

ECE 544 Final Project

RVfpga-SoC Curriculum Testing

...

Spring 2021
Chuck Faber and Brett Thornhill

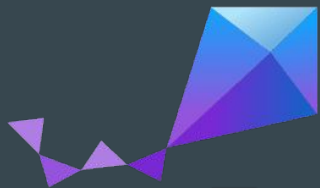
Project Overview

- Test RVfpga-SoC curriculum
- Find potential issues
- Provide feedback
- Test viability of using RVfpga-SoC for ECE 544
- Stretch goals
 - Implement PMOD and customize hardware
 - Create and run a demo program using Zephyr RTOS
 - Implement project 2 on RVfpga/Zephyr setup



RVfpga-SoC Introduction

- RVfpga-SoC
 - Series of 5 labs
 - Build a subset of SweRVolf-SoC from scratch
 - Using Vivado GUI
 - Using FuseSoC
 - Load Zephyr real-time operating system onto SweRVolf
 - Run Tensorflow Lite on top of the operating system



Zephyr™ Project



TensorFlow Lite

Our Testing Process

Both of us independently ran through all five labs and made comments on Lab Instructions.

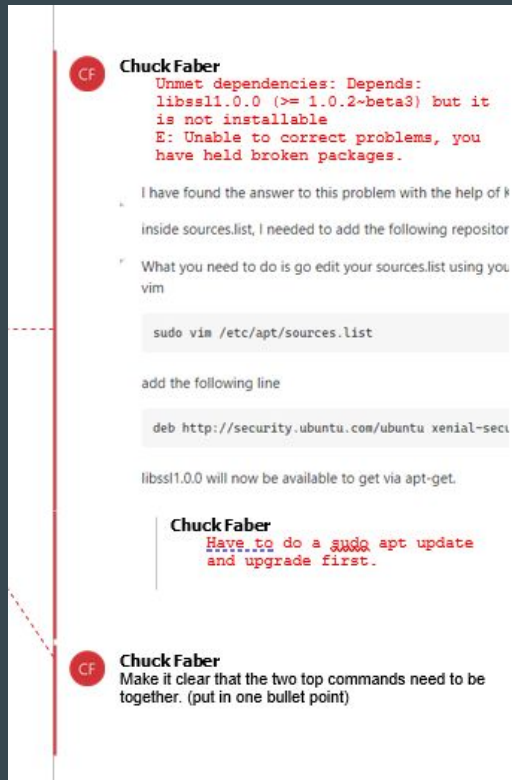
Chuck attempted to implement a PMOD at the end.

Brett attempted to customize a ZephyrRTOS program at the end.

We both ran on an Ubuntu VM (Chuck: 20.04, Brett: 18.04).

VM Specs: Used VirtualBox (Familiarity with it from ECE 373 LDD). 80 GB hard-drive, and 8GB memory needed minimum (to install and run Vivado). Chuck gave his 4 cores and Brett gave hers 2 cores.

Note: Ubuntu is *required* to run Lab 4 (ZephyrRTOS) and Lab 5 (TensorFlow) according to installation guide.



Installation Guide

- Mostly clear guidance
- More explanation for each command line would have been helpful
- Prior experience with Linux seems to be a requirement

Open your Ubuntu terminal and enter the following [commands](#) :

1. Prerequisites and Dependencies:

```
❑ sudo apt update
❑ sudo apt upgrade
❑ sudo apt install --no-install-recommends git cmake
  ninja-build gperf \
❑ ccache dfu-util device-tree-compiler wget \
❑ python3-dev python3-pip python3-setuptools python3-tk
  python3-wheel xz-utils file \
❑ make gcc gcc-multilib g++-multilib libsdl2-dev
```

