Application Note

Wake Up Methods for CAN SBCs

Parker Dodson

ABSTRACT

System Basis Chips (SBCs) allow designers the opportunity to integrate power management, protection features, and CAN/LIN communication into one single package unified by a single control system. These devices also have the benefit of being low-power with many being able to achieve <100uA of current during sleep mode, which is critical for battery applications. However, the SBC cannot always be in sleep mode and must switch between modes to complete the designed tasks. By default, most SBCs cannot communicate through a SPI bus during sleep mode, so the question is asked: *how does the SBC wake up after it is asleep?* This is where the WAKE functionality of SBCs is used and this document gives an overview of commonly used wake methods for various SBCs.

Table of Contents

Trademarks

1 Introduction

System Basis Chips (SBCs) are designs for designers to use in designs due to the high number of features that are integrated into one package as well as the ability to enter into a low-power sleep mode. SBCs generally turn off bus communication during sleep mode to save power. This leads to a critical question: *How does the SBC "know" when to wake up?*. The answer to that question is simply through device wake up routines. This document discusses the following: the sleep mode differences between the SBC and integrated transceiver, how to wake the device if the communication bus is turned on during sleep, and local wake up through WAKE pins. Other topics discussed are digital wake ups from the MCU, cyclic wake up, cyclic sensing wake, and CAN BWRR, with a note on partial networking and selective wake.

2 Sleep Mode of the Transceiver vs. Sleep Mode of the SBC

Before discussing the various wake up methods, a few concepts must be discussed. The transceiver integrated into an SBC have some independence from the rest of the SBC, meaning that the mode of the transceiver does not reflect the mode of the entire SBC. However, note that the SBCs current state determines what modes are available. This means wake can have two different meanings: waking the SBC or waking the transceiver. In general, for the transceiver to be fully functioned and communicating the SBC must be in *normal mode*; however, the SBC being in normal mode does not verify that the transceiver is active. The transceiver can be in a wake-capable state. The main result is that a wake signal can come to the SBC while the SBC is awake but be meant for the transceiver specifically. See a device specific data sheet for more information on the transceiver and SBC modes.

Wake Up With SPI Communication Active

The simplest wake-up is using a SPI command to switch transceiver modes and SBC modes when the application requires the device to be awake. In general, the SPI bus is powered by a regulator that is internal to the SBC, which by default is going to be shut off during sleep mode to allow lower current consumption. However, many mid-range SBCs such as TCAN24xx-Q1 and TCAN28xx-Q1 have an option to turn on the SPI regulator (VCC1 in both devices) while the SBC is in sleep mode. This means that the SBC can be in a sleep state. If VCC1 is active that means the SPI bus is also active and the end user can send a SPI command to switch the SBC into different modes such as standby or normal modes. This allows the SBC to be woken up by the host controller through a simple SPI command. This does come at the cost of increased current usage during sleep due to the internal regulator being on and active.

That covers a simple wake up method for an SBC with the internal regulator turned on, but what about the transceiver? Since the transceiver state can be changed through a SPI command, a simple SPI command can be used to wake the transceiver as well. Check the device data sheet to verify if a specific SBC mode allows the end user to change the transceiver mode as well. If this is allowed and the SPI bus is active – a simple SPI command, such as an SBC mode switch, can be used as the only thing that changes is the register address and the data being written.

This largely covers the trivial forms of wake-up, but in many applications the SBC does not have the main regulator active so other wake methods must be considered.

3 Local Wake Up (LWU)

Now with a basic understanding of trivial wake methods, it is important to understand wake up methods for when the primary regulator of the SBC (typically denoted VCC1) is off. Many SBC designers have this requirement in the system and is why many SBCs includes a discrete WAKE pin in which the primary function of the pin is to act as a local wake-up input that can be activated during the sleep mode of the SBC with or without primary power active. Many SBCs contain more than one WAKE pin; the TCAN28XX-Q1 family of devices includes three discrete WAKE inputs while the TCAN24xx-Q1 family includes four discrete WAKE inputs. This means that multiple wake up sources can be present in a design that forces the SBC out of sleep mode and into an operational state through these WAKE pins.

A wake signal input on the WAKE pins is referred to as a local wake up (LWU) because the source of the wake up is from a signal that is local to the SBC (within the same subsystem). While there can be various types of a wake-up signal, TI SBCs contain the following wake-up schemes: bidirectional edge detection, rising edge detection, falling edge detection, and a pulse detection. All edge detection works relatively similarly. If the wake pin crosses the appropriate threshold and the signal is held for the minimum amount of time (t_wake) a wake-up condition is sent to the SBC control block. So for a rising edge, that means the WAKE pin voltage goes from a low state to a high state and is held for at least t_wake before a wake-up signal is generated internally to the SBC. Falling edge is just the reverse of the rising edge, for example if the WAKE pin voltage goes from a high state to a low state and is held for at least t_wake before the wake-up signal is generated internally to the SBC. Bidirectional edge detection generates a wake-up signal for any high to low or low to high transition on the WAKE pin that meets the minimum timing requirements. An example of this can be shown by looking at the rising edge and falling edge local wake up signals from TI SBC TCAN2847-Q1 - however it applies to multiple SBC devices from TI.

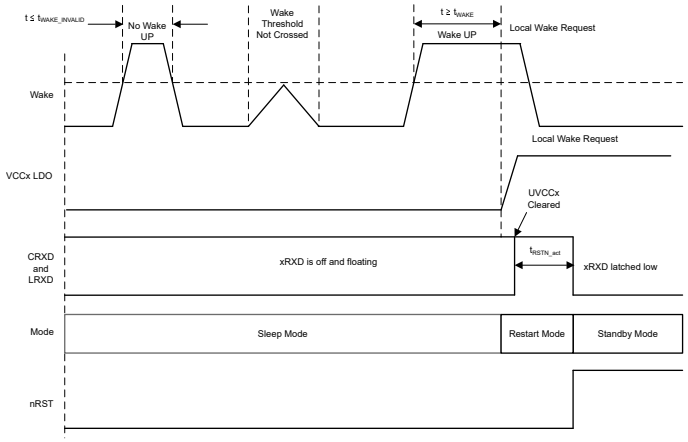

Figure 3-1. Local Wake Up: Rising Edge Shown (TCAN2847-Q1)

