 10-bit duty cycle resolution is enabled). If configured to vary/ramp duty cycle, this is used for the constant integer divisor MSB of the switching frequency. If configured for static PWM output, this is used for the duty cycle MSB (if 10-bit duty cycle resolution is enabled).

**Figure 7-158. PWM1\_CONST\_MSB Register**

| 7    | 6 | 5 | 4 | 3 | 2        | 1       | 0 |
|------|---|---|---|---|----------|---------|---|
| RSVD |   |   |   |   | CV_F[10] | CV[9:8] |   |
| R-0h |   |   |   |   | R/W-0h   | R/W-0h  |   |

**Table 7-201. PWM1\_CONST\_MSB Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                          |
|-----|----------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD     | R    | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                      |
| 2   | CV_F[10] | R/W  | 0h    | Constant Value (Frequency Only)<br>This is a multi-purpose register.<br>PWM1_CTRL.MODE = Duty Cycle, then this is the integer value of the switching frequency divisor CONST_FREQ[10].<br>If PWM1_CTRL.MODE = Frequency or Static, then this bit is unused                                                                                                                                           |
| 1-0 | CV[9:8]  | R/W  | 0h    | Constant Value<br>This is a multi-purpose register.<br>PWM<br>1_CTRL.MODE = Duty Cycle, then this is the integer value of the switching frequency divisor CONST_FREQ[9:8].<br>If PWM<br>1_CTRL.MODE = (Frequency OR Static) AND PWM<br>1_CTRL.<br>8B =<br>0 (10-bit), then this is the duty cycle value CONST_DC[9:8] (CONST_DC/1024 = duty cycle).<br>For<br>8-bit resolution mode, this is unused. |

#### 7.6.9.10 PWM1\_CONST\_LSB Register (Offset = 410Ch) [Reset = 00h]

PWM1\_CONST\_LSB is shown in [Figure 7-159](#) and described in [Table 7-202](#).

Return to the [Summary Table](#).

If configured to vary/ramp frequency, this is used for the static duty cycle LSB. If configured to vary/ramp duty cycle, this is used for the constant integer divisor LSB of the switching frequency. If configured for static PWM output, this is used for the duty cycle LSB.

**Figure 7-159. PWM1\_CONST\_LSB Register**

| 7       | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---------|---|---|---|---|---|---|---|
| CV[7:0] |   |   |   |   |   |   |   |
| R/W-0h  |   |   |   |   |   |   |   |

**Table 7-202. PWM1\_CONST\_LSB Register Field Descriptions**

| Bit | Field   | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-----|---------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | CV[7:0] | R/W  | 0h    | <p>Constant Value</p> <p>This is a multi-purpose register</p> <p>PWM</p> <p>1_CTRL.MODE = Duty Cycle, then this is the integer value of the switching frequency divisor CONST_FREQ[7:0].</p> <p>If PWM</p> <p>1_CTRL.MODE = Frequency or Static, then this is the duty cycle divisor CONST_DC[7:0].</p> <p>If</p> <p>8-bit mode, then CONST_DC/256 = duty cycle.</p> <p>If</p> <p>10-bit mode, then CONST_DC/1024 = duty cycle</p> |

#### 7.6.9.11 PWM1\_STOP\_VAL\_FRAC\_F Register (Offset = 410Dh) [Reset = 00h]

PWM1\_STOP\_VAL\_FRAC\_F is shown in [Figure 7-160](#) and described in [Table 7-203](#).

Return to the [Summary Table](#).

If configured to vary/ramp frequency, this is used for the stop/off-target fractional divisor of the frequency. This is not used if configured to ramp duty cycle or for static PWM.

**Figure 7-160. PWM1\_STOP\_VAL\_FRAC\_F Register**

| 7    | 6        | 5 | 4 | 3 | 2 | 1 | 0 |
|------|----------|---|---|---|---|---|---|
| RSVD | SPV_FR_F |   |   |   |   |   |   |
| R-0h | R/W-0h   |   |   |   |   |   |   |

**Table 7-203. PWM1\_STOP\_VAL\_FRAC\_F Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                                                      |
|-----|----------|------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | RSVD     | R    | 0h    |                                                                                                                                                                                                                                  |
| 6-0 | SPV_FR_F | R/W  | 0h    | Stop Value Fractional Value (Frequency Only)<br>Only used when PWM1_CTRL.MODE = Frequency.<br>Interpreted as the x/128 fractional portion of the frequency divisor.<br>If PWM1_CTRL.MODE = Duty Cycle or Static, this is unused. |

#### 7.6.9.12 PWM1\_STOP\_VAL\_MSB Register (Offset = 410Eh) [Reset = 00h]

PWM1\_STOP\_VAL\_MSB is shown in [Figure 7-161](#) and described in [Table 7-204](#).

Return to the [Summary Table](#).

If configured to vary/ramp duty cycle, this is used for the duty cycle MSB (only used if 10-bit duty cycle resolution is enabled). If configured to vary/ramp frequency, this is used for the stop/off-target integer divisor MSB of the frequency. This is not used if configured for static PWM

**Figure 7-161. PWM1\_STOP\_VAL\_MSB Register**

| 7 | 6 | 5    | 4 | 3 | 2         | 1 | 0        |
|---|---|------|---|---|-----------|---|----------|
|   |   | RSVD |   |   | SPV_F[10] |   | SPV[9:8] |
|   |   | R-0h |   |   | R/W-0h    |   | R/W-0h   |

**Table 7-204. PWM1\_STOP\_VAL\_MSB Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                                                                                                                                                                                                                                  |
|-----|-----------|------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD      | R    | 0h    |                                                                                                                                                                                                                                                                                                              |
| 2   | SPV_F[10] | R/W  | 0h    | Stop Value (Frequency Only)<br>This is a multi-purpose register.<br>If PWM1_CTRL.MODE = Frequency, then this is the integer value of the switching frequency divisor STOP_VAL[10].<br>If PWM1_CTRL.MODE = Duty Cycle, then this bit is unused.<br>If PWM1_CTRL.MODE = Static, this is unused.                |
| 1-0 | SPV[9:8]  | R/W  | 0h    | Stop Value<br>This is a multi-purpose register.<br>If PWM1_CTRL.MODE = Duty Cycle AND PWM1_CTRL.MODE = Frequency, then this is the integer value of the switching frequency divisor STOP_VAL[10].<br>If PWM1_CTRL.MODE = Duty Cycle, then this bit is unused.<br>If PWM1_CTRL.MODE = Static, this is unused. |

### 7.6.9.13 PWM1\_STOP\_VAL\_LSB Register (Offset = 410Fh) [Reset = 00h]

PWM1\_STOP\_VAL\_LSB is shown in [Figure 7-162](#) and described in [Table 7-205](#).

Return to the [Summary Table](#).

If configured to vary/ramp duty cycle, this is used for the duty cycle LSB. If configured to vary/ramp frequency, this is used for the stop/off-target integer divisor LSB of the frequency. This is not used if configured for static PWM

**Figure 7-162. PWM1\_STOP\_VAL\_LSB Register**

| 7        | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|---|---|---|---|---|---|---|
| SPV[7:0] |   |   |   |   |   |   |   |
| R/W-0h   |   |   |   |   |   |   |   |

**Table 7-205. PWM1\_STOP\_VAL\_LSB Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-----|----------|------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | SPV[7:0] | R/W  | 0h    | <p>Stop Value</p> <p>This is a multi-purpose register.</p> <p>PWM</p> <p>1_CTRL.MODE = Duty Cycle, then this is the duty cycle value SPV[7:0].</p> <p>If</p> <p>8-bit mode, then SPV/256 = duty cycle.</p> <p>If</p> <p>10-bit mode, then SPV/1024 = duty cycle.</p> <p>If PWM</p> <p>1_CTRL.MODE = Frequency, then this is the integer value of the switching frequency divisor STOP_VAL[7:0].</p> <p>If PWM</p> <p>1_CTRL.MODE = Static, this is unused.</p> |

#### 7.6.9.14 PWM1\_STOP\_SL\_MSB Register (Offset = 4110h) [Reset = 00h]

PWM1\_STOP\_SL\_MSB is shown in [Figure 7-163](#) and described in [Table 7-206](#).

Return to the [Summary Table](#).

Used to configure the slope scale of the stop ramp

**Figure 7-163. PWM1\_STOP\_SL\_MSB Register**

| 7    | 6 | 5 | 4            | 3 | 2 | 1 | 0 |
|------|---|---|--------------|---|---|---|---|
| RSVD |   |   | SP_SL[20:16] |   |   |   |   |
| R-0h |   |   | R/W-0h       |   |   |   |   |

**Table 7-206. PWM1\_STOP\_SL\_MSB Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                                                                                                                                                               |
|-----|--------------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD         | R    | 0h    |                                                                                                                                                                                                                           |
| 4-0 | SP_SL[20:16] | R/W  | 0h    | Stop Slope<br>The slope for the stop ramp.<br>The fields vary depending on the value of PWM1_CTRL.SLOPE_SCALE.<br>See <a href="#">Section 7.5.7.4</a> for more information<br>If PWM1_CTRL.MODE = Static, this is unused. |

### 7.6.9.15 PWM1\_STOP\_SL\_MID Register (Offset = 4111h) [Reset = 00h]

PWM1\_STOP\_SL\_MID is shown in [Figure 7-164](#) and described in [Table 7-207](#).

Return to the [Summary Table](#).

Used to configure the slope scale of the stop ramp

**Figure 7-164. PWM1\_STOP\_SL\_MID Register**

|             |   |   |   |   |   |   |   |
|-------------|---|---|---|---|---|---|---|
| 7           | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SP_SL[15:8] |   |   |   |   |   |   |   |
| R/W-0h      |   |   |   |   |   |   |   |

**Table 7-207. PWM1\_STOP\_SL\_MID Register Field Descriptions**

| Bit | Field       | Type | Reset | Description                                                                                                                                                                                                               |
|-----|-------------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | SP_SL[15:8] | R/W  | 0h    | Stop Slope<br>The slope for the stop ramp.<br>The fields vary depending on the value of PWM1_CTRL.SLOPE_SCALE.<br>See <a href="#">Section 7.5.7.4</a> for more information<br>If PWM1_CTRL.MODE = Static, this is unused. |

#### 7.6.9.16 PWM1\_STOP\_SL\_LSB Register (Offset = 4112h) [Reset = 00h]

PWM1\_STOP\_SL\_LSB is shown in [Figure 7-165](#) and described in [Table 7-208](#).

Return to the [Summary Table](#).

Used to configure the slope scale of the stop ramp

**Figure 7-165. PWM1\_STOP\_SL\_LSB Register**

|            |   |   |   |   |   |   |   |
|------------|---|---|---|---|---|---|---|
| 7          | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SP_SL[7:0] |   |   |   |   |   |   |   |
| R/W-0h     |   |   |   |   |   |   |   |

**Table 7-208. PWM1\_STOP\_SL\_LSB Register Field Descriptions**

| Bit | Field      | Type | Reset | Description                                                                                                                                                                                                               |
|-----|------------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | SP_SL[7:0] | R/W  | 0h    | Stop Slope<br>The slope for the stop ramp.<br>The fields vary depending on the value of PWM1_CTRL.SLOPE_SCALE.<br>See <a href="#">Section 7.5.7.4</a> for more information<br>If PWM1_CTRL.MODE = Static, this is unused. |

#### 7.6.9.17 PWM1\_START\_VAL\_FRAC\_F Register (Offset = 4113h) [Reset = 00h]

PWM1\_START\_VAL\_FRAC\_F is shown in [Figure 7-166](#) and described in [Table 7-209](#).

Return to the [Summary Table](#).

If configured to vary/ramp frequency, this is used for the start fractional divisor of the frequency. This is not used if configured to ramp duty cycle or for static PWM.

**Figure 7-166. PWM1\_START\_VAL\_FRAC\_F Register**

| 7    | 6        | 5 | 4 | 3 | 2 | 1 | 0 |
|------|----------|---|---|---|---|---|---|
| RSVD | SPV_FR_F |   |   |   |   |   |   |
| R-0h | R/W-0h   |   |   |   |   |   |   |

**Table 7-209. PWM1\_START\_VAL\_FRAC\_F Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                                                       |
|-----|----------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | RSVD     | R    | 0h    |                                                                                                                                                                                                                                   |
| 6-0 | SPV_FR_F | R/W  | 0h    | Start Value Fractional Value (Frequency Only)<br>Only used when PWM1_CTRL.MODE = Frequency.<br>Interpreted as the x/128 fractional portion of the frequency divisor.<br>If PWM1_CTRL.MODE = Duty Cycle or Static, this is unused. |

### 7.6.9.18 PWM1\_START\_VAL\_MSB Register (Offset = 4114h) [Reset = 00h]

PWM1\_START\_VAL\_MSB is shown in [Figure 7-167](#) and described in [Table 7-210](#).

Return to the [Summary Table](#).

If configured to vary/ramp duty cycle, this is used for the duty cycle MSB (only used if 10-bit duty cycle resolution is enabled). If configured to vary/ramp frequency, this is used for the start integer divisor MSB of the frequency. This is not used if configured for static PWM

**Figure 7-167. PWM1\_START\_VAL\_MSB Register**

| 7 | 6 | 5    | 4 | 3 | 2         | 1 | 0        |
|---|---|------|---|---|-----------|---|----------|
|   |   | RSVD |   |   | STV_F[10] |   | STV[9:8] |
|   |   | R-0h |   |   | R/W-0h    |   | R/W-0h   |

**Table 7-210. PWM1\_START\_VAL\_MSB Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-----|-----------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD      | R    | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 2   | STV_F[10] | R/W  | 0h    | Start Value (Frequency Only)<br>This is a multi-purpose register.<br>If PWM1_CTRL.MODE = Frequency , then this is the integer value of the switching frequency divisor START_VAL[10].<br>If PWM1_CTRL.MODE = Duty Cycle, then this bit is unused.<br>If PWM1_CTRL.MODE = Static, this is unused..                                                                                                                                                                |
| 1-0 | STV[9:8]  | R/W  | 0h    | Start Value<br>This is a multi-purpose register.<br>PWM<br>1_CTRL.MODE = Duty Cycle AND PWM<br>1_CTRL.<br>8B =<br>0 (<br>10-bit), then this is the duty cycle value STV[<br>9:<br>8], otherwise this is unused.<br>STV/<br>1024 = duty cycle, and in<br>8-bit mode is unused.<br>If PWM<br>1_CTRL.MODE = Frequency, then this is the integer value of the switching frequency divisor START_VAL[<br>9:<br>8].<br>If PWM<br>1_CTRL.MODE = Static, this is unused. |

### 7.6.9.19 PWM1\_START\_VAL\_LSB Register (Offset = 4115h) [Reset = 00h]

PWM1\_START\_VAL\_LSB is shown in [Figure 7-168](#) and described in [Table 7-211](#).

Return to the [Summary Table](#).

If configured to vary/ramp duty cycle, this is used for the duty cycle LSB. If configured to vary/ramp frequency, this is used for the start integer divisor LSB of the frequency. This is not used if configured for static PWM

**Figure 7-168. PWM1\_START\_VAL\_LSB Register**

| 7        | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|---|---|---|---|---|---|---|
| STV[7:0] |   |   |   |   |   |   |   |
| R/W-0h   |   |   |   |   |   |   |   |

**Table 7-211. PWM1\_START\_VAL\_LSB Register Field Descriptions**

| Bit | Field    | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-----|----------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | STV[7:0] | R/W  | 0h    | <p>Start Value</p> <p>This is a multi-purpose register.</p> <p>If PWM</p> <p>1_CTRL.MODE = Duty Cycle, then this is the duty cycle value STV[7:0].</p> <p>If</p> <p>8-bit mode, then STV/256 = duty cycle.</p> <p>If</p> <p>10-bit mode, then STV/1024 = duty cycle.</p> <p>If PWM</p> <p>1_CTRL.MODE = Frequency, then this is the integer value of the switching frequency divisor START_VAL[7:0].</p> <p>If PWM</p> <p>1_CTRL.MODE = Static, this is unused.</p> |

### 7.6.9.20 PWM1\_START\_SL\_MSB Register (Offset = 4116h) [Reset = 00h]

PWM1\_START\_SL\_MSB is shown in [Figure 7-169](#) and described in [Table 7-212](#).

Return to the [Summary Table](#).

Used to configure the slope scale of the start ramp

**Figure 7-169. PWM1\_START\_SL\_MSB Register**

| 7    | 6 | 5 | 4            | 3 | 2 | 1 | 0 |
|------|---|---|--------------|---|---|---|---|
| RSVD |   |   | ST_SL[20:16] |   |   |   |   |
| R-0h |   |   | R/W-0h       |   |   |   |   |

**Table 7-212. PWM1\_START\_SL\_MSB Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                                                                                                                                                                 |
|-----|--------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD         | R    | 0h    |                                                                                                                                                                                                                             |
| 4-0 | ST_SL[20:16] | R/W  | 0h    | Start Slope<br>The slope for the start ramp.<br>The fields vary depending on the value of PWM1_CTRL.SLOPE_SCALE.<br>See <a href="#">Section 7.5.7.4</a> for more information<br>If PWM1_CTRL.MODE = Static, this is unused. |

#### 7.6.9.21 PWM1\_START\_SL\_MID Register (Offset = 4117h) [Reset = 00h]

PWM1\_START\_SL\_MID is shown in [Figure 7-170](#) and described in [Table 7-213](#).

Return to the [Summary Table](#).

Used to configure the slope scale of the start ramp

**Figure 7-170. PWM1\_START\_SL\_MID Register**

| 7           | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-------------|---|---|---|---|---|---|---|
| ST_SL[15:8] |   |   |   |   |   |   |   |
| R/W-0h      |   |   |   |   |   |   |   |

**Table 7-213. PWM1\_START\_SL\_MID Register Field Descriptions**

| Bit | Field       | Type | Reset | Description                                                                                                                                                                                                                 |
|-----|-------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | ST_SL[15:8] | R/W  | 0h    | Start Slope<br>The slope for the start ramp.<br>The fields vary depending on the value of PWM1_CTRL.SLOPE_SCALE.<br>See <a href="#">Section 7.5.7.4</a> for more information<br>If PWM1_CTRL.MODE = Static, this is unused. |

#### 7.6.9.22 PWM1\_START\_SL\_LSB Register (Offset = 4118h) [Reset = 00h]

PWM1\_START\_SL\_LSB is shown in [Figure 7-171](#) and described in [Table 7-214](#).

Return to the [Summary Table](#).

Used to configure the slope scale of the start ramp

**Figure 7-171. PWM1\_START\_SL\_LSB Register**

|            |   |   |   |   |   |   |   |
|------------|---|---|---|---|---|---|---|
| 7          | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ST_SL[7:0] |   |   |   |   |   |   |   |
| R/W-0h     |   |   |   |   |   |   |   |

**Table 7-214. PWM1\_START\_SL\_LSB Register Field Descriptions**

| Bit | Field      | Type | Reset | Description                                                                                                                                                                                                                 |
|-----|------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | ST_SL[7:0] | R/W  | 0h    | Start Slope<br>The slope for the start ramp.<br>The fields vary depending on the value of PWM1_CTRL.SLOPE_SCALE.<br>See <a href="#">Section 7.5.7.4</a> for more information<br>If PWM1_CTRL.MODE = Static, this is unused. |

### 7.6.9.23 PWM1\_END\_VAL\_CONST\_FRAC\_F Register (Offset = 4119h) [Reset = 00h]

PWM1\_END\_VAL\_CONST\_FRAC\_F is shown in [Figure 7-172](#) and described in [Table 7-215](#).

Return to the [Summary Table](#).

If configured to vary frequency, this is the ending value (target value) fractional divisor for frequency. If configured to vary/ramp duty cycle, then this is the fractional divisor of the switching frequency. If configured as static PWM output, this register is used for the fractional portion of the frequency divisor.

**Figure 7-172. PWM1\_END\_VAL\_CONST\_FRAC\_F Register**

| 7    | 6         | 5 | 4 | 3 | 2 | 1 | 0 |
|------|-----------|---|---|---|---|---|---|
| RSVD | EVCV_FR_F |   |   |   |   |   |   |
| R-0h | R/W-0h    |   |   |   |   |   |   |

**Table 7-215. PWM1\_END\_VAL\_CONST\_FRAC\_F Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-----|-----------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7   | RSVD      | R    | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 6-0 | EVCV_FR_F | R/W  | 0h    | End Value Fractional (Frequency Mode) or Const Value Fractional (DC Mode)<br>End value fractional portion.<br>This is always used, but the field changes based upon the mode currently in use.<br>Interpreted as the x/128 fractional portion of the frequency divisor.<br>PWM1_CTRL.MODE = Duty Cycle, then this is the integer value of the switching frequency divisor CONST_FREQ[10].<br>If PWM1_CTRL.MODE = Frequency or Static, then this bit is unused |

#### 7.6.9.24 PWM1\_END\_VAL\_MSB Register (Offset = 411Ah) [Reset = 00h]

PWM1\_END\_VAL\_MSB is shown in [Figure 7-173](#) and described in [Table 7-216](#).

Return to the [Summary Table](#).

Settings for the varied PWM ending value (target value) if configured to ramp. If configured as static PWM output, this register is used for the integer MSB portion of the frequency divisor.

**Figure 7-173. PWM1\_END\_VAL\_MSB Register**

| 7    | 6 | 5 | 4 | 3         | 2 | 1            | 0 |
|------|---|---|---|-----------|---|--------------|---|
| RSVD |   |   |   | EV_FR[10] |   | END_VAL[9:8] |   |
| R-0h |   |   |   | R/W-0h    |   | R/W-0h       |   |

**Table 7-216. PWM1\_END\_VAL\_MSB Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----|--------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-3 | RSVD         | R    | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 2   | EV_FR[10]    | R/W  | 0h    | End Value (Frequency Only)<br>This is a multi-purpose register.<br>PWM1_CTRL.MODE = Duty Cycle or Static, then this is the integer value of the switching frequency value START_VAL[10].<br>If PWM1_CTRL.MODE = Frequency, then this bit is unused                                                                                                                                                                           |
| 1-0 | END_VAL[9:8] | R/W  | 0h    | End Value<br>This is a multi-purpose register.<br>PWM<br>1_CTRL.MODE = Duty Cycle or Static, then this is the integer value of the switching frequency divisor END_VAL[9:8].<br>If PWM<br>1_CTRL.MODE = Frequency AND PWM<br>1_CTRL.MODE = 8B = 0 (10-bit), then this is the duty cycle value END_VAL[9:8], otherwise this is unused.<br>If<br>10-bit mode, then END_VAL/1024 = duty cycle, but in 8-bit mode this is unused |

#### 7.6.9.25 PWM1\_END\_VAL\_LSB Register (Offset = 411Bh) [Reset = 00h]

PWM1\_END\_VAL\_LSB is shown in [Figure 7-174](#) and described in [Table 7-217](#).

Return to the [Summary Table](#).

Settings for the varied PWM ending value (target value) if configured to ramp. If configured as static PWM output, this register is used for the integer LSB portion of the frequency divisor.

**Figure 7-174. PWM1\_END\_VAL\_LSB Register**

| 7            | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|--------------|---|---|---|---|---|---|---|
| END_VAL[7:0] |   |   |   |   |   |   |   |
| R/W-0h       |   |   |   |   |   |   |   |

**Table 7-217. PWM1\_END\_VAL\_LSB Register Field Descriptions**

| Bit | Field        | Type | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                            |
|-----|--------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-0 | END_VAL[7:0] | R/W  | 0h    | <p>End Value</p> <p>This is a multi-purpose register.</p> <p>PWM</p> <p>1_CTRL.MODE = Duty Cycle or Static, then this is the integer value of the switching frequency divisor END_VAL[7:0].</p> <p>If PWM</p> <p>1_CTRL.MODE = Frequency, then this is the duty cycle value END_VAL[7:0].</p> <p>If</p> <p>8-bit mode, then END_VAL/256 = duty cycle.</p> <p>If</p> <p>10-bit mode, then END_VAL/1024 = duty cycle</p> |

#### 7.6.9.26 PWM1\_PULSE\_STOP\_RAMP[y] Register (Offset = 411Ch + formula) [Reset = 00000000h]

PWM1\_PULSE\_STOP\_RAMP[y] is shown in [Figure 7-175](#) and described in [Table 7-218](#).

Return to the [Summary Table](#).

If configured for automatic stop-ramp, this is the number of generated PWM pulses from the starting pulse that the module will automatically set the stop ramp command. The lowest address is corresponds to the MSB of the counter.

Offset = 411Ch + (y \* 4h); where y = 0h to 3h

**Figure 7-175. PWM1\_PULSE\_STOP\_RAMP[y] Register**

|                      |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
|----------------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| 31                   | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PWM0_PULSE_STOP_RAMP |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| R/W-0h               |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

**Table 7-218. PWM1\_PULSE\_STOP\_RAMP[y] Register Field Descriptions**

| Bit  | Field                | Type | Reset | Description                                                                                                                                                                                    |
|------|----------------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 31-0 | PWM0_PULSE_STOP_RAMP | R/W  | 0h    | Once the specified number of PWM pulses are generated, the stop-ramp is automatically started.<br>This is only used if the PWM1_ACTION.AUTO_STOP bit is set.<br>The lowest address is the MSB. |

**7.6.9.27 PWM1\_PULSE\_MAX[y] Register (Offset = 4120h + formula) [Reset = 00000000h]**

PWM1\_PULSE\_MAX[y] is shown in [Figure 7-176](#) and described in [Table 7-219](#).

Return to the [Summary Table](#).

If configured for the automatic stop-ramp, this is the maximum number of PWM pulses that will be allowed. If the number of PWM pulses hits this limit, then the output will immediately be stopped. The lowest address is corresponds to the MSB of the counter.

Offset = 4120h + (y \* 4h); where y = 0h to 3h

**Figure 7-176. PWM1\_PULSE\_MAX[y] Register**

|                |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
|----------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PWM0_PULSE_MAX |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |
| R/W-0h         |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |    |   |   |   |   |   |   |   |   |   |   |

**Table 7-219. PWM1\_PULSE\_MAX[y] Register Field Descriptions**

| Bit  | Field          | Type | Reset | Description                                                                                                                                                                        |
|------|----------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 31-0 | PWM0_PULSE_MAX | R/W  | 0h    | Once the specified number of PWM pulses are generated, the PWM output is disabled.<br>This is only used if the PWM1_ACTION.AUTO_STOP bit is set.<br>The lowest address is the MSB. |

### 7.6.9.28 PWM1\_ACTION Register (Offset = 4124h) [Reset = 00h]

PWM1\_ACTION is shown in [Figure 7-177](#) and described in [Table 7-220](#).

Return to the [Summary Table](#).

**Figure 7-177. PWM1\_ACTION Register**

| 7    | 6 | 5 | 4        | 3         | 2         | 1         | 0         |
|------|---|---|----------|-----------|-----------|-----------|-----------|
| RSVD |   |   | RESERVED | AUTO_STOP | UDS       | START     | STOP      |
| R-0h |   |   | R-0h     | R/W-0h    | RH/W1S-0h | RH/W1S-0h | RH/W1S-0h |

**Table 7-220. PWM1\_ACTION Register Field Descriptions**

| Bit | Field     | Type   | Reset | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-----|-----------|--------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD      | R      | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 4   | RESERVED  | R      | 0h    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 3   | AUTO_STOP | R/W    | 0h    | Use automatic stop-ramp<br>If enabled, once the pulse count hits the specified<br>0h = Do not use automatic-stop-ramp behavior<br>1h = Automatically begin the stop-ramp based on the pulse count registers                                                                                                                                                                                                                                                                |
| 2   | UDS       | RH/W1S | 0h    | Use Defined Start<br>Use the start point defined in the start value register.<br>Used to ensure the start point is known, or if you want the ramp to take place from the current value (mid ramp or existing end point).<br>If this is set when START is set, then the ramp will begin at the defined starting point.<br>Being set by itself will not start a ramp<br>0h = Use the existing current value as the starting point<br>1h = Start from the defined start point |
| 1   | START     | RH/W1S | 0h    | Start<br>Start the defined ramp profile based on configured settings.<br>If STOP is also set at same time, the START will be ignored.<br>Bit is cleared by hardware.                                                                                                                                                                                                                                                                                                       |
| 0   | STOP      | RH/W1S | 0h    | Stop<br>Will ramp to the specified stop point and then turn off.<br>If START and STOP are set together, STOP will occur.<br>Bit is cleared by hardware                                                                                                                                                                                                                                                                                                                     |

**7.6.9.29 PWM1\_IAS\_CTRL Register (Offset = 4130h) [Reset = 00h]**

PWM1\_IAS\_CTRL is shown in [Figure 7-178](#) and described in [Table 7-221](#).

