Application Note

DDR-Less Design Using OSPI PSRAM on AM243x/AM64x

Aryamaan Chaurasia, Laxman Chinnannavar

Abstract

This application note evaluates the OSPI of the AM64x processor with external OSPI PSRAM, which provides a cost-effective alternative to DDR memory for industrial real-time applications. To address the challenges of rising DDR costs and supply constraints, this application note details the implementation of code execution and data storage using the APS6408L OSPI PSRAM device on the modified TMDS243EVM. To validate this approach for industrial use, this application note provides comprehensive performance benchmarks using PROFINET communication profiles. These measurements quantify read/write throughput, CPU load, and fast start-up times.

System engineers can learn to integrate OSPI PSRAM into AM64x designs, configure bootloaders for code execution from OSPI PSRAM, and utilize measured performance data to optimize industrial control systems, motor drives, and PROFINET applications.

Table of Contents

1 Introduction 2
2 Hardware Architecture 3
2.1 EVM Configuration 3
2.2 APS6408L PSRAM Device 3
3 Software Architecture 4
4 Implementation Guide 5
4.1 Prerequisites 5
4.2 Example Applications 5
4.3 Integrating OSPI PSRAM into Custom Applications 7
5 Performance Analysis and Benchmarks 9
5.1 Read and Write Throughput 9
5.2 Simulated Application Benchmarks 9
5.3 PROFINET Benchmarks 11
6 Summary 14
7 References 15
8 Revision History 16

Trademarks

Sitara™ and Code Composer Studio™ are trademarks of Texas Instruments.
FreeRTOS™ is a trademark of Amazon Web Services, Inc.
TIA Portal™ and SIMATIC™ are trademarks of Siemens AG.
Saleae™ is a trademark of Saleae, Inc.
All trademarks are the property of their respective owners.

Copyright © 2026 Texas Instruments Incorporated

1 Introduction

The increasing cost and limited availability of DDR memory have become critical concerns for embedded system designers, particularly in cost-sensitive industrial applications. Global supply chain constraints and growing demand have significantly impacted bill-of-materials (BOM) costs, forcing engineers to evaluate alternative memory architectures.

The AM64x Sitara™ processor family addresses this challenge through its octal serial peripheral interface (OSPI) controller, which supports external pseudo-SRAM (PSRAM) devices. OSPI PSRAM devices allow the processor to execute code directly from external memory, effectively expanding available code space without the need for large internal RAM buffers.

This application note presents a complete implementation of OSPI PSRAM support on the modified TMDS243EVM using the APS6408L 8MB OSPI PSRAM device. The implementation includes hardware configuration, low-level driver architecture, and bootloader modifications to support execute-in-place (XIP) operation. To validate this approach for industrial use, comprehensive performance benchmarks are provided using PROFINET communication profiles. These benchmarks quantify read/write throughput, CPU load, and Fast Startup times, enabling engineers to evaluate OSPI PSRAM performance against stringent application requirements.

Copyright © 2026 Texas Instruments Incorporated

2 Hardware Architecture

2.1 EVM Configuration

The OSPI PSRAM from AP Memory is not available on the default TMDS243EVM, and is only available on the modified TMDS243EVM that follows the PROC101E1_OSPI schematic. The modification made to the default EVM is the addition of the TS3DDR3812RUAR high-speed switch that multiplexes the OSPI signals between the OSPI NOR Flash and OSPI PSRAM.

The OSPI_SW_SEL signal is driven by pin 23 of the TCA6424A I/O expander, which is controlled over I2C1. When the OSPI_SW_SEL is asserted (driven high), the OSPI signals are routed to the OSPI PSRAM, enabling the SoC to communicate with the OSPI PSRAM. By default, the OSPI_SW_SEL is low, so that the OSPI signals are routed to the OSPI NOR Flash, enabling the default boot from OSPI NOR Flash.

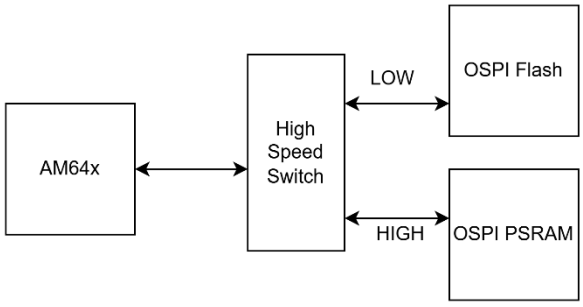


Figure 2-1. OSPI Signal Routing Through High-Speed Switch

2.2 APS6408L PSRAM Device

The APS6408L-OBMx is a volatile memory from AP Memory that supports octal DDR with a maximum frequency of 200MHz. The APS6408L-OBMx has a memory size of 8MB and supports 1KB linear burst wraps for read and write operations.

