Functional Safety Manual for MSPM0G3x0x-Q1

Functional Safety Information

Literature Number: SFFS624B
MARCH 2024 – REVISED AUGUST 2025

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

This document is a *Functional Safety Manual* for the Texas Instruments MSPM0G310x-Q1 and MSPM0G350x-Q1 component. The specific orderable part numbers supported by this functional safety manual are as follows:

- MSPM0G3507-Q1 (Orderable Part Numbers: M0G3507QPMRQ1, M0G3507QPTRQ1, M0G3507QRGZRQ1, M0G3507QRHBRQ1, M0G3507QDGS32RQ1, M0G3507QDGS28RQ1)
- MSPM0G3506-Q1 (Orderable Part Numbers: M0G3506QPMRQ1, M0G3506QPTRQ1, M0G3506QRGZRQ1, M0G3506QRHBRQ1, M0G3506QDGS32RQ1, M0G3506QDGS28RQ1)
- MSPM0G3505-Q1 (Orderable Part Numbers: M0G3505QPMRQ1, M0G3505QPTRQ1, M0G3505QRGZRQ1, M0G3505QRHBRQ1, M0G3505QDGS32RQ1, M0G3505QDGS28RQ1)
- MSPM0G3107-Q1 (Orderable Part Numbers: M0G3107QPMRQ1, M0G3107QPTRQ1, M0G3107QRGZRQ1, M0G3107QRHBRQ1, M0G3107QDGS32RQ1, M0G3107QDGS28RQ1, M0G3107QDGS20RQ1)

This functional safety manual provides information needed by system developers to help in the creation of a functional safety system using a MSPM0G3x0x-Q1 component. This document includes:

- An overview of the component architecture
- An overview of the development process used to decrease the probability of systematic failures
- An overview of the functional safety architecture for management of random failures
- The details of architecture partitions and implemented functional safety mechanisms

The following information is documented in the MSPM0G3x0x-Q1 Functional Safety Analysis Report and is not repeated in this document:

- Summary of failure in time (FIT) rates of the component
- Summary of functional safety metrics of the hardware component for targeted standards (for example IEC-61508, ISO-26262)
- Quantitative functional safety analysis (also known as FMEDA, Failure Modes, Effects, and Diagnostics Analysis) with detail of the different parts of the component, allowing for customized application of functional safety mechanisms
- Assumptions used in the calculation of functional safety metrics

The following information is documented in the Functional Safety Report and is not repeated in this document:

- Results of assessments of compliance to targeted standards

The user of this document needs a general familiarity with the MSPM0G3x0x-Q1 component. For more information, refer to the MSPM0G310x-Q1 and MSPM0G350x-Q1 data sheets. This document is intended to be used in conjunction with the pertinent data sheets, technical reference manuals, and other component documentation.

For information that is beyond the scope of the listed deliverables, contact your TI sales representative or go to TI Functional Safety.

Trademarks

Chapter 2

MSPM0G3x0x-Q1 Hardware Component Functional Safety Capability

This section summarizes the component functional safety capability.

This hardware component:

- Was developed as a functional Safety Element out of Context (SEooC)
- Was developed according to the relevant requirements of ISO 26262:2018
- Achieves systematic integrity level of ASIL D
- Includes sufficient functional safety mechanisms for random fault integrity up to ASIL B

Chapter 3

Development Process for Management of Systematic Faults

For functional safety development, it is necessary to manage both systematic and random faults. Texas Instruments follows a new-product development process for all of its components which helps to decrease the probability of systematic failures. This new-product development process is described in Section 3.1. Components being designed for functional safety applications will additionally follow the requirements of TI's functional safety development process, which is described in Section 3.2.

3.1 TI New-Product Development Process

Texas Instruments has been developing components for automotive and industrial markets since 1996. Automotive markets have strong requirements regarding quality management and product reliability. The TI new-product development process features many elements necessary to manage systematic faults. Additionally, the documentation and reports for these components can be used to assist with compliance to a wide range of standards for customer's end applications including automotive and industrial systems (for example, ISO 26262-4, IEC 61508-2).

This component was developed using TI's new product development process which has been certified as compliant to ISO 9001 / IATF 16949 as assessed by Bureau Veritas (BV).

The standard development process breaks development into phases:

- Assess
- Plan
- Create
- Validate

Section 3.1 shows the standard process.

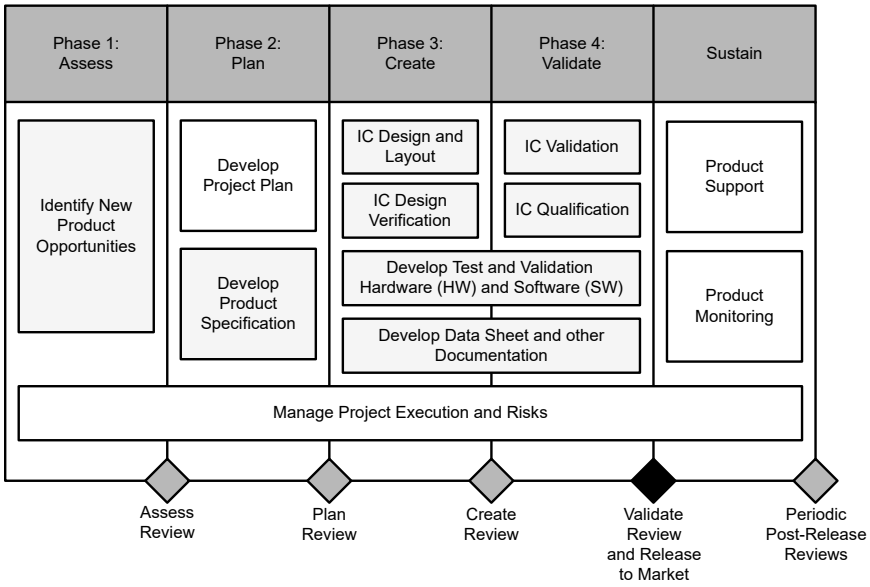

Figure 3-1. TI New-Product Development Process

3.2 TI Functional Safety Development Process

The TI functional safety development flow derives from ISO 26262 and IEC 61508 a set of requirements and methodologies to be applied to semiconductor development. This flow is combined with TI's standard new product development process to develop TI functional safety components. The details of this functional safety development flow are described in the TI internal specification - TI Functional Safety Hardware.

Key elements of the TI functional safety-development flow are as follows:

- Assumptions on system level design, functional safety concept, and requirements based on TI's experience with components in functional safety applications
- Qualitative and quantitative functional safety analysis techniques including analysis of silicon failure modes and application of functional safety mechanisms
- Base FIT rate estimation based on multiple industry standards and TI manufacturing data
- Documentation of functional safety work products during the component development
- Integration of lessons learned through multiple functional safety component developments, functional safety standard working groups, and the expertise of TI customers

Table 3-1 lists these functional safety development activities which are overlaid atop the standard development flow in Figure 3-1.

Refer to Appendix B for more information about which functional safety lifecycle activities TI performs.

The customer facing work products derived from this TI functional safety process are applicable to many other functional safety standards beyond ISO 26262 and IEC 61508.

Table 3-1. Functional Safety Activities Overlaid on top of TI's Standard Development Process

Assess	Plan	Create	Validate	Sustain and End-of-Life
Determine if functional safety process execution is required	Define component target SIL/ASIL capability	Develop component level functional safety requirements	Validate functional safety design in silicon	Document any reported issues (as needed)
Nominate a functional safety manager	Generate functional safety plan	Include functional safety requirements in design specification	Characterize the functional safety design	Perform incident reporting of sustaining operations (as needed)
End of Phase Audit	Verify the functional safety plan	Verify the design specification	Qualify the functional safety design (per AEC-Q100)	Update work products (as needed)
	Initiate functional safety case	Start functional safety design	Finalize functional safety case	
	Analyze target applications to generate system level functional safety assumptions	Perform qualitative analysis of design (i.e. failure mode analysis)	Perform assessment of project	
	End of Phase Audit	Verify the qualitative analysis	Release functional safety manual	
		Verify the functional safety design	Release functional safety analysis report	
		Perform quantitative analysis of design (i.e. FMEDA)	Release functional safety report	
		Verify the quantitative analysis	End of Phase Audit	
		Iterate functional safety design as necessary		
		End of Phase Audit		

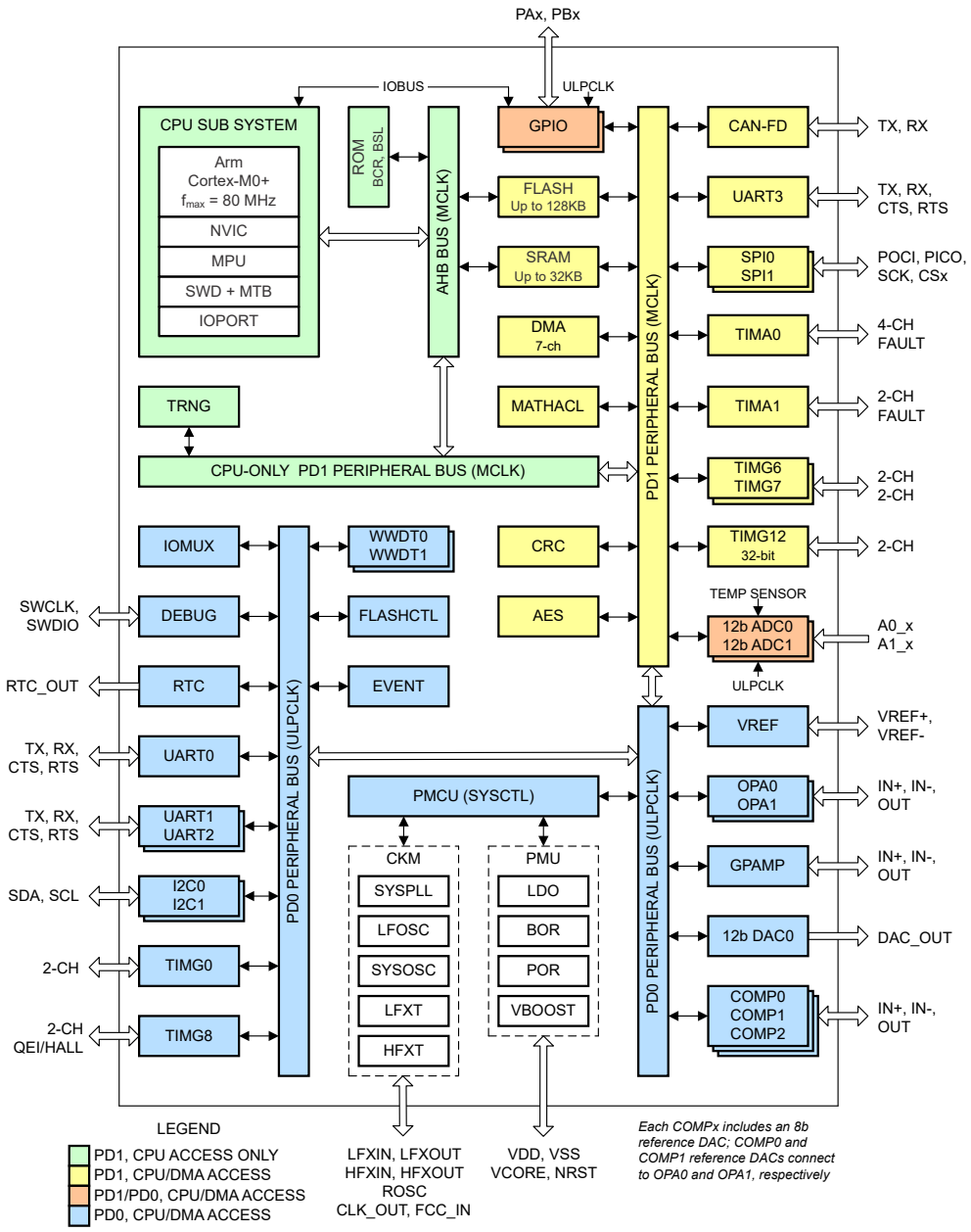

Figure 4-1. MSPM0G3x0x-Q1 Block Diagram

4.1 Targeted Applications

The MSPM0G3x0x-Q1 component is targeted at general-purpose functional safety applications. This is called Safety Element out of Context (SEooC) development according to ISO 26262-10:2018. In this case, the development is done based on assumptions on the conditions of the semiconductor component usage, and then the assumptions are verified at the system level. This section describes some of the target applications for this component, the component safety concept, and then describes the assumptions about the systems (also know as Assumptions of Use or AoU) that were made in performing the safety analysis.

Example target applications include, but are not limited to, the following:

- Automotive - Person occupancy detection
- Automotive - Lighting
- Automotive - Seat heaters
- Automotive - Window control

Figure 4-2 shows a generic block diagram for a seat heater (body application) system. This diagram is only an example and does not necessarily represent a complete system.

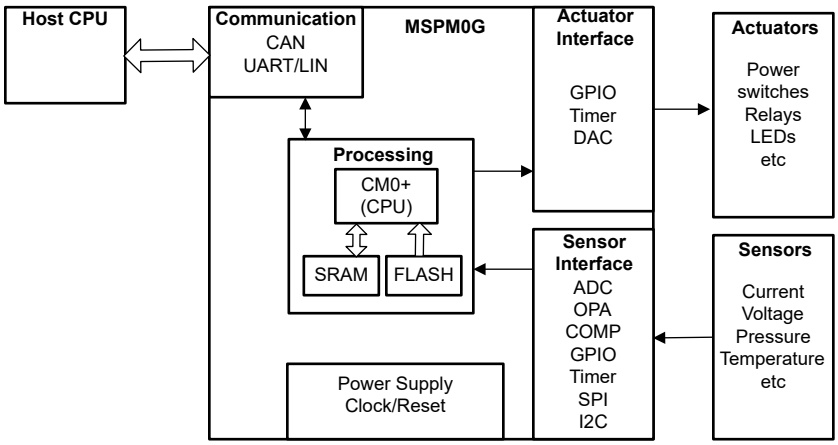

Figure 4-2. MSPM0G3x0x-Q1 Typical Application

4.2 Hardware Component Functional Safety Concept

In case of internal errors in the component, one or more of the following actions can be taken:

- Take the actuator output pins to a safe state (for example, the fault logic in timers can take the outputs to a safe state).
- Reset the device.
- Communicate the error to the host CPU and let the system take the appropriate action.

