Safety Manual for Hercules™ TMS470M ARM® Safety Critical Microcontrollers

User's Guide

Literature Number: SPNU554
April 2012

Contents

List of Figures

List of Tables

User's Guide
SPNU554–April 2012

Safety Manual for Hercules™ TMS470M ARM® Safety Critical Microcontrollers

1 Introduction

You, as a system and equipment manufacturer or designer, are responsible to ensure that your systems (and any Texas Instruments hardware or software components incorporated in your systems) meet all applicable safety, regulatory, and system-level performance requirements. All application and safety related information in this document (including application descriptions, suggested safety measures, suggested TI products, and other materials) is provided for reference only. You understand and agree that your use of TI components in safety critical applications is entirely at your risk, and that you (as buyer) agree to defend, indemnify, and hold harmless TI from any and all damages, claims, suits, or expense resulting from such use.

This document is a safety manual for the Texas Instruments Hercules TMS470M safety critical microcontroller product family. The product family utilizes a common safety architecture that is implemented in multiple application focused products. Product implementations covered by this safety manual include:

- Hercules TMS470M Automotive Safety Critical Microcontrollers
  - TMS470MF06x
  - TMS470MF04x
  - TMS470MF03x

This Safety Manual provides information needed by system developers to assist in the creation of a safety critical system using a supported Hercules TMS470M microcontroller. This document contains:

- An overview of the superset product architecture
- An overview of the development process utilized to reduce systematic failures
- An overview of the safety architecture for management of random failures
- The details of architecture partitions, implemented safety mechanisms, and recommended usage

The user of this document should have a general familiarity with the Hercules TMS470M product families. More information can be found at http://www.ti.com/hercules. This document is intended to be used in conjunction with the pertinent data sheets, technical reference manuals, and other documentation for the products under development. This partition of technical content is intended to simplify development, reduce duplication of content, and avoid confusion.

2 TMS470M Product Overview

The TMS470M product family implements the ARM Cortex™-M3 CPU into the proven TMS470 platform architecture. A simplified graphical view of the product superset architecture can be seen in Figure 1. This is a basic representation of the architecture and is not all inclusive. For example, products in the family may scale based on the number of peripherals, separate or merged level two bus matrices, or amount of memory - but the programmer's model remains consistent.

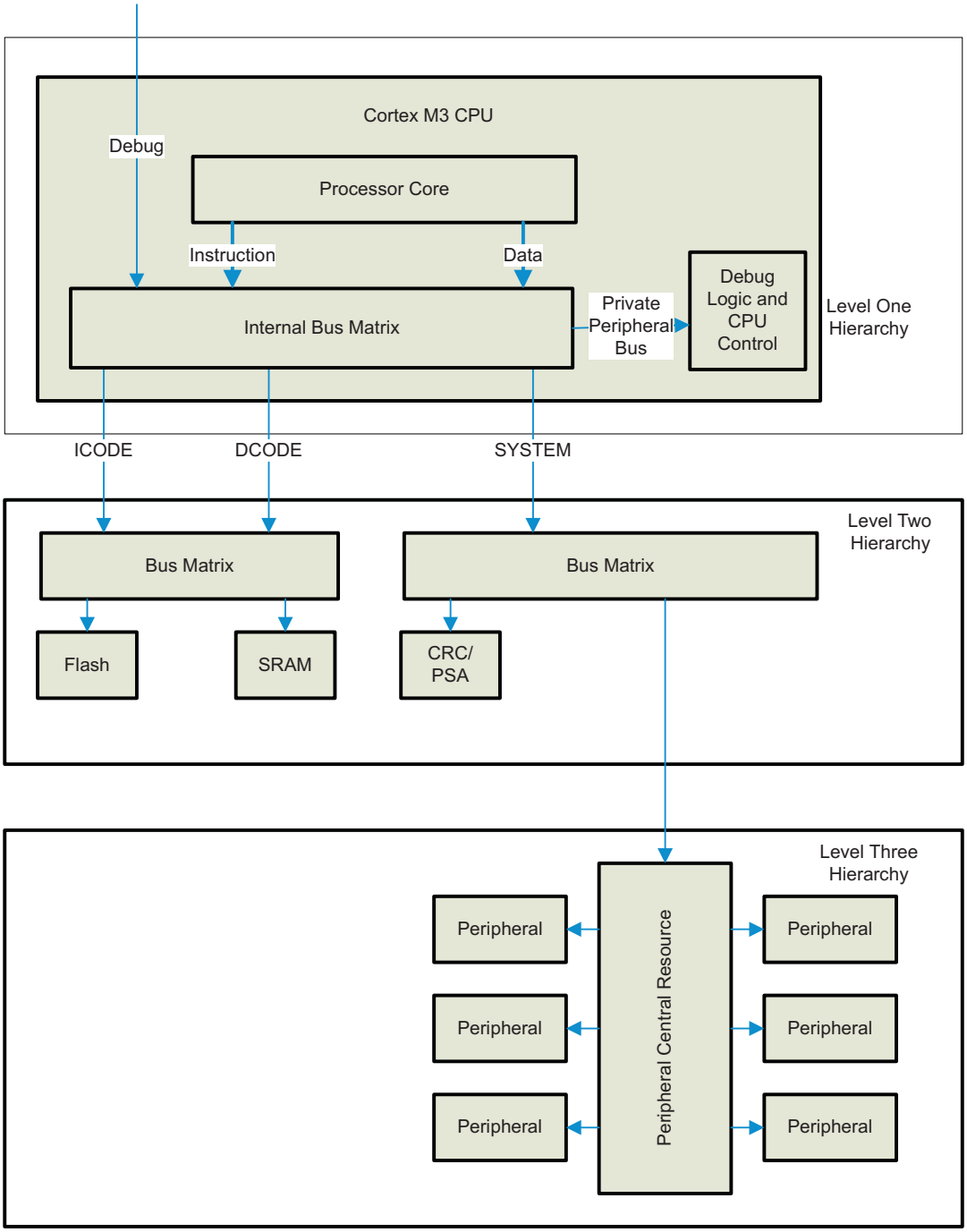

Figure 1. Hercules TMS470M Product Architecture Overview

The Hercules TMS470M product architecture can be conceptualized in three hierarchies. The level one hierarchy includes the ARM Cortex-M3 CPU. The Cortex-M3 CPU includes a processing core, debug logic, and an internal bus matrix. There are three bus masters and four bus slaves on the internal bus matrix. Internal bus masters include the processor core instruction and data as well as external debug. The bus slaves are the ICODE bus (for instruction fetch from Flash and SRAM), the DCODE bus (for data read/write from Flash and SRAM), the SYSTEM bus (for peripheral read/write access), and the Private Peripheral bus (for configuration of debug logic and CPU control functions). The internal bus matrix provides arbitration, prioritization, routing, and decode functionality for bus transactions.

The level two device hierarchy is dominated by two bus matrices (also known as switched central resources or crossbars). This is a device level interconnect, which allows the Cortex-M3 CPU to access multiple bus slaves while providing arbitration, prioritization, routing, decode, and decode functionality. Bus masters to the level two device hierarchy include ICODE, DCODE, and SYSTEM busses from the Cortex-M3 CPU. Bus slaves on the level two hierarchy include Flash memory, SRAM, a CRC/PSA engine, and access to the level three peripheral hierarchy.

The level three hierarchy is primarily composed of peripherals. Peripherals are grouped into a single peripheral bus segment, managed by a peripheral central resource. The peripheral central resource provides address decode functionality for bus transactions targeting peripherals.

2.1 Targeted Applications

The Hercules TMS470M MCU family is targeted at general-purpose safety applications. Multiple safety applications were analyzed during concept phase. Example target applications include:

- Automotive braking systems, including anti-lock braking (ABS), anti-lock braking with traction control (ABS+ TC), and electronic stability control (ESC)
- Motor control systems, particularly electronic power steering (EPS) systems and electrical vehicle (EV) power train
- General purpose safety computation, such as integrated sensor cluster processing
- Industrial automation such as programmable logic controllers (PLCs) and programmable automation controllers (PACs) for safety critical process control

While TI considered certain applications during the development of these devices, this should not restrict a customer who wishes to implement other systems. With all safety critical components, rationalization of the component safety concept to the system safety concept must be executed by the system integrator.

2.2 Product Safety Constraints

For safety components developed according to many safety standards, it is expected that the component safety manual will provide a list of product safety constraints. For a simple component, or more complex components developed for a single application, this is a reasonable response. However, the Hercules TMS470M product family is both a complex design and is not developed targeting a single, specific application. Therefore, a single set of product safety constraints cannot govern all viable uses of the product. The *Safety Analysis Report Summary for Hercules TMS470M ARM Safety Critical Microcontrollers (SPNU561)* provides an example implementation of the TMS470M product in a common system with relevant product safety constraints.

3 TMS470M Development Process for Management of Systematic Faults

For a safety critical development it is necessary to manage both systematic and random faults. Hercules TMS470M products are created using a standard quality managed development process which greatly reduces occurance of systematic faults.

3.1 TI Standard MCU Automotive Development Process

Texas Instruments has been developing automotive microcontrollers for safety critical and non-safety critical automotive applications for over twenty years. Automotive markets have strong requirements on quality management and high reliability of product. Though not explicitly developed for compliance to a functional safety standard, the TI standard MCU automotive development process already features many elements necessary to manage systematic faults. This development process can be considered to be Quality Managed (QM), but does not achieve an IEC 61058 Safety Integrity Level (SIL) or ISO 26262 Automotive Safety Integrity Level (ASIL). The TI standard MCU Automotive development process is certified compliant to ISO TS 16949 as assessed by Det Norske Veritas Certification, Inc. (Katy, Texas) under certificate CERT-07319CC10-2004-AQ-HOU-IATF (IATF certificate No 0113679). The development is also certified compliant to ISO 9001:2008 as assessed by DNV Certification B.V. (Netherlands) under certificate CERT-06185-2003-AQ-HOU-RvA Rev. 2.

The standard process breaks development into phases:

- Business opportunity pre-screen
- Program planning
- Create
- Validate, sample, and characterize
- Qualify
- Ramp to production and sustaining production

The standard process is illustrated in Figure 2.