Return to the [Summary Table](#).

**Figure 7-178. PWM1\_IAS\_CTRL Register**

| 7    | 6 | 5 | 4      | 3        | 2 | 1         | 0      |
|------|---|---|--------|----------|---|-----------|--------|
| RSVD |   |   | IN_POL | GPIO_SEL |   | STOP_MODE | AS_EN  |
| R-0h |   |   | R/W-0h | R/W-0h   |   | R/W-0h    | R/W-0h |

**Table 7-221. PWM1\_IAS\_CTRL Register Field Descriptions**

| Bit | Field     | Type | Reset | Description                                                                                                                                                                                                                                                                        |
|-----|-----------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 7-5 | RSVD      | R    | 0h    | Reserved                                                                                                                                                                                                                                                                           |
| 4   | IN_POL    | R/W  | 0h    | Input Polarity<br>Selects which level is used to trigger the stop behavior<br>0h = Active low input signal<br>1h = Active high input signal                                                                                                                                        |
| 3-2 | GPIO_SEL  | R/W  | 0h    | GPIO Select<br>Specifies which GPIO is used.<br>GPIO must be configured as a GPIO function instead of special function in order to work.<br>0h = GPIO0<br>1h = GPIO1<br>2h = GPIO7<br>3h = GPIO8                                                                                   |
| 1   | STOP_MODE | R/W  | 0h    | Stop Mode<br>When a GPIO triggered stop event occurs, select between immediately disabling PWM output, or requesting the stop ramp.<br>0h = GPIO triggered event requests a stop ramp<br>1h = GPIO triggered event immediately disables PWM output                                 |
| 0   | AS_EN     | R/W  | 0h    | Input Auto Stop Enable<br><br><b>Note</b><br><br>This does NOT require the AUTO_STOP bit to be set, and is a separate function to stop the output based on a GPIO input<br><br>0h = No auto stop is enabled from GPIO signal<br>1h = Enable GPIO triggered auto stop functionality |

## 8 Application and Implementation

### Note

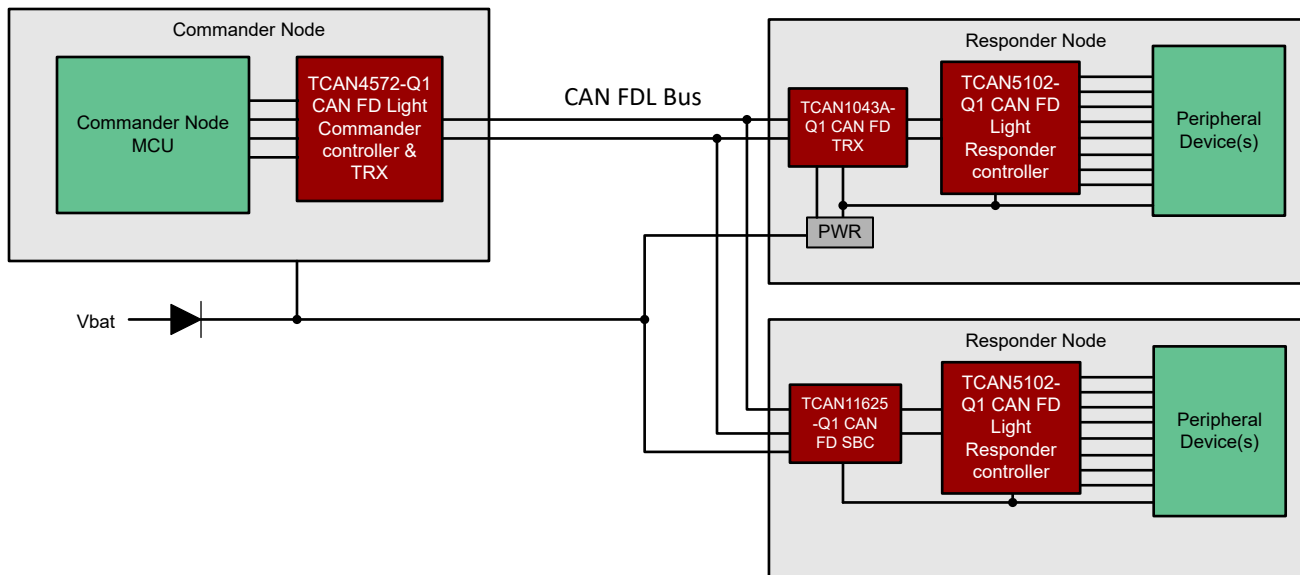
Information in the following applications sections is not part of the TI component specification, and TI does not warrant its accuracy or completeness. TI's customers are responsible for determining suitability of components for their purposes, as well as validating and testing their design implementation to confirm system functionality.

### 8.1 Application Information

The TCAN5102-Q1 is a Control Area Network (CAN) Flexible Data (FD) Light responder device. All control to the responder node is through the CAN bus from the commander node processor which eliminates the need for responder node processor and software. As such it is very flexible and can be used in many applications, much as end node motor control. Here TCAN5102 interfaces with a CAN FD transceiver such as the TCAN11625 or TCAN11623 and a stepper motor driver such as the DRV8889. Here all the control for the motor can come from the commander node via CANH and CANL, the TCAN5102-Q1 can understand these instructions and configuration, to operate the stepper motor driver. Here no processor software overhead is required for the remote control edge (RCE).

### 8.2 Typical Application

#### RCE CAN FD Light Stepper Motor Control Application



**Figure 8-1. Typical Commander-Responder Bus Architecture**

### 8.2.1 Design Requirements

**Table 8-1. Design Parameters**

| PARAMETER          | VALUE                                                                                                                                        |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| VDD                | 5V or 3V                                                                                                                                     |
| SPI COnfiguration  | nSCS, SCLK, SDO, SDI                                                                                                                         |
| GPIO Configuration | GPIO0 - fault detection<br>GPIO1 - motor disable<br>GPIO8 - motor direction<br>GPIO11 (PWM) - increment motor 1 step<br>GPIO9 - sleep enable |

### 8.2.2 Detailed Design Procedure

To implement the stepper motor driver RCE, the TCAN5102-Q1 should be configured appropriately for SPI mode and GPIO. The SPI and GPIO commands are translated directly from the CAN to the motor driver. [Section 7.6.2](#) overviews the register map to configure the SPI data rate, polarity and mode to the corresponding motor driver. The GPIO pins can also be configured via the corresponding registers. For example, GPIO11 needs to be configured as a PWM for the stepper motor example. First, GPIO11\_CFG should be set to 1h (special function PWM0). Then PWM0 can be configured depending on how the stepper motor needs to be controlled/stepped, register map is outlined in [Section 7.6.8](#).