4.3 Functional Safety Constraints and Assumptions

In creating a functional Safety Element out of Context (SEooC) concept and doing the functional safety analysis, TI generates a series of assumptions on system level design, functional safety concept, and requirements. These assumptions (sometimes called Assumptions of Use) are listed below. Additional assumptions about the detailed implementation of safety mechanisms are separately located in Section 6.3 .

The MSPM0G3x0x-Q1 Functional Safety Analysis was done under the following system assumptions:

- **[SA_1]** The MSPM0G3x0x-Q1 MCU has interfaces to external sensors.
- **[SA_2]** The MSPM0G3x0x-Q1 MCU has interfaces to external actuators.
- **[SA_3]** The MSPM0G3x0x-Q1 MCU has interfaces to communicate with an external host controller.
- **[SA_4]** The MSPM0G3x0x-Q1 MCU has a programmable CPU to execute a controller function taking sensor inputs and controlling an actuator.

- **[SA_5]** The system integrator reviews the recommended diagnostics in the safety analysis report (FMEDA) and safety manual and determines the appropriate diagnostics to include in the system. These diagnostics are implemented according to the device safety manual and data sheet.
- **[SA_6]** The external power supply provides the appropriate power on for each of the power inputs. These rails are monitored for deviations outside the device specifications and a reset asserts, if the voltage is outside the range.
- **[SA_7]** The MSPM0G3x0x-Q1 MCU monitors failures on the external clock (if present).
- **[SA_8]** The MSPM0G3x0x-Q1 MCU monitors failures on external sensors.
- **[SA_9]** The MSPM0G3x0x-Q1 MCU monitors failures on external actuators.
- **[SA_10]** In case of internal errors in the MSPM0G3x0x-Q1 MCU or the interfacing sensors and actuators, the MSPM0G3x0x-Q1 MCU can be reset. The host controller monitors communication loss and determines that the MSPM0G3x0x-Q1 MCU is in a faulted state.
- **[SA_11]** The system integrator provisions an actuator disable-mechanism controller by the host controller.
- **[SA_12]** The system is assumed to require architectural metrics (random fault) complying to (up to) ASIL B.
- **[SA_14]** The system is assumed to have a FTTI > 10ms.
- **[SA_16]** The DEBUG function is considered as not safety critical.
- **[SA_17]** The RTC function is considered as not safety critical.
- **[SA_18]** The AES function is considered as not safety critical.
- **[SA_19]** The TRNG function is considered as not safety critical.
- **[SA_24]** The QM IPs are not used in safety-critical applications.
- **[SA_25]** TI assumes that the internal low power modes are not used in safety-critical applications.
- **[SA_26]** The system integrator considers all potential failure modes and mitigation measures associated with communication interfaces while implementing any end-to-end communication protection diagnostics techniques.
- **[SA_27]** The MATHACL function shall be considered as not safety critical.
- **[SA_28]** The GPAMP function shall be considered as not safety critical.
- **[COEX0]** The following components are assumed not safety related (NSR components):
  - RTC
  - TRNG
  - AES
  - DFT
  - DEBUG
  - MATHACL
- **[COEX1]** TI recommends that unused components are disabled in the application software.
- **[COEX2]** TI recommends that the unused interrupt sources of components are disabled.
- **[COEX3]** TI recommends that DMA unused triggers of components are disabled.
- **[COEX4]** TI recommends that unused fault inputs in timers are disabled.
- **[COEX5]** If external safety mechanisms are used, the system integrator is responsible for completing a dependent failure analysis at the system level.
- **[COEX6]** TI assumes that the NSR components are not used in the safety context.
- **[COEX7]** TI recommends that debug is disabled in safety-critical applications.
- **[COEX8]** TI recommends that a default interrupt service routine is coded for even the unused interrupts.
- **[COEX9]** TI recommends that the application does not use IPs as the trigger source of other IPs when those IPs are not safety related.
- **[COEX10]** TI recommends that the application does not program flash during safety-critical tasks.

There are some safety mechanisms required to cover dependent failures, refer to the section on Section 6.3.74 for more details.

During integration activities these assumptions of use and integration guidelines described for this component shall be considered. Use caution if one of the above functional safety assumptions on this component cannot be met, as some identified gaps can be unresolvable at the system level.

Chapter 5

Description of Hardware Component Parts

A semiconductor component can be divided into parts to enable a more granular functional safety analysis. This can be useful to help assign specific functional safety mechanisms to portions of the design where they provide coverage ending up with a more complete and customizable functional safety analysis. This section includes a brief description of each hardware part of this component and lists the functional safety mechanisms that can be applied to each. The content in this section is also summarized in Appendix A.

More details of the hardware components can be found in *MSPM0 G-Series 80MHz Microcontrollers Technical Reference Manual* .

5.1 ADC

Both 12-bit analog-to-digital converter (ADC) modules in these devices, ADC0 and ADC1, support fast 12-bit conversions with single-ended inputs and simultaneous sampling operation.

ADC features include:

- 12-bit output resolution at 4Msps with greater than 11 ENOB
- HW averaging enables 14-bit effective resolution at 250ksps
- Up to 17 total external input channels with individual result storage registers
- Internal channels for temperature sensing, supply monitoring, and analog signal chain (interconnection with, DAC)
- Software selectable reference:
  - Configurable internal reference voltage of 1.4V and 2.5V (requires decoupling capacitor on VREF± pins)
  - MCU supply voltage (VDD)
  - External reference supplied to the ADC through the VREF± pins
- Operates in RUN, SLEEP, and STOP modes

The following tests can be applied as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-1. ADC Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
ADC1	Periodic software read back of static configuration registers	Targeted toward configuration registers in ADC.
ADC2	Software Test for Functionality	Targeted toward the ADC functionality, including the sample pulse generation, ADC conversion, logic which captures the result, interrupt flag setting, DMA trigger generation, proper function of the analog-to-digital circuit, and so forth.
ADC3	ADC trigger overflow	Targeted toward ADC triggering circuit related faults.
ADC4	ADC window comparator	Covers faults in analog-to-digital conversion logic, and also covers faults in ADC inputs which result in the converted result going out of range.

Table 5-1. ADC Safety Mechanisms (continued)

Safety Mechanism	Description	Faults | Failure Modes
ADC5 (latent fault coverage)	Test of window comparator	This is a test to cover faults in the window comparator (the window comparator being a safety mechanism). This test covers latent faults within the window comparator.
ADC6	ADC trigger and output plausibility check	This is an application-specific check. Based on the application, software can check on signal properties, like range, bandwidth, sampling rate, and so forth. This check can potentially cover faults in ADC sampling and conversion, triggering logic, interrupt logic, and so forth.
WDT	Windowed watchdog event	Used to cover faults in sample generation, interrupt generation, DMA trigger generation, and event triggering to other IPs. Any periodic event which does not occur, or occurs at a different rate than expected, can be covered.

5.2 Comparator

The comparator peripheral in the device compares the voltage levels on two input terminals and provides a digital output (based on this comparison). The comparator supports the following key features:

- Programmable hysteresis
- Programmable reference voltage:
  - External reference voltage (VREF I/O)
  - Dedicated Internal reference voltage (1.4V, 2.5V) available in RUN, SLEEP, STOP, and STANDBY modes.
  - Integrated 8-bit reference DAC
- Configurable operation modes:
  - High-speed mode
  - Lower-power mode (not to be used in safety critical applications)
- Programmable output glitch filter delay
- Supports output wake-up device from all low power modes (not to be used in safety critical applications)
- Output is connected to an advanced timer fault handling mechanism
- The IPSEL and IMSEL bits in comparator registers can be used to select the comparator channel inputs from device pins or from internal analog modules.

The following tests must be applied for the targeted ASIL as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-2. Comparator Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
COMP1	Periodic software read back of static configuration registers	Targeted toward configuration registers in comparator.
COMP2	Software Test of Comparator using Internal DAC	Targeted toward testing comparator functioning using device DAC as first input and comparators internal reference DAC as second input.
COMP3	Comparator test using an external pin as a test input	Targeted to test the comparator functioning using an external pin as test input.
COMP4	Comparator hysteresis	This is a fault avoidance technique, which can be used to filter the noise on the input to the comparator.
COMP5	Redundant Comparator	A second comparator us used to monitor same signal as first comparator to create redundancy for monitoring a fault event response

Table 5-2. Comparator Safety Mechanisms (continued)

Safety Mechanism	Description	Faults | Failure Modes
WDT	Windowed watchdog event	Targeted toward interrupt generation logic. If a periodic interrupt is expected from the comparator watchdog, this mechanism can be used to detect faults in the interrupt logic.

5.3 DAC

The 12-bit buffered digital-to-analog converter (DAC) in these devices converts a digital input value into an analog voltage to a buffered output channel. The DAC supports the following key features:

- Up to 1Msps output sampling rate
- 8-bit or 12-bit voltage-output resolution
- Self-calibration option for offset error correction
- Straight binary or twos-complement data format
- Integrated sample time generator for generation of predefined sampling rates
- Integrated FIFO and support DMA operation
- Two hardware triggers from event fabric for conversion
- Programmable voltage reference options:
  - Supply voltage (VDD)
  - External reference voltage (VREF IO)
  - Internal reference voltage (1.4V, 2.5V)

The following tests can be applied as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-3. DAC Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
DAC1	Periodic software read back of static configuration registers	Targeted toward configuration registers in DAC.
DAC2	DAC to ADC Loopback	Targeted toward testing DAC functioning using ADC.
DAC3	FIFO underrun interrupt	Targeted toward FIFO control logic related faults and interrupt logic related faults.

5.4 OPA

The zero-drift op amps (OPAs) in these devices (OPA0 and OPA1) are chopper stabilized operational amplifiers with rail-to-rail input and output and a programmable gain stage feedback loop.

The OPA peripherals support the following key features:

- Software-selectable zero-drift chopper stabilization for improved accuracy and drift performance
- Factory trimming to remove offset error
- 6MHz GBW in standard (STD) mode and 100µA quiescent current in low-power (LP) mode
- Burnout current source (BCS) integrated to monitor sensor health
- Programmable gain amplifier (PGA) up to 32x

The OPA features configurable input muxes P-MUX, N-MUX, and M-MUX to support various analog signal chain amplifier configurations that include general purpose, inverting, noninverting, unity gain, cascade, noninverting cascade, difference, and more. The following tables list the input channel mapping for each OPA.

1. The connection to OPA and DAC_OUT connects using the PA15 pin. When connecting DAC_OUT to OPA, avoid using external circuitry on the PA15 pin.

The following tests can be applied as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-4. OPA Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
OA1	Periodic software read back of static configuration registers	Targeted toward configuration registers in OPA.
OA2	Test of OA, using internal DAC as driver	Targeted toward testing OPA functioning using ADC (to measure) and DAC.
OA3	ADC monitoring of OA input	Targeted toward testing OPA functioning by online monitoring using ADC.

5.5 CPU

The CPU subsystem (MCPUSS) implements an Arm® Cortex®-M0+ CPU, an instruction prefetch and cache, a system timer, a memory protection unit, and interrupt management features. The Arm Cortex-M0+ is a 32-bit CPU which delivers high performance and low power to embedded applications. Key features of the CPU subsystem includes:

- Arm Cortex-M0+ CPU supporting clock frequencies from 32MHz to 80MHz
  - ARMv6-M thumb instruction set (little endian) with single-cycle 32×32 multiply instruction
  - Single-cycle access to GPIO registers through Arm single-cycle I/O port
- Prefetch logic to improve sequential code execution, and I-cache with four 64-bit cache lines
- System timer (SysTick) with 24-bit down counter and automatic reload
- Memory protection unit (MPU) with eight programmable regions
- Nested vectored interrupt controller (NVIC) with four programmable priority levels and tail-chaining
- Interrupt groups, for expanding the total interrupt sources, with jump index for low-interrupt latency

The following tests must be applied for the targeted ASIL as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-5. CPU Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
CPU1	ARM software test library	Targeted toward the Cortex-M0+ CPU and NVIC.
CPU2	Writes and reads back data to different regions of memory to detect faults in the bus interconnect components.	Targeted toward the bus decoders and interface logic in the CPU subsystem. These decoders route the CPU access to different components based on the address.
CPU3	Software diversified redundancy	Used to target CPU functioning. This is an application-specific check, in which the same computation is performed in two different software functions and the results are compared.
CPU4	Periodic software read back of static configuration registers	Targeted toward the configuration registers in the CPU subsystem (registers in the interrupt grouping logic, and so forth).
SYSCTL11	Boot process timeout	Targeted toward boot ROM.
WDT	Windowed watchdog event	Targeted toward CPU control flow, any CPU bus related faults, faults in the CPU interrupt logic, and so forth.

5.6 RAM

The MSPM0Gxx MCUs include a low-power, high-performance SRAM memory with zero wait state access across the supported CPU frequency range of the device. The MSPM0Gxx MCUs also provide up to 32KB of SRAM with hardware parity. SRAM memory can be used for storing volatile information, such as the call stack, heap, global data, and code. The SRAM memory content is fully retained in run, sleep, stop, and standby operating modes and is lost in shutdown mode. A write protection mechanism is provided to allow the application to prevent unintended modifications to the SRAM memory. Write protection is useful when placing executable

code into SRAM; as write protection provides a level of protection against unintentional overwrites of code by either the CPU or DMA. Placing code in SRAM can improve performance of critical loops by enabling a zero wait state operation and lower power consumption.

The following tests can be applied as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-6. RAM Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
SYSMEM1	Software read of memory DMA	Targeted toward the DMA bus decoder in the SRAM controller and the arbitration logic.
SYSMEM2	Software read of memory CPU	Targeted toward the CPU bus decoder in the SRAM controller and the arbitration logic.
SYSMEM4	Parity protection on SRAM	Targeted toward faults in SRAM.
SYSMEM3 (Latent fault coverage)	Parity logic test	Targeted toward latent faults in parity logic.

5.7 FLASH

A single bank of non-volatile Flash memory is provided for storing executable program code and application data. Key features of the Flash include:

- Hardware ECC protection (encode and decode) with single-bit error correction and double-bit error detection
- In-circuit program and erase operations supported across the entire recommended supply range
- Small 1KB sector sizes (minimum erase resolution of 1KB)
- Up to 100,000 program and erase cycles on the lower 32kB of the Flash memory, with up to 10,000 program and erase cycles on the remaining Flash memory (devices with 32kB support 100,000 cycles on the entire Flash memory). For a complete description of the Flash memory, refer to the NVM chapter of the technical reference manual.

The following tests can be applied as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-7. Flash Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
CPU3	Software diversified redundancy	Targeted toward the arbitration logic. If due to fault in arbitration logic, incorrect data is returned, this test can be used to cover such faults.
DMA2	Software DMA transfer test	Targeted toward DMA bus decoder and the arbitration logic.
FLASH1	Flash ECC	Targeted toward the faults in the flash memory.
FLASH2	Flash CRC	Targeted towards multipoint latent faults in the flash memory.
FXBAR2	Periodic software read back of flash data	Targeted toward the decoding logic on the CPU read bus.
FXBAR3 (latent fault coverage)	Software check of ECC checker logic	This is a test of diagnostic, used to check the function of the ECC checker.
FXBAR4	Write protection of flash	Targeted toward the faults in the programming interface.
WDT	Windowed watchdog event	Targeted toward the arbitration logic and flash read interface. Any fault in this logic which results in incorrect data being returned to CPU causing a CPU decoding an incorrect instruction.

5.8 GPIO

The general purpose input output (GPIO) peripheral provides the user with a means to write data out, read data into the device pins, and read data from the device pins. Through the use of the Port A and Port B GPIO peripherals, these devices support up to 60 GPIO pins.

The key features of the GPIO module include:

- Zero wait state MMR access from the CPU
- Set, clear, and toggle, multiple bits without the need of a read-modify-write construct in software
- GPIOs with *Standard with Wake* drive functionality able to wake the device from SHUTDOWN mode
- The *FastWake* feature enables low-power wakeup from STOP and STANDBY modes for any GPIO port
- User controlled input filtering

The following tests must be applied for the targeted ASIL as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-8. GPIO Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
GPIO1	Online monitoring using I/O loopback	Targeted toward the GPIO DOUT register and the interface to I/O.
GPIO2	Periodic software read back of static configuration registers	Targeted toward GPIO configuration registers.
GPIO3	GPIO multiple (redundant) inputs and outputs	This is an application-level test. For critical GPIOs, a redundancy mechanism can be employed to get high coverage.
WDT	Windowed watchdog event	This is targeted toward the interrupt generation logic in GPIO. If an interrupt is expected to occur periodically, this mechanism can be used. WDT is also used to cover pin failures in case the pin is used as an external interrupt source.

5.9 DMA

The direct memory access (DMA) controller allows movement of data from one memory address to another without CPU intervention. For example, the DMA can be used to move data from ADC conversion memory to SRAM. The DMA reduces system power consumption by allowing the CPU to remain in low-power mode, without having to wake to move data to or from a peripheral.

The DMA in these devices support the following key features:

- Seven independent DMA transfer channels
  - Four basic channel supports (single transfer modes)
  - Three full-feature channel supports (repeated transfer modes)
- Configurable DMA channel priorities
- Byte (8-bit), short word (16-bit), word (32-bit), and long word (64-bit) or mixed byte and word transfer capability
- Transfer counter block size supports up to 64k transfers of any data type
- Configurable DMA transfer trigger selection
- Active channel interruption to service other channels
- Early interrupt generation for ping-pong buffer architecture
- Cascading channels upon completion of activity on another channel
- Stride mode to support data reorganization, such as 3-phase metering applications

The following tests must be applied for the targeted ASIL as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-9. DMA Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
DMA1	Periodic software read back of static configuration registers	Targeted toward configuration registers of DMA.
DMA2	Software DMA transfer test	Targeted toward DMA bus interface logic and the DMA channel which sequences the read and write access based on the channel configuration.
DMA3	Software DMA channel test	Targeted toward the channels used in the application context.
DMA4	CRC check of the transferred data	Targeted toward DMA bus interface which results in data corruption.
SYSCTL11	Boot process timeout	DMA is used to transfer trims during the boot process. This test targets the DMA logic which is used for this.
WDT	Windowed watchdog event	Target faults which result in either DMA transfer not starting (triggers not getting generated, for example) or transfers not completing (DMA channel faults and bus hangs) resulting in the CPU program sequence also malfunctioning.

5.10 SPI

The serial peripheral interface (SPI) peripherals in these devices support the following key features:

- Support ULPCLK/2 bit rate and up to 32Mbits/s in both controller and peripheral mode
- Configurable as a controller or a peripheral
- Configurable chip select for both controller and peripheral
- Programmable clock prescaler and bit rate
- Programmable data frame size from
- Programmable data frame size from 7-bits to 16-bits (peripheral mode)
- Separated transmit and receive FIFOs support DMA data transfer
- Supports TI mode, Motorola® mode and National Microwire™ format

The following tests can be applied as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-10. SPI Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
SPI1	Software test of function using I/O loopback	Targets the transmit and receive function, including the clocking, shift registers, FIFOs, and the associated control logic.
SPI2	Periodic software read back of static configuration registers	Targets the static configuration registers in SPI.
SPI3	SPI periodic safety message checks	This is an application-level check, in which safety messages can be exchanged periodically. This mechanism covers faults, which result in communication breakdown. These faults can be in the external line, I/Os, the transmit and receive logic, interrupt generation logic, and so forth.
SPI4	Information redundancy techniques including end-to-end safing	This is an application-level check, in which additional information (for example, the CRC of the message) is included along with the message. These checks can be used to cover faults resulting in data corruption. For example, faults in FIFO, in the shift registers, and so forth.

Table 5-10. SPI Safety Mechanisms (continued)

Safety Mechanism	Description	Faults | Failure Modes
SPI5	Transmission redundancy	This test is an application-level check, in which the same message is transmitted multiple times. This test is effective for detecting transient faults resulting in some messages getting corrupted. For example, transient faults in FIFOs.
WDT	Windowed watchdog event	Targeted toward faults which result in missing interrupts (periodic interrupts) affecting the program sequence of the CPU. These faults can be faults in the interrupt logic, the logic which sets the interrupt flags, and so forth.

5.11 I2C

The inter-integrated circuit interface (I2C) peripherals, in these devices, provide bidirectional data transfers with other I2C devices on the bus and support the following key features:

- 7-bit and 10-bit addressing mode with multiple 7-bit target addresses
- Multiple-controller transmitter or receiver mode
- Target receiver or transmitter mode with configurable clock stretching
- Support standard-mode (Sm), with a bit rate up to 100kbit/s
- Support fast-mode (Fm), with a bit rate up to 400kbit/s
- Support fast-mode plus (Fm+), with a bit rate up to 1 Mbit/s
- Separated transmit and receive FIFOs support DMA data transfer
- Support SMBus 3.0 with PEC, ARP, timeout detection, and host support
- Wakeup from low-power mode on address match
- Support analog and digital glitch filter for input signal glitch suppression

The following tests must be applied for the targeted ASIL as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-11. I2C Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
I2C1	Software test of function using I/O loopback	Targets the transmit and receive function, including the clocking, shift registers, FIFOs, and the associated control logic.
I2C2	Periodic software read back of static configuration registers	Targets the static configuration registers in I2C.
I2C3	Information redundancy techniques including end-to-end safing	This is an application-level check, in which additional information (for example, the CRC of the message) is included along with the message. These checks can be used to cover faults resulting in data corruption. For example, a fault in FIFO, in the shift registers, and so forth.
I2C4	Transmission redundancy	This test is an application-level check, in which the same message is transmitted multiple times. This test is effective to detect transient faults resulting in some messages getting corrupted. For example, transient faults in FIFOs.
I2C5	Timeout monitoring	This is an application-level check, in which safety messages can be exchanged periodically. This covers faults which result in communication breakdown. These faults can be in the external line, I/Os, the transmit and receive logic, interrupt generation logic, and so forth.

Table 5-11. I2C Safety Mechanisms (continued)

Safety Mechanism	Description	Faults | Failure Modes
I2C6	Test of CRC function	I2C has a CRC checker when configured in the SMBUS mode. This checker checks the proper function of the CRC checker logic by sending corrupt messages as part of the application.
I2C7	Packet error check in SMBUS mode	This targets faults which result in data corruption, for example, faults on the external line, in the FIFOs, shift registers, and so forth.
WDT	Windowed watchdog event	Targeted toward faults which result in missing interrupts (periodic interrupts) affecting the program sequence of the CPU. These faults can be faults in the interrupt logic, the logic which sets the interrupt flags, and so forth.

5.12 UART

The UART peripherals (UART0, UART1, UART2 and UART3) provide the following key features:

- Standard asynchronous communication bits for start, stop, and parity
- Fully programmable serial interface
  - Five, six, seven, or eight data bits
  - Even, odd, stick, or no-parity bit generation and detection
  - One or two stop bit generation
  - Line-break detection
  - Glitch filter on the input signals
  - Programmable baud rate generation with oversampling by 16, 8, or 3
  - Local interconnect network (LIN) mode support
- Separated transmit and receive FIFOs support DAM data transfer
- Support transmit and receive loopback mode operation
- See Table 5-12 for detailed information on supported protocols

Table 5-12. UART Features

UART Features	UART0 (Extend)	UART2, UART3, and UART4 (Main)
Active in stop and standby mode	Yes	Yes
Separate transmit and receive FIFOs	Yes	Yes
Supports hardware flow control	Yes	Yes
Supports 9-bit configuration	Yes	Yes
Supports LIN mode	Yes	-
Supports DALI	Yes	-
Supports IrDA	Yes	-
Supports ISO7816 Smart Card	Yes	-
Supports Manchester coding	Yes	-

The following tests must be applied for the targeted ASIL as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-13. UART Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
UART1	Software test of function using I/O loopback	Targets the transmit and receive function, including the clocking, shift registers, FIFOs, and the associated control logic.
UART2	Periodic software read back of static configuration registers	Targets the static configuration registers in UART.
UART3	Information redundancy techniques including end-to-end safing	This is an application-level check, in which additional information (for example, the CRC of the message) is included along with the message. These checks can be used to cover faults resulting in data corruption. For example, fault in FIFO, in the shift registers, and so forth.
UART4	Transmission redundancy	This test is an application-level check, in which the same message is transmitted multiple times. This test is effective to detect transient faults resulting in some messages getting corrupted. For example, transient faults in FIFOs.

Table 5-13. UART Safety Mechanisms (continued)

Safety Mechanism	Description	Faults | Failure Modes
UART5	Timeout monitoring	This is an application-level check, in which safety messages can be exchanged periodically. This covers faults which result in communication breakdown. These faults can be in the external line, I/Os, the transmit and receive logic, interrupt generation logic, and so forth.
UART6	UART error flags	This safety mechanism provides detection for some typical failures which result in protocol violations and can cover faults in the transmit and receive control logic or on the external line.
UART7	UART glitch filter	This mechanism is a fault-avoidance measure against glitches on the RX line.
WDT	Windowed watchdog event	Targeted toward faults which result in missing interrupts (periodic interrupts) affecting the program sequence of the CPU. These faults can be faults in the interrupt logic, the logic which sets the interrupt flags, and so forth.

