TI Designs
Interfacing Nonvolatile Storage Memory With Application Download and Execution on SDRAM for High-Performance Microcontrollers Design Guide

TI Designs

TI Designs provide the foundation that you need including methodology, testing and design files to quickly evaluate and customize the system. TI Designs help *you* accelerate your time to market.

Design Resources

TIDM-TM4C129SDRAMNVM	Tool Folder Containing Design Files
EK-TM4C1294XL	Tool Folder
TM4C1294NCPDT	Product Folder
TM4C123GH6PM	Product Folder
TPD4S012	Product Folder
TPS2052B	Product Folder
TPS62177	Product Folder
TPS73733-Q1	Product Folder

TI E2E™ Community

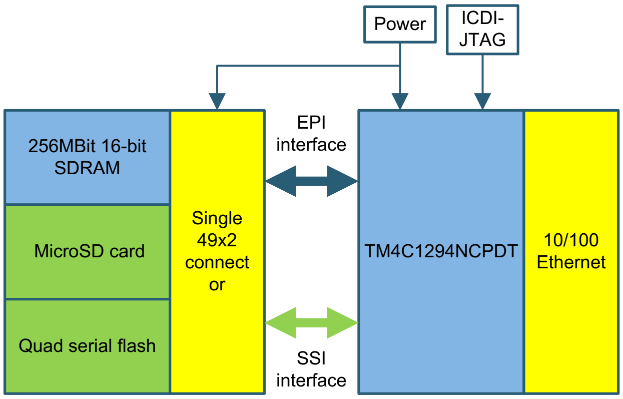
ASK Our E2E Experts
WEBENCH® Calculator Tools

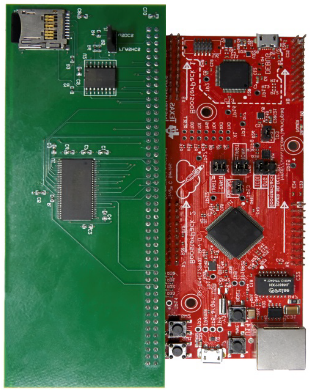
Design Features

- Extends the Useable Memory Space to 512-Mb, 16-bit SDRAM with 60-MHz External Peripheral Interface (EPI) for High-Memory Throughput and Footprint Applications
- Designed for EK-TM4C1294XL Connected LaunchPad ™(Formerly Tiva MCU™)
- Implements Serial Interface Bootloaders for SD Card or Quad Serial Flash Nonvolatile Memory
- Offers Additional Support for Quad-Serial Flash Mode Command for Custom Implementation of Bare Metal External Memory Loggers
- Offers Source Code That Contains Project Examples for Code Composer Studio™

Featured Applications

- Interactive Human-Machine Interfaces
- Industrial Automation
- Internet of Things (IoT) Solutions
- Test and Measurement





An IMPORTANT NOTICE at the end of this TI reference design addresses authorized use, intellectual property matters and other important disclaimers and information.

LaunchPad, Tiva MCU, Code Composer Studio, TivaWare are trademarks of Texas Instruments.
All other trademarks are the property of their respective owners.

1 System Description

The memory space of the TM4C129x family can be extended when the internal memory is insufficient. This TI Design describes the hardware-interfacing requirement and example software for the TM4C129x microcontrollers from TI. The external peripheral interface (EPI) can be used to extend the executable memory region to 16-bit, 512Mb of SDRAM. The QSSI interface at 60 MHz can extend storage of NVM code. This capability lets applications use microSD cards or QSSI flash memory greater than 512Mb. The design files include schematics, BOM, layer plot, Altium files, Gerber files, a reference example code for an easy-to-use SDRAM, an SD card boot, and an QSSI boot with a TM4C1294NCPDT Connected LaunchPad.

1.1 TM4C1294NCPDT Microcontroller

The TM4C1294NCPDT is a 120-MHz high-performance microcontroller with 1MB of on-chip flash and 256KB of on-chip SRAM. The device features an integrated Ethernet MAC+PHY for connected applications. The device has high-bandwidth interfaces like memory controller and a high-speed USB 2.0 digital interface. Integrating low- to mid-speed serials, up to 4 MSPS, 12-bit ADC, and motion control peripherals, this TI Design is a unique solution for a variety of applications, from industrial communication equipment to Smart Energy and Smart Grid applications.

For a high-level block diagram of the TM4C1294NCPDT Microcontroller, see Figure 1.

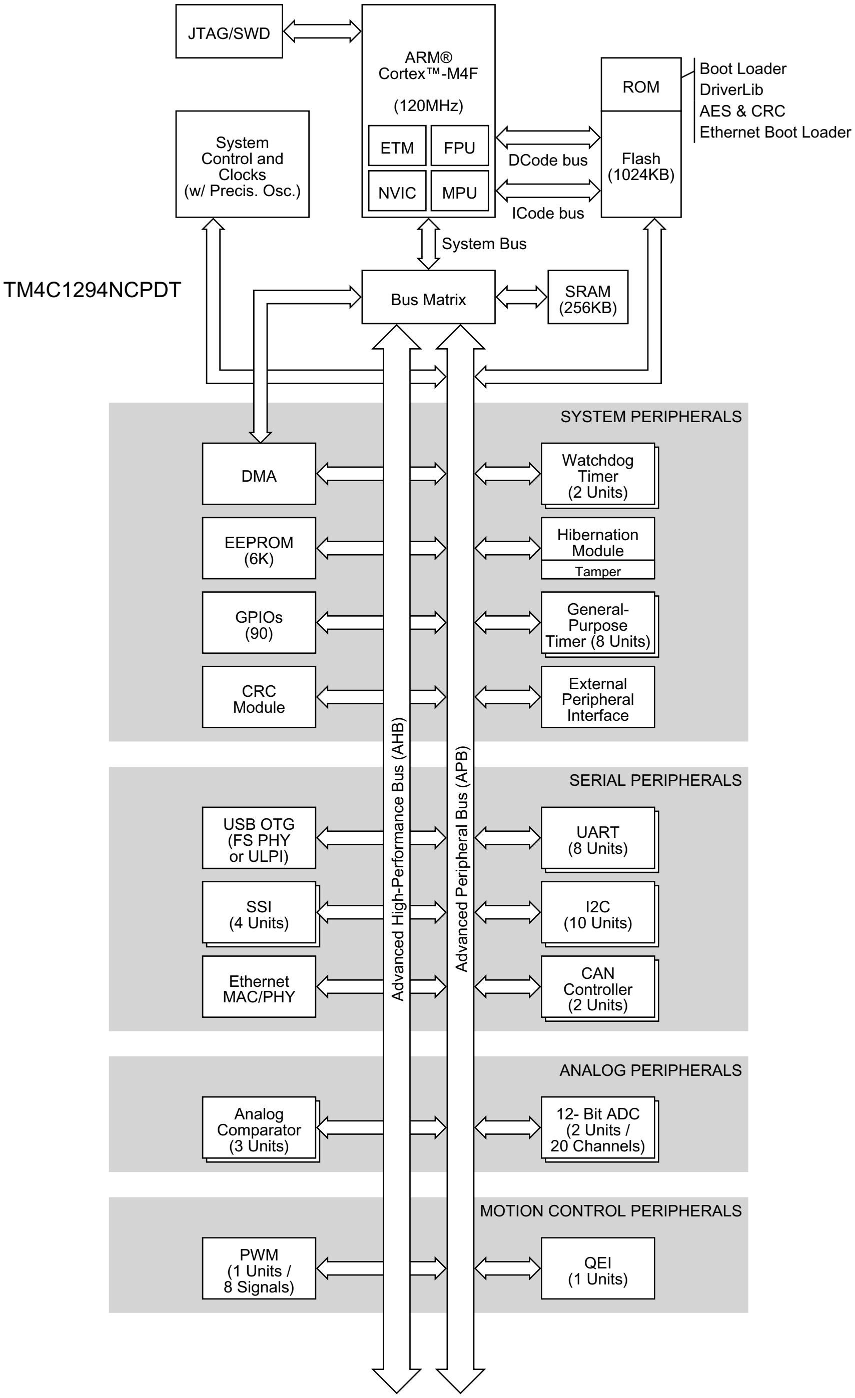
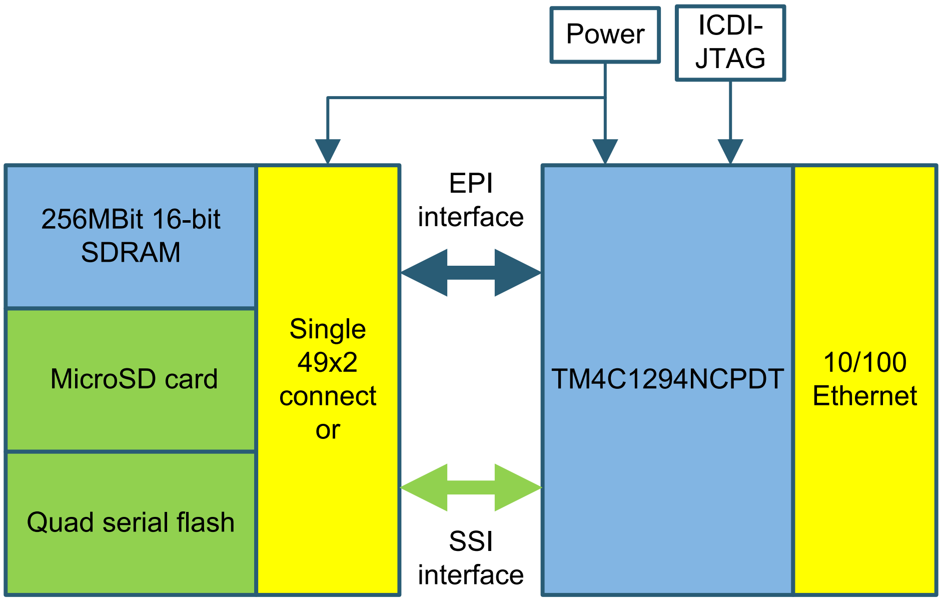


Figure 1. TM4C1294NCPDT Microcontroller High-Level Block Diagram

2 Block Diagram



SDRAM-NVM Memory Extender Block Diagram

3 Getting Started Hardware

Interfacing the SDRAM-NVM memory to the TM4C1294NCPDT device on an EK-TM4C1294XL Connected LaunchPad requires a daughterboard that can connect to the breadboard connector X11.