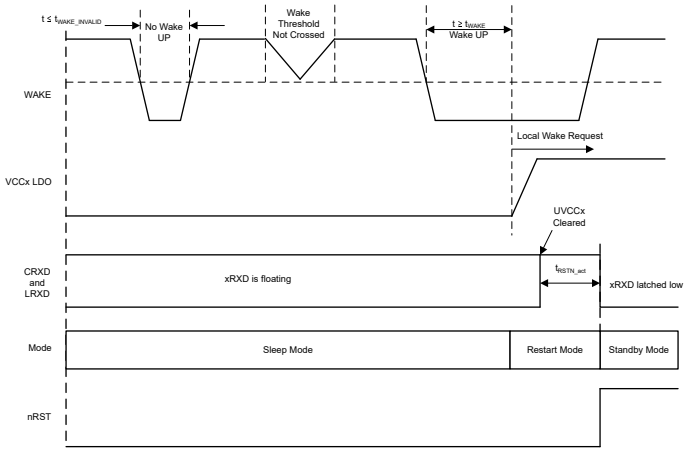

Figure 3-2. Local Wake Up: Falling Edge Shown (TCAN2847-Q1)

Beyond edge detection, there is also pulse detection which is more involved than standard edge detection as there are now a few more specifications to consider. The standard threshold levels are the same for both edge and pulse detection; these are generally configurable within the SBC. See a specific SBC data sheet for more information on the specific SBC device thresholds. However, the directionality of the pulse must also be specified, either as a low to high to low pulse or a high to low to high pulse so the polarity of the pulse matters if pulse detection wake up is used. There are also timing considerations with pulse detection. There are three primary timing specifications that are used in pulse detection and the specifications are from shortest to longest t_wk_width_invalid, t_wk_width_min, and t_wk_width_max. These time ranges are generally configurable within the SBC. For a valid pulse to be detected the pulse must be >= t_wk_width_min and <= t_wk_width_max; if the pulse is < t_wk_width_min and > t_wk_width_invalid a wake up condition can be detected. This is observed when looking to the behavior of the TCAN2847-Q1 and similar devices that allow for pulse-based wake on the WAKE pins.

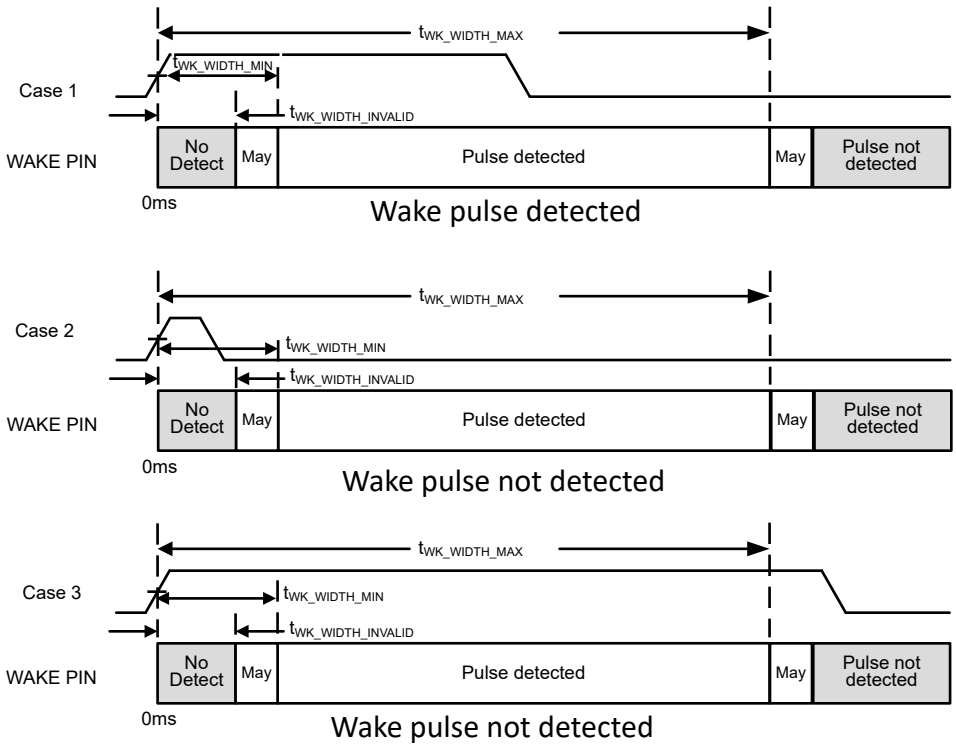

Figure 3-3. Local Wake Up: Pulse Detection (TCAN2847-Q1)

Note that these WAKE pins are high voltage tolerant and usually can withstand higher voltages than the supply pin of the SBC. be sure to check a specific device data sheet. While the above definitions define the most common use cases using the WAKE pins. There are a few of the uses that are further explained later in this application note.

The WAKE pins are generally simple to use. How does a designer use them in a production system? For this, look at a real-world example of an SBC that is positioned in a car door that must be woken up every time the car door opens or closes. Assume that a Hall-effect sensor is used to determine if the door is shut. For example, the sensor detects when the car door is closed and current does not flow through the sensor when the car door is shut showing a binary signal that states the condition of the door. For simplicity, assume the Hall effect sensor only has three pins (VCC, GND, and OUT) where OUT indicates if the door is closed or open. This is common of devices such as the TMAG5131-Q1 and can be used as the SBC wake-up source.

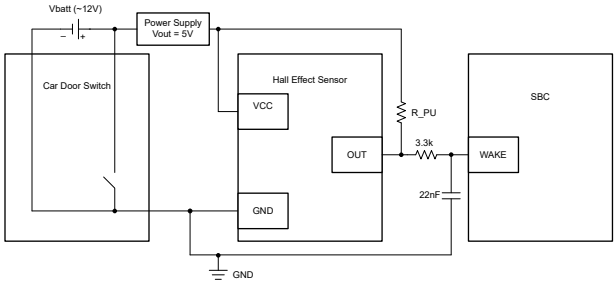

Figure 3-4. Hall Effect Sensor as Local Wake Source for SBC Diagram

In this simplified example the car door has a switch that conducts current when the door is closed and when the door is open, the circuit is an open circuit and not conduct current. This state of current and no current can be detected through the Hall effect sensor. The sensor output is open drain in the diagram but it does not have to be. This depends on the Hall effect sensor selected. If it is assumed that no current means a low output on the OUT pin and a detected current is a high output on the OUT pin, then what wake state must be used can be determined. Since this example is has the SBC wake up every time the door changes state and each door state is represented by either a high or a low logic level, the WAKE pin must be configured as bidirectional edge detection. Another important concept is the wake level configuration. Since this specific example uses a 5V Hall-effect sensor that means the maximum output is also going to be limited to around 5V. So the rising threshold must be lower than the maximum output voltage of the sensor. This is not a problem for an SBC if the proper wake configuration is implemented. The previous example is not the only way to use the SBC WAKE pin, but merely a common use case example of how a designer can implement wake into a system.

4 Digital Wake Up

Another type of wake available in some SBCs including the TCAN28xx-Q1 and TCAN24xx-Q1 family of mid-range SBC devices is a digital wake up through the software debug (SW) pin. The SW pin on these devices is primarily an input that can be used to prevent the SBC from taking action due to a missed watchdog. However, this pin also has an alternate function as a digital wake input during sleep or fail-safe mode.

The digital wake up function differs depending on the SBC mode. In both cases, either an MCU or wake-capable CAN/LIN transceiver changes the state on the SW pin (either high to low or low to high transitions) to trigger a wake condition for the SBC. For sleep mode, if digital wake is enabled, a state change on the SW pin is enough to trigger a wake on the SBC

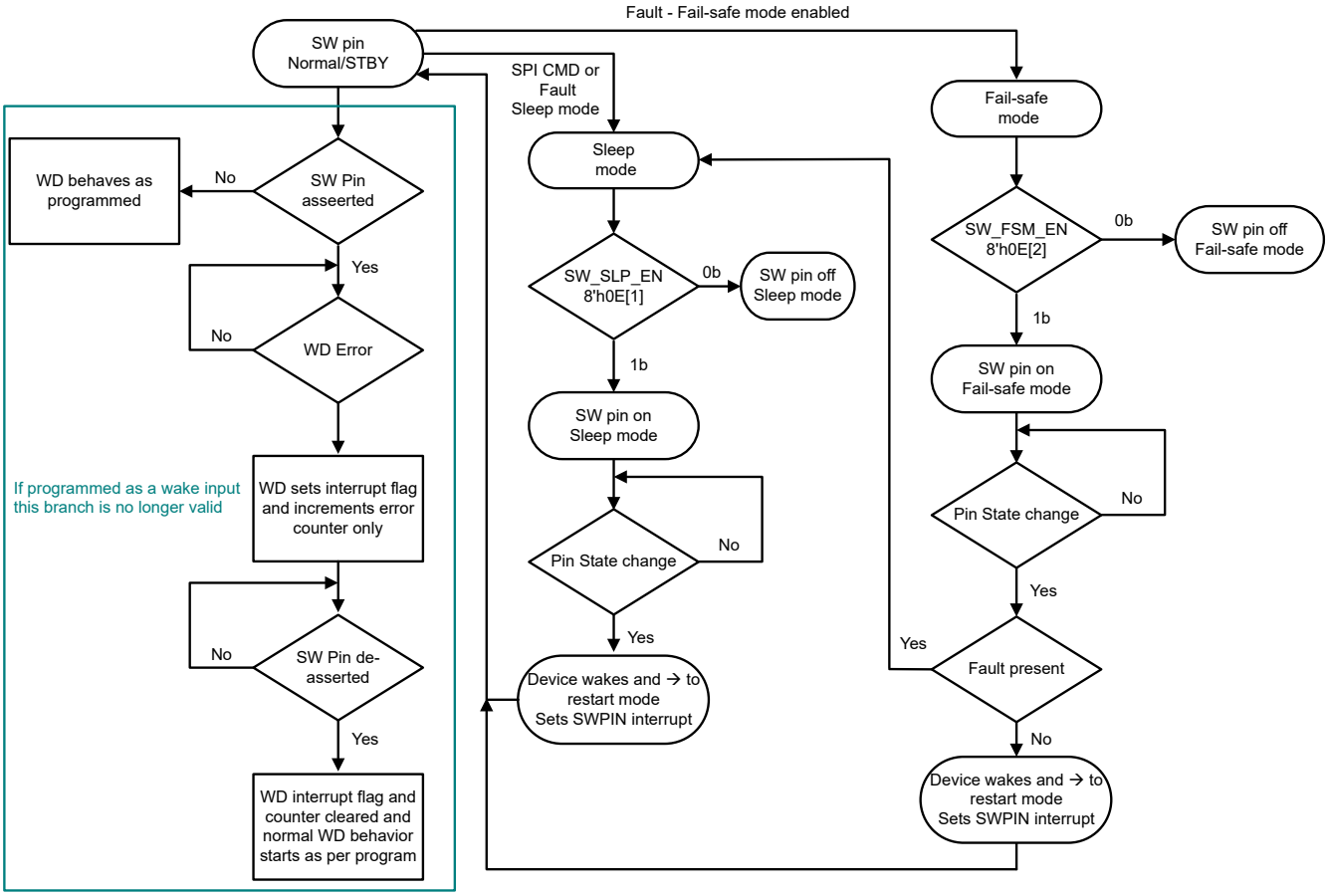

Figure 4-1. SW Pin State Diagram - Including Digital Wake

5 Cyclic Wake Up

Not all wake methods require external stimuli. Some SBC devices integrate an internal timer that can be used to wake up the SBC periodically. This idea is called cyclic wake-up: configure the integrated timers to wake up the SBC on regular repeating intervals. While timer specifics can vary across SBC device, devices in the TCAN28xx-Q1 or TCAN24xx-Q1 mid-range SBC families utilize a 10-bit timer. To understand how to utilize cyclic wake-up in a system one needs to understand how the operation varies between different SBC modes, how the wake methodology responds depending on device mode, as well as the potential options for configuration, and finally a brief note on why a cyclic wake-up scheme can be employed in a system.