5.13 Timers (TIMx)

There are two timer peripherals in these devices (which support the following key features); (1) TIMGx (general-purpose timer) and a TIMAx (advanced timer). The TIMGx is a subset of TIMAx, which means these timers share many common features that are compatible in software. For specific configuration, see Table 5-14:

Specific features for the general-purpose timer **(TIMGx)** include:

- 16-bit and 32-bit timers with up, down, or up-down counting modes, with repeat-reload mode
- Selectable and configurable clock source
- 8-bit programmable prescaler to divide the counter clock frequency
- Two independent CC channels for
  - Output compare
  - Input capture
  - PWM output
  - One-shot mode
- Support quadrature encoder interface (QEI) for positioning and movement sensing available in TIMG8
- Support synchronization and cross trigger among different TIMx instances in the same power domain
- Support interrupt and DMA trigger generation and cross peripherals (such as ADC) trigger capability
- Cross trigger event logic for Hall sensor inputs (TIMG8)

Specific features for the advanced timer **(TIMAx)** include:

- 16-bit timer with up, down, or up-down counting modes, with repeat-reload mode
- Selectable and configurable clock source
- 8-bit programmable prescaler to divide the counter clock frequency
- Repeat counter to generate an interrupt or event only after a given number of cycles of the counter
- Up to four independent CC channels for
  - Output compare
  - Input capture
  - PWM output
  - One-shot mode
- Two additional capture and compare channels for internal events (CC4 and CC5)
- Shadow register for load and CC register available in TIMA0
- Complementary output PWM
- Asymmetric PWM with programmable dead band insertion
- Fault handling mechanism to verify the output signals in a safe user-defined state when a fault condition is encountered
- Support synchronization and cross trigger among different TIMx instances in the same power domain
- Support interrupt and DMA trigger generation and cross peripherals (such as ADC) trigger capability
- Two additional capture and compare channels for internal events

Table 5-14. TIMx Configurations

Timer Name	Power Domain	Resolution	Prescaler	Repeat Counter	Capture / Compare Channels	Phase Load	Shadow Load	Shadow CC	Deadband	Fault	QEI
TIMG0	PD0	16-bit	8-bit	–	2	–	–	–	–	–	–
TIMG6	PD1	16-bit	8-bit	–	2	–	–	–	–	–	–
TIMG7	PD1	16-bit	8-bit	–	2	–	Yes	Yes	–	–	–
TIMG8	PD0	16-bit	8-bit	–	2	–	–	–	–	–	Yes
TIMG12	PD1	32-bit	–	–	2	–	–	Yes	–	–	–
TIMA0	PD1	16-bit	8-bit	8-bit	4	Yes	Yes	Yes	Yes	Yes	–
TIMA1	PD1	16-bit	8-bit	8-bit	2	Yes	Yes	Yes	Yes	Yes	–

Table 5-15. TIMx Cross Trigger Map (PD1)

TSEL.ETSEL Selection	TIMA0	TIMA1	TIMG6	TIMG7	TIMG12
0	TIMA0.TRIG0	TIMA0.TRIG0	TIMA0.TRIG0	TIMA0.TRIG0	TIMA0.TRIG0

Table 5-15. TIMx Cross Trigger Map (PD1) (continued)

TSEL.ETSEL Selection	TIMA0	TIMA1	TIMG6	TIMG7	TIMG12
1	TIMA1.TRIG0	TIMA1.TRIG0	TIMA1.TRIG0	TIMA1.TRIG0	TIMA1.TRIG0
2	TIMG6.TRIG0	TIMG6.TRIG0	TIMG6.TRIG0	TIMG6.TRIG0	TIMG6.TRIG0
3	TIMG7.TRIG0	TIMG7.TRIG0	TIMG7.TRIG0	TIMG7.TRIG0	TIMG7.TRIG0
4	TIMG12.TRIG0	TIMG12.TRIG0	TIMG12.TRIG0	TIMG12.TRIG0	TIMG12.TRIG0
5	TIMG8.TRIG0	TIMG8.TRIG0	TIMG8.TRIG0	TIMG8.TRIG0	TIMG8.TRIG0
6 to 15	Reserved				
16	Event Subscriber Port 0 (FSUB0)				
17	Event Subscriber Port 1 (FSUB1)				
18-31	Reserved				

Table 5-16. TIMx Cross Trigger Map (PD0)

TSEL.ETSEL SELECTION	TIMG0	TIMG8
0	TIMG0.TRIG0	TIMG0.TRIG0
1	TIMG8.TRIG0	TIMG8.TRIG0
2 to 15	Reserved	
16	Event Subscriber Port 0 (FSUB0)	
17	Event Subscriber Port 1 (FSUB1)	
18-31	Reserved	

For more details, see the TIMx chapter of the *MSPM0 G-Series 80MHz Microcontrollers Technical Reference Manual* .

The following tests can be applied as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-17. Timers Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
TIM1	Test for PWM generation	Targeted toward PWM generation logic, including the counters, compare registers, clocking logic, output generation logic, and so forth.
TIM2	Periodic software read back of IP static configuration registers	Targets the static configuration registers in timer.
TIM3 (latent fault coverage)	Test for fault generation	This test is a test for diagnostic, which checks the functioning of fault detection logic in timer. **Note** This test is applicable only to TIMAx.
TIM4	Fault detection to take the PWMs to safe state	This safety mechanism can be used to detect faults which result in system-level failures like overvoltage and undervoltage and overcurrent and undercurrent. The external faults can be monitored using the fault pins or the analog comparators. The faults which can be covered include the faults in the PWM generation logic, faults in external drivers, and so forth. **Note** This test is applicable only to TIMAx.

Table 5-17. Timers Safety Mechanisms (continued)

Safety Mechanism	Description	Faults | Failure Modes
TIM5	Input capture on two or more timer instances	This test is used to cover the faults in the capture mode logic. The faults can be in clocking, capture logic, counter logic, and so forth.
TIM6	Timer period monitoring.	This test is a run time check, in which the duration between two interrupts can be measured (using another timer). This check is useful in detecting faults which result in the counter taking more or less time than expected and can also cover the clocking related faults.
WDT	Windowed watchdog event	Targeted toward faults which result in missing interrupts (periodic interrupts) affecting the program sequence of the CPU. These faults can be faults in the interrupt logic, the logic which sets the interrupt flags, the logic which generates hardware triggers for other IPs (ADC, for example), and so forth.

5.14 Power Management Unit (PMU)

The power management unit (PMU) generates the internally regulated core supplies for the device and provides supervision of the external supply (VDD). The PMU also contains the band gap voltage reference used individually by the PMU as well as analog peripherals. Key features of the PMU include:

- Power-on-reset (POR) supply monitor
- Brown-out-reset (BOR) supply monitor with early warning capability using three programmable thresholds
- Core regulator with support for RUN, SLEEP, STOP, and STANDBY operating modes to dynamically balance performance with power consumption.
- Parity-protected trim to immediately generate a power-on-reset (POR) in the event that a power management trim is corrupted.

The following tests must be applied for the targeted ASIL as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-18. PMU Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
CPU3	Software diversified redundancy	Targeted toward faults which can result in incorrect clock frequency (resulting in some computation failures) or failure to do a reset when required, and so forth.
GEN_IP_READ_STATIC_CONFIGURATION	Periodic software read back of IP static configuration registers	Targeted toward logic which is used to enable and disable IPs. If a fault results in an IP not getting enabled, the software read of the configuration registers of the IP detects such failures.
SYSCTL10	External voltage monitor	This is a system-level diagnostic to detect faults on the internal LDO.
SYSCTL14	Brownout voltage monitor	This is a safety mechanism which monitors the power supply to the chip and can detect undervoltage conditions early (generates an NMI).
SYSCTL16	External watchdog timer	This is a system-level diagnostic which can be used to cover faults in the reset generation logic (reset asserted when not required). In general, this diagnostic covers any fault which results in incorrect program execution timing.

Table 5-18. PMU Safety Mechanisms (continued)

Safety Mechanism	Description	Faults | Failure Modes
SYSCTL16 (latent fault coverage)	External watchdog timer	This mechanism (if used) can also detect faults in the MCLK monitor circuit.
SYSCTL8	Brownout reset (BOR) supervisor	This is a safety mechanism, which monitors the power supply to the chip and can detect undervoltage conditions (generates a reset). This mechanism is triggered if the voltage falls below the specified range.
SYSCTL8 (latent fault coverage)	Brownout reset (BOR) supervisor	In case brownout supervisor is not functioning, this mechanism can trigger when the voltage falls below the specified range.

5.15 Clock Module (CKM)

The clock module provides the following oscillators:

1. **LFOSC**: Internal low-frequency oscillator (32KHz)
2. **SYSOSC**: Internal high-frequency oscillator (4MHz or 32MHz with factory trim, 16MHz or 24MHz with user trim)
3. **LFXT/LFCKIN**: Low-frequency external crystal oscillator or digital clock input (32KHz)
4. **HFXT/HFCKIN**: High-frequency external crystal oscillator or digital clock input (4MHz to 48MHz)
5. **SYSPLL**: System phase locked loop with three outputs (32MHz to 80MHz)

The following clocks are distributed by the clock module for use by the processor, bus, and peripherals:

- **MCLK**: Main system clock for PD1 peripherals, derived from SYSOSC, LFCLK, or HSCLK, active in RUN and SLEEP modes
- **CPUCLK**: Clock for the processor (derived from MCLK), active in RUN mode
- **ULPCLK**: Ultra-low power clock for PD0 peripherals, active in RUN, SLEEP, STOP, and STANDBY modes
- **MFCLK**: 4MHz fixed mid-frequency clock for peripherals, available in RUN, SLEEP, and STOP modes
- **MFPCLK**: 4MHz fixed mid-frequency precision clock, available in RUN, SLEEP, and STOP modes
- **LFCLK**: 32kHz fixed low-frequency clock for peripherals or MCLK, active in RUN, SLEEP, STOP, and STANDBY modes
- **ADCCLK**: ADC clock, available in RUN, SLEEP and STOP modes
- **CLK_OUT**: Used to output a clock externally, available in RUN, SLEEP, STOP, and STANDBY modes
- **HFCLK**: High-frequency clock derived from HFXT or HFCLK_IN, available in RUN and SLEEP mode
- **HSCLK**: High-speed clock derived from HFCLK or the SYSPLL, available in RUN and

The following tests can be applied as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-19. CKM Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
CPU3	Software diversified redundancy	Targeted toward faults which can result in incorrect clock frequency (resulting in some computation failures) or failure to do a reset when required, and so forth.
SYSCTL1	MCLK monitor	This safety mechanism is targeted toward faults which result in MCLK frequency going very low or MCLK not toggling.
SYSCTL11	Boot process timeout	This safety mechanism can detect faults in logic which interacts with the boot software to indicate boot completion, boot failures, and so forth.
SYSCTL2	HFCLK start-up monitor	This safety mechanism can be used to check if HFCLK is functioning before switching the clock source to HFCLK.

Table 5-19. CKM Safety Mechanisms (continued)

Safety Mechanism	Description	Faults | Failure Modes
SYSCTL3	LFCLK monitor	This safety mechanism is used detect failures on LFCK.
SYSCTL5	Periodic software read back of static configuration registers	Targets the static configuration registers in SYSCTL.
SYSCTL9	Clock frequency measurement	This safety mechanism can be used to do a periodic check of the clock frequency.
WDT	Windowed watchdog event	Targeted toward faults which result in incorrect clock frequencies and covers faults in various clocking components.
WDT (latent fault coverage)	Windowed watchdog event	This mechanism also acts as a backup in case the MCLK monitor malfunctions, for example.

5.16 CAN-FD

The controller area network (CAN) controller enables communication with a CAN2.0A, CAN2.0B, or CAN-FD bus and is compliant to ISO 11898-1:2015 standard supporting up to 5Mbit/s bit rate. Key features of the CAN-FD peripheral include:

- Full support for 64-byte CAN-FD frames
- Dedicated 1kB message SRAM with ECC
- Configurable transmit FIFO, transmit queue and event FIFO (up to 32 elements)
- Up to 32 dedicated transmit buffers and 64 dedicated receive buffers
- Two configurable receive FIFOs (up to 64 elements each)
- Up to 128 filter elements
- Two interrupt lines
- Power-down and wake-up support
- Timestamp counter

The following tests can be applied as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-20. CAN-FD Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
MCAN1	Software test of function using IO loopback.	Targeted toward all the functionality in MCAN. These tests can be run periodically.
MCAN2	Information redundancy techniques including end to end safing.	This is an application-level check, in which additional information (for example, the CRC of the message) is included along with the message.
MCAN3	Periodic software read back of static configuration registers	Targeted towards static configuration registers.
MCAN4	SRAM ECC	Targeted toward the SRAM used to store messages.
MCAN5 (Latent fault coverage of ECC check logic)	Software test of ECC logic.	Targeted toward latent faults in ECC check logic.
MCAN6	MCAN timeout function	Targeted toward message RAM update logic and other faults which prevents message updates.
MCAN7	MCAN timestamp function	Targeted toward message RAM update logic and other faults which prevents message updates.
WDT	Windowed watchdog event	Targeted toward interrupt logic.

5.17 Events

The event manager transfers digital events from one entity (for example, a peripheral) to another (for example, a second peripheral, the DMA, or the CPU). The event manager implements event transfer through a defined set of event publishers (generators) and subscribers (receivers) which are interconnected through an event fabric containing a combination of static and programmable routes. Events that are transferred by the event manager include:

- Peripheral event transferred to the CPU as an interrupt request (IRQ) (static event)
  - Example: ADC interrupt is sent to the CPU
- Peripheral event transferred to the DMA as a DMA trigger (DMA Event)
  - Example: UART data receives a trigger to DMA to request a DMA transfer
- Peripheral event transferred to another peripheral to directly trigger an action in hardware (generic event)
  - Example: The TIMx timer peripheral publishes a periodic event to the ADC subscriber port, and the ADC uses the event to trigger start-of-sampling

The following tests must be applied for the targeted ASIL as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-21. Events Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
ADC3	ADC trigger overflow	Covers faults in the event fabric which can result in spurious triggers to ADC.
EVENT1	Periodic software read back of static configuration registers	Targets the static configuration registers in event fabric.
GPIO3	GPIO multiple (redundant) inputs and outputs	The GPIO module can be triggered by other IPs and this mechanism, which is defined to cover faults in GPIO, can also cover faults in the event fabric which results in one of the GPIOs missing the trigger.
WDT	Windowed watchdog event	Targets faults in the event fabric which results in the trigger not propagating through the event fabric (events which are periodic in nature). Such failures can lead to a CPU program sequence getting altered and triggering a watchdog event.

5.18 IOMUX

The IOMUX manages the selection of the peripheral function used on a digital I/O. The IOMUX also provides the controls for the output driver, input path, and the wake-up logic for wakeup from SHUTDOWN mode.

The following tests must be applied for the targeted ASIL as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-22. IOMUX Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
GPIO1	Online monitoring using I/O loopback	This tests the functioning of GPIO. This test also provides coverage for the faults in the IOMUX logic and the I/O drivers.
IOMUX1	Periodic software read back of static configuration registers	Targets the static configuration registers in IOMUX.
IOMUX2	IOMUX coverage as part of other IP safety mechanisms	In general, any fault in IOMUX is covered by the safety mechanisms of the IPs.

5.19 VREF

The shared voltage reference module (VREF) in these devices contain a configurable voltage reference buffer, which allows users to supply a stable reference to on-board analog peripherals. VREF also supports bringing in an external reference for applications where higher accuracy is required. VREF features include:

- 1.4V and 2.5V user-selectable internal references
- Internal reference supports full speed ADC operation
- Support for bringing in an external reference on the VREF± device pins
- Requires a decoupling capacitor placed on the VREF± pins for proper operation

The following tests must be applied for the targeted ASIL as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-23. VREF Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
ADC4	ADC window comparator	If ADC is using the internal reference, faults in the VREF can result in ADC results being out of range. These failure modes can be detected using the window comparator in ADC.
COMP3	External pin input to COMP	If the comparator uses the internal reference, then the test which checks the comparator function also provides coverage of faults in the internal reference.
REF1	Periodic software read back of static configuration registers	Targets the static configuration registers in comparator.
REF2	VREF to ADC reference input	VREF can be tested as part of test which checks the ADC functioning by choosing to use the internal reference for the ADC test.

