**Software versions for Luna H-Map**

**V1.0:** This is the version found on GitHub as of 1/30/2017. The software files on GitHub are currently:

* Lunah\_DevKit\_v1.0
* Design\_1\_wrapper\_hw\_platform\_0\_v1.0
* Standalone\_bsp\_0
* .gitignore
* Readme.md
* repositoryDownloadInstructionsv2.pdf

Going down the list, the first three items are folders for the software (SW) that runs on the uZed board. The first of these, Lunah\_DevKit (devkit) contains the source and header files for the FPGA logic, as well as the terminal interface for testing and acquiring data with the uZed board. The important files in this folder are src\LApp.c, and src\LApp.h. These contain the code for the aforementioned functions.

The next item, hw\_platform\_0 specifies the hardware platform of the uZed board so that we can interface properly with the hardware.

Next, standalone\_bsp\_0 is a Xilinx-generated set of files called a board support package. It is a simple, low-level software layer that provides access to basic processor features, as well as the basic features of a hosted environment.

The .gitignore is a GitHub specific file that allows us to control which files in a folder are under version control and which files Git will merely ignore. Files in the .gitignore file are often ones that should be generated (or regenerated) each time a new clone of the repository is created. This includes the first-stage-bootloader (FSBL) and possibly the standalone\_bsp.

The readme file is one which is auto-generated by Xiliinx and describes some of the files in the project, specifically detailing the ones it has created when the project was generated.

The repository download instructions are self-explanatory, but please reach out if they are out-of-date or if something is not working as detailed.

Details for the SW

After powering on the uZed board with the software loaded, the board will boot from the SW. Open Teraterm and connect via serial port. If a menu does not appear, press any key and a response will be generated. The user should then be presented with a menu of options like the following:

\*insert screenshot of teraterm menu\*

The 0th option is to set the running mode of operation for the uZed board. When selected the user will be presented with another set of options, they are:

* 0 – Waveform data: This will dump the digitized pulses (waveforms) from the detector; this is raw data.
* 1 – LPF Waveform data: this is a processed waveform, where LPF = low pass filter. Not currently enabled, but would dump raw waveform data modified by a custom filter.
* 2 – DFF Waveform data: another processed waveform; raw waveform modified by a different filter.
* 3 – TRG Waveform data: the waveform of the trigger signal; it is processed from the raw waveform.
* 4 – Processed data: this dumps the processed integral data. The data will show up on the terminal in a format shown below:
  + 0: ID (111111)
  + 1: Time (262.144 us per tick)
  + 2: Total Events
  + 3: Event number
  + 4: Baseline integral
  + 5: Short Integral
  + 6: Long Integral
  + 7: Full Integral

V1.1:

**V1.2**: This version has SD card functionality fully implemented. The software will take in a buffer of data from the dram, process it, and save it to a file on the SD card. The process will loop until the user enters a ‘q’ into Teraterm to exit the loop. The SD card file is a binary file which must be converted, so I have written a converter program in C++ to do that. It’s called bin\_to\_txt located on the shared drive here:

K:\users\GStoddard\Miscellany\bin\_to\_txt

The program asks the user for a filename, converts a file named “userfilename.bin” to “userfilename.txt”, then prompts the user for another file to convert. The functionality will be improved at a later time to add a way for the user to enter the size of the file to be converted and a way to search through multiple folders for the relevant file.

**V1.3:** current issues are: finding a sleep time that is optimal without affecting the data. The sleep command can only take integer numbers as input, so that command ‘usleep()’ is being tested. Usleep takes a time in microseconds as input, giving more flexibility as to how long the designated sleep takes place.

Adding a version header for the SW overall; currently this is just a line that is added to the ‘Main Menu’ print statement saying what version it is. Perhaps in the future I can have the SW check for a global variable which is stored somewhere.

A large issue is the calculation of the baseline integral (BL). Every fourth event after ~30 events has a BL value equal to the short integral value. The two values are exactly equal to each other for an unknown reason. This compromises the values of the processed integrals and makes the data useless. The SW is currently set to only print out data from the dram and this pattern is found there.

* This pattern is not seen:
  + In the data stream from older versions of the SW
* This pattern is seen:
  + Using this version of the SW with sleep or usleep functions
  + Using this version of the SW with sleep(1), or usleep(500000)
  + Using just the PrintData() module
  + Using just the ReadDataIn() module
  + For multiple sets of integral stop time (-52, 88, 472, 6000; default)

**V1.4:**  Sam has updated the files in hw\_platform\_0 and the bug with duplicate data in the data stream has been fixed. There was a timing problem that he found that was fixed somewhere in those files.