3.1 The SDRAM-NVM Daughtercard

The SDRAM-NVM daughtercard interfaces to the EK-TM4C1294XL Connected LaunchPad using the 49x2 breadboard connectors. The daughtercard has one jumper (J1) that can be used to select between the microSD card or the QSSI serial flash. The microSD card uses the Legacy SPI mode of the QSSI module to interface with the microcontroller while the QSSI serial flash uses the advanced mode of the QSSI module to interface with the microcontroller. For an overview of the connector mounting for the daughtercard, see Figure 2.

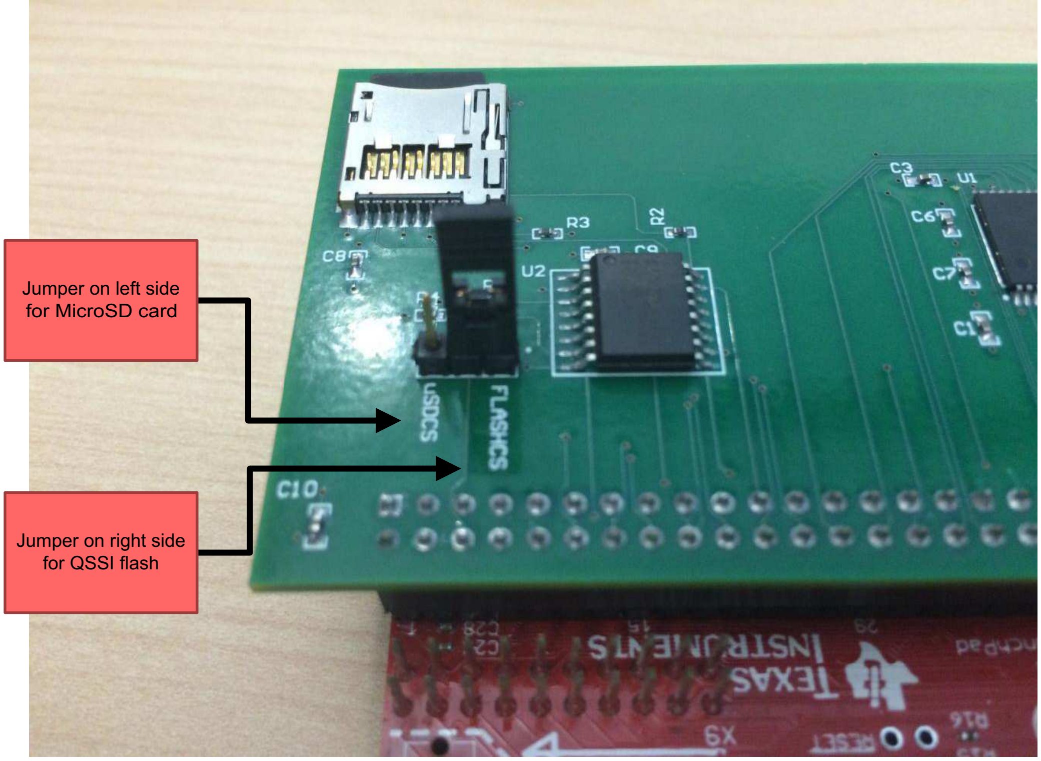

Jumper on left side for MicroSD card
Jumper on right side for QSSI flash
USDCS
FLASHCS
R3
R2
U2
C8
C9
C10
C3
U1
C6
C7
C1

Figure 2. SDRAM-NVM Connector Mounting

4 Getting Started Software

The software for this reference design comes with three codes that you can import in Code Composer Studio and use as a starting point for your application.

4.1 *MicroSD Card Boot With SDRAM Code Execution*

The *MicroSD Card Boot With SDRAM Code* uses the internal flash of the TM4C1294NCPDT to hold the FAT file system and bootloader. The bootloader configures the QSSI modules to run the FAT file system and EPI to interface to a 512-Mb SDRAM at interface frequency of 60 MHz. You can have multiple images on an microSD card configured during compile time to execute from an EPI Address Space of 0x6000 0000. You can select one of the image files that the bootloader copies to the EPI peripheral-connected SDRAM. After the image is copied, the Cortex M4 disables the interrupts, updates the NVIC_VTABLE register to map to the external address map, and jumps to the external address space of 0x6000 0000. All subsequent code execution occurs in the external address space until the next board reset. You must use a PC to copy the images to the microSD Card.

4.2 *QSSI Serial Flash Boot With SDRAM Code Execution*

The *QSSI Serial Flash Boot With SDRAM Code* uses the internal flash of the TM4C1294NCPDT to hold a custom bootloader. The bootloader configures the QSSI modules to read a QSSI flash memory and the EPI to interface to a 512-Mb SDRAM at an interface frequency of 60 MHz for executing code.

The lowest sector (Sector-0) of QSSI flash memory holds a table indicating the start address, size, and validity of an image. The bootloader updates this location when you download the binary file to the external QSSI flash memory through UART0. Figure 3 provides an overview of the structure of the information held in Sector-0 pertaining to an actual application image.

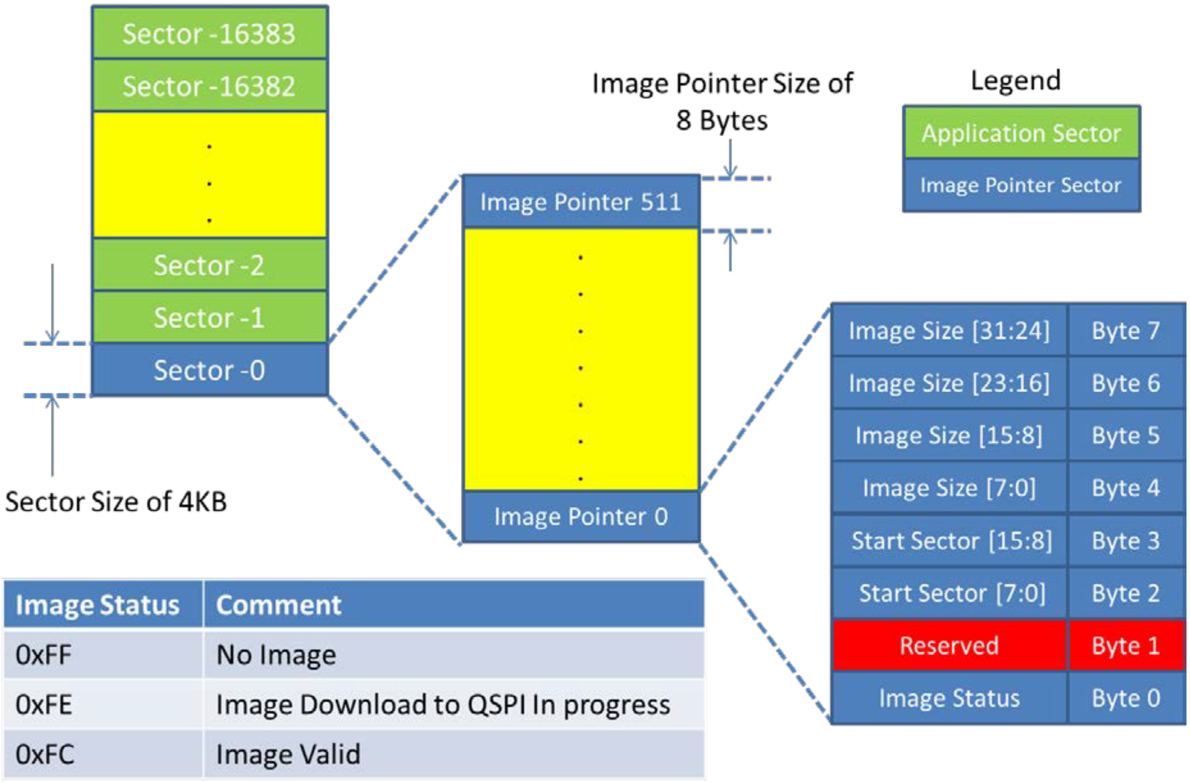


Image Size [31:24]	Byte 7
Image Size [23:16]	Byte 6
Image Size [15:8]	Byte 5
Image Size [7:0]	Byte 4
Start Sector [15:8]	Byte 3
Start Sector [7:0]	Byte 2
Reserved	Byte 1
Image Status	Byte 0

Image Status	Comment
0xFF	No Image
0xFE	Image Download to QSPI In progress
0xFC	Image Valid

Figure 3. Sector-0 Information Block Structure

You can download an image to the QSSI flash memory that has been configured at compile time to execute from the EPI address space 0x6000 0000. The image is downloaded from a PC or other controller. To load a new image to the QSSI flash, press USR_SW1 when powering up or resetting the LaunchPad. This causes the bootloader to enter download mode. If the USR_SW1 is not pressed, the bootloader will check Sector-0 for a valid image pointer and execute the last image available on QSSI flash. If no valid image pointer is found, the bootloader will enter download mode and wait for a new image. During the execution phase, the bootloader copies the image to the SDRAM memory connected to the EPI peripheral. When the image finishes copying, the Cortex M4 disables the interrupts, updates the NVIC_VTABLE register to map to the external address map, and jumps to the external address space 0x6000 0000. All subsequent code execution occurs in the external address space until the next board reset. For an overview of the flow chart that shows how the code operates, see Figure 4.

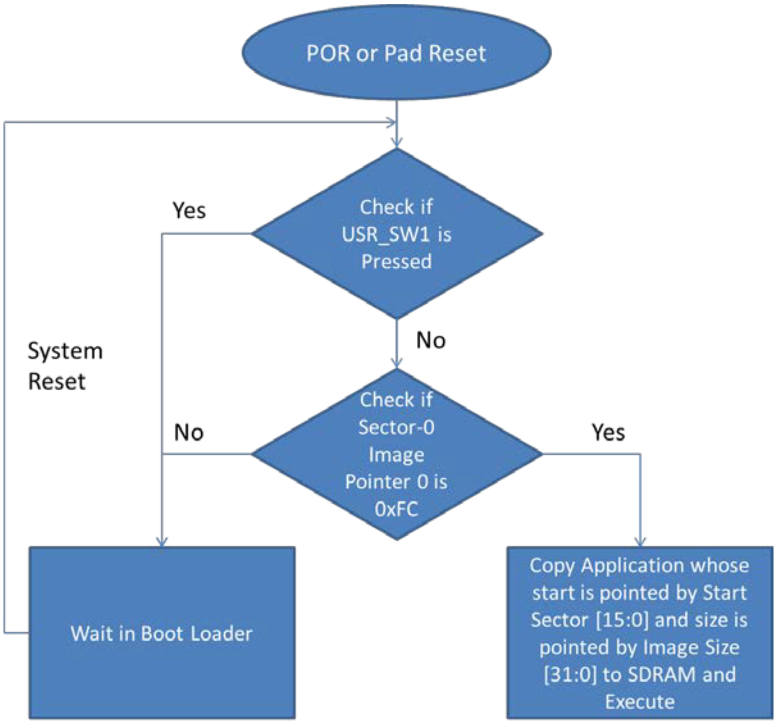

POR or Pad Reset
Check if USR_SW1 is Pressed
Yes
No
System Reset
Check if Sector-0 Image Pointer 0 is 0xFC
No
Yes
Wait in Boot Loader
Copy Application whose start is pointed by Start Sector [15:0] and size is pointed by Image Size [31:0] to SDRAM and Execute

Figure 4. Example Code Program Flow

4.3 QSSI Bare Metal Code

The QSSI bare metal code configures the QSSI module of the TM4C1294NCPDT to perform advanced and quad mode programing for write operations and advanced, bi, and quad mode read for readback operations. For this example, the QSSI serial interface operates at 60 MHz, which demonstrates the maximum achievable throughput.

5 Test Setup

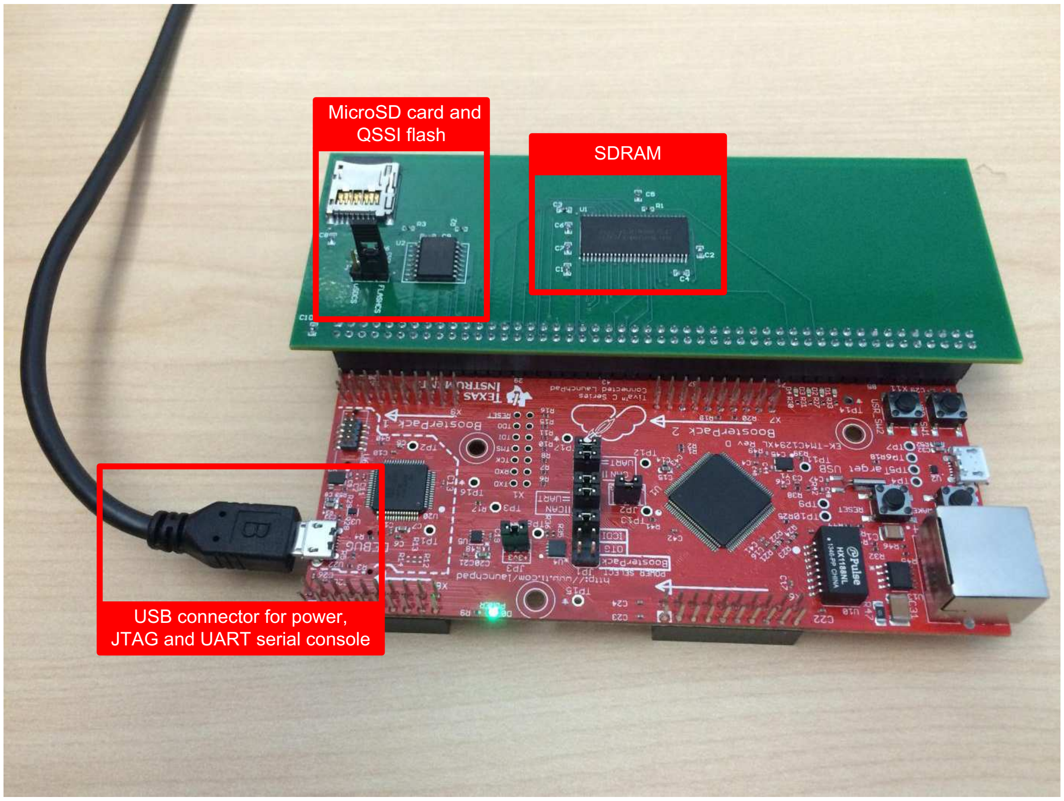
The test setup is as follows:

1. Import the test project into CCS.
2. Build and Compile the project.
3. Download the executable to the EK-TM4C1294XL Connected LaunchPad.
4. Execute the test code on the target.

During execution, the test code will first erase a 4-KB sector and check to ensure the erase was successful. If the erase was successful, the test code will then perform a program and read operation on the 4-KB sector. After performing the read operation, the test concludes with an erase and erase confirmation of the 4-KB test sector.

5.1 Hardware Setup

For an overview of the hardware setup, see Figure 5. The USB cable on the left side of the EK-TM4C129XL Connected LaunchPad provides power, connects to JTAG, and connects the UART for communication between a PC terminal window application such as Putty or Tera Term and the hardware setup. The SDRAM-NVM Memory Extender uses header X11 to connect to the LaunchPad.

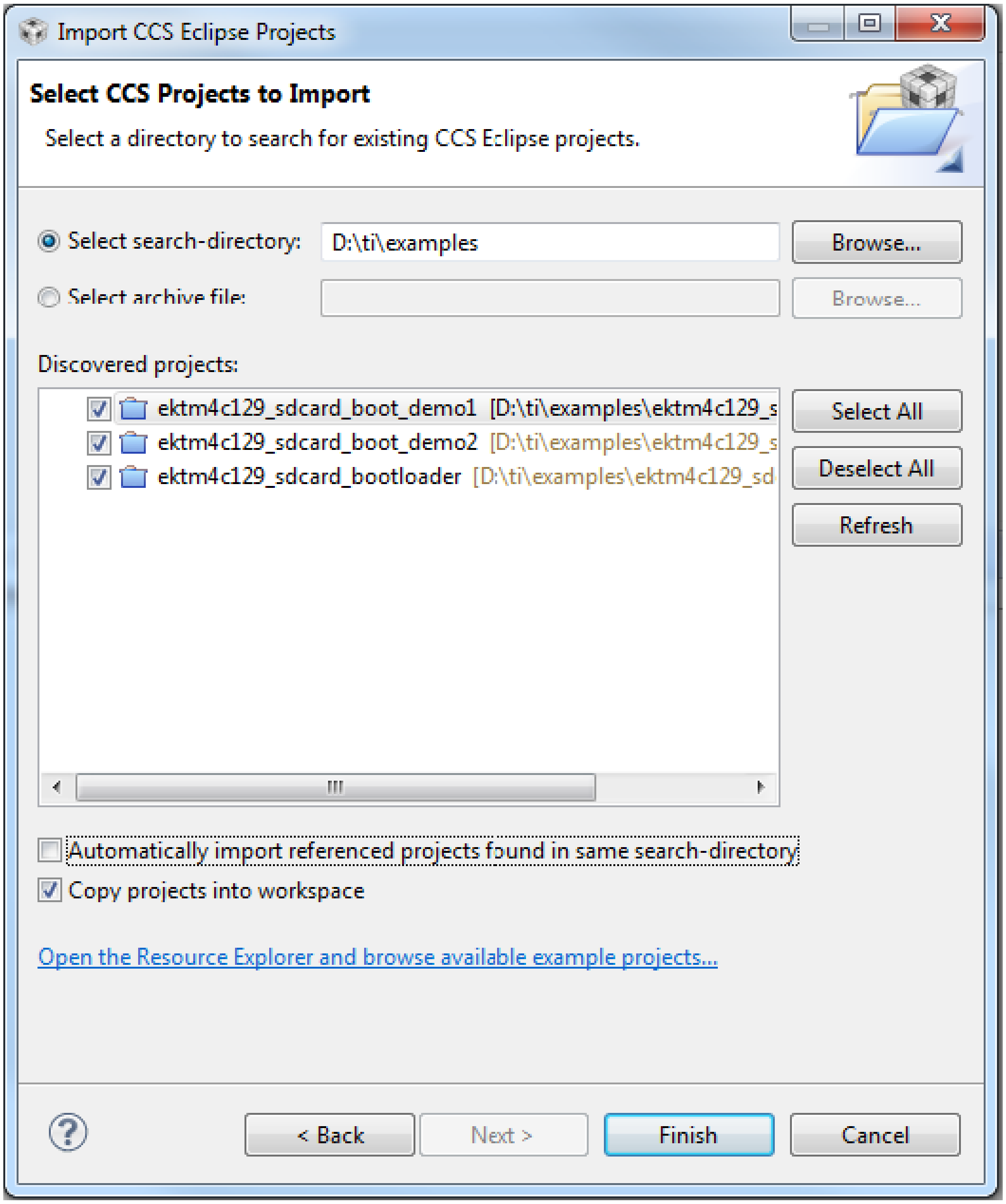

MicroSD card and QSSI flash
SDRAM
USB connector for power, JTAG and UART serial console

Download and install a serial console application (for example, PuTTY, TeraTerm, and so forth), Code Composer Studio v6.0.1, and TivaWare™ for C Series v2.1.0-12573 or later to use this example.

