

Introduction

This document describes how to configure and use the Tightly Coupled Memory (TCM) features in the PIC32C and SAM MCUs using MPLAB® X IDE with the MPLAB XC32 compiler and MPLAB Harmony v3 software development framework.

TCM brings a lot of advantages to application developers. It provides the same performance as if the code is located in cache.

The following are some recommendations for optimizing the use of TCM in applications:

- Application stack can be located in the DTCM
- Application heap can be located in the DTCM
- Critical variables can be located in the DTCM
- Frequently updated variables can be located in the DTCM
- Critical functions/routines can be located in the ITCM
- A copy of the Interrupt vector table can be located in the ITCM
- Interrupt service routines can be located in the ITCM

Table of Contents

Introduction.....	1
1. Overview of Microchip Cortex-M7 based MCUs.....	3
1.1. PIC32CZ CA Family Microcontrollers.....	3
1.2. SAM E70/S70/V7x Family Microcontrollers.....	4
2. Cache Memory vs TCM Overview.....	6
3. Tightly Coupled Memory	7
3.1. Tightly Coupled Interface Unit (TCU).....	7
3.2. Access.....	7
3.3. Configuration.....	8
4. Software Implications of TCM.....	10
4.1. PIC32CZ CA Family.....	10
4.2. SAM E70/S70/V7x Family.....	12
5. Programming Sequence for TCM Configuration.....	16
5.1. PIC32CZ CA Family	16
5.2. SAM E70/S70/V7x Family	16
6. Hardware and Software Requirements.....	18
6.1. Prerequisites.....	18
6.2. Application Overview.....	18
6.3. Application Code.....	19
7. Building and Programming the Application.....	20
8. Observing the Output on the MPLAB Data Visualizer.....	23
9. Resources.....	32
10. Abbreviations.....	33
11. Revision History.....	34
Microchip Information.....	35
Trademarks.....	35
Legal Notice.....	35
Microchip Devices Code Protection Feature.....	35

1. Overview of Microchip Cortex-M7 based MCUs

The PIC32CZ CA family and SAM E70/S70/V7x family of Microcontrollers (MCUs) from Microchip are high-performing MCUs with an Arm Cortex-M7 core. This series of MCUs offers a rich set of peripherals with a large built-in code and data memory. To ensure high-speed, low-latency, and deterministic access for time-critical code and data, the Arm Cortex-M7 core is connected to the TCM.

1.1 PIC32CZ CA Family Microcontrollers

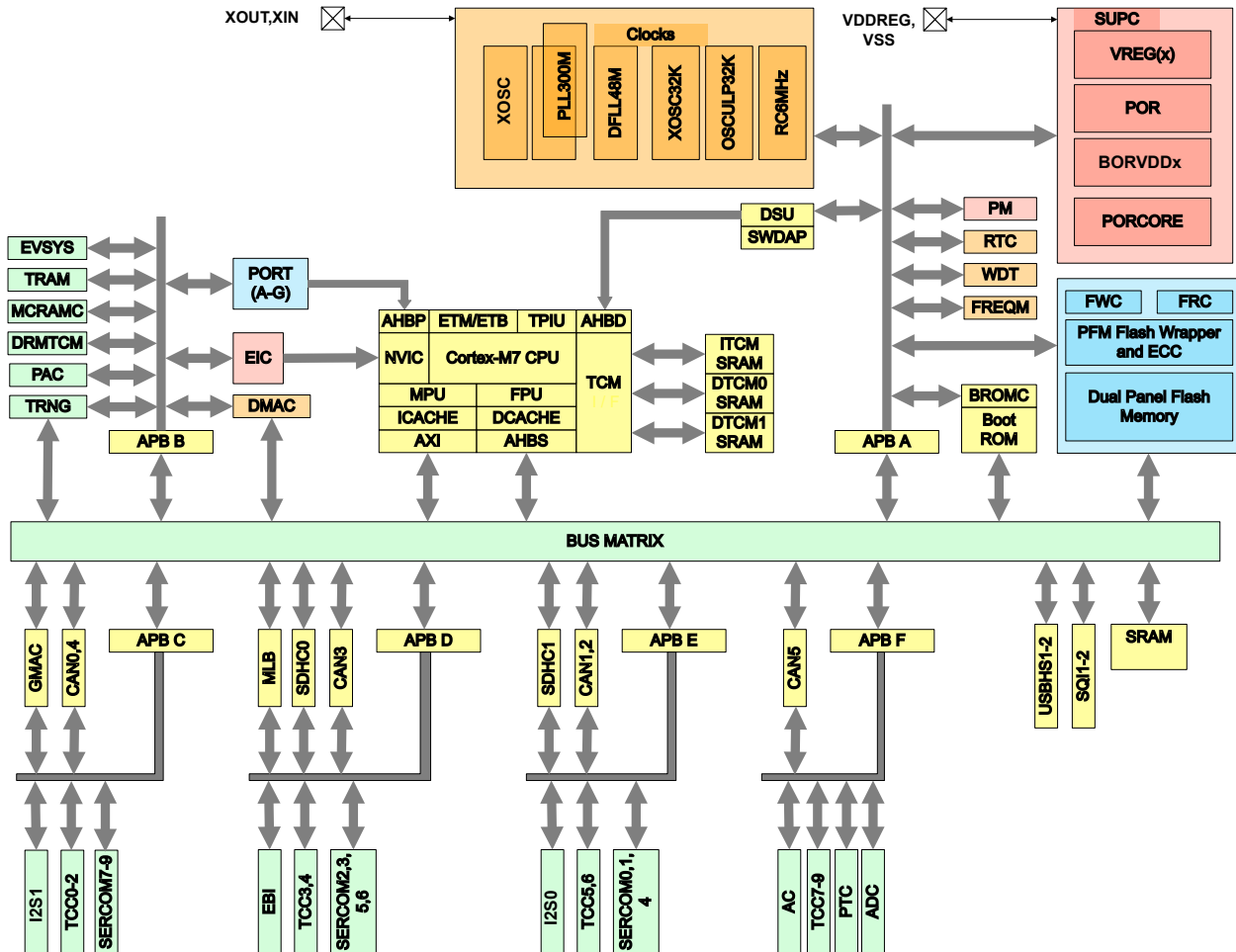
The PIC32CZ CA series of microcontrollers features a high-performance Arm Cortex-M7 core and is designed with an advanced memory architecture. These microcontrollers offer up to 8 MB of Flash memory and 1 MB of SRAM to meet the rigorous memory demands of sophisticated applications. The Arm Cortex-M7 core includes an interface with a dedicated TCM that incorporates Error Correction Code (ECC) for enhanced high-speed, low-latency, and deterministic processing.

The PIC32CZ CA9x family offers an integrated Hardware Security Module (HSM) and a diverse range of connectivity features, making it an outstanding choice for applications in the automotive and industrial sectors, as well as other areas requiring robust security measures, such as communications encryption, secure boot processes, authentication protocols, and secure key storage.

It offers an extensive array of connectivity features, such as high-speed USB, CAN FD, Serial Quad I/O (SQI), Secure Digital High Capacity (SDHC), Inter-IC Sound (I²S™), Media Local Bus (Media LB™), External Bus Interface (EBI), 10/100/1000 Ethernet, and Serial Communication Interface (SERCOM).

These MCUs are equipped with a Memory Protection Unit (MPU) to enhance security, incorporate a clock failure detection mechanism complemented by a fail-safe internal RC oscillator, and provide write protection for selected peripherals to ensure system integrity.

Figure 1-1. PIC32CZ CA Architecture Block Diagram



1.2 SAM E70/S70/V7x Family Microcontrollers

The SAM E70/S70/V7x series of microcontrollers are equipped with a high-performance Arm Cortex-M7 core and features an advanced memory architecture, including up to 384 KB of multi-port internal SRAM.

In conjunction with a robust set of peripherals, these microcontrollers are exceptionally suitable for sophisticated applications, such as IoT and Smart Energy gateways, industrial automation, and building control systems.

The comprehensive peripheral set encompasses an Ethernet MAC with dedicated support for Ethernet AVB, high-speed USB with an integrated PHY, I²S, QSPI, and dual CAN-FD. For additional information on these device's peripherals and package options, refer to the relevant data sheets.

To facilitate high-speed, low-latency, and deterministic processing for critical code and data, the Arm Cortex-M7 core is interfaced with TCM. A multi-layer bus matrix also provides efficient interconnection between the peripherals and the multi-port RAM.

Overall, the SAM E70/S70/V7x family of devices delivers an unparalleled combination of connectivity options, top-tier Arm Cortex-M7 processing power, and a distinctive multi-port memory architecture, positioning it as a leading choice for high-end applications.

2. Cache Memory vs TCM Overview

This chapter highlights the key differences between Cache memory and TCM.

Table 2-1. TCM vs Cache

Tightly Coupled Memory	Cache Memory
Tightly Coupled Memory is accessed by a dedicated connection from the core. In the PIC32CZ CA and SAM E70/S70/V7x family of microcontrollers, there are two dedicated connections from the Cortex-M7 core to the internal SRAM for Instruction TCM and another for Data TCM.	Cache memory is RAM integrated inside the Cortex-M7 core itself. In the PIC32CZ CA and SAM E70/S70/V7x series of microcontrollers, 16 KB of instruction cache (I-cache) and 16 KB of data cache (D-cache) are available.
Tightly coupled memory is part of the system memory map with a definite start address. The size of the TCM determines the end address	Cache memory is not part of the system memory map. It does not have a physical memory address.
A programmer can decide where the content will be stored in TCM at compilation time.	A control logic, not the programmer, determines what is stored in cache memory.
TCM is directly accessible to software.	During program execution, the cache is stored with instructions or data fetched from memory to the CPU.
TCM can be accessed by CPU and DMA.	Cache cannot be accessed by DMA.
TCM has deterministic access time. It always takes a single cycle to access contents from TCM.	Cache memory serves as an intermediate buffer between the processor and memory to reduce memory access time. The number of cycles needed to access a memory location differs for a cache hit and a cache miss.

Note: TCM contents are not cached. TCM is directly connected to the Cortex-M7 core by a bus. It can be accessed at similar speeds as accessing the cache without penalty for a cache miss and coherence issues.

3. Tightly Coupled Memory

The PIC32CZ CA80/90 and SAM E70/S70/V7x family of devices support two TCM instances: Instruction TCM (ITCM) and Data TCM (DTCM). The base address of each TCM is fixed by the Arm System Address map:

- ITCM at 0x00000000
- DTCM at 0x20000000

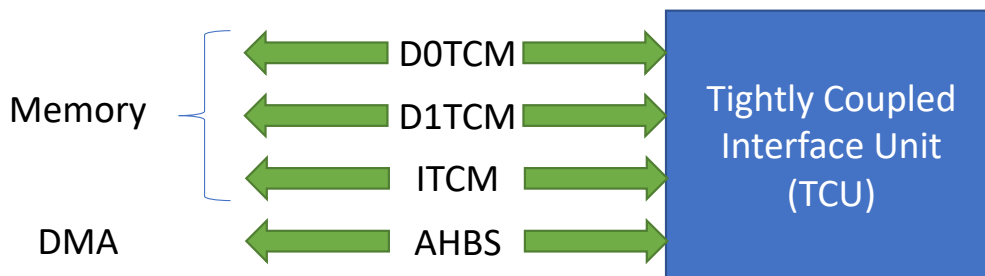
In the Arm microcontroller memory map, the ITCM and DTCM occupy lower address ranges than the Flash and internal SRAM.

3.1 Tightly Coupled Interface Unit (TCU)

The TCU interfaces to the following:

- **ITCM:** A single 64-bit Interface
- **DTCM:** This is connected by 2 x 32-bit interfaces:
 - The dual 32-bit interfaces enable two simultaneous access to DTCM. Two 32-bit data can be fetched concurrently in a single cycle.
- **AHBS:** TCM can be accessed by DMA through the AHB Client bus interface:
 - This enables peripherals, such as GMAC, to access the TCM memory without any CPU intervention.

Figure 3-1. Tightly Coupled Interface Unit



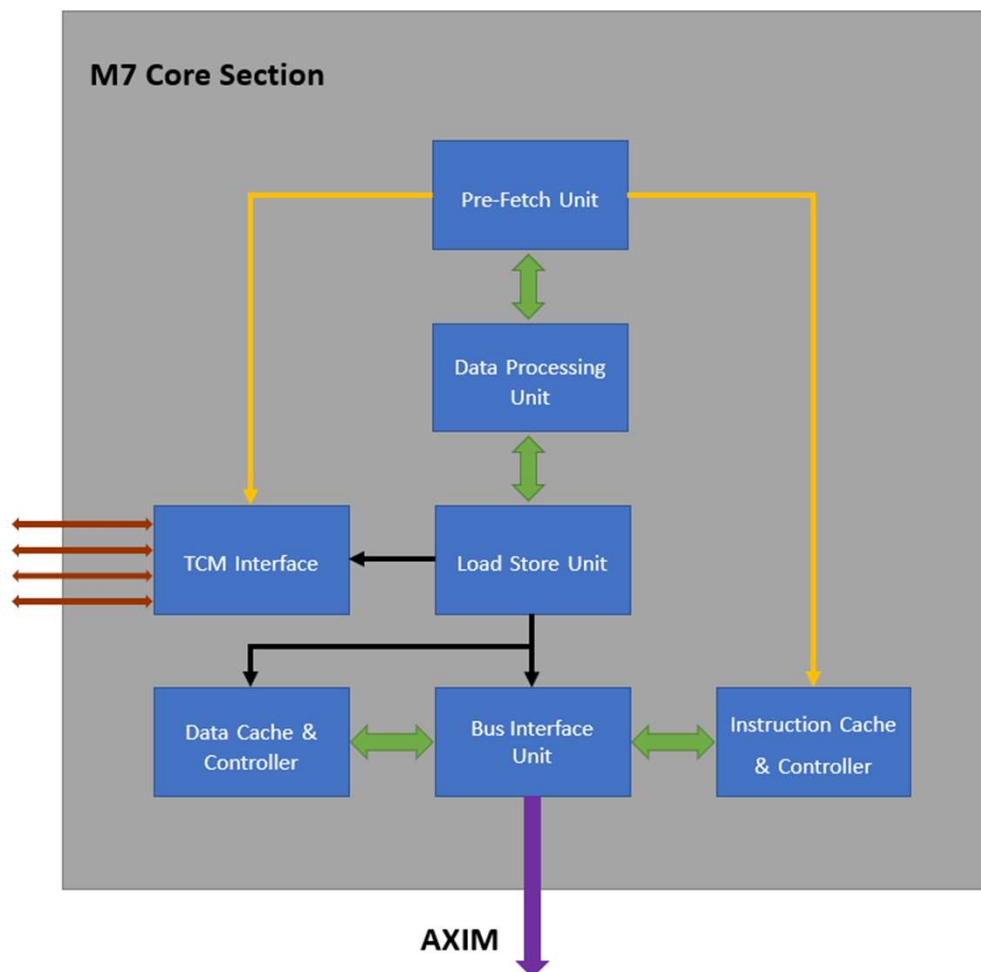
3.2 Access

Read/write request for each TCM interface are triggered from the following interfaces of the core:

- Load Store Unit (LSU)
- Pre-fetch Unit (PFU)
- System AHBS interface (AHBS)
- Debug Unit (DGU)
- Bus Interface Unit (BIU)

The TCU contains arbitration logic to arbitrate between TCM access requests by various interfaces. The PFU can only read from the TCM, whereas other interfaces can read and write to the TCM.

Figure 3-2. Cortex-M7 Architecture for TCM Interface



3.3 Configuration

3.3.1 PIC32CZ CA Family Microcontrollers

The PIC32CZ CA family of microcontrollers features integrated TCM with a total capacity of 256 KB, which is allocated equally between 128 KB of Instruction TCM (ITCM) and 128 KB of Data TCM (DTCM). The size of ITCM and DTCM is fixed and configured by default through MPLAB Harmony v3 by using MPLAB Code Configurator (MCC).

Notes:

- Access to the TCM is at full processor speed (HCLK). In the PIC32CZ CA family, the HCLK can be up to a maximum of 300 MHz without any wait states. All accesses from the core are configured for single-cycle access. Therefore, the deterministic behavior of TCM access.
- TCM can be explicitly enabled or disabled by a register configuration in the system control block of the Cortex-M7.

3.3.2 SAM E70/S70/V7x Family Microcontrollers

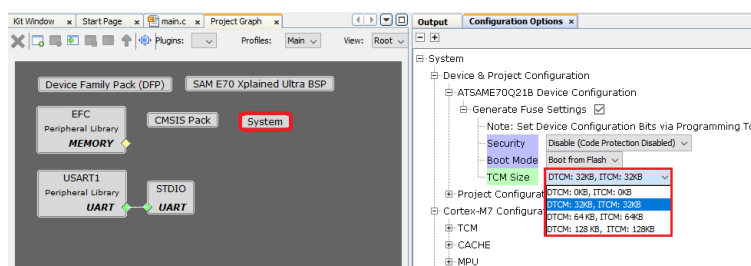
The TCM can be set to one of the four different configurations by software. Configuration can be changed by programming the GPNVM bits (8:7) with values as shown in the following table. Multiple TCM configurations provide flexibility for application developers.

Table 3-1. TCM Configuration in KB

ITCM (KB)	DTCM (KB)	Available SRAM (KB)		GPNVM Bits (8:7)
		384 KB Variant	256 KB Variant	
0	0	384	256	0
32	32	320	192	1
64	64	256	128	2
128	128	128	0	3

To adjust the sizes of the ITCM and DTCM, users need to navigate to the MCC Project Graph window and select the System module. Configure the ITCM and DTCM settings according to the parameters specified in the Configuration Options.

Note: The PIC32CZ CA70/MC70 family of microcontrollers shares a TCM configuration similar to that of the SAM E/S/V family.

Figure 3-3. ITCM and DTCM Configuration**Notes:**

- There is no explicit or extra memory TCM. It is the internal SRAM of the microcontroller that gets partitioned as TCM and SRAM. The GPNVM bits control the amount of memory available as TCM.
- Access to the TCM is at full processor speed (HCLK). In the case of the SAM E70/S70/V7x family, the HCLK can be up to a maximum of 300 MHz without any wait states. All accesses from the core are configured for single-cycle access. Therefore, the deterministic behavior of TCM access.
- The remaining part of the total internal SRAM, not used as TCM, is accessed at a Main Clock (MCK) frequency of up to 150 MHz.
- TCM can be explicitly enabled or disabled by a register configuration in the system control block of the Cortex-M7.

4. Software Implications of TCM

4.1 PIC32CZ CA Family

4.1.1 Address Mapping After Reset

The PIC32C family of devices have a Boot ROM that is responsible for managing the loading of configurations and providing a root of trust for secure boot. After the Boot ROM finishes, it prepares the CPU to start executing from the base of BFM which holds the vector table.

Based on the SEQNUM (15:0) of FSEQ in the User Configurations and SEQNUM (15:0) of BROM_BSEQ in the Boot User Configurations, the respective Flash panel is mapped as the lowest memory region is as follows.

Boot Flash Memory (BFM):

- If the sequence number of BFM panel 1 > panel 2, panel 1 is mapped at the start of BFM and followed by panel 2.
- If the sequence number of BFM panel 2 > panel 1, panel 2 is mapped at the start of BFM and followed by panel 1.

Program Flash Memory (PFM):

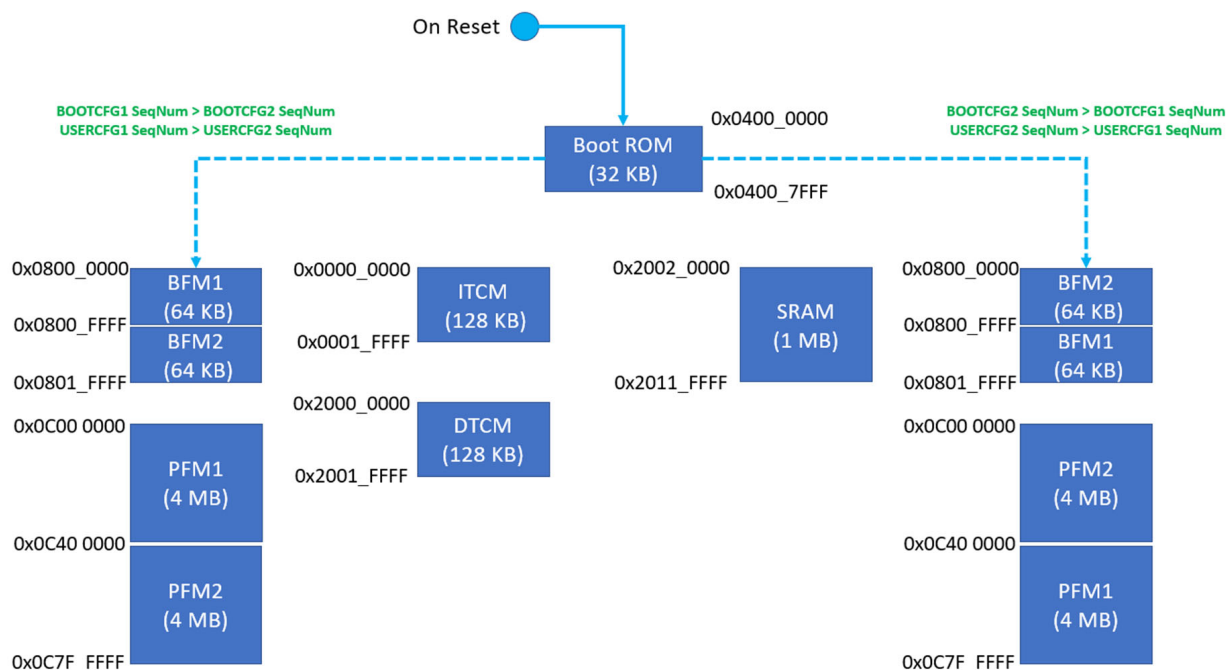
- If the sequence number of PFM panel 1 > panel 2, panel 1 is mapped at the start of PFM and followed by panel 2.
- If the sequence number of PFM panel 2 > panel 1, panel 2 is mapped at the start of PFM and followed by panel 1.

In a fully erased microcontroller, the values of Flash memory configurations are as follows.

- SEQNUM (15:0) of FSEQ in the User 1 Configuration is set to 0x0001
- SEQNUM (15:0) of FSEQ in the User 2 Configuration is set to 0x0000
- SEQNUM (15:0) of BROM_BSEQ in the Boot User 1 Configuration is set to 0x0001
- SEQNUM (15:0) of BROM_BSEQ in the Boot User 2 Configuration is set to 0x0000

Example: The [PIC32CZ8110CA90208](#) microcontroller has 8 MB Flash and 1 MB internal SRAM.

Consider a case wherein the USERCFG1_FSEQ -> SEQNUM[15:0] is greater than USERCFG2_FSEQ -> SEQNUM (15:0) and BOOTCFG1_BROM_BSEQ -> SEQNUM (15:0) is greater than BOOTCFG2_BROM_BSEQ -> SEQNUM (15:0). There is no swap between the available Boot Flash Memory (BFM) and Program Flash Memory (PFM).

Figure 4-1. Address '0' Mapping on Reset

4.1.2 ITCM is at Address 0x0

In PIC32CZ CA, the ITCM RAM region is mapped to 0x0000 0000 by ARMv7-M architecture.

The address 0x0 mapped to ITCM can be accessed only if the bit '0' of the ITCM Control Register (ITCMCR) at 0xE000EF90 is set.

The bit '0' of the ITCMCR register can be used to enable or disable the ITCM.

- When this bit is set, the Cortex-M7 core shall fetch all mapped instructions in the ITCM range (between 0x0 – last address of ITCM) through the ITCM bus and not the AXIM interface.
- The Flash memory is still available in its physical memory address starting at 0x0C00 0000.
- On any reset, the bit '0' of the ITCMCR register gets cleared.
- Vector tables and start-up code can be located in the Boot Flash Memory and Program Flash Memory.

The Cortex-M7 core performs all instruction accesses after a reset through the AXI bus and does not access the TCM interface, see Figure 4-2. *Cortex-M7 Architecture for TCM Interface*. However, as soon as the ITCMCR is configured to enable the ITCM, any further access to address 0 will be to ITCM through the TCM interface instead of the AXI bus.

4.1.3 Mapping ITCM and Vector Table Relocation (VTOR) Register

The default value for VTOR is the base address of the BFM. As specified in [Address Mapping After Reset](#), vector tables are typically located at the beginning of the same, that is, at address 0x080_0000. Any interrupt triggered during the execution has its interrupt service routine correctly fetched from the VTOR offset.

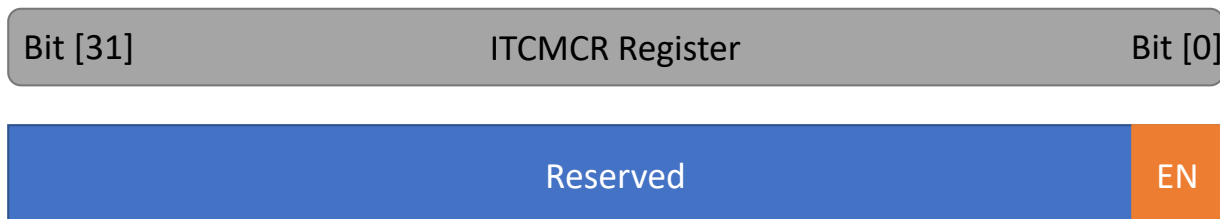
The following are two possible configurations for re-mapping the VTOR register:

- Write the VTOR register with the address of the interrupt vector table in Flash.
- Copy the Vector Table into ITCM after start-up and update the VTOR register with this address.

Note: The user can also copy or locate any critical interrupt routines to ITCM to increase the access speed of the routines and reduce latency in servicing the interrupt. For the consistently fastest interrupt response to time-critical events, both the vector table and the time-critical event handlers must be located in the ITCM.

After changing the VTOR register at address 0xE000ED08 appropriately, enable the bit '0' in the ITCMCR register at address 0xE000EF90. The memory map is described in the [Address '0' Mapping on Reset](#) section.

Figure 4-2. ITCMCR Register



4.1.4 DTCM

The DTCM size is the same as the ITCM. By default, the DTCM is enabled.

If needed, the bit '0' in the DTCMCR register can be used to disable or enable the DTCM. This memory remains unused in the system. For additional information, refer to the *TCM Configuration*.

4.1.5 Software Build

The TCM should not be dynamically changed. The memory map used in the linker script must match the GPNVM configuration of the TCMs. The reason for this is the code or data inside TCM are not mirrored but actually copied to the memory in the start-up code, and copy routines must aware of the boundaries of the TCM.

4.2 SAM E70/S70/V7x Family

4.2.1 Address 0x0

The bit '1' of GPNVM selects the memory that is mirrored at address 0x0000 0000.

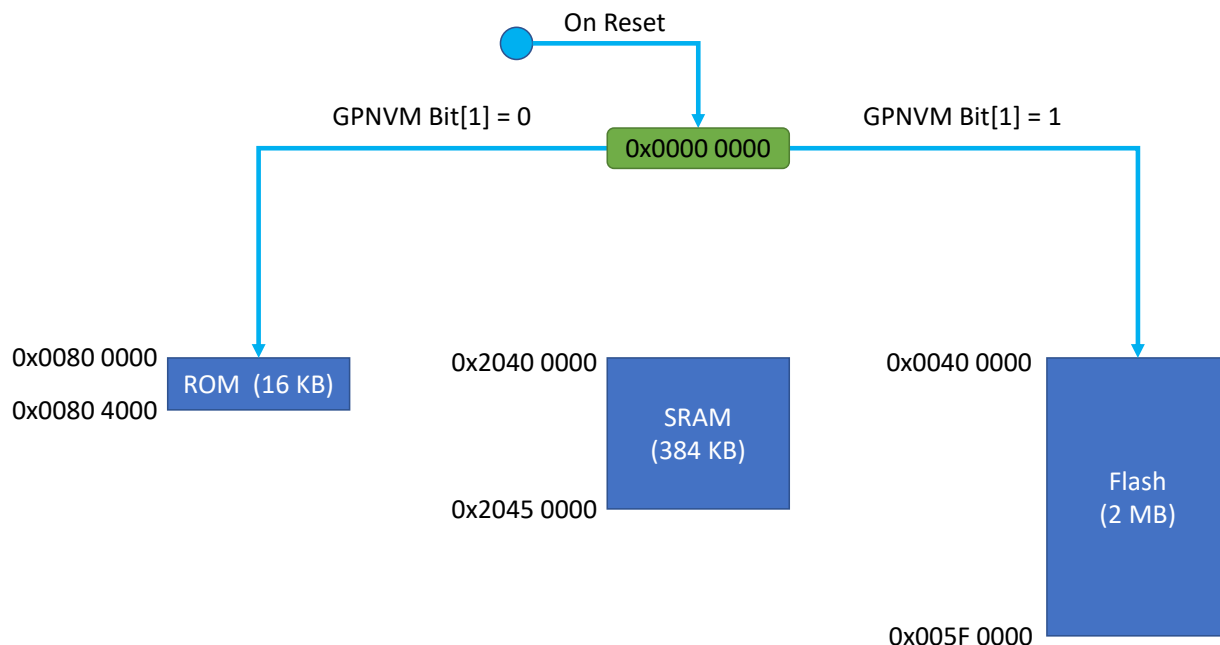
- **0:** Internal ROM, ROM contents at 0x0080 0000 get mirrored at address 0
- **1:** Internal Flash, Flash contents at 0x0040 0000 get mirrored at address 0

In a fully erased microcontroller, the value of this GPNVM bit is set to '0'. This bit value enables the boot-loader code in ROM to execute and program the Flash. The GPNVM bit '1' should be set to '1' after programming a valid application in Flash.

Note: The PIC32CZ CA70/MC70 family of microcontrollers shares a TCM configuration similar to that of the SAM E/S/V family.

Example: The [ATSAME70Q21](#) microcontroller, 2 MB Flash, 384 KB internal SRAM.

Consider a case wherein the GPNVM (8:7) = b'00. Here ITCM = 0KB; DTCM = 0KB. There is no reduction in available SRAM.

Figure 4-3. Address "0" Mapping on Reset and GPNVM (8:7) = b'00

4.2.2 ITCM at Address 0x0

In SAM E70, the ITCM RAM region is also mapped to 0x0000 0000 by ARMv7-M architecture.

The address 0x0 mapped to ITCM can be accessed only if both the conditions below are satisfied.

- The GPNVM (8:7) bits setting indicates a non-zero TCM.
- The bit '0' of the ITCM Control Register (ITCMCR) @ 0X E000EF90 is set.

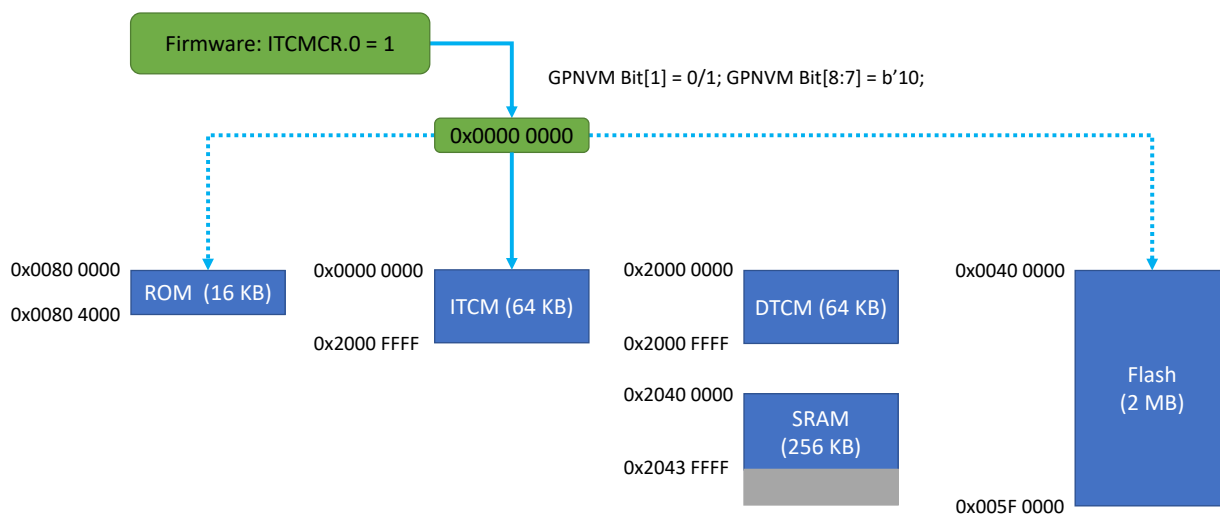
The bit '0' of the ITCMCR register can be used to enable or disable the ITCM.

- When this bit is SET, the Cortex-M7 core shall fetch all mapped instructions in the ITCM range (between 0x0 – last address of ITCM) through the ITCM bus and not the AXIM interface.
- The Flash memory is still available in its physical memory address starting at 0x0040 0000.
- On any reset, the bit '0' of ITCMCR gets cleared: Vector tables and start-up code can be located in ROM or Flash irrespective of the GPNVM bits configuration.

The Cortex-M7 core performs all instruction accesses after a reset through the AXI bus and does not access the TCM interface, refer to Figure 4-2. [Cortex-M7 Architecture for TCM Interface](#). Therefore, an address 0x0 on a reset map to the Flash or ROM, even if the GPNVM bits (8:7) are non-zero. However, as soon as the ITCMCR is configured to enable the ITCM, any further access to address 0 will be to ITCM through the TCM interface instead of the AXI bus.

Example: The SAMV71Q21 microcontroller , 2 MB Flash, 384 KB internal SRAM.

Consider a case wherein the GPNVM (8:7) = b'10. Where, ITCM = 64 KB; DTCM = 64 KB. The internal SRAM is reduced by 128 KB. The available SRAM is 256 KB instead of 384 KB.

Figure 4-4. Address '0' Mapping on the ITCM Configuration

4.2.3 Mapping ITCM and Vector Table Relocation (VTOR) Register

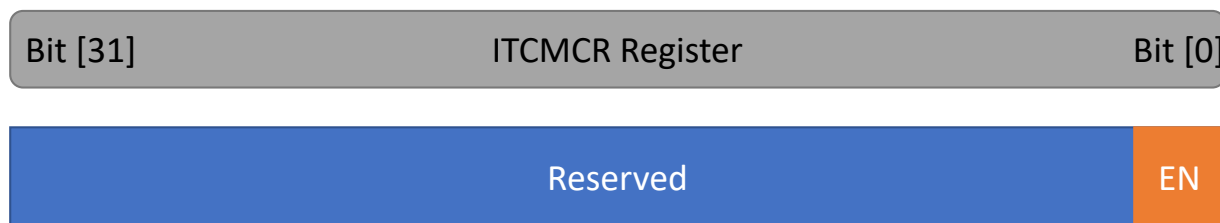
The default value for VTOR is 0x0000_0000. As specified in [Address 0x0](#), in cases where ITCM is not used, this address 0x0000_0000 mirrors the start of the Flash memory. Vector Tables are typically located at the beginning of the same, that is, at address 0x0040_0000. Any interrupt triggered during the execution has its interrupt service routine correctly fetched from the VTOR offset.

There are two possible configurations for re-mapping the VTOR register:

- Write the VTOR register with the address of the interrupt vector table in Flash.
- Copy the Vector Table into ITCM after start up and update VTOR with this address.

Note: The user can also copy/locate any critical interrupt routines to ITCM to increase the access speed of the routines and reduce latency in servicing the interrupt. For the consistently fastest interrupt response to time-critical events, both the vector table and the time-critical event handlers must be located in the ITCM.

After changing the VTOR register at address 0xE000ED08 appropriately, enable the bit '0' in the ITCMCR register at address 0xE000EF90. The memory map immediately gets altered. The new memory map will include an interface into ITCM as described in [Address "0" Mapping on ITCM Configuration](#).

Figure 4-5. ITCMCR Register

4.2.4 DTCM

The DTCM size is the same as the ITCM.

A non-zero TCM size by GPNVM (8:7) configuration is sufficient to enable DTCM. Unlike ITCM, which must be explicitly enabled by software after a reset, the DTCM is enabled at reset if configured for a non-zero size.

If needed, the bit '0' in DTCMCR can be used to disable or enable the DTCM. This memory remains unused in the system. It is necessary to select an appropriate TCM configuration based on application needs. Refer to *TCM Configuration* for additional information.

4.2.5 Software Build

The TCM must not be dynamically changed. The memory map used in the linker script must match the GPNVM configuration of the TCMs. The reasons for this are as follows:

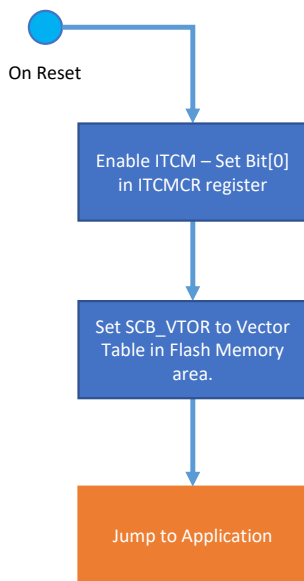
- The code/data inside TCM are not mirrored but actually copied to the memory in the start-up code:
 - Copy routines must know the boundaries of the TCM.
- Configuration of TCM results in a corresponding reduction of available internal SRAM:
 - TCM partition is located within the SRAM. The amount of available SRAM depends on the chosen TCM configuration.
 - The Flash download routines must be aware of the available SRAM when the TCM is configured.
- A reset must be triggered whenever the GPNVM configuration bits are changed to execute the change.

5. Programming Sequence for TCM Configuration

5.1 PIC32CZ CA Family

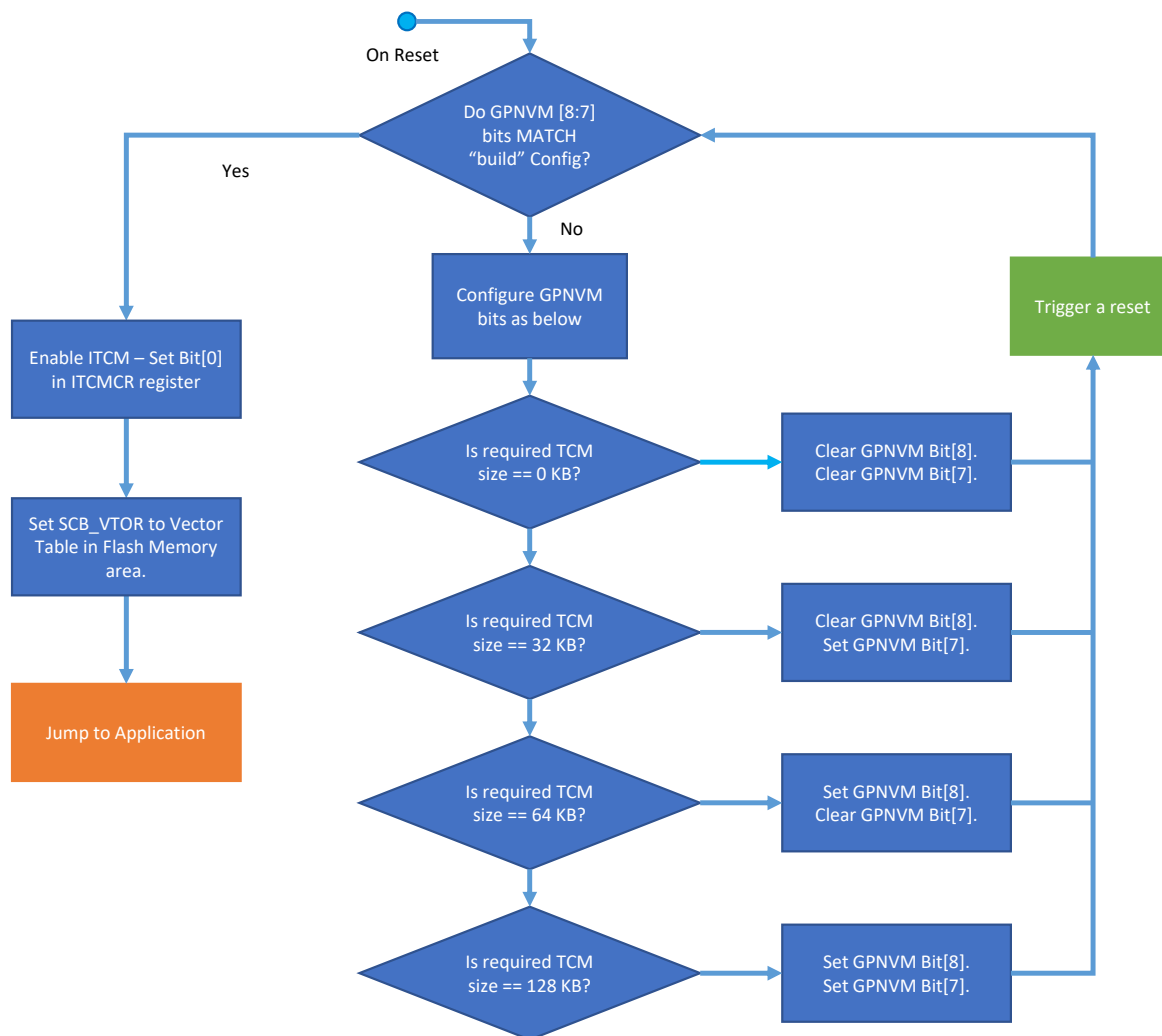
The TCM memory configuration in the PIC32CZ CA family is illustrated below.

Figure 5-1. TCM Configuration for the PIC32CZ CA family in Software



5.2 SAM E70/S70/V7x Family

The GPNVM bits configuration sequence is illustrated for the TCM configuration.

Figure 5-2. TCM Configuration for the SAM E/S/V family in Software

Note: The PIC32CZ CA70/MC70 family of microcontrollers shares a TCM configuration similar to that of the SAM E/S/V family.

6. Hardware and Software Requirements

The following sections cover hardware and software prerequisites used for this application, and they also cover the overview of the application.

6.1 Prerequisites

Hardware:

- [PIC32CZ CA90 Curiosity Ultra Evaluation Board](#)
- [SAME70 Xplained Ultra Pro Evaluation Kit](#)

Software:

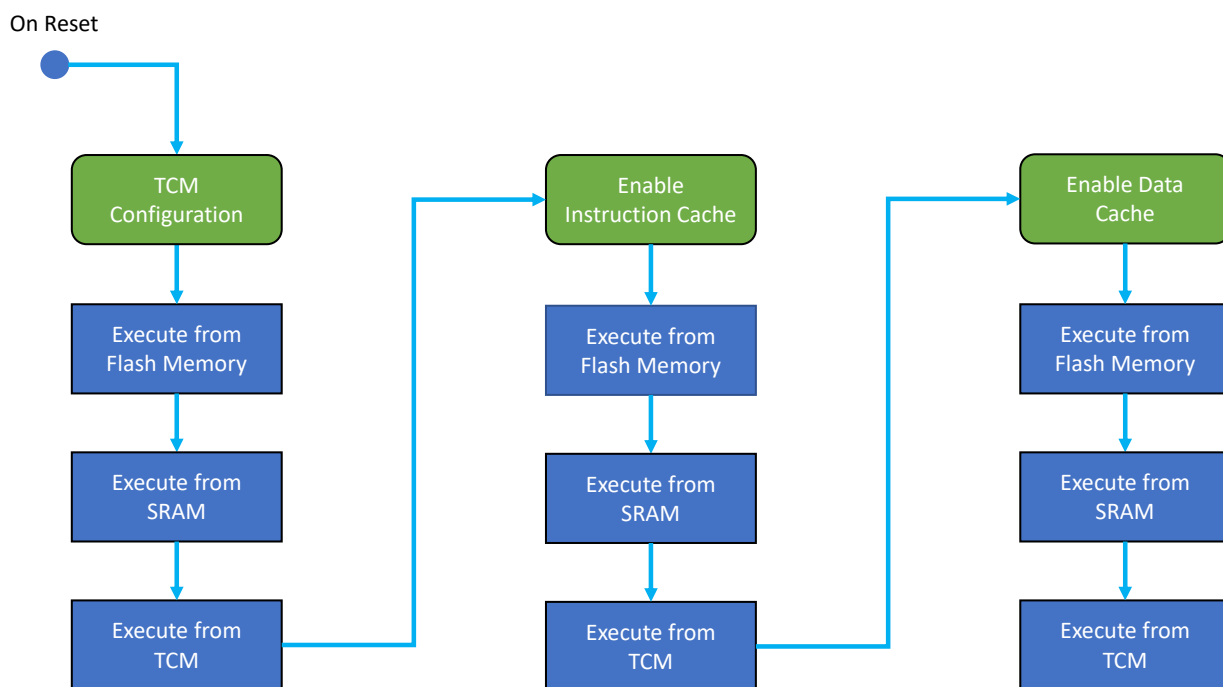
- [MPLAB® X IDE](#) v6.20 or above
- [MPLAB® XC32 compiler](#) v4.45 or above
- [MPLAB® Code Configurator](#) v5.5.1 or above
 - [csp](#) v3.19.7 (MPLAB Harmony v3 Chip Support Package) or above

Note: The application discussed in this document uses the above-mentioned MPLAB Harmony v3 repositories. These repositories can be downloaded from GitHub or use [MCC Content Manager](#) to download the above mentioned repositories. These steps can also be used for the PIC32CZ CA70/MC70 family of microcontrollers.