2. Update ***cmake*** version to 3.13.1 or higher:

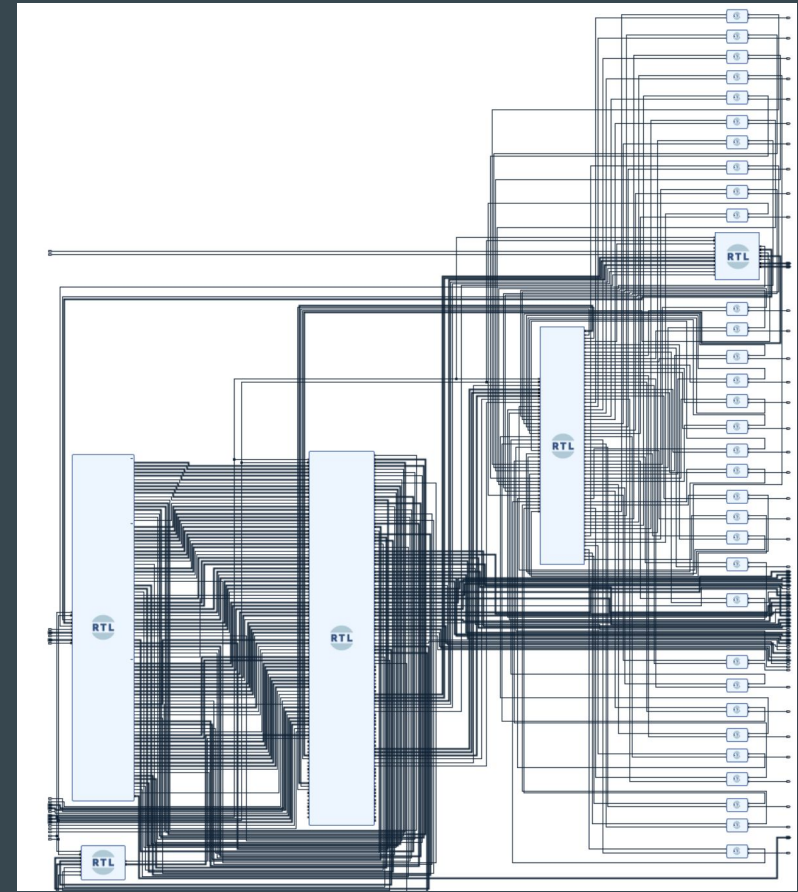
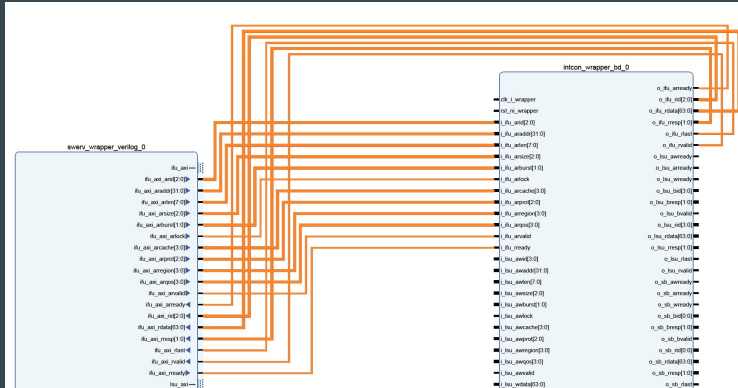
```
❑ wget -O - https://apt.kitware.com/keys/kitware-
  archive-latest.asc 2>/dev/null ||
❑ sudo apt-key add -
❑ sudo apt-add-repository 'deb
  https://apt.kitware.com/ubuntu/ bionic main'
❑ sudo apt update
❑ sudo apt install cmake
```

Example: The bullet points make it seem like separate commands here, but they are continuations of the commands before them. This syntax makes sense with experience with Linux but may be confusing to those unfamiliar.

<- This one in particular needs to be run this way for it to work.

Lab 1: Introduction to RVfpga-Soc

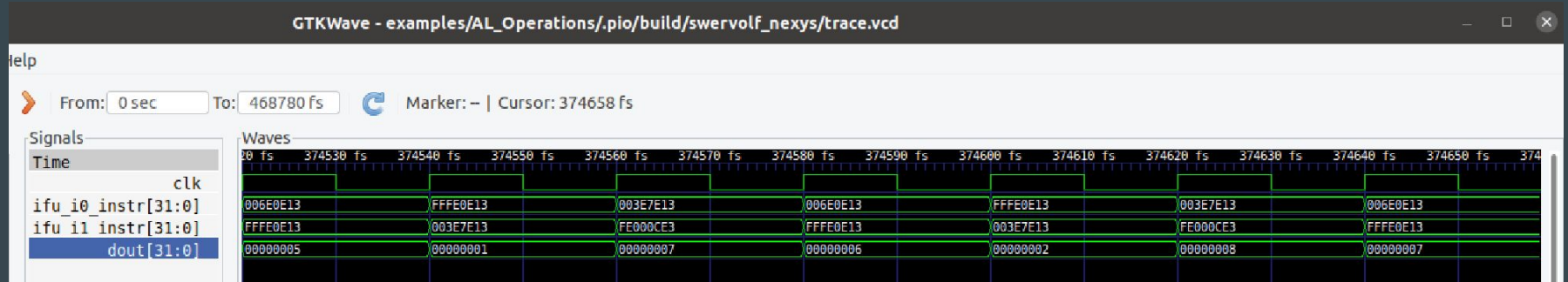
- Extremely tedious
- Very long to complete
- Connection automator should have been used
- PDF block diagrams were very helpful



^ This all had to be done manually, wire by wire.

