TI BLE Stack CC26xx R2 ROM Build Guide

August 17, 2016

# Setup

* pull latest ble\_core, ble\_examples, lprf-sd-common-components, and lprf-sd-rom
* download and install:

<http://www.sanb.design.ti.com/tisb_releases/TIRTOS/2_20_01_07/>

Note: Current BIOS is 6.46.01.29, which does not exclude certain driverlib calls from being in ROM. To fix this before the next 6.46.01.xx is released, do the following:

In: **ti/sysbios/rom/cortexm/cc26xx/r2/CC26xx\_extern.xs**

Comment out: **ROM.configDefs.$add("-DDRIVERLIB\_NOROM");**

* checkout the FPGA CC26xxware to C:\svn\cc26xxware:

<https://oslosvn.norway.design.ti.com/svn/satsw/projects/cc26xxware/trunk/r2_fpga_pg_1_0>

* checkout the FPGA software to C:\work\cc26xx:

<https://oslosvn.norway.design.ti.com/svn/srf04/cc26xx/trunk>

* for FPGA specific file changes, please see:

<https://spsemea.itg.ti.com/sites/lpw/proj/CC26xx/wikidigi/Wiki%20Pages/fpga_fw_mods.aspx>

# To Update ROM Image

* use workspace ble\_core\build\rom\r2\rom.eww to test-build the ROM image
* add/remove files as required
* update ROM Jump Table as necessary
* normally entire files are either included or excluded from ROM, but if you do need to change a function within a file:
  + wrap functions that stay in ROM with:

**#if defined( ROM\_BUILD ) || defined( FLASH\_ONLY\_BUILD )**

**#endif // ROM\_BUILD || FLASH\_ONLY\_BUILD**

* + wrap functions that don’t stay in ROM with:

**#ifndef ROM\_BUILD**

**#endif // !ROM\_BUILD**

* build and sort out compiler/linker errors.
  + If function is in ROM and is called from other ROM function, that call needs MAP\_ and the function must be in the ROM JT as a ROM-to-ROM function call.
  + If the function is not in ROM and is called from ROM, then that call needs MAP\_ and the function must be in the ROM JT as a ROM-to-Flash function call.
  + If function is in ROM and is called from Flash, no MAP\_ needed.
  + If function is not in ROM and is called from Flash, no MAP\_ is needed.
  + Create extern for all functions built in ROM and put in header file.
  + Remove all static’s (except within function for auto variables) using “#define static”
* build FlashROM projects to ensure ROM JT and linked ROM symbols are okay
* build FlashOnly projects to ensure no missing defines in map\_direct.h

# To Generate ROM Image and patch bit file for FPGA

* open command line at ble\_core\build\rom
* run local\_paths.bat
* if the ROM JT was updated, then:
  + cd to .\tools\gen\_rom\_init
  + run gen\_rom\_innit\_r2.bat to regenerate the rom\_init.c file
  + cd ..\..
* build rom image and patch FPGA by running:
  + gen\_release.bat r2 R2\_sandiego
  + Note: FPGA related files are output to ble\_core\build\rom\release\_r2

# Prep for Release

* Note: All FPGA Stack FlashROM projects have been modified to link with:
  + lprf-sd-rom\ble\_rom\_releases\cc26xx\_r2\Final\_Release\ble\_r2.symbols
* Note: All FPGA App projects (FlashOnly or FlashROM) have been modified to link with:
  + lprf-sd-rom\ble\_rom\_releases\cc26xx\_r2\Final\_Release\common\_r2.symbols
* copy all files under ble\_core\build\rom\release\_r2 to:
  + lprf-sd-rom\ble\_rom\_releases\cc26xx\_r2\Final\_Release
* FPGA:
  + if you need CF cards for FPGA, generate ACE files (see below) and move to:
    - lprf-sd-rom\ble\_rom\_releases\cc26xx\_r2\Final\_Release\Xilinx
  + if you have Xilinx support on your PC, then just run:
    - ble\_core\build\rom\program\_fpgas.bat
* test build as needed (Host Test FlashOnly and FlashROM)
* commit/push all ble\_core and lprf-sd-rom changes to repo’s

# Generate FPGA ACE Files

* run iMPACT 64 bit
* create a new project
* select “Prepare a System ACE File”, and click OK
* click Next
* click Next
* fill in Name: cc26xxR2
* click Next
* select the first checkbox “Configuration Address 0”, and click Next
* click Finish
* click OK
* select ble\_core\build\rom\ cc26\_fpga\_top.bit, then click Open
* click No
* in the menu, select “Operations->Generate File…”
* wait for blue indicate that the generation was successful
* in the menu, select “File->Exit”
* Note: The file directory is located at: C:\Xilinx\14.5\LabTools\cc26xxR2.
* cd to C:\Xilinx\14.5\LabTools\cc26xxR2
* move xilinx.sys to parent directory
* CF card:
  + copy cc26xxR2 directory and xilinx.sys file to CF; eject card
* Note: Xilinx tools installed on notebook at FPGA station 2.

# Formatting CF Cards

The CF must be 2GB, and formatted correctly.

If you have a CF that is larger than 2GB, you can re-partition it using diskpart:

1. Put the compact flash into a USB read/writer, and attach it to a Windows computer.
2. Open a Command prompt window, and type diskpart to open the utility.
3. Type: list disk
4. Type: select disk X

Where X = the disk number assigned to your CF device.

1. Type: list volume
2. Type: select volume X

Where X = the volume number assigned to your CF device

1. Type: clean all

This will remove all partitions and data and can take some time to complete. You should see “Diskpart succeeded in cleaning the disk” when done.

1. Type: create partition primary
2. Type: shrink querymax

This will display amount by which the volume size can be reduced.

1. Type: shrink desired X

Where X = the desired size subtracted from the size listed by the shrink querymax command.

1. Type: exit the leave the utility.

For example, if you have a 16GB CF, step 9 might show 15389MB. You would subtract 2048MB from this to shrink by 13341MB. This would give you a 2GB partition. Now you can format using mkdosfs using the default values.

For example, if the removable driver is H:

mkdosfs H:

# Additional Notes

* The gen\_rom\_init\_r2.bat script scrubs the JT, sequences the indexes, processes meta-commands, and inserts the table into the rom\_init\_r2.template and generates as output git\ble\_core\src\rom\r2\rom\_init.c, which is built in FlashROM project.
* Future Scripts
  + Script to generate git\ble\_core\src\rom\map\_direct.h, then scrub all functions in ROM project to ensure all function calls are prepended with MAP\_.
  + Script to scrub ROM JT to ensure no entries are present that are not used by ROM code .
* For information on the release, please see:
  + TI\_BLE\_Stack\_CC26xx\_R2\_ROM\_Release\_Notes\_2016-08-12.docx under:
  + lprf-sd-rom\ble\_rom\_releases\cc26xx\_r2\Final\_Release
* The Host Test builds with the FPGA use a UART baud rate of 19200 for test stability.
* The HCI Test builds with the FPGA use a UART baud rate of 115200.
* If a CF card goes “bad”, i.e. when installed into an FPGA but when the LCD is black-boxed and the fan on high, run through the Formatting CF Cards section again.

# Known Issues

* There is a known issue when flashing a large image like Host Test whereby the device does not appear to boot (i.e. there is no HCI Tester response to the command Read BDADDR). When the problem is checked using the debugger, a fault has occurred. When the fault is investigated, corrupted flash is discovered (a series of all 0xFF’s where opcodes should be). The workaround for this is to re-boot the FPGA (power cycle), then flash the Stack, then flash the App with Ctrl-D (i.e. flash and connect debugger in one operation). Then run, wait a few seconds (takes some time to checksum verify the ROM code), then stop the debugger to verify it’s at WFI. Then run, and try the command. If the HCI Tester still does not respond, try:
  + Select menu View->Options
  + Select Port Connection
  + Select tab “None”
  + Click Apply
  + Select tab Serial Port (always verify port, baud, and flow control None while there)
  + Select OK

At this point, you should see the Read BDADDR respond. Then you can disconnect the debugger, and reset the FPGA (button press), and re-test Read BDADDR. Should be good to go!