6.2 Application Overview

This example illustrates the use of TCM in the PIC32CZ CA and SAM E70/S70/V70/V71 family of devices. It compares the time taken to execute the code from Flash memory, SRAM, and TCM with or without the I-cache and D-cache.

Figure 6-1. Flowchart



6.3 Application Code

To illustrate code execution from memory locations, a simple data transfer function is created on Flash memory, SRAM, and TCM. Then, the time taken is compared for each data transfer with the help of the core cycle counter. Using the MPLAB XC32 compiler attributes, the functions and data buffers are placed in different memory locations.

- `__attribute__((tcm))`: Places the function or buffer in TCM
- `__attribute__((ramfunc, long_call))`: Places the function or buffer in SRAM

In the `main.c` file,

- Two buffers, `TCM_SrcBuff` and `TCM_DestBuff`, are defined and placed in DTCM
- Two buffers, `SrcBuff` and `DestBuff`, are defined and placed in SRAM
- A function, `TCM_memcpy()`, is in ITCM
- A function, `SRAM_memcpy()`, is in SRAM
- A function, `flash_memcpy()`, is in Flash

7. Building and Programming the Application

Follow these steps to program the TCM performance application for the PIC32CZ CA and SAM E70/S70/V70/V71 family of devices.

1. Download the TCM performance application from the following link:
 - PIC32CZ CA90 Curiosity Ultra Evaluation Board ([Click Here](#)) or access from [GitHub](#)
 - SAM E70 Xplained Ultra Evaluation Kit ([Click Here](#)) or access from [GitHub](#)
2. Open the PIC32CA CA90 TCM performance project (path: *<downloaded path>/pic32cz_ca90_tcm_performance/firmware/tcm_performance_pic32cz_ca90_cult.X*) from MPLAB X IDE.
Note: Choose the SAM E70 TCM performance project (path: *<downloaded path>/same70_tcm_performance/firmware/tcm_performance_same70_xult.X*) for the SAM E70 Xplained Ultra Evaluation Kit.
3. The PIC32CZ CA90 Curiosity Ultra Development Board supports an on-board debugger (PKoB4) for debugging. Connect the Type-A male to micro-B USB cable to the micro-B DEBUG USB port to power and debug the PIC32CZ CA90 Curiosity Ultra Evaluation Board. Connect the External Power Supply (6.5v to 14v) to the J100 connector.

Figure 7-1. PIC32CZ CA90 Curiosity Ultra Evaluation Board Hardware Setup

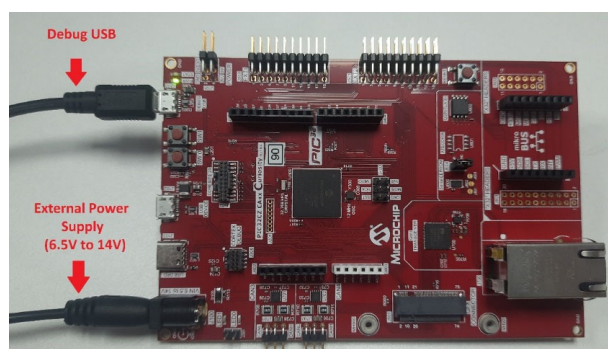
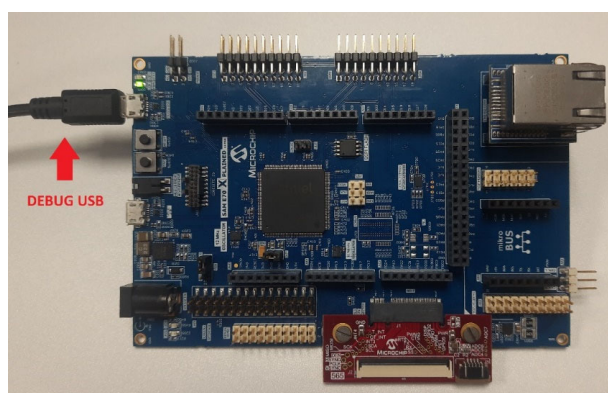


Figure 7-2. SAM E70 Xplained Ultra Evaluation Kit Hardware Setup

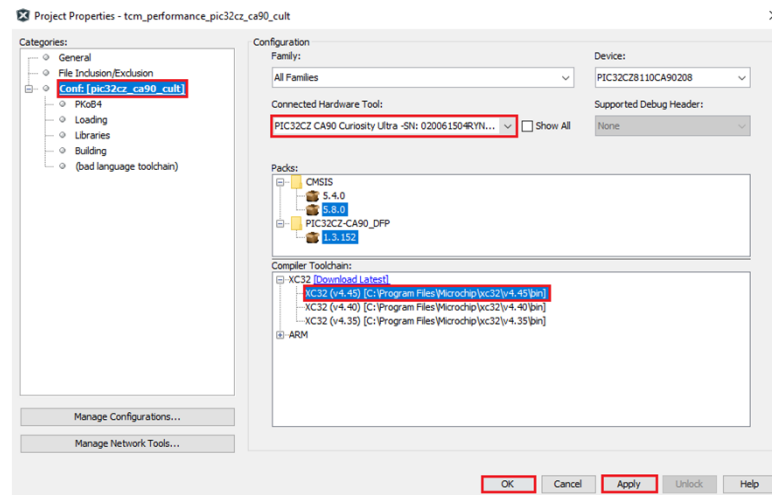


Note: Connect the Type-A male to the micro-B USB cable to the micro-B DEBUG USB port to power and debug the SAM E70 Xplained Ultra Evaluation Kit only.

4. In the MPLAB X IDE **Project Properties** Window:
 - a. Under the left **Categories** pane, select **Conf: [pic32cz_ca90_cult]**.

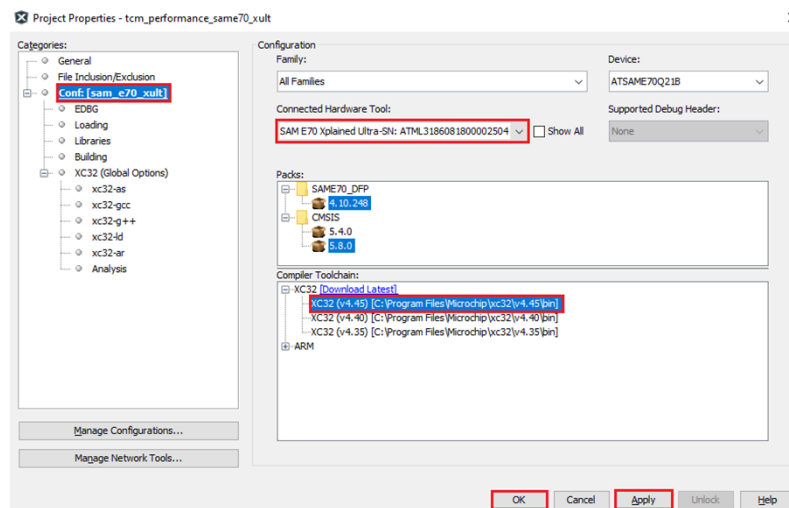
- b. In the right **Configuration** properties page:
 - i. For the Connected Hardware Tool, select **PIC32CZ CA90 Curiosity Ultra**.
 - ii. For the Compilar Toolchain, click and expand XC32 and then select **XC32 (V4.45)**.

Figure 7-3. Project Properties - PIC32CZ CA90 Curiosity Ultra Evaluation Board

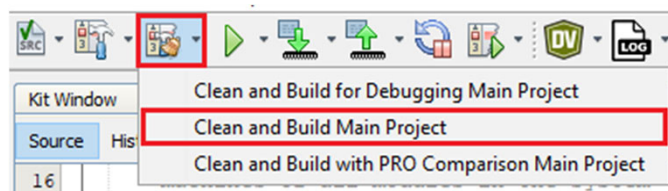


Note: Users need to select Conf: [sam_e70_xult] for the SAM E70 Xplained Ultra Evaluation Kit as shown below.

Figure 7-4. Project Properties - SAM E70 Xplained Ultra Evaluation Kit



5. Click **Apply**, and then click **OK**.
6. To build the project, click on the **Clean and Build** icon or select **Clean and Build Main Project** from the drop-down item list and verify that the project builds successfully. At this point, the application code can be implemented.

Figure 7-5. Clean and Build Project

7. Program the application by clicking the **Program the Device** icon.

Figure 7-6. Program the Device

8. Observing the Output on the MPLAB Data Visualizer

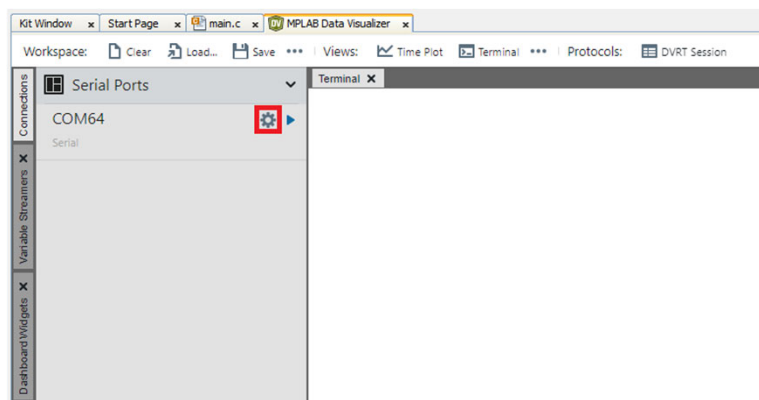
1. After building and programming the application, open MPLAB Data Visualizer by clicking the highlighted icon.

Figure 8-1. Launching MPLAB Data Visualizer



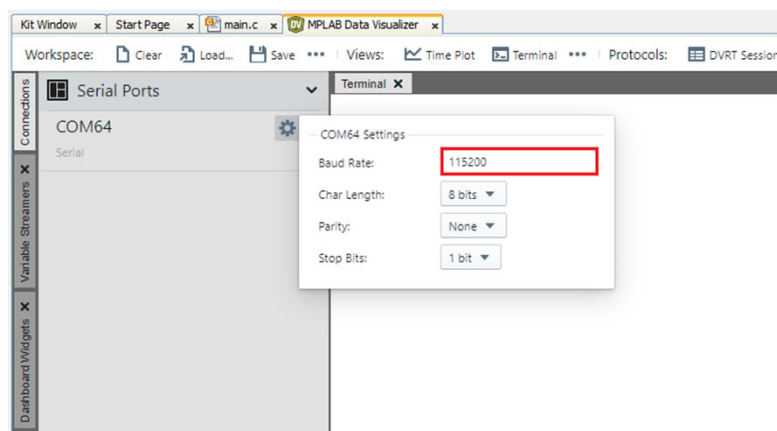
2. To configure the Serial Port of the PIC32CZ CA90 Curiosity Ultra Evaluation Board, click on the **Gear** icon as shown below:

Figure 8-2. Serial Port Setup

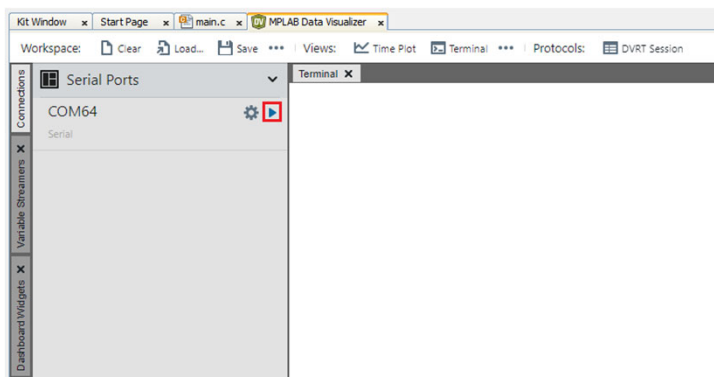


3. On the **COM Settings** property sheet, for Baud Rate, enter **115200**.

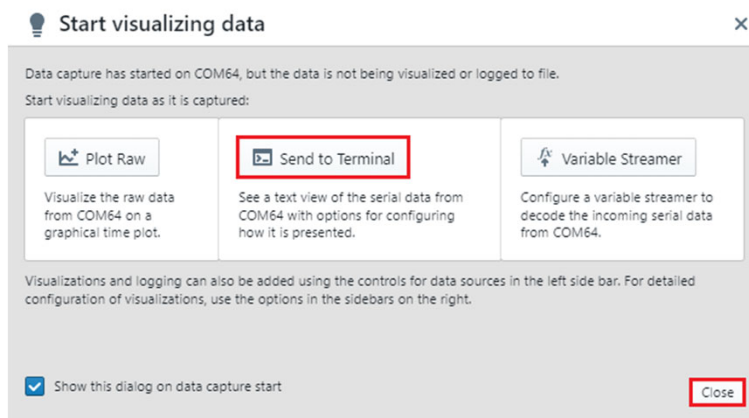
Figure 8-3. Selecting Baud Rate



4. Open the Serial Port of the PIC32CZ CA90 Curiosity Ultra Evaluation Board by clicking the **Play** icon.

Figure 8-4. Opening the Serial Port

5. Select **Send to Terminal** to visualize the TCM performance test results, and then click **Close**.

Figure 8-5. Select the Terminal Option

6. Press the **Reset** button on the Evaluation Board to reset the device.

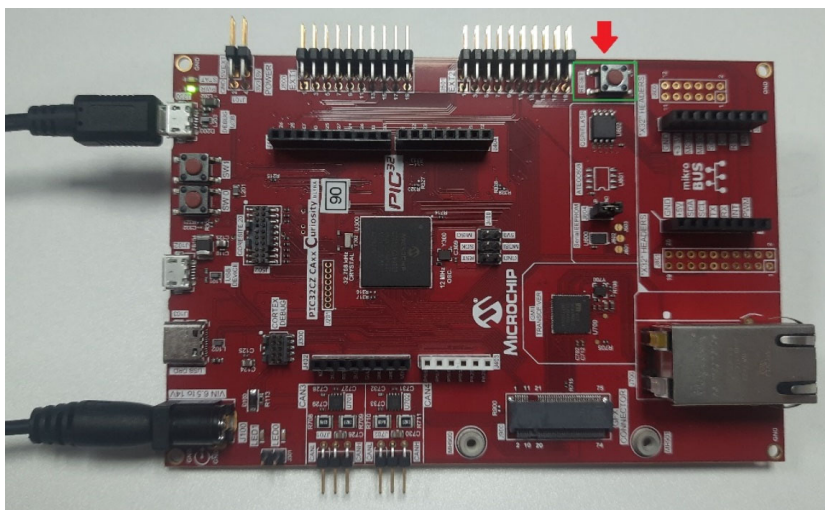
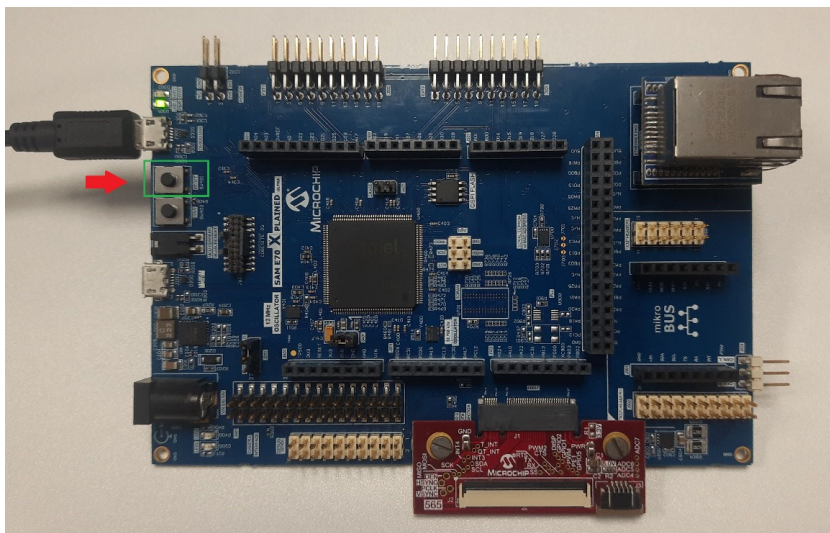
Figure 8-6. Resetting the PIC32CZ CA90 Curiosity Ultra Evaluation Board

Figure 8-7. Resetting the SAM E70 Xplained Ultra Evaluation Kit

7. Observe the console message **TCM Performance Demo** displayed on the MPLAB Data Visualizer.

Figure 8-8. TCM Performance Comparison

```

Terminal X
-----
TCM Performance Demo
-----

Disabled Instruction & Data Cache
Code/Buffer - Cycles
Flash/SRAM - 1085171
Flash/DTCM - 988133
SRAM/SRAM - 955135
SRAM/DTCM - 876161
ITCM/SRAM - 916673
ITCM/DTCM - 834131

Enabled Instruction Cache
Code/Buffer - Cycles
Flash/SRAM - 1035921
Flash/DTCM - 938207
SRAM/SRAM - 944315
SRAM/DTCM - 853327
ITCM/SRAM - 898183
ITCM/DTCM - 834117

Enabled Instruction & Data Cache
Code/Buffer - Cycles
Flash/SRAM - 166569
Flash/DTCM - 163984
SRAM/SRAM - 163995
SRAM/DTCM - 163982
ITCM/SRAM - 163981
ITCM/DTCM - 163981

```

Notes:

- a. The optimization level is set to None / O0 for both projects.
 - b. Execution metrics collected for the PIC32CZ CA and SAM E70/S70/V70/V71 family of devices are provided in the following tables:
 - [PIC32CZ CA: TCM Memory Configuration - 128 KB, Optimization – None / O0](#)
 - [SAM E70/S70/V70/V71: TCM Memory Configuration - 32 KB, Optimization – Speed / O0](#)
8. The following tables provide the benefits of ITCM and DTCM compared to traditional Flash/SRAM combinations. It is noted that the advantage is minimal when cache is enabled, as the example program is very small and fits entirely within the cache. However, real-world applications are significantly more complex and cannot be fully contained within the cache. For such applications, TCM proves to be more advantageous in managing critical and deterministic code segments.

Table 8-1. PIC32CZ CA: TCM Configuration - 128 KB, Optimization – None / O0

Configuration	Code	Buffer	Cycles
Instruction Cache = OFF; Data Cache = OFF;	Flash	SRAM	1085171
	Flash	DTCM	988133
	SRAM	SRAM	955135
	SRAM	DTCM	876161
	ITCM	SRAM	916673
	ITCM	DTCM	834131
Instruction Cache = ON; Data Cache = OFF;	Flash	SRAM	1035921
	Flash	DTCM	938207
	SRAM	SRAM	944315
	SRAM	DTCM	853327
	ITCM	SRAM	898183
	ITCM	DTCM	834117
Instruction Cache = ON; Data Cache = ON;	Flash	SRAM	166569
	Flash	DTCM	163984
	SRAM	SRAM	163995
	SRAM	DTCM	163982
	ITCM	SRAM	163981
	ITCM	DTCM	163981

Table 8-2. SAM E70/S70/V70/V71: TCM Configuration - 32 KB, Optimization – None / O0

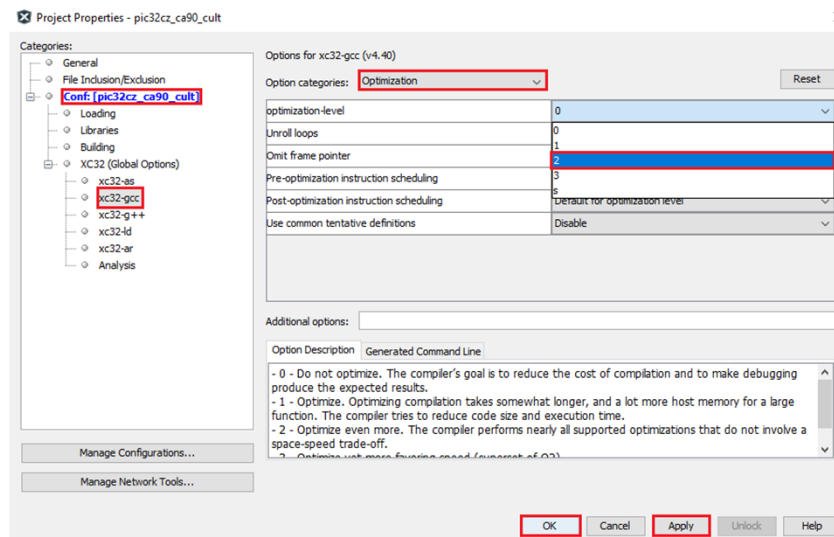
Configuration	Code	Buffer	Cycles
Instruction Cache = OFF; Data Cache = OFF;	Flash	SRAM	1317577
	Flash	DTCM	1258257
	SRAM	SRAM	1186391
	SRAM	DTCM	1064289
	ITCM	SRAM	1039041
	ITCM	DTCM	964571
Instruction Cache = ON; Data Cache = OFF;	Flash	SRAM	1266171
	Flash	DTCM	1161249
	SRAM	SRAM	1169437
	SRAM	DTCM	1051603
	ITCM	SRAM	1052657
	ITCM	DTCM	963431

.....continued

Configuration	Code	Buffer	Cycles
Instruction Cache = ON; Data Cache = ON;	Flash	SRAM	167639
	Flash	DTCM	163973
	SRAM	SRAM	164019
	SRAM	DTCM	164034
	ITCM	SRAM	163971
	ITCM	DTCM	163973

9. To enhance these figures, adjust the optimization level to O2 to observe the improvements:
 - a. In the MPLAB X IDE **Project Properties** window, under Categories, click and expand **Conf: [pic32cz_ca90_cult]** > **XC32 (Global Options)**, and then select **xc32-gcc**.
 - b. In the right **Options for xc32-gcc** properties sheet:
 - i. For Option categories, select **Optimization**.
 - ii. For Optimization level, select **2**.

Figure 8-9. Project Properties



10. Click **Apply**, and then click **OK**.
11. To rebuild and program the PIC32CZ CA90 Curiosity Ultra Evaluation Board, refer to the *Building and Programming the Application*.

