Instructions for loading SINGR project onto a uZ board

1. Download a new version of the SINGR DevKit from GitHub:
   1. <https://github.com/RMDInc/SINGR>
   2. Click the green “Clone or Download” button on the right side of the page to get a new version of the project.
   3. Define the “workspace” to keep the project in and move this new version of the DevKit there.
2. Start SDK at the location of the “workspace”.
3. Import files:
   1. File > Import > General folder > Existing projects into workspace
   2. Click next, then browse to the workspace you defined
   3. Select Add projects to working sets
      1. The instruction sets should populate under projects.
      2. Make sure they are selected.
   4. Select “Copy projects into workspace”
   5. Select “Add project to working sets”
   6. Click Finish
   7. SDK will briefly build the workspace.
4. Close the “Welcome” tab and the “Project Explorer should be visible.
5. In the Project Explorer Right click on Software (ex: Lunah\_DevKit\_v1.0\_2015\_3)
   1. Select Generate linker script > generate
6. Bring in the new Hardware Platform files:
   1. In SDK, right click on the “hw\_platform\_x” folder and select delete to remove the files from the project. If the name is the same for the new hardware platform files, then check the box for “delete contents on disk”, so that SDK will not run into naming collisions.
   2. Copy the new hardware platform files into the workspace that you are using under the DevKit/ folder. They will show up in windows explorer, but not in SDK, yet.
   3. File > Import > General folder > Existing projects into workspace
   4. Click next, then browse to the workspace you defined
   5. If the hw\_platform\_x folder you want to import is checked, click finish and move on to step 8.
   6. If it shows up in the list, but is not checked and is instead greyed out we have to work around a small bug:
      1. Uncheck copy projects into workspace
      2. Click add project to working sets
      3. At the top of the window, choose Select archive file
      4. Then click back to select root directory
      5. The hw\_platform\_x folder should now be selected > click finish
7. Change the hardware specification file:
   1. Right click on the hw\_platform\_x folder > change hardware specification > click yes
   2. Click browse and navigate to the “system.hdf” file in the hw\_platform\_x folder > click ok
8. Close the “standalone\_bsp\_0” folder and create a new board support package
   1. Right click on the “standalone\_bsp\_0” folder > close project
   2. File > New > Board Support Package
   3. SDK will name the new bsp sequentially with the old one
   4. Check to make sure the hardware platform shown is the new version that was imported
   5. Check that the Board Support Package OS is “standalone”
   6. Click finish
   7. A window will pop up allowing you to change the board support package settings
   8. Select the boxes next to
      1. Lwip140
      2. Xilffs
      3. Xilrsa
   9. Click ok
9. Right click on the “Echo3” project > Change referenced BSP
   1. Choose the new bsp that we created > Click ok
10. There will be errors in the Echo3 project
    1. Expand the folder Echo3 to show its subfolders and other files
    2. Expand the subfolder src
    3. Double click on echo.c
    4. Comment out lines 123-128, 131, 132, 135-143
    5. Save the project
    6. SDK will rebuild the project automatically when it is saved, so the build will happen and the errors should disappear
11. Make the First Stage Bootloader
    1. File > New > Application Project
    2. Type In Project Name (example: MZ\_FSBL)
    3. Click Next
    4. Select Zynq FSBL
    5. Click Finish
12. The First Stage bootloader files will be generated (example: MZ\_FSBL and MZ\_FSBL\_bsp)
13. If the project is going to be run in debug mode follow the next instructions, otherwise go to step 14. Set FSBL and Echo3 for debug
    1. Right Click on FSBL (example: MZ\_FSBL)
    2. Select Build Configurations > Set Active > Debug
    3. Right Click on Software (example: Echo3)
    4. Select Build Configurations > Set Active > Debug
    5. Project > Build All
    6. Xilinx Tools > Program FPGA
    7. For the bitstream textbox, ensure that the correct “x\_wrapper.bit” file is selected > Program
       1. The Blue LED on the uZ board will turn off while the FPGA is being programmed
    8. Run > Debug Configurations…
    9. There will be a list of debuggers, from the list choose Xilinx C/C++ application (GDB)
    10. Click the “New” button to create a configuration of that type
    11. Ensure that the hardware platform shown is the correct one for the project
    12. For Bitstream File, browse the the bitstream that is found in the hw\_platform\_x folder
    13. Click the Application tab
    14. Click browse next to Project Name and select Echo3
    15. For Application: Debug/Echo3.elf will be automatically selected
    16. Click Apply > Debug
    17. SDK will ask you if you want to switch perspectives, choose yes
    18. Along the toolbar at the top of the screen there will be buttons related to debugging that will illuminate
    19. When the system has loaded, click Resume (the play button) to start
14. Set FSBL and Software for Release
    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