5.20 WWDT

The windowed watchdog timer (WWDT) can be used to supervise the operation of the device, specifically code execution. The WWDT can be used to generate a reset or an interrupt if the application software does not successfully reset the watchdog within a specified window of time. Key features of the WWDT include:

- 25-bit counter
- Programmable clock divider
- Eight software selectable watchdog timer periods
- Eight software selectable window sizes
- Support for stopping the WWDT automatically when entering a sleep mode
- Interval timer mode for applications which do not require watchdog functionality
  (Interval mode is not to be used in safety-critical applications, watchdog is used as a safety mechanism)
- A 25-bit counter with closed and open window
- A counter driven from the LFOSC (fixed 32kHz clock path) with a programmable clock divider
- Eight selectable watchdog timer periods

The following tests must be applied for the targeted ASIL as functional safety mechanisms for this module (to provide diagnostic coverage on a specific function):

Table 5-24. WWDT Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
WDT1 (latent fault coverage)	Periodic software read back of static configuration registers	Targets the static configuration registers in watchdog.
WDT2 (latent fault coverage)	WWDT counter check	This test is a test of diagnostic which checks the functioning of the counter.
WDT3 (latent fault coverage)	WWDT software test	This test is a test of diagnostic, which targets the reset or NMI generation logic.

Table 5-24. WWDT Safety Mechanisms (continued)

Safety Mechanism	Description	Faults | Failure Modes
WDT4 (latent fault coverage)	Redundant WDT	This is a test of diagnostic which covers the latent faults in the watchdog by using a redundant watchdog.

Note

The watchdog is a safety mechanism and only latent faults are considered.

5.21 CRC

The cyclical redundancy check (CRC) module provides a signature for an input data sequence. Key features of the CRC module include:

- Support for 16-bit CRC based on CRC16-CCITT
- Support for 32-bit CRC based on CRC32-ISO3309
- Support for bit reversal

Many safety mechanisms involve computing CRC on a block of data. This module can be used to do those checks efficiently (to provide diagnostic coverage on a specific function):

Table 5-25. CRC Safety Mechanisms

Safety Mechanism	Description	Faults | Failure Modes
CRC (latent fault coverage)	CRC Checker	Any fault in CRC computation is covered by the CRC check being different from the golden CRC.
CRC1 (latent fault coverage)	Periodic software readback of static configuration registers	Targets the static configuration registers in CRC.

Note

Since CRC is a safety mechanism, only latent faults are considered.

Chapter 6

MSPM0G3x0x-Q1 Management of Random Faults

For a functional safety critical development it is necessary to manage both systematic and random faults. The MSPM0G3x0x-Q1 component architecture includes many functional safety mechanisms, which can detect and respond to random faults when used correctly. This section of the document describes the architectural functional safety concept for each sub-block of theMSPM0G3x0x-Q1 component. The system integrator shall review the recommended functional safety mechanisms in the functional safety analysis report (FMEDA) in addition to this safety manual to determine the appropriate functional safety mechanisms to include in their system. The component data sheet or technical reference manual (if available) are useful tools for finding more specific information about the implementation of these features.

6.1 Fault Reporting

Internal faults are reported to the host controller through the host bus.

6.2 Functional Safety Mechanism Categories

This section includes a description of the different types of functional safety mechanisms that are applied to the design blocks of the MSPM0G3x0x-Q1 component.

The functional safety mechanism categories are defined as follows:

Component Hardware Functional Safety Mechanisms	A safety mechanism that is implemented by TI in silicon which can communicate error status upon the detection of failures. The safety mechanism may require software to enable its functionality, to take action when a failure is detected, or both.
Component Hardware and Software Functional Safety Mechanisms	A test recommended by TI which requires both, safety mechanism hardware which has been implemented in silicon by TI, and which requires software. The failure modes of the hardware used in this safety mechanisms are analyzed or described as part of the functional safety analysis or FMEDA. The system implementer is responsible for analyzing the software aspects for this safety mechanism.
Component Software Functional Safety Mechanisms	A software test recommended by TI. The failure modes of the software used in this safety mechanism are not analyzed or described in the functional safety analysis or FMEDA. For some components, TI may provide example code or supporting code for the software functional safety mechanisms. This code is intended to aid in the development, but the customer shall do integration testing and verification as needed for their system functional safety concept.
System Functional Safety Mechanisms	A safety mechanism implemented externally of this component. For example an external monitoring IC would be considered to be a system functional safety mechanism.
Test for Safety Mechanisms	This test provides coverage for faults on a safety mechanism only. It does not provide coverage for the primary function.
Alternative Safety Mechanisms	An alternative safety mechanism is not capable of detecting a fault of safety mechanism hardware, but instead is capable of recognizing the primary function fault (that another safety mechanism may have failed to detect). Alternate safety mechanisms are typically used when there is no direct test for a safety mechanism.

6.3 Description of Functional Safety Mechanisms

This section provides a brief summary of the functional safety mechanisms available on this component.

6.3.1 ADC1, COMP1, DAC1, DMA1, GPIO2, TIM2, I2C2, IOMUX1, SPI2, UART2, SYSCTL5, MCAN3, CPU4, CRC1, EVENT1, REF1, WDT1: Periodic Read of Static Configuration Registers

A periodic software read of static configuration registers is one of the methods listed in ISO26262 (Table D-4, Chapter 5). This mechanism involves the software verifying the values of static configuration registers against a known reference. One of the methods is to compute a reference CRC signature for the static configuration register values. During the course of the application, the CRC is computed periodically and compared with the reference CRC signature.

Table 6-1. Configuration Registers Associated With Different Safety Mechanisms

SM	Configuration Registers
UART2	ADDR, AMASK, CLKDIV2, CTL0, FBRD, GFCTL, IBRD, IFLS, IRCTL, LCRH, TDR, IMASK, CLKSEL, CLKDIV, PDBGCTL
SPI2	CTL0, CTL1, CLKCTL, IFLS, IMASK, CLKSEL, CLKDIV
I2C2	GFCTL, CCR, CCTR, CFIFOCTL, CSA, CTPR, PECCTL, TCTR, TFIFOCTL, TOAR, TOAR2, TIMEOUT, IMASK, CLKSEL, CLKDIV, TEST0, PDBGCTL
CPU4	NVIC.IRQEN, NVIC.IRQLVL, SYST_CSR, SYST_RVR, MPU_CTRL
DMA1	IMASK, DMATSEL, DMACTL, DMAPRIO
CRC1_latent	CRCCTRL, CRCPOLY
WDT1	WDT: WDTCTL0, WDTCTL1, IMASK, PDBGCTL IWDT: WDTCTL, WDTDBGCTL and WDTEN
EVENT1	CHANID
GPIO2	CLKOVR, CTL, DMAMASK, FASTWAKE, FILTEREN, IMASK, PDBGCTL, POLARITY, SUB0CFG, SUB1CFG, TEST0
IOMUX1	PINCM
COMP1	CTL0, CTL1, CTL2, CTL3, IMASK
TIM2	LD, PL, CTRCTL, CCACT, CCCTL, IFCTL, IMASK, CCLKCTL, CCPD, CLKDIV, CLKSEL, CPS, CTRIGCTL, FSCTL, ODIS, PDBGCTL
ADC1	CLKFREQ, CTL0, CTL1, CTL2, DEBUG1, DEBUG2, DEBUG3, DEBUG4, MEMCTL, SCOMP0, SCOMP1, TEST0, TEST1, TEST2, TEST3, TEST4, TEST5, WCHIGH, WCLOW
REF1	CTL0, CTL1, CTL2
SYSCTL5	FLASHSRAMCFG, GENCLKCFG, GENCLKEN, HFCLKCLKCFG, HSCLKCFG, HSCLKEN, MCLKCFG, PMODECFG, SYSOSCCFG, LFCLKCFG, SYSPLLCFG0, SYSPLLCFG1,
DAC1	CTL0, CTL1, CTL2, CTL3, CALCTL, CALDATA, IMASK
MCAN3	MCAN_TSCC, MCAN_TOCC, MCAN_GFC, MCAN_SIDFC, MCAN_XIDFC, MCAN_RXF0C, MCAN_RXBC, MCAN_RXF1C, MCAN_RXESC, MCAN_TXBC, MCAN_TXESC, MCAN_TXEFC

Note

Static configuration registers, are those registers whose contents are configured once and remain constant through the course of application.

6.3.2 ADC2: Software Test of Functionality

In this method, the internal DAC output is used to set up a known voltage on the ADC input channel and the other parameters of the ADC, for example, the sampling time, reference sources, and sequencer modes are set up similar to the actual application. A software trigger can be utilized to trigger the ADC. The output of the ADC

can be compared against the expected range. The test can include multiple samples to check various features of ADC being used in the application context (for example, averaging, check for offset errors, and so forth).

6.3.3 ADC3: ADC Trigger Overflow Check

If a trigger to ADC is fired, while a sample or conversion operation is in progress, the TOVIFG flag is set. This flag can be configured to generate an interrupt and appropriate action can be taken.

6.3.4 ADC4: Window Comparator

There is one window comparator unit available in the ADC that can be used to check if the input signal is within the predefined threshold values set by software. The ADC result (digital value corresponding to the input signal level) that goes into MEMRES or FIFO is what gets checked against the threshold values of the window comparator.

Based on the comparison, the window comparator can generate 3 interrupt conditions:

1. LOWIFG – Conversion result is below the low threshold (WCLOW)
2. HIGHIFG – Conversion result is above the high threshold (WCHIGH)
3. INIFG – Conversion result is in between, or equal to, the low and high thresholds

The window comparator low and high threshold values are global for all channels and the window comparison feature can be enabled for each channel, as needed, using the WINCOMP bit in the MEMCTL register.

When the ADC result data format (CTL2.DF) or resolution (CTL2.RES) configuration is changed, the window comparator threshold values are not reset by hardware and are retained as is. The software application is expected to reconfigure the threshold values as appropriate after changing the data format and resolution configuration.

6.3.5 ADC5: Test of Window Comparator

This test checks if the window comparator is working. As part of the ADC7 test, or a separate test, the ADC7 can be set so the threshold results in the ADC crossing the predefined thresholds. The test can check if the window comparator logic detects these conditions.

6.3.6 ADC6: ADC Trigger, Output Plausibility Checks

This checks the plausibility of inputs being sampled by ADC. For example, if information is related to:

- Expected bandwidth of the input signal
- Range of the signal
- Expected sampling rate

Checks can be performed to see if the readings meet these plausibility conditions. In addition to doing checks on the ADC output, the time duration from the last interrupt can be checked in software and compared against the expected ADC sampling rate. If the duration from the last interrupt is beyond the acceptable range, corrective actions can be taken.

6.3.7 OA2: Test of OA Using Internal DAC as a Driver

In this test method, the DAC output is chosen as input to OPA. The OPA is configured per the application configuration. The output of OPA can be measured using ADC.

6.3.8 OA3: ADC Monitoring of OA Output

OA outputs are connected to ADC. ADC can be used to check the operation of OA.

6.3.9 COMP2: Software Test of Comparator Using Internal DAC

In this test method, one input to the comparator is connected to DAC, the other input of comparator is connected to the internal 8-bit DAC. The result of the comparison can be checked using the comparator output monitoring.

6.3.10 COMP3: External Pin Input to COMP

In case the 12-bit DAC is unavailable, an external input can be connected to the positive terminal input and the negative terminal is fed by internal DAC8 (COMPDAC). Software can vary the DAC8 setting and check the output from COMP (for example, 0 or 1).

6.3.11 COMP4: Comparator Hysteresis

The comparator supports programmable hysteresis voltages for fast and ultra-low-power modes to avoid spurious output transitions in case of noisy input signals. This comparator support can be used as a fault avoidance measure against noise in the input.

6.3.12 COMP5: Redundant Comparator

In this method, the same signal is connected to two different comparators. The outputs from the two comparators are compared in the interrupt routine. By using two comparators to monitor the same signal, a fault in one of the comparators can be detected, even if one of the comparators is faulty, because the other comparator can still respond.

6.3.13 WDT: Windowed Watchdog Timer

TI recommends using a windowed watchdog timer (WWDT) to monitor the application program sequence. The WWDT can be used to generate a reset or an interrupt if the application software does not successfully reset the watchdog within a specified window of time. Key features of the WWDT include:

- 25-bit counter
- Programmable clock divider
- Eight software selectable watchdog timer periods
- Eight software selectable window sizes
- Support for stopping the WWDT automatically when entering a sleep mode
- Interval timer mode for applications that do not require watchdog functionality

For more details, see the WWDT chapter of the *MSPM0 G-Series 80-MHz Microcontrollers Technical Reference Manual*.

6.3.14 WDT2: WWDT Counter Check

At start-up, the counter can be configured in the interval-timer mode and the clock dividers can be configured to smaller values so that the counter overflows fast. In the interrupt routine, WDT can be reset using the RSTCTL register. If the WDT timer does not timeout in the given time, an error is flagged.

6.3.15 WDT3: WWDT Software Test

The WWDT module can be tested periodically by inducing a fault condition and checking that the fault is triggering a reset or NMI. The test can include checking that counter is operational.

6.3.16 WDT4: Redundant WDT

To obtain latent fault coverage on WDT, two WDTs can be simultaneously enabled with the same configuration. This configuration enables covering latent faults in WDT.

6.3.17 REF2: VREF to ADC Reference Input

The internal reference voltage given out from VREF is used as a reference voltage in ADC for ADC operations, and the ADC converted result is compared against the expected digital value.