Because XIP from the external memory devices with the current AM64x OSPI controller cannot be performed in PHY mode, Tap mode is used for code execution from OSPI PSRAM.

In Tap mode, the AM64x device supports a maximum output OSPI clock frequency of 25MHz for DDR operations and 50MHz for SDR operations.

This means that the OSPI PSRAM is operating at an output OSPI clock frequency of 25MHz with octal DDR. Since DQS (Data Strobe) is not being used, the OSPI PSRAM is set to fixed-latency mode, accounting for any automatic self-refresh collisions that can occur when trying to read the data from the OSPI PSRAM.

Because the OSPI PSRAM and OSPI NOR Flash share the same OSPI controller, the memory-mapped region starting at 0x60000000 is the same for both devices. Hence, a high-speed switch is implemented to prevent bus contention and accidental overwrites of either memory device's contents.

2.2.1 OSPI PSRAM Configurations

Table 2-1. OSPI PSRAM Configurations

Field	Value
OSPI Input Clock Frequency	200MHz
OSPI Clock Divider	8
Linear Burst Read Command	0x20
Linear Burst Write Command	0xA0
Read Register Command	0x40
Write Register Command	0xC0
Reset Command	0xFF
Row Boundary Crossing	Enabled

Copyright © 2026 Texas Instruments Incorporated

3 Software Architecture

The OSPI PSRAM implementation is provided as a collection of example applications, SDK patches, and reference documentation organized in the Beyond-SDK Repository. Two patches extend the MCU+ SDK with OSPI PSRAM support: the first patch optimizes OSPI DAC (Direct Access Controller) write operations by replacing byte-by-byte transfers with DMA-enabled MEMCPY operations, while the second patch adds the complete OSPI PSRAM driver stack, including board-level APIs, device-specific APS6408L support, and I/O expander control for multiplexing between OSPI NOR Flash and OSPI PSRAM.

The example applications demonstrate three key use cases for OSPI PSRAM integration. Two secondary bootloader (SBL) examples, SBL OSPI PSRAM and SBL SD PSRAM, showcase boot-time section loading from non-volatile storage to OSPI PSRAM with dynamic switch control, enabling XIP operation. The OSPI PSRAM I/O example provides basic read/write verification and throughput measurement, serving as a communication validation test between the OSPI controller and OSPI PSRAM device. The Hello World example demonstrates linker configuration and SysConfig settings required to place application sections in OSPI PSRAM memory regions.

The examples, patches, and documents can be found in the Beyond-SDK Repository.

- The SBL OSPI PSRAM example can be found in the following path: Beyond-SDK/am64x/mcupsdk/ospi_psram/examples/drivers/boot/sbl_ospi_psram
- The SBL SD PSRAM example can be found in the following path: Beyond-SDK/am64x/mcupsdk/ospi_psram/examples/drivers/boot/sbl_sd_psram
- The Hello World example can be found in the following path: Beyond-SDK/am64x/mcupsdk/ospi_psram/examples/hello_world_psram
- The OSPI PSRAM I/O example can be found in the following path: Beyond-SDK/am64x/mcupsdk/ospi_psram/examples/drivers/ospi/ospi_psram_io
- The .cfg file for flashing the application images can be found in the following path: Beyond-SDK/am64x/mcupsdk/ospi_psram/tools/boot/sbl_prebuilt/am64x-evm/default_sbl_ospi_psram.cfg
- The patches are in the "patches" folder at the following path: Beyond-SDK/am64x/mcupsdk/ospi_psram/patches
- The OSPI PSRAM datasheet and the PROC101E1_OSPI schematic can be found in the "Docs" folder at the following path: Beyond-SDK/am64x/mcupsdk/ospi_psram/Docs

Copyright © 2026 Texas Instruments Incorporated

4 Implementation Guide

4.1 Prerequisites

- The modified TMDS243EVM is required to flash and run all the following example applications.
- All the following example applications and patches are compatible with the AM64x MCU+ SDK 12.00.00.27 [1] and the AM64x Industrial Communications SDK 2026.00.00.06 [2].
- Apply the patches before building the following example applications.

4.2 Example Applications

4.2.1 Secondary Bootloaders

4.2.1.1 SBL OSPI PSRAM

This secondary bootloader enables code execution from OSPI PSRAM by transferring the XIP image (the image ending with the .mcelf_xip extension) from the OSPI NOR Flash to the OSPI PSRAM. By dynamically switching between the OSPI NOR Flash and OSPI PSRAM, this example also demonstrates how different sections are loaded to different memory regions such as the OSPI PSRAM.

Perform the following steps to enable code execution from OSPI PSRAM using SBL OSPI PSRAM:

1. After building the example, use the offsets specified in default_sbl_ospi_psram.cfg to flash the .mcelf.hs_fs and the .mcelf_xip images.
2. After flashing the binaries in UART boot mode, power off the EVM, switch to the OSPI boot mode, and power on the EVM.

Note

Use the offsets specified in the .cfg file, because the offset at which the XIP image is written must match the XIP_FLASH_OFFSET defined in the main.c file of the SBL OSPI PSRAM example.

When the OSPI_SW_SEL is driven high, the following memory layout of the OSPI PSRAM is visible to the CPU in DAC mode:

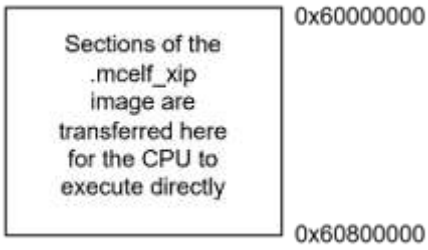


Figure 4-1. OSPI PSRAM Memory Layout

When the OSPI_SW_SEL is driven low, the following memory layout of the OSPI NOR Flash is visible to the CPU in DAC mode:

Copyright © 2026 Texas Instruments Incorporated

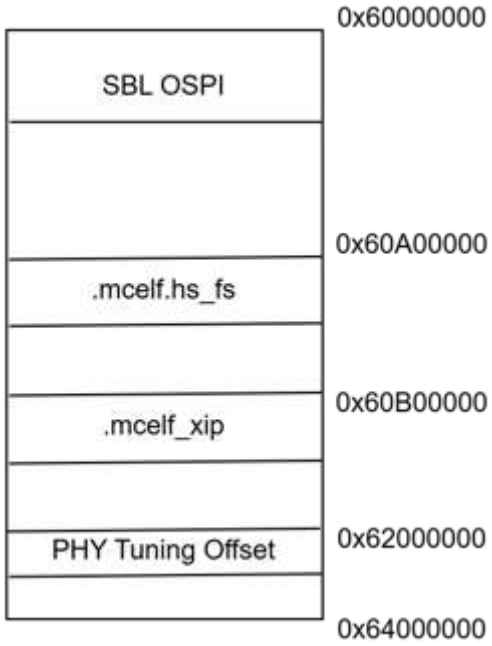


Figure 4-2. OSPI NOR Flash Memory Layout

Note

The memory region defined for OSPI PSRAM must always be marked as "Cached" and the memory region for OSPI NOR Flash must be marked as "Strongly Ordered" if Flash writes are being performed in the memory region.

4.2.1.2 SBL SD PSRAM

This secondary bootloader enables code execution from OSPI PSRAM by transferring the XIP image from the SD card to the OSPI PSRAM.

Perform the following steps to enable code execution from OSPI PSRAM using SBL SD PSRAM:

1. Copy the tiboot3.bin file from the sbl_sd_psram example to the SD card. Rename the .mcelf.hs_fs binary to "app" and the .mcelf_xip binary to "app_xip", and copy them to the SD card.
2. Insert the SD card, set the SD boot mode, and power on the EVM.

Note

The above examples only demonstrate the enablement and operation of code execution from OSPI PSRAM. The size of the buffers, the OCRAM (on-chip RAM) region, and the values defined in the above two SBLs can be changed to suit the application.

4.2.2 OSPI PSRAM I/O

This example demonstrates the communication between the OSPI controller present in the SoC and the OSPI PSRAM. This application writes 512KB of randomly generated data into the OSPI PSRAM, and the same data is read back and compared. If there is no mismatch between the data sent and received, the read and write throughputs are then calculated.

This example can be run by using SBL NULL, then loading and running the .out file through Code Composer Studio™ (CCS). This application can also be run with SBL OSPI PSRAM or SBL SD PSRAM by following the steps described in Section 4.2.1.1 and Section 4.2.1.2.

Copyright © 2026 Texas Instruments Incorporated

4.2.3 Hello World with Code Execution from OSPI PSRAM

This example demonstrates how the memory region can be configured to enable OSPI PSRAM and place sections of the application into OSPI PSRAM.

This example application can also be run with SBL OSPI PSRAM or SBL SD PSRAM by following the steps described in Section 4.2.1.1 and Section 4.2.1.2.

