

Enhancing System Performance Using OptiFlash Memory Technology



Deepshikha Gusain, Shivasharan Nagalika and Mihir Mody

ABSTRACT

Emerging applications in the Microcontroller world have led to an increase in memory and performance requirements. For instance, application software in automotive industry like networking, zonal, and so forth that runs on high-performance microcontrollers demand for more memory size for storing firmware, stacks and libraries, and more processing power- multi-core CPUs running at frequencies of 200MHz to 1GHz. Firmware updates further add up to the memory requirement, for downloading the new application image, in addition to what is available for running the existing image.

OptiFlash technology, which is TI's differentiated memory technology, enables high performance and low-cost solution by right mix of on-chip SRAM and external flash. AM26x family of SoCs with integrated OptiFlash technology as discussed in this paper aims to solve the limitations faced by traditional high-performance MCUs with external flash, by providing hybrid execution from internal SRAM and directly from external flash (also known as, execute in place (XIP)), with a goal of reaching XIP performance from external flash to that of internal SRAM.

The major cost advantage of this solution is that the internal SRAM can now be sized lower than the application image size, and the silicon die-cost without embedded flash/NVM is much cheaper due to reduced number of masks and cheaper equipment. This solution also provides scalability in terms of the size (in the order of MBytes) and parts on Board based on Customer need.

Table of Contents

1 Introduction	2
2 OptiFlash Technology	3
3 OptiFlash Hardware Accelerators	3
3.1 RL2_OF Accelerators.....	4
3.1.1 RL2-Flash Cache.....	4
3.1.2 FLC - Fast Local Copy (Image Download Acceleration).....	4
3.1.3 Region-Based Address Translation (RAT).....	5
3.2 FSS Accelerators.....	5
3.2.1 On-the-fly-Safety Engine.....	5
3.2.2 On-the-fly-Security Engine.....	5
3.2.3 FOTA HW Engine.....	5
4 OptiFlash SW Tooling	6
4.1 Smart Placement.....	6
4.2 Smart Layout.....	6
4.3 Optishare.....	6
4.4 Dynamic Overlay.....	6
5 Benchmarks and Performance Data	6
6 Usecases for OptiFlash Accelerators	7
7 Getting Started With OptiFlash	7
8 Conclusion	8

List of Figures

Figure 1-1. Traditional MCU.....	2
Figure 1-2. High Performance MCU.....	2

Figure 2-1. Proposed MCU With OptiFlash..... 3
 Figure 3-1. OptiFlash Key Hardware Components Diagram..... 4
 Figure 3-2. FLC - Image Download..... 4
 Figure 3-3. OptiFlash FOTA HW With RWW Flash..... 5
 Figure 5-1. Benchmarks..... 6
 Figure 7-1. Standard Application Development Flow..... 7
 Figure 7-2. Enhanced Application Development Flow With OptiFlash..... 8

List of Tables

Table 6-1. Usecases for OptiFlash Accelerators..... 7

Trademarks

Arm® is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All trademarks are the property of their respective owners.

1 Introduction

Current indicators suggest that traditional MCUs with embedded flash on the same die, as shown in [Figure 1-1](#), will not scale beyond a 22 nm technology node due to high-voltage circuits required for programming and erasing flash. Flash-based processes are also costly due to large number of masks required to integrate embedded flash on MCU/SOC die using deep sub-micron digital CMOS technology. Thus, these MCUs will only work for older process nodes (such as 28nm, 40nm, 65nm...).

There are also alternate new NVM technologies coming up such as MRAM, RRAM, and so forth, but even these are not yet ready for high-reliability applications such as automotive.

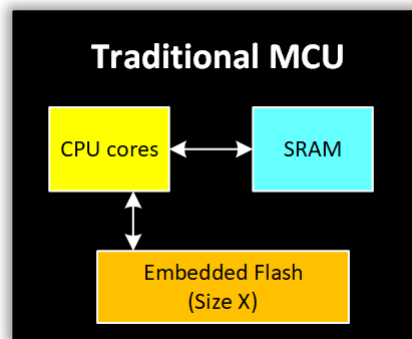


Figure 1-1. Traditional MCU

High-performance MCUs, as shown in [High Performance MCU](#), use an external Flash Memory part and typically download the entire image to SRAM during boot time. These MCUs, depending on their processing and power needs based on application they are addressing, work across process nodes (for example, down to newer process nodes such as 3/5nm or custom nodes like 45nm), where flash technology is not available. Also, having a separate flash part is cheaper due to the flash only process node. Most common disadvantage of these MCUs is the cost due to large SRAM size (>= Application SW Image Size) and large boot/startup time.

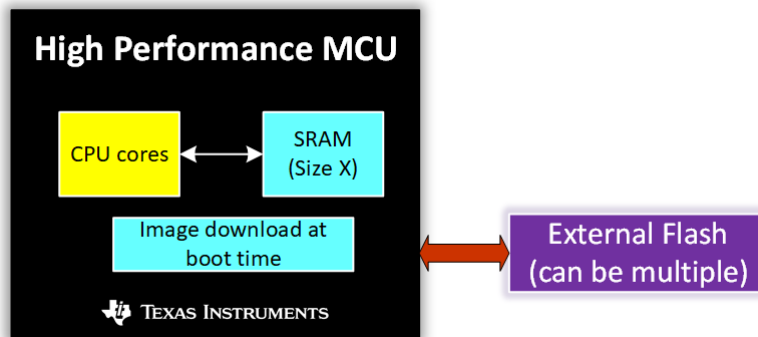


Figure 1-2. High Performance MCU

2 OptiFlash Technology

OptiFlash technology, which is TI's differentiated memory technology, enables high performance and low-cost solution by right mix of on-chip SRAM and external flash. An SoC with integrated OptiFlash technology, as shown in [Figure 2-1](#), aims to solve the limitations faced by traditional high-performance MCUs with external flash, by providing hybrid execution from internal SRAM and directly from external flash (also known as Execute in Place (XIP)), with a goal of reaching XIP performance from external flash to that of internal SRAM. The major cost advantage of this solution is that the internal SRAM can now be sized lower than the application image size, and the silicon die-cost without embedded flash/NVM is much cheaper due to reduced number of masks and cheaper equipment. This solution also provides scalability in terms of the size (in the order of MBytes) and parts on board based on customer need.

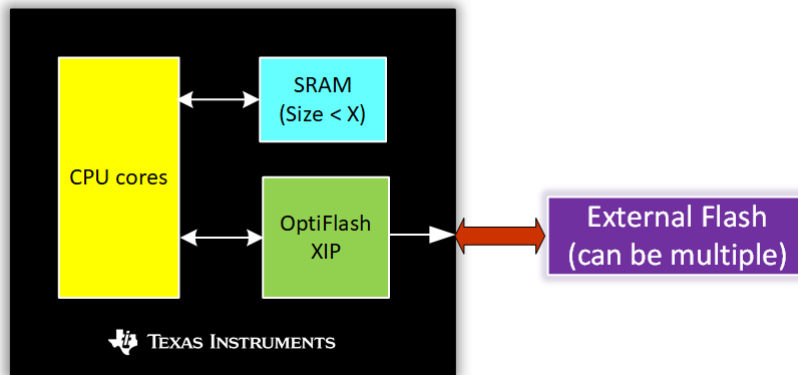


Figure 2-1. Proposed MCU With OptiFlash

Executing from external Flash brings a host of challenges that needs to be solved such as: Performance, Functional Safety, Security, Startup-time and Firmware upgrade. OptiFlash solution integrated in an MCU/SoC chip, as shown in [Figure 3-1](#), provides a set of innovative Hardware (HW) Accelerators and Software (SW) tooling, that aims to solve all the above challenges when executing from external Flash. OptiFlash can address up to 128MB of external flash while enabling the lowest possible system BOM cost.

3 OptiFlash Hardware Accelerators

There are two types of OptiFlash hardware accelerators, as seen in [OptiFlash Key Hardware Components Diagram](#):

- RL2_OF (Remote L2 OptiFlash) Accelerators
- FSS (Flash Subsystem) Accelerators

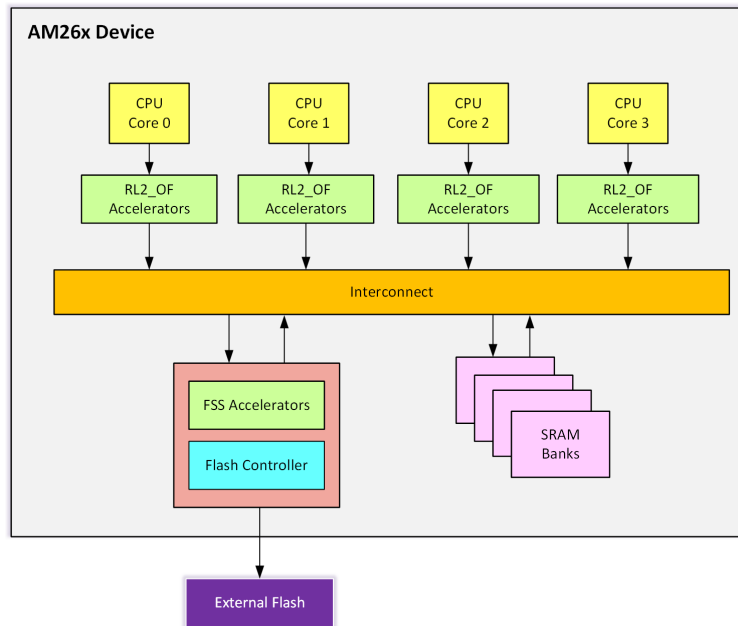


Figure 3-1. OptiFlash Key Hardware Components Diagram

3.1 RL2_OF Accelerators

RL2_OF Hardware Accelerator has three submodules that are close to CPU.

3.1.1 RL2-Flash Cache

OptiFlash supports a Remote L2 controller (RL2) for caching that is customized for flash and application characteristics for optimized system performance. It can reduce external Flash access by up to 65-95% based on application profile (RL2 size allocated in the application). It acts as a Level 2 cache controller as it provides additional caching, specific to Flash storage, beyond CPU Core's L1 caching. The cache is remote, that is, the actual cache memory can be part of any system memory. For example, on-chip memory (remote cache data storage memory), instead of a dedicated cache storage within the controller. User has flexibility to specify the size of cache based on target application needs.

3.1.2 FLC - Fast Local Copy (Image Download Acceleration)

OptiFlash supports a Fast Local Copy (FLC) engine for image download acceleration, during startup or run time, to enable code download along with CPU execution in parallel. It redirects the CPU access to Flash when the copy is in progress and redirecting the access to SRAM when the content is in valid in SRAM, as shown in Figure 3-2. This enables CPU to execute immediately without waiting for the copy to complete. The operation is transparent to CPU, and results in reduced startup time, in order to meet startup time goals similar to an embedded flash and also provide dynamic overlay for run time performance.

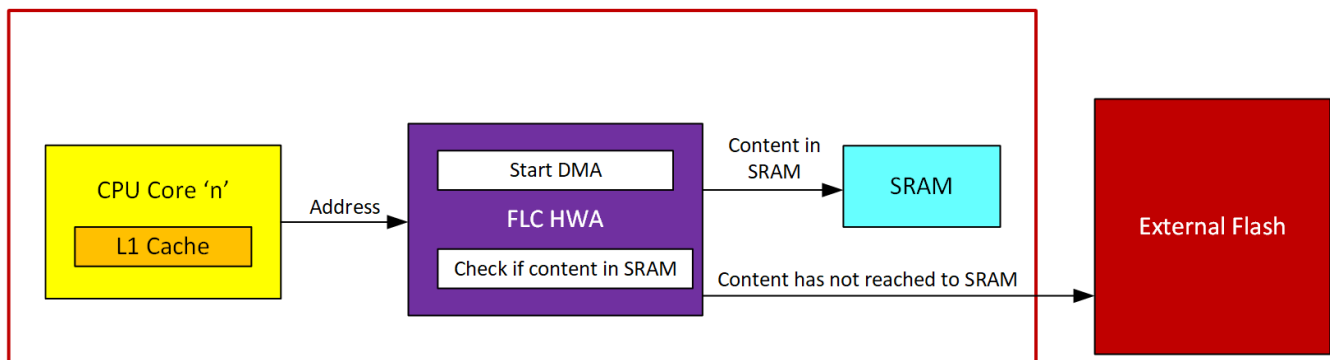


Figure 3-2. FLC - Image Download

3.1.3 Region-Based Address Translation (RAT)

Region based address translation (RAT) engine, placed in front of each CPU core, allows segments of the memory map to be relocated to a different address in the system. It is a dynamic address translator for common code and constant (placed in SRAM) instead of duplicated/redundant code/constant across multiple cores, that is downloaded to on-chip SRAM during boot-time.

3.2 FSS Accelerators

FSS Accelerator has three submodules that are close to Flash controller.

3.2.1 On-the-fly-Safety Engine

OptiFlash provides a Safety Engine with In-line ECC insertion during write and checking/correction during read to provide safety and reliability with external flash. It supports on-the-fly address translation to provide SW transparent view to account for additional storage of ECC data bytes.

3.2.2 On-the-fly-Security Engine

OptiFlash provides a security engine with in-line encryption/decryption/authentication (AES/GCM) on flash data to enable secure external flash use. It supports on-the-fly address translation to provide SW transparent view to account for additional storage of Message Authentication Code (MAC). MAC size is programmable

3.2.3 FOTA HW Engine

For emerging applications like ADAS, Automotive Gateway, Industry Automation, Firmware Over the Air (FOTA) updates are required to address multiple features, security vulnerabilities, bug fixes, and so forth. In order to meet the system cost, a single flash solution is required, which would typically require to pause the application execution till the update (firmware download) is completed. In past, these updates were scheduled during system start up (key-on) or system shutdown (key-off). To reduce overall down-time of system during update (unlike cell phone), the new requirement is to update new firmware/software image during concurrent system operation, that is, reading from external flash (execute in place (XIP)).

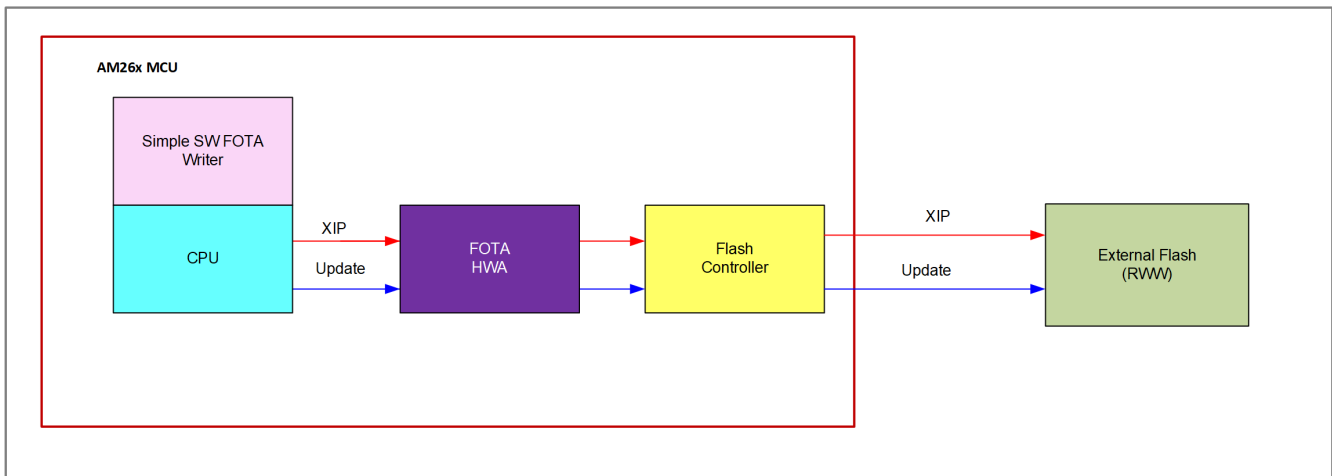


Figure 3-3. OptiFlash FOTA HW With RWW Flash

Typical FOTA solutions address this problem by performing read while write in software. But without any hardware support, it becomes complex as it requires complex synchronization across threads/CPU's, increasing the XIP downtime. With the OptiFlash FOTA Hardware Accelerator IP, as shown in [OptiFlash FOTA HW with RWW Flash](#), it is possible to further reduce the XIP downtime and be able to perform concurrent XIP read(s) while FOTA update happens in background, with zero software overheads on MCU. Primarily this is useful when using Read While Write (RWW) capable Flash memory with dual/multiple banks, which allows reads while write/erase is in progress (which can take >1ms to complete) in a different bank.

4 OptiFlash SW Tooling

OptiFlash provides a set of innovative SW Tooling (PC-based offline), as part of Arm® CLANG Compiler Enhancements.