In general, most TI SBCs have a few different modes. These can be grouped as either transitional modes which are *in-between* states that cannot be directly switched to; these include initialization state at device power up and the restart state that the device must transition to enter standby. Beyond the transitory states there are generally also 3 *operational* modes, which are generally standby, normal, and sleep mode. In standby mode the device is on and functional but communication is not occurring. This is the *wait/idle* mode that the device enters after power up. When the SBC enters standby mode a long window watchdog timer starts which must be triggered correctly to stay in standby mode. Otherwise, the device restarts and potentially enters a fail-safe or sleep mode. Normal mode is where the device is fully functional including bus communication. Sleep mode is a low power mode where most of the SBC is shut down but can be woken up if there is bus communication or a system function the device must complete. Finally, some SBCs also include a fail-safe mode which is similar to sleep mode except that there are more restrictions about how the device returns to standby mode, including removing any faults that caused the device to move into fail-safe mode. These modes are important to understand because the cyclic wake-up functionality varies depending on the SBC mode. To help illustrate this the TCAN2847-Q1 mid-range SBC device is used to highlight common ways that the cyclic timer functions depending on the SBCs operational mode.

Cyclic wake is available for use in the normal, standby, sleep, and fail-safe modes of the SBC device, but the function varies slightly depending on SBC mode. In normal and standby modes cyclic wake functions the same. After selecting either integrated timer1 or timer2 and the timer *on-time* is set, the timer starts at the beginning of the ON timer1/2 programmed by the interrupt pin (nINT pin on the TCAN2847-Q1) pulls low for the duration of the programmed on-time. The SBC ignores the first on-time period but every subsequent period uses the interrupt pin to alert the host processor that there has been an interrupt flag generated. In general, using cyclic wake up in this way is not as common as using cyclic wake up in sleep, but this is still possible. In sleep mode the general set up and configuration is the same as standby and normal modes – but the key difference is how the device reacts to the on-time period. When the timer enters the *on-time* period the SBC wakes up, transitions to restart mode, when VCC1 is available, this generates an interrupt flag to acknowledge the wake up, and then transition to standby mode. Upon entrance into standby mode, the long window watchdog timer starts and the host processor must correctly trigger the watchdog timer or else the device returns to sleep mode and waits for the next *on-time* period of for the timer1/2.

Cyclic wake can also be used to wake a device from a fail-safe mode; however, the configuration options are typically more limited. Fail-safe mode is like sleep mode except that the device cannot exit the mode unless the fault that caused fail-safe mode is cleared or the system is power cycled. When looking at the TCAN28xx-Q1 mid-range family of SBC devices, cyclic wake up is available in fail-safe mode but the on-time period options are restricted to 500ms, 1s, or 2s. When in fail-safe mode and the cyclic wake up timer enters the ON period, the device checks to see if the fault has cleared; if not, the process repeats until the Sleep Wake Error (SWE) timer expires and the device transitions to sleep mode.

So why does cyclic wake used in a system where there are multiple other wake methods? The answer varies with use case, but one of the most common implementations is to use cyclic wake in sleep mode. This gives the designer a way to check system status periodically while in a low power or sleep mode without requiring an event to take place on the local WAKE pins. It also gives a controlled way to exit fail-safe mode by essentially polling to check if the fault is present, still without direct intervention from the host MCU or controller to poll. While in standby and normal modes the cyclic wake can be used to alert the host MCU or processor of time passing, which the host MCU or processor could use to modify the configuration of the SBC device while in standby and normal modes – a modification could be simply turning on the CAN transceiver to enable bus communication.

6 Extending the Timer for Cyclic Wake Ups with External Components

Cyclic wake up has many advantages when used in a system and there are clear reasons why a user wants to use cyclic wake up. Currently, implementations of the cyclic timer on TI SBCs have a maximum timer period of 2s. In many use cases the maximum period of 2s is sufficient. However, imagine a system that is in a low power *sleep* mode where the main power is derived from a battery. Current consumption is critical in these types of applications and every time the device is woken up by the cyclic timer there is an increase in supply current due to the mode transition into a higher power operational mode. With current consumption in mind, assume the system is in a low power mode for hours at a time. If power usage spikes every 2 seconds a user can see a negative impact on battery power over time. Generally, the alternative choice in these designs is to increase the wake timer period beyond 2 seconds. Having a longer timer period reduces the overall current consumption by reducing the polling of the device in sleep mode To increase the wake timer period for TCAN24xx-Q1 or TCAN28xx-Q1 or any TI SBC device that has limited options for timer period configurations but also has local wake up pins available, an external design can be used.

At first glance, the design to this system requirement seems simple: the internal SBC timer is not long enough, so an external timer must be used. The external timer can have the output connected to a local SBC WAKE pin to achieve the goal of a longer period for the wake timer. In this type of application, the SBC is in sleep mode and all the integrated regulators to the SBC are off, so the external timer cannot be powered by the SBC during this time. This means that the timer must be powered off battery, which is typically 12V in automotive systems, while the SBC can withstand up to 28V. Select an external timer with similar considerations. Based on the input voltage requirement, a 555 timer device such as the TLC3555-Q1 can be a fit.

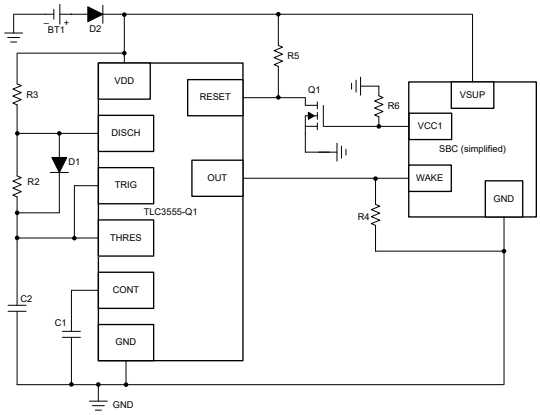

Figure 6-1. Extended Cyclic Wake Timer Using 555 Timer

This design is based off a classic 555 timer in an a stable configuration that can be tuned using simple RC components. When the SBC is awake the VCC1 output turns on the NMOS and put the timer into reset (forcing output low); so during normal operation the 555 timer is in reset. When the SBC goes to sleep,VCC1 is off and the 555 timer begins to send a pulse on the OUT pin based on the RC configuration of the circuit. This output pulse from the 555 timer can exceed the 2s max on the internal SBC timer. Switch the wake pin to only look for either low – high – low pulse or low to high transitions. By default, many SBCs are edge detection.