4.3 Integrating OSPI PSRAM into Custom Applications

To enable sections of the application to execute from OSPI PSRAM, perform the following steps:

1. Use the SBL OSPI PSRAM or SBL SD PSRAM and perform the necessary configurations detailed in Section 4.2.1.1 and Section 4.2.1.2.
2. On the application side, open SysConfig, and in the MPU ARMv7 module, add the following memory configuration for the memory-mapped region used by OSPI PSRAM, as shown in Figure 4-3:

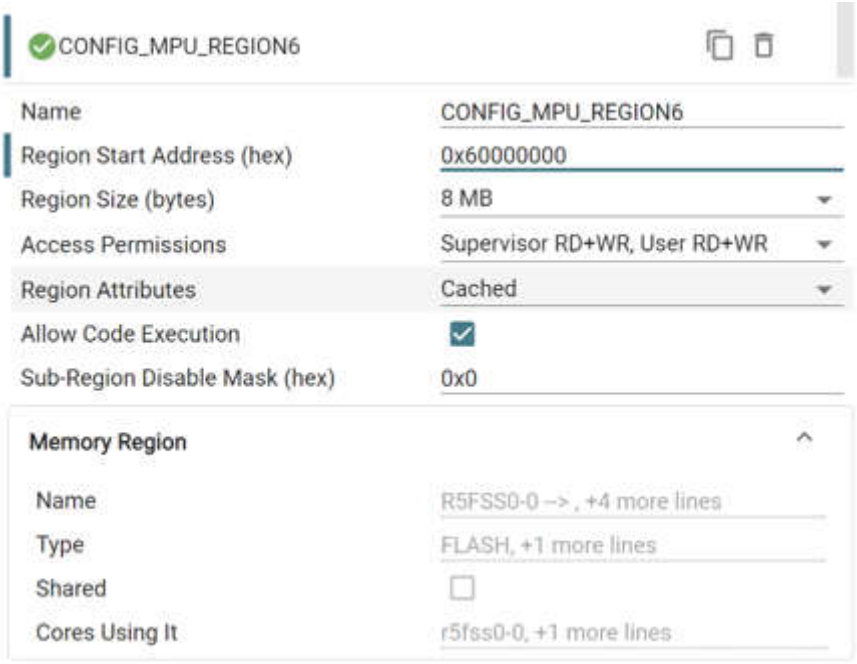


Figure 4-3. MPU Region Configuration for OSPI PSRAM

3. Next, in the Memory Region tab, add the following memory region, as shown in Figure 4-4:

Note

Because the OSPI NOR Flash and OSPI PSRAM share the same memory region, the memory region starting at 0x60000000 is set as "FLASH" type. This is only a naming convention; the same memory region can be configured as OSPI PSRAM by performing the above steps.

Copyright © 2026 Texas Instruments Incorporated

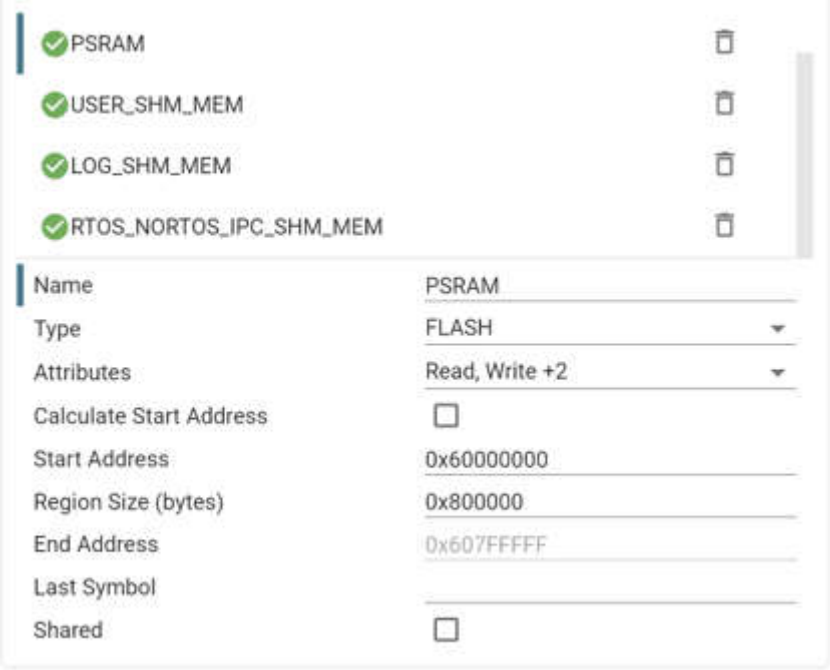


Figure 4-4. Memory Region Configuration for OSPI PSRAM

4. After completing the above steps, in the Sections tab, place the required sections in the OSPI PSRAM memory region as shown in Figure 4-5:

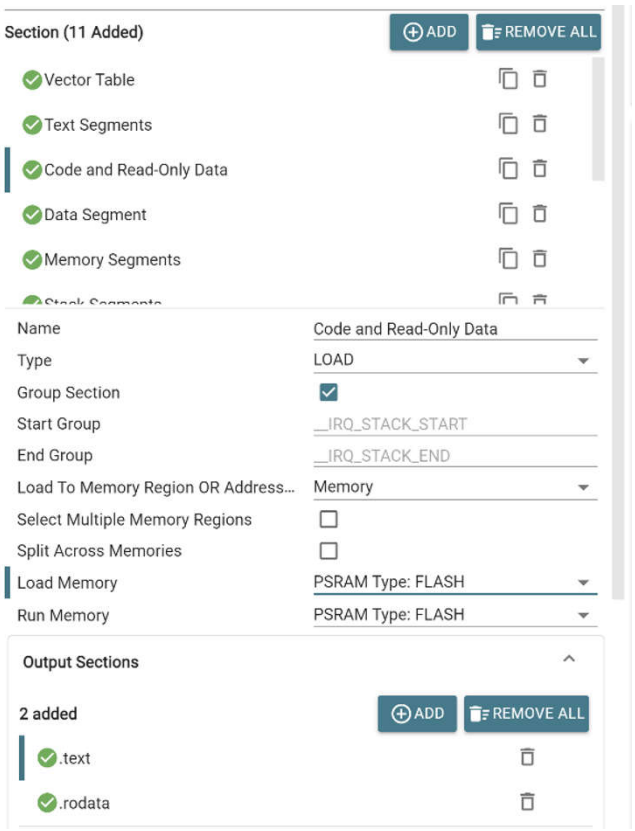


Figure 4-5. Configuring Sections to Load into OSPI PSRAM

Copyright © 2026 Texas Instruments Incorporated

5 Performance Analysis and Benchmarks

Note

Due to the PHY tuning limitation with XIP, the following benchmarks have been obtained by operating the OSPI PSRAM and OSPI NOR Flash in Tap mode, which limits the output OSPI clock frequency to 25MHz.

Table 5-1 shows the setup used for benchmarking the throughput and the two simulated examples:

Table 5-1. Benchmark Setup for Simulated Examples

Board	Modified TMDS243EVM
Device	AM6442
DDR4 Part	MT40A1G16KD-062E
OSPI NOR Flash Part	S28HS512TGABHM010
OSPI PSRAM Part	APS6408L-OBMx
Core	R5F

5.1 Read and Write Throughput

Table 5-2 shows the read and write throughput obtained across various memory devices:

Table 5-2. Read and Write Throughput Across Different Memory Devices

Memory Devices	DDR		OSPI PSRAM		OSPI NOR Flash	
Interface Frequency	800MHz		25MHz DDR		25MHz DDR	
DMA	Enabled		Enabled		Enabled	
Theoretical Throughput	3200MBps		50MBps		50MBps	
Memory-Mapped Region	Cached		Cached		Strongly Ordered	
Operations	Write	Read	Write	Read	Write	Read
Data Size - 512KB	975MBps	975MBps	40.2MBps	49.9MBps	0.43MBps(1)	48.7MBps

(1) The default S28HS512T OSPI NOR Flash present on the TMDS243EVM does not support DAC writes with the OSPI controller. Hence, INDAC (Indirect Access Controller) writes are used, which do not have DMA and PHY support.

5.2 Simulated Application Benchmarks

5.2.1 XIP Benchmark

Example used: XIP Benchmark

Table 5-3 shows the performance metrics obtained for the XIP benchmark example:

Table 5-3. Performance Metrics for XIP Benchmark

Memory Devices	OCRAM	DDR	OSPI PSRAM (25MHz DDR)	OSPI NOR Flash (25MHz DDR)
MPU Region Configuration				
Region Attributes	Cached	Cached	Cached	Cached
Sections				
RO Data, Text	OCRAM	DDR	PSRAM	Flash
Rest of the Application	OCRAM	OCRAM	OCRAM	OCRAM
Benchmarks				

Copyright © 2026 Texas Instruments Incorporated

Table 5-3. Performance Metrics for XIP Benchmark (continued)

