Loading Instruction Set on to uZed EEPROM

SDK = Software Design Kit

uZed = Avnet development kit consisting of Zynq SOC and peripherals.

Workspace = Local Directory consisting of the instructions for the FPGA and processor on the Zynq

Zynq = Xilinx System on a Chip. This is the name of a large family of chips, where we have utilized only the Z010 and Z020 chips.

BSP = ………………Board Support Package

1. Identify the instruction set for the current project.
   1. Note the Zynq Chip set Z010 or Z020. The chip set is explicitly linked to the project.
   2. Note the software version for the operations required.
   3. LunaH software runs with Xilinx 2014.1
   4. The instruction sets are deposited on GitHub
   5. The instruction sets consist of:
      1. Hardware platform (example: ../hw\_platform\_0/)
      2. Board support package (example: ../standalone\_bsp\_0/)
      3. Software package (example: ../Lunah\_DevKit\_v1.0\_2014\_1/)
2. Define the “workspace” on your local computer, which is downloading or copying the instruction set to a specified folder location.
   1. Go to github ([link](https://github.com/RMDInc/LunaH_FlightSys_Z020_SDK2014.1))and download the project “LunaH\_FlightSys\_Z020\_2014\_1”.
   2. If the project was downloaded, unzip/extract the folder into your workspace.
   3. To clone the project using github, follow the instructions in the “GitHub Basics” document.
3. Start Xilinx SDK (note to use the appropriate SDK version).
4. At the startup of the SDK, define the workspace as the folder location of your instruction set.
   1. Note: the source code may be a few folders below the main directory; make sure to navigate to the “DevKit” subfolder.
5. Import Files:
   1. File > Import > General > Existing projects into workspace
   2. Click next, then for “Select Root Directory”, browse to the workspace you defined
      1. Three projects should have their boxes ticked (hw\_platform\_0, Lunah\_DevKit\_v1.0\_2014\_1, standalone\_bsp\_0)
   3. Select Add projects to working sets
      1. The instruction sets should populate under projects.
      2. Make sure they are selected.
   4. Select “Add project to working sets”
   5. Click Finish
6. Close the Welcome Tab, the Project Explorer should be visible.
7. Solve the two errors after the SDK has loaded.
   1. In the Project Explorer Right Click on the Board Support Package (ex: standalone\_bsp\_0)
      1. Select Re-generate BSP sources
   2. In the Project Explorer Right click on Software (ex: Lunah\_DevKit\_v1.0\_2014\_1)
      1. Select Generate linker script > generate
         1. Note: ensure the output script will be placed into the workspace location.
   3. Click the triangle next to the BSP to view its subfolders.
   4. Navigate to: ps7\_cortexa9\_0 > libsrc > xilffs\_v2.0 > src >include > ffconf.h
   5. Double click to display ffconf.h in the main window
   6. Change two of the #define statements
      1. \_FS\_MINIMIZE 1 -> \_FS\_MINIMIZE 0
      2. \_USE\_LFN 0 -> \_USE\_LFN 1
   7. Save the file
      1. If SDK does not automatically rebuild the project, click on LunaH\_Devkit\_v1.0\_2014\_1 then go to the menu bar: Project > Build All
   8. There should be no errors, warnings, or infos in the “Problems” box at this point.
8. Make the First Stage Bootloader
   1. File > New > Application Project
   2. Type In Project Name (example: MZ\_FSBL)
   3. Make sure that the box “Use default location” is checked
   4. The target hardware is hw\_platform\_0
   5. The processor is ps7\_cortexa9\_0
   6. OS Platform is standalone
   7. Click Next
   8. Select Zynq FSBL
   9. Click Finish
9. The First Stage bootloader files will be generated (example: MZ\_FSBL and MZ\_FSBL\_bsp)
10. Build the FSBL and Software in Release mode
    1. Right Click on FSBL (example: MZ\_FSBL)
    2. Select Build Configurations > Set Active > Release
    3. Right Click on Software (example: Lunah\_DevKit\_v1.0\_2014\_1)
    4. Select Build Configurations > Set Active > Release
11. Build the both the FSBL and Software projects by clicking on the Hammer Icon for each.
12. Create the Bootimage:
    1. Xilinx Tools > Create Zynq Boot Image
    2. Create New BIF file
       1. Browse to the Software folder (example: Lunah\_DevKit\_v1.0\_2014\_1)
       2. Create New Folder called “bootimage”
       3. Set filename to bootimage.bif in that folder
       4. Under boot image partitions add:
          1. Delete any existing file paths
          2. Click Add > Select FSBL.elf file (ex: MZ\_FSBL.elf) from the release folder of FSBL> Set the Partition type as “bootloader”
          3. Click Add > Select bitstream from the hardware platform   
             (example: ..\hw\_platform\_0\design\_1\_wrapper.bit) >   
             Set the Partition type as “datafile”
          4. Click Add > Select the software release   
             (example: Lunah\_DevKit\_v1.0\_2014\_1\Release\  
             Lunah\_DevKit\_v1.0\_2014\_1.elf) >   
             Set the Partition type as “datafile”
          5. Set the output path as BOOT.mcs in the bootimage folder created above.
13. Program the EEPROM
    1. Connect the board (uZed) to the computer via the JTag cable (Digilent JTAG-HS3 Rev. A)
    2. Select Xilinx Tools > Program Flash
    3. Set Image File as BOOT.mcs generate in previous step.
    4. Click Program
    5. Note the comments in console window, and the operation is completed and successful with it reports “Flash Operation Successful”
    6. A red LED will be lit on the uZed board.
    7. Power cycle the board or press the reset button
    8. A blue LED will be lit if the system boot was successful. The system is now prepared.

**To run the system in debug mode:**

1. Build the FSBL and software in debug mode
   1. Right Click on FSBL (example: MZ\_FSBL)
   2. Select Build Configurations > Set Active > Debug
   3. Right Click on Software (example: Lunah\_DevKit\_v1.0\_2014\_1)
   4. Select Build Configurations > Set Active > Debug
2. Create a debug configuration
   1. Run > Debug Configurations…
   2. Right click on Xilinx C/C++ application (GDB) from the left box and choose New
   3. For the bitstream file, browse to the \DevKit folder in your workspace, then to \hw\_platform\_0 and design\_1\_wrapper.bit should be your selection.
   4. Go to the Application tab and across from Project Name, click browse and select Lunah\_Devkit\_v1.0\_2014\_1
   5. For Application, click browse and navigate to the DevKit\Lunah\_Devkit\_v1.0\_2014\_1\Release folder where Lunah\_Devkit\_v1.0\_2014\_1.elf should be. Select this file and click open.
   6. The option to change the name of the configuration at the top of the box, this can be useful for future debug sessions.
   7. Click Apply then Debug to begin the session.
      1. As a shortcut, once the debug configuration has been created, press the “Bug” button in the toolbar to launch the currently selected debug configuration.
   8. As a default, SDK will stop at main(), so in the toolbar near the top of the screen press the green ‘Resume’ button (F8) to start the debug session.