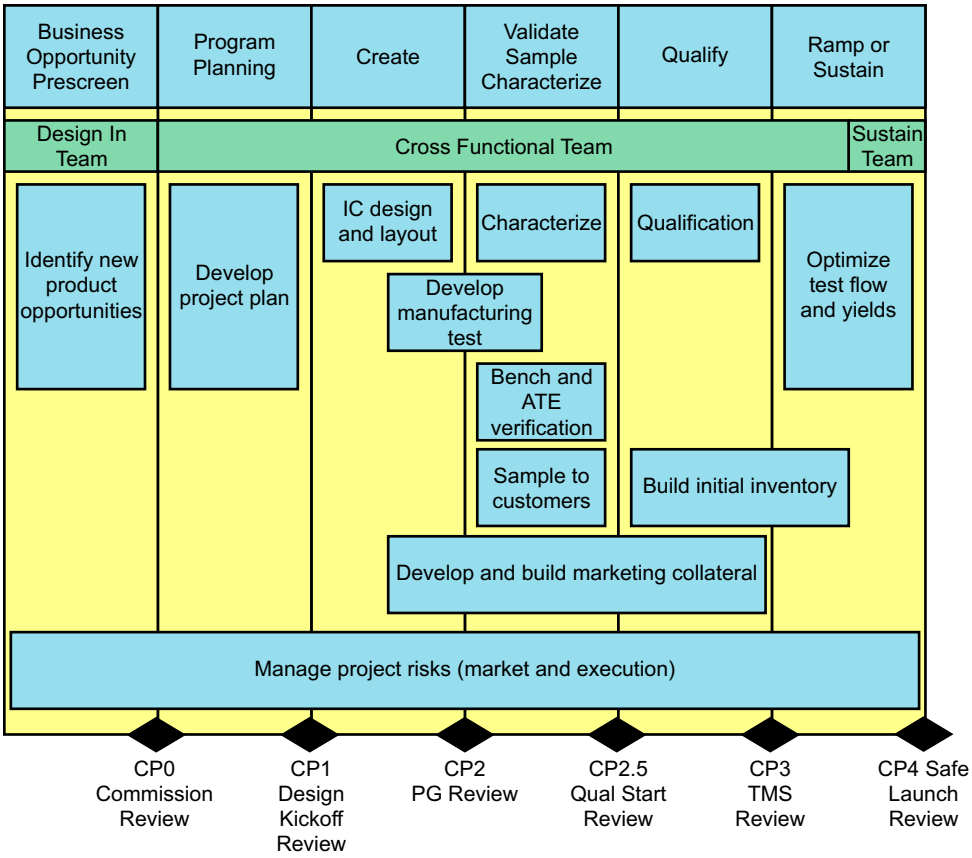

Figure 2. TI Standard MCU Automotive QM Development Process

3.2 Development Process Gaps to IEC 61508 and ISO 26262

The development of the Hercules TMS470M products predates the IEC 61508:2010 and ISO 26262:2011 functional safety standards. As such, there are development processes recommended by these standards which were not applied to the Hercules TMS470M products. Texas Instruments has conducted a gap analysis of the applied process to the IEC 61508 standard with the assistance of exida Consulting. The results of this activity can also be applied to ISO 26262 development due to similarity of standards. The results are summarized below.

Key findings of gap analysis:

- The hardware development process is already compliant to most aspects of Annex F of IEC 2nd edition of IEC 61508-2.
- A number of gaps have been identified, mostly related to requirements that are unique to functional safety, such as:
  - A functional safety management plan was not utilized to manage safety during program development.
  - Templates, checklists, or both were not created for functional safety related documents recommended by standard.
  - Functional safety assessment was not incorporated into the development process.
  - Safety requirements were not explictly defined and validated.

All findings of this exercise have been addressed in subsequent Hercules product families.

4 TMS470M Product Architecture for Management of Random Faults

For a safety critical development it is necessary to manage both systematic and random faults. The Hercules TMS470M product architecture includes many safety mechanisms, which can detect and respond to random faults when used correctly. This section of the document describes the architectural safety concept for the MCU.

4.1 Architecture Partition for Safety Analysis

The Hercules TMS470M processors share a common safety architecture. The basic concept involves a balance between application of hardware diagnostics and software diagnostics to manage functional safety, while balancing cost concerns. In this approach, a core set of elements are allocated hardware safety mechanisms. This core set of elements, including power and clock and reset, CPU, Flash memory, and SRAM can be used to assure functionally correct execution of software. Once correct operation of these elements is confirmed, software execution can begin on these elements in order to provide software-based diagnostics on other device elements, such as peripherals.

Figure 3 illustrates the safety concept overlaid to a superset configuration of the Hercules TMS470M product architecture.

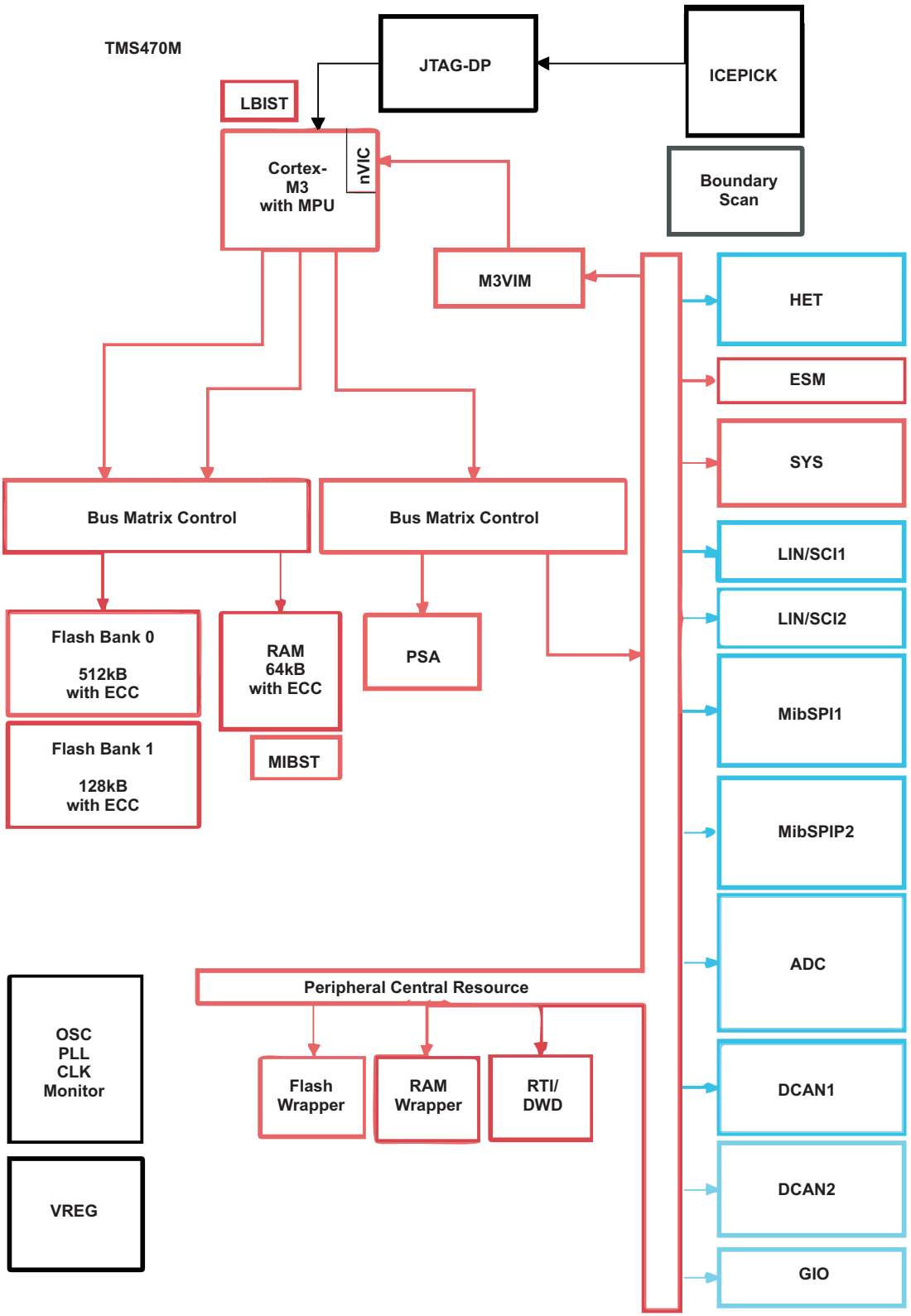

Figure 3. Partition of Hercules TMS470M MCU for Safety Analysis

Figure 3 illustrates three architectural partitions:

- "Hardware Layer" (RED) – This is the region of logic that is needed for all processing operations. This logic is protected heavily by on board hardware diagnostics and specific assumptions of use to assure a high level of confidence in safe operation. Once this region is safed, it can be used to provide comprehensive software diagnostics on other design elements.
- "Blended Layer" (BLUE) – This is the region of logic that includes most safety critical peripherals. This region has less reliance on hardware diagnostics. Software diagnostics and application protocols are overlaid to provide the remainder of needed diagnostic coverage.
- "Offline Layer" (BLACK) – This region of logic has minimal or no integrated hardware diagnostics. Many features in this layer are used only for debug, test, and calibration functions; which are not active during safety critical operation. Logic in this region could be utilized for safety critical operation assuming appropriate software diagnostics or system level measures are added by the system integrator.

4.2 Management of Family Variants

The Hercules TMS470M family architecture supports multiple product variants. These products could be implemented as unique silicon designs or they can be shared silicon designs that have elements disabled or not assured by specification, even if present in silicon. Only the elements of the superset architecture that are specifically detailed in the device-specific data sheet and technical reference manual are assured to be present and operate. When developing for the Hercules TMS470M platform, it is recommended that the safety concept be based on the superset product architecture to enable maximum scalability across family variants. The superset architecture shown in the previous section is valid for all device part numbers noted in the introduction of the safety manual.

4.3 Operating States

The Hercules TMS470M MCU products have a common architectural definition of operating states. These operating states should be observed by the system developer in their software and system level design concepts. The operating states state machine is shown in Figure 4 and described below.

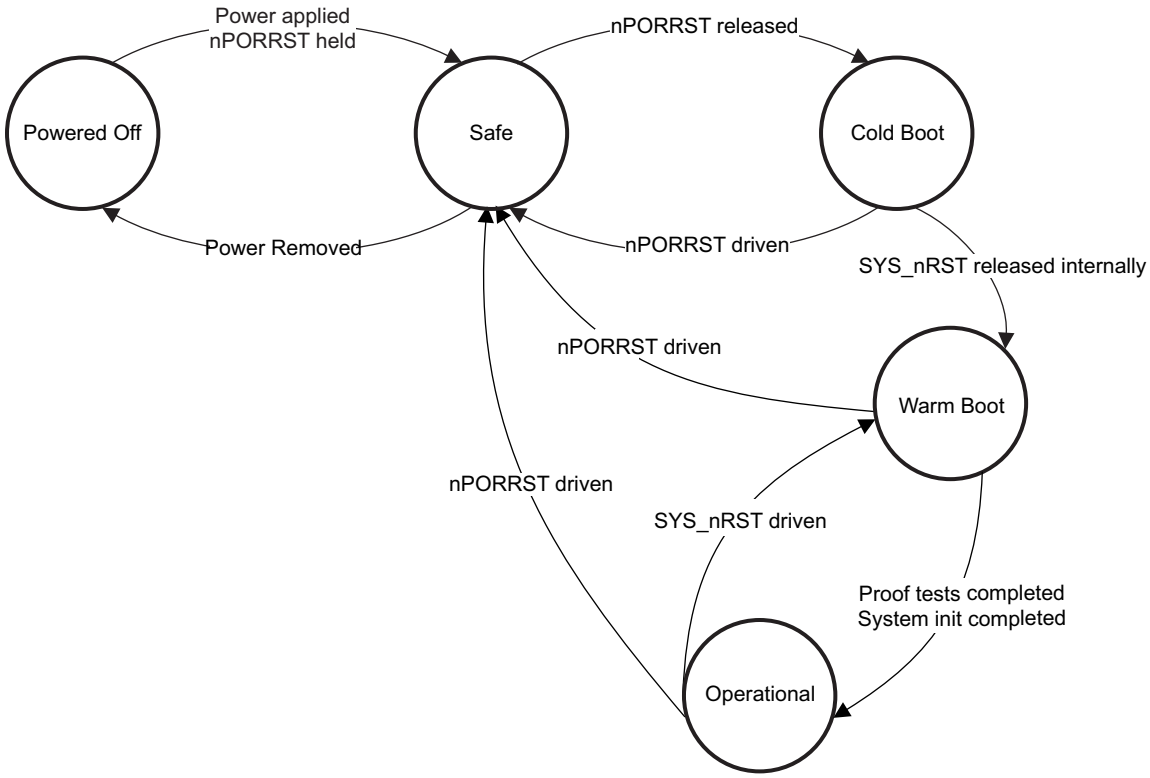

Figure 4. Operating States of the Hercules TMS470M MCU

- "Powered Off" - This is the initial operating state of the Hercules TMS470M MCU. No power is applied to either core or I/O power supply and the device is non-functional. This state can only transition to the safe state, and can only be reached from the safe state.
- "Safe" - In the safe state, the Hercules TMS470M MCU is powered but non-operational. The nPORRST (power-on reset, also known as cold reset) is asserted by the system but is not released until power supplies have ramped to a stable state. The internal voltage monitor (VMON) safety mechanism also continues to assert the nPORRST internal to the device if power supplies are not within a minimum operational range. When the product is in the safe state, the CPU and peripherals are non-functional. Output drivers are tri-stated and input/output pins are kept in an input only state.
- "Cold Boot" - In the cold boot state, key analog elements, digital control logic, and debug logic are initialized for future use. The CPU remains powered but non-operational. When the cold boot process is completed, the SYS_nRST signal is internally released, leading to the warm boot stage. The SYS_nRST signal transition change can be monitored externally on the SYS_nRST I/O pin.
- "Warm Boot" - The warm boot mode resets digital logic and enables the CPU. The CPU begins executing software from Flash memory and software initialization of the device can begin. There is no hardware interlock to say that warm boot is completed; this is a software decision.
- "Operational" - During the operational mode, the device is capable of supporting safety critical functionality.

4.4 Management of Errors

When a diagnostic detects a fault, the error must be indicated. The Hercules TMS470M product architecture provides aggregation of fault indication from internal safety mechanisms using a peripheral logic known as the error signaling module (ESM). The ESM provides mechanisms to classify errors by severity and to provide programmable error response. The error classifications in the ESM are summarized in Table 1.

Table 1. Summary of ESM Error Indication [(1)]

Error Group	Interrupt Response	Notes
1	Programmable interrupt and programmable interrupt priority	For errors that are generally not of critical severity
2	Non maskable interrupt generated	For errors that are generally of critical severity
3	No interrupt response	For critical errors that are seen by diagnostic implemented in CPU

(1) Unlike other Hercules products, the Hercules TMS470M family does not feature a device error pin due to reduced pin count requirements.

The error response is action that is taken by the MCU or system when an error is indicated. There are multiple potential of error response possible for the Hercules TMS470M product. The system integrator is responsible to determine what error response should be taken and to ensure that this is consistent with the system safety concept.

- CPU abort - This response is implemented directly in the CPU, for diagnostics implemented in the CPU. During an abort, the program sequence transfers context to an abort handler and software has an opportunity to manage the fault.
- CPU interrupt - This response can be implemented for diagnostics outside the CPU. An interrupt allows events external to the CPU to generate a program sequence context transfer to an interrupt handler where software has an opportunity to manage the fault.
- Generation of SYS_nRST - This response allows the device to change states to warm boot from operational state. The SYS_nRST could be generated from an external monitor or internally by the software reset or watchdog. Re-entry to the warm reset state allows possibility for software recovery when recovery in the operational state was not possible.

- Generation of nPORRST - This response allows the device to change state to safe state from cold boot, warm boot, or operational states. From this state, it is possible to re-enter cold boot to attempt recovery when recovery via warm boot is not possible. It is also possible to move to the powered-down state, if desired, to implement a system level safe state. This response can be generated from the internal voltage monitor, but is primarily driven by monitors external to the MCU.

The ESM provides multiple registers that can be read by the CPU to determine the current status of diagnostics. For the severe group 2 errors, a shadow register is provided that is not reset by SYS_nRST. This allows the possibility of warm reset reinitialization to identify that a group 2 error initiated the external reset.

It is possible for the CPU to trigger the error response manually to test system behavior. The CPU is responsible to clear indicated errors in the ESM.

System level management of the external error response can be simplified through the use of a TI system basis chip developed for use with the Hercules TMS470M family.

5 TMS470M Architecture Safety Mechanisms and Assumptions of Use

You, as a system and equipment manufacturer or designer, are responsible to ensure that your systems (and any TI hardware or software components incorporated in your systems) meet all applicable safety, regulatory, and system-level performance requirements. All application and safety related information in this document (including application descriptions, suggested safety measures, suggested TI products, and other materials) is provided for reference only. You understand and agree that your use of TI components in safety critical applications is entirely at your risk, and that you (as buyer) agree to defend, indemnify, and hold TI harmless from any and all damages, claims, suits, or expense resulting from such use.

In this section, the safety mechanisms for each major functional block of the Hercules TMS470M architecture are summarized and general assumptions of use are provided. This information should be used to determine the strategy for utilizing safety mechanisms. The details of each safety mechanism can be found in the device-specific technical reference manual for the MCU used. The effectiveness of the hardware safety mechanisms is noted in the *Safety Analysis Report Summary for Hercules TMS470M ARM Safety Critical Microcontrollers (SPNU561).*

TI classifies technical recommendations for the use of safety mechanisms in this section into a number of categories. The TI recommendations should not be considered infallible. There are many diverse ways to implement safe systems and alternate safety mechanisms may be possible that can provide support to achieve desired safety metrics. The categories of recommendation are as follows:

- Mandatory - A mandatory notation indicates a safety mechanism that is always operable during normal functional operation and cannot be disabled by user action.
- Highly Recommended - A highly recommended notation indicates a safety mechanism that TI believes to provide a high value of diagnostics that are difficult to implement by other means. The user retains the choice whether or not to utilize the safety mechanism in their design, as user action is either needed to enable the safety mechanism or user action can disable the safety mechanism.
- Recommended - A recommended notation indicates a safety mechanism that TI believes to provide a valuable diagnostic that can also be implemented by other means. The user retains the choice whether or not to utilize the safety mechanism in their design, as user action is either needed to enable the safety mechanism or user action can disable the safety mechanism.
- Optional - An optional notation indicates a safety mechanism that TI believe to provide a lower value diagnostic that can also be implemented by other means. The user retains the choice whether or not to utilize the safety mechanism in their design, as user action is either needed to enable the safety mechanism or user action can disable the safety mechanism.

Depending on the safety standard and end equipment targeted, it may be necessary to manage not only single point faults, but also latent faults. Per ISO 26262:2011, the latent faults are considered when there are faults both in a function and in the safety mechanism. Latent fault testing does not need to occur during the fault tolerant time interval, but can be performed at boot time, at shut down, or periodically as determined by the system developer. Many of the safety mechanisms described in this section can be used as primary diagnostics, diagnostics for latent fault, or both. When considering system design for management of latent faults, take care to include failure of CPU and memories in consideration for any primary diagnostic that is executed via software.

5.1 Power Supply + Voltage Regulator (VREG)

The Hercules TMS470M device family products contain an internal voltage regulator (VREG) that provides the necessary internal voltages for proper operation.

5.1.1 VREG Embedded Core Voltage Monitor

The Hercules TMS470M VREG incorporates a simple embedded voltage monitor that can detect grossly out of range supply voltages on the core power supply. This monitor operates continuously and requires no software configuration or CPU overhead. When the supply is grossly over or under specified voltages (for product specific values, see the device-specific data sheet), the voltage monitor drives the nPORRST (power-on reset) signal internally. This response holds the device in a safe operating state. When power supplies are in range, the voltage monitor will not interfere with the nPORRST signal. For more information on VREG and voltage monitor operation, see the device-specific data sheet.

The voltage monitor is a continuously operating diagnostic. It is not possible to disable the voltage monitor diagnostic. In-system test of the voltage monitor diagnostic is generally not feasible as tight control of external voltages is needed to trigger the voltage monitor error response. If improperly applied, such voltage could result in permanent damage to the MCU. Use of the voltage monitor is mandatory.

5.1.2 External Voltage Supervisor

The Hercules TMS470M platform highly recommends the use of an external voltage supervisor to monitor all voltage rails - both I/O and core voltages. The voltage supervisor should be configured with overvoltage and undervoltage thresholds matching the voltage range supported by the target device (as noted in the device-specific data sheet). Error response, diagnostic testability, and any necessary software requirements are defined by the external voltage supervisor selected by the system integrator.

5.1.3 Notes

- Devices can be implemented with multiple power rails that are intended to be ganged together on the system PCB. For proper operation of power diagnostics, it is recommended to implement one voltage supervisor per ganged rail.
- Common mode failure analysis of the external voltage supervisor may be useful to determine dependencies in the voltage generation and supervision circuitry

5.2 Clocks

The Hercules TMS470M device family products are primarily synchronous logic devices and as such require clock signals for proper operation. The clock management logic includes clock sources, clock generation logic including clock multiplication by phase lock loops (PLLs), clock dividers, and clock distribution logic. The registers that are used to program the clock management logic are located in the system module.

5.2.1 Low Power Oscillator Clock Detector (LPOCLKDET)

The low-power oscillator clock detector (LPOCLKDET) is a safety diagnostic that can be used to detect failure of the primary clock oscillator. LPOCLKDET utilizes the embedded high-frequency, low-power oscillator (HF LPO). The clock detect circuit works by checking for a rising edge on one clock (oscillator or HF LPO) between rising edges of the other clock. The result is that in addition to flagging incorrect, repeating frequencies, the circuit also fails due to transient conditions. The low end of the clock detect window ignores a transient low phase of at least 12 HF LPO cycles. Note that this filtering of the transient response does not change the input frequency range.

The LPOCLKDET circuitry is enabled by default during the power-on reset state. The diagnostic can be disabled via software. Use of the LPOCLKDET is highly recommended.

5.2.2 PLL Slip Detection

The PLL logic includes an embedded diagnostic that can detect a slip of the PLL output clock. The slip is a result of a loss of phase lock between reference clock and feedback clock. Error response and indication is dependent on the programming of the PLL control registers that are located in the system module. An ESM error indication can be generated or masked when PLL slip is detected. In addition, it is possible to generate an internal reset or to revert to operation from the oscillator clock in case of a detected PLL slip error. For more information on programming this diagnostic, see the device-specific technical reference manual.

The PLL slip detection diagnostic is active whenever the PLL is enabled and has locked on a target frequency. The diagnostic cannot be disabled by the software, but the error indication and error response can be modified by the software. Use of the PLL slip detection diagnostic is highly recommended.

5.2.3 Monitoring of External Clock Outputs (ECLK)

The Hercules TMS470M platform provides the capability to export select internal clocking signals for external monitoring. This feature can be configured via software by programming registers in the system module. To determine the number of external clock outputs implemented and the register mapping of internal clocks that can be exported, see the device-specific data sheet.

Export of internal clocks on the ECLK outputs is not enabled by default and must be enabled via software. It is possible to disable and configure this diagnostic via software. Use of the ECLK feature for external monitoring of internal clocks is optional.

5.2.4 Internal Watchdog

The Hercules TMS470M platform supports the use of an internal digital watchdog that is implemented in the real time interrupt (RTI) module. For details of programming the internal digital watchdog, see the device-specific technical reference manual.

The DWD is a traditional single threshold watchdog. The user programs a timeout value to the watchdog and must provide a predetermined "pet" response to the watchdog before the timeout counter expires. Expiration of the timeout counter or an incorrect "pet" response triggers an error response. The DWD can issue either an internal (warm) system reset or a CPU non-maskable interrupt upon detection of a failure. The DWD is not enabled after reset. Once enabled by the software, the DWD cannot be disabled except by system reset or power-on reset. Use of the DWD functionality is recommended.

5.2.5 External Watchdog

When using an external watchdog, there is a possibility to reduce common mode failure with the MCU clocking system, as the watchdog can utilize clock, reset, and power that are separate from the system being monitored. Error response, diagnostic testability, and any necessary software requirements are defined by the external watchdog selected by the system integrator. The Hercules TMS470M platform highly recommends the use of an external watchdog over the internally provided watchdogs.

5.2.6 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of a read back of configuration registers mechanism is recommended.

5.2.7 Software Read Back of Written Configuration

In order to ensure proper configuration of memory-mapped clock control registers in the system module, it is highly recommended to software implement a test to confirm proper operation of all control register writes. To support this software test, it is highly recommended to configure the system module memory space as a strongly ordered, non-bufferable memory region using the Cortex-M3 memory protection unit. This ensures that the register write has completed before the read back is initiated.

5.2.8 Notes

- Management of the external watchdog functionality at system level can be simplified by using a TI system basis chip developed for use with the Hercules TMS470M family.
- User can improve the accuracy of the LPOCLKDET diagnostic via programming the trim values in the HF LPO. This would require the customer to determine the LPO trim value during their manufacturing test via comparison to a calibrated clock source.
- There are many possible implementations of watchdogs for use in providing clock and CPU diagnostics. In general, TI recommends the use of an external watchdog over an internal watchdog for reasons of reduced common mode failure.
- Driving a high-frequency clock output on the ECLK pin may have EMI implications.

5.3 Reset

The Hercules TMS470M device family products require an external reset at cold and power-on (nPORRST) to place all asynchronous and synchronous logic into a known state. The power-on reset generates an internal warm reset (nRST) signal to reset the majority of digital logic as part of the boot process. The nRST signal is provided at device level as an I/O pin; it will toggle when asserted internally and can be driven externally to generate a warm reset. For more information on the reset functionality, see the device-specific data sheet.

5.3.1 External Monitoring of Warm Reset (nRST)

The nRST warm reset signal is implemented as an I/O. An external monitor can be utilized to detect expected or unexpected changes to the state of the internal warm reset control signal. Error response, diagnostic testability, and any necessary software requirements are defined by the external monitor selected by the system integrator. Use of this feature is considered optional.

5.3.2 Software Check of Cause of Last Reset

The system module provides a status register (SYSESR) that latches the cause of the most recent reset event. A boot software that checks the status of this register to determine the cause of the last reset event is typically implemented by software developers. This information can be used by the software to manage failure recovery. Software use of the SYSESR to check last reset cause is highly recommended.

5.3.3 Software Warm Reset Generation

The system module provides the ability to the software to generate an internal warm reset (nRST). This is accomplished by writing appropriate control bits in the SYSECR control register. Software can utilize this feature to attempt failure recovery. Use of the software warm reset is optional.

5.3.4 Glitch Filtering on nRST and nPORRST

Glitch filters are implemented on the cold and warm reset pins of the device. These structures filter out noise and transient signal spikes on the input reset pins in order to reduce unintended activation of the reset circuitry. The glitch filters are continuously operating and their behavior cannot be changed by the software. Use of the glitch filters is mandatory.

5.3.5 Shadow Registers

The use of a two stage cold and warm reset scheme on the device allows the implementation of shadow registers. Shadow registers are reset only by power-on reset. These registers can be used to store device status or other critical information that remain persistent after a warm reset changes the system state. The system module includes shadow registers that can be used to support failure recovery via software. Use of the shadow register status information by boot software is highly recommended.

5.3.6 External Watchdog

An external watchdog can provide secondary diagnostic. For more information on this diagnostic, see External Watchdog.

5.3.7 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration registers mechanism is recommended.

5.3.8 Software Read Back of Written Configuration

In order to ensure proper configuration of memory-mapped reset control registers in the system module, it is highly recommended that software implement a test to confirm proper operation of all control register writes. To support this software test, it is highly recommended to configure the system module memory space as a strongly ordered, non-bufferable memory region using the Cortex-M3 memory protection unit. This ensures that the register write has completed before the read back is initiated.

5.3.9 Notes

- Management of reset at system level can be simplified by using a TI system basis chip developed for use with the TMS470 family.
- Internal watchdogs are not a viable option for reset diagnostics as the monitored reset signals interact with the internal watchdogs.

5.4 System Module

The system control module contains the memory-mapped registers to interface clock, reset, and other system related control and status logic. The system control module is also responsible for generating the synchronization of system resets and delivering the warm system reset nRST.

5.4.1 Privileged Mode Access and Multi-Bit Enable Keys

The system module design includes features to support avoidance of unintentional control register programmation. These features include limitation of write commands to privilege bus master transactions and the implementation of multi-bit keys for critical controls. The multi-bit keys are particularly effective for avoiding unintentional activation. For more details on the register safety mechanisms and error response, see the device-specific technical reference manual.

The operation of this safety mechanism is continuous and cannot be altered by the software. This mechanism can be tested by generating software transactions and reviewing the device response. Use of this safety mechanism is mandatory.

5.4.2 Software Read Back of Written Configuration

In order to ensure proper configuration of memory-mapped control registers in the system module, it is highly recommended that software implement a test to confirm proper operation of all control register writes. To support this software test, it is highly recommended to configure the system module memory space as a strongly ordered, non-bufferable memory region using the Cortex-M3 memory protection unit. This ensures that the register write has completed before the read back is initiated.

5.4.3 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration registers mechanism is recommended.

5.4.4 Notes

- Depending on targeted metrics, a user can elect to implement a periodic software test of static configuration registers in the system module. Such a test can provide additional diagnostic coverage for disruption by soft error.
- Review the clock and reset sections as these features are closely controlled by the system module.

5.5 Error Signaling Module (ESM)

The ESM provides unified aggregation and prioritization of on-board hardware diagnostic errors. For more information, see Management of Errors

5.5.1 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration registers mechanism is recommended.

5.5.2 Software Test of Error Path Reporting

A software test can be utilized to inject diagnostic errors and check for proper reporting. Such a test can be executed at boot or periodically. Necessary software requirements are defined by the software implemented by the system integrator. The use of a boot time software test of error path reporting is highly recommended. The use of a periodic software test of error path reporting is recommended.

5.5.3 Shadow Registers

The use of a two stage cold and warm reset scheme on the device allows the implementation of shadow registers. Shadow registers reset only by power-on reset. These registers can be used to store device status or other critical information that remain persistent after a warm reset changes the system state. The error signaling module includes shadow registers that can be used to support failure recovery via software. Use of the shadow register status information by boot software is highly recommended.

5.5.4 Software Read Back of Written Configuration

In order to ensure proper configuration of memory-mapped control registers in the ESM, it is highly recommended that the software implement a test to confirm proper operation of all control register writes. To support this software test, it is highly recommended to configure the ESM memory space as a strongly ordered, non-bufferable memory region using the Cortex-M3 memory protection unit. This ensures that the register write has completed before the read back is initiated.

5.5.5 Notes

- Software testing of the ESM error path can be combined with boot time latent tests of hardware diagnostics to reduce startup time.

5.6 CPU Subsystem

The Hercules TMS470M product family relies on the ARM Cortex-M3 CPU to provide general-purpose processing.

5.6.1 CPU Logic Built In Self Test (LBIST) Self -Test Controller (STC)

The Hercules TMS470M family architecture supports the use of a hardware logic BIST (LBIST) engine self-test controller (STC). This logic is used to provide a very high diagnostic coverage on the CPU at a transistor level. This logic utilizes the same design for test (DFT) structures inserted into the device for rapid execution of high quality manufacturing tests, but with an internal test engine rather than external automated test equipment (ATE). This technique has proven to be drastically more effective than software-based tests of logic, particularly for the complex logic structures seen in a modern CPU.

The LBIST tests must be triggered by the software. User may elect to run all tests, or only a subset of the tests based on the execution time, which can be allocated to the LBIST diagnostic. This time sliced test feature enables the LBIST to be used effectively as a runtime diagnostic with execution of test time slices per safety critical loop as well as a comprehensive test for CPU fault during MCU initialization.

Execution of the LBIST STC results in a much higher level of transistor switching per clock cycle than during normal software execution due to the high efficiency of the test. A software control is implemented in the STC that allows the user to reduce the CPU clock for the duration of the test. This feature allows the user to make a compromise between fast execution with higher current consumption or slower execution with reduced current consumption.

The LBIST mechanism requires isolation of the CPU from the remainder of device logic while under test. It is also necessary to perform a complete context save before the LBIST. When test execution is complete, the CPU will be reset. The remainder of device logic continues normal operation. After CPU reset, software should read the system module SYSESR to identify the reason for the reset and can then restore the CPU context.

LBIST logic includes capabilities for testing proper operation of the diagnostic. As the test time for the diagnostic is deterministic, a timeout counter has been included that can detect a failure to complete the test within expected time. In addition, there is the possibility to force an input error to check error detection and propagation of the error response at system level. This test is performed as follows:

- Enable the self_check_key and fault_ins bits in the STCSTSCR register.
- Enable STC test interval zero and execute the test
- The fail bit should be set to 1 in the STC global status register upon completion of test.
- Disable either or both the self_check_key and fault_ins bits in the STCSTSCR register.
- Restart the self test by programming bit 0 of the STCGCR, causing self-test restart.
- The fail bit should be set to 0 in the STC global status register upon completion of the test.

Use of the LBIST logic at boot time is highly recommended. Use of the LBIST logic for periodic execution during normal execution is recommended. The cyclical check applied by the LBIST module provides an inherent level of self checking (autocoverage), which can be considered for application in latent fault diagnostics.

5.6.2 CPU Memory Protection Unit (MPU)

The Hercules TMS470M implementation of the Cortex-M3 CPU includes an MPU. The MPU logic can be used to provide spatial separation of software tasks in the device memory. It is expected that the operating system controls the MPU and changes the MPU settings based on the needs of each task. A violation of a configured memory protection policy results in a CPU abort. Use of the MPU is highly recommended.

The MPU can also be used to configure the memory ordering policies of the memory system. By default, all peripheral accesses are strongly ordered - meaning that all transactions complete in the sequence are issued and no write transactions are buffered. If desired, the operating system can configure accesses to be device - meaning that writes are buffered. This can improve performance over a strongly ordered model, at the cost of some determinism. It is highly recommended that the system module and other modules deemed to have critical configurations be set to a strongly ordered access model.

The LBIST STC diagnostic provides a check of the MPU when it performs a test of the CPU. Additional software-based testing of the MPU for proper operation and error response is recommended.

5.6.3 Internal and External Watchdog

An internal or external watchdog can provide secondary diagnostics. For more information on these diagnostics, see Internal Watchdog or External Watchdog.

5.6.4 Illegal Operation and Instruction Trapping

The Cortex-M3 CPU includes diagnostics for illegal operations and instructions that can serve as safety mechanisms. Many of these traps are not enabled after reset and must be configured by the software. Installation of software handlers to support the hardware illegal operation handling and instruction trapping is highly recommended. Examples of CPU illegal operation and instruction traps include:

- Illegal instruction
- Privilege violation

5.6.5 Software Read Back of Written Configuration

In order to ensure proper configuration of CPU coprocessor control registers, it is highly recommended that software implement a test to confirm proper operation of all control register writes. The CPU control registers are not memory mapped and must be accessed via the CPU coprocessor read and write commands.

5.6.6 Notes

- Many safety critical microcontrollers utilize a software-based test of CPU functionality as opposed to a hardware scheme such as the TI LBIST STC. TI does not recommend such tests for a CPU of medium to high complexity, such as the Cortex-M3. Software-based options have higher memory cost, lower detection capability, and longer execution times than the equivalent LBIST STC solution.

5.7 Embedded Flash

The embedded Flash memory on Hercules TMS470M devices is non-volatile on-chip memory that is primarily used for CPU instruction access, though data access is also possible. Access to the Flash memory can take multiple CPU cycles. The Flash wrapper logic provides multiple pipelined read buffers to improve CPU access time on sequential address fetches.

5.7.1 Flash ECC

The on-chip Flash memory is supported by single error correction, dual error detection (SECDED) error-correcting code (ECC) diagnostic which is located in the Flash wrapper. It is connected to the device bus matrix via a 64b data interface. In this SECDED scheme, an 8-bit code word is used to store the ECC data as calculated for each 64-bit data payload. Errors detected by the Flash ECC in the Flash wrapper are reported to the ESM. The ESM then provides notification of the error to the Cortex-M3 CPU.

The ECC logic for the Flash is disabled at reset and must be configured in the Flash wrapper. Use of the Flash ECC is highly recommended. The cyclical check applied by the ECC module provides an inherent level of self checking (autocoverage), which can be considered for application in latent fault diagnostics.

5.7.2 Flash Contents Check by Hardware CRC

The platform includes a hardware cyclic redundancy check (CRC) implementing the ISO CRC-64 standard polynomial. The CRC module can be used to test the integrity of Flash contents by calculating a CRC for all Flash contents and comparing this value to a previously generated "golden" CRC. The comparison of results, indication of fault, and fault response are the responsibility of the software managing the test. It is highly recommended to perform a CRC integrity check of Flash contents at boot time. Periodic execution of the CRC integrity check at runtime is recommended. The cyclical check applied by the hardware CRC module provides an inherent level of self checking (autocoverage), which can be considered for application in latent fault diagnostics.

5.7.3 Bit Multiplexing in Flash Memory Array

The Flash modules implemented in the Hercules TMS470M architecture have a bit multiplexing scheme implemented such that the bits accessed to generate a logical (CPU) word are not physically adjacent. This scheme helps to reduce the probability of physical multi-bit faults resulting in logical multi-bit faults; rather they manifest as multiple single bit faults. As the SECDED Flash ECC can correct a single bit fault in a logical word, this scheme improves the usefulness of the Flash ECC diagnostic. Bit multiplexing is a mandatory feature of the architecture and cannot be modified by the software.

5.7.4 Flash Sector Protection

It is possible to prevent a write operation on a sector by the software configuration of the Flash wrapper. The sector protection registers, Bank Sector Enable Register (BSE), contain a bit for each sector in the Flash bank, which enables or disables a sector for write operation. The BSE register can only be written in privilege mode while the software PROTLIDIS protection bit is set high. This mechanism can reduce probability of unintended programming of Flash memory. Use of the Flash sector protection feature is highly recommended.

5.7.5 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration register mechanism is recommended.

5.7.6 Software Read Back of Written Configuration

In order to ensure proper configuration of memory-mapped control registers in the Flash wrapper, it is highly recommended that software implement a test to confirm proper operation of all control register writes. To support this software test, it is highly recommended to configure the Flash wrapper memory space as a strongly ordered, non-bufferable memory region using the Cortex-M3 memory protection unit. This ensures that the register write has completed before the read back is initiated.

5.7.7 Notes

- By executing a CRC read back of Flash contents while ECC is enabled on the Flash, it is possible to perform two diagnostics of the Flash in parallel.
- The Flash module may have unique power supply pins depending on the product configuration. It is highly recommended to implement voltage supervision on these pins as described in the power supply section.
- The Flash module may have unique test signal pins depending on the product configuration. For proper board level management of these signals, see the device-specific data sheet.

5.8 Flash EEPROM Emulation (FEE)

The second bank of embedded Flash on the device can be used either as standard Flash, or to support EEPROM emulation. All recommendations for standard Flash apply to the FEE bank.

5.9 Primary Embedded SRAM

The primary embedded SRAM is a volatile memory that is primarily used for CPU data access, though instruction access is also possible. The SRAM has much faster access time than Flash memory, such that no wait states are necessary at maximum CPU frequency.

5.9.1 Data ECC

The on-chip SRAM is supported by SECDED ECC diagnostic embedded in the SRAM wrapper. The SRAM wrapper is connected to the bus matrix via a 64b data interface. In this SECDED scheme, an 8-bit code word is used to store the ECC data as calculated over the 64-bit data payload. Errors detected by the SRAM ECC in the SRAM wrapper are reported to the ESM. The ESM then provides notification of the error to the Cortex-M3 CPU.

The ECC logic for the SRAM is disabled at reset and must be configured in the SRAM wrapper. Use of the SRAM ECC is highly recommended. The cyclical check applied by the ECC logic provides an inherent level of self checking (autocoverage), which can be considered for application in latent fault diagnostics.

5.9.2 Correctable ECC Profiling

The SRAM wrapper includes a capability to count the number of correctable ECC errors detected. When the error count exceeds a user programmed threshold, an error event is signaled to the ESM. This mechanism is disabled by default and must be enabled by the software in the SRAM wrapper. Use of the correctable SRAM ECC profiling feature is recommended.

5.9.3 Programmable Memory BIST (MBIST)

The Hercules TMS470M family architecture supports the use of a hardware memory BIST (MBIST) engine. This logic is used to provide a very high diagnostic coverage on the implemented SRAMs at a transistor level. The MBIST logic utilizes the same design for test (DFT) logic and algorithms used by TI for manufacturing tests. This technique has proven to be drastically more effective than software-based tests of SRAM, particularly for the devices with complex CPUs whose addressing modes do not enable an optimal software-based test.

The MBIST tests must be triggered by the software. The user can elect to run the MBIST on one SRAM or on groups of SRAMs based on the execution time, which can be allocated to the MBIST diagnostic. The MBIST tests are destructive to memory contents, and as such are typically run only at MCU initialization. However, the user has the freedom to initiate the tests at any time when the CPU is operable.

Execution of the MBIST results in a much higher level of transistor switching per clock cycle than during normal software execution due to the high efficiency of the test. If lower current is desired during operation, the MBIST memory tests can be run sequentially by the software, instead of in parallel.

The most effective SRAM tests known to TI require test across a full physical memory module and are destructive to previous memory contents. If MBIST is to be implemented during operation, it is recommended to copy the data from the SRAM to be tested to a non-tested memory before test execution and to restore the data once the test is complete. When test execution is complete, the SRAM can be utilized for normal operation. The remainder of device logic continues normal operation during SRAM test. Any fault detected by the MBIST results in an error indicated in MBIST status registers.

Use of the MBIST logic at device initialization is highly recommended. Use of the MBIST logic for periodic execution during normal execution is optional. The cyclical check applied by the MBIST logic provides an inherent level of self checking (autocoverage), which can be considered for application in latent fault diagnostics.

5.9.4 SRAM Bit Multiplexing

The SRAM implemented in the Hercules TMS470M architecture has a bit multiplexing scheme implemented such that the bits accessed to generate a logical (CPU) word are not physically adjacent. This scheme helps to reduce the probability of physical multi-bit faults resulting in logical multi-bit faults; rather they manifest as multiple single bit faults. As the SECDED SRAM ECC can correct a single bit fault in a logical word, this scheme improves the usefulness of the SRAM ECC diagnostic. Bit multiplexing is a mandatory feature of the architecture and cannot be modified by the software.

5.9.5 SRAM Hardware CRC-64

The platform includes a hardware CRC implementing the ISO CRC-64 standard polynomial. The CRC module can be used to test the integrity of static contents in the SRAM by calculating a CRC for all static contents and comparing this value to a previously generated "golden" CRC. The comparison of results, indication of fault, and fault response are the responsibility of the software managing the test. Because most static values are stored in Flash, execution of a CRC on static contents of the SRAM is optional. The cyclical check applied by the CRC logic provides an inherent level of self checking (autocoverage), which can be considered for application in latent fault diagnostics.

5.9.6 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration registers mechanism is recommended.

5.9.7 Software Read Back of Written Configuration

In order to ensure proper configuration of memory-mapped control registers in the SRAM wrapper, it is highly recommended that software implement a test to confirm proper operation of all control register writes. To support this software test, it is highly recommended to configure the SRAM wrapper memory space as a strongly ordered, non-bufferable memory region using the Cortex-M3 memory protection unit. This ensures that the register write has completed before the read back is initiated.

5.9.8 Notes

- By executing a CRC read back of SRAM contents while ECC is enabled on the SRAM, it is possible to perform two diagnostics of the SRAM in parallel.
- The SRAM module may have unique power supply pins depending on the product configuration. It is highly recommended to implement voltage supervision on these pins as described in the power supply section.
- The redundant address decode does not provide hardware fault tolerance. As such, the logic may not be considered fully redundant per the definition of redundancy in some safety standards.

5.10 Interconnect Subsystem

The interconnect subsystem provides the datapath between the CPU bus masters and the Flash, SRAM, and peripheral bus slaves. This subsytem provides arbitration, prioritization, routing, and decode functionality for bus transactions.

5.10.1 Error Trapping

The interconnect subsystem includes a number of mechanisms to detect and trap errors. Address decoders in the diagnostic respond with a bus error to the initiator if a bus transaction does not decode to a valid target. Logic is also present that can detect the timeout of certain transactions and respond with a bus error to the transaction initiator.

The interconnect error trapping functionality is enabled by default and cannot be disabled by the software. Use of this safety mechanism is mandatory. These features can be tested via software through the insertion of invalid bus transactions.

5.10.2 Peripheral Central Resource (PCR) Access Management

The peripheral central resource (PCR) provides two safety mechanisms that can limit access to peripherals. Peripherals can be clock gated per peripheral chip select in the PCR. This can be utilized to disable unused features such that they cannot interfere with active safety functions. In addition, each peripheral chip select can be programmed to limit access based on privilege level of transaction. This feature can be used to limit access to entire peripherals to privileged operating system code only.

These safety mechanisms are disabled after reset. Software must configure and enable these mechanisms. Use of these mechanisms is highly recommended.

5.10.3 Internal / External Watchdog

An internal or external watchdog can provide secondary indication of a transaction that has timed out due to an issue with interconnect. For more information on these diagnostics, see Internal Watchdog or External Watchdog.

5.10.4 Information Redundancy Techniques

Information redundancy techniques can be applied via software as an additional runtime diagnostic on the interconnect. There are many techniques that can be applied, such as read back of written values and multiple reads of the same target data with comparison of results. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of information redundancy techniques in interconnect transactions is recommended.

5.10.5 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration registers mechanism is recommended.

5.10.6 Software Test of Basic Functionality

A software test can be utilized to inject diagnostic errors and check for proper error response. Such a test can be executed at boot or periodically. Software requirements necessary are defined by the software implemented by the system integrator. The use of a boot time software test of basic functionality is highly recommended. The use of a periodic software test of basic functionality reporting is recommended.

5.10.7 Software Read Back of Written Configuration

In order to ensure proper configuration of memory-mapped control registers in the PCR, it is highly recommended that software implement a test to confirm proper operation of all control register writes. To support this software test, it is highly recommended to configure the PCR memory space as a strongly ordered, non-bufferable memory region using the Cortex-M3 memory protection unit. This ensures that the register write has completed before the read back is initiated.

5.10.8 Notes

- An end to end communications safety mechanism implemented on a networked peripheral provides an indirect form of information redundancy diagnostic on the L2 and L3 interconnect.
- The only module in the L2 and L3 interconnect subsystem that has memory-mapped registers is the PCR

5.11 M3 Vectored Interrupt Module (M3VIM)

The vectored interrupt module (M3VIM) is used to interface peripheral interrupts to the NVIC interface of the Cortex-M3 CPU. The M3VIM provides programmable interrupt prioritization, masking, and sleep mode wake up functionality. The M3VIM includes a local SRAM that is used to hold the address of the interrupt handler per channel.

5.11.1 Periodic Software Test of M3VIM Operation

Per guidance found in IEC 61508, a software test for detecting continuous interrupts, no interrupts, and crossover interrupts can be implemented. Such testing could include software forced interrupts to check VIM and CPU response, or periodic interrupts from the RTI module specifically for the purpose of VIM testing. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of periodic software test of VIM operation is highly recommended.

5.11.2 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration registers mechanism is recommended.

5.11.3 Software Read Back of Written Configuration

In order to ensure proper configuration of memory-mapped control registers in the M3VIM, it is highly recommended that software implement a test to confirm proper operation of all control register writes. To support this software test, it is highly recommended to configure the VIM memory space as a strongly ordered, non-bufferable memory region using the Cortex-M3 memory protection unit. This ensures that the register write has completed before the read back is initiated.

5.11.4 Internal and External Watchdog

An internal or external watchdog can provide secondary indication of a transaction that has timed out due to an issue with interconnect. For more information on these diagnostics, see Internal Watchdog or External Watchdog.

5.12 Real Time Interrupt (RTI)

The real time interrupt (RTI) module provides the operating system timer for the device. The OS timer function is used to generate internal event triggers or interrupts as needed to provide periodic operation of safety critical functions.

5.12.1 Use of 2nd Counter as Diagnostic

The RTI module contains at minimum two up-counters that can be used to provide an operating system time-tick. While one up-counter is used as the operating system timebase, it is possible to use the second up counter as a diagnostic on the first, via periodic check via software of the counter values in the two timers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of a second counter to diagnose faults in RTI is recommended.

5.12.2 Internal / External Watchdog

A internal or external watchdog can provide indication of a fault in the RTI module. For more information on these diagnostics, see Internal Watchdog or External Watchdog.

5.12.3 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration registers mechanism is recommended.

5.12.4 Notes

- When using one counter to the operating system timebase counter, a diverse configuration of clock source, scaling factor, and so forth can be used to reduce probability of common mode failure.

5.13 High-End Timer (HET)

The HET module is a programmable timer with input/output capabilities. The HET is implemented as a simple RISC processor with instruction set dedicated for timing operations. Complex inputs can be captured and pre-processed by the HET for later processing by the CPU. Output generation is typically pulse width modulation (PWM), but may also be simple general-purpose input/output (GIO) type signals.

5.13.1 Software Test of Function Using I/O Loopback

A software test can be utilized to inject diagnostic errors and check for proper error response. Such a test can be executed at boot or periodically. Software requirements necessary are defined by the software implemented by the system integrator. The use of a boot time software test of basic functionality is highly recommended. The use of a periodic software test of basic functionality reporting is optional.

The HET implementation supports both digital and analog loopback capabilities for the I/Os. Digital loopback tests the signal path to the module boundary. Analog loopback tests the signal path from the module to the I/O cell with output driver disabled. For best results any tests of the HET functionality should include the I/O loopback.

5.13.2 HET SRAM Parity

The HET SRAM includes a parity diagnostic that can detect single bit errors in the memory. When a parity error is detected the ESM is notified. This feature is disabled after reset. Software must configure and enable this feature. Use of the HET SRAM parity feature is highly recommended.

5.13.3 HET and SRAM MBIST

The HET SRAM can be tested using the MBIST memory test. Execution of MBIST tests on HET SRAM after reset is highly recommended. For more details on this diagnostic, see PBIST.

5.13.4 HET SRAM Bit Multiplexing

The HET SRAM is implemented with a memory design such that logically adjacent bits are not physically adjacent. Use of this safety mechanism is mandatory. For more details on this safety mechanism, see SRAM Bit Multiplexing.

5.13.5 HET SRAM CRC-64 Testing

The HET SRAM contents can be tested periodically using the hardware CRC-64 diagnostic. For more details on this diagnostic, see SRAM Hardware CRC-64.

5.13.6 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration registers mechanism is recommended.

5.14 Multi-Buffered Analog-to-Digital Converter (MibADC)

The MibADC module is used to convert analog inputs into digital values. Results are stored in internal MibADC SRAM buffers for later transfer by the CPU.

5.14.1 Input Self-Test

The Hercules TMS470M MibADC module implements an input self-test engine that can detect a short to ADREFLO, ADREFHI or open input. Software must configure and enable and evaluate the result of this diagnostic. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of the input self-test mechanism is highly recommended.

5.14.2 Converter Calibration

The Hercules TMS470M MibADC module implements calibration logic normally used to improve converter accuracy. This logic can also be used as a safety mechanism. Software comparison of the conversion of known reference values from the calibration logic can provide a diagnostic on converter functionality. Repeated execution of the calibration routine can be used to detect drift during application.

Software must configure and enable and evaluate the result of this diagnostic. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of the converter calibration mechanism at boot is highly recommended. The use of the converter calibration mechanism periodically is optional.

5.14.3 Information Redundancy Techniques

Information redundancy techniques can be applied via software as an additional runtime diagnostic on ADC conversions. There are many techniques that can be applied, such as using multiple channels on the same converter or by doing multiple conversions on the same channel followed with comparison of results.

Filtering or a plausibility check for the converted values are in expected range can also improve diagnostic coverage.

Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of information redundancy techniques on ADC conversions is highly recommended.

5.14.4 ADC SRAM Parity

The MibADC SRAM includes a parity diagnostic that can detect single bit errors in the memory. When a parity error is detected the ESM is notified. This feature is disabled after reset. Software must configure and enable this feature. Use of the MibADC SRAM parity feature is highly recommended.

5.14.5 ADC SRAM MBIST

The MibADC SRAM can be tested using the MBIST memory BIST engine. Execution of MBIST tests on MibADC SRAM after reset is highly recommended. For more details on this diagnostic, see PBIST.

5.14.6 ADC SRAM Bit Multiplexing

The MibADC SRAM is implemented with a memory design such that logically adjacent bits are not physically adjacent. Use of this safety mechanism is mandatory. For more details on this safety mechanism, see SRAM Bit Multiplexing.

5.14.7 ADC SRAM CRC-64 Testing

The MibADC SRAM contents can be tested periodically using the hardware CRC-64 diagnostic. As MibADC SRAM contents tend to be more dynamic, use of this diagnostic is optional. For more details on this diagnostic, see SRAM Hardware CRC-64.

5.14.8 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration registers mechanism is recommended.

5.14.9 Notes

- ADC module voltages should be supervised as noted in Power Supply

5.15 Multi Buffered Serial Peripheral Interface (MIBSPI)

The MibSPI modules provide serial I/O compliant to the MibSPI protocol. MibSPI communications are typically used for communication to smart sensors and actuators, serial memories, and external logic such as a watchdog device. The MibSPI modules contain internal SRAM buffers. If not used for MibSPI communication, the MibSPI modules support the usage of their I/O as general purpose I/O.

5.15.1 Software Test of Function Using I/O Loopback

A software test can be utilized to inject diagnostic errors and check for proper error response. Such a test can be executed at boot or periodically. Software requirements necessary are defined by the software implemented by the system integrator. The use of a boot time software test of basic functionality is highly recommended. The use of a periodic software test of basic functionality reporting is optional.

The MibSPI implementation supports both digital and analog loopback capabilities for the I/Os. Digital loopback tests the signal path to the module boundary. Analog loopback tests the signal path from the module to the I/O cell with output driver disabled. For best results any tests of the MibSPI functionality should include the I/O loopback.

5.15.2 Message Parity

The Hercules TMS470M MIBSPI supports insertion of a parity bit into the data payload of every outgoing MIBSPI message by hardware. Evaluation of incoming message parity is also supported by hardware. Detected errors generate an interrupt to the CPU. The use of this feature is highly recommended.

5.15.3 Information Redundancy Techniques

Information redundancy techniques can be applied via software as an additional runtime diagnostic for MIBSPI communication. There are many techniques that can be applied, such as read back of written values and multiple reads of the same target data with comparison of results. Alternatively redundancy can be achieved by implementation of multiple channels in the system. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of information redundancy techniques in MIBSPI transactions is highly recommended.

5.15.4 MIBSPI SRAM Parity

The MIBSPI SRAM includes a parity diagnostic that can detect single bit errors in the memory. When a parity error is detected the ESM is notified. This feature is disabled after reset. Software must configure and enable this feature. Use of the MIBSPI SRAM parity feature is highly recommended.

5.15.5 MIBSPI SRAM MBIST

The MIBSPI SRAM can be tested using the MBIST memory BIST engine. Execution of MBIST tests on MIBSPI SRAM after reset is highly recommended. For more details on this diagnostic, see PBIST.

5.15.6 MIBSPI SRAM Bit Multiplexing

The MIBSPI SRAM is implemented with a memory design such that logically adjacent bits are not physically adjacent. Use of this safety mechanism is mandatory. For more details on this safety mechanism, see SRAM Bit Multiplexing.

5.15.7 MIBSPI SRAM CRC-64 Testing

The MIBSPI SRAM contents can be tested periodically using the hardware CRC-64 diagnostic. As MIBSPI SRAM contents tend to be more dynamic, use of this diagnostic is optional. For more details on this diagnostic, see SRAM Hardware CRC-64.

5.15.8 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration registers mechanism is recommended.

5.15.9 Notes

- It is possible to use the MIBSPI in standard SPI mode.

5.16 Local Interconnect Network (LIN)

The LIN module provides serial I/O compliant to the LIN protocol. LIN is a low throughput time triggered protocol. The module can also be configured in SCI mode and used as a general purpose serial port. An external transceiver is used for LIN communications.

5.16.1 Software Test of Function Using I/O Loopback

A software test can be utilized to inject diagnostic errors and check for proper error response. Such a test can be executed at boot or periodically. Software requirements necessary are defined by the software implemented by the system integrator. The use of a boot time software test of basic functionality is highly recommended. The use of a periodic software test of basic functionality reporting is optional.

The LIN implementation supports both digital and analog loopback capabilities for the I/Os. Digital loopback tests the signal path to the module boundary. Analog loopback tests the signal path from the module to the I/O cell with output driver disabled. For best results any tests of the LIN functionality should include the I/O loopback.

5.16.2 Information Redundancy Techniques Including End-to-End Safing

Information redundancy techniques can be applied via software as an additional runtime diagnostic for LIN communication. There are many techniques that can be applied, such as read back of written values and multiple reads of the same target data with comparison of results.

In order to provide diagnostic coverage for network elements outside the MCU (wiring harness, connectors, transceiver) end-to-end safing mechanisms are applied. These mechanisms can also provide diagnostic coverage inside the MCU. There are many different schemes applied, such as additional message checksums, redundant transmissions, time diversity in transmissions, and so forth. Most commonly checksums are added to the payload section of a transmission to ensure the correctness of a transmission. These checksums are applied in addition to any protocol level parity and checksums. As the checksum is generated and evaluated by the software at either end of the communication, the whole communication path is safed, resulting in end-to-end safing.

Error response, diagnostic testability, and any necessary software requirements are defined by the system integrator. Use of this mechanism is highly recommended.

5.16.3 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration registers mechanism is recommended.

5.17 Controller Area Network (DCAN)

The DCAN interface provides medium throughput networking with event based triggering, compliant to the CAN protocol. The DCAN modules requires an external transceiver to operate on the CAN network.

5.17.1 Software Test of Function Using I/O Loopback

A software test can be utilized to inject diagnostic errors and check for proper error response. Such a test can be executed at boot or periodically. Software requirements necessary are defined by the software implemented by the system integrator. The use of a boot time software test of basic functionality is highly recommended. The use of a periodic software test of basic functionality reporting is optional.

The DCAN implementation supports both digital and analog loopback capabilities for the I/Os. Digital loopback tests the signal path to the module boundary. Analog loopback tests the signal path from the module to the I/O cell with output driver disabled. For best results any tests of the DCAN functionality should include the I/O loopback.

5.17.2 Information Redundancy Techniques Including End-to-End Safing

Information redundancy techniques can be applied via software as an additional runtime diagnostic for CAN communication. There are many techniques that can be applied, such as read back of written values and multiple reads of the same target data with comparison of results.

In order to provide diagnostic coverage for network elements outside the MCU (wiring harness, connectors, transceiver) end-to-end safing mechanisms are applied. These mechanisms can also provide diagnostic coverage inside the MCU. There are many different schemes applied, such as additional message checksums, redundant transmissions, time diversity in transmissions, and so forth. Most commonly checksums are added to the payload section of a transmission to ensure the correctness of a transmission. These checksums are applied in addition to any protocol level parity and checksums. As the checksum is generated and evaluated by the software at either end of the communication, the whole communication path is safed, resulting in end-to-end safing.

Error response, diagnostic testability, and any necessary software requirements are defined by the system integrator. Use of this mechanism is highly recommended.

5.17.3 DCAN SRAM Parity

The DCAN SRAM includes a parity diagnostic that can detect single bit errors in the memory. When a parity error is detected the ESM is notified. This feature is disabled after reset. Software must configure and enable this feature. Use of the DCAN SRAM parity feature is highly recommended.

5.17.4 DCAN SRAM MBIST

The DCAN SRAM can be tested using the MBIST memory BIST engine. Execution of MBIST tests on DCAN SRAM after reset is highly recommended. For more details on this diagnostic, see PBIST.

5.17.5 DCAN SRAM Bit Multiplexing

The DCAN SRAM is implemented with a memory design such that logically adjacent bits are not physically adjacent. Use of this safety mechanism is mandatory. For more details on this safety mechanism, see SRAM Bit Multiplexing.

5.17.6 DCAN SRAM CRC-64 Testing

The DCAN SRAM contents can be tested periodically using the hardware CRC-64 diagnostic. As DCAN SRAM contents tend to be more dynamic, use of this diagnostic is optional. For more details on this diagnostic, see SRAM Hardware CRC-64.

5.17.7 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration registers mechanism is recommended.

5.18 General-Purpose Input/Output (GIO)

The GIO module provides digital input capture and digital input/output. There is no processing function in this block. The GIO is typically used for static or rarely changed outputs, such as transceiver enable signals, warning lights, and so forth. The GIO can also be used to provide external interrupt input capabilities.

5.18.1 Software Test of Function Using I/O Checking

A software test can be utilized to inject diagnostic errors and check for proper error response. Such a test can be executed at boot or periodically. Software requirements necessary are defined by the software implemented by the system integrator. The use of a boot time software test of basic functionality is highly recommended. The use of a periodic software test of basic functionality reporting is optional.

The GIO module does not support a distinct I/O loopback mode. However it is possible to support I/O checking using normal functionality. To do this software generates output and reads back and checks for the same value in the input registers. This implements functionality similar to analog loopback in other modules. For best results, any tests of the GIO functionality should include the I/O loopback.

5.18.2 Information Redundancy Techniques

Information redundancy techniques can be applied via software as an additional runtime diagnostic on GIO function. There are many techniques that can be applied, such as multiple inputs and read back of output with an input channel. Signals from many other peripherals can be used as GIO if not used for primary function. Use of a GIO module signal and a non GIO module signal for multi-channel implementation can reduce probability of common mode failures.

Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of information redundancy techniques on GIO functions is highly recommended.

5.18.3 Periodic Read Back of Configuration Registers

Periodic read back of configuration registers can provide a diagnostic for inadvertent writes or disturb of these registers. Error response, diagnostic testability, and any necessary software requirements are defined by the software implemented by the system integrator. The use of read back of configuration registers mechanism is recommended.

5.18.4 Notes

- To reduce probability of common mode failure, user should consider implementing multiple channels using non adjacent pins.

5.19 JTAG Debug and Test Access

The Hercules TMS470M platform supports debug and test implemented over an IEEE 1149.1 JTAG debug port. The physical debug interface is internally connected to a TI debug multiplexor logic (ICEPICK), which arbitrates access to test, debug, and calibration logic. Boundary scan is connected in parallel to the ICEPICK to support usage without preamble scan sequences for easiest manufacturing board test.

5.19.1 Hardware Disable of JTAG Port

The JTAG debug port can be physically disabled to prevent JTAG access in deployed systems. The recommended scheme is to hold test clock (TCK) to ground and hold test mode select (TMS) high, though alternate schemes are possible. Hardware disable of the JTAG port is recommended.

5.19.2 Notes

- A watchdog may provide indication of inadvertent activation.

5.20 Cortex-M3 Central Processing Unit (CPU) Debug

The Hercules TMS470M platform supports CPU debug compliant to the ARM CoreSight standard. Each CoreSight element is accessible over a memory-mapped debug bus, which can be accessed by the CPU or the JTAG port. The CPU debug logic includes an independent debug bus master (AHB-AP), the debug unit inside the CPU. This module is not recommended to be used during safety critical operation and safety mechanisms are in place to disable this logic.

5.20.1 Disable JTAG Port to Limit Functional Access

Most debug and trace activities are initiated via an external debug tool, which writes commands to the device using the JTAG port. The JTAG port can be disabled on production hardware as described in JTAG Debug/Trace/Calibration/Test Access. Disabling the JTAG port to limit debug module access is highly recommended.

5.20.2 Block Access to Memory Mapped Debug

The CoreSight debug peripherals are accessible over a memory-mapped debug bus. Access to this region can be blocked via the use of bus master based memory protection. For more information on memory protection, see CPU Memory Protection Unit (MPU). Blocking of access to memory-mapped debug components is highly recommended.

5.20.3 CoreSight Debug Logic Key Enables

To enable operation of the memory-mapped CoreSight debug components, it is necessary to write a defined 32-bit key to an unlock register in each debug module. This debug lock protection provides an additional safety mechanism to limit undesired activation. Use of the debug module unlock key is highly recommended.

5.20.4 Notes

- A watchdog may provide indication of inadvertent activation.
- Not all package variants may include trace functionality

6 Next Steps in Your Safety Development

TI's support for your safety development does not stop with the delivery of this safety document. Customers have a wide range of support options, such as:

- Access Hercules TMS470M documentation including safety docs and application reports any time on the web: http://www.ti.com/hercules
- Discuss questions and concerns with TI experts and other Hercules TMS470M developers using the TI Connected Community (E2E forums): http://www.ti.com/hercules-support
- The Hercules TMS470M Wiki page provides answers to many commonly asked questions: http://www.ti.com/hercules-wiki

Feedback and concerns about the safety manual are always welcome and can be submitted via the web by clicking on the feedback link at the bottom of each page of this document.

Appendix A Summary of Recommended Safety Feature Usage

Table 2 provides a summary of the safety concept recommendations noted in Section 5 and organized by device partition. Each recommendation is given a unique identifier to aid in requirements management. For each safety feature or diagnostic the recommendation is noted in simplified form as follows:

- M --> Mandatory application
- ++ --> highly recommended
- + --> recommended
- O --> optional

In addition, a list of possible latent diagnostic measures for each device partition and safety feature and diagnostic combination is provided. For details on each safety feature or diagnostic, see Section 5.

Table 2. Summary of Safety Features and Diagnostics

Device Partition	Unique Identifier	Safety Feature or Diagnostic	Feature Recommendation	Possible Latent Diagnostics
Power Supply + VREG	PWR1	VREG embedded core voltage monitor	M	External Voltage Supervisor
	PWR2	External voltage supervisor	++	Core Voltage Monitor
Clocks	CLK1	LPOCLKDET	++	ECLK, watchdog
	CLK2	PLL slip detector	++	ECLK, watchdog
	CLK3	External monitoring via ECLK	O	LPOCLKDET, PLL slip detector, watchdog
	CLK4A	Internal watchdog - DWD	+	External watchdog, software test of watchdog configuration and error response
	CLK4B	External watchdog	++	Internal watchdog, software test of watchdog configuration and error response
	CLK5	Periodic software readback of static clock configuration registers	+	LBIST
	CLK6	Software readback of written configuration	++	LBIST
Reset	RST1	External monitoring of warm reset	O	Watchdog
	RST2	Software check of last reset	++	LBIST
	RST3	Software warm reset generation	O	LBIST
	RST4	Glitch filtering on reset pins	M	External Watchdog
	RST5	Use of status shadow registers	++	LBIST
	RST6	External watchdog	++	Software test of external watchdog configuration and error response
	RST7	Periodic software readback of static configuration registers	+	LBIST
	RST8	Software readback of written configuration	++	LBIST
System Control	SYS1	Privileged mode access and multi-bit enable keys	M	Software test of register configuration and error response
	SYS2	Software readback of written configuration	++	LBIST
	SYS3	Periodic software readback of static configuration registers	+	LBIST
Error Signaling Module (ESM)	ESM1	Periodic software readback of static configuration registers	+	LBIST
	ESM2A	Boot time software test of error path reporting	++	LBIST
	ESM2B	Periodic software test of error path reporting	+	LBIST
	ESM3	Use of status shadow registers	++	LBIST
	ESM4	Software readback of written configuration	++	LBIST

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

Device Partition	Unique Identifier	Safety Feature or Diagnostic	Feature Recommendation	Possible Latent Diagnostics
Cortex-M3Central Processing Unit (CPU)	CPU1A	Boot time execution of LBIST STC	++	LBIST autocoverage
	CPU1B	Periodic execution of LBIST STC	+	LBIST autocoverage
	CPU2	MPU	++	LBIST
	CPU3A	Internal watchdog - DWD	+	External watchdog, software test of watchdog configuration and error response
	CPU3B	External watchdog	++	Internal watchdog, software test of watchdog configuration and error response
	CPU4	Illegal operation and instruction trapping	++	LBIST
	CPU5	Software readback of written configuration	++	LBIST
Embedded Flash	FLA1	Flash Data ECC	++	LBIST, ECC autocoverage
	FLA2A	Boot time hardware CRC check of Flash memory contents	++	CRC autocoverage
	FLA2B	Periodic hardware CRC check of Flash memory contents	+	CRC autocoverage
	FLA3	Bit multiplexing in Flash array	M	ECC, CRC test
	FLA4	Flash sector protection	++	CRC test
	FLA5	Periodic software readback of static configuration registers	+	LBIST
	FLA6	Software readback of written configuration	++	LBIST
Flash Emulated EEPROM (FEE)	FEE1	Flash Data ECC	++	LBIST, ECC autocoverage
	FEE2A	Boot time hardware CRC check of Flash memory contents	++	CRC autocoverage
	FEE2B	Periodic hardware CRC check of Flash memory contents	+	CRC autocoverage
	FEE3	Bit multiplexing in Flash array	M	ECC, CRC test
	FEE4	Flash sector protection	++	CRC test
	FEE5	Periodic software readback of static configuration registers	+	LBIST
	FEE6	Software readback of written configuration	++	LBIST
Primary Embedded SRAM	RAM1	Data ECC	++	LBIST, ECC autocoverage, MBIST
	RAM2	Correctable ECC profiling	+	LBIST
	RAM3A	Boot time MBIST check of RAM	++	MBIST autocoverage
	RAM3B	Periodic MBIST check of RAM	O	MBIST autocoverage
	RAM4	Bit multiplexing in SRAM array	M	ECC
	RAM5	Periodic hardware CRC check of SRAM contents	O	CRC autocoverage
	RAM6	Periodic software readback of static configuration registers	+	CRC autocoverage
	RAM7	Software readback of written configuration	++	LBIST
Interconnect Subsystem	INC1	Error trapping	M	Software test of basic functionality and error response
	INC2	PCR access management	++	Software test of basic functionality and error response
	INC3A	Internal watchdog - DWD	O	External watchdog, software test of watchdog configuration and error response
	INC3B	External watchdog	++	Internal watchdog, software test of watchdog configuration and error response
	INC4	Information redundancy	+	
	INC5	Periodic software readback of static configuration registers	+	LBIST
	INC6A	Boot time software test of basic functionality	++	LBIST
	INC6B	Periodic software test of basic functionality	+	LBIST
	INC7	Software readback of written configuration	++	LBIST

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

Device Partition	Unique Identifier	Safety Feature or Diagnostic	Feature Recommendation	Possible Latent Diagnostics
M3 Vectored Interrupt Module (VIM)	VIM1	Periodic software test of VIM functionality	++	LBIST
	VIM2	Periodic software readback of static configuration registers	+	LBIST
	VIM3	Software readback of written configuration	++	LBIST
	VIM4A	Internal watchdog - DWD	O	External watchdog, software test of watchdog configuration and error response
	VIM4B	External watchdog	++	Internal watchdog, software test of watchdog configuration and error response
Real Time Interrupt (RTI) Operating System Timer	RTI1	Use of 2nd counter as diagnostic	+	LBIST
	RTI2	Internal watchdog - DWD	O	External watchdog, software test of watchdog configuration and error response
	RTI2B	External watchdog	++	Internal watchdog, software test of watchdog configuration and error response
	RTI3	Periodic software readback of static configuration registers	+	LBIST
High-End Timer (HET)	HET1A	Boot time software test of function using I/O loopback	++	LBIST
	HET1B	Periodic software test of function using I/O loopback	O	LBIST
	HET2	HET SRAM Data Parity	++	MBIST
	HET3A	Boot time MBIST check of HET RAM	++	MBIST autocoverage
	HET3B	Periodic MBIST check of HET RAM	O	MBIST autocoverage
	HET4	Bit multiplexing in HET RAM array	M	MBIST, parity
	HET5	Periodic hardware CRC check of HET SRAM contents	O	MBIST
	HET6	Periodic software readback of static configuration registers	+	LBIST
Multi-Buffered Analog to Digital Converter (MibADC)	ADC1	Boot time input self test	++	LBIST
	ADC2A	Boot time converter calibration	++	LBIST
	ADC2B	Periodic converter calibration	O	LBIST
	ADC3	Information redundancy techniques	++	ADC converter calibration, ADC self-test
	ADC4	MibADC SRAM Data Parity	++	MBIST
	ADC5A	Boot time MBIST check of MibADC RAM	++	MBIST autocoverage
	ADC5B	Periodic MBIST check of MibADC RAM	O	MBIST autocoverage
	ADC6	Bit multiplexing in MibADC RAM array	M	MBIST, parity
	ADC7	Periodic hardware CRC check of MibADC SRAM contents	O	MBIST
	ADC8	Periodic software readback of static configuration registers	+	LBIST
Multi-Buffered Serial Peripheral Interface (MibSPI)	MSP1A	Boot time software test of function using I/O loopback	++	LBIST
	MSP1B	Periodic software test of function using I/O loopback	O	LBIST
	MSP2	Parity in message	++	LBIST
	MSP3	Information redundancy techniques	++	LBIST
	MSP4	MibSPI SRAM Data Parity	++	MBIST
	MSP5A	Boot time MBIST check of MibSPI RAM	++	MBIST autocoverage
	MSP5B	Periodic MBIST check of MibSPI RAM	O	MBIST autocoverage
	MSP6	Bit multiplexing in MibSPI RAM array	M	MBIST, parity
	MSP7	Periodic hardware CRC check of MibSPI SRAM contents	O	MBIST
	MSP8	Periodic software readback of static configuration registers	+	LBIST

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

Device Partition	Unique Identifier	Safety Feature or Diagnostic	Feature Recommendation	Possible Latent Diagnostics
Local Interconnect Network (LIN)	LIN1A	Boot time software test of function using I/O loopback	++	LBIST
	LIN1B	Periodic software test of function using I/O loopback	O	LBIST
	LIN2	Information redundancy techniques including end to end safing	++	LBIST
	LIN3	Periodic software readback of static configuration registers	+	LBIST
Controller Area Network (DCAN)	CAN1A	Boot time software test of function using I/O loopback	++	LBIST
	CAN1B	Periodic software test of function using I/O loopback	O	LBIST
	CAN2	Information redundancy techniques including end to end safing	++	LBIST
	CAN3	DCAN SRAM Data Parity	++	MBIST
	CAN4A	Boot time MBIST check of DCAN RAM	++	MBIST autocoverage
	CAN4B	Periodic MBIST check of DCAN RAM	O	MBIST autocoverage
	CAN5	Bit multiplexing in DCAN RAM array	M	MBIST, parity
	CAN6	Periodic hardware CRC check of DCAN SRAM contents	O	MBIST
	CAN7	Periodic software readback of static configuration registers	+	LBIST
General Purpose Input/Output (GIO)	GIO1A	Boot time software test of function using I/O checking	++	LBIST
	GIO1B	Periodic software test of function using I/O checking	O	LBIST
	GIO2	Information redundancy techniques	++	LBIST
	GIO3	Periodic software readback of static configuration registers	+	LBIST
Joint Technical Action Group (JTAG) Debug/Trace/ Calibration Access	JTG1	Hardware disable of JTAG port	+	Watchdog detection of inadvertent activation that impacts program flow
Cortex-M3 Central Processing Unit Debug	DBG1	Hardware disable of JTAG port	+	Watchdog detection of inadvertent activation that impacts program flow
	DBG2	Use MPU to block access to memory-mapped debug	++	Watchdog detection of inadvertent activation that impacts program flow
	DBG3	Use of CoreSight debug logic key enable scheme	++	Watchdog detection of inadvertent activation that impacts program flow

Functional Safety Disclaimer for Safety Critical Solutions

TI's safety critical solutions, including integrated circuits, software and tools help TI's customers create end products that may be used in appropriately designed safety-critical applications to comply with functional safety standards or requirements.

Buyers represent and agree that they have all the necessary expertise to design, manage and assure effective system-level safeguards to anticipate, monitor and control system failures in safety-critical applications. Buyers agree and accept sole responsibility to meet and comply with all applicable regulatory standards and safety-related requirements concerning their systems and end-products which use TI's safety-critical applications. Buyers will fully indemnify TI and its representatives against any damages arising out of the use of TI products in safety-critical applications.

TI integrated circuits are not authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

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