Memory Devices			OCRAM	DDR	OSPI PSRAM (25MHz DDR)	OSPI NOR Flash (25MHz DDR)
CPU Stats (Number of CPU cycles used)	FIR	Min	24072	24072	24072	24072
		Max	28447	38103	74978	93792
		Avg	24512	25510	29654	31026
	MEMCPY	Min	1565	1565	1565	1565
		Max	2010	3040	5203	8608
		Avg	1614	1720	1932	2275
Relative Degradation with respect to OCRAM (Max cycles)	FIR		1x	Approximately 1.3x	Approximately 2.6x	Approximately 3.3x
	MEMCPY		1x	Approximately 1.5x	Approximately 2.6x	Approximately 4.3x
Relative Degradation with respect to OCRAM (Average cycles)	FIR		1x	Approximately 1x	Approximately 1.2x	Approximately 1.3x
	MEMCPY		1x	Approximately 1.1x	Approximately 1.2x	Approximately 1.4x

The XIP Benchmark measures how the memory that holds the code and read-only data affects a CPU-bound workload running in XIP mode. The SoC boots and runs the application on the R5F core under FreeRTOS™. The application executes two functions: FIR and MEMCPY.

The linker command file places the text and read-only (RO) data sections in the memory under test, and the rest of the application remains in OCRAM. In the baseline case, all sections are placed in OCRAM. Each function is run a total of 10 times and the worst-case, best-case, and average-case performance is printed on the UART terminal in units of CPU cycles. The maximum (which comes from the cold-cache first run) and average number of CPU cycles for OCRAM are taken as the baseline and the relative performance degradation are then calculated for each test case.

5.2.2 On-Chip Memory Benchmark

Example used: On-Chip Memory Benchmark

Table 5-4 shows the performance metrics obtained for the on-chip memory benchmark example:

Table 5-4. Performance Metrics for On-Chip Memory Benchmark

Memory Devices	OCRAM	DDR		OSPI PSRAM (25MHz DDR)		OSPI NOR Flash (25MHz DDR)
MPU Region Configuration						
Region Attributes	Cached	Cached		Cached		Cached
Sections						
Startup Code	OCRAM	OCRAM	OCRAM	OCRAM	PSRAM	OCRAM
RO Data, Text	OCRAM	DDR	DDR	PSRAM	PSRAM	Flash
RW Data	OCRAM	OCRAM	DDR	OCRAM	PSRAM	OCRAM
Benchmarks						
Memory Copy Size (Bytes)	2250	2250	2250	2250	2250	2250
Execution Time in µsec	17022	25871	30980	66428	137616	68925
Relative Degradation with respect to OCRAM	1x	Approximately 1.5x	Approximately 1.8x	Approximately 3.9x	Approximately 8.1x	Approximately 4.0x
Number of Instruction Cache Misses (ICM)	56082	56537	56808	59923	60199	57938
Number of Data Cache Misses	38631	41509	39886	43520	42676	43264

Copyright © 2026 Texas Instruments Incorporated

Table 5-4. Performance Metrics for On-Chip Memory Benchmark (continued)

Memory Devices	OCRAM	DDR		OSPI PSRAM (25MHz DDR)		OSPI NOR Flash (25MHz DDR)
Number of Instructions Executed (INST)	6688800 Approximately 6.69 Million	6694168 Approximately 6.69 Million	6694944 Approximately 6.69 Million	6700356 Approximately 6.7 Million	6707012 Approximately 6.71 Million	6700748 Approximately 6.7 Million
ICM/sec	3294677 Approximately 3.29 Million	2185342 Approximately 2.19 Million	1833699 Approximately 1.83 Million	902074 Approximately 0.90 Million	437441 Approximately 0.44 Million	840594 Approximately 0.84 Million
INST/sec	392950299 Approximately 393 Million	258751807 Approximately 259 Million	216105358 Approximately 216 Million	100866441 Approximately 101 Million	48737152 Approximately 49 Million	97217961 Approximately 97 Million

Table 5-5. Datapoints Obtained for On-Chip Memory Benchmark

Application Output	Description
Mem Cpy size => 2250	Size of the MEMCPY in bytes executed by each task
Exec Time in µsec => 17022	Execution time in microseconds
Iter => 1	Number of times the test was run
Task calls => 500	Number of randomly ordered calls to the 16 tasks
Inst Cache miss (ICM) => 56082	Total instruction cache misses
Data Cache miss (DCM) => 38631	Total data cache misses
Num Switches => 1469	Total number of context switches
Num Instr Exec => 6688800	Total number of executed instructions
ICM/sec => 3294677	Instruction cache misses per second
INST/sec => 392950299	Instructions executed per second

The On-Chip Memory Benchmark example measures the performance of a simulated application whose sections are placed in OCRAM or in external memory (DDR, OSPI PSRAM, or OSPI NOR Flash). The application runs 10 configurations of the same code that differ only in MEMCPY size. Each test makes 500 randomly ordered calls across the 16 tasks.