There are two design considerations when taking this approach. The first being that the TLC3555-Q1 can only withstand up to 20V on the supply input before the device can be damaged. In 12V automotive systems this is generally not enough for common transients as to why TI's SBCs are rated to 28V. The second reason related to low power design. A common requirement in many automotive applications is that the sleep current is less than 100uA. The TLC3555-Q1 sleep current at 12V input is going to be between 240uA to 310uA worst case This leaves no room in the 100uA limit, if the 100uA limit is a system requirement. If 100uA sleep current is required, there is a way around this issue.

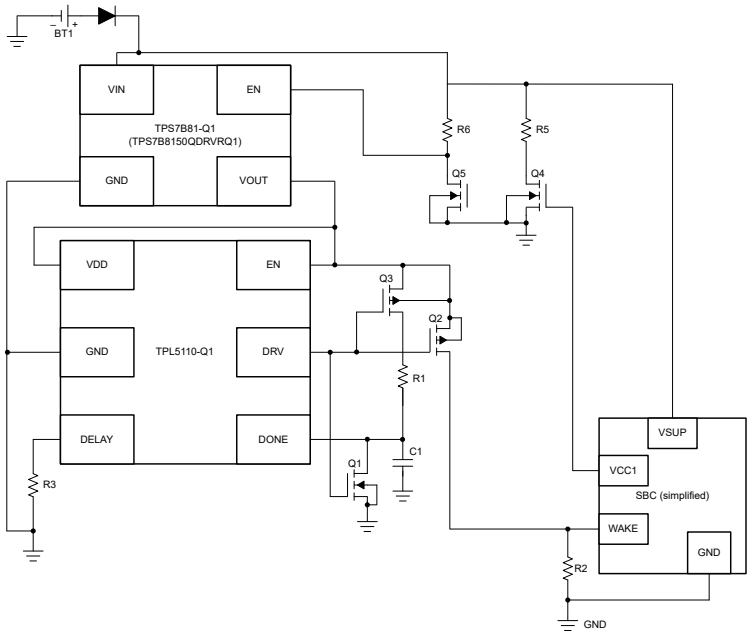

Figure 6-2. Low Power Extended Cyclic Wake Timer Implementation on TI SBCs

The adjusted design is more complex, but the basic idea is similar to the previous idea. When the SBC goes into sleep mode, the timer starts sending pulses to the wake pin based on the timers RC configuration. To achieve lower power, a lower input voltage timer must be used such as the TPL5110-Q1. Since this is a 5V timer the 12V nominal system input needs to be regulated to 5V – to generate a 5V rail an LDO (the TPS7B81-Q1) can be used while the rest of the system is in sleep mode. However, based on the circuit diagram, there are a few more design considerations to make. R3 is used to set the interval period, which can be between 100ms to 7200s (2 hours). During the *ON-TIME* of the timer the DRV pin goes low and turns on the PMOS connecting the 5V signal to the WAKE pin initiating a wake signal. However, what about the pulse width? The answer lies in the operation of the DONE pin – when the DONE pin gets an active high pulse the DRV pin returns to a high level. If the DONE pin is not used the pulse width is the interval period of 50ms (typically). To be able to control that, two MOSFETs are added: a PMOS (Q3) and an NMOS (Q1). At the start of the drive period Q3 begins to conduct a signal into the RC timing circuit made of R1 and C1. When the voltage at C1 reaches the proper level the DRV pin returns high, shutting off Q3 but turning on Q1 to bring the DONE pins voltage back to ground so Q1 can restart the pulse again. The pulse width is some scalar multiple of the RC time constant. But what about the LDO, and why are there two MOSFETs in the circuit that are not directly used with the timer? The answer is quite simple: the sleep mode LDO must only be enabled in sleep mode. To do this, two NMOS devices (Q4 and Q5) form an inverter that allows VCC1 (either 3.3V or 5V) to shut off the LDO when VCC1 is active (for example, not in sleep or fail-safe mode) and when VCC1 shuts off, the LDO turns on powering the timer. This design allows full control of interval period, pulse width, and LDO control just from the turning on and off of VCC1. This design also has the benefit of being able to save on power unlike the 555-timer based design.

7 Cyclic Sensing Wake

Cyclic sensing wake is another potential wake feature that many mid-range SBCs have integrated. Cyclic sensing wake uses a High-Side Switch (HSS) output (typically HSS4), internal timers, one of the WAKE pins, plus a wake stimulus source that outputs a low signal to indicate a wake condition. Cyclic sensing wake can be simplified to six pins or components.