(16)

### 8.3 Power Supply Recommendations

The TCAN5102-Q1 is designed to operate off of a single 5V or 3V supply. Power likely comes from Vbatt which can be stepped down with a CAN FD SBC such as the TCAN11625 or TCAN11623 which supplies 100mA or 70mA to VIO/VCCOUT which is more than enough for the TCAN5102 (<2mA). This is outlined in [Section 8.2](#).

Typically a 1μF and 100nF capacitor should be placed as close as possible to VDD on the TCAN5102-Q1. And a 1μF capacitor is required on DIGFLTR, to provide filtering for the internal digital core regulator. These help to reduce supply voltage ripple present on the outputs of the switched-mode power supplies and also helps to compensate for the resistance and inductance of the PCB power planes and traces.

### 8.4 Layout

#### 8.4.1 Layout Guidelines

Use at least two vias for supply and ground connections of bypass capacitors and protection devices to minimize trace and via inductance. Bypass and bulk capacitors should be placed as close as possible to the supply terminals of transceiver.

Route the high-speed traces using a minimum amount of vias and corners. This will reduce the amount of impedance changes. When it becomes necessary to make the traces turn 90°, use two 45° turns or an arc instead of making a single 90° turn. Do not route high-speed traces near crystals, oscillators, external clock signals, switching regulators, mounting holes or magnetic devices. Avoid stubs on the signal lines.

## 8.4.2 Layout Example

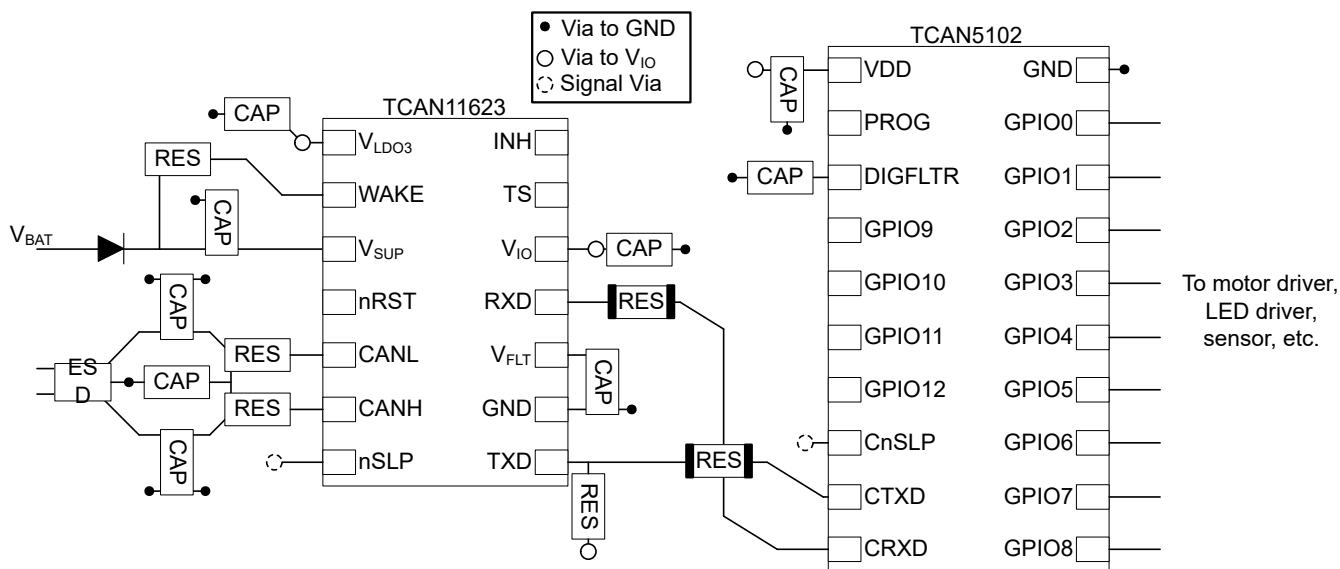


Figure 8-2. Layout Example

## 9 Device and Documentation Support

TI offers an extensive line of development tools. Tools and software to evaluate the performance of the device, generate code, and develop solutions are listed below.

### 9.1 Documentation Support

#### 9.1.1 Related Documentation

### 9.2 Receiving Notification of Documentation Updates

To receive notification of documentation updates, navigate to the device product folder on [ti.com](http://ti.com). Click on *Notifications* to register and receive a weekly digest of any product information that has changed. For change details, review the revision history included in any revised document.

### 9.3 Support Resources

TI E2E™ [support forums](#) are an engineer's go-to source for fast, verified answers and design help — straight from the experts. Search existing answers or ask your own question to get the quick design help you need.

Linked content is provided "AS IS" by the respective contributors. They do not constitute TI specifications and do not necessarily reflect TI's views; see TI's [Terms of Use](#).

### 9.4 Trademarks

TI E2E™ is a trademark of Texas Instruments.

All trademarks are the property of their respective owners.

### 9.5 Electrostatic Discharge Caution



This integrated circuit can be damaged by ESD. Texas Instruments recommends that all integrated circuits be handled with appropriate precautions. Failure to observe proper handling and installation procedures can cause damage.

ESD damage can range from subtle performance degradation to complete device failure. Precision integrated circuits may be more susceptible to damage because very small parametric changes could cause the device not to meet its published specifications.

### 9.6 Glossary

[TI Glossary](#) This glossary lists and explains terms, acronyms, and definitions.

## 10 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

### Changes from Revision \* (September 2025) to Revision A (October 2025)

Page

## 11 Mechanical, Packaging, and Orderable Information

The following pages include mechanical, packaging, and orderable information. This information is the most current data available for the designated devices. This data is subject to change without notice and revision of this document. For browser-based versions of this data sheet, refer to the left-hand navigation.