Notes:

- Execution metrics collected for the PIC32CZ CA and SAM E70/S70/V70/V71 family are provided in the following tables:
 - [PIC32CZ CA: TCM Memory Configuration - 128 KB, Optimization – O2](#)
 - [SAM E70/S70/V70/V71: TCM Memory Configuration - 32 KB, Optimization – O2](#)
- These above steps can be used to change the optimization level for the SAM E70 Xplained Ultra Evaluation Kit.

Table 8-3. PIC32CZ CA: TCM Configuration - 128 KB, Optimization – O2

Configuration	Code	Buffer	Cycles
Instruction Cache = OFF; Data Cache = OFF;	Flash	SRAM	383503
	Flash	DTCM	385553
	SRAM	SRAM	179778
	SRAM	DTCM	134154
	ITCM	SRAM	181595
	ITCM	DTCM	133951
Instruction Cache = ON; Data Cache = OFF;	Flash	SRAM	179485
	Flash	DTCM	136405
	SRAM	SRAM	179496
	SRAM	DTCM	133842
	ITCM	SRAM	181515
	ITCM	DTCM	133851
Instruction Cache = ON; Data Cache = ON;	Flash	SRAM	45442
	Flash	DTCM	45203
	SRAM	SRAM	39094
	SRAM	DTCM	42650
	ITCM	SRAM	41132
	ITCM	DTCM	42640

Table 8-4. SAM E70/S70/V70/V71: TCM Configuration - 32 KB, Optimization – O2

Configuration	Code	Buffer	Cycles
Instruction Cache = OFF; Data Cache = OFF;	Flash	SRAM	394969
	Flash	DTCM	397011
	SRAM	SRAM	215731
	SRAM	DTCM	159931
	ITCM	SRAM	215575
	ITCM	DTCM	159709
Instruction Cache = ON; Data Cache = OFF;	Flash	SRAM	214951
	Flash	DTCM	163591
	SRAM	SRAM	216901
	SRAM	DTCM	161029
	ITCM	SRAM	216859
	ITCM	DTCM	161013
Instruction Cache = ON; Data Cache = ON;	Flash	SRAM	47036
	Flash	DTCM	45180
	SRAM	SRAM	41120
	SRAM	DTCM	42636
	ITCM	SRAM	41101
	ITCM	DTCM	42633

The following graphs represent the TCM performance on the PIC32CZ CA and SAM E70/S70/V70/V71 Family of devices.

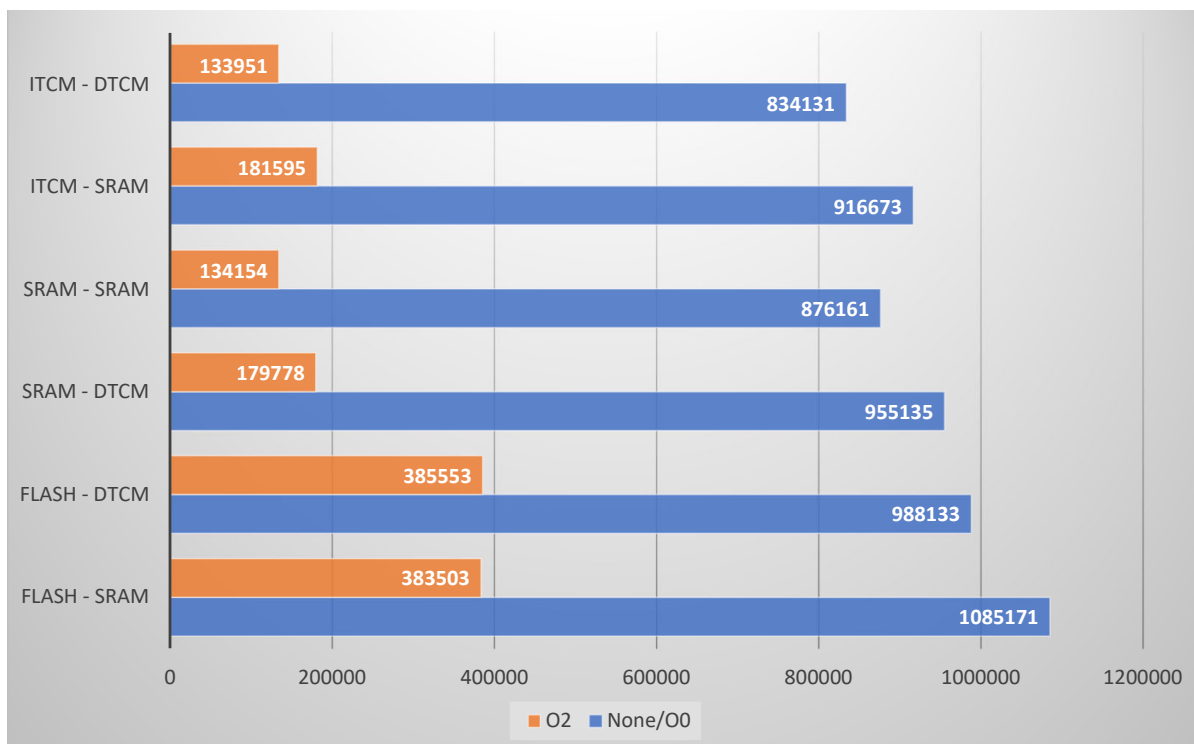
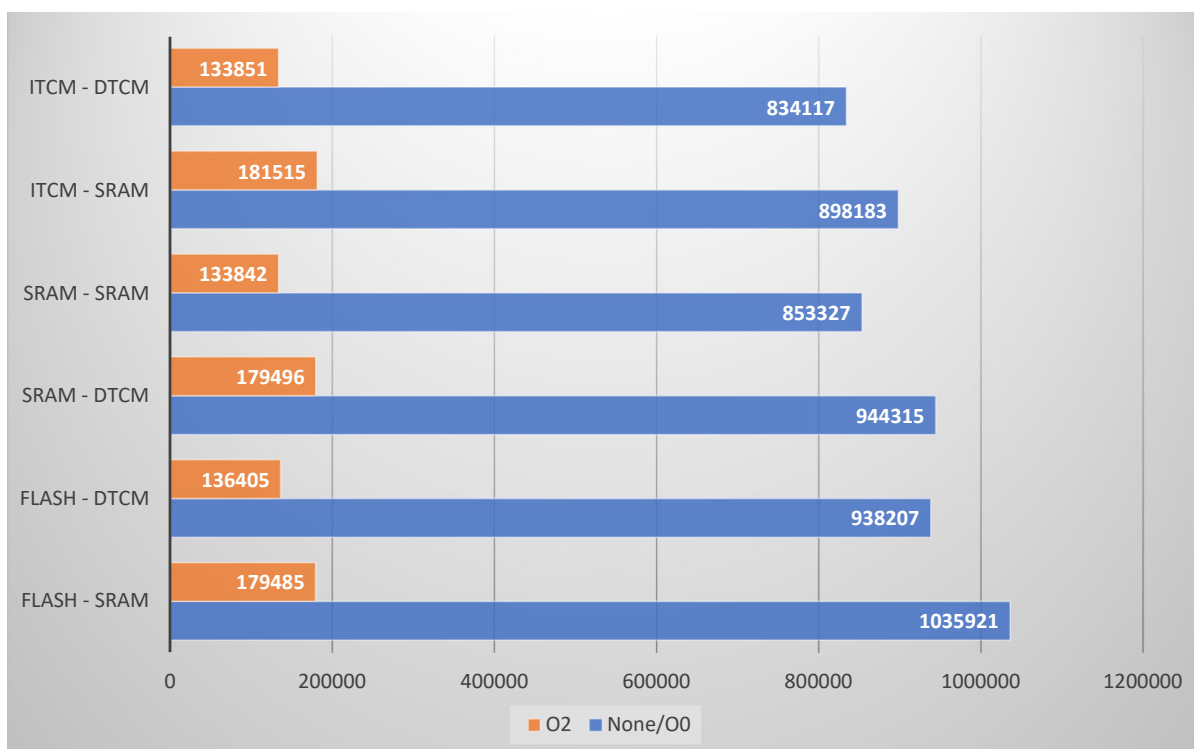
Figure 8-10. PIC32CZ CA: Disabled Instruction Cache and Data Cache**Figure 8-11.** PIC32CZ CA: Enabled Instruction Cache

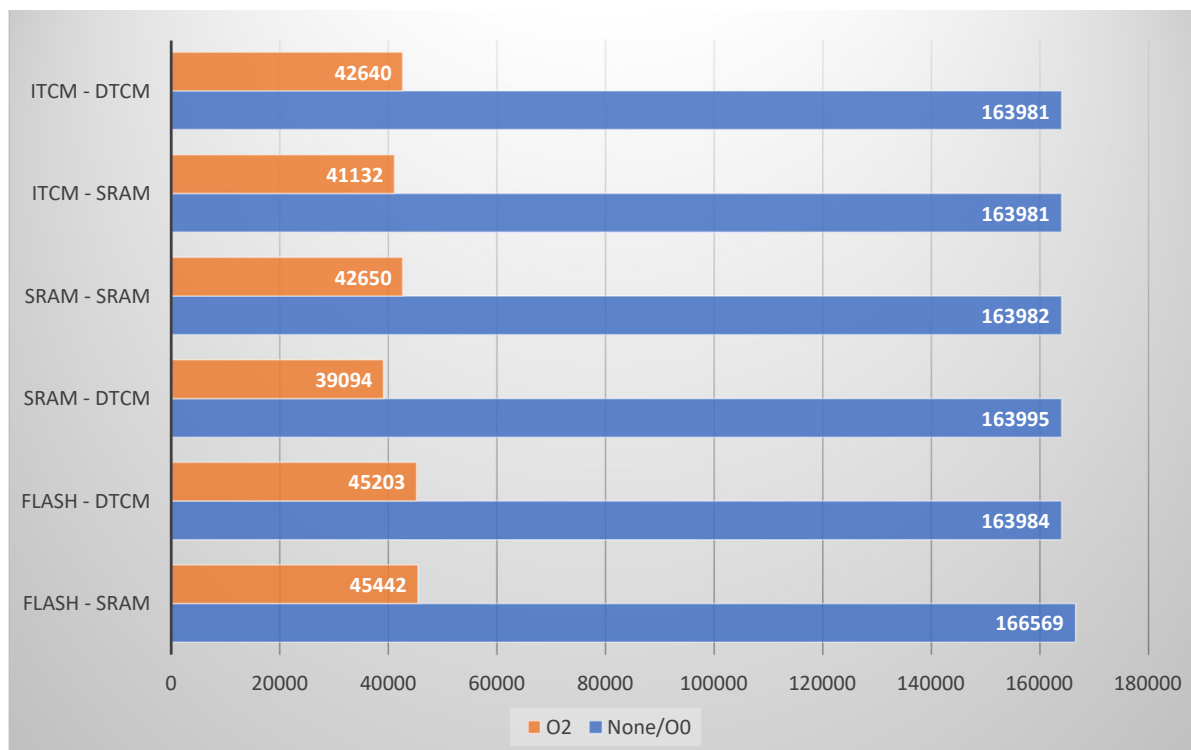
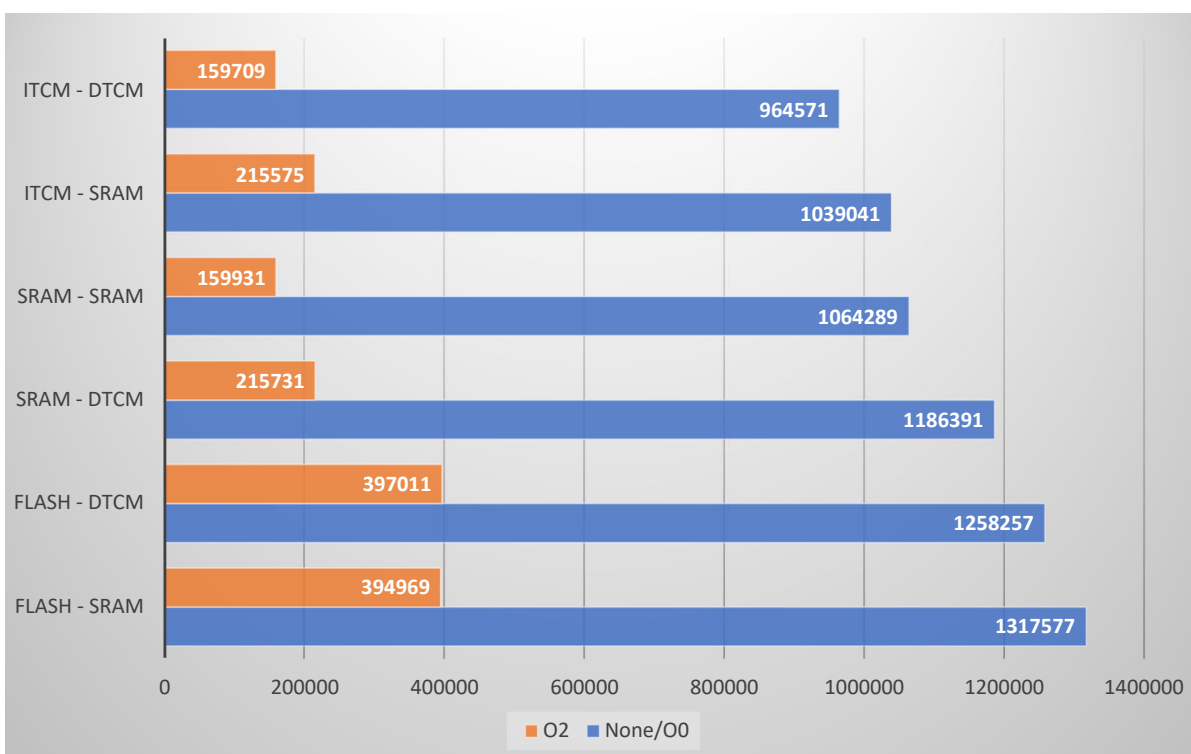
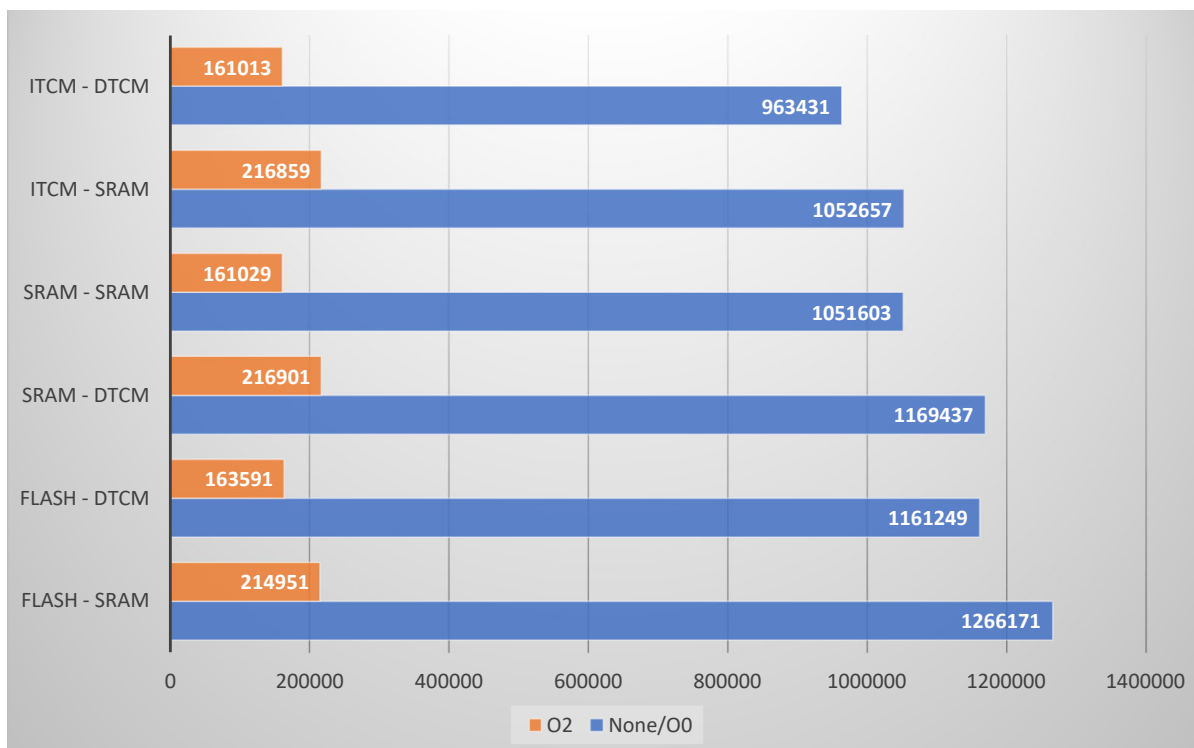
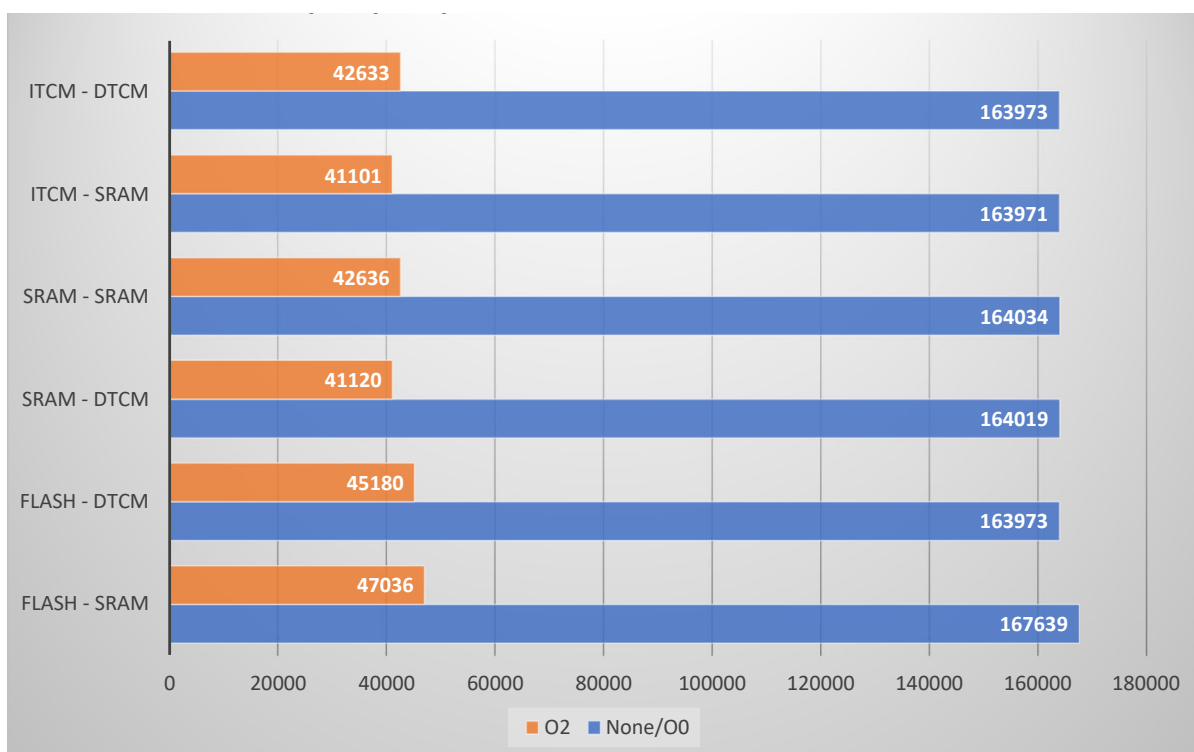
Figure 8-12. PIC32CZ CA: Enabled Instruction Cache and Data Cache**Figure 8-13.** SAM E70/S70/V70/V71: Disabled Instruction Cache and Data Cache

Figure 8-14. SAM E70/S70/V70/V71: Enabled Instruction Cache**Figure 8-15.** SAM E70/S70/V70/V71: Enabled Instruction Cache and Data Cache

9. Resources

- [PIC32CZ CA90 Curiosity Ultra Evaluation Board](#)
- [PIC32CZ CA80/CA90 Curiosity Ultra User Guide](#)
- [Creating the First Application on PIC32CZ Cx Microcontrollers Using MPLAB Harmony v3 with MPLAB Code Configurator \(MCC\)](#)
- [SAME70 Xplained Ultra Pro Evaluation Kit](#)
- [SAM E70 Xplained Ultra User's Guide](#)
- [Create your First Project with SAM E70 using MPLAB® Harmony v3](#)
- For additional information on MPLAB Harmony v3, refer to the Microchip web site:
<https://www.microchip.com/mplab/mplab-harmony> and microchipdeveloper.com/xwiki/bin/view/software-tools/harmony/
- For more information on various applications, refer to:
github.com/Microchip-MPLAB-Harmony/reference_apps
- For additional info about 32-bit Microcontroller Collaterals and Solutions, refer to:
<http://ww1.microchip.com/downloads/aemDocuments/documents/MCU32/ProductDocuments/ReferenceManuals/32-bit-Microcontroller-Collateral-and-Solutions-Reference-Guide-DS70005534.pdf>

10. Abbreviations

The following abbreviations are used in this document:

Table 10-1. Abbreviations

TCM	Tightly Coupled Memory
ITCM	Instruction Tightly Coupled Memory
DTCM	Data Tightly Coupled Memory
ECC	Error Correction Code
MPU	Memory Protection Unit
MCC	MPLAB Code Configurator
USB	Universal Serial Bus
I²S	Inter-IC Sound
QSPI	Quad-Serial Peripheral Interface
SRAM	Static Random Access Memory
ROM	Read Only Memory
DMA	Direct Memory Access
AHBS	Advanced High-Performance Bus Client
AXIM	Advanced eXtensible Interface Manager
CPU	Central Processing Unit
TCU	Tightly Coupled Interface Unit
LSU	Load Store Unit
PFU	Pre-Fetch Unit
DGU	Debug Unit
HCLK	Host Clock
MCK	Main Clock
BFM	Boot Flash Memory
PFM	Program Flash Memory
ITCMCR	Instruction Tightly Coupled Memory Control Register
VTOR	Vector Table Offset Register
SEQNUM	Sequence Number
GPVM	General-Purpose Non-Volatile Memory
KB	Kilo Bytes

11. Revision History

Revision A - 12/2024

This is the initial release of the document.

Microchip Information

Trademarks

The “Microchip” name and logo, the “M” logo, and other names, logos, and brands are registered and unregistered trademarks of Microchip Technology Incorporated or its affiliates and/or subsidiaries in the United States and/or other countries (“Microchip Trademarks”). Information regarding Microchip Trademarks can be found at <https://www.microchip.com/en-us/about/legal-information/microchip-trademarks>.

ISBN: 979-8-3371-0314-3

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip products are strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.