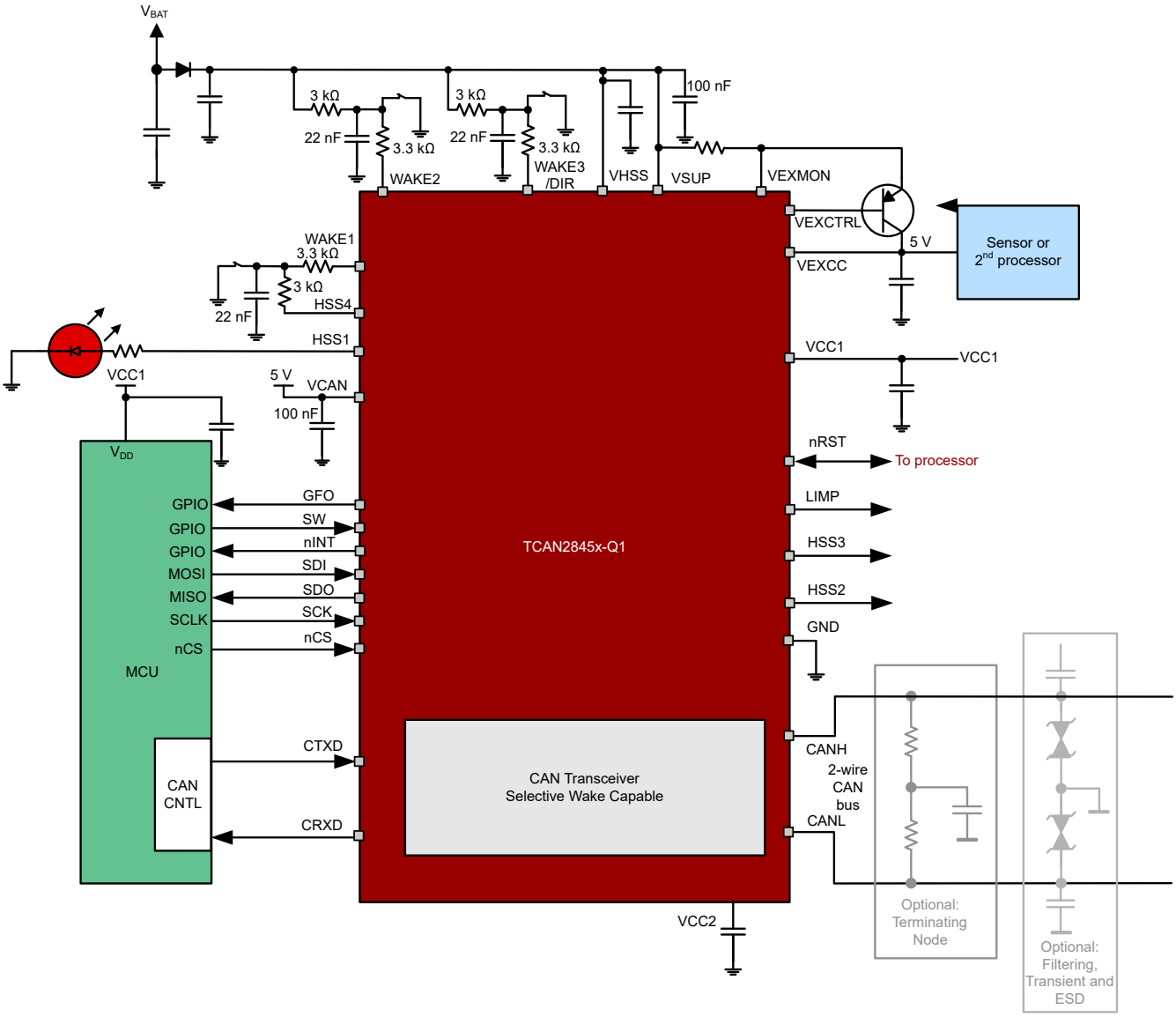

Figure 7-1. Application Diagram Showing Cyclic Sensing Wake Connection on WAKE1 and HSS4

Cyclic sensing mode is used to limit power consumption during sleep mode while the WAKE circuitry is only active during the *on-time* of the HSS module. Typically, the HSS module has a duty cycle that is very small to allow for only a small amount of current from VSUP to operate. The power consumption from cyclic sensing wake is generally going to consist of four main components: leakage current due to HSS module on VSUP and VHSS (assume that VHSS and VSUP are shorted), current required for timer setup, leakage from wake circuitry, and finally any current pulled out of the HSS output pin of interest. This can be simplified with the following equation but this does not include all potential current draw.

$$I_{cyclicsensewake} = \alpha + DC \times (ISUP_{HSS} + IHSS_{NOLOAD} + I_{SUP-WAKE}) + I_{LOAD-HSS} \quad (1)$$

These terms can be found directly in the data sheet, except for two of them. However, a user can still derive the values not included in the data sheet. ISUP_HSS, IHSS_NOLOAD, and ISUP_WAKE are all denoted in the data sheet – for example on the TCAN2847-Q1 device ISUP_HSS is set to 35uA (typical) and 60uA (max at TJ <= 85C), IHSS_NOLOAD is set to 100uA (typ) and 140uA (maximum at TJ <= 85C), and ISUP_WAKE is set to 1uA (typ) and 2uA (maximum with TJ<= 85C). The other specs must be derived – the α is a generic stand-in for the timer setup current – it is not specified directly – but it is included in another specification, I_SUPCSWAKE which is the additional current added for cyclic sensing wake and for the TCAN2847-Q1 is set to 5uA (typical) and 8uA(max, TJ <= 85C) with a duty cycle of 1%. This specification is really just the calculated result of the following equation.

$$I_{SUPCS-WAKE} = \alpha + DC \times I_{SUPHSS} \; \alpha = I_{SUPCS-WAKE} - DC \times I_{SUP_{HSS}} \; \alpha = 5\mu A\,(typ) - 0.01 \times (35\mu A(typ)) = 4.65\mu A \; \alpha = 8\mu A(MAX) - 0.01 \times (60\mu A(MAX)) = 7.4\mu A \quad (2)$$

For other similar SBCs, a user can apply the same analysis to find the specific α by device. The other specification is the load on the HSS pin. Looking back at the example the current is going to be extremely low when there is no external stimulus forcing the pin low. However, if there is external stimulus pulling the pin low there can be VHSS voltage level divided by 3k, so if VHSS equals 12V then when the WAKE pin node goes to 0V there can be 4mA of current drawn from HSS. Essentially during the wake stimulus application, large current is possible, but in general the load from HSS pins is negligible when the node is pulled up and is somewhere between 1uA and 3uA.

The smaller the duty cycle the smaller the total power consumption is, but the tradeoff is a smaller window for wake pin monitoring that can increase missed wake conditions.

8 CAN BWRR

The next wake up method is CAN Bus Wake Receive Request (BWRR), which is found in most CAN compliant transceivers and SBCs. This method can be used to wake both the SBC and the integrated CAN transceiver. If you are trying to activate the SBC, the SBC must be in sleep mode and the CAN transceiver must be wake capable. If the SBC is not in sleep mode, but in standby or normal mode, the CAN transceiver can be woken up through a BWRR as long as the transceiver is wake capable. A BWRR is initiated by a Wake-Up Pattern (WUP) which consists of a filtered dominant bit, followed by a filtered recessive bit, and ending with a filtered dominant bit again. The filter bits in the WUP are going to be longer than a standard bit length in a CAN network so communication can happen in-between WUP pulses and the WUP can still be considered valid.

