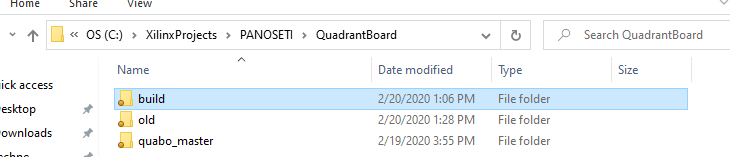
**Building Quabo from quabo\_master**

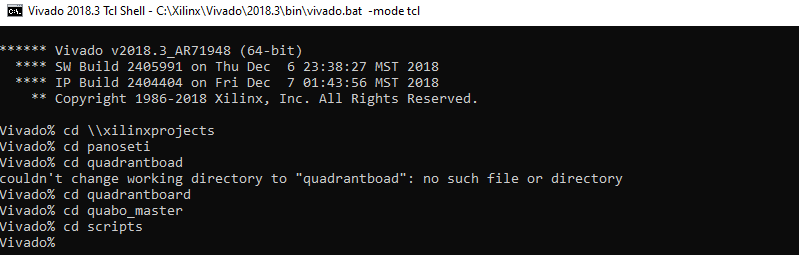
RR February 20, 2020 (revised version)

Download the “quabo\_master” repository from git and unzip it somewhere.

Make a directory called “build” **next** to that directory:



Open the Vivado 2018.3 tcl shell and navigate to the quabo\_master\scripts directory



Note the double backslashes

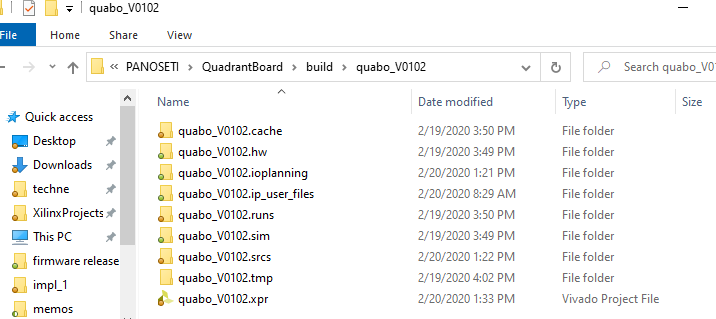
You can open the build\_quabo.tcl script and change the project name if you like. I changed it to quabo\_V0102 (version 10.2) for this demo. It can be whatever you want.



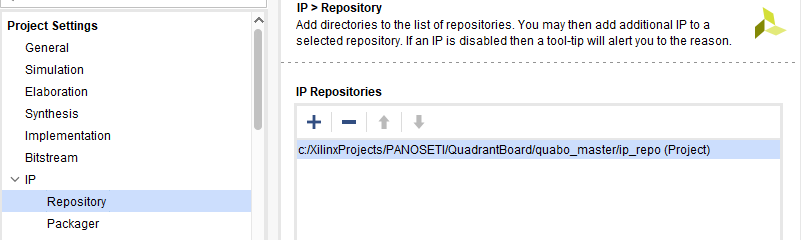
Enter *source build\_quabo.tcl*



This will take a few minutes to execute the script and build the project with the given name, in the ./build directory:



You can then open the project in Vivado and run the toolflow to generate a bitstream. Make sure the IP repository is pointed to the one included in the quabo\_master repository:

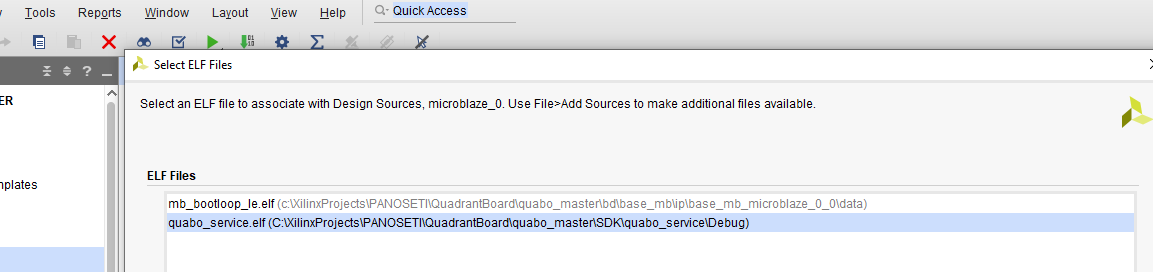


If you add any source files (hdl, xdc, for instance) make sure you add them in the quabo\_master/hdl or /constraints folder, not “local to project”.