Table 5-5 shows the 2250-byte configuration. This is the most instruction-intensive configuration. When run entirely from OCRAM, it generates about 3.3 million instruction cache misses per second (ICM/sec), a rate similar to that seen in real-world customer applications. Configurations with larger copy sizes are more data-intensive and spend more time in repetitive code, so their ICM rates are much lower.

5.3 PROFINET Benchmarks

Table 5-6 shows the setup used for benchmarking the PROFINET examples:

Table 5-6. Benchmark Setup for PROFINET

Pins used	Pin 23 of the J1 connector for GND
	TP33 for MCU_PORz
	Pin 5 of the J1 connector for the GPIO signal
Board	Modified TMDS243EVM
Device	AM6442
DDR4 Part	MT40A1G16KD-062E
OSPI NOR Flash Part	S28HS512TGABHM010
OSPI PSRAM Part	APS6408L-OBMx
Software	TIA Portal™ V21
PLC	SIMATIC™ S7-1500
Logic Analyzer	Saleae™ Logic 8

Copyright © 2026 Texas Instruments Incorporated

Note

The data points obtained using the OSPI NOR Flash operating at 166MHz are used only to show the difference in performance when compared with the OSPI NOR Flash operating at 25MHz.

5.3.1 Performance Degradation using Fast Startup Time

Example used: PROFINET Device Demo

To depict the performance degradation, only the text and read-only data were moved to the memory under test.

Table 5-7 shows the memory configurations used for profiling Fast Startup (FSU) times:

Table 5-7. Memory Layout Configurations for Fast Startup Time

Memory Sections	DDR Case	OSPI PSRAM Case	OSPI NOR Flash Case
Vectors	TCMA	TCMA	TCMA
HWI & TCM text	TCMA	TCMA	TCMA
Stack	OCRAM	OCRAM	OCRAM
RW Data	OCRAM	OCRAM	OCRAM
BSS	OCRAM	OCRAM	OCRAM
Startup Code	OCRAM	OCRAM	OCRAM
ICSS Packet Buffer	OCRAM	OCRAM	OCRAM
Sys Mem	OCRAM	OCRAM	OCRAM
Text (803KB)	DDR	PSRAM	Flash
RO Data (66KB)	DDR	PSRAM	Flash

FSU time is obtained by measuring the difference between two hardware signals. The first signal is measured after the device has been powered on. For this, the MCU_PORz signal is used. The second signal is measured after the first successful PROFINET cyclic data exchange. The time between the two signals going high is the FSU time. The Delta time shown is the FSU time minus SBL time and indicates the time taken for the RO data and the text sections to be executed from DDR/OSPI PSRAM/OSPI NOR Flash.

The SBL times vary significantly across memory configurations due to differences in data transfer requirements and OSPI controller reconfiguration overhead. For DDR execution, the SBL transfers application sections from OSPI NOR Flash to DDR, requiring 103ms. For OSPI NOR Flash execution at 25MHz DDR, sections remain in OSPI NOR Flash after initial programming, reducing SBL time to 55ms because no transfer occurs. OSPI NOR Flash execution at 166MHz DDR adds PHY tuning algorithm overhead of approximately 25ms, resulting in an 81ms SBL time.

OSPI PSRAM execution exhibits the longest SBL time of 194ms due to the volatile nature of OSPI PSRAM combined with the shared OSPI controller architecture. The SBL must load sections into a temporary OCRAM buffer, strip the section headers, and manually transfer the contents to the OSPI PSRAM, frequently toggling the multiplexing switch and reinitializing OSPI drivers throughout the process. The AM64x device contains only one OSPI controller with a single set of configuration registers, and the sections have to be transferred from the OSPI NOR Flash to the OSPI PSRAM. As a result, the OSPI PSRAM SBL time is inherently higher than that of the other configurations.

Table 5-8 shows the Fast Startup times obtained using the above setup:

Table 5-8. Performance Metrics for Fast Startup Time

Time Taken	DDR	OSPI PSRAM (25MHz DDR)	OSPI NOR Flash (25MHz DDR)	OSPI NOR Flash (166MHz DDR)
Rest of the Application	OCRAM, TCM	OCRAM, TCM	OCRAM, TCM	OCRAM, TCM
RO Data, Text	DDR	PSRAM	Flash	Flash
SBL (in ms)	103	194	55	81
FSU (in ms)	519	867	735	701

Copyright © 2026 Texas Instruments Incorporated

Table 5-8. Performance Metrics for Fast Startup Time (continued)

Time Taken	DDR	OSPI PSRAM (25MHz DDR)	OSPI NOR Flash (25MHz DDR)	OSPI NOR Flash (166MHz DDR)
Delta (FSU-SBL)	416	673	680	620

5.3.2 Performance Degradation using CPU Load

Example used: PROFINET Device Demo

To depict the performance degradation, only the text and read-only data were moved to the memory under test.

Table 5-9 shows the memory configurations used for profiling CPU load:

Table 5-9. Memory Layout Configurations for CPU Load

Memory Sections	DDR Case	OSPI PSRAM Case	OSPI NOR Flash Case
Vectors	TCMA	TCMA	TCMA
HWI & TCM text	TCMA	TCMA	TCMA
Stack	OCRAM	OCRAM	OCRAM
RW Data	OCRAM	OCRAM	OCRAM
BSS	OCRAM	OCRAM	OCRAM
Startup Code	OCRAM	OCRAM	OCRAM
ICSS Packet Buffer	OCRAM	OCRAM	OCRAM
Sys Mem	OCRAM	OCRAM	OCRAM
Text (880KB)	DDR	PSRAM	Flash
RO Data (66KB)	DDR	PSRAM	Flash

The memory layouts used to profile CPU load are identical to those used to measure FSU performance degradation. The Text section has increased (from 803KB to 880KB) due to added code for CPU profiling. The Base CPU Load measures the CPU load generated by the communication established between the TMDS243EVM and the PLC. During the Netload Test, 30Mbps of background traffic is injected while the TMDS243EVM communicates with the PLC. This injection results in a considerable increase in CPU load.

The cycle time used in this benchmark was 1ms, and Table 5-10 below lists the highest (worst-case) CPU load obtained in each test case. No data packets were lost, and each test case was able to meet all the timing requirements.

Table 5-10. Performance Metrics for CPU Load

CPU Load	DDR	OSPI PSRAM (25MHz DDR)	OSPI NOR Flash (25MHz DDR)	OSPI NOR Flash (166MHz DDR)
Rest of the Application	OCRAM, TCM	OCRAM, TCM	OCRAM, TCM	OCRAM, TCM
RO Data, Text	DDR	PSRAM	Flash	Flash
Base CPU Load (with PLC communication)	23.31%	65.49%	69.57%	34.27%
CPU Load With Netload Test 30Mbps Traffic	66.61%	95.18%	96.90%	76.51%

Copyright © 2026 Texas Instruments Incorporated

6 Summary

This application note presents a complete OSPI PSRAM implementation on the AM64x processor using the APS6408L device, detailing both hardware and software architecture required for code execution from OSPI PSRAM. The hardware section covers the EVM configuration with high-speed multiplexing between OSPI NOR Flash and OSPI PSRAM along with detailed device specifications, while the software section describes the driver stack implementation and bootloader integration necessary for code execution from OSPI PSRAM.

This application note also covers a comprehensive performance analysis that compares OSPI PSRAM against DDR and OSPI NOR Flash. At the 25MHz Tap-mode clock currently required for XIP, OSPI PSRAM reaches 49.9 MBps read and 40.2 MBps write throughput, and its worst-case XIP cycle count is approximately 2.6 times that of OCRAM, compared with 3.3 times to 4.3 times for OSPI NOR Flash. In the PROFINET application, with text and read-only data moved to the memory under test and no further optimization, OSPI PSRAM met the 1ms cycle time with no packet loss. Its base CPU load was about 2.8 times that of DDR (65.49% vs. 23.31%), and its startup time after the SBL was about 1.6 times that of DDR (673ms vs. 416ms).

These performance characteristics help system engineers integrate OSPI PSRAM into AM64x designs for industrial control systems, motor drives, and PROFINET applications.

Copyright © 2026 Texas Instruments Incorporated

7 References

1. Texas Instruments, AM64x MCU+ SDK, design resource.
2. Texas Instruments, AM64x Industrial Communications SDK, design resource.

Copyright © 2026 Texas Instruments Incorporated

8 Revision History

Changes from Revision * (October 2026) to Revision A (October 2026)	Page
• Added AM243x to title	0
• Updated the numbering format for tables, figures, and cross-references throughout the document	0
• Changed the board names from TMDS64EVM to TMDS243EVM throughout document	2
• Corrected Table 5-2 to show *Interface Frequency*	9

Copyright © 2026 Texas Instruments Incorporated

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you fully indemnify TI and its representatives against any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale, TI’s General Quality Guidelines, or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products. Unless TI explicitly designates a product as custom or customer-specified, TI products are standard, catalog, general purpose devices.

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

Copyright © 2026, Texas Instruments Incorporated

Last updated 10/2025