4.1 Smart Placement

Performance degradation when code executes from external FLASH is about 2-3x compared to when code executes from internal RAM. The smart placement tool helps to minimize the performance degradation by “smart” placement of “critical” code (a function is marked critical if it is executed higher no. of times vs another function). It generates a “linker command file”, which places “critical” function in RAM in a fixed size RAM code buffer and execute the rest from external FLASH.

4.2 Smart Layout

Startup code is usually a code that is badly cached (it always executes once and it calls functions that are generally at different locations in memory) and hence, the effectiveness of pre-fetch hardware in CPU is less. The Smart Layout tool solves this by performing auto-identification of startup code by grouping frequently executed functions in execution order into an address region that can be programmed using Fast Local Copy (FLC) engine to download + concurrent CPU execution. This sort of memory placement compliments the instruction fetching pattern of CPU of a given type of code and helps to maximizes usage of pre-fetch hardware.

4.3 Optishare

Typically, in a high performance MCU, each CPU runs an independent application, but each application uses certain common software libraries and functions, for example, same RTOS, drivers, networking stack. This results in the same code or read-only (RO) data getting duplicated multiple times once for each core, and will soon start to run out of memory. The OptiShare tool aims to solve this by finding common code/data segments across multiple CPU images and convert it to a single shared copy, to avoid duplication of code/data and reduce the amount on chip SRAM. This tool is used along with the RAT engine placed in front of each CPU core, which performs address translation to map the common code to the shared address in the system.

4.4 Dynamic Overlay

OptiFlash FLC DMA engine can also be used to implement the overlay algorithm, to transfer critical code from on chip memory and Flash to TCM on demand basis. Along with ease of implementation, this can significantly improve the run-time program performance as overlay can happen in background without any CPU intervention.

5 Benchmarks and Performance Data

Application benchmarks have been run to measure the XIP performance when executing from external Flash when various OptiFlash components are enabled.

RL2 Cache Size (KB)	Performance Degradation (Flash vs OCRAM)
0	2.36
8	2.23
32	1.93
128	1.10

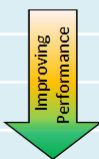


Figure 5-1. Benchmarks

The application used for benchmarking XIP Performance is an instruction intensive application that simulates an AUTOSAR cache miss rate. The cache miss has been calibrated with real networking application (around 3-4 million instructions miss/second). Performance degradation has been obtained by measuring the processing time when executing the application from external Flash vs On-Chip SRAM (OCSRAM). As can be seen from [Figure 5-1](#), after enabling RL2 Flash Cache, there is a significant improvement in XIP Performance seen with increasing cache size (up to 1.1x times than that of OCSRAM with 128KB cache).

Note that this is one example case study. The real application can have different performance improvement based on actual traffic pattern.

6 Usecases for OptiFlash Accelerators

The key performance considerations when working on a system with external flash are: XIP Performance, Image download time, safety, security, firmware update, and so forth. [Table 6-1](#) provides a list of OptiFlash HW Accelerators and SW Tools that are needed for a given system use case target. For improving XIP Performance, OptiFlash provides the Smart Placement Tool which would place critical code on On-Chip RAM, and the Flash Cache (RL2) HW Accelerator, which would reduce access to external Flash. For image download acceleration, OptiFlash provides the smart layout tool that is to be used along with the FLC engine to group frequently executed functions in execution order, and perform concurrent code download and execution. To perform run-time dynamic overlay, FLC engine along with the overlay manager tool is to be used. To reduce the amount of common code size in case of a multi-core application, the OptiShare tool converts common code and data to a single shared copy and RAT engine performs address translation to the shared memory location. To provide safety and security with external flash while performing XIP, the on-the-fly safety engine and on-the-fly security engine are present inside the OptiFlash Flash subsystem. To perform firmware over the air update, OptiFlash provides a FOTA HW engine, to enable image download to flash along with concurrent XIP execution, with minimum XIP downtime.

Table 6-1. Usecases for OptiFlash Accelerators

System Usecases	OptiFlash HW Accelerators	OptiFlash SW Tooling
XIP performance improvement	None	Smart placement tool
	Remote L2 (RL2) cache	
Image download acceleration	Fast local copy (FLC) engine	Smart layout tool
Dynamic overlay		Overlay manager
Multicore application-code size reduction	Region address translator (RAT)	Optishare tool
Safety over XIP	On-the-fly safety engine	None
Security over XIP	On-the-fly security engine	None
Over the air firmware update	FOTA HW engine	None ⁽¹⁾

(1) The firmware run on FOTA HW is provided as part of MCU+ SDK drivers for Safety/Security and OTA.