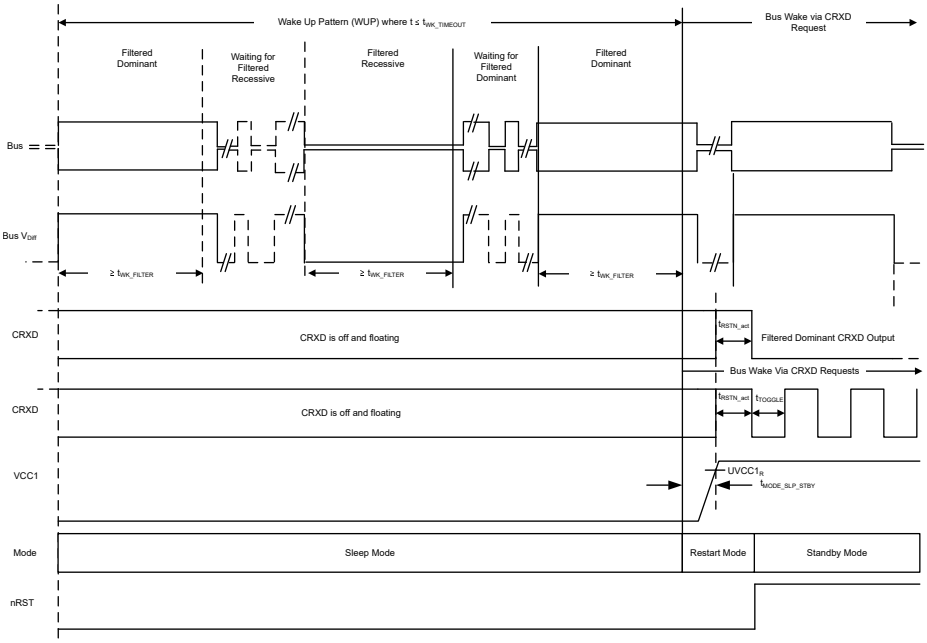

Figure 8-1. CAN BWRR Diagram (TCAN28xx and TCAN24xx Families)

9 Partial Networking

Partial networking is a medium power saving mode for a select group of CAN based SBCs as well as a select group of CAN based transceivers. Partial networking allows for specific transceivers or nodes to be turned on while the rest of the nodes on the CAN bus remain in an off state. Turning on only some of the transceivers/ nodes saves total power consumption in the system. There are up to three different partial networking conditions that can be implemented to give the end user precise control of what devices wake up during a selective wake/ Partial Networking application – these are ID verification, DLC verification, and data verification. However, most use cases only use ID verification as that is generally enough control for most systems.

After the device has been configured for partial networking (check the specific device for instructions on how to configure) the device can be put to sleep with Partial Networking enabled. Partial Networking does not begin the verification process until the reception of a WUP – which has the same properties as described in the CAN BWRR section. Upon the reception of the final filtered dominant bit, the device starts the verification process, which dramatically increases the supply current during the detection phase (for example, why it is only a medium power saving mode). The first check is ID verification. Traditionally this is an 11-bit binary string and is used to indicate device priority. During the configuration stage each CAN transceiver device has an ID assigned as well as the option to add an ID mask. The ID verification happens with a combination of the devices own ID as well as the ID mask. The ID mask indicates whether or not a specific ID bit is relevant or if it is a *don't care*. For example, assume the device ID is 0b00011000101 and there is an ID mask value of 0b11111111100 – the ones indicate a *don't care* value in the ID mask and the zeros represent bits that must be matched. This leaves a final ID check as 0bxxxxxxxxx01 – so for a proper ID match to occur the device requesting the wake up must have an ID with the last 2 bits being 0b01. A proper match can look like 0b11100011101 or 0b10101010101 while an ID value of 0b00011000110 is not matched. If the ID matches and DLC / Data verification isn't enabled the device then wakes up. The ID can also be extended with a 29-bit address; most modern devices that support partial networking supports classic 11-bit IDs as well as extended 29-bit IDs.

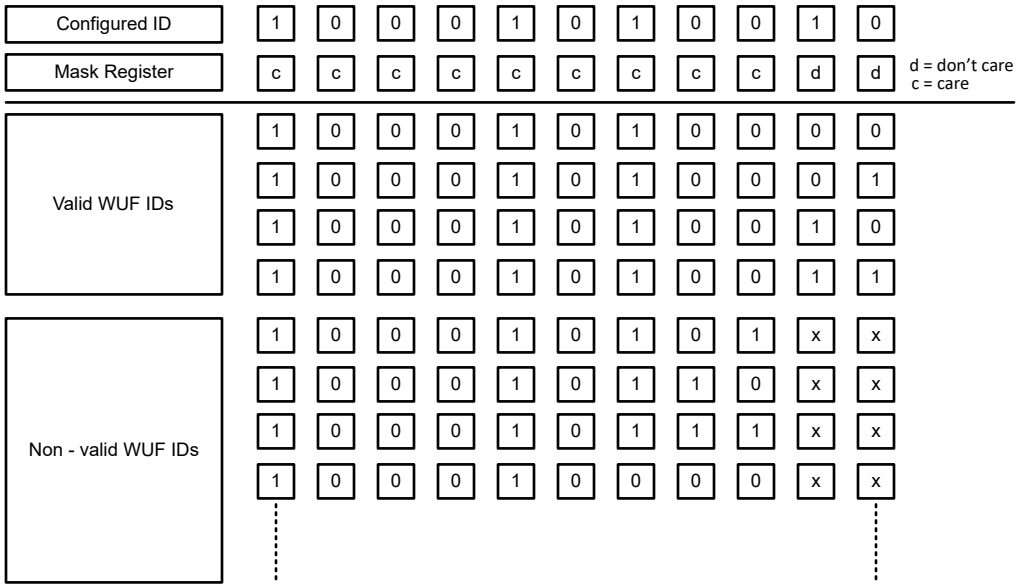

Figure 9-1. ID and ID Mask Example for Wake Up Frame (WUF)

The next step of verification if Data Length Code (DLC) and data verification are enabled is to check and verify these values. The DLC is the number of bytes being sent in the wake-up message. The DLC code indicates how many bytes are in the wake-up message. This can be anywhere from 0 bytes to 8 bytes. This is checked alongside data verification. Data verification starts by configuring data registers on the device to have a template to match to. For a match to be made there needs to be at least one matched logic one in the same byte and same position within byte.

