Application Note

Obtain UL/IEC 60730-1/60335-1 Class B Certification Based on C2000™ MCU Diagnostic Library in Appliances

Jim Chen *System Engineering & Marketing*

ABSTRACT

As technologies development, the functionality of appliances becomes increasingly powerful, leading to a growing complexity in electrical systems. In response to the market demand for safer, smarter, and more energy-efficient appliances, a series of worldwide standards were introduced, for example UL/IEC 60730-1/60335-1. To meet the functionality demands of household appliances and obtain worldwide recognition for safety and quality, the microcontroller (MCU) must comply with the requirements of UL/IEC 60730-1/60335-1.

Texas Instruments (TI) boasts a diverse range of MCU product lines, and TI's C2000™ MCU series stands out with abundant peripheral resources, robust CPU core, and cost-effectiveness, which are widely used in appliance applications. Also, TI provides a complete and mature C2000 MCU diagnostic library to meet the requirements of the UL/IEC 60730-1/60335-1 Class B.

This application note mainly focuses on various specifications of C2000 MCUs for appliances, the software libraries compliant with UL/IEC 60730-1/60335-1. By providing insights into these aspects, this document aims to assist engineers in swiftly selecting the most cost-effective C2000 MCU design based on both system functionality requirements and UL/IEC 60730-1/60335-1 certification needs.

Table of Contents

List of Figures

List of Tables

Trademarks

C2000™ is a trademark of Texas Instruments.
Arm® and Cortex® are registered trademarks of Arm Limited.
All trademarks are the property of their respective owners.

1 Introduction

As technologies development, the functionality of appliances becomes increasingly powerful, leading to a growing complexity in the electrical systems. Consequently, MCUs which possess numerous features and ample resources, play a pivotal role in this domain. In response to the market demand for safer, smarter, and more energy-efficient household appliances, a series of worldwide standards were introduced. The majority of these standards are built upon the foundation of IEC/UL 60730-1, which specifically addresses the MCU and the automatic control component in household appliances. IEC/UL 60335-1, conversely, serves as the safety standard for appliances, with the MCU section directly referencing the requirements set forth in IEC/UL 60730-1.

UL certification enjoys high visibility and recognition globally, and VDE certification is regarded as a hallmark of quality for electrical products especially in the European market. The evaluation criteria for MCU in both UL and VDE certifications align closely with those specified in IEC/UL 60730-1, albeit with higher requirements in certain aspects emphasized by VDE.

Texas Instruments (TI) boasts a diverse range of MCU product lines, and TI's C2000 MCU series stands out with abundant peripheral resources, robust central processing unit (CPU) core, and cost-effectiveness. Additionally, the series is accompanied by a comprehensive diagnostic library, which enables the fulfillment of appliances functionality requirements and UL/IEC 60730-1/60335-1 certification requirement at a low cost and within a short development cycle.

2 Overview of C2000™ MCU Devices in Appliances

In the realm of appliance systems, the required MCU computing power, memory and resources differ for different functionalities. TI's C2000 MCU series products, ranging from the entry-level 32-pin 64KB Flash F280013x to the 256-pin 1.28MB Flash F28P65x, cover nearly all MCU application needs in appliance systems. Additionally, F280015x has a lock step for the safety requirement. The detailed C2000 MCU resources comparison for both entry-level and performance-level in appliance applications is found in the *Implementing IEC 60730 / UL 1998 Compliance for C2000 Real-Time Microcontrollers* user's guide.

The C28x core of the third-generation C2000 MCU, is equipped with a Floating-point Unit (FPU), enabling fast floating-point operations, while the TMU supports rapid trigonometric function calculations. Compared to the Arm® Cortex®-M7 core, the frequency of the C28x core is equivalent to twice the frequency capability of the corresponding Arm core due to optimized pipeline structure. With F280013x as an example, the 120MHz frequency capability is equivalent to that of a 240MHz Arm core.

The ePWM module in the C2000 MCU boasts comprehensive and mature function configurations, supporting high-precision PWM modes to deliver accurate and reliable drive control signals. Additionally, all C2000 MCUs come with multiple independent ADC modules, allowing for synchronous sampling to enhanced sampling speed. The built-in comparators can be easily configured to protect AD signals, with protection thresholds set by the internal DAC, providing quick, sensitive, and reliable protection function.

The *Real-time Benchmarks Showcasing C2000™ Control MCU's Optimized Signal Chain* application note describes a real-time benchmark created around a real-world control application that highlights the intricacies of real-time control and the need for this more comprehensive benchmarking approach.

When selecting a C2000 MCU based on system functionalities, look at the example of an outdoor air conditioning unit. The MCU selection primarily revolves around the basic control requirements of the driver board, which can be categorized as follows: single motor control, single motor + digital PFC control, dual motor + digital PFC control, and triple motor + PFC control.

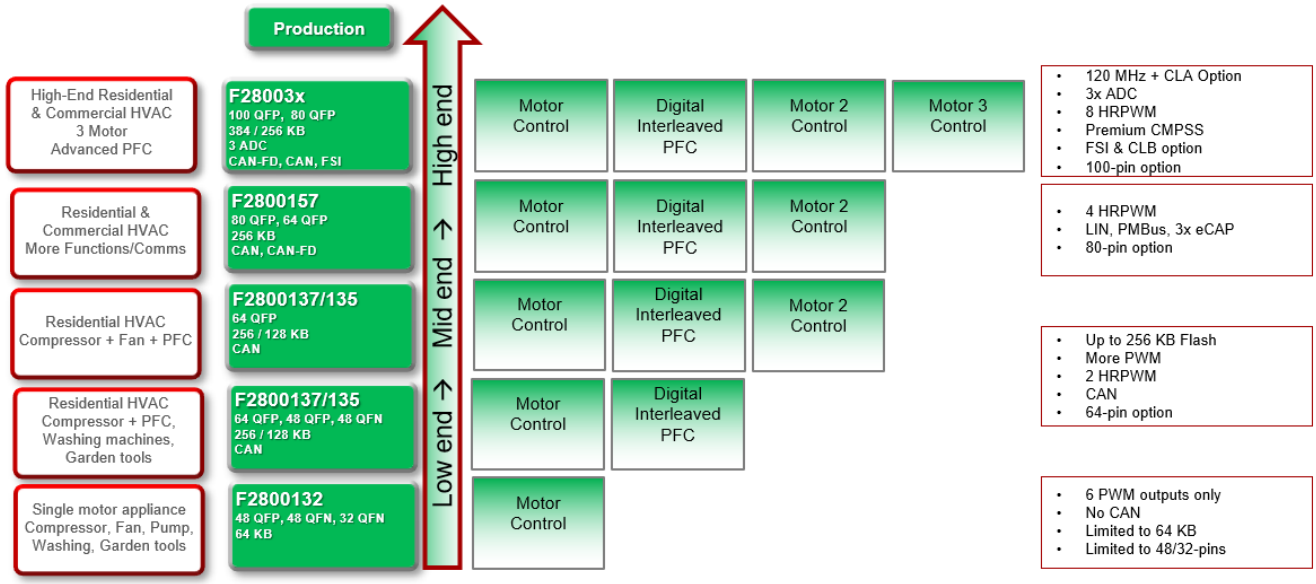

Figure 2-1. Select C2000™ MCU Based on System Control Requirements

In the TIDA-010265 reference design for single-motor control, a self-test program that meets the requirements of IEC/UL 60730-1 is added to the software, which is found in `C2000Ware_MotorControl_SDK_x_0x_00_00\solutions\tida_010265_wminv`.

3 Introduction of IEC/UL 60730-1/60335-1 Standards

UL 60335 is a safety standard designed for household and similar electrical appliances and developed by Underwriters Laboratories (UL), a reputable American electrical safety certification organization. The primary goal of this standard is to ensure the safety and well-being of users. UL 60335 is a globally recognized standard that many countries and regions adopt or refer to for the safety review and certification of electrical products.

UL 60335 electrical standard encompasses a range of safety requirements for electrical products, including but not limited to product structure and materials, electrical connections and insulation, heating and mechanical strength, temperature, and electrical parameters. This application note primarily focuses on the safety requirements related to MCU. The reference appendix R in UL 60335-1 lists the software evaluation requirements, with the majority of MCU control failure self-test requirements almost entirely based on IEC 60730-1, including the regulations referenced for each safety requirement. Therefore, the following discussion is about IEC 60730-1.

IEC 60730 is an internationally recognized standard established by the International Electrotechnical Commission (IEC), which specifies safety and functional requirements for automatic electrical controls. This standard primarily applies to household and similar appliances and electronic devices, including but not limited to household appliances, lighting equipment, air conditioning systems, and power tools. Analyzing the IEC 60730 standard from a professional standpoint, the following aspects can be evaluated:

1. Device Safety Requirements: IEC 60730 imposes clear safety requirements on automatic control devices, encompassing electrical safety, mechanical safety, fire safety, and other aspects. These requirements aim to make sure that devices do not pose harm to humans or the environment during regular operation.
2. Device Functionality Requirements: IEC 60730 also defines requirements for the functional performance of automatic control devices, including measurement, detection, control, and regulation. These requirements ensure that devices can meet user needs and exhibit stable and reliable performance.
3. Device Reliability Requirements: IEC 60730 sets reliability requirements for automatic control devices, encompassing device lifespan, fault diagnosis, fault recovery, and other aspects. These requirements make sure that devices do not frequently malfunction during long-term use and can effectively handle faults in a timely manner.
4. Device Marking and Labeling Requirements: IEC 60730 also regulates the marking and labeling requirements for automatic control devices, including device model, serial number, safety-related information, and so forth. These requirements help users utilize devices correctly and promptly understand the safety performance.

In conclusion, the IEC 60730 standard provides detailed requirements from a professional standpoint for the safety, functional performance, reliability, marking, and labeling of automatic control devices. The standard helps make sure that devices can achieve the expected safety and functional performance during the design, manufacturing, usage, and maintenance processes. This standard plays a crucial guiding role in the automatic electrical control equipment industry, providing manufacturers, designers, and users with a shared benchmark to adhere to.

IEC 60730 defines 3 classes:

1. Class A: functions such as room thermostats, humidity controls, lighting controls, timers and switches. These are distinguished by not being relied upon for the safety of the equipment.
2. Class B: functions such as thermal cutoffs are intended to prevent unsafe operation of appliances such as washing machines, dishwashers, dryers, refrigerators, freezers, and cookers or stoves.
3. Class C: functions are intended to prevent special hazards such as explosions. These include automatic burner controls and thermal cutouts for closed, unvented water heaters.

Safety requirements in household appliance applications primarily revolve around Class B. As a result, subsequent standard requirements and testing methods are introduced and analyzed based on Class B.

IEC 60730-1 has outlined specific specifications for various types of commodities, with requirements for electronic control devices listed in reference Appendix H. In Appendix H. H.11.12.7 provides detailed descriptions of components that must be tested, with the specific tests depending on the software classification. The types of faults and testing methods for each component are listed in Table H.1. Table 3-1 presents all the items that need to be considered for MCU in appliances according to the IEC-60730-1 Class B requirements.

Table 3-1. Class B Failure Modes of IEC 60730-1 Table H.1

Component to be Tested		Hardware Fault | Error to Detect
		Class B
1. CPU	1.1 Register	Stuck-at
	1.2 Instruction decode and execution	N/A
	1.3 Program counter	Stuck-at
	1.4 Addressing	N/A
	1.5 Data paths	N/A
2. Interrupts		None or too frequent
3. Clock		Wrong frequent
4. Memory	4.1 Non-volatile	All single bit faults
	4.2 Volatile	DC fault
	4.3 Addressing	Stuck at
5. Internal data path	5.1 Data	Stuck-at
	5.2 Addressing	Wrong address
6. External communication	6.1 Data	All single-bit and double bit errors
	6.2 Addressing	Wrong address
	6.3 Timing	Wrong point in time
		Wrong sequence
7. Input/output periphery	7.1 Digital I/O	Open and short circuit or as specified in the product standard
	7.2 Analog I/O 7.2.1 A/D and D/A converter	Open and short circuit or as specified in the product standard
	7.2 Analog I/O 7.2.2 Analog multiplexer	Wrong addressing

Items 1, 3, 4, 5, and 7 mainly pertain to the functional aspects of the MCU, while items 2 and 6 correspond to the test requirements for application functionality. Regarding the fault detection for application functionality, it is necessary to add corresponding test code based on the software structure and specific requirements of the certification agencies. For the MCU-related fault detection, TI's C2000 series MCU provides the principles and examples for all the relevant test requirements.

In conclusion, by incorporating the required C2000 self-test libraries into the application program, most appliances can quickly meet the Class B test requirements of IEC/UL 60730-1, also satisfying those of IEC/UL 60335-1 safety certification.

4 Diagnostic Libraries for UL/IEC 60730-1/60335-1 Provided by C2000™

The requirements for safety standards are almost uniform, but the understanding and requirements of various certification agencies are different. Therefore, TI provides a mature diagnostic library for the MCU section in the standards, so that engineers can make library calls according to the specific requirements of the agency. To meet the testing requirements of IEC/UL 60730-1/60335-1, a round of self-test is usually performed after power-on or reset, and periodic self-tests are performed after the application program is functioning properly. Figure 4-1 shows the commonly-used self-test software structure.

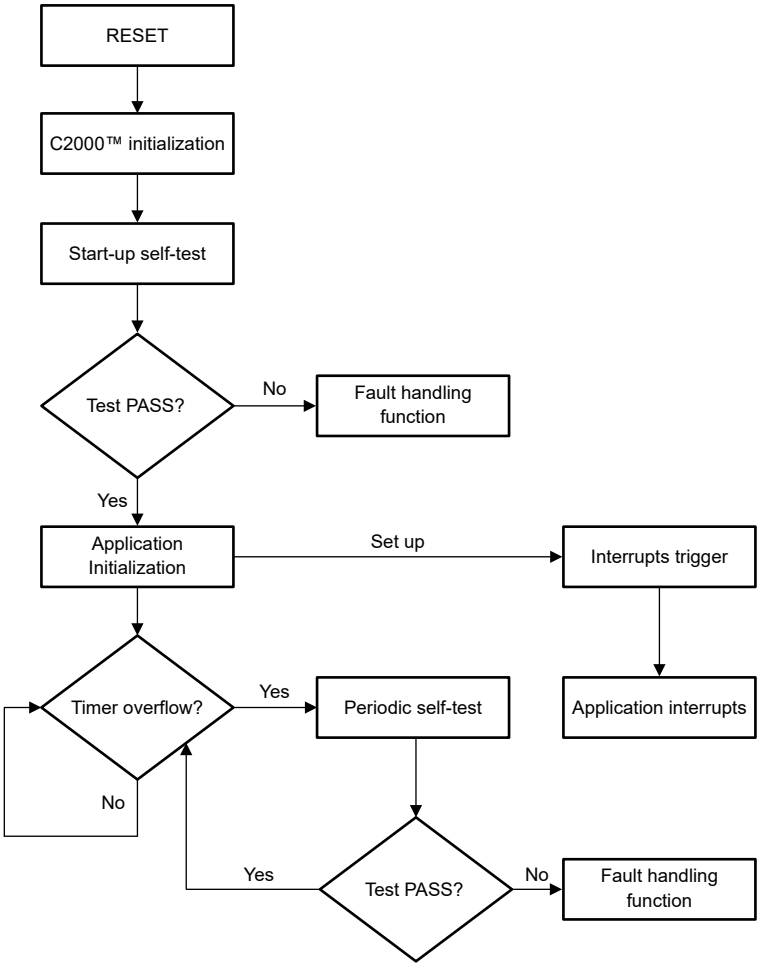

Figure 4-1. Commonly Used Self-Test Software Structure

After power-on, when the MCU initialization is completed, a round of self-test is performed, including Stack Overflow Detection, watchdog test, CPU and FPU registers test, PC test, Clock test, RAM test (March13N), Flash test (CRC or ECC), ADC test, and so on. After the system functions are running, periodic self-tests are performed. The main difference between power-on detection and periodic cycle detection is in the stack and watchdog. Stack overflow detection is hardware-triggered and can be configured in the power-on detection. Watchdog detection only needs to be performed once during power-on detection because the watchdog is fed periodically while the program is running.

TI's C2000ware SDK provides the software diagnostic libraries related to the self-test of the previously-mentioned MCU. Taking F280013x as an example, the code can be found in `C2000Ware_x_0x_00_00\libraries\diagnostic\f280013x` and the example project is in `f280013x \examples\test_application`. To help engineers better understand the principles and methods of C2000 MCU self-test, and to facilitate the application of the required source code to the system project, and also for communication and confirmation with security certification agencies, detailed introductions follow for each self-test item.

4.1 Stack Overflow Detection

The enhanced bus comparator (EBC) unit in the ERAD module can monitor the internal address and data buses, and the EBC triggers the RTOSINT interrupt when a specified bus and mask matches a specified value. Hence, the basic approach for detecting stack overflow is to configure the EBC unit to trigger an interrupt when the data write address bus falls within some range prior to the end of a stack. Figure 4-2 illustrates this concept. Since this memory is reserved for stack use only, a data write within the specified address range indicates that the stack usage is approaching the allocated size limit. Detection of an impending stack overflow triggers a maskable interrupt. Programmed error response and any necessary software requirements are defined by the system integrator.

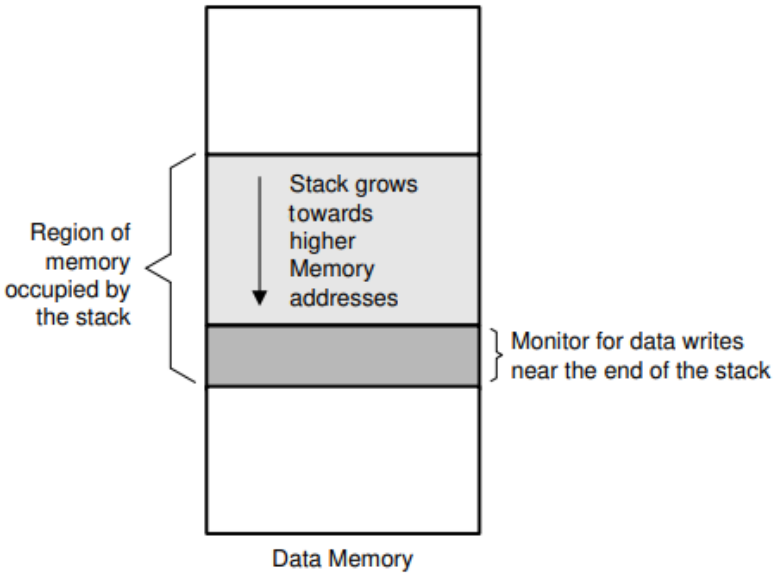

Figure 4-2. Stack Overflow Monitoring

The API function STL_SP_configSP(const STL_SP_Handle spHandle) is found in the example project, and the source code is in `f280013x\source\stl_sp.c`. The STL_SP_MARGIN parameter needs to be configured, with a reference value of 0x50. This parameter defines the memory space range observed from the end of the stack.

4.2 Watchdog

The main purpose of watchdog self-test is to verify whether the watchdog counter overflow can be triggered properly. During the self-test, the watchdog counter overflow trigger function is switched from reset to interrupt. By checking whether the program jumps to the watchdog interrupt within a certain period of time and counting the number of triggers, the normal functionality of the watchdog can be determined. Figure 4-3 shows the watchdog test structure. The example project is found in `f280013x\examples\ sdl_ex_watchdog`.

When the system is working normally, the watchdog reset mode needs to be enabled and the application interrupts need to feed the watchdog regularly. This can meet the test requirements of periodic watchdog test and PC pointer test at the same time.

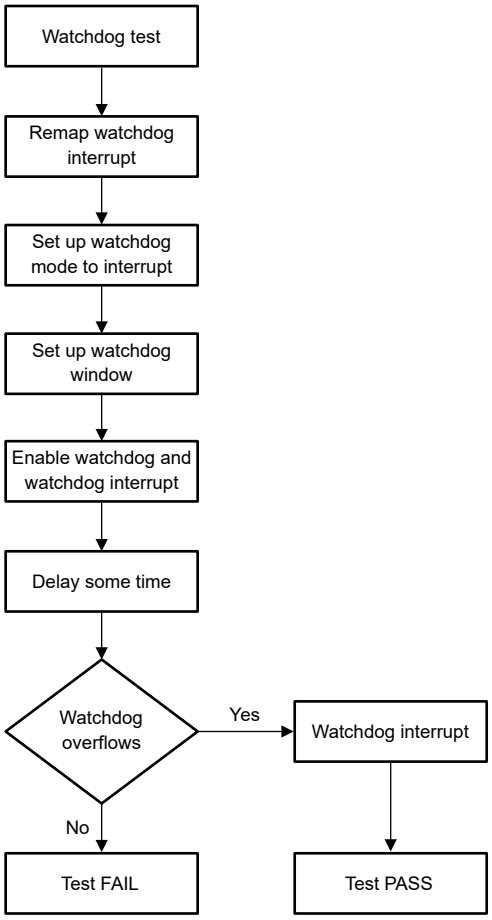

Figure 4-3. Watchdog Test Structure

4.3 CPU and FPU Registers

This function tests CPU core registers and FPU registers for stuck bits. The following CPU registers are tested: ACC, P, XAR0 to XAR7, XT, SP, IFR, IER, DBGIER, ST0, ST1 (excluding IDLESTAT and LOOP bits), and DP. The following FPU registers are tested: R0 to R7, RND32, TF, ZI, NI, ZF, NF bits of STF register, Shadow registers for R0 to R7 and STF. The values of ST0, ST1, DP, IER, IFR, and DBGIER and the save-on-entry XAR registers, STF and the save-on-entry RnH registers, as defined by the compiler calling convention, are saved and restored in this test. Disable interrupts before calling CPU and FPU test API.

The self-test of CPU and FPU is quite similar. First, the test pattern is set to a specific value, such as 0XAAAA, according to the requirements of the agency. During the test, the CPU and FPU register data needs to be stored in the stack, and then the test pattern is written into CPU and FPU registers. After reading the data of these registers, the data are verified whether the data match the preset test pattern. Finally, the CPU and FPU register data are restored from the stack, completing the register self-test of CPU or FPU. The test_application project contains corresponding cases for CPU and FPU self-test, namely STA_CPU_REG and STA_FPU_REG, with the testing functions as STL_CPU_REG_testCPURegisters(bool injectError) and STL_CPU_REG_testFPURegisters(bool injectError). Both functions consist of assembly code, the source code is found at `f280013x\source\stl_cpu_reg.asm`. Figure 4-4 shows the CPU and FPU registers test structure.

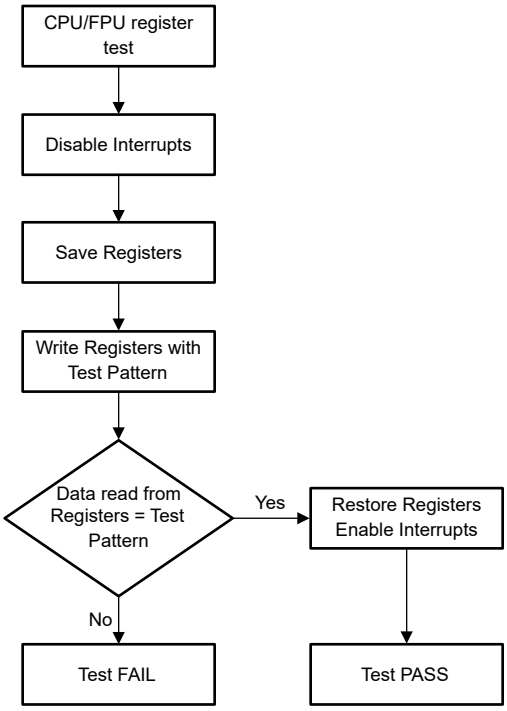

Figure 4-4. CPU, FPU Registers Test Structure

Remember that INTM being set does not prevent interrupts from propagating to the IFR registers, so although this function saves and restores IFR, an interrupt arriving in IFR during this test is cleared by the restore or can cause a failure of this test. If this test needs to be used during runtime, STL_CPU_REG_testCPURegisters() in stl_cpu_reg.asm can be modified to remove the IFR test (and related save and restore). Alternatively, disable all interrupts at the PIE level (PIEIER) or in the case of non-PIE interrupts (like CPU Timers 1 and 2) at the peripheral level and restore them after STL_CPU_REG_checkCPURegisters() completes. Also the IFR test part can be deleted, which is also acceptable for some certification agencies.

4.4 Program Counter (PC)

Traditionally, the detection of the PC pointer is achieved by jumping to a specified function address through the self-test program and then comparing the returned address with the preset address. If the addresses are the same, the self-check passes. The essence of this check is to verify the correctness of function jumps.

In actual applications, watchdogs are enabled and watchdog feeding operations are performed in fixed-frequency interrupts. Therefore, entering the watchdog interrupt to perform feeding indicates that the PC pointer is functioning properly. If the PC pointer fails, a reset caused by watchdog timer overflow occurs, achieving the same protective purpose. Therefore, the PC pointer is usually not individually self-checked. The detection of the PC pointer can be combined with enabling the watchdog and explained in the documentation provided to the certification agency.

4.5 Clock

In appliance applications, internal oscillators are usually used as clock sources. The C2000 MCU has two internal oscillators, which can be used to mutually verify the stability of the clock frequency. There are two methods for clock self-test in C2000 MCUs. One is using the HRPWM module, and the other is using the CPU Timer. Both test methods are found in the test_application project, STA_OSC_HR is for HRPWM and STA_OSC_CT is for CPU Timer.

Use the CPU Timer module to detect incorrect clock frequencies and drift between clock sources. CPU Timer2 has a programmable counter whose prescale value and clock source can be selected. The frequency relationship between selected clock and system clock can be determined by using the system clock as a reference time base. Common cause failures can be reduced by using different clock sources and different prescale values for the reference clock and measured clock.

The principle of using a CPU Timer for detection is that the system clock is usually set as the internal oscillator 2 (XTAL2), and the clock source of CPU Timer2 is switched to the internal crystal oscillator 1 (XTAL1). The system clock and CPU Timer2 are used simultaneously to measure a period of time, and the difference between the counts is compared to determine whether there is a frequency deviation in the internal oscillators. If the elapsed number of ticks is not within the designated boundaries, the function returns STL_OSC_CT_FAIL. Otherwise, the function returns STL_OSC_CT_PASS. The source code is found at `f280013x\source\stl_osc_ct.c`.

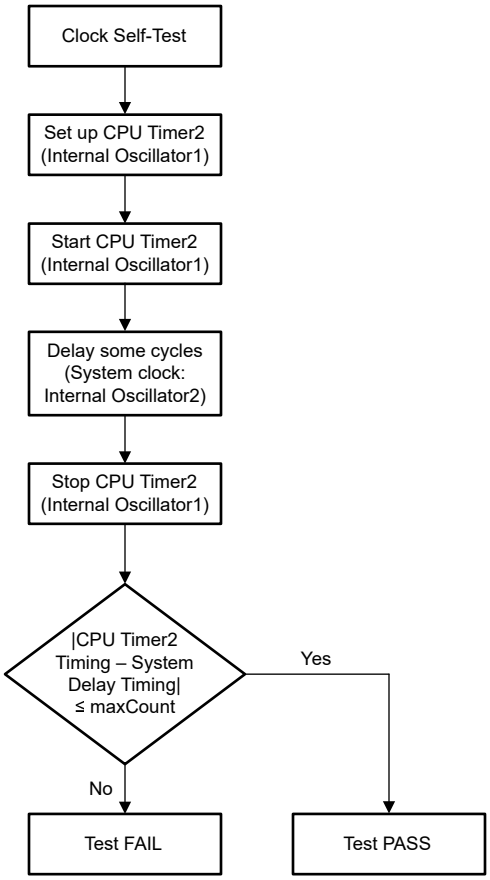

Figure 4-5. Clock Test Structure

4.6 RAM

RAM test often performs a March13N non-destructive memory test on the specified RAM memory. The text begins by copying the original contents of the memory to a preset address, performs the memory test, and then restores the original contents back to the memory under test. Figure 4-6 shows the RAM March test structure. The RAM march test is found in the test_application project called STA_MARCH_COPY, and the source code is in `f280013x\source\stl_march_s.asm`.

If this code is running from RAM, be careful not to perform a self-test on this memory test, meaning do not perform the March13N memory test on the March13N program code in RAM. This likely leads to an ITRAP. To test the program code for this March13N algorithm, create a copy of this function in RAM or flash and run the memory test code from the copy. This function disables global CPU interrupts (DINT) and then re-enables them after the test has completed.

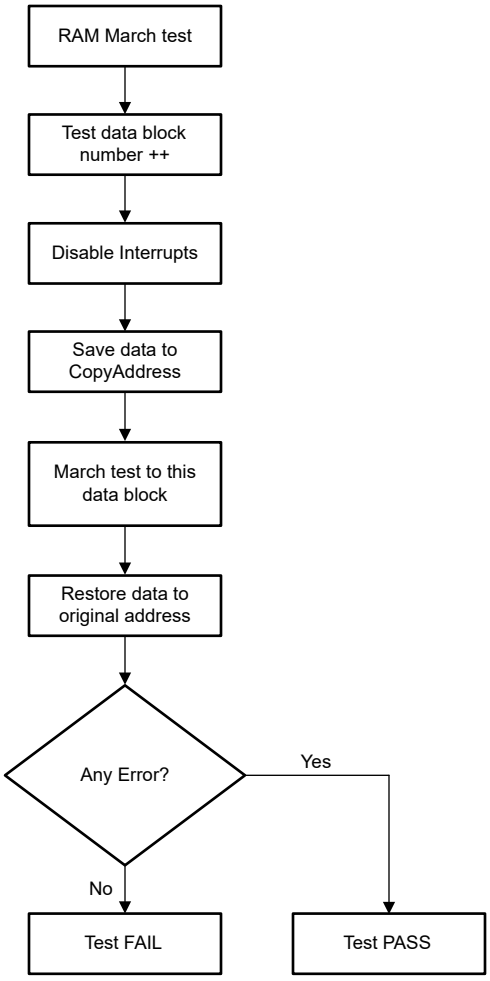

Figure 4-6. RAM March Test Structure

Since global interrupts need to be disabled, do not allow the duration of each detection to be too long so as to affect the execution of system functional interrupts. When adding this function to an application, confirm the size and address of the area to be checked first, and then determine the data length for each detection by balancing the test frequency and execution time.

4.7 Flash

The main purpose of flash verification is to check whether the code and data in the flash memory are consistent with the initial content. Traditionally, flash verification is achieved through CRC. The flash memory blocks are sequentially calculated for CRC, and the calculated CRC is stored as the Golden CRC array. During power-on and periodic cyclic verifications, the flash memory blocks are sequentially calculated for CRC again, and the results are compared with the corresponding Golden CRC. If the results match, the verification passes. The problem with this verification method is that global interrupts still need to be disabled during flash verification, so the size of each memory block for a single detection needs to not be too large to avoid affecting the execution of system functional interrupts. With the increasing functionality of appliances and the gradual increase in the size of flash memory of the MCU, the time required for one round of flash self-test can become too long to meet the security requirements of certification organizations. Therefore, when using CRC to verify flash memory, it is necessary to balance the size of each memory block for a single detection and the time required for one round of self-checking. Figure 4-7 shows the flash CRC test structure. Flash CRC test is found in the test_application project called STA_FLASH_CRC, and the source code is in `f280013x\source\stl_crc.c`.

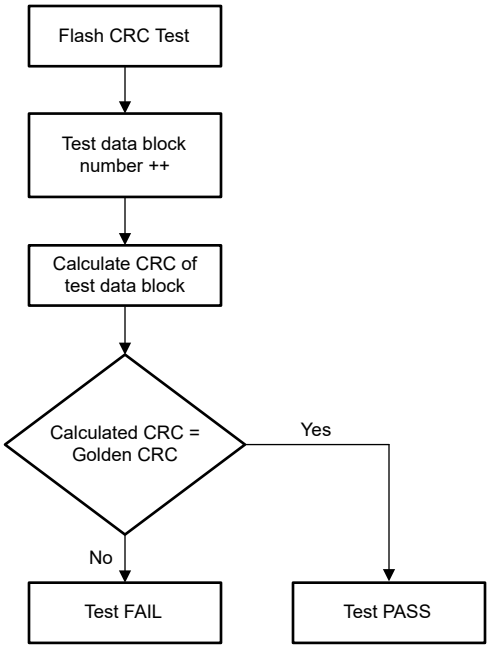

Figure 4-7. Flash CRC Test Structure

In the third-generation C2000 MCU recommended by TI for appliance applications, the flash memory comes with ECC verification functionality, which greatly reduces the time required for flash verification. When the flash memory is programmed, ECC codes are generated and stored in the corresponding memory locations. When the flash memory is read, the ECC module performs ECC calculation again and compares the results with the ECC generated during programming. This can detect single-bit errors and double-bit errors, with single-bit errors being correctable.

The on-chip flash memory is supported by single-error correction, double-error detection (SECDED) error correcting code (ECC) diagnostic. In this SECDED scheme, an 8-bit code word is used to store the ECC of 64-bit data and the corresponding address. The ECC decoding logic at the flash bank output checks the correctness of memory content. ECC evaluation is done on every data and program read. The data and program interconnects that connect the CPU and flash memory is not protected by ECC. Detected correctable errors can be corrected or not corrected, depending on whether correction functionality is enabled. Single-bit address ECC errors are flagged as uncorrectable errors. Errors that cannot be corrected generate an NMI and the ERRORSTS pin is asserted. Count of the corrected errors (single-bit data errors) is monitored by the flash wrapper and an interrupt is generated once the count exceeds the programmed threshold. The corrupted memory address of the last error location is also logged in the flash wrapper.

It is possible to test the functionality of ECC by injecting single-bit and double-bit errors in test mode and performing reads on locations with ECC errors, and checking for the error response. Flash ECC logic is checked with the help of ECC test registers (FECC_CTRL, FADDR_TEST, FECC_TEST, FDATAH_TEST, FDATAL_TEST). Correct functioning of error counter and threshold interrupt associated with single-bit errors can also be verified using this technique. Figure 4-8 shows the flash ECC test structure. Flash ECC test is found in the example\sdl_ex_flash_ecc_test project, and the source code is in `f280013x\examples\sdl_ex_flash_ecc_test\sdl_ex_flash_ecc_test.c`.

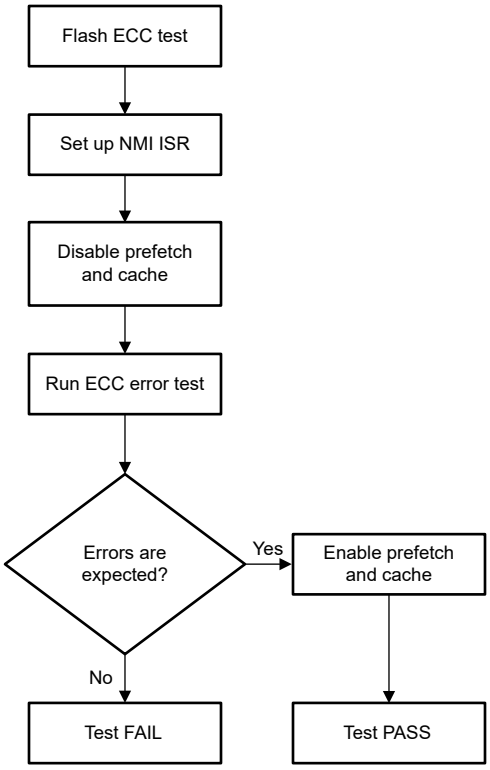

Figure 4-8. Flash ECC Test Structure

4.8 ADC

In addition, for UL/IEC 60730-1/60335-1 certification, the ADC module usually needs to be self-tested. An external voltage reference chip like the TLV431 can be added to the peripheral of the C2000 MCU connected to the AD port. A single C2000 integrated circuit usually contains multiple ADC modules, and each ADC module needs to be verified. Therefore, the AD port connected to the voltage reference chip is usually able to be sampled by any ADC module. This signal is sampled during power-on self-checking and periodic self-checking, and is considered passed if the signal falls within a certain deviation range.

4.9 Cycle Time and Memory Usage

Table 4-1 lists the main self-test items, and the memory usage, and execution cycles of the corresponding functions.

Table 4-1. Cycle Time and Memory Usage

Item	API	Memory (16-bit words)	PASS Cycles (RAM)
CPU	STL_CPU_REG_testCPURegisters	205	652
FPU	STL_CPU_REG_testFPURegisters	106	287
CLOCK	STL_OSC_CT_startTest & STL_OSC_CT_stopTest	96	145
RAM	STL_March_testRAMCopy (16 words)	47	501
FLASH(CRC)	STL_CRC_checkCRC (32 words)	36	234
FLASH(ECC)	RunECCErrorTest	67	276

Among them, the RAM and Flash (CRC) only list the pass cycles of 16 words and 32 words. Complete RAM and Flash (CRC) tests need to consider the total testing memory space and the self-test execution cycle. Flash (ECC) only needs to be executed once in a round of self-test, which is why Flash (ECC) can significantly reduce the CPU usage of the self-test function.

5 References

1. IEC 60730-1 Automatic Electrical Controls - Part1: General Requirements, International Electrotechnical Commission, Edition, Edition 5.2 2020-04
2. IEC 60335-1 Household and similar electrical appliances - Safety - Part1: General Requirements, International Electrotechnical Commission, Edition, Edition 5.2 2016-05
3. Texas Instruments, *Industrial Functional Safety for C2000™ Real-Time Microcontrollers Product Overview*
4. Texas Instruments, *Implementing IEC 60730 / UL 1998 Compliance for C2000 Real-Time Microcontrollers Product Overview*
5. Texas Instruments, *Safety Manual for C2000™ MCUs in IEC60730 Safety Applications User's Guide*
6. Texas Instruments, *Functional Safety Manual for TMS320F280015x Functional Safety Information*
7. Texas Instruments, *Safety Manual for TMS320F28002x Functional Safety Information*
8. Texas Instruments, *Functional Safety Manual for TMS320F28003x Real-Time Microcontrollers*
9. Texas Instruments, *Functional Safety Manual for TMS320F28004x User's Guide*
10. Texas Instruments, *Functional Safety Manual for TMS320F2837xD, TMS320F2837xS, and TMS320F2807x User's Guide*
11. Texas Instruments, *Online Stack Overflow Detection on the TMS320C28x DSP Application Note*
12. Texas Instruments, *C2000™ CPU Memory Built-In Self-Test Application Note*
13. Texas Instruments, *C2000™real-time MCU Safety Mechanisms Product Overview*

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