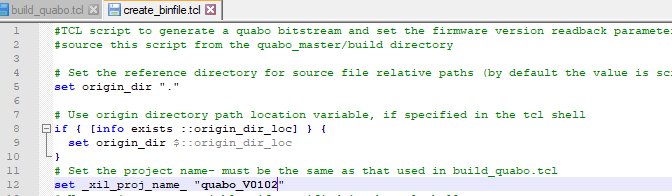
The SDK project for the MicroBlaze software is in the quabo\_master/SDK directory. You can work on the software here. When you are ready to create a programming file, here’s how to do it.

**Creating the binfile**

In Vivado, make sure the compiled .elf file is associated with the MicroBlaze processor:

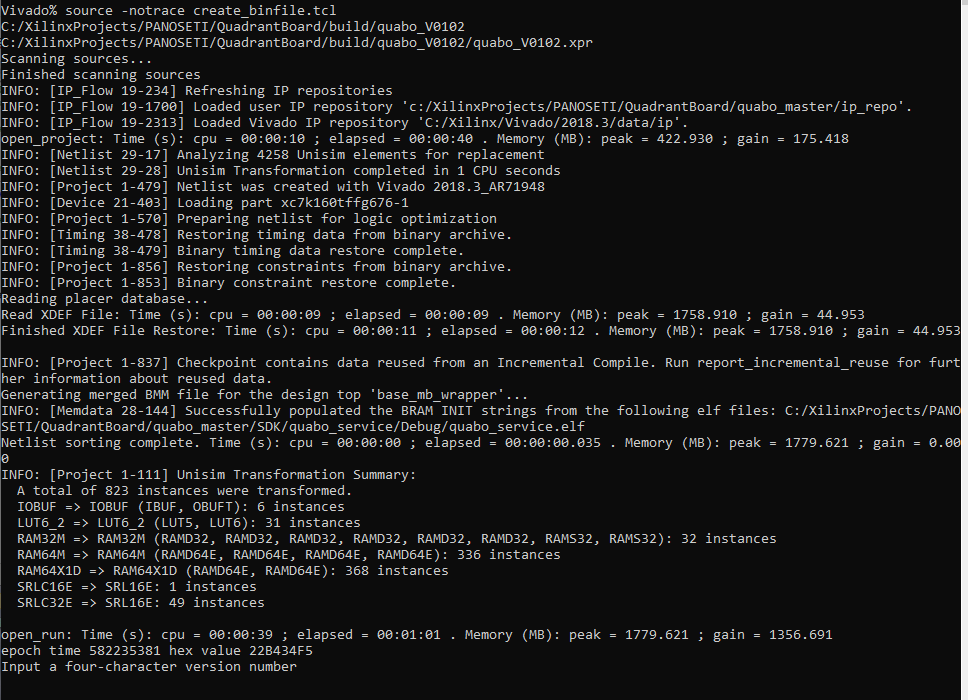


In the create\_binfile tcl script, make sure the project name is the same as that chosen before:



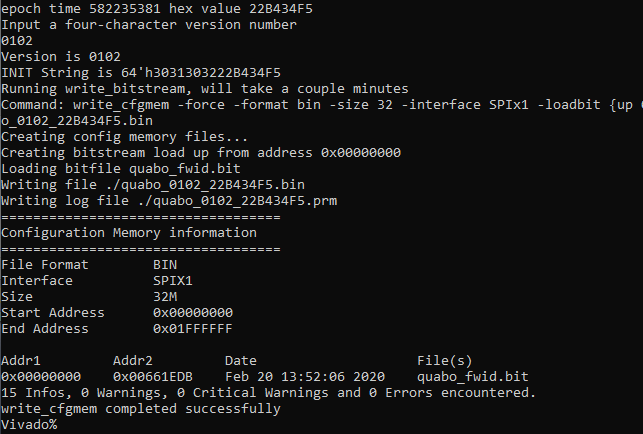
Enter the following command at the Vivado tcl prompt: *source -notrace create\_binfile.tcl*

This will take a few minutes to open the design and ask you to input four characters:



These could be any four characters, but we’ve been using the format eg, v 10.2 = “0102”.

This will again take a few minutes to change the INIT attribute of the FW\_ID 64-bit LUT, where we store the four ASCII characters that were entered, plus and “epoch time” value to guarantee a unique identity, and then write the bitstream:



The new binfile will be stored in the scripts directory, along with a .bit version that can be loaded into the FPGA if desired:

