Booting from the SPI NOR on C6670/C6678 EVM.

Step 1 :-

The Application Code that has to be flashed on to the SPI Nor , has to be first converted into a Boot Table Format, using the hex6x utility present in CCS installation folder. By default it should be present in the following folder (C:\ti\ccsv5\tools\compiler\c6000_7.4.2\bin).

Way to use the command is as below :-

```
C:\Users\a0132217>cd C:\ti\ccsv5\tools\compiler\c6000_7.4.2\bin

C:\ti\ccsv5\tools\compiler\c6000_7.4.2\bin>hex6x.exe C:\Users\a0132217\CDOT_ROM_Bootloader\ROM_Bootloader\led_play.rmd
Translating to ASCII-Hex format...
   "C:\Users\a0132217\CDOT_ROM_Bootloader\ROM_Bootloader\led_play.out"   ::> .text     (BOOT LOAD)
   "C:\Users\a0132217\CDOT_ROM_Bootloader\ROM_Bootloader\led_play.out"   ::> .cinit    (BOOT LOAD)
   "C:\Users\a0132217\CDOT_ROM_Bootloader\ROM_Bootloader\led_play.out"   ::> .const    (BOOT LOAD)
   "C:\Users\a0132217\CDOT_ROM_Bootloader\ROM_Bootloader\led_play.out"   ::> .switch   (BOOT LOAD)
```

The RMD file contains, few of the following information :-

a) The Application.out file that has to be flashed.
b) –a for the output hex format in ASCII
c) –e the entry point for the address, i.e _c_init00
d) Output file that contains the application.out in boottable format.
e) Memory sections with the MEM and ROW WIDTH.

Sample RMD used for the example :-

```
C:\Users\a0132217\CDOT_ROM_Bootloader\ROM_Bootloader\led_play.out
-a
-boot
-e _c_int00

ROMS
{
    ROM1:  org = 0x0C000000, length = 0x100000, memwidth = 32, romwidth = 32
    files = { C:\Users\a0132217\CDOT_ROM_Bootloader\ROM_Bootloader\led_play.btbl }
}
```

Step 2:-

From the generated output in previous step which is in the boot table format convert it into the i2c/spi format by passing through the b2i2c. i.e. The byte-aligned boot table is then divided into 0x80 byte blocks and appended with length and checksum to adhere to the format required by the RBL, this is generated by passing through the b2i2c utility.

The b2i2c utility is part of the MCSDK installation and present in the following folder.

mcsdk_2_01_02_06\tools\boot_loader\ibl\src\util\btoccs

Way to run the b2i2c utility is as follows:-

```
C:\ti\mcsdk_2_01_02_06\tools\boot_loader\ibl\src\util\btoccs>b2i2c.exe C:\Users\a0132217\CDOT_ROM_Bootloader\ROM_Bootloader\led_play.btbl C:\Users\a0132217\CDOT
_ROM_Bootloader\ROM_Bootloader\led_play.btbl.i2c
```

Step 3:-

The output of the previous step, i2c/spi based format of the application table convert it into the CCS acceptable .dat format using b2ccs utility present in the mscdk\tools\boot_loader\ibl\src\util\btoccs.

Run the utility from the command prompt as below :-

C:\ti\mcsdk_2_01_02_06\tools\boot_loader\ibl\src\util\btoccs>b2ccs.exe
C:\Users\a0132217\CDOT_ROM_Bootloader\ROM_Bootloader\led_play.btbl.i2c
C:\Users\a0132217\CDOT_ROM_Bootloader\ROM_Bootloader\led_play.i2c.ccs

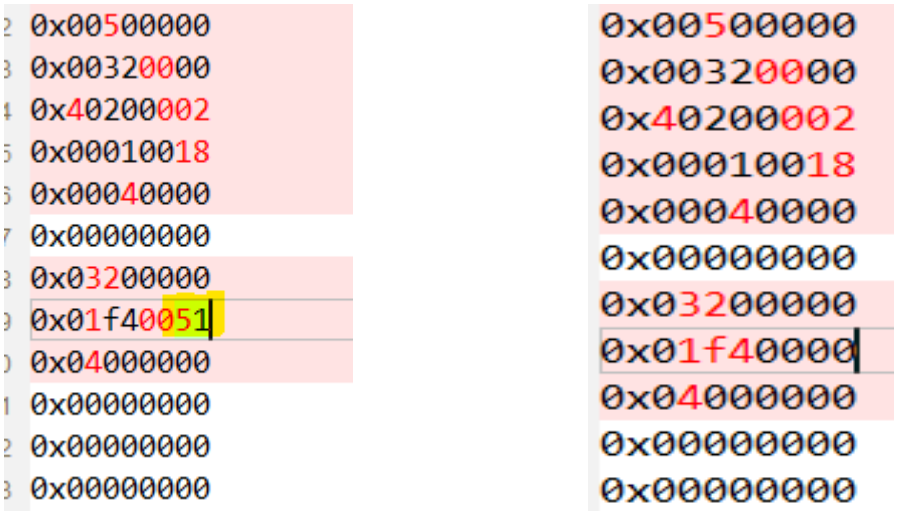
Step 4:-

Using the ROM PARSE utility present in mcsdk/tools/boot_loader/ibl/src/util/romparse to merge the boot table and boot parameter table. The *.map file which acts as the input to the romparse utility contains the following information.

a) The Input Application file that has to be flashed into the NOR in CCS acceptable dat format.
b) Boot configurations for SPI boot mode , containing the configurations details. Example in the zip file attached.

```
C:\ti\mcsdk_2_01_02_06\tools\boot_loader\ibl\src\util\romparse>romparse.exe C:\Users\a0132217\CDOT_ROM_Bootloader\ROM_Bootloader\nysh.spi.map
```

The output i2crom.ccs will be generated in the ROM PARSE directory. In the output i2crom.ccs please change the 51 marked in the below into 00.



Snap shoot shown above.

Step 5:-

The output of the above format has to be converted into the big endian format using the byte swap application present part of the document(RBL always works in BIG ENDIAN MODE). The byte swap application has to be used if the end target is little endian and the orginal application that has to be flashed is compiled in LE mode.

Byteswapccs.c has been attached part of the mail and can be used to convert the application from little endian to big endian format.

Create an EXE from byteswapccs.c and run it with the input file as the **i2crom.ccs** geenrated at previous step and output as the **application.dat** that will be falshed to the NOR.

Step 6:- Flashing using the NOR writer on EVM.

The output at this stage is a .dat file which has to flashed into the NOR using the CCS NOR writer. Provided part of the mcsdk folder. mcsdk_2_01_02_06\tools\writer\nor\evmc6670l

The EVM uses 24 bit NOR connected on CHIP select 0.

While flashing using the NOR writer use the **.dat as is and don't convert the .dat into a bin file**.

Rebuild then Nor write present in the MCSDK folder , with the following change to the nor_writer_input.txt , replace the file name from app.bin to the file name(application.dat) that was generated in step 5.

Once the NOR is flashed correctly . Boot the EVM in the SPI boot mode with following swtich settings.

Switch positionss:	4	3	2	1
SW3 :	1	1	0	1
SW4 :	0	0	0	0
SW5 :	0	1	0	0
SW6 :	0	0	0	1

Details of the switch settings/meaning of each of this bit can be found in the C6670 data sheet under SPI boot mode.

Attached below contains the output for all the stages described above. It is for an example to blink the leds present on the EVM.

SPI_Bootloader.zip