6.3.18 CPU1: CPU Test Using Software Test Library

The Cortex®M0+ CPU and MPU are tested using the ARM® software test library (M0+ STL rev. r0p0).

6.3.19 CPU2: Software Test of CPU Data Buses

This test method involves writing a known value to different addresses in the memory map, reading back the values, and checking the read-back value. The peripheral region is covered when the respective static register

tests execute. Depending on the application use, the flash and SRAM regions can be tested so the address is in powers of two.

6.3.20 CPU3: Software Diversified Redundancy

In this method, the computation done by the primary function is also computed in an alternate function (different algorithm or implementation), and the results of the primary and alternate functions are compared. This provides redundancy in software and uses the same underlying hardware.

6.3.21 SYSMEM1: Software Read of Memory, DMA Write

In this method, different data values can be written to different address locations in SRAM from DMA, and the data can be verified by CPU reads (a CRC check can also be performed).

6.3.22 SYSMEM2: DMA Read from SRAM, CPU Write

Different data values can be written to different address locations within SRAM by CPU, and DMA can be configured to copy this data to another region in SRAM. The CPU can be used to verify the transferred data.

6.3.23 SYSMEM3: Parity Logic Test

In this method, any one of the bit values is flipped and a read from the same location is performed. The test checks that a parity error occurs. Using unchecked memory aperture, data bits can be written independent of parity bits. When the same location is accessed from a parity checked region, a parity error occurs. This can be used to check the functioning of parity logic.

6.3.24 SYSMEM4: Parity Protection on SRAM

The RAM in MSPM0Gx parts have parity-check bits associated with each byte of data. When RAM contents are read, a parity check is performed in hardware and compared against the stored parity bits. This mechanism can detect single-bit errors.

6.3.25 SYSMEM9: RAM Software Test

To cover the multipoint latent faults in SRAM, a software test can be executed at start-up to cover all the locations used in the application. This test gives coverage for multiple bit fails in locations not normally accessed during application run time.

Note

Only when multiple test patterns (or March C algorithms, and so forth) are used, can the DC reach medium level.

6.3.26 FLASH1: FLASH Single Error Correction, Double Error Detection Mechanism

The FLASH in MSPM0Gx parts have ECC check bits associated with data bits. When FLASH contents are read, the expected code bits are computed in hardware and compared against the stored code bits. This mechanism can correct single-bit errors and can detect double-bit errors. The address bits are also included in the code computation to cover errors in address decoding logic.

6.3.27 FLASH2: Flash CRC

This test provides increased coverage on multipoint latent faults (locations in FLASH which are accessed only on specific exception conditions that cannot be accessed during multiple on and off cycles). These locations can develop multipoint faults over time. To cover these kind of locations, a CRC test is done on the regions that are expected to be used by application.

6.3.28 FXBAR2: Periodic Software Read Back of Flash Data

In this method, a known set of test data values are stored in flash and periodically this flash data is read and compared against expected data.

6.3.29 FXBAR3: Software Test of ECC Checker Logic

In this method, a few locations in flash can have corrupted data or ECC bits. These locations can be read to check that the ECC checker logic and interrupt or NMI generation is working properly.

6.3.30 FXBAR4: Write Protection of Flash

MSPM0 MCUs implement a static write protection scheme to lock out user-defined sectors in the main flash from unintended program and erase.

6.3.31 DAC2: DAC Test Using Internal ADC as DAC Output Checker

In this test method, DAC is setup per application configuration, the output of DAC is monitored using ADC. Anytime DAC values are updated, ADC has to be triggered, allowing for the DAC settling time and the ADC output to be checked against the expected value.

6.3.32 DAC3: DAC FIFO Underrun Interrupt

The FIFO in DAC has a built-in hardware check to detect FIFO underflows and set a flag. This can be used to check for unexpected runtime errors, which causes a FIFO underflow condition.

6.3.33 DMA2: Software Test of DMA Function

In this test mechanism, one of the DMA channels is dedicated to diagnostic test. This channel can be configured to do transfers of known data content from a fixed source (SRAM or FLASH) to a fixed destination (SRAM or CRC engine). Periodically the diagnostic channel can be triggered in software and the proper transfer of data can be checked in software.

6.3.34 DMA3: Software DMA Channel Test

In this method, DMA channels in use are periodically triggered after changing the source address to flash and destination address to SRAM. Known data is transferred from flash to SRAM and the data in SRAM is compared to the data in flash. In addition, a timer can be set up to monitor completion within a reasonable time.

6.3.35 DMA4: CRC Check of the Transferred Data

In this method, on a DMA transfer-done interrupt, data coherency can be checked. The data packet can include an expected CRC value. In software, a CRC is computed and checked against the expected CRC.

Note

Medium coverage can be claimed under the following circumstances:

1. Medium overall coverage of failure modes in data transmission; a Hamming distance of three or more.
2. A CRC value for message information is embedded in message; with a CRC size of 8 bits and a 0x97 polynomial results in a Hamming distance of four for a data length of less than 119 bits. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value (typically used in a LIN bus).
3. A CRC value for message information and message ID is embedded in the message; with a CRC size of 10 bits and a 0x319 polynomial results in a Hamming distance of four for a data length of less than 501 bits. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value.
4. A CRC value for message information and message ID is embedded in the message; with a CRC size of 15 bits and a 0x4599 polynomial results in a Hamming distance of five for a data length of less than 127 bits. Additionally, burst errors of a length up to 15 can be detected. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value (as used in CAN).
5. A CRC value for message information is embedded in the message, with a CRC size of 24 bits and a 0x5D6DCB polynomial results in a Hamming distance of the CRC of six for a data length of less than, or equal to, 248 bytes and a Hamming distance of the CRC of four for a data length of greater than 248 bytes. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value (as used in FlexRay™ for the frame CRC).
6. A CRC value for message header (including the message ID) is embedded in the message; with a CRC size of 11 bits and a 0x385 polynomial results in a Hamming distance of six for a data length of less than, or equal to, 20 bits. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value (as used in FlexRay™ for the header CRC).

6.3.36 GPIO1: GPIO Test Using Pin I/O Loopback

In this test method, the GPIO pin is setup with a known value and the status of the pin can be read back using the DIN register. The loopback happens at the I/O driver, and hence, can read the pin status. This method can be applied when the GPIO is configured as an output.

6.3.37 GPIO3: GPIO Multiple (Redundant) Inputs/Outputs

If GPIO is controlling critical functions, the same outputs can be generated on two or more pins and an external function can be controlled based on majority voting. In the event a trigger mechanism (triggers can be from other peripherals, like GP timers, and so forth) is used to change the state of the GPIO pins, one subscriber can be used for one set of pins and another subscriber can be used for the other set of pins.

In the case of critical inputs being read through the GPIO, the same signal can be connected to two, or more, pins and the values that are read on the different pins can be compared to detect differences.

Note

The redundant pins must not be adjacent to prevent any coupling. TI recommends that the redundant pins are from two different GPIO instances.

6.3.38 TIM1: Test for PWM Generation

In this test, a second timer can be used to check that the PWM signal properties generate properly. To make this check, the PWM output from the main timer is looped to another TIMER. The measuring timer can then be then used to measure the pulse width and period of the PWM waveform.

Note

Use the same clock for both timers.

6.3.39 TIM3: Test for Fault Generation

In this method, when the timer is idle, the polarity of the fault pin can be changed (FCTL.FSENEXTx) to the alternate value. This triggers a fault that can be checked by reading the status register. This method can be used to check if the fault detection circuit is working as expected.

6.3.40 TIM4: Fault Detection to Take the PWMs to Safe State

If the timer outputs are driving power stages, the overcurrent (undercurrent) or overvoltage (undervoltage) signals from the power stage can be connected to fault pins. These fault signals can be monitored by the timer. The timer can be configured to take the output to a safe state by appropriately configuring the fault entry action. For more details, refer to the TIMx chapter of *MSPM0 G-Series 80MHz Microcontrollers Technical Reference Manual* .

6.3.41 TIM5: Input Capture on Two or More Timer Instances

In the case where timing parameters of safety-critical inputs are measured, two or more pins can be connected to the same signal (redundancy), and the measured parameters (in different timer instances) can be compared to determine if the parameters are within the acceptable range. Input capture must be done on different timer instances. This covers faults in the clock divider as the measured timing parameters deviate from the expected values on the second instance.

6.3.42 TIM6: Timer Period Monitoring

The timer generates periodic interrupts. In the interrupt routine, the timer period can be monitored to check if there is any deviation.

6.3.43 I2C1: Software Test of I^2C Function Using Internal Loopback Mechanism

The I^2C modules can be placed into an internal loopback mode for diagnostic, or debug work, by setting the LPBK bit in the I^2C controller configuration I2Cx.MCR register. In loopback mode, the SDA and SCL signals from the controller part of the I^2C are tied to the SDA and SCL signals of the target part of the I^2C module to allow internal testing of the device without having to connect the I/Os.

This loopback mechanism can be used to transmit known data from transmit to receive. The I^2C configuration can be similar to the application configuration, regarding bit rate, FIFO usage, and so forth. The completion of the test can be timed to be within expected limits to detect any faults in the bit rate timing.

Note

The loopback does not cover failures of I/O pins. These failures are covered by other safety mechanisms.

6.3.44 I2C3, SPI4, UART3, MCAN2: Information Redundancy Techniques Including End-to-End Safing

End-to-end safing includes a combination of techniques like information redundancy (adding CRC checks to the payload, for example), redundant transmissions, time diversity in transmissions, and so forth. Most commonly, checksums are added to the payload section of a transmission to verify the correctness of a transmission. The checksums, sequence counter, and timeout expectation (or time stamp) are applied, in addition to any protocol-level parity and checksums. As these parameters are generated and evaluated by the software, at either end of the communication, the whole communication path is safed, resulting in end-to-end safing.

Note

Medium coverage using CRC can be claimed under following circumstances:

1. Medium overall coverage of failure modes in data transmission; a Hamming distance of three or more.
2. A CRC value for message information is embedded in message; with a CRC size of 8 bits and a 0x97 polynomial results in a Hamming distance of four for a data length of less than 119 bits. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value (typically used in a LIN bus).
3. A CRC value for message information and message ID is embedded in the message; with a CRC size of 10 bits and a 0x319 polynomial results in a Hamming distance of four for a data length of less than 501 bits. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value.
4. A CRC value for message information and message ID is embedded in the message; with a CRC size of 15 bits and a 0x4599 polynomial results in a Hamming distance of five for a data length of less than 127 bits. Additionally, burst errors of a length up to 15 can be detected. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value (as used in CAN).
5. A CRC value for message information is embedded in the message, with a CRC size of 24 bits and a 0x5D6DCB polynomial results in a Hamming distance of the CRC of six for a data length of less than, or equal to, 248 bytes and a Hamming distance of the CRC of four for a data length of greater than 248 bytes. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value (as used in FlexRay™ for the frame CRC).
6. A CRC value for message header (including the message ID) is embedded in the message; with a CRC size of 11 bits and a 0x385 polynomial results in a Hamming distance of six for a data length of less than, or equal to, 20 bits. The transmitter includes the CRC value mentioned and the receiver confirms the data after calculating and comparing the CRC value (as used in FlexRay™ for the header CRC).

6.3.45 I2C4, SPI5, UART4: Transmission Redundancy

This is an application-level diagnostic where information is transferred several times in sequence using the same module instance and then compared. When the same data path is used for duplicate transmissions, transmission redundancy is only useful for detecting transient faults. The diagnostic coverage can be improved by sending inverted data during the redundant transmission.

6.3.46 I2C5, UART5: Timeout Monitoring

Periodic safety messages (an application-level technique) can be transmitted and received. When a safety message is not received in the expected time interval, the event can indicate loss of data communication. This monitoring is done continuously in the application.

6.3.47 I2C6: Test of CRC Function

I2C has a packet error check mechanism in the SMBUS mode. To check this function, messages are sent with corrupted data and a check is made to see if the CRC logic flags an error.

6.3.48 I2C7: Packet Error Check in SMBUS Mode

When I2C is used in SMBUS mode, the packet error check is done in hardware as part of the protocol. This is a fault-avoidance measure.

6.3.49 IOMUX2: IOMUX Coverage as Part of Other IP Safety Mechanisms

The majority of faults in the IOMUX logic are covered by the safety mechanisms of the blocks interfacing with the IOMUX (refer to Table 6-2).

Table 6-2. IOMUX Coverage

Signal Category	Diagnostics That Address IOMUX
Serial ports	I2C1, I2C3, SPI1, SPI4, SPI5, UART1, UART3, UART4, UART5, MCAN1, MCAN2
Timers	TIM1, TIM3, TIM4, TIM5
GPIO static pins	GPIO1, GPIO3
GPIO interrupts (periodic)	WDT
Clock inputs	SYSCTL1, SYSCTL3, SYSCTL9

6.3.50 SPI1: Software Test of SPI Function

The SPI can be placed into an internal loopback in controller mode by setting the LBM bit in the SPI.CTL1 register. In loopback mode, data transmitted on the TX output is received on the RX input. FIFO related faults can be diagnosed using this test. For checking the clock setting, the test execution time can be timed using timers.

Note

The loopback does not cover failures of I/O pins. These failures are covered by other safety mechanisms.

6.3.51 SPI3: SPI Periodic Safety Message Exchange

An application-level check can be added to periodically send and receive a test message when SPI is in peripheral mode. In software, a timeout mechanism must be implemented to cover this.

6.3.52 UART1: Software Test of UART Function

The UART can be placed into an internal loopback mode for diagnostic, or debug work, by setting the LBE bit in the UART.CTL0 register. In loopback mode, data transmitted on the TXD output is received on the RXD input. Data received on the RXD IO pin is ignored when loopback is enabled. A timer can be used to check if the communication completed within the expected amount of time.

Note

The loopback does not cover failures of I/O pins. These failures are covered by other safety mechanisms.

6.3.53 UART6: UART Error Flags

UART has error flags for frame error, break error, parity error, and overrun error.

