1. Copy \Target\FR5739\_I2C folder and rename as \Target\FR5969\_I2C
2. Import project in CCS
3. Migrate MSPBoot
   1. Select the target configuration (in this case, BSLBased\_DualImg)
   2. Change device variant to MSP430FR5969
   3. Modify the desired .cmd file (in this case, lnk\_msp430FR5739\_I2C\_1\_5KB\_Dual\_Boot.cmd)
      1. Use perl script as a guide
      2. Modify manually if necessary, check datasheet for memory regions
      3. Rename file (in this case, lnk\_msp430FR5969\_I2C\_2KB\_Dual\_Boot.cmd)
   4. CCS will add a new .cmd file (lnk\_msp430fr5969.cmd). Delete that file.
   5. Modify device initialization
      1. Modify Clock\_init to configure CS (check datasheet or code examples)
      2. Modify MPU\_init to configure MPU (check datasheet or code examples and make sure you use the new memory assignment from the .cmd file)
      3. Modify HW\_Init as needed. We’ll change the button to use P1.1 (instead of P4.1) and FR5969 needs to unlock GPIO
   6. Modify CI
      1. The PHY\_DL layer can be modified as needed but in this case, we’ll use eUSCIB0 with P1[6:7] which is the same as FR5739
   7. Modify for addresses above 16bits:

This device has more than 16-bit memory, but the application always resides and executes below 16bits. Because of this, pointers can use 16-bits, except when writing the download area and validating the application

* + 1. Modify Flash routines
       1. Add GetPhysicalAddressFromVirtual to calculate download address from application address
       2. Modify TI\_MSPBoot\_MI\_EraseSector and TI\_MSPBoot\_MI\_WriteByte to calculate the physical address correctly
       3. Modify TI\_MSPBoot\_MI\_EraseSectorDirect and TI\_MSPBoot\_MI\_WriteByteDirect to use 32-bit addresses. Note that we can use intrinsics to write 32-bit addresses.
       4. Modify TI\_MSPBoot\_MI\_EraseSectorDirect and TI\_MSPBoot\_MI\_WriteByteDirect according to new MPU settings
       5. Modify TI\_MSPBoot\_MI\_EraseAppDirect to erase both download areas
       6. Modify TI\_MSPBoot\_MI\_ReplaceApp to write App area with both download areas
       7. Note that new constants can be added to the linker file as needed
  1. Modify CRC calculation
     1. Add crc16Init and crc16AddByte in order to calculate CRC for non-contiguous locations
     2. Modify TI\_MSPBoot\_AppMgr\_AppisValid to calculate CRC of app and download areas

1. Migrate Apps
   1. Select the target configuration (in this case, BSLBased\_DualImg)
   2. Change device variant to MSP430FR5969
   3. Modify the desired .cmd file (in this case, lnk\_msp430FR5739\_I2C\_1\_5KB\_Dual\_App.cmd.cmd)
      1. Use perl script as a guide
      2. Modify manually if necessary, check datasheet for memory regions
      3. Rename file (in this case, lnk\_msp430FR5969\_I2C\_2KB\_Dual\_App.cmd)
   4. CCS will add a new .cmd file (lnk\_msp430fr5969.cmd). Delete that file.
   5. Modify the app as needed
      1. Change port for LEDs
      2. Change port for Switch
      3. Change definition of Vectors for dummy\_isr
      4. Use cast for pointers of Boot vectors to remove warnings
2. Modify host
   1. G2 demo is used for host
   2. Add new target copying TARGET\_FR5739 to TARGET\_FR5969
   3. Replace definition of TARGET\_FR5739 -> TARGET\_FR5969
   4. Add new demo files for FR5969
      1. Use perl script 430txt2c.pl to convert files as shown in PrepareCCSOutput\_G2\_FR5969.bat
   5. Modify project to send more sections of application if needed (new App2 has 3 sections instead of 2)
3. Test
   1. Download G2Host to G2 Launchpad
   2. Download MSPBoot to FR5969 Launchpad
   3. If desired, debug MSPBoot
      1. to check if device is getting data from I2C (TI\_MSPBoot\_CI\_PHYDL\_Poll**)**
      2. If packets are decoded correctly (TI\_MSPBoot\_CI\_Process**)**
      3. If data is written correctly to Flash (TI\_MSPBoot\_MI\_FRAMDualImg.c)
      4. If CRC sent by host is the same as calculated (TI\_MSPBoot\_AppMgr\_AppisValid)
      5. If bootloader is jumping to App (TI\_MSPBoot\_APPMGR\_JUMPTOAPP)
   4. Run demo
      1. After reset, LED2 and LED1 are ON in FR5969 launchpad
      2. Press S2 in G2 launchpad, LED should blink twice and it will send App1
      3. FR5969’s LED2 will blink twice fast and then it will blink slowly running App1
      4. Pressing S2 in FR5969 launchpad forces bootloader mode
      5. Press S2 in G2 Launchpad again, LED should blink twice and it will send App2
      6. FR5969’s LED1 will blink twice. Pressing S2 will toggle LED1.
      7. Pressing S2 in G2 launchpad again will force bootloader mode in FR5969 (by sending a special command via I2C) and then it will send App1 again.