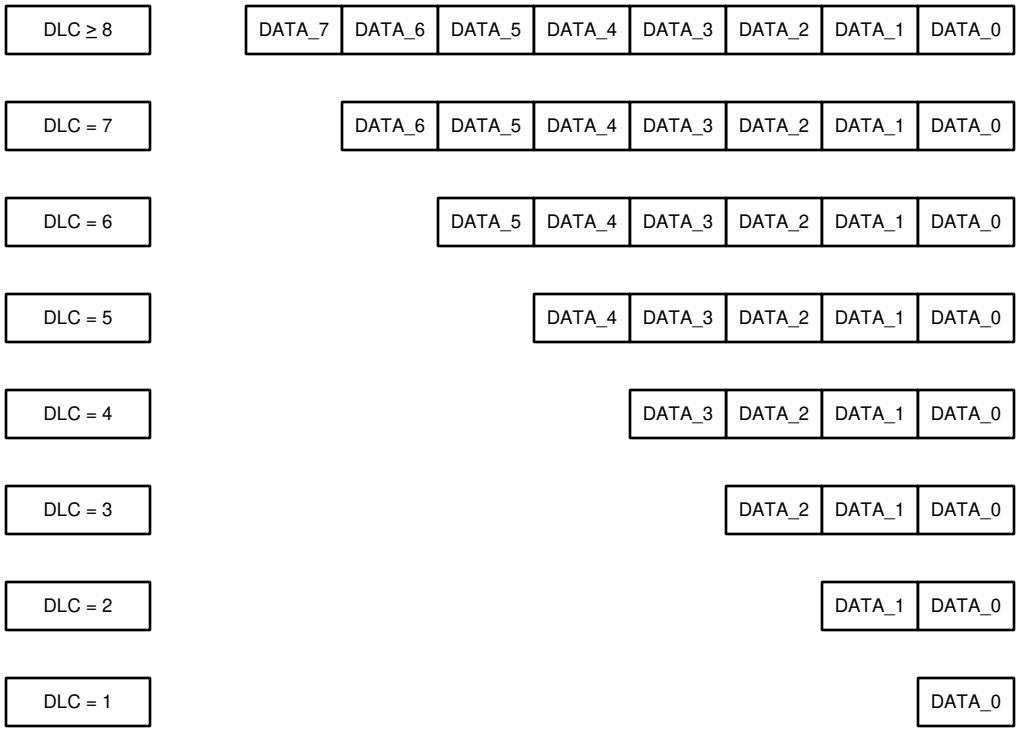

Figure 9-2. DLC to Data Byte

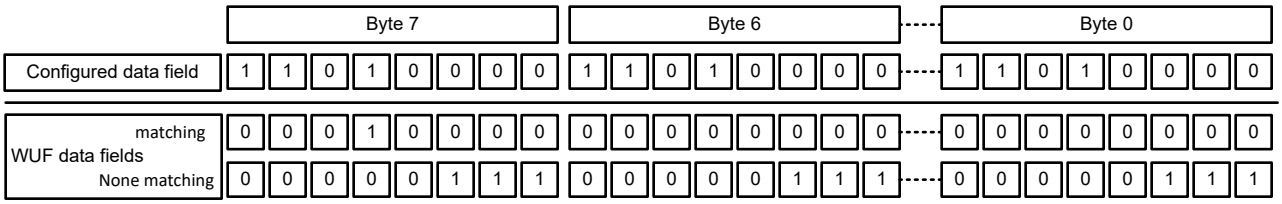

Figure 9-3. Data Validation Example

10 Summary

Note that SBCs have multiple ways to wake the device. Some of these methods apply to both discrete transceivers as well as integrated SBCs. The varied methods give designers a lot of flexibility in how the device transitions out of the low power sleep mode to perform work as needed by the application. The combination of local high voltage tolerant wake pins, digital wake ups, integrated timers for wake up, as well as being able to wake from the CAN bus through simple BWRR or a more complicated partial networking scheme give designers the tools to create a robust control system with TI SBCs. After understanding the wake methods a designer can implement varied use cases in automotive and even industrial contexts.

11 References

- Hubbard, Richard , Texas Instruments, *How Selective Wake Enables Partial Networking*, application note.
- Texas Instruments, *TCAN241x-Q1 Automotive CAN FD System Basis Chip (SBC) with Integrated Buck Regulator and Watchdog*, data sheet.
- Texas Instruments, *TCAN245x-Q1 Automotive Signal Improvement Capable CAN FD System Basis Chip (SBC) and Integrated Buck Regulator and Watchdog*, data sheet.
- Texas Instruments, *TCAN284x-Q1 Automotive CAN FD and LIN System Basis Chip (SBC) with Wake Inputs and High-Side Switches*, data sheet.
- Texas Instruments, *TCAN285x-Q1 Automotive CAN FD SIC and LIN System Basis Chip (SBC) with Wake Inputs and High-side Switches*, data sheet.
- Texas Instruments, *TLC3555-Q1 Automotive High-Speed CMOS Timer*, data sheet.
- Texas Instruments, *TMAG5131-Q1 Automotive Low-Power, High-Precision, Hall-Effect Switch*, data sheet.
- Texas Instruments, *AEC-Q100 Nano-Power System Timer for Power Gating,*, data sheet.
- Texas Instruments, *TPS7B81-Q1 Automotive, 150mA, Off-Battery, Ultra-Low IQ (3µA), Low-Dropout Regulator*, data sheet.

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.

Last updated 10/2025