6.3.54 UART7: UART Glitch filter

A set of analog and digital glitch filters are present on the RX line. These filters can be used as a fault-avoidance measure for the glitches on the RX line.

6.3.55 SYSCTL1: MCLK Monitor

A digital clock monitor is provided to verify that MCLK is active. An MCLK fault is asserted by the MCLK monitor if there is no MCLK activity within a period of 1 to 12 LFCLK cycles. An MCLK fault is always considered fatal to the system and generates a BOOTRST.

The MCLK monitor can be enabled once LFCLK is configured and running. To enable the MCLK monitor, set the MCLKDEADCHK bit in the MCLKCFG register in SYSCTL. When enabled, the MCLK monitor runs in all operating modes except for STANDBY1 and SHUTDOWN.

6.3.56 SYSCTL2: HFCLK Start-Up Monitor

The HFXT takes time to start after being enabled. A start-up monitor is provided to indicate to the application software if the HFXT has successfully started, at which point the HFCLK can be selected to source a variety

of system functions. The HFCLK start-up monitor also supports checking the HFCLK_IN digital clock input for a clock stuck fault.

When HFXT is started, or the HFCLK_IN is selected as the HFCLK source, the HFCLKGOOD and HFCLKOFF bits in the CLKSTATUS register in SYSCTL are cleared.

6.3.57 SYSCTL3: LFCLK Monitor

A low-power analog circuitry clock monitor is provided to verify that LFCLK is running when LFCLK is not sourced internally (for example, in cases when LFCLK is sourced from LFXT or LFCLK_IN and not from LFOSC). The LFCLK monitor is only intended to check for clock stuck faults. The monitor is not intended to be used to verify that the frequency of LFCLK is within a specific tolerance.

6.3.58 SYSCTL6: SYSPLL Start-Up Monitor

The SYSPLL takes time to start and settle after being enabled. A start-up monitor is provided to indicate to the application software if the SYSPLL has successfully started, at which point the clock outputs from the SYSPLL can be selected to source a variety of system functions.

When the SYSPLL is started, the SYSPLLGOOD and SYSPLLOFF bits in the CLKSTATUS register in SYSCTL are cleared. After the start-up and settling time has expired, the SYSPLL status is tested. If the SYSPLL started successfully, the SYSPLL start-up monitor asserts the SYSPLLGOOD bit in the CLKSTATUS register and the SYSPLLGOOD interrupt is also asserted. If the SYSPLL did not start within the specified time, the SYSPLLOFF bit is set, indicating that the SYSPLL was dead at start-up.

6.3.59 SYSCTL8: Brownout Reset (BOR) Supervisor

The brownout reset (BOR) supervisor monitors the external supply (VDD) and asserts or deasserts a BOR violation to SYSCTL.

6.3.60 SYSCTL9: FCC Counter Logic to Calculate Clock Frequencies

The frequency clock counter (FCC) enables flexible in-system testing and calibration of a variety of oscillators and clocks on the device. The FCC counts the number of clock periods seen on the selected source clock within a known trigger period (derived from a secondary reference source) to provide an estimation of the frequency of the source clock.

6.3.61 SYSCTL10: External Voltage Monitor

Use an external voltage monitor on VCORE PAD to monitor the LDO output.

6.3.62 SYSCTL11: Boot Process Monitor

In the event of a boot fail during execution of the boot configuration routine (BCR), SYSCTL asserts a BOOTRST to re-attempt a successful boot. A boot fail can be caused by any of the following:

1. Boot configuration data integrity error (this can be used for BOOTROM/ FLASH)
2. Device trim integrity error (this can be used for FLASH)
3. BCR timeout (BCR takes significantly longer than expected to complete for any other reason, this can be used for BOOTROM)

6.3.63 SYSCTL14: Brownout Voltage Monitor

The brownout circuit can be configured to monitor the external supply (VDD) above the BOR reset level. If tripped, the monitor generates a non-maskable interrupt (NMI) event and reconfigures the BOR circuit to be a BOR supervisor reset. This allows software to either warn the user of a depleting battery, or initiate a graceful shutdown of the system application, before the BOR circuit issues a BOR supervisor reset.

6.3.64 SYSCTL15: External Voltage Monitor

An external voltage supervisor can be used to monitor the power supplies (main supply and the internal LDO output).

6.3.65 SYSCTL16: External Watchdog Timer

An external watchdog monitor can be employed to check (sanity check) the working of MCU. In case watchdog monitor flags an error, the system software can take the required action.

6.3.66 MCAN1: Software test of function using I/O Loopback

The MCAN module can be set into internal loopback mode by programming MCAN_TEST.LBCK and MCAN_CCCR.MON bits to 1. The internal loopback mode is used for a hot self-test. The hot self-test allows the MCAN module to be tested without affecting a running CAN system connected to the TX and RX pins. In this mode, the RX pin is disconnected from the MCAN module and the TX pin is held recessive.

6.3.67 MCAN4: SRAM ECC

The message RAM in the MCAN module stores computed check bits (ECC bits) for each word of data stored. An ECC memory always protects against single-bit errors in the data. Address decode logic for the memory is not covered by the memory ECC logic. There is an ECC aggregator module inside the module that aggregates status from the ECC memories into a single interrupt to the host. The aggregator also supports software readable status of ECC single- and double-bit errors and associated info such as RAM address and data bit (or bits) that are in error.

6.3.68 MCAN5: Software Test of ECC Check Logic

Testing the functionality of the ECC detection logic is possible by forcing an ECC error into the data output from the memory and checking if the ECC detection logic reports an error. A shared module interface, which is referred to as an ECC aggregator, provides the system integrator with access to configure the logic and force errors. Reporting of forced errors uses the same mechanism that reports unforced errors. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator

6.3.69 MCAN6: MCAN Timeout Function

MCAN implements a timeout counter that is programmed during the INIT phase for the module. The timeout function can be continuous (preset by writing to TOCC.TOP on a periodic basis before the timeout function expires) or associated with Tx, Rx0, or Rx1 FIFOs (a FIFO empty presets the counter and first push starts the down counting). A CAN system implementing a periodic messaging can use the timeout diagnostic to ascertain the presence of a system heart beat.

6.3.70 MCAN7: MCAN Timestamp Function

MCAN implements a timestamp for received and transmitted messages. An external timestamp counter is required for CAN FD messages. The timestamp counter provided by MCAN is equipped with a prescaler for tradeoff between resolution and wraparound period. The timestamp counter value is stored in the message buffer for each transmitted or received message. Software can perform sanity checks on messages to determine if the messages have been sent in the order expected by the system as a diagnostic. For example, multiple messages with the same timestamp (taking into consideration the wraparound time) are not expected as the CAN protocol can carry one message at a time. End-to-end safing that includes numbering the messages can be used to indicate linear incrementing timestamps that software can verify.

6.3.71 CRC: CRC Checker

The hardware CRC module can be used to speed up some of the safety mechanisms involving CRC computation. Any fault within the CRC logic is detected when the CRC signature is incorrect.

6.3.72 EVENT2: Interrupt Connectivity Check

The interrupt connectivity between the CPU and the peripheral blocks can be checked by writing to the ISET (interrupt set) register in the peripheral block and checking that the corresponding flag is set in NVIC. Once the check is complete, the interrupt flag can be cleared by writing to the ICLR register in the peripheral. The flag in the NVIC can also be cleared by software.

6.3.73 Safety Mechanisms Covering PIN Failures

Package pin-related failures are covered by one or more of the safety mechanisms described. There are two different types of pins.

- GPIOs: These pins multiplex the signals from different IPs; depending on the application context, one of the possible signals can be chosen.
- Dedicated pins: These pins can be power or reset pins.

In the case of GPIO pins, depending on the signal chosen in the context of the application, the safety mechanism applicable to the corresponding IP covers the pin failures. For cases where the pins are reset or power pins, the safety mechanisms covering reset or power are used. For more information, refer to the safety analysis report.

6.3.74 Safety Mechanisms Covering Common Cause Failures

Safety analyses to assess common cause failures and dependent failures were performed, and the table below summarizes the safety mechanisms applicable to such failures.

Table 6-3. Safety Mechanisms Related to Common Cause Failures

Failure Applicable	Safety Mechanism
MCLK failure	SYSCTL1
VDD failure	SYSCTL14, SYSCTL15
Reset failure	SYSMEM1, SYSMEM2
Failure on common access bus between SRAM parity checker and SRAM	SYSMEM1, SYSMEM2
Common configuration bus failure	Typically covered by static configuration register tests.

Appendix A
Summary of Recommended Functional Safety Mechanism Usage

Appendix A summarizes the functional safety mechanisms present in hardware or recommend for implementation in software or at the system level as described in Chapter 5. Table A-1 describes each column in Table A-2 and gives examples of what content can appear in each cell.

Table A-1. Legend of Functional Safety Mechanisms

Functional Safety Mechanism	Description
TI Safety Mechanism Unique Identifier	A unique identifier assigned to this safety mechanism for easier tracking.
Safety Mechanism Name	The full name of this safety mechanism.
Safety Mechanism Category	**Safety Mechanism** - This test provides coverage for faults on the primary function. It may also provide coverage on another safety mechanism. These tests also provide coverage of primary function multiple-point faults if it covers permanent faults. **Test for Safety Mechanism** - This test provides coverage for faults of a safety mechanism only. It does not provide coverage on the primary function. **Fault Avoidance** - This is typically a feature used to improve the effectiveness of a related safety mechanism.
Safety Mechanism Type	Can be either hardware, software, a combination of both hardware and software, or system. See Section 6.2 for more details.
Safety Mechanism Operation Interval	The timing behavior of the safety mechanism with respect to the test interval defined for a functional safety requirement / functional safety goal. Can be either continuous, or on-demand. **Continuous** - the safety mechanism constantly monitors the hardware-under-test for a failure condition. **Periodic or On-Demand** - the safety mechanism is executed periodically, when demanded by the application. These tests have to be performed within the FTTI (Fault tolerant time interval) determined by the application. The test of diagnostics have to be performed once within the multiple-point fault detection interval. This includes Built-In Self-Tests that are executed one time per drive cycle or once every few hours.
Test Execution Time	Time period required for the safety mechanism to complete, not including error reporting time. Note: Certain parameters are not set until there is a concrete implementation in a specific component. When component specific information is required, the component data sheet must be referenced. Note: For software-driven tests, the majority contribution of the Test Execution Time is often software implementation-dependent.
Action on Detected Fault	The response that this safety mechanism takes when an error is detected. Note: For software-driven tests, the Action on Detected Fault can depend on software implementation.

Table A-1. Legend of Functional Safety Mechanisms (continued)

Functional Safety Mechanism	Description
Time to Report	Typical time required for safety mechanism to indicate a detected fault to the system. Note: For software-driven tests, the majority contribution of the Time to Report is often software implementation-dependent.
Diagnostic Evaluation	Basis of diagnostic coverage evaluation: • Injection - Diagnostic coverage evaluation is based on TI-driven fault injection campaign • Insertion - Diagnostic coverage evaluation is based on customer-driven fault insertion campaign • Calculation - Diagnostic coverage evaluation is based on analytical or independent experimental data • Estimation - Diagnostic coverage evaluation is based on expert judgment Note: Insertion refers to the ability of a system integrator to inject faults using external test fixtures, e.g. an Ethernet CRC check can be tested by sending in Ethernet packets with a bad CRC. Insertion is not expected to replace functional verification performed by TI. The intention of insertion is first, that the system can verify that the diagnostic is working and the system response to the diagnostic. The second is that a system integrator can, if necessary, measure overall diagnostic effectiveness in the context of their system. For example, the overall diagnostic coverage for methods protecting Ethernet can be measured at the system level by having the entire solution (hardware + software) running and sending in bad packets. The customer can choose to perform these techniques to update the DC values with more accurate numbers pertaining to their use cases. For these diagnostic where insertion is used as DC evaluation method, TI provides the DC coverage based on assumed safety use case in a SEOOC development.
Diagnostic Detection Capability	• Permanent - Diagnostic is capable of detecting permanent faults and runs within FTTI. The diagnostic is also capable of detecting primary function multiple-point faults. • Transient - Diagnostic is capable of detecting transient faults and runs within FTTI. • Permanent /Transient - Diagnostic is capable of detecting permanent and transient faults and runs within FTTI. • Latent - Diagnostic is capable of detecting multiple-point latent faults and runs within MPFTTI.