Lab 2: Running Software on Rvfpga-SoC

- Straightforward and easy to complete if lab 1 done correctly
- GTK wave should be emphasized more in ECE 540 and 544
- No opportunities for expansion on sample program (blinky)



Lab 3: Introduction to SweRVolf and FuseSoC

- Replaces all the manual connection work done in lab 1
- Defeats the purpose of using Vivado's IP Integrator
- Again needs more ways to expand on the “Hello World”
- Wanted more instruction on using the OCD (Open On Chip Debugger)

```
chuck@Dusky-Darjeeling:~/ece544_final_project/chuck/RVfpgaSoC-May21_2021/RVfpgaSoC/Labs/LabResources/SweRVolf/fusesoc_libraries/swervolf/data$ telnet localhost 4444
```

```
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^['.
Open On-Chip Debugger
> mwb 0x80001010 1
> mwb 0x80001010 0
>
```

```
INFO: Running
INFO: Running simulation
Starting jtag_vpi server: interface 127.0.0.1 (loopback), port 5555/tcp ...
jtag_vpi server created.
Waiting for client connection...
Client connection accepted.
JTAG VPI enabled. Not loading RAM
Releasing reset
20936880: gpio0 is on
31060880: gpio0 is off
```


Lab 4: Running Zephyr on SweRVolf

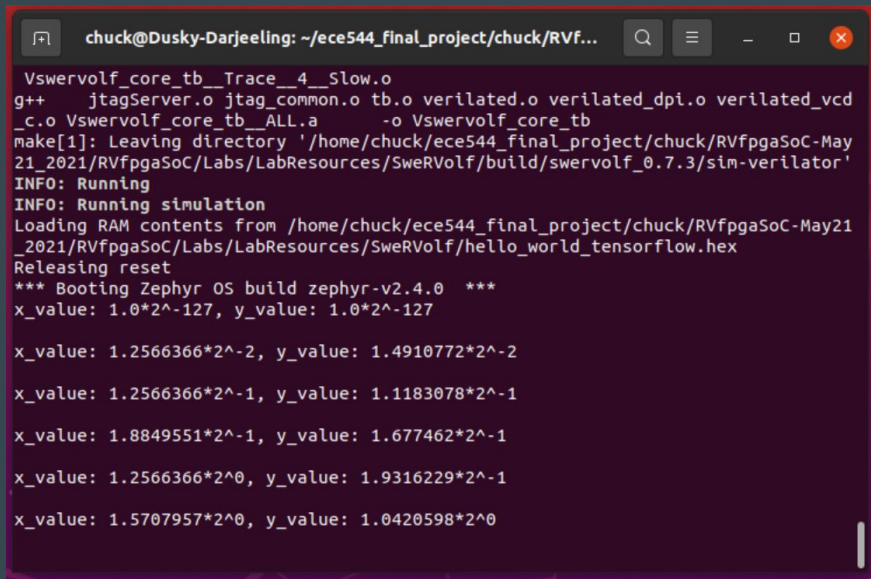
- Ran into issues in this lab that required redoing part of lab 3
- Unsure why redoing part of lab 3 solved the problem
- Quite a few typos and errors in the instructions
- Again wanted more 'next steps' after the sample program

```
chuck@Dusky-Darjeeling:~/ece544_final_project/chuck/RVfpgaSoC-May21_2021/RVfpgaSoC/Labs/LabResources/SweRVolf$ sudo putty
[sudo] password for chuck:

(chuck@172.17.0.13:~) ssh -p 2222 root@172.17.0.13
root@swervolf_nexys:~#
```

Lab 5: Running Tensorflow Lite on SweRVolf

- Straight-forward
- Wasn't clear what machine learning concepts were being demonstrated in the hello world application
- Again, would like to have had next steps



```
chuck@Dusky-Darjeeling: ~/ece544_final_project/chuck/RVf...  
Vswervolf_core_tb__Trace__4__Slow.o  
g++ jtagServer.o jtag_common.o tb.o verilated.o verilated_dpi.o verilated_vcd  
_c.o Vswervolf_core_tb_ALL.a -o Vswervolf_core_tb  
make[1]: Leaving directory '/home/chuck/ece544_final_project/chuck/RVfpgaSoC-May21_2021/RVfpgaSoC/Labs/LabResources/SweRVolf/build/swervolf_0.7.3/sim-verilator'  
INFO: Running  
INFO: Running simulation  
Loading RAM contents from /home/chuck/ece544_final_project/chuck/RVfpgaSoC-May21_2021/RVfpgaSoC/Labs/LabResources/SweRVolf/hello_world_tensorflow.hex  
Releasing reset  
*** Booting Zephyr OS build zephyr-v2.4.0 ***  
x_value: 1.0*2^-127, y_value: 1.0*2^-127  
  
x_value: 1.2566366*2^-2, y_value: 1.4910772*2^-2  
  
x_value: 1.2566366*2^-1, y_value: 1.1183078*2^-1  
  
x_value: 1.8849551*2^-1, y_value: 1.677462*2^-1  
  
x_value: 1.2566366*2^0, y_value: 1.9316229*2^-1  
  
x_value: 1.5707957*2^0, y_value: 1.0420598*2^0
```

Overall Impressions of RVfpga-SoC Curriculum

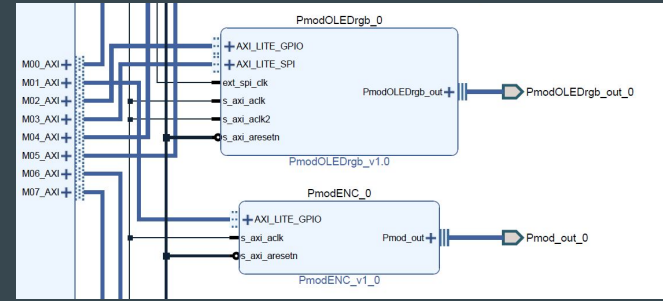
- Better than we expected for a beta version but needs some improvement
- Feels like one large “installation guide”.
- A good starting point but next steps need to be provided and how to extend the platforms we’ve just installed to be useable and customizable in projects.



Stretch Goals

Hardware Customization (implementing a PMOD)

- Attempted to implement, but ran into the issue that we couldn't just add new IP into the IP integrator like with the Microblaze.
- PMOD IP requires AXI bus interconnects. These aren't available on RVfpga SoC.
- Adding new PMODs would likely include the same method we used in ECE 540 of editing the verilog files (`wb_intercon.v/vh`, `swervolf_core.v`, `rvfpga.v`) directly bypassing the IP integrator entirely to connect them to the WishBone bus.
 - Again begs the question then, what is the point of using the IP Integrator with RVfpga?
- Guidance on how to implement new hardware is conspicuously absent in the RVfpga SoC curriculum.



Stretch Goals

Software Customization

- Running other sample programs required updating device tree files
- Included documentation was vague
- Recommend lab 4 cover this
- There were several sample files so the possibilities for customization was there

Requirements

The board hardware must have a push button connected via a GPIO pin. These are called "User buttons" on many of Zephyr's :ref:`boards`.

The button must be configured using the ``sw0`` :ref:`devicetree <dt-guide>` alias, usually in the :ref:`BOARD.dts file <devicetree-in-out-files>`. You will see this error if you try to build this sample for an unsupported board:

```
.. code-block:: none
```

```
Unsupported board: sw0 devicetree alias is not defined
```

```
../src/main.c: In function 'blink1':
../src/main.c:101:2: error: #error "Unsupported board: led1 devicetree alias is not defined"
  101 | #error "Unsupported board: led1 devicetree alias is not defined"
      |      ^~~~~
[19/110] Building C object zephyr/CMakeFiles/zephyr.dir/lib/os/printk.c.obj
ninja: build stopped: subcommand failed.
FATAL ERROR: command exited with status 1: /usr/bin/cmake --build /home/chuck/ec
e544_final_project/chuck/RVfpgaSoC-May21_2021/RVfpgaSoC/Labs/LabResources/SweRVo
lf/zephyr/samples/basic/threads/build
```

Recommendations

- In its current stage, RVfpgaSoC should not replace the current Xilinx curriculum. It has potential but more improvements need to be made.
 - Would need someone to work further with the platform and develop the extensions and next steps for each lab.
- The following questions need to be answered and front-loaded with students:
 - How does one customize the hardware? Add a PMOD?
 - How does one customize the software?
 - How does one apply the RTOS constructs to create new applications?
 - How does one apply TensorFlow to fully use its capabilities?
- A choice needs to be made between teaching the IP Integrator with Vivado or just using RVfpga-SoC with FuseSoC, since the IP Integrator/RVfpga combination seemed pointless.
 - Pros - RVfpgaSoC with FuseSoC workflow is actually quite nice and quick on Ubuntu.
 - Cons - We don't get the benefit of easily adding new IP and hardware using the IP Integrator

Video Demo



Individual Final Thoughts and Questions