## 11.1 Mechanical Data

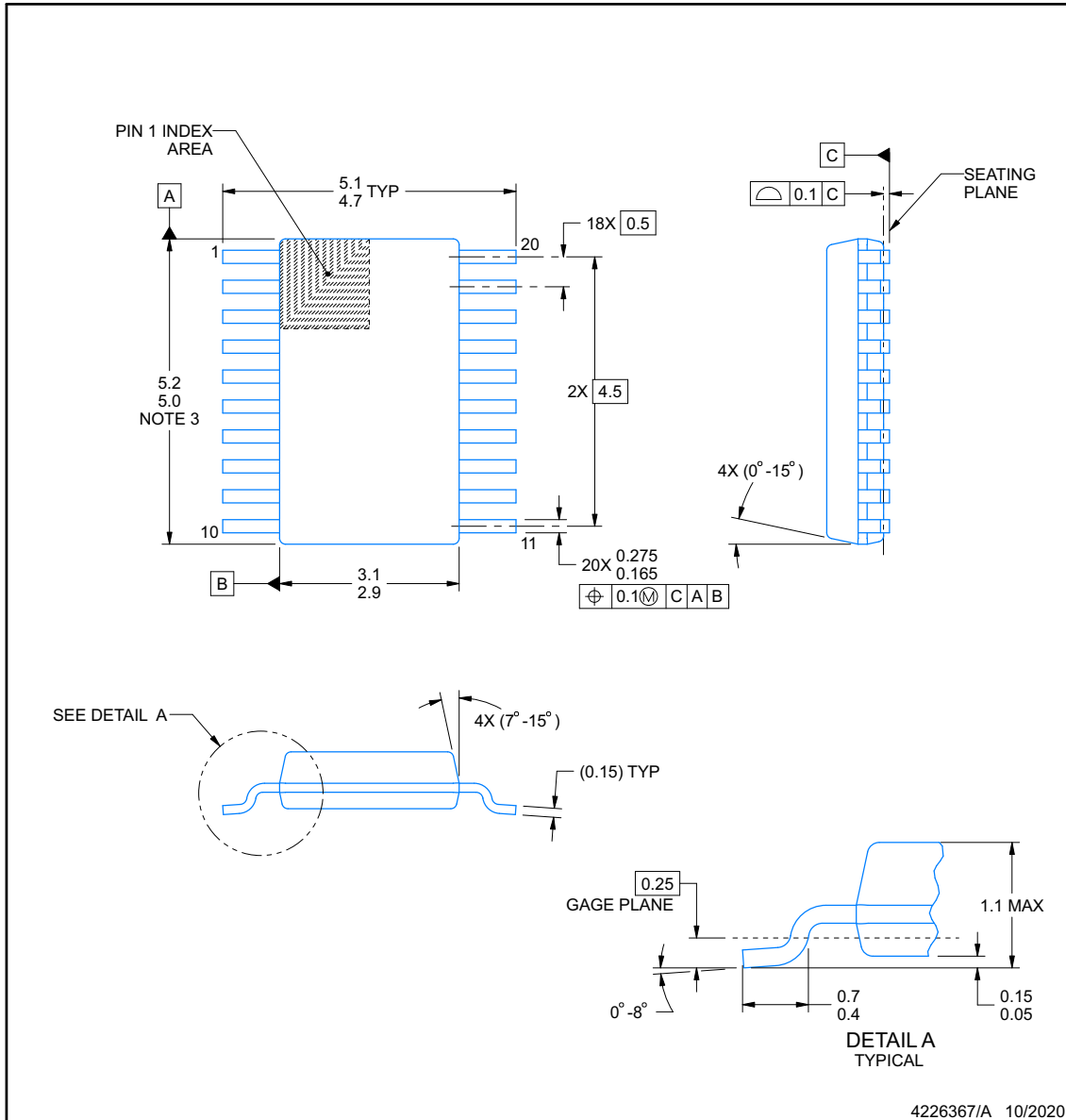
**DGS0020A**



### PACKAGE OUTLINE

**VSSOP - 1.1 mm max height**

SMALL OUTLINE PACKAGE



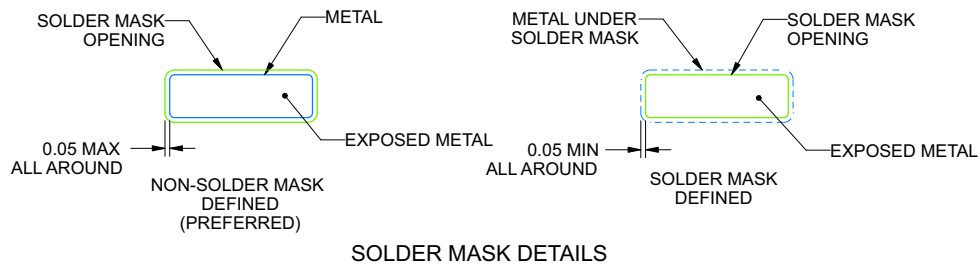
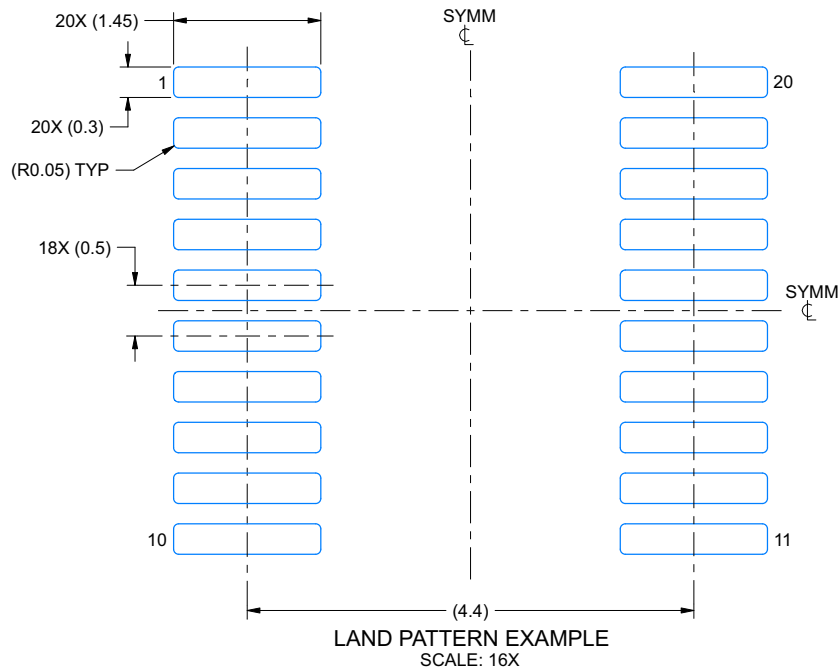
#### NOTES:

PowerPAD is a trademark of Texas Instruments.

1. All linear dimensions are in millimeters. Any dimensions in parenthesis are for reference only. Dimensioning and tolerancing per ASME Y14.5M.
2. This drawing is subject to change without notice.
3. This dimension does not include mold flash, protrusions, or gate burrs. Mold flash, protrusions, or gate burrs shall not exceed 0.15 mm per side.
4. No JEDEC registration as of September 2020.
5. Features may differ or may not be present.

**EXAMPLE BOARD LAYOUT****DGS0020A****VSSOP - 1.1 mm max height**

SMALL OUTLINE PACKAGE



4226367/A 10/2020

NOTES: (continued)

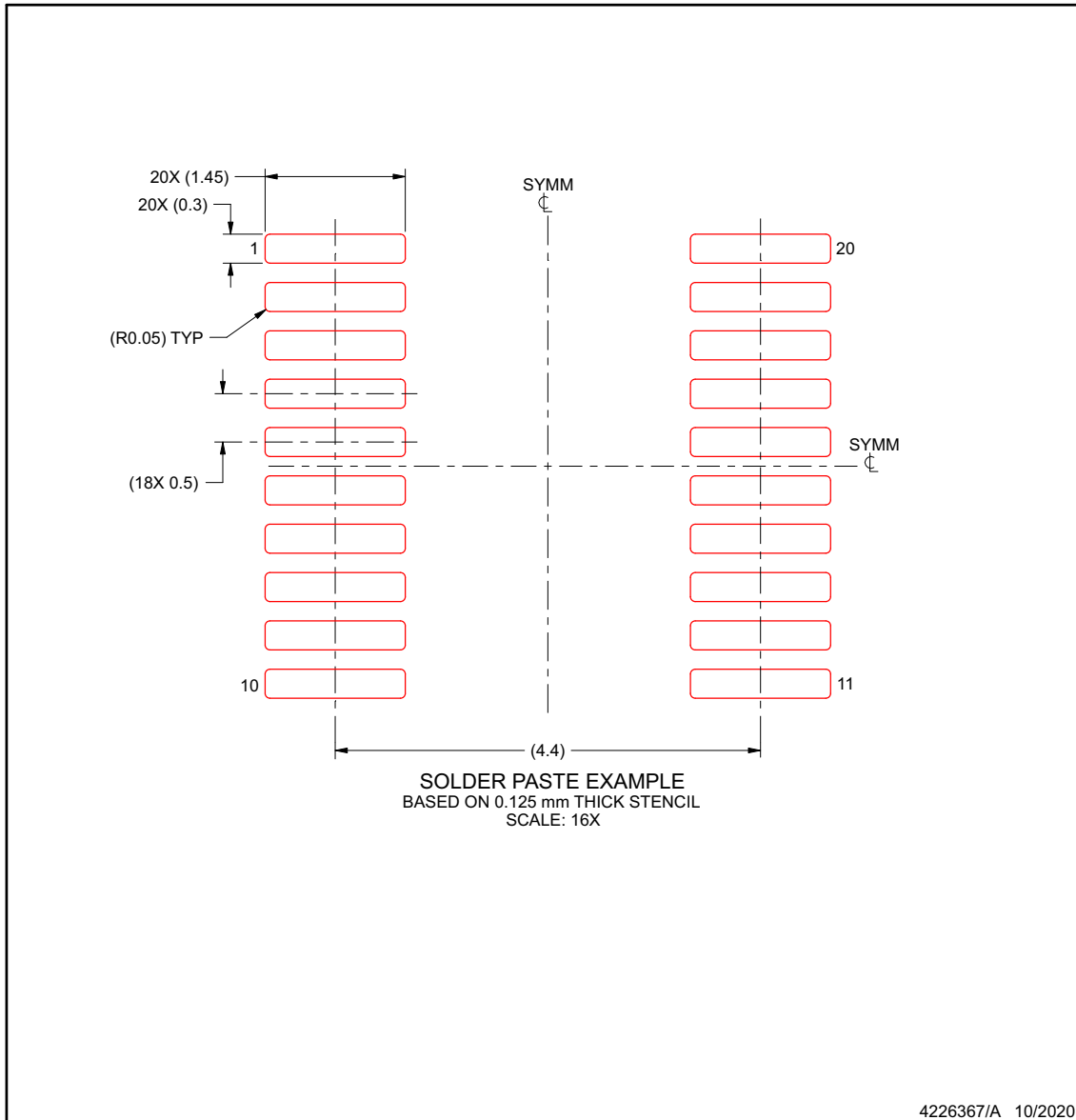
6. Publication IPC-7351 may have alternate designs.
7. Solder mask tolerances between and around signal pads can vary based on board fabrication site.
8. This package is designed to be soldered to a thermal pad on the board. For more information, see Texas Instruments literature numbers SLMA002 ([www.ti.com/lit/slma002](http://www.ti.com/lit/slma002)) and SLMA004 ([www.ti.com/lit/slma004](http://www.ti.com/lit/slma004)).
9. Size of metal pad may vary due to creepage requirement.
10. Vias are optional depending on application, refer to device data sheet. It is recommended that vias under paste be filled, plugged or tented.

## EXAMPLE STENCIL DESIGN

**DGS0020A**

**VSSOP - 1.1 mm max height**

SMALL OUTLINE PACKAGE



NOTES: (continued)

11. Laser cutting apertures with trapezoidal walls and rounded corners may offer better paste release. IPC-7525 may have alternate design recommendations.
12. Board assembly site may have different recommendations for stencil design.

PACKAGING INFORMATION

| Orderable part number | Status<br>(1) | Material type<br>(2) | Package   Pins    | Package qty   Carrier | RoHS<br>(3) | Lead finish/<br>Ball material<br>(4) | MSL rating/<br>Peak reflow<br>(5) | Op temp (°C) | Part marking<br>(6) |
|-----------------------|---------------|----------------------|-------------------|-----------------------|-------------|--------------------------------------|-----------------------------------|--------------|---------------------|
| PTCAN5102DGQRQ1       | Active        | Preproduction        | HVSSOP (DGQ)   20 | 2500   LARGE T&R      | -           | Call TI                              | Call TI                           | -40 to 125   |                     |

- (1) **Status:** For more details on status, see our [product life cycle](#).
- (2) **Material type:** When designated, preproduction parts are prototypes/experimental devices, and are not yet approved or released for full production. Testing and final process, including without limitation quality assurance, reliability performance testing, and/or process qualification, may not yet be complete, and this item is subject to further changes or possible discontinuation. If available for ordering, purchases will be subject to an additional waiver at checkout, and are intended for early internal evaluation purposes only. These items are sold without warranties of any kind.
- (3) **RoHS values:** Yes, No, RoHS Exempt. See the [TI RoHS Statement](#) for additional information and value definition.
- (4) **Lead finish/Ball material:** Parts may have multiple material finish options. Finish options are separated by a vertical ruled line. Lead finish/Ball material values may wrap to two lines if the finish value exceeds the maximum column width.
- (5) **MSL rating/Peak reflow:** The moisture sensitivity level ratings and peak solder (reflow) temperatures. In the event that a part has multiple moisture sensitivity ratings, only the lowest level per JEDEC standards is shown. Refer to the shipping label for the actual reflow temperature that will be used to mount the part to the printed circuit board.
- (6) **Part marking:** There may be an additional marking, which relates to the logo, the lot trace code information, or the environmental category of the part.

Multiple part markings will be inside parentheses. Only one part marking contained in parentheses and separated by a "-" will appear on a part. If a line is indented then it is a continuation of the previous line and the two combined represent the entire part marking for that device.

**Important Information and Disclaimer:**The information provided on this page represents TI's knowledge and belief as of the date that it is provided. TI bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. TI has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. TI and TI suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

In no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI to Customer on an annual basis.

## IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES "AS IS" AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you fully indemnify TI and its representatives against any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to [TI's Terms of Sale](#), [TI's General Quality Guidelines](#), or other applicable terms available either on [ti.com](#) or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products. Unless TI explicitly designates a product as custom or customer-specified, TI products are standard, catalog, general purpose devices.

TI objects to and rejects any additional or different terms you may propose.

Copyright © 2025, Texas Instruments Incorporated

Last updated 10/2025