7 Getting Started With OptiFlash

Getting started with OptiFlash is quite easy. All OptiFlash SW features are provided as part of TI Arm Clang Compiler tool-chain to enable easy and seamless development by the user. Developing an application for an MCU with integrated OptiFlash Technology is quite similar to the standard application development flow, as shown in [Figure 7-1](#), with added options to enable OptiFlash tooling as part of code compilation and build, and modifying the SBL to configure the required OptiFlash HW accelerators during system startup, as shown in [Figure 7-2](#). Few tools, such as smart placement and smart layout, would require an initial test run to collect code coverage statistics that would be fed to the tool to generate a linker command file with optimal placement of code and data. Next step would be to configure various parameters of OptiFlash HW components as part of SBL system initialization. For example, when using the OptiShare tool in a multi-core application, there is a separate binary created by the tool for shared code and RO data, mapped to a common shared memory address space. Now, the SBL needs to be configured to load the shared code and RO data once into the OCSRAM, and set up the RAT engine for each CPU such that the shared code and RO data map from shared memory address space to CPU unique address space. Finally, in the last step, the SBL and application is downloaded to Flash and the system will boot after power ON.

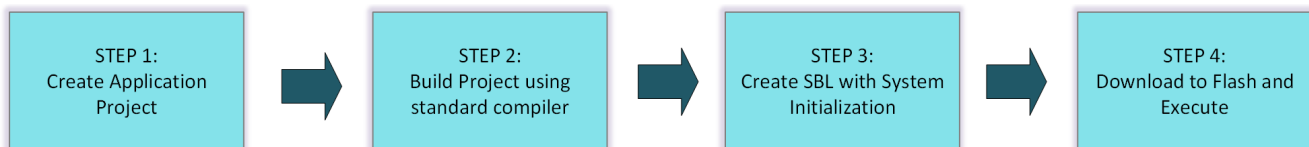


Figure 7-1. Standard Application Development Flow

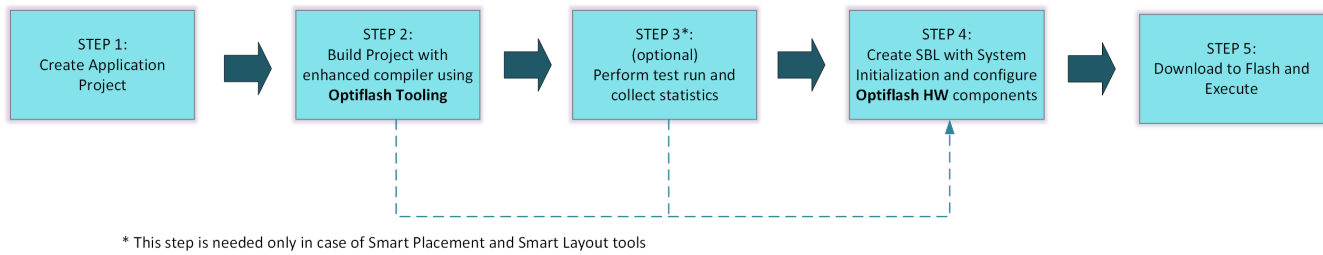


Figure 7-2. Enhanced Application Development Flow With OptiFlash

8 Conclusion

In the high-performance MCU world, due to increase in the complexity of emerging applications, the memory and performance requirements of these architectures are going high. Hence, there is an increasing demand for using external flash in these Microcontrollers. However, using external flash brings a set of challenges in the system that need to be solved. TI's OptiFlash memory technology solution provides a set of hardware accelerators and software tooling, which are easy to deploy, solving key concerns of using external flash. It is a low-cost solution that aims to enhance the system performance when executing from external Flash, and also provides scalability in terms of Flash size based on system need. This allows TI MCUs that take advantage of deep submicron technology nodes to deliver the required CPU performance. OptiFlash solution also scales with devices supporting Embedded Flash to further increase the overall Flash size using external flash.

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](https://www.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
Copyright © 2024, Texas Instruments Incorporated