Figure 5. Full Test Assembly

5.2 Software Setup (microSD Card Boot With SDRAM Code Execution)

1. Download the example software package from TIDM-TM4C129SDRAMNVM. Unzip the software package.
2. Launch Code Composer Studio v6.0.1 or later→ Click Import→ Click CCS Projects→ Click Next. Browse to the directory with the software examples. Select "ektm4c129_sdcard_bootloader", "ektm4c129_sdcard_boot_demo1", and "ektm4c129_sdcard_boot_demo2". Click Finish.



Import CCS Eclipse Projects

Select CCS Projects to Import

Select a directory to search for existing CCS Eclipse projects.

● Select search-directory: D:\ti\examples

○ Select archive file:

Discovered projects:

☑ ektm4c129_sdcard_boot_demo1 [D:\ti\examples\ektm4c129_s

☑ ektm4c129_sdcard_boot_demo2 [D:\ti\examples\ektm4c129_s

☑ ektm4c129_sdcard_bootloader [D:\ti\examples\ektm4c129_sd

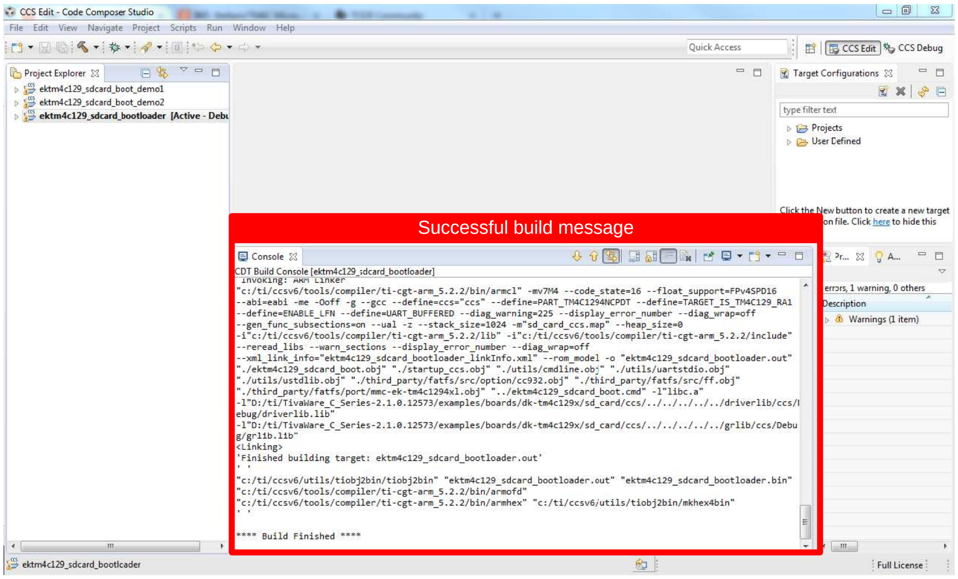
□ Automatically import referenced projects found in same search-directory

☑ Copy projects into workspace

Open the Resource Explorer and browse available example projects...

Figure 6. SDCARD Boot Project Import

3. Build each project. To build a project, right-click on a project. Click Rebuild Project. Ensure the projects compile free of errors.


CCS Edit - Code Composer Studio
File Edit View Navigate Project Scripts Run Window Help
Quick Access
CCS Edit
CCS Debug
Project Explorer
ektm4c129_sdcard_boot_demo1
ektm4c129_sdcard_boot_demo2
ektm4c129_sdcard_bootloader [Active - Debu
Target Configurations
type filter text
Projects
User Defined
Click the New button to create a new target
on file. Click here to hide this
Successful build message
Console
errors, 1 warning, 0 others
Description
Warnings (1 item)
ektm4c129_sdcard_bootloader
Full License

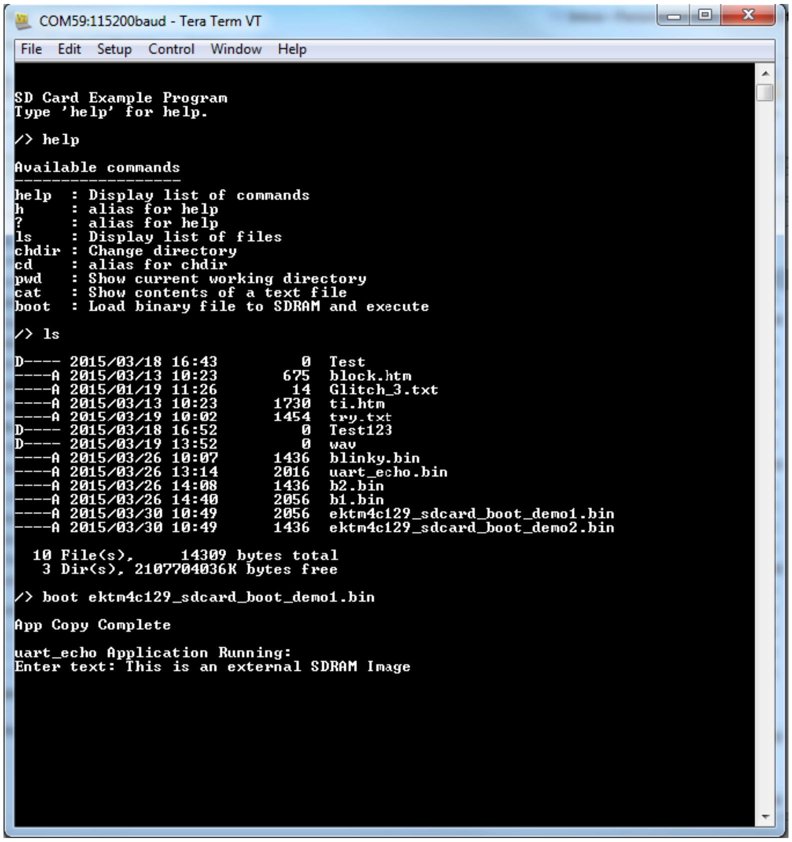
CDT Build Console [ektm4c129_sdcard_bootloader]

```
 Invoking: ARM Linker
"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/bin/armcl" -mv7M4 --code_state=16 --float_support=FPv4SPD16
--abi=eabi -me -Ooff -g --gcc --define=ccs="ccs" --define=PART_TM4C1294NCPDT --define=TARGET_IS_TM4C129_RA1
--define=ENABLE_LFN --define=UART_BUFFERED --diag_warning=225 --display_error_number --diag_wrap=off
--gen_func_subsections=on --ual -z --stack_size=1024 -m"sd_card_ccs.map" --heap_size=0
-i"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/lib" -i"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/include"
--reread_libs --warn_sections --display_error_number --diag_wrap=off
--xml_link_info="ektm4c129_sdcard_bootloader_linkInfo.xml" --rom_model -o "ektm4c129_sdcard_bootloader.out"
"./ektm4c129_sdcard_boot.obj" "./startup_ccs.obj" "./utils/cmdline.obj" "./utils/uartstdio.obj"
"./utils/ustdlib.obj" "./third_party/fatfs/src/option/cc932.obj" "./third_party/fatfs/src/ff.obj"
"./third_party/fatfs/port/mmc-ek-tm4c1294xl.obj" "../ektm4c129_sdcard_boot.cmd" -l"libc.a"
-l"D:/ti/TivaWare_C_Series-2.1.0.12573/examples/boards/dk-tm4c129x/sd_card/ccs/../../../../../driverlib/ccs/D
ebug/driverlib.lib"
-l"D:/ti/TivaWare_C_Series-2.1.0.12573/examples/boards/dk-tm4c129x/sd_card/ccs/../../../../../grlib/ccs/Debu
g/grlib.lib"
<Linking>
'Finished building target: ektm4c129_sdcard_bootloader.out'
' '
"c:/ti/ccsv6/utils/tiobj2bin/tiobj2bin" "ektm4c129_sdcard_bootloader.out" "ektm4c129_sdcard_bootloader.bin"
"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/bin/armofd"
"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/bin/armhex" "c:/ti/ccsv6/utils/tiobj2bin/mkhex4bin"
' '

**** Build Finished ****
```

Figure 7. Compiling The SDCARD Boot Software

4. Copy the bin files from "ektm4c129_sdcard_boot_demo1" and "ektm4c129_sdcard_boot_demo2" to a microSD card connected to a PC. Insert the microSD card with the copied files into the slot on the SDRAM-NVM Memory Extender. Position the J1 jumper to the microSDCS side to assert the microSD chip select.

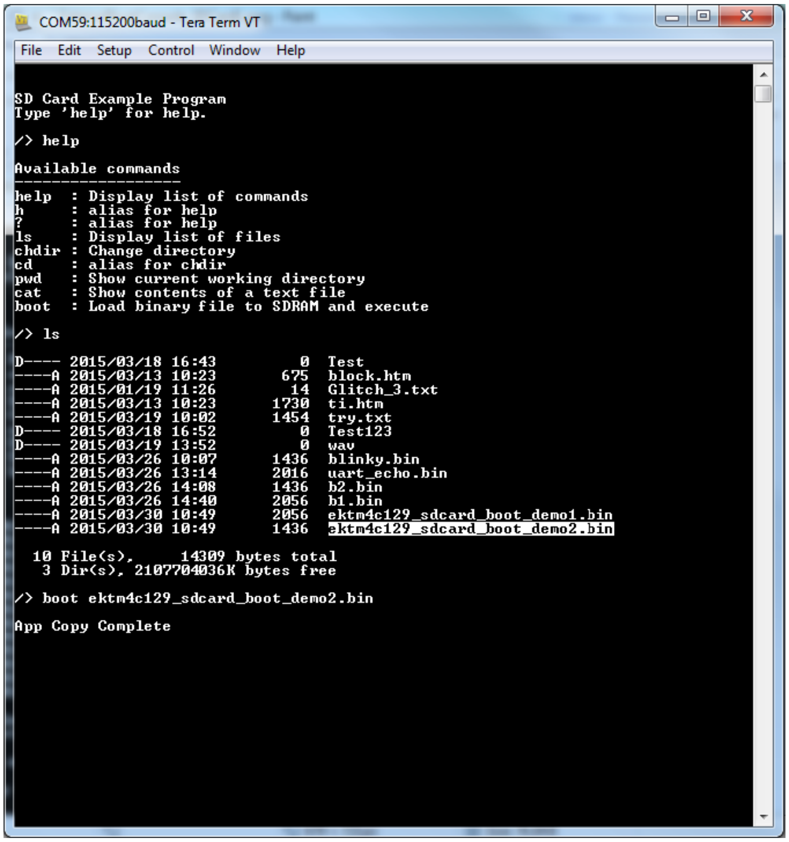
5. Press Debug to run the main bootloader, "ektm4c129_sdcard_bootloader", in the TM4C1294NCPDT flash. Press Play when the code loads. When you will see the prompt for the microSD card on a serial console, type "help" to see the options. Type "ls" to see the list of files. To select a binary image, type "boot ektm4c129_sdcard_boot_demo1". The uart_echo application will copy and execute from the SDRAM.



```
SD Card Example Program
Type 'help' for help.

/> help

Available commands
------------------
help  : Display list of commands
h     : alias for help
?     : alias for help
ls    : Display list of files
chdir : Change directory
cd    : alias for chdir
pwd   : Show current working directory
cat   : Show contents of a text file
boot  : Load binary file to SDRAM and execute

/> ls

D---- 2015/03/18 16:43         0  Test
----A 2015/03/13 10:23       675  block.htm
----A 2015/01/19 11:26        14  Glitch_3.txt
----A 2015/03/13 10:23      1730  ti.htm
----A 2015/03/19 10:02      1454  try.txt
D---- 2015/03/18 16:52         0  Test123
D---- 2015/03/19 13:52         0  wav
----A 2015/03/26 10:07      1436  blinky.bin
----A 2015/03/26 13:14      2016  uart_echo.bin
----A 2015/03/26 14:08      1436  b2.bin
----A 2015/03/26 14:40      2056  b1.bin
----A 2015/03/30 10:49      2056  ektm4c129_sdcard_boot_demo1.bin
----A 2015/03/30 10:49      1436  ektm4c129_sdcard_boot_demo2.bin

  10 File(s),     14309 bytes total
   3 Dir(s), 2107704036K bytes free

/> boot ektm4c129_sdcard_boot_demo1.bin

App Copy Complete

uart_echo Application Running:
Enter text: This is an external SDRAM Image
```

Figure 8. Serial Console Output For ektm4c129_sdcard_boot_demo1.bin

Type "boot ektm4c129_sdcard_boot_demo2" to copy and execute the D2 LED blinky application from the SDRAM.



COM59:115200baud - Tera Term VT
File Edit Setup Control Window Help

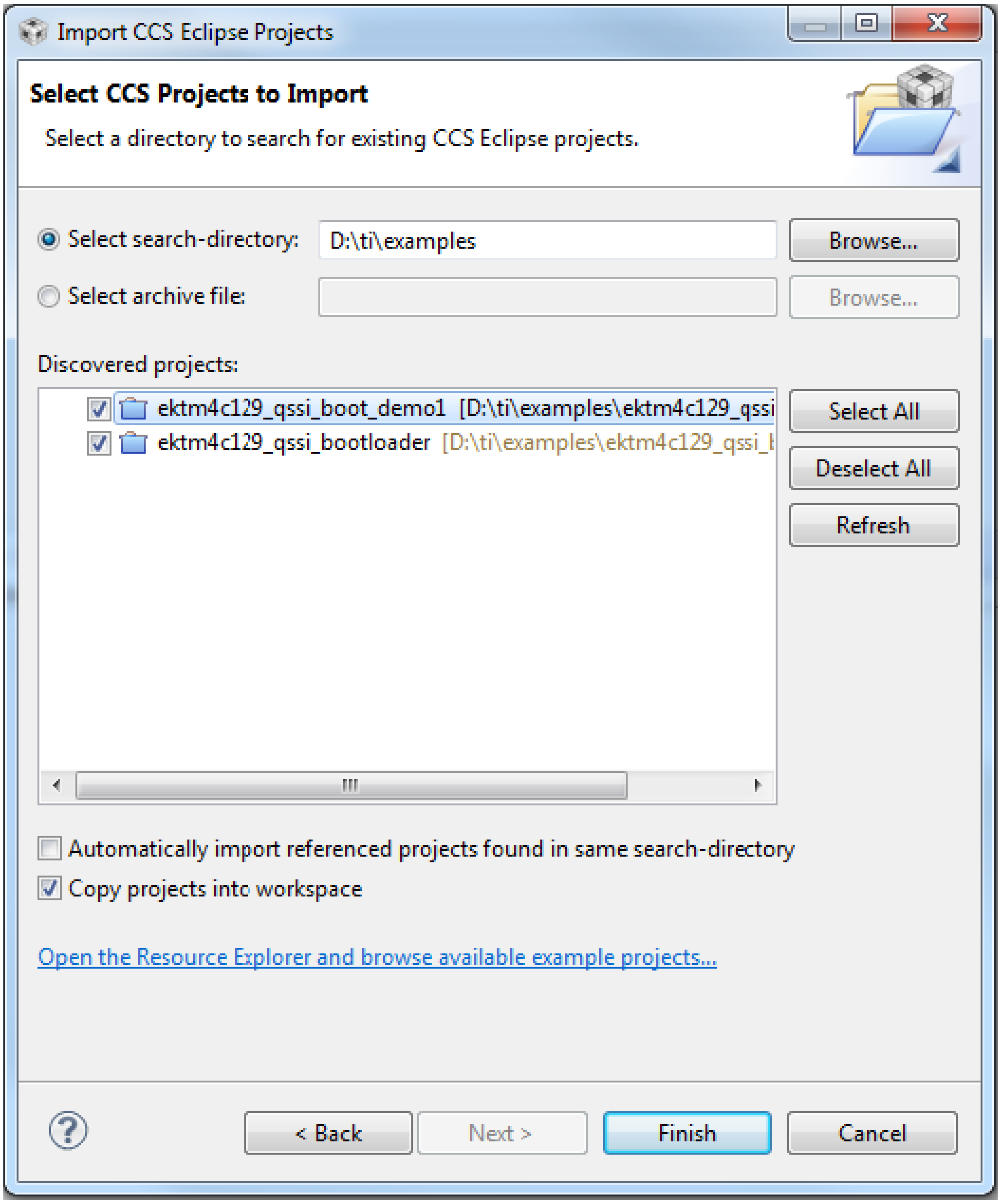
```
SD Card Example Program
Type 'help' for help.

/> help

Available commands
------------------
help  : Display list of commands
h     : alias for help
?     : alias for help
ls    : Display list of files
chdir : Change directory
cd    : alias for chdir
pwd   : Show current working directory
cat   : Show contents of a text file
boot  : Load binary file to SDRAM and execute

/> ls

D---- 2015/03/18 16:43         0  Test
----A 2015/03/13 10:23       675  block.htm
----A 2015/01/19 11:26        14  Glitch_3.txt
----A 2015/03/13 10:23      1730  ti.htm
----A 2015/03/19 10:02      1454  try.txt
D---- 2015/03/18 16:52         0  Test123
D---- 2015/03/19 13:52         0  wav
----A 2015/03/26 10:07      1436  blinky.bin
----A 2015/03/26 13:14      2016  uart_echo.bin
----A 2015/03/26 14:08      1436  b2.bin
----A 2015/03/26 14:40      2056  b1.bin
----A 2015/03/30 10:49      2056  ektm4c129_sdcard_boot_demo1.bin
----A 2015/03/30 10:49      1436  ektm4c129_sdcard_boot_demo2.bin

  10 File(s),      14309 bytes total
   3 Dir(s), 2107704036K bytes free

/> boot ektm4c129_sdcard_boot_demo2.bin

App Copy Complete
```

Figure 9. Serial Console Output For ektm4c129_sdcard_boot_demo2.bin

6. To restart the microSD card boot, press reset. The current application does not jump to the microSD card prompt.

5.3 Software Setup (QSSI Serial Flash Boot with SDRAM Code Execution)

1. Download the example software package from TIDM-TM4C129SDRAMNVM. Unzip the software package.
2. Launch Code Composer Studio v6.0.1 or later → Click Import→ Click CCS Projects→ Click Next. Browse to the directory with the software examples. Select "ektm4c129_qssi_bootloader", "ektm4c129_qssi_boot_demo1", and “ektm4c129_sdcard_boot_demo2”. Click Finish.



Import CCS Eclipse Projects

Select CCS Projects to Import

Select a directory to search for existing CCS Eclipse projects.

● Select search-directory: D:\ti\examples

○ Select archive file:

Discovered projects:

☑ ektm4c129_qssi_boot_demo1 [D:\ti\examples\ektm4c129_qssi

☑ ektm4c129_qssi_bootloader [D:\ti\examples\ektm4c129_qssi_l

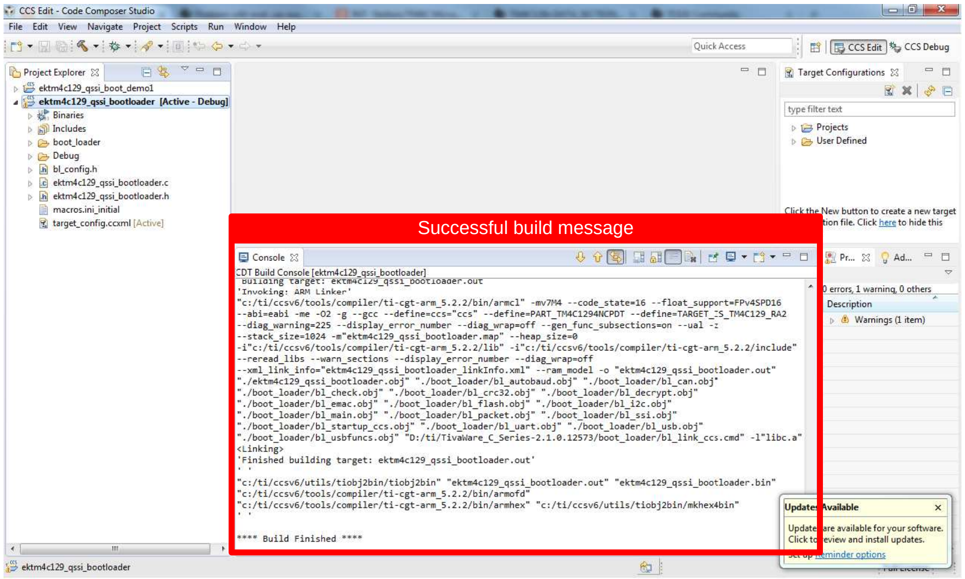
□ Automatically import referenced projects found in same search-directory

☑ Copy projects into workspace

Open the Resource Explorer and browse available example projects...

Figure 10. QSSI Example Project Import

3. Build each project. To build a project, right-click on a project. Click Rebuild Project. Ensure the projects compile free of errors.



CCS Edit - Code Composer Studio
File Edit View Navigate Project Scripts Run Window Help
Quick Access
CCS Edit
CCS Debug
Project Explorer
ektm4c129_qssi_boot_demo1
ektm4c129_qssi_bootloader [Active - Debug]
Binaries
Includes
boot_loader
Debug
bl_config.h
ektm4c129_qssi_bootloader.c
ektm4c129_qssi_bootloader.h
macros.ini_initial
target_config.ccxml [Active]
Target Configurations
type filter text
Projects
User Defined
Console
0 errors, 1 warning, 0 others
Description
Warnings (1 item)
Updates Available
ektm4c129_qssi_bootloader

Successful build message

Click the New button to create a new target ...tion file. Click here to hide this

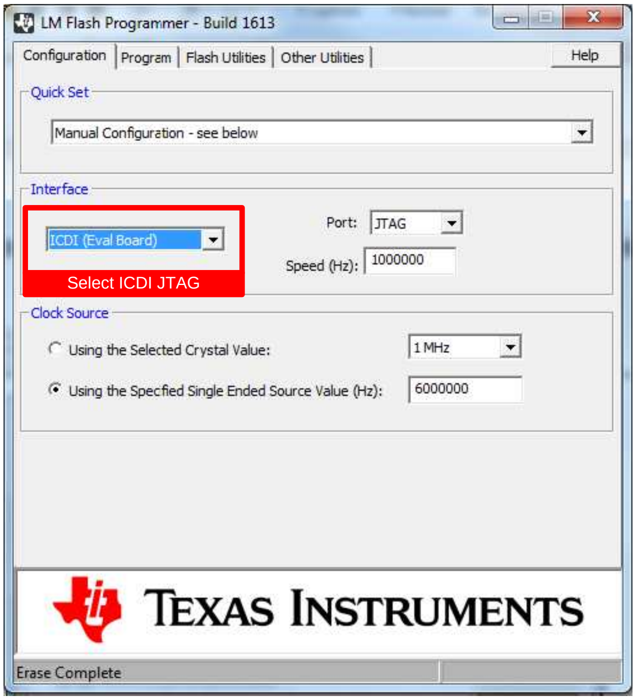
Updates are available for your software. Click to review and install updates.

Set up Reminder options

```
CDT Build Console [ektm4c129_qssi_bootloader]
 Building target: ektm4c129_qssi_bootloader.out
'Invoking: ARM Linker'
"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/bin/armcl" -mv7M4 --code_state=16 --float_support=FPv4SPD16
--abi=eabi -me -O2 -g --gcc --define=ccs="ccs" --define=PART_TM4C1294NCPDT --define=TARGET_IS_TM4C129_RA2
--diag_warning=225 --display_error_number --diag_wrap=off --gen_func_subsections=on --ual -z
--stack_size=1024 -m"ektm4c129_qssi_bootloader.map" --heap_size=0
-i"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/lib" -i"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/include"
--reread_libs --warn_sections --display_error_number --diag_wrap=off
--xml_link_info="ektm4c129_qssi_bootloader_linkInfo.xml" --ram_model -o "ektm4c129_qssi_bootloader.out"
"./ektm4c129_qssi_bootloader.obj" "./boot_loader/bl_autobaud.obj" "./boot_loader/bl_can.obj"
"./boot_loader/bl_check.obj" "./boot_loader/bl_crc32.obj" "./boot_loader/bl_decrypt.obj"
"./boot_loader/bl_emac.obj" "./boot_loader/bl_flash.obj" "./boot_loader/bl_i2c.obj"
"./boot_loader/bl_main.obj" "./boot_loader/bl_packet.obj" "./boot_loader/bl_ssi.obj"
"./boot_loader/bl_startup_ccs.obj" "./boot_loader/bl_uart.obj" "./boot_loader/bl_usb.obj"
"./boot_loader/bl_usbfuncs.obj" "D:/ti/TivaWare_C_Series-2.1.0.12573/boot_loader/bl_link_ccs.cmd" -l"libc.a"
<Linking>
'Finished building target: ektm4c129_qssi_bootloader.out'
' '
"c:/ti/ccsv6/utils/tiobj2bin/tiobj2bin" "ektm4c129_qssi_bootloader.out" "ektm4c129_qssi_bootloader.bin"
"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/bin/armofd"
"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/bin/armhex" "c:/ti/ccsv6/utils/tiobj2bin/mkhex4bin"
' '

**** Build Finished ****
```

Figure 11. Compiling the QSSI Software Example

4. Ensure that the J1 jumper is connected to the FLASHCS side. Use the LM Flash Programmer Download to download ektm4c129_qssi_bootloader on an TM4C1294NCPDT Connected LaunchPad that has been erased. (After performing checks for QSSI and SDRAM memory, the bootloader activates UART0 to download an image to the external QSSI Flash, . For the setting of the LM Flash Programmer, see Figure 12 and Figure 13.)



LM Flash Programmer - Build 1613

Quick Set: Manual Configuration - see below

Interface: ICDI (Eval Board)

Select ICDI JTAG

Port: JTAG

Speed (Hz): 1000000

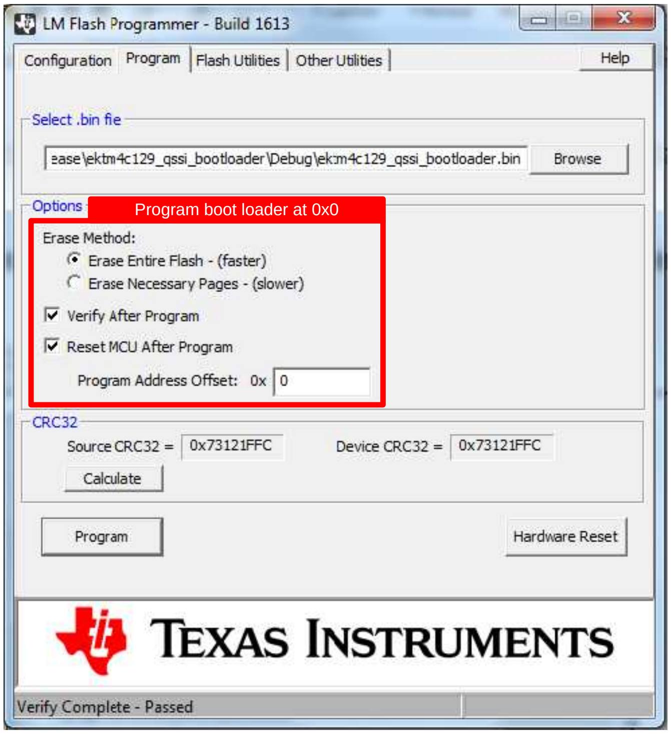
Clock Source:

○ Using the Selected Crystal Value: 1 MHz

● Using the Specfied Single Ended Source Value (Hz): 6000000

Erase Complete

Figure 12. LM Flash Programmer Main Bootloader Programming - Interface Setting (JTAG)



LM Flash Programmer - Build 1613

Select .bin fie: ease\ektm4c129_qssi_bootloader\Debug\ektm4c129_qssi_bootloader.bin

Program boot loader at 0x0

Options

Erase Method:

● Erase Entire Flash - (faster)

○ Erase Necessary Pages - (slower)

☑ Verify After Program

☑ Reset MCU After Program

Program Address Offset: 0x 0

CRC32

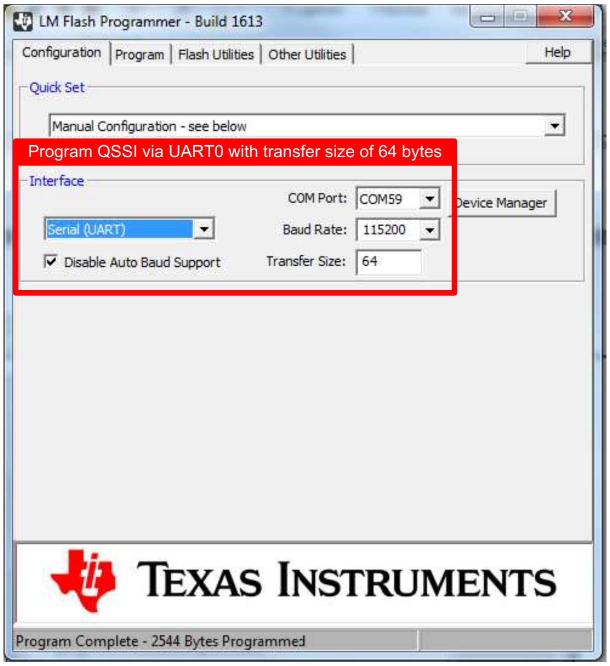
Source CRC32 = 0x73121FFC

Device CRC32 = 0x73121FFC

Verify Complete - Passed

Figure 13. LM Flash Programmer Main Bootloader Programming - Options Settings

5. Using the LM Flash Programmer in serial mode, download ektm4c129_qssi_boot_demo1. Ensure Disable Auto Baud Support is checked. Select the correct COM port. Ensure the Transfer Size is 64 bytes or less. On the Program Tab, select the Program Address Offset as the start of a sector of QSSI Flash other than Sector-0. (For the setting of the LM Flash Progammer for downloading the demo code, see Figure 14 and Figure 15. When the demo code downloads, the LED D3 will start blinking.)



Quick Set: Manual Configuration - see below

Program QSSI via UART0 with transfer size of 64 bytes

Interface: Serial (UART)

COM Port: COM59

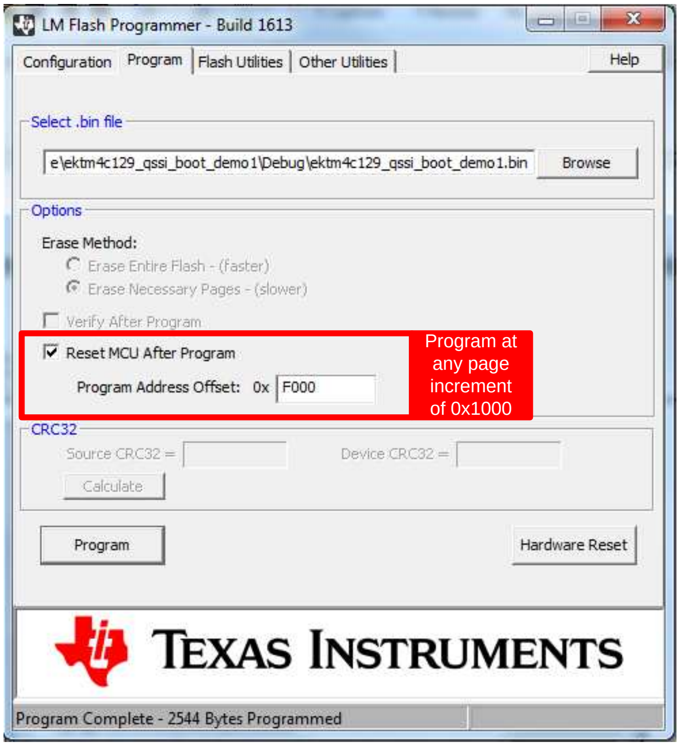
Baud Rate: 115200

☑ Disable Auto Baud Support

Transfer Size: 64

Program Complete - 2544 Bytes Programmed

Figure 14. LM Flash Programmer QSSI Boot Demo1 Programming - Interface Settings (UART)



Select .bin file: e\ektm4c129_qssi_boot_demo1\Debug\ektm4c129_qssi_boot_demo1.bin

Options

Erase Method:

○ Erase Entire Flash - (faster)

● Erase Necessary Pages - (slower)

□ Verify After Program

☑ Reset MCU After Program

Program Address Offset: 0x F000

Program at any page increment of 0x1000

CRC32

Source CRC32 =

Device CRC32 =

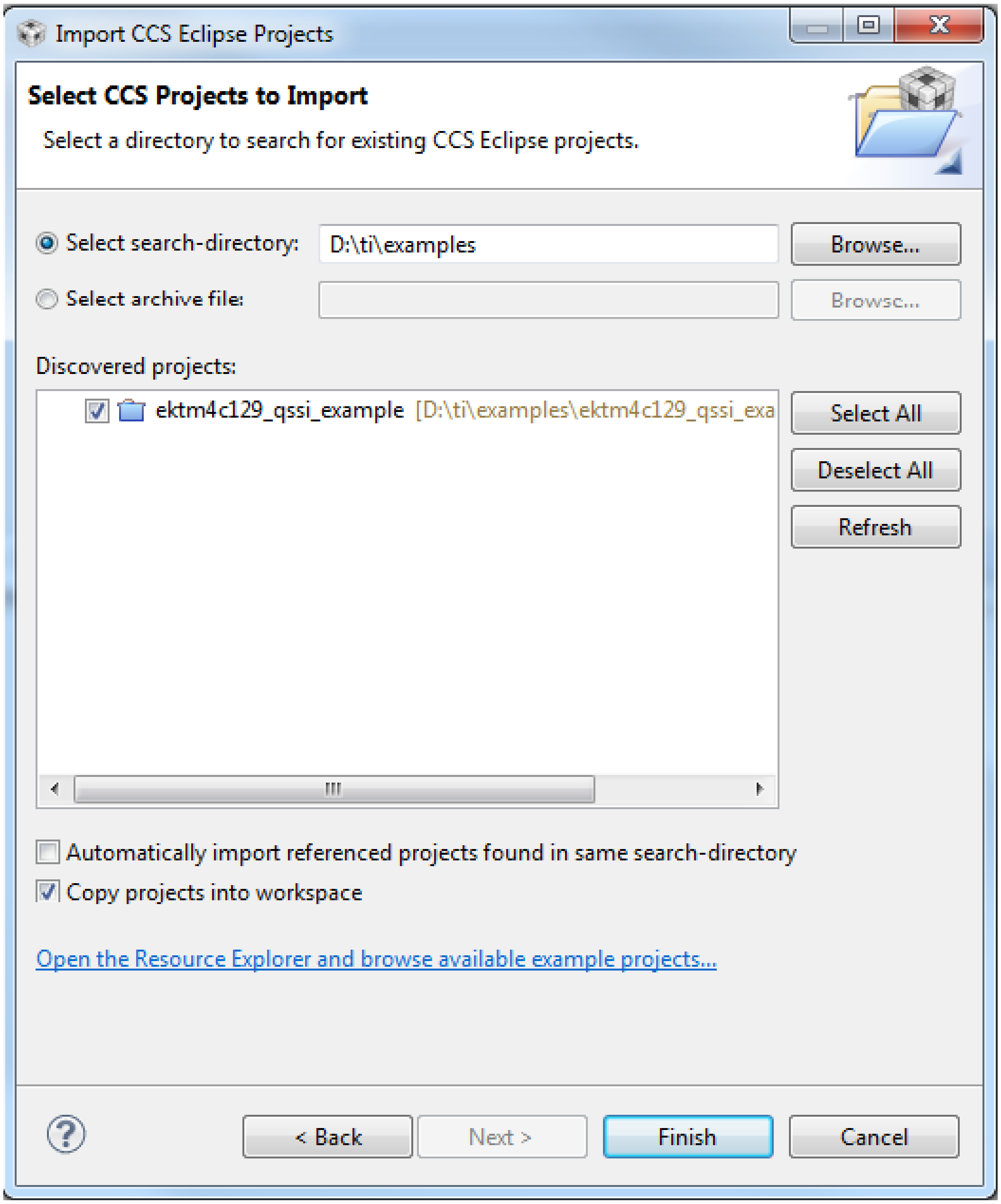
Program Complete - 2544 Bytes Programmed

Figure 15. LM Flash Programmer QSSI Boot Demo 1 Programming - Options (Address Offset)

6. Use USR_SW2 to accelerate the blinking rate and USR_SW1 to reduce the blinking rate.

5.4 Software Setup (QSSI Bare Metal Example)

1. Download the example software package from TIDM-TM4C129SDRAMNVM. Unzip the software package.
2. Launch Code Composer Studio v6.0.1 or later→ Click Import→ Click CCS Projects→ Click Next. Browse to the directory with the software examples. Select " ektm4c129_qssi_example". Click Finish.



Import CCS Eclipse Projects

Select CCS Projects to Import

Select a directory to search for existing CCS Eclipse projects.

● Select search-directory: D:\ti\examples

○ Select archive file:

Discovered projects:

☑ ektm4c129_qssi_example [D:\ti\examples\ektm4c129_qssi_exa

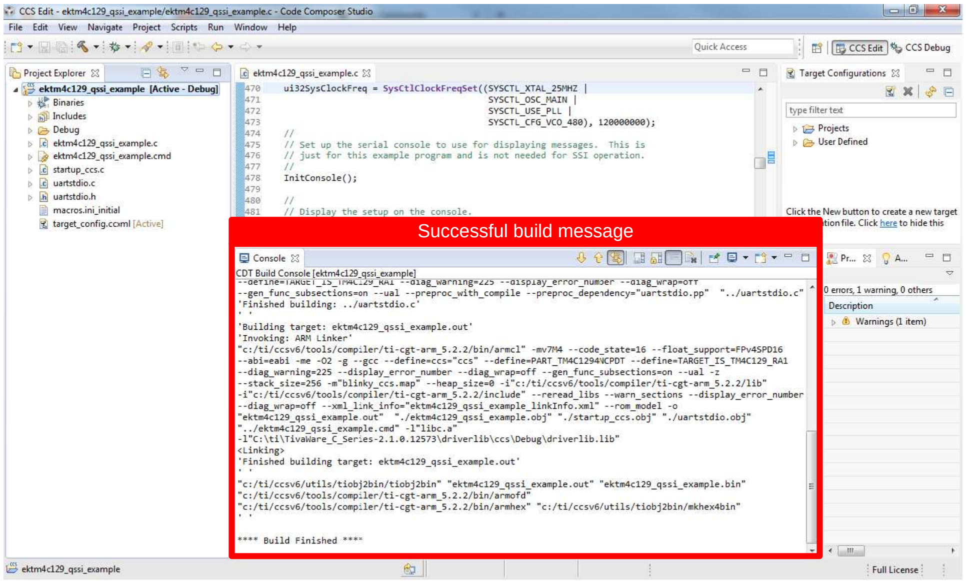
□ Automatically import referenced projects found in same search-directory

☑ Copy projects into workspace

Open the Resource Explorer and browse available example projects...

Figure 16. QSSI Bare Metal Project Import

3. Build each project. To build a project, right-click on a project. Click Rebuild Project. Ensure the projects compile free of errors.

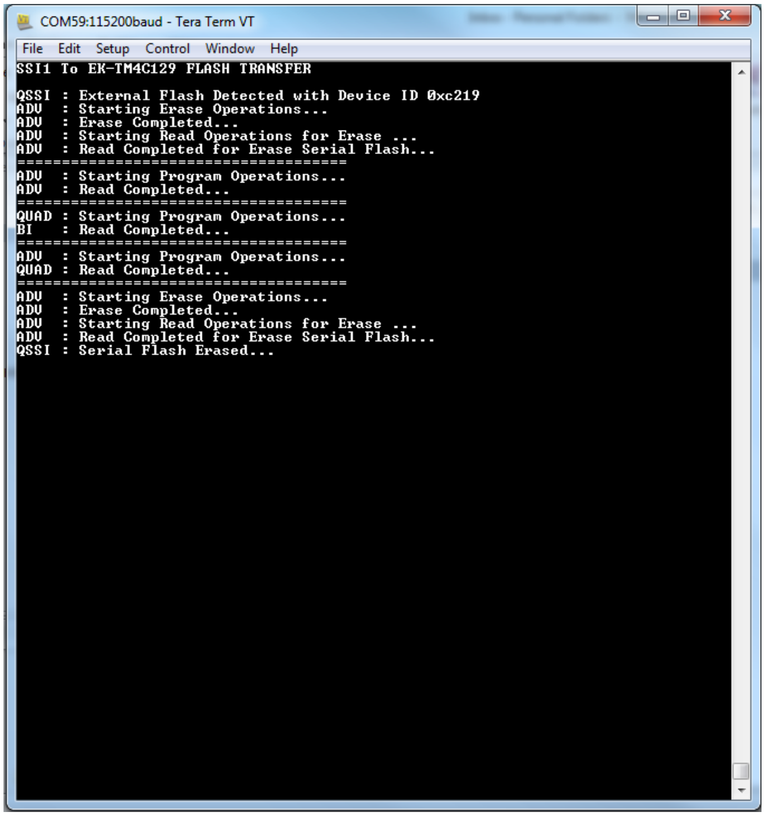

CCS Edit - ektm4c129_qssi_example/ektm4c129_qssi_example.c - Code Composer Studio
File Edit View Navigate Project Scripts Run Window Help
Quick Access
CCS Edit
CCS Debug
Project Explorer
ektm4c129_qssi_example [Active - Debug]
Binaries
Includes
Debug
ektm4c129_qssi_example.c
ektm4c129_qssi_example.cmd
startup_ccs.c
uartstdio.c
uartstdio.h
macros.ini_initial
target_config.ccxml [Active]
ektm4c129_qssi_example.c
Target Configurations
type filter text
Projects
User Defined
Click the New button to create a new target
tion file. Click here to hide this
Successful build message
Console
Pr...
A...
0 errors, 1 warning, 0 others
Description
Warnings (1 item)
ektm4c129_qssi_example
Full License

```
    ui32SysClockFreq = SysCtlClockFreqSet((SYSCTL_XTAL_25MHZ |
                                           SYSCTL_OSC_MAIN |
                                           SYSCTL_USE_PLL |
                                           SYSCTL_CFG_VCO_480), 120000000);
    //
    // Set up the serial console to use for displaying messages.  This is
    // just for this example program and is not needed for SSI operation.
    //
    InitConsole();

    //
    // Display the setup on the console.
```

```
CDT Build Console [ektm4c129_qssi_example]
--define=TARGET_IS_TM4C129_RA1 --diag_warning=225 --display_error_number --diag_wrap=off
--gen_func_subsections=on --ual --preproc_with_compile --preproc_dependency="uartstdio.pp"  "../uartstdio.c"
'Finished building: ../uartstdio.c'
' '
'Building target: ektm4c129_qssi_example.out'
'Invoking: ARM Linker'
"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/bin/armcl" -mv7M4 --code_state=16 --float_support=FPv4SPD16
--abi=eabi -me -O2 -g --gcc --define=ccs="ccs" --define=PART_TM4C1294NCPDT --define=TARGET_IS_TM4C129_RA1
--diag_warning=225 --display_error_number --diag_wrap=off --gen_func_subsections=on --ual -z
--stack_size=256 -m"blinky_ccs.map" --heap_size=0 -i"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/lib"
-i"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/include" --reread_libs --warn_sections --display_error_number
--diag_wrap=off --xml_link_info="ektm4c129_qssi_example_linkInfo.xml" --rom_model -o
"ektm4c129_qssi_example.out"  "./ektm4c129_qssi_example.obj" "./startup_ccs.obj" "./uartstdio.obj"
"../ektm4c129_qssi_example.cmd" -l"libc.a"
-l"C:\ti\TivaWare_C_Series-2.1.0.12573\driverlib\ccs\Debug\driverlib.lib"
<Linking>
'Finished building target: ektm4c129_qssi_example.out'
' '
"c:/ti/ccsv6/utils/tiobj2bin/tiobj2bin" "ektm4c129_qssi_example.out" "ektm4c129_qssi_example.bin"
"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/bin/armofd"
"c:/ti/ccsv6/tools/compiler/ti-cgt-arm_5.2.2/bin/armhex" "c:/ti/ccsv6/utils/tiobj2bin/mkhex4bin"
' '

**** Build Finished ****
```

Figure 17. QSSI Bare Metal Compile

4. Ensure the J1 jumper is connected to the FLASHCS side. Press Debug to download "ektm4c129_qssi_example" and load the code into the TM4C1294NCPDT flash. Press Play when the code has loaded. On the serial console, ensure you see the log file generated for erase, advanced mode program, quad mode program, advanced mode read, bi-mode read, and quad mode read.


COM59:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
SSI1 To EK-TM4C129 FLASH TRANSFER

QSSI : External Flash Detected with Device ID 0xc219
ADV  : Starting Erase Operations...
ADV  : Erase Completed...
ADV  : Starting Read Operations for Erase ...
ADV  : Read Completed for Erase Serial Flash...
=====================================
ADV  : Starting Program Operations...
ADV  : Read Completed...
=====================================
QUAD : Starting Program Operations...
BI   : Read Completed...
=====================================
ADV  : Starting Program Operations...
QUAD : Read Completed...
=====================================
ADV  : Starting Erase Operations...
ADV  : Erase Completed...
ADV  : Starting Read Operations for Erase ...
ADV  : Read Completed for Erase Serial Flash...
QSSI : Serial Flash Erased...
```

Figure 18. QSSI Bare Metal Example Serial Console Output

6 Design Files

6.1 Schematics

To download the schematics for the board, see the design files at TIDM-TM4C129SDRAMNVM.

6.2 Bill of Materials

To download the bill of materials (BOM), see the design files at TIDM-TM4C129SDRAMNVM.

6.3 PCB Layout Recommendations

When performing the layout, ensure that the EPI0S31 (the SDRAM clock pin) has the shortest trace. To ensure that reflections from the shared data and address pins minimized, use a single route from the connector pin to either of the address or data pin without creating a stub.

6.4 Layout Prints

To download the layout prints for the board, see the design files at TIDM-TM4C129SDRAMNVM.

7 Altium Project

To download the Altium project files, see the design files at TIDM-TM4C129SDRAMNVM.

8 Gerber Files

To download the Gerber files, see the design files at TIDM-TM4C129SDRAMNVM.

9 Software Files

To download the software files, see the design files at TIDM-TM4C129SDRAMNVM.

10 References

1. ISSI 512Mbit SDRAM Memory
2. Macronix 5112Mbit QSSI Serial Flash

11 About the Author

AMIT ASHARA is an application engineer at TI, where he develops applications for the TM4C12x family of high-performance microcontrollers. Amit brings to this role his extensive experience and expertise in high-speed digital and microcontroller system-level design. Amit earned his Bachelor of Engineering (BE) from the University of Pune in India.

IMPORTANT NOTICE FOR TI REFERENCE DESIGNS

Texas Instruments Incorporated ("TI") reference designs are solely intended to assist designers (“Buyers”) who are developing systems that incorporate TI semiconductor products (also referred to herein as “components”). Buyer understands and agrees that Buyer remains responsible for using its independent analysis, evaluation and judgment in designing Buyer’s systems and products.

TI reference designs have been created using standard laboratory conditions and engineering practices. **TI has not conducted any testing other than that specifically described in the published documentation for a particular reference design.** TI may make corrections, enhancements, improvements and other changes to its reference designs.

Buyers are authorized to use TI reference designs with the TI component(s) identified in each particular reference design and to modify the reference design in the development of their end products. HOWEVER, NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY THIRD PARTY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT, IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI REFERENCE DESIGNS ARE PROVIDED "AS IS". TI MAKES NO WARRANTIES OR REPRESENTATIONS WITH REGARD TO THE REFERENCE DESIGNS OR USE OF THE REFERENCE DESIGNS, EXPRESS, IMPLIED OR STATUTORY, INCLUDING ACCURACY OR COMPLETENESS. TI DISCLAIMS ANY WARRANTY OF TITLE AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, QUIET ENJOYMENT, QUIET POSSESSION, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS WITH REGARD TO TI REFERENCE DESIGNS OR USE THEREOF. TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY BUYERS AGAINST ANY THIRD PARTY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON A COMBINATION OF COMPONENTS PROVIDED IN A TI REFERENCE DESIGN. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, SPECIAL, INCIDENTAL, CONSEQUENTIAL OR INDIRECT DAMAGES, HOWEVER CAUSED, ON ANY THEORY OF LIABILITY AND WHETHER OR NOT TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, ARISING IN ANY WAY OUT OF TI REFERENCE DESIGNS OR BUYER’S USE OF TI REFERENCE DESIGNS.

TI reserves the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques for TI components are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

Reproduction of significant portions of TI information in TI data books, data sheets or reference designs is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards that anticipate dangerous failures, monitor failures and their consequences, lessen the likelihood of dangerous failures and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in Buyer’s safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed an agreement specifically governing such use.

Only those TI components that TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components that have ***not*** been so designated is solely at Buyer's risk, and Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265