ARM Cortex-A15 Bare-Mental Big-Endian CCS Project Operation Guide

1. Environment Requirements

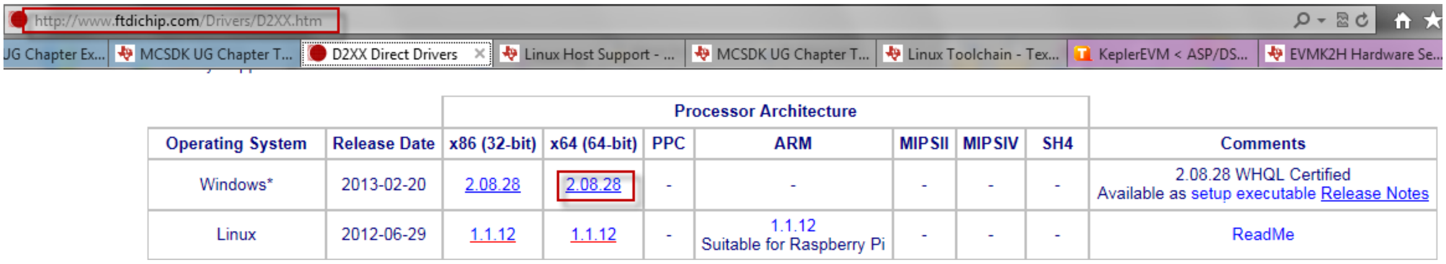
1.1 Download the Latest Version of CCS

Due to ARM Cortex-A15 need to be compiled by GCCv4.7.x or CL470v5.x.x, which need CCS supports these versions of compilers and relative hardware targets, user should use the CCS v5.4.0.00091 or higher version, (can get from http://processors.wiki.ti.com/index.php/Download_CCS)

1.2 Install TCI6638 EVM USB to COM Driver and Hardware Configuration

Due to the TCI6638 EVM have the ftdi chip to convert the COM to USB interface, we need to download and install the ftdi drivers on the PC side to use the COM port. You should download and install the driver from the following link:

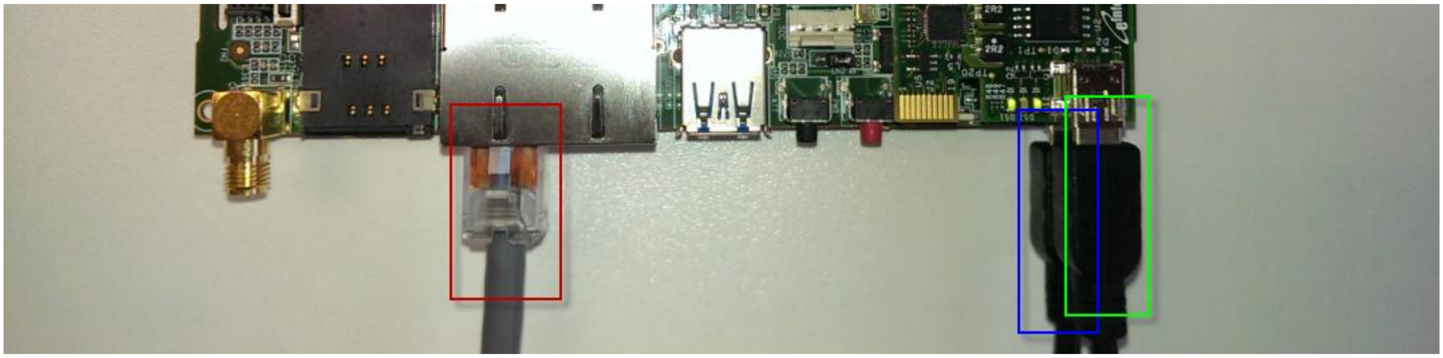
http://www.ftdichip.com/Drivers/D2XX.htm



		Processor Architecture							
Operating System	Release Date	x86 (32-bit)	x64 (64-bit)	PPC	ARM	MIPSII	MIPSIV	SH4	Comments
Windows*	2013-02-20	2.08.28	2.08.28	-	-	-	-	-	2.08.28 WHQL Certified Available as setup executable Release Notes
Linux	2012-06-29	1.1.12	1.1.12	-	1.1.12 Suitable for Raspberry Pi	-	-	-	ReadMe

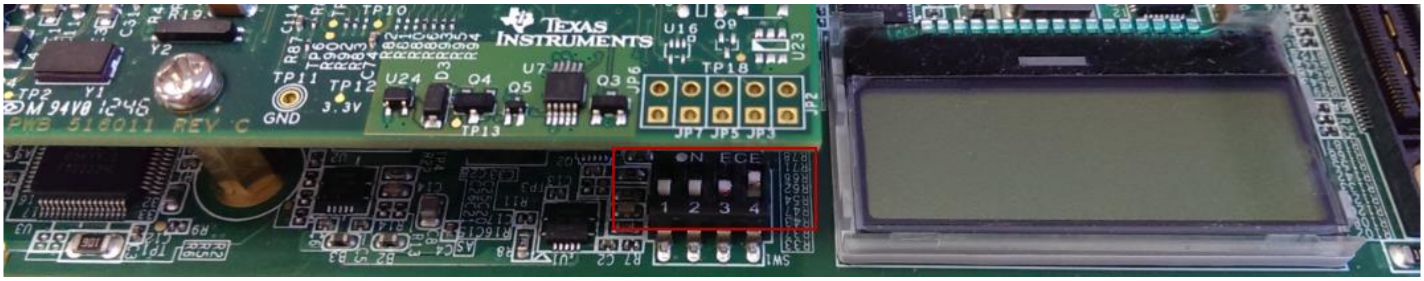
Choose the latest Windows version to download and install, e.g: “CDM 2.08.28 WHQL Certified.7z”

Connect the USB cable before power on the EVM board:



There are 2 mini-USB ports, the one on the mezzanine board is the USB XDS200 Emulator cable, the one on the main EVM board is the USB-COM cable. Make sure to connect 2 USB cables on your PC.

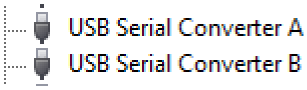
Before we power on the EVM, please make sure the boot DIP pins on EVM in the position like:



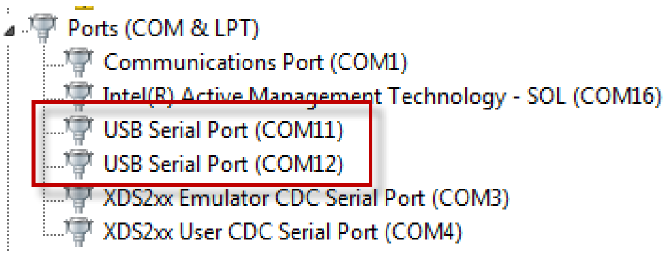
This makes the board in “no boot” mode for running U-boot through CCS.

Power on the EVM and the driver of the Emulator and USB-COM will be installed on your PC.

If the USB-COM drivers are correctly installed, you should found 2 more items were showed in your device manager:

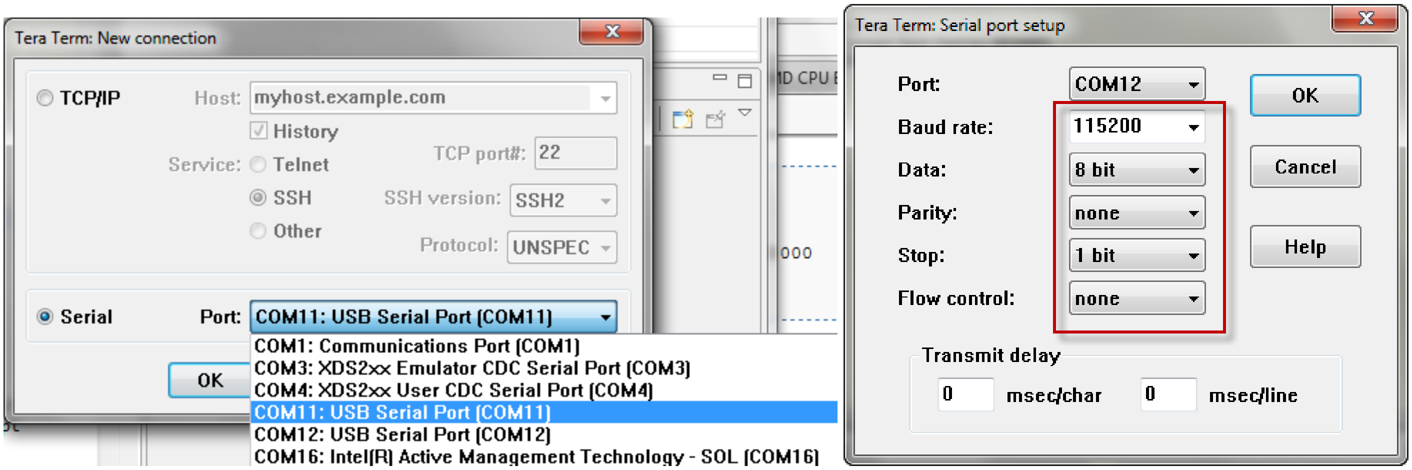


USB Serial Converter A
USB Serial Converter B



Ports (COM & LPT)
Communications Port (COM1)
Intel(R) Active Management Technology - SOL (COM16)
USB Serial Port (COM11)
USB Serial Port (COM12)
XDS2xx Emulator CDC Serial Port (COM3)
XDS2xx User CDC Serial Port (COM4)

Then open the hyper-terminal tool (we use TeraTerm as example) to create 2 sessions on your PC:

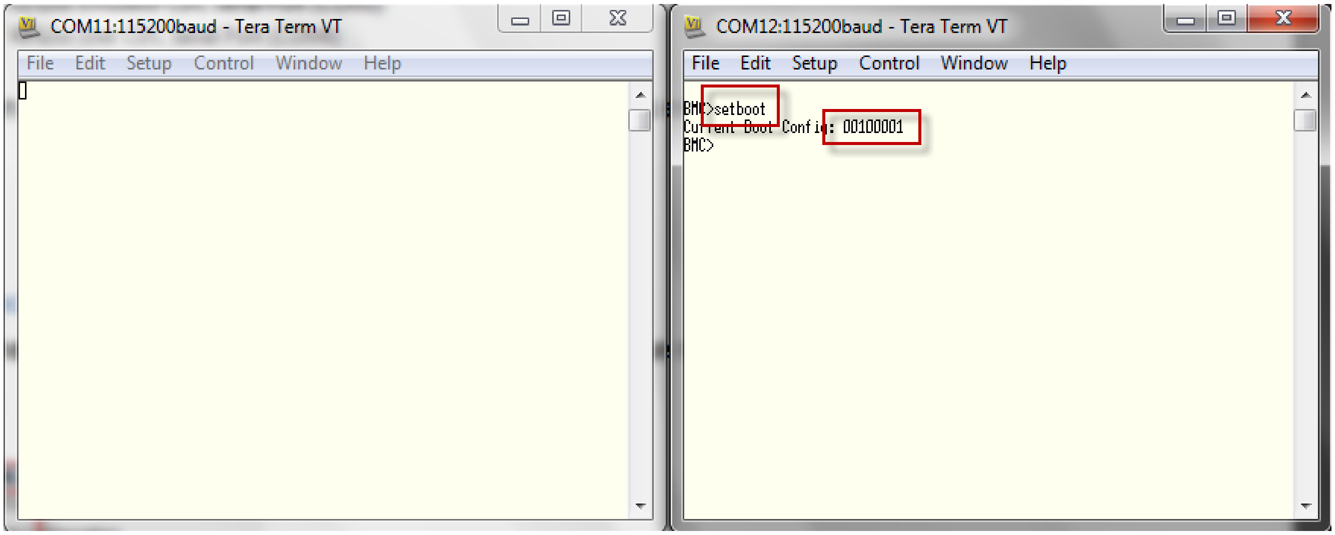


Host: myhost.example.com
TCP port#: 22
SSH version: SSH2
Protocol: UNSPEC
Port: COM11: USB Serial Port (COM11)

COM1: Communications Port (COM1)
COM3: XDS2xx Emulator CDC Serial Port (COM3)
COM4: XDS2xx User CDC Serial Port (COM4)
COM11: USB Serial Port (COM11)
COM12: USB Serial Port (COM12)
COM16: Intel(R) Active Management Technology - SOL (COM16)

Port: COM12
Baud rate: 115200
Data: 8 bit
Parity: none
Stop: 1 bit
Flow control: none
Transmit delay
0 msec/char 0 msec/line

After we created 2 sessions, you will see the terminals such as:


COM11:115200baud - Tera Term VT
File Edit Setup Control Window Help
COM12:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
BMC>setboot
Current Boot Config: 00100001
BMC>
```

The one which started with “BMC>” is the control port to configure the boot mode pins of TCI6638, the most used command of BMC is listed as follow:

`setboot`: print the current boot mode

`setboot 00100001` or `00112005`: set the boot mode, 00100001: no boot (for running u-boot on CCS); 00112005: SPI flash boot (for booting u-boot on SPI flash)

`fullrst`: full reset to make the boot mode change take effect.

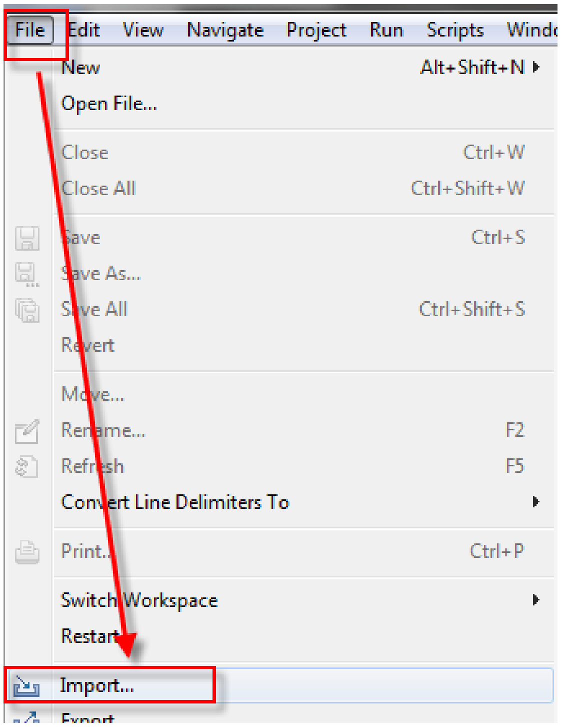
Please note that the DIP switch only takes effect on the first power on when connecting the power cable, other boot modes can be decided by the “setboot” commands on BMC and take effect after the command “fullrst”.

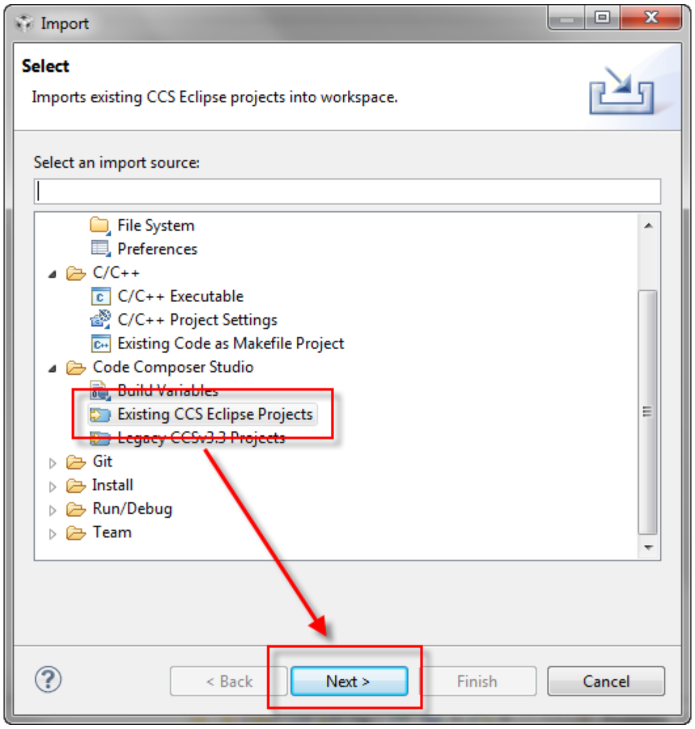
If you change the boot mode by using the BMC command, the boot mode will change back to the value of the DIP switch after you disconnect then connect the power cable.

For more information about the boot mode DIP switch and the BMC encoding, please refer to the section “Appendix”

2. Import CCS Projects

Click “File” and “Import”, choose “Existing CCS Eclipse Projects” and click “Next”:



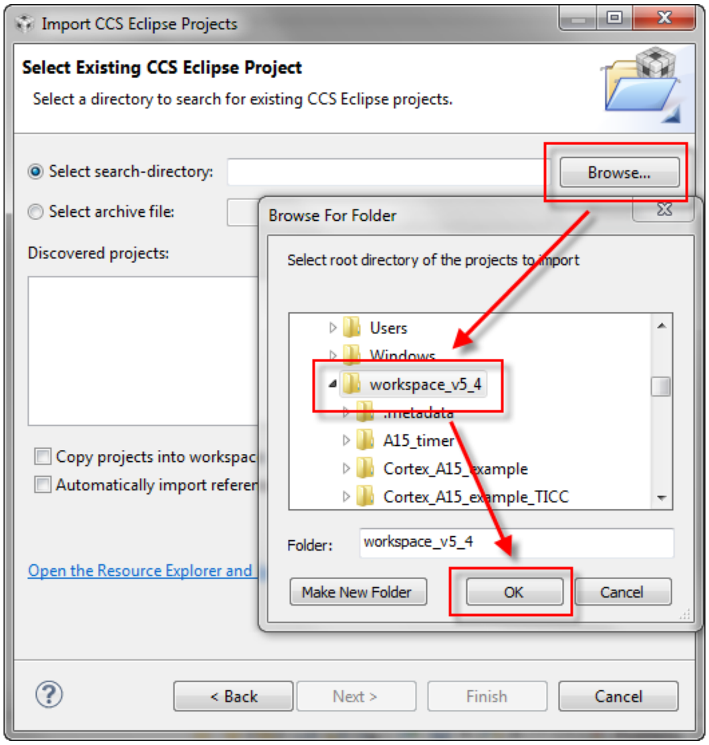

Import

Select

Imports existing CCS Eclipse projects into workspace.

Select an import source:

Then choose the project path:


Import CCS Eclipse Projects

Select Existing CCS Eclipse Project

Select a directory to search for existing CCS Eclipse projects.

● Select search-directory:

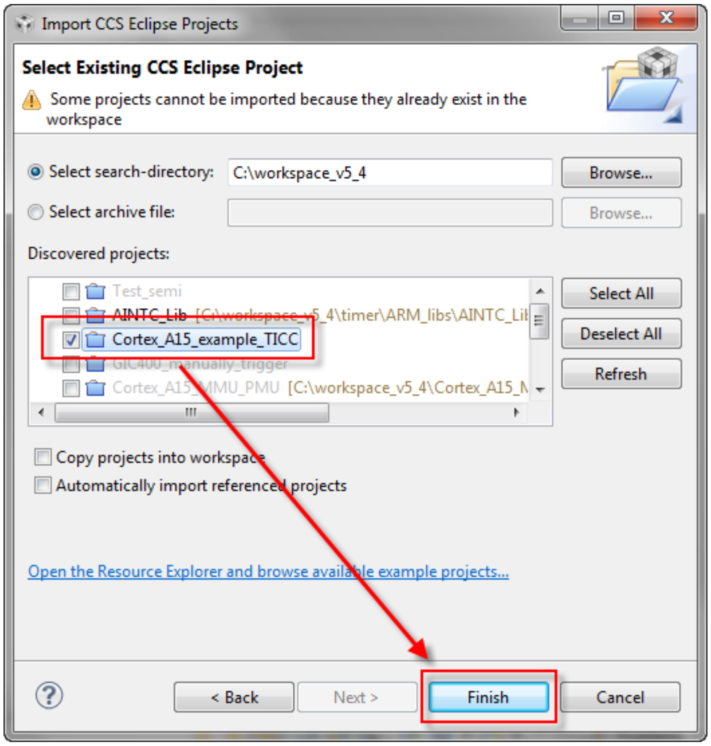
○ Select archive file:

Discovered projects:

Browse For Folder

Select root directory of the projects to import

Folder: workspace_v5_4


Import CCS Eclipse Projects

Select Existing CCS Eclipse Project

Some projects cannot be imported because they already exist in the workspace

● Select search-directory: C:\workspace_v5_4

○ Select archive file:

Discovered projects:

☐ Test_semi

☐ AINTC_Lib [C:\workspace_v5_4\timer\ARM_libs\AINTC_Lil

☑ Cortex_A15_example_TICC

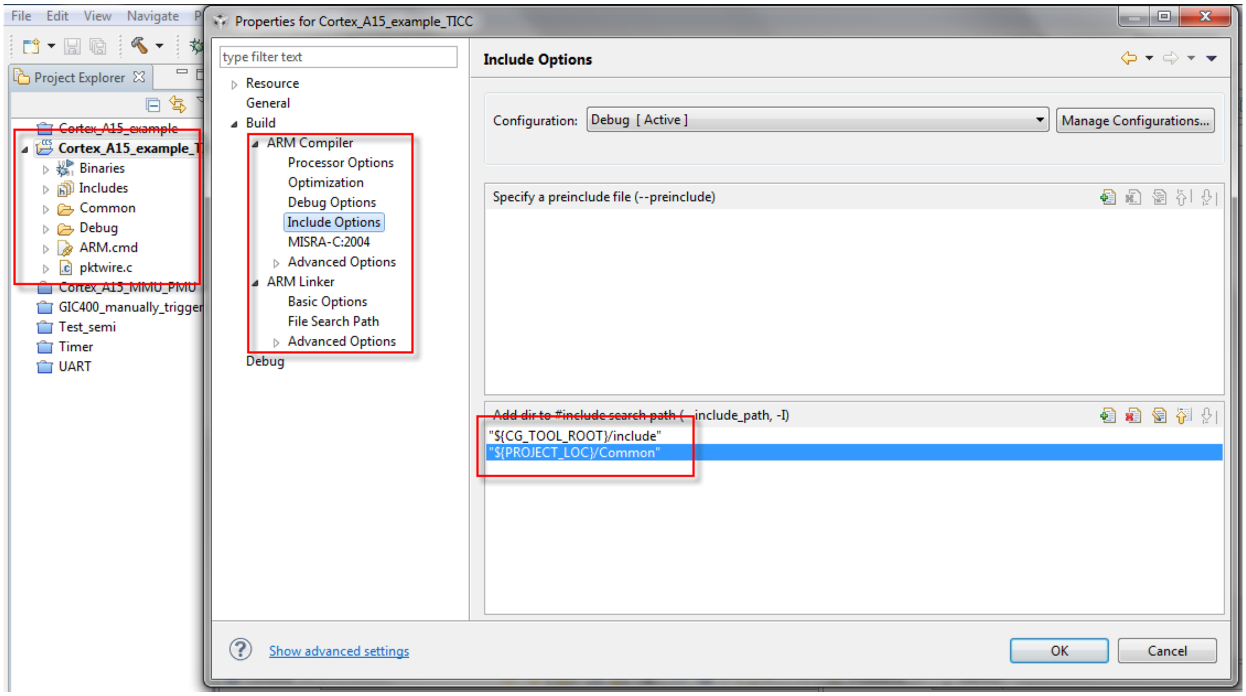
☐ GIC400_manually_trigger

☐ Cortex_A15_MMU_PMU [C:\workspace_v5_4\Cortex_A15_N

☐ Copy projects into workspace

☐ Automatically import referenced projects

After importing the project, you can see this project in the “Project Explorer” windows, you should change relative project link path, include path environmental variables to your PC environment:



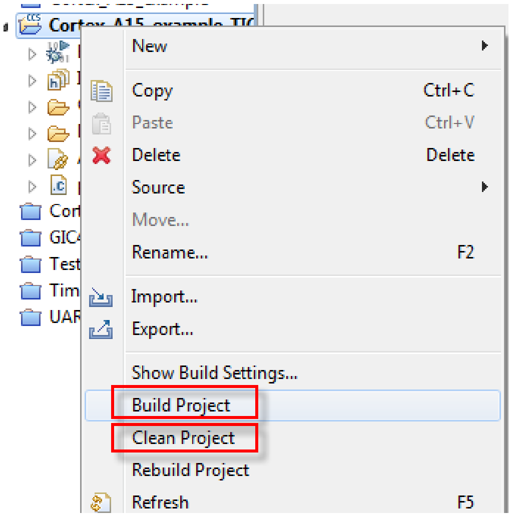
Configuration: Debug [Active]

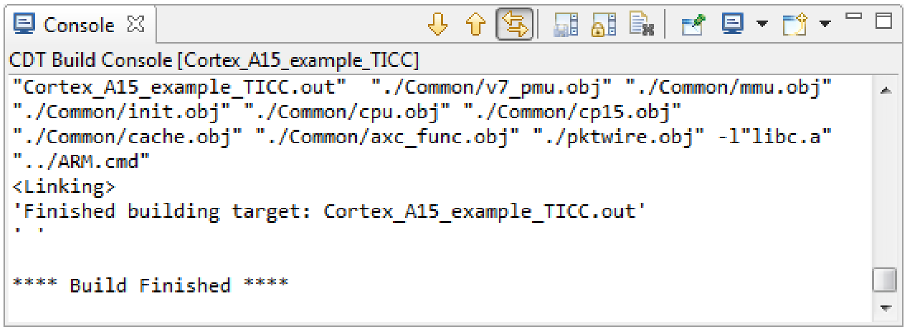
Specify a preinclude file (--preinclude)

Add dir to #include search path (--include_path, -I)

```
"${CG_TOOL_ROOT}/include"
"${PROJECT_LOC}/Common"
```

Then clean the project and build the project again, if the configuration is correct, it should pass and build the executable:



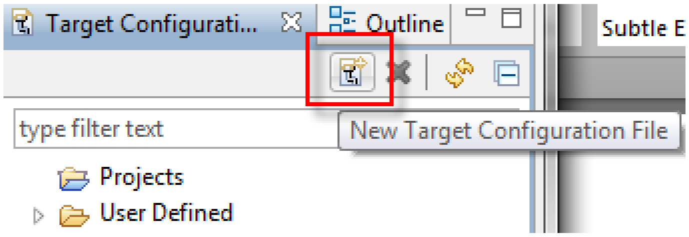


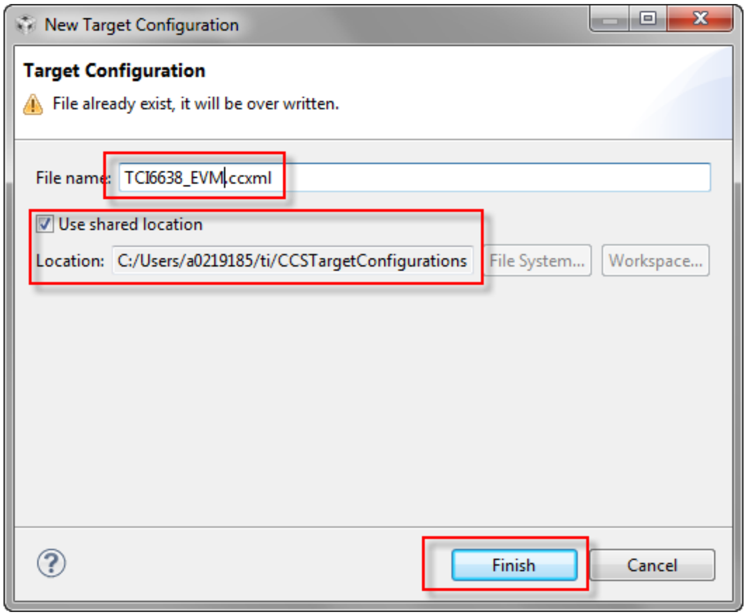
CDT Build Console [Cortex_A15_example_TICC]

```
"Cortex_A15_example_TICC.out"  "./Common/v7_pmu.obj" "./Common/mmu.obj"
"./Common/init.obj" "./Common/cpu.obj" "./Common/cp15.obj"
"./Common/cache.obj" "./Common/axc_func.obj" "./pktwire.obj" -l"libc.a"
"../ARM.cmd"
<Linking>
'Finished building target: Cortex_A15_example_TICC.out'
' '


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

3. Create CCXML Configuration File

Create a new CCXML configuration file and choose the expected name and location:




New Target Configuration

Target Configuration

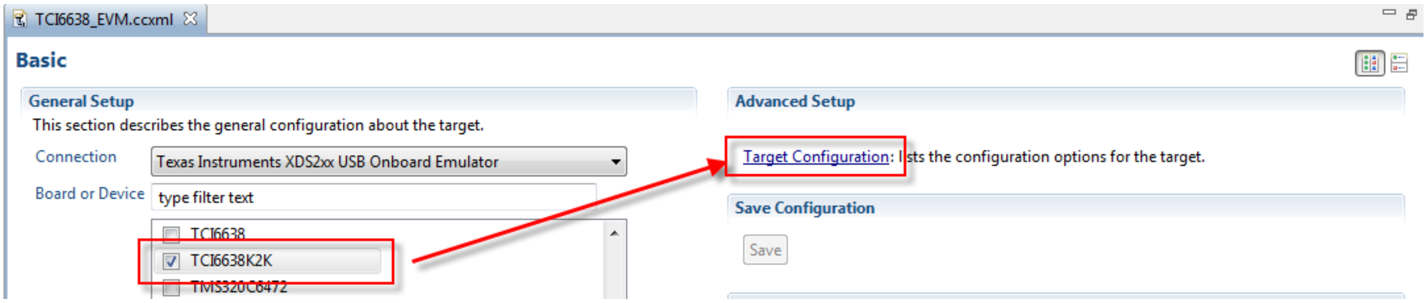
File already exist, it will be over written.

File name: TCI6638_EVM.ccxml

☑ Use shared location

Location: C:/Users/a0219185/ti/CCSTargetConfigurations

Choose the target “TCI6638K2K” and then click “Target Configurations”:


TCI6638_EVM.ccxml

Basic

General Setup

This section describes the general configuration about the target.

Connection: Texas Instruments XDS2xx USB Onboard Emulator

Board or Device: type filter text

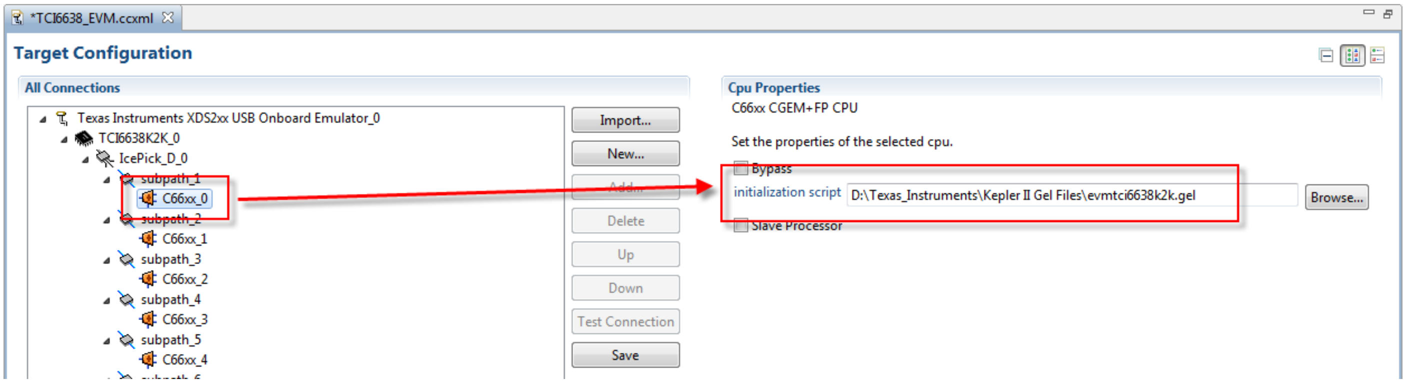
☐ TCI6638
☑ TCI6638K2K
☐ TMS320C6472

Advanced Setup

Target Configuration: lists the configuration options for the target.

Save Configuration

Choose DSP Core0 and select the related GEL file for DSP target:


*TCI6638_EVM.ccxml

Target Configuration

All Connections

- Texas Instruments XDS2xx USB Onboard Emulator_0
  - TCI6638K2K_0
    - IcePick_D_0
      - subpath_1
        - C66xx_0
      - subpath_2
        - C66xx_1
      - subpath_3
        - C66xx_2
      - subpath_4
        - C66xx_3
      - subpath_5
        - C66xx_4

Cpu Properties

C66xx CGEM+FP CPU

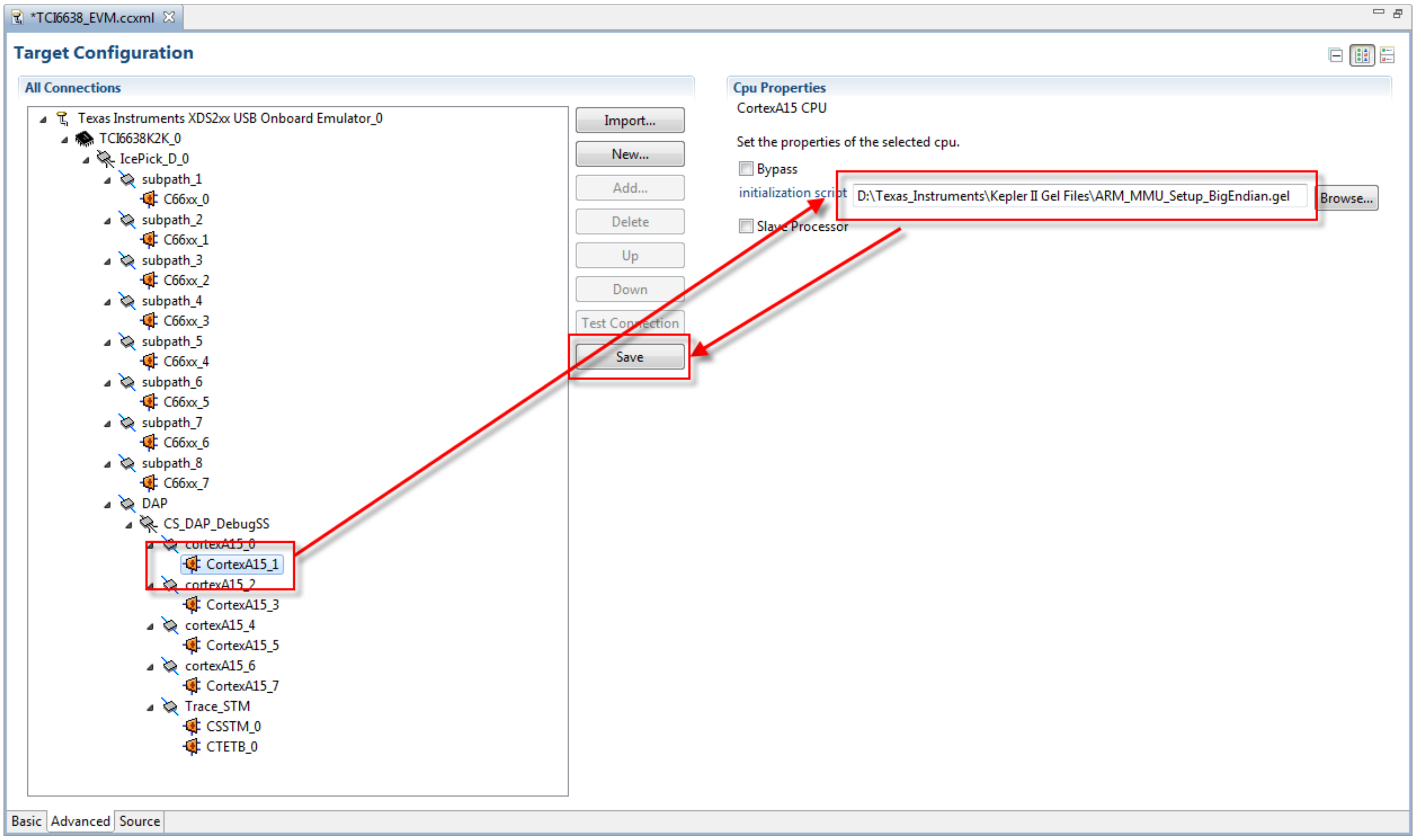
Set the properties of the selected cpu.

☐ Bypass

initialization script: D:\Texas_Instruments\Kepler II Gel Files\evmtci6638k2k.gel

☐ Slave Processor

Choose the ARM Core0 and select the relative GEL file for ARM target and then save the CCXML configuration file:



CortexA15 CPU

Set the properties of the selected cpu.

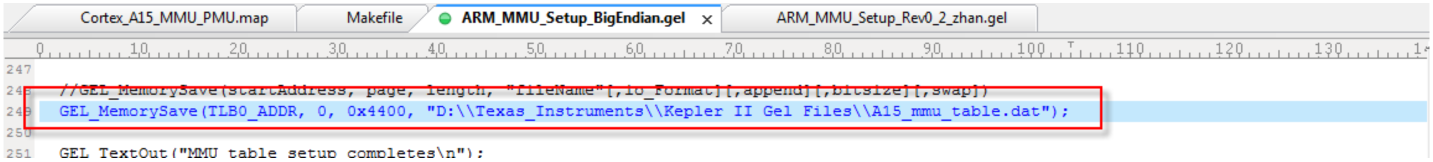
☐ Bypass

initialization script: D:\Texas_Instruments\Kepler II Gel Files\ARM_MMU_Setup_BigEndian.gel

☐ Slave Processor

4. Modify the GEL File

In Gel file “ARM_MMU_Setup_BigEndian.gel”, it will use some dat file which includes the dat file path in the gel file, you should modify the dat file path to your actual path on your PC.

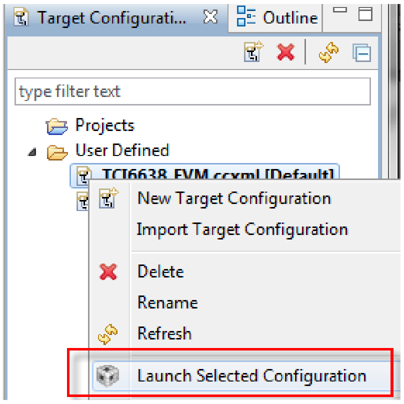


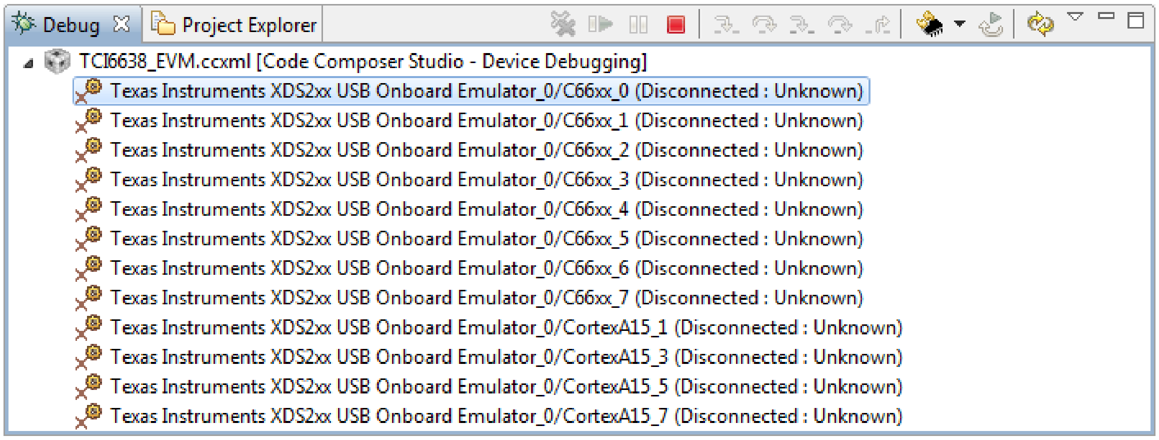
```
//GEL_MemorySave(startAddress, page, length, "fileName"[,io_Format][,append][,bitsize][,swap])
GEL_MemorySave(TLB0_ADDR, 0, 0x4400, "D:\\Texas_Instruments\\Kepler II Gel Files\\A15_mmu_table.dat");

GEL_TextOut("MMU table setup completes\n");
```

There will be 4 places contains the dat file path in the gel file, modify all of them and save the gel file.

5. Launch the Debug Session and Run Executable

Launch the debug session with the CCXML file which you created previously, then you can see the targets:

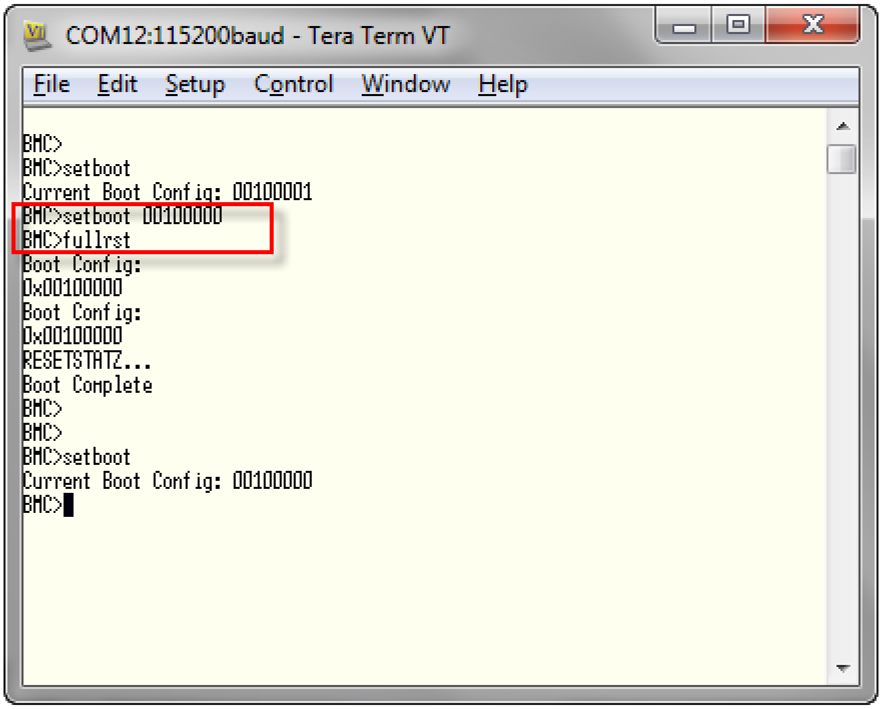




```
TCI6638_EVM.ccxml [Code Composer Studio - Device Debugging]
    Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_0 (Disconnected : Unknown)
    Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_1 (Disconnected : Unknown)
    Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_2 (Disconnected : Unknown)
    Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_3 (Disconnected : Unknown)
    Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_4 (Disconnected : Unknown)
    Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_5 (Disconnected : Unknown)
    Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_6 (Disconnected : Unknown)
    Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_7 (Disconnected : Unknown)
    Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_1 (Disconnected : Unknown)
    Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_3 (Disconnected : Unknown)
    Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_5 (Disconnected : Unknown)
    Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_7 (Disconnected : Unknown)
```

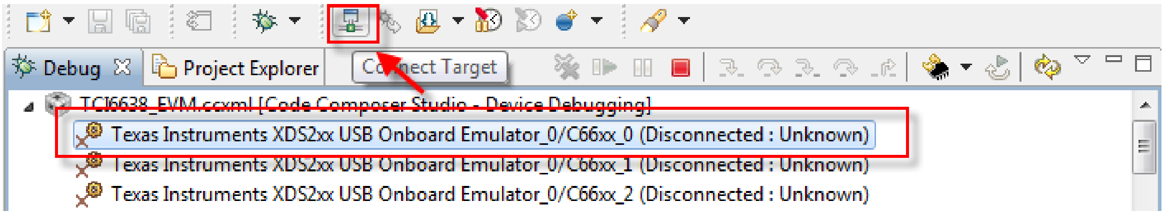
After you see the print info on the TeraTerm, we should use the BMC command to change the DSP to BE mode:

```
BMC> setboot 00100000     // Bit0 stands for DSP Endian, 0:BE 1:LE

BMC> fullrst
```

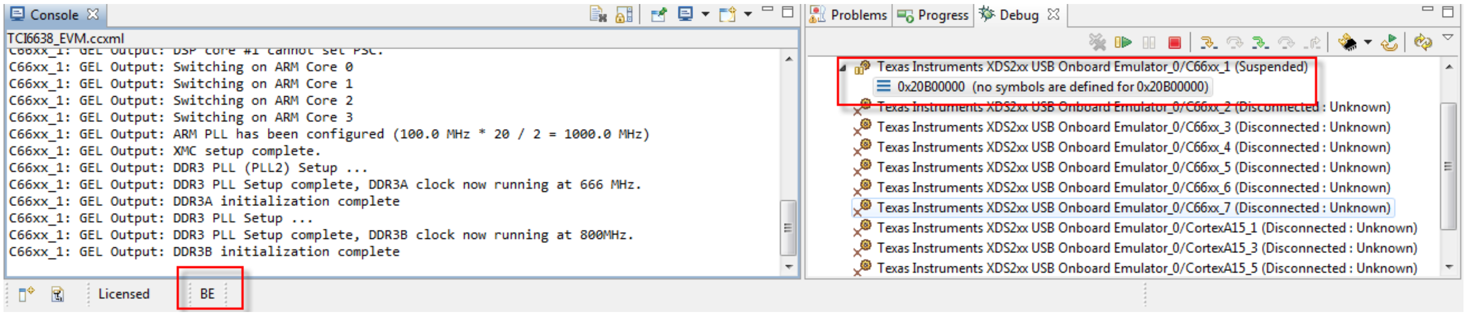


```
BMC>
BMC>setboot
Current Boot Config: 00100001
BMC>setboot 00100000
BMC>fullrst
Boot Config:
0x00100000
Boot Config:
0x00100000
RESETSTATZ...
Boot Complete
BMC>
BMC>
BMC>setboot
Current Boot Config: 00100000
BMC>
```

Then connect the DSP Core0, you will see the DSP in BE mode and GEL takes effect automatically:



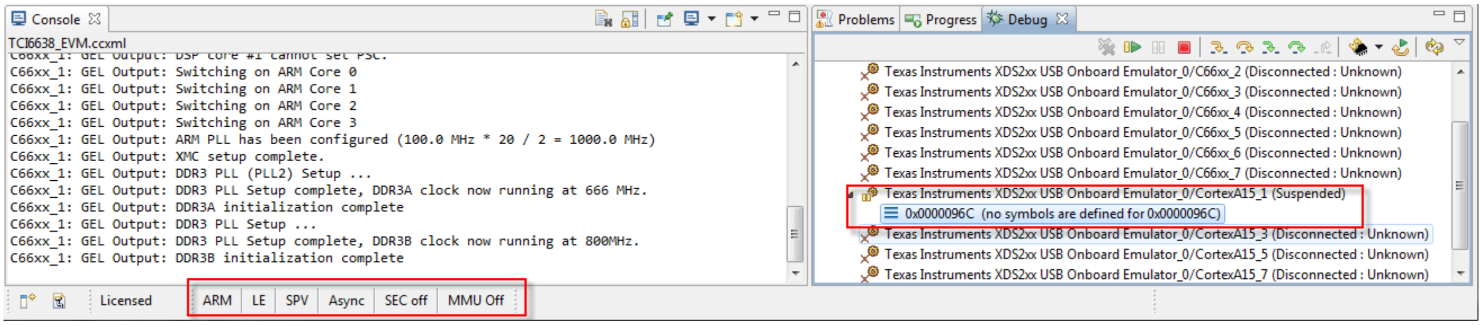
```
TCI6638_EVM.ccxml [Code Composer Studio - Device Debugging]
  Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_0 (Disconnected : Unknown)
  Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_1 (Disconnected : Unknown)
  Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_2 (Disconnected : Unknown)
```



```
TCI6638_EVM.ccxml
C66xx_1: GEL Output: DSP core #1 cannot set PSC.
C66xx_1: GEL Output: Switching on ARM Core 0
C66xx_1: GEL Output: Switching on ARM Core 1
C66xx_1: GEL Output: Switching on ARM Core 2
C66xx_1: GEL Output: Switching on ARM Core 3
C66xx_1: GEL Output: ARM PLL has been configured (100.0 MHz * 20 / 2 = 1000.0 MHz)
C66xx_1: GEL Output: XMC setup complete.
C66xx_1: GEL Output: DDR3 PLL (PLL2) Setup ...
C66xx_1: GEL Output: DDR3 PLL Setup complete, DDR3A clock now running at 666 MHz.
C66xx_1: GEL Output: DDR3A initialization complete
C66xx_1: GEL Output: DDR3 PLL Setup ...
C66xx_1: GEL Output: DDR3 PLL Setup complete, DDR3B clock now running at 800MHz.
C66xx_1: GEL Output: DDR3B initialization complete
```

```
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_1 (Suspended)
  0x20B00000  (no symbols are defined for 0x20B00000)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_2 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_3 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_4 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_5 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_6 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_7 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_1 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_3 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_5 (Disconnected : Unknown)
```

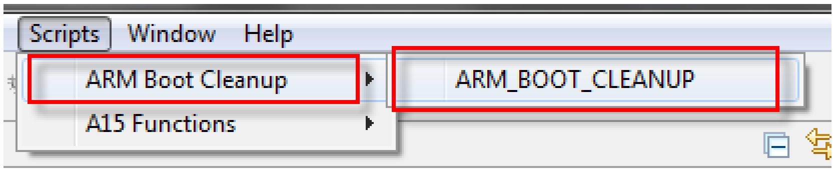
Disconnect the ARM Core0, then connect the ARM Core0, you will see ARM in LE and Supervisor mode:



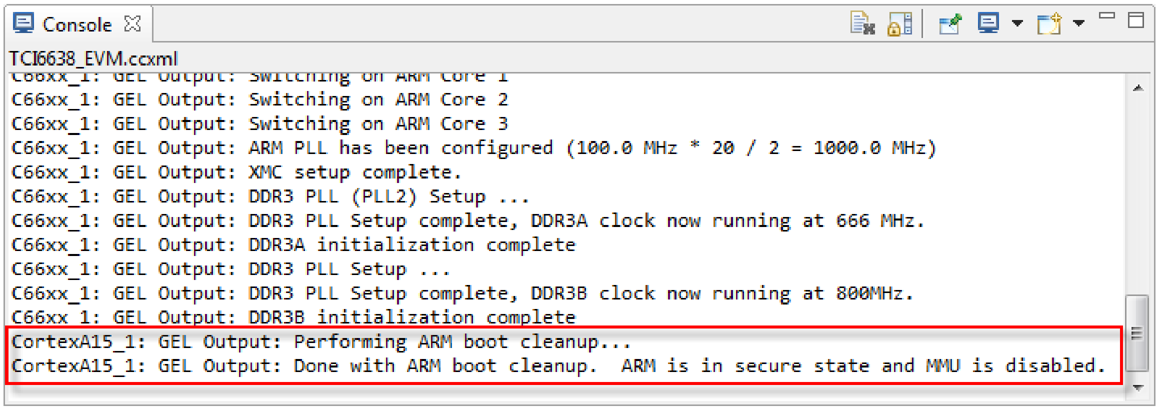
```
TCI6638_EVM.ccxml
C66xx_1: GEL Output: DSP core #1 cannot set PSC.
C66xx_1: GEL Output: Switching on ARM Core 0
C66xx_1: GEL Output: Switching on ARM Core 1
C66xx_1: GEL Output: Switching on ARM Core 2
C66xx_1: GEL Output: Switching on ARM Core 3
C66xx_1: GEL Output: ARM PLL has been configured (100.0 MHz * 20 / 2 = 1000.0 MHz)
C66xx_1: GEL Output: XMC setup complete.
C66xx_1: GEL Output: DDR3 PLL (PLL2) Setup ...
C66xx_1: GEL Output: DDR3 PLL Setup complete, DDR3A clock now running at 666 MHz.
C66xx_1: GEL Output: DDR3A initialization complete
C66xx_1: GEL Output: DDR3 PLL Setup ...
C66xx_1: GEL Output: DDR3 PLL Setup complete, DDR3B clock now running at 800MHz.
C66xx_1: GEL Output: DDR3B initialization complete
```

```
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_2 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_3 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_4 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_5 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_6 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_7 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_1 (Suspended)
  0x0000096C  (no symbols are defined for 0x0000096C)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_3 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_5 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_7 (Disconnected : Unknown)
```

Click “Scripts”->”ARM Boot Cleanup”->”ARM_BOOT_CLEANUP”:



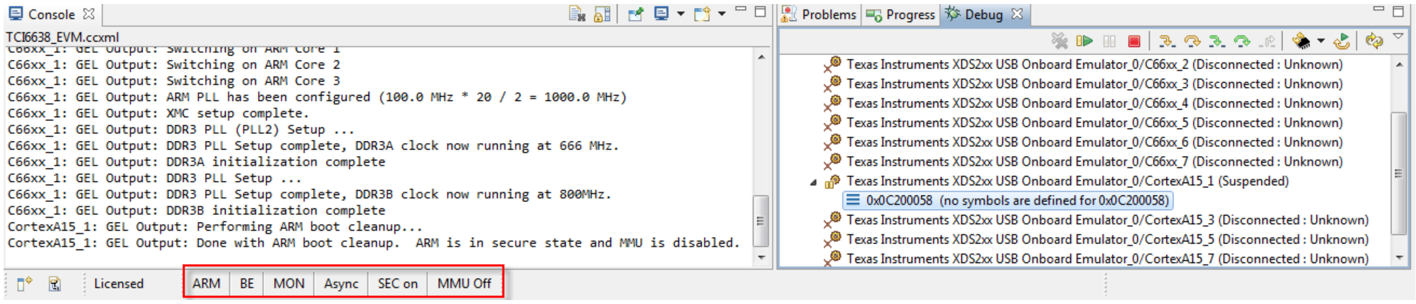
You’ll see the GEL output:



TCI6638_EVM.ccxml

```
C66xx_1: GEL Output: Switching on ARM Core 1
C66xx_1: GEL Output: Switching on ARM Core 2
C66xx_1: GEL Output: Switching on ARM Core 3
C66xx_1: GEL Output: ARM PLL has been configured (100.0 MHz * 20 / 2 = 1000.0 MHz)
C66xx_1: GEL Output: XMC setup complete.
C66xx_1: GEL Output: DDR3 PLL (PLL2) Setup ...
C66xx_1: GEL Output: DDR3 PLL Setup complete, DDR3A clock now running at 666 MHz.
C66xx_1: GEL Output: DDR3A initialization complete
C66xx_1: GEL Output: DDR3 PLL Setup ...
C66xx_1: GEL Output: DDR3 PLL Setup complete, DDR3B clock now running at 800MHz.
C66xx_1: GEL Output: DDR3B initialization complete
CortexA15_1: GEL Output: Performing ARM boot cleanup...
CortexA15_1: GEL Output: Done with ARM boot cleanup.  ARM is in secure state and MMU is disabled.
```

Then disconnect and reconnect the ARM Core0, you will see the ARM Core now in BE and Monitor mode:



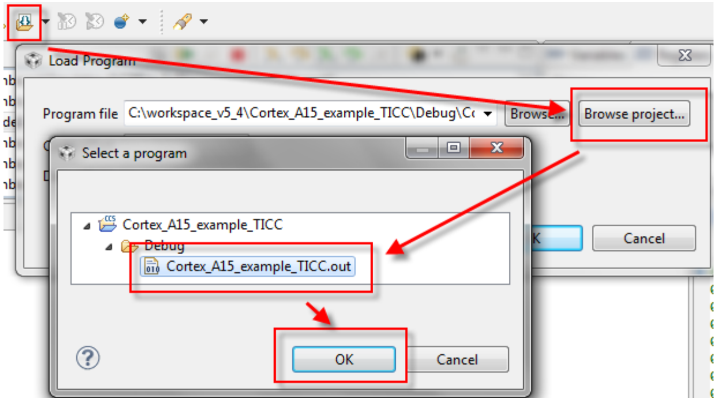
TCI6638_EVM.ccxml

```
C66xx_1: GEL Output: Switching on ARM Core 1
C66xx_1: GEL Output: Switching on ARM Core 2
C66xx_1: GEL Output: Switching on ARM Core 3
C66xx_1: GEL Output: ARM PLL has been configured (100.0 MHz * 20 / 2 = 1000.0 MHz)
C66xx_1: GEL Output: XMC setup complete.
C66xx_1: GEL Output: DDR3 PLL (PLL2) Setup ...
C66xx_1: GEL Output: DDR3 PLL Setup complete, DDR3A clock now running at 666 MHz.
C66xx_1: GEL Output: DDR3A initialization complete
C66xx_1: GEL Output: DDR3 PLL Setup ...
C66xx_1: GEL Output: DDR3 PLL Setup complete, DDR3B clock now running at 800MHz.
C66xx_1: GEL Output: DDR3B initialization complete
CortexA15_1: GEL Output: Performing ARM boot cleanup...
CortexA15_1: GEL Output: Done with ARM boot cleanup.  ARM is in secure state and MMU is disabled.
```

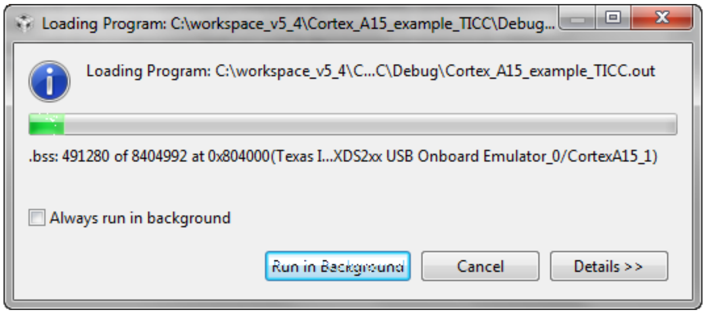
```
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_2 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_3 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_4 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_5 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_6 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_7 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_1 (Suspended)
    0x0C200058  (no symbols are defined for 0x0C200058)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_3 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_5 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_7 (Disconnected : Unknown)
```

Licensed | ARM | BE | MON | Async | SEC on | MMU Off

Then you can load your project executable on ARM Core0:



Program file C:\workspace_v5_4\Cortex_A15_example_TICC\Debug\C

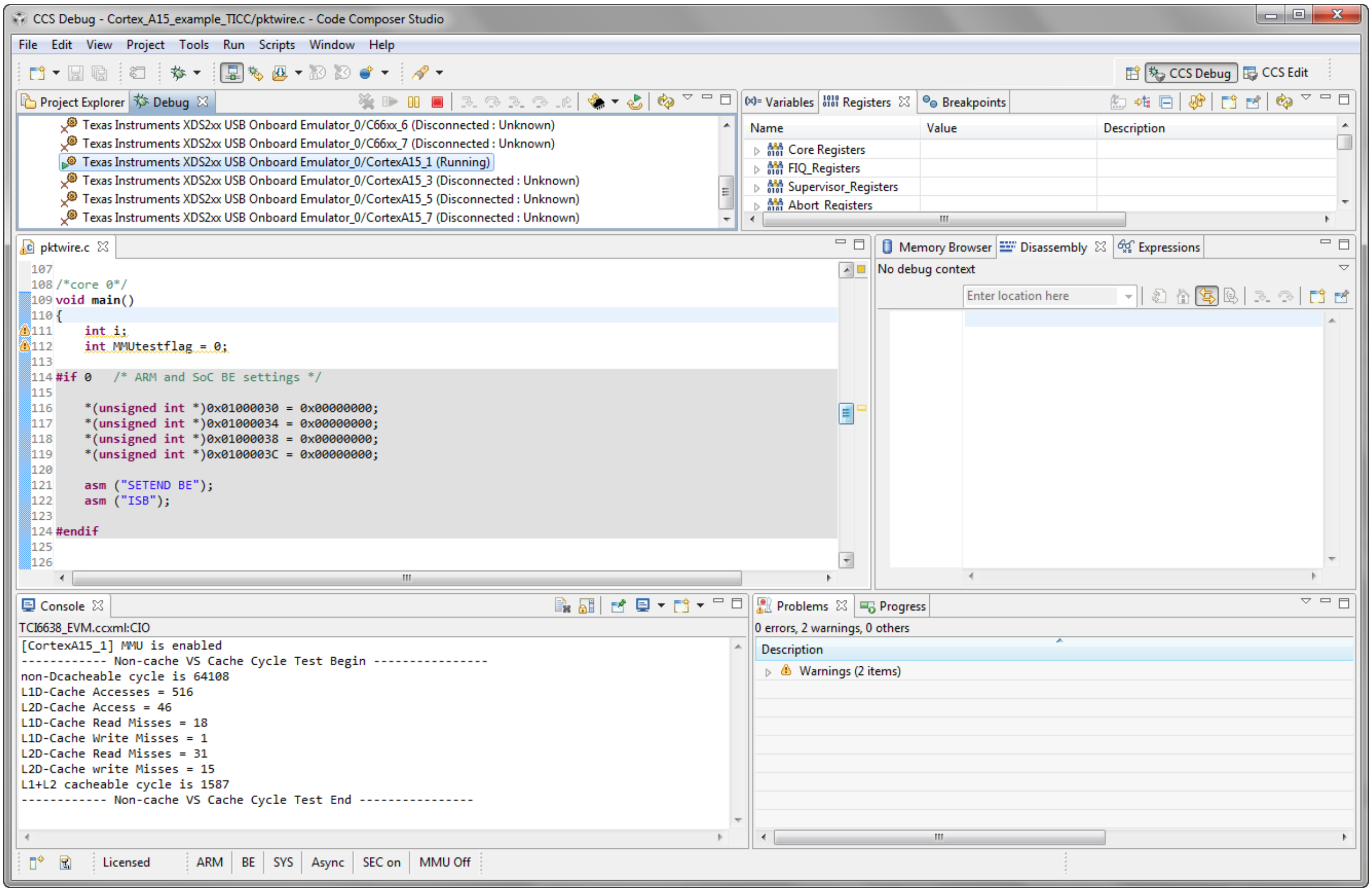

Loading Program: C:\workspace_v5_4\Cortex_A15_example_TICC\Debug...
Run in Background
Cancel
Details >>

Loading Program: C:\workspace_v5_4\C...C\Debug\Cortex_A15_example_TICC.out

.bss: 491280 of 8404992 at 0x804000(Texas I...XDS2xx USB Onboard Emulator_0/CortexA15_1)

☐ Always run in background

Click “Run” to execute the executable on ARM Core0, you can see the project result on Console window:



Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_6 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/C66xx_7 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_1 (Running)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_3 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_5 (Disconnected : Unknown)
Texas Instruments XDS2xx USB Onboard Emulator_0/CortexA15_7 (Disconnected : Unknown)

```

/*core 0*/
void main()
{
    int i;
    int MMUtestflag = 0;

#if 0   /* ARM and SoC BE settings */

    *(unsigned int *)0x01000030 = 0x00000000;
    *(unsigned int *)0x01000034 = 0x00000000;
    *(unsigned int *)0x01000038 = 0x00000000;
    *(unsigned int *)0x0100003C = 0x00000000;

    asm ("SETEND BE");
    asm ("ISB");

#endif


```

TCI6638_EVM.ccxml:CIO

```
[CortexA15_1] MMU is enabled
------------ Non-cache VS Cache Cycle Test Begin ----------------
non-Dcacheable cycle is 64108
L1D-Cache Accesses = 516
L2D-Cache Access = 46
L1D-Cache Read Misses = 18
L1D-Cache Write Misses = 1
L2D-Cache Read Misses = 31
L2D-Cache write Misses = 15
L1+L2 cacheable cycle is 1587
------------ Non-cache VS Cache Cycle Test End ----------------
```

Appendix. TCI6638 EVM Boot Mode DIP and BMC Encoding

A.1 TCI6638 EVM DIP Switch Reference

These are the bootmodes for the Phase 2 BMC (1.0.1.3[a])

DIPSW	Title	Boot Value(hex)	Comments
0000	Default	0x00110CE7	This setting should be the default bootmode for each board. For K2X, this is expected to be (and is currently) ARM NAND.
0001	JTAG	0x00100001	This setting should be the default JTAG mode for each mode. This mode is sleep w/o no PLLs.
0010	ARM SPI	0x00112005	(These will be DSP on a no-ARM device)
0011	ARM I2C	0x00100003	""
0100	ARM UART	0x00100DEF	"" NOTE: this value is currently incorrect in BMC v1.0.1.2 and will be fixed in v1.0.1.3 and up. (Correct value: 0x00000CEF)
0101	ARM RBL ENET	0x00111CEB	"", SPI or NAND bootloader will also have ENET boot modes
0110	Reserved	0x001010E1	Right now this is Sleep with Max speed SYS PLL and ARM Bypass. We could use it for 2nd I2C or 2nd SPI config. We could also use it for ARM NAND even for boards where ARM NAND is not the default.
0111	PWR OFF	0x00103EE1	Wait with power off until told what to do by BMC console or AMC MMC command (Right now this is Sleep with Max speed PLLs)
1000	Reserved	0x001101E7	These should be used for alternate configurations of the default bootloader (in NAND or SPI) Right now this is DSP NAND
1001	Reserved	0x001010C1	"" Right now this is Sleep with Slow Sys PLL and Bypass ARM PLL
1010	Reserved	0x00112105	"" Right now this is DSP SPI

DIPSW	Title	Boot Value(hex)	Comments
1011	Reserved	0x00100103	"" Right now this is DSP I2C
1100	User Programmable	0x00100DEF	Right now this is DSP UART
1101	User Programmable	0x001111EB	Right now this is DSP RBL ENET
1110	User Programmable	0x00103CC1	Right now this is Sleep w/ slow SYS PLL and slow ARM PLL
1111	User Programmable	N/A	Right now this is HW DBG / BMC Phase 1 mode, this mode will go away to be replaced by commands to enter hardware debug mode.

A.2 TCI6638 EVM BMC Encoding Reference

The BMC uses an extended boot mode value to encode all the boot configuration values into one 32 bit value. The lower 17 bits of this value matches the 17bit values used in the boot mode description tables.

(Note: Bit 0 value of 0 gives big endian for DSP, and 1 gives little endian for DSP. Also note that the ARMENDIAN bit is opposite of this. ARMENDIAN should always be 0; if ARM big endian operation is desired, the processors endian bit should be changed in SW.)

Bit	Devstat Bit	Config Pin Function	Normal Pin Function	Comments
31	na	na	na	reserved for wait in power off
30	na	na	na	reserved for wait in reset
29	na	na	na	reserved for other BMC SW config
28	na	na	na	reserved for other BMC SW config
27				Reserved for future constant drive config bits
26				Reserved for future constant drive config bits
25		PACLKSEL	PACLKSEL	
24		CORECLKSEL	CORECLKSEL	

Bit	Devstat Bit	Config Pin Function	Normal Pin Function	Comments
23				Reserved for future boot config latched values
22		AVSIFSEL1	TIMI1	Reserved: EVM forces these bits to strap values during reset
21		AVSIFSEL0	TIMI0	""
20		DDR3_REMAP_EN	GPIO16	
19		ARM_LENDIAN	GPIO15	0 = little, 1 = is not supported; do in SW
18		MAINPLLODSEL	GPIO14	
17		ARMAVSSHARED	CORESEL3	
16	16	BOOTMODE15	CORESEL2	
15	15	BOOTMODE14	CORESEL1	
14	14	BOOTMODE13	CORESEL0	
13	13	BOOTMODE12	GPIO13	
12	12	BOOTMODE11	GPIO12	
11	11	BOOTMODE10	GPIO11	
10	10	BOOTMODE9	GPIO10	
9	9	BOOTMODE8	GPIO9	
8	8	BOOTMODE7	GPIO8	
7	7	BOOTMODE6	GPIO7	
6	6	BOOTMODE5	GPIO6	
5	5	BOOTMODE4	GPIO5	
4	4	BOOTMODE3	GPIO4	
3	3	BOOTMODE2	GPIO3	
2	2	BOOTMODE1	GPIO2	
1	1	BOOTMODE0	GPIO1	
0	0	LENDIAN	GPIO0	

To run a custom boot mode value not available on the dip switch, do the following:

```
BMC> setboot <relative boot mode encodings>
```

```
BMC> fullrst
```