Table A-2. Summary of Safety Features and Diagnostics

TI Safety Mechanism Unique Identifier	Safety Mechanism Name	Safety Mechanism Category	Safety Mechanism Type	Safety Mechanism Operation Interval	Test Execution Time	Action on Detected Fault	Time to Report	Diagnostic Evaluation	Diagnostic Detection Capability
ADC1	Periodic Software Read Back of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
ADC2	ADC Software Test of Functionality	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
WDT	Windowed Watchdog Event	Safety Mechanism	Hardware + Software	Continuous	Application Dependent	Reset the Device	<1µs	Estimation	Permanent and Transient
ADC3	ADC Trigger Overflow	Safety Mechanism	Hardware	Continuous	Application Dependent	Generate an interrupt	<100 bus clock cycles	Estimation	Permanent and Transient
ADC4	ADC Window Comparator	Safety Mechanism	Hardware	Continuous	Application Dependent	Generate an interrupt	<100 bus clock cycles	Estimation	Permanent and Transient

Table A-2. Summary of Safety Features and Diagnostics (continued)

TI Safety Mechanism Unique Identifier	Safety Mechanism Name	Safety Mechanism Category	Safety Mechanism Type	Safety Mechanism Operation Interval	Test Execution Time	Action on Detected Fault	Time to Report	Diagnostic Evaluation	Diagnostic Detection Capability
ADC5	Test of Window Comparator	Test for Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Latent
ADC6	ADC Trigger or Output Plausibility Check	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
COMP1	Periodic Software Read Back of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
COMP2	DAC to COMP Loopback	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
COMP3	External Pin Input to COMP	Safety Mechanism	System Level Diagnostic	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
COMP4	Comparator Hysteresis	Fault Avoidance	Hardware	Continuous	Application Dependent	Refer to Section 4.2	N/A	N/A	N/A
COMP5	Redundant Comparator	Fault Avoidance	Hardware + Software	Continuous	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
CPU1	ARM® Software Test Library	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Injection	Permanent
CPU2	Write or Read Back of Data to Different Regions of Memory to Detect Faults in the Bus Interconnect Components	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
SYSCTL11	Boot Process Timeout	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent		10 bus clock cycles	Estimation	Permanent and Transient
CPU3	Software Diversified Redundancy	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
CPU4	Periodic Software Read Back of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
CRC	CRC Checker	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
CRC1	Periodic Software Read Back of Static Configuration Registers	Test for Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Latent

Table A-2. Summary of Safety Features and Diagnostics (continued)

TI Safety Mechanism Unique Identifier	Safety Mechanism Name	Safety Mechanism Category	Safety Mechanism Type	Safety Mechanism Operation Interval	Test Execution Time	Action on Detected Fault	Time to Report	Diagnostic Evaluation	Diagnostic Detection Capability
DAC1	Periodic Software Read Back of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
DAC2	DAC to ADC Loopback	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
DAC3	FIFO Underrun interrupt	Safety Mechanism	Hardware	Continuous	Application Dependent	Generate an interrupt	<100 bus clock cycles	Insertion	Permanent and Transient
DMA1	Periodic Software Read Back of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
DMA2	Software DMA Transfer Test	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
DMA3	Software DMA Channel Test	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
DMA4	CRC Check of the Transferred Data	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
EVENT1	Periodic Software Readback of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
GPIO3	GPIO Multiple (Redundant) Inputs/ Outputs	Safety Mechanism	Hardware + Software	Continuous	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
FXBAR2	Periodic Software Read Back of Flash Data	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent
FLASH1	Flash ECC Checker	Safety Mechanism	Hardware	Continuous	Application Dependent	Generate an interrupt	<100 bus clock cycles	Calculation	Permanent and Transient
FLASH2	Flash CRC	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Latent
FXBAR3	Software Test of ECC Checker Logic	Test for Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Latent
FXBAR4	Write Protection of Flash	Safety Mechanism	Hardware	Continuous	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
GPIO1	Online Monitoring Using I/O Loopback	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
GPIO2	Periodic Software Readback of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient

Table A-2. Summary of Safety Features and Diagnostics (continued)

TI Safety Mechanism Unique Identifier	Safety Mechanism Name	Safety Mechanism Category	Safety Mechanism Type	Safety Mechanism Operation Interval	Test Execution Time	Action on Detected Fault	Time to Report	Diagnostic Evaluation	Diagnostic Detection Capability
TIM1	Test for Basic PWM Generation	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
TIM2	Periodic Software Read Back of IP Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
TIM3	Test for Fault Generation	Test for Safety Mechanism	System	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Latent
TIM4	Fault Detection to Take the PWMs to Safe State	Safety Mechanism	Hardware + Software	Continuous	Application Dependent	Refer to Section 4.2	<100 bus clock cycles	Estimation	Permanent and Transient
TIM5	Input Capture on Two or More Timer Instances	Safety Mechanism	Hardware	Continuous	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
TIM6	Timer Period Monitoring	Safety Mechanism	Hardware + Software	Continuous	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
I2C1	Software Test of Function Using I/O Loopback	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
I2C2	Periodic Software Read Back of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
I2C3	Information Redundancy Techniques Including End-to-End Safing	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
I2C4	Transmission Redundancy	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Transient
I2C5	Timeout Monitoring	Safety Mechanism | Test of Safety Mechanism	Software	Continuous	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
I2C6	Test of CRC Function	Test for Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Latent
I2C7	Packet Error Check in SMBUS Mode	Fault Avoidance	N/A	Continuous	Application Dependent	Refer to Section 4.2	N/A	N/A	N/A
IOMUX1	Periodic Software Readback of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
IOMUX2	IOMUX Coverage as Part of Other IP Safety Mechanisms.	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent

Table A-2. Summary of Safety Features and Diagnostics (continued)

TI Safety Mechanism Unique Identifier	Safety Mechanism Name	Safety Mechanism Category	Safety Mechanism Type	Safety Mechanism Operation Interval	Test Execution Time	Action on Detected Fault	Time to Report	Diagnostic Evaluation	Diagnostic Detection Capability
MCAN1	Software test of function using I/O loopback	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
MCAN2	Information Redundancy Techniques Inclusing End-to-End Safing	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
MCAN3	Periodic Software Read Back of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
MCAN4	SRAM ECC	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Generate an Interrupt	< 100 bus clock cycles	Calculation	Permanent and Transient
MCAN5	Software Test ECC Logic	Test of Safety Mechanism	Hardware + Software	Continuous	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Latent
MCAN6	Timeout on FIFO Activity	Safety Mechanism	Hardware + Software	Continuous	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
MCAN7	Timestamp Consistency Checks	Safety Mechanism	Hardware / Software	Continuous	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
SPI1	Software Test of Function Using I/O Loopback	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
SPI2	Periodic Software Read Back of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
SPI3	SPI Periodic Safety Message Checks	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
SPI4	Information Redundancy Techniques Including End-to-End Safing	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
SPI5	Transmission Redundancy	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Transient
SYSCTL1	MCLK Monitor	Safety Mechanism	Hardware + Software	Continuous	Application Dependent	Reset the Device	<1µs	Estimation	Permanent and Transient
SYSCTL2	HFCLK Start-up Monitor	Fault Avoidance	Hardware	Periodic or On-Demand	Application Dependent	N/A	N/A	N/A	N/A
SYSCTL3	LFCLK Monitor	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Generate an interrupt	<100 bus clock cycles	Estimation	Permanent and Transient

Table A-2. Summary of Safety Features and Diagnostics (continued)

TI Safety Mechanism Unique Identifier	Safety Mechanism Name	Safety Mechanism Category	Safety Mechanism Type	Safety Mechanism Operation Interval	Test Execution Time	Action on Detected Fault	Time to Report	Diagnostic Evaluation	Diagnostic Detection Capability
SYSCTL5	Periodic Software Read Back of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
SYSCTL6	SYSPLL Startup monitor	Fault Avoidance	N/A	Periodic or On-Demand	Application Dependent	Refer to Section 4.2			
SYSCTL8	Brownout Reset (BOR) Supervisor	Safety Mechanism	Hardware	Periodic or On-Demand	Application Dependent	Generate an interrupt	<100 bus clock cycles	Estimation	Permanent
SYSCTL9	Clock Frequency Measurement	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
SYSCTL10	External Voltage Monitor	Safety Mechanism	System Level Diagnostic	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent
SYSCTL14	Brownout Voltage Monitor	Safety Mechanism	Hardware	Periodic or On-Demand	Application Dependent	Reset the Device	<1µs	Estimation	Permanent
SYSCTL15	External Voltage Supervisor	Safety Mechanism	System Level Diagnostic	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
SYSCTL16	External Watchdog Timer	Safety Mechanism	System Level Diagnostic	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
SYSMEM1	Software Read of Memory DMA	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
SYSMEM2	Software Read of Memory CPU	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
SYSMEM8	ECC Logic test	Test for Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Latent
SYSMEM7	RAM ECC	Safety Mechanism	Hardware	Continuous	Application Dependent	Generate an interrupt	<100 bus clock cycles	Calculation	Permanent and Transient
SYSMEM9	RAM Software Test	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Latent
UART1	Software Test of Function Using I/O Loopback	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
UART2	Periodic Software Read Back of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
UART3	Information Redundancy Techniques Including End-to-End Safing	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent and Transient
UART4	Transmission Redundancy	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Transient

Table A-2. Summary of Safety Features and Diagnostics (continued)

TI Safety Mechanism Unique Identifier	Safety Mechanism Name	Safety Mechanism Category	Safety Mechanism Type	Safety Mechanism Operation Interval	Test Execution Time	Action on Detected Fault	Time to Report	Diagnostic Evaluation	Diagnostic Detection Capability
UART5	Timeout Monitoring	Safety Mechanism | Test of Safety Mechanism	Software	Continuous	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
UART6	UART Error Flags	Safety Mechanism	Hardware	Continuous	Application Dependent	Generate an interrupt.	<100 bus clock cycles	Estimation	Permanent and Transient
UART7	UART Glitch Filter	Fault Avoidance	Hardware	Continuous	Application Dependent	N/A	N/A	N/A	N/A
REF1	Periodic Software Read Back of Static Configuration Registers	Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Permanent and Transient
REF2	VREF to ADC Reference Input	Safety Mechanism	Hardware + Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Permanent
WDT1	Periodic Software Read Back of Static Configuration Registers	Test for Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Estimation	Latent
WDT2	WWDT Counter Check	Test for Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Latent
WDT3	WWDT Software Test	Test for Safety Mechanism	Software	Periodic or On-Demand	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Latent
WDT4	Redundant Watchdog	Test for Safety Mechanism	Hardware + Software	Continuous	Application Dependent	Refer to Section 4.2	Application Dependent	Insertion	Latent

Appendix B
Distributed Developments

A Development Interface Agreement (DIA) is intended to capture the agreement between two parties towards the management of each party's responsibilities related to the development of a functional safety system. TI functional safety components are typically designed for many different systems and are considered to be Safety Elements out of Context (SEooC) hardware components. The system integrator is then responsible for taking the information provided in the hardware component safety manual, safety analysis report and safety report to perform system integration activities. Because there is no distribution of development activities, TI does not accept DIAs with system integrators.

TI functional safety components are products that TI represents, promotes or markets as helping customers mitigate functional safety related risks in an end application and/or as compliant with an industry functional safety standard or FS-QM. For more information about TI functional safety components, go to TI Functional Safety.

B.1 How the Functional Safety Lifecycle Applies to TI Functional Safety Products

TI has tailored the functional safety lifecycles of ISO 26262 and IEC 61508 to best match the needs of a functional Safety Element out of Context (SEooC) development. The functional safety standards are written in the context of the functional safety systems, which means that some requirements only apply at the system level. Since TI functional safety components are hardware or software components, TI has tailored the functional safety activities to create new product development processes for hardware and for software that makes sure state-of-the-art techniques and measures are applied as appropriate. These new product development processes have been certified by third-party functional safety experts. To find these certifications, go to TI Functional Safety.

B.2 Activities Performed by Texas Instruments

The TI functional safety products are hardware components developed as functional Safety Elements out of Context. As such, TI's functional safety activities focus on those related to management of functional safety around hardware component development. System level architecture, design, and functional safety analysis are not within the scope of TI activities and are the responsibility of the customer. Some techniques for integrating the SEooC safety analysis of this hardware component into the system level can be found in ISO 26262-11.

Table B-1. Activities Performed by Texas Instruments Versus Performed by the Customer

Functional Safety Lifecycle Activity[1]	TI Execution	Customer Execution
Management of functional safety	Yes	Yes
Definition of end equipment and item	No	Yes
Hazard analysis and risk assessment (of end equipment/ item)	No	Yes
Creation of end equipment functional safety concept	No. Assumptions made for internal development.	Yes
Allocation of end equipment requirements to sub-systems, hardware components, and software components	No. Assumptions made for internal development.	Yes

Table B-1. Activities Performed by Texas Instruments Versus Performed by the Customer (continued)

Functional Safety Lifecycle Activity[1]	TI Execution	Customer Execution
Definition of hardware component safety requirements	Yes	No
Hardware component architecture and design execution	Yes	No
Hardware component functional safety analysis	Yes	No
Hardware component verification and validation (V&V)	V&V executed to support internal development.	Yes
Integration of hardware component into end equipment	No	Yes
Verification of IC performance in end equipment	No	Yes
Selection of safety mechanisms to be applied to IC	No	Yes
End equipment level verification and validation	No	Yes
End equipment level functional safety analysis	No	Yes
End equipment level functional safety assessment	No	Yes
End equipment release to production	No	Yes
Management of functional safety issues in production	Support provided as needed	Yes

(1) For component technical questions, ask our TI E2E™ support experts.

B.3 Information Provided

Texas instruments has summarized what it considers the most critical functional safety work products that are available to the customer either publicly or under a nondisclosure agreement (NDA). NDAs are required to protect proprietary and sensitive information disclosed in certain functional safety documents.

Table B-2. Product Functional Safety Documentation

Deliverable Name	Contents
Functional Safety Product Preview	Overview of functional safety considerations in product development and product architecture. Delivered ahead of public product announcement.
Functional Safety Manual	User guide for the functional safety features of the product, including system level assumptions of use.
Functional Safety Analysis Report	Results of all available functional safety analysis documented in a format that allows computation of custom metrics.
Functional Safety Report[1]	Summary of arguments and evidence of compliance to functional safety standards. References a specific component, component family, or TI process that was analyzed.
Assessment Certificate[1]	Evidence of compliance to functional safety standards. References a specific component, component family, or TI process that was analyzed. Provided by a 3rd party functional safety assessor.

(1) When an Assessment Certificate is available for a TI functional safety product, the Functional Safety Report may not be provided. When a Functional Safety Report is provided, an Assessment Certificate may not be available. These two documents fulfill the same functional safety requirements and will be used interchangeably depending on the TI functional safety product.

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from Revision A (December 2024) to Revision B (August 2025)	Page
• Corrected *Revision A* date	5
• Added GPAMP in non-safety critical IP list	11
• Removed the statement "This can be a stand alone test or continuous monitoring, depending the application context." from OA3 description	34

Changes from Revision * (March 2024) to Revision A (December 2024)	Page
• Removed statements regarding ASIL-B compliance, re-worded to state the development process is quality managed	5
• Updated the part list	5
• Removed *Advanced Information* note	5
• Added additional information related to the safety mechanisms	33

This page intentionally left blank.

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), 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 will 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 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.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265