FROM VIVADO HLS TO EAGLE25

The logic for a particular problem undergoes several stages before it is finally realized on an FPGA. This can be labelled as follows:

1. Coding for the problem in High Level Synthesis: Vivado HLS
2. Testing if the logic implemented is correct: Test Bench in Vivado HLS
3. Virtual synthesis of the IP to get an estimate of the time and resource utilization of the IP: Synthesis in Vivado HLS
4. To generate an IP: Export IP in Vivado HLS
5. To be implemented on the FPGA card: Generate bit map file in Vivado Tool
6. To implement physically on Virtex-7 FPGA sitting in a test crate at UWM: Use eagle25 login to manipulate the input file and download the output file.

The following link on Twiki webpage explains in detail how the entire process is carried out:
https://twiki.cern.ch/twiki/bin/viewauth/CMS/L1TriggerPhase2HLSProjects

Here, I am going to discuss how I implemented the bit file generated during the HLS Workshop at TIFR in August 2017 and what errors came during the process.

The MakeHT code (https://github.com/SridharaDasu/VivadoHLSProjects/tree/master/HT) was modified to implement a LUT to see the 8-bit ET values and add them if they were above a certain set threshold for both, rgnET and hfET. After successfully writing the logic of the code in Vivado HLS tool, we utilized the test bench to confirm the logic and outputs. Also, we were generating the link maps which would be useful later to edit the VHDL file in Vivado tool in the test bench file.

After successfully synthesizing in Vivado HLS tool, open the Vivado tool after sourcing the license libraries. Follow these steps in Vivado:

1. Migrate to Project Settings tab on the left side of the tool. Choose IP -> Repository Manager. Click on the green '+' icon in that tab. Browse to the 'solution' folder of your Vivado HLS code which was used to generate the IP. Select that 'solution' folder and click 'OK', the IP should be visible to you in the window. Then click 'Apply'.

2. Secondly, migrate to 'Project Manager' option on the left side tab, click on it. A new tab titled 'IP Catalog' will pop up on the right side of the tool. Amongst the options, search for the one titled as your function name. The case of the function name will be different than in the original function, with only the first letter capitalized and all others in small case. Click on the function name. A new window will open up, with the implementation number. For example, if you are generating it for the first time, it will be titled 'MakeHT_0'. Keep all the options same and click on 'Generate'. This might take a while.

3. Till the synth design is running, you can edit the ctp7_top.vhd file. This file contains all the port and component mapping for the inputs and outputs of your IP. For example, for MakeHT, all the 252 rgnET and 144 hfET input ports along with 1 HT output port. Following changes are to be made:

    1. After 'Begin User Code', Line 193: Rename the COMPONENT as the correct iteration number, like MakeHT_0.
    2. In the same COMPONENT PORT map, add all the IN and OUT STD_LOGIC_VECTORS with their correct bit size.
    3. Following this, add the signals declarations for all the inputs and outputs. No change is to be made after the 'End User Code' comment.
    4. Scroll down to the next 'Begin User Code' comment, originally on line 409. Here first chamge both the label title and the instantiation from hls_demo to your function name, like MakeHT_0.
    5. Following this, the PORT MAP, add all the input and output ports with the correct sign '=>'.
    6. Following this, remove the configuration registers and add all the input and output links, s_INPUT_LINK_ARRAY with correct bit size and bit postion according to the link to which it is connceted. For this CTP7, we have a total of 67 links, each of 192 bits.
    7. Remove code corresponding to 'algo_config', 'algo_in' and 'algo_out' from throughout the vhd file (a total of 3 places).
    8. Make sure you have not made any syntax error (commas and semicolons with braces etc). This file follows VHDL coding rules.
    9. Do not get confused between signal name and port name. They can be kept same or different.
    10. Save this file after all the changes.

Next, to generate bitstream, three important steps are carried out sequentially:

1. Run synthesis
2. Run Implementation
3. Genrate Bitstream

The steps can be done all at once or my manually clicking on the options one-by-one.

After successfully generating the bitsteam, a ctp7_top.bit file is generated in the impl1 folder. This file needs to be loaded on the FPGA and tested. To do this, we have been facilitated with 'eagle25', a Super User login which enables us to use the ZYNQ processor which has a linux platform running on it, which in turn communicates with the Virtex-7 FPGA.

Login command for superuser: su ctp7hls
Password: hls

To run our bit file on this FPGA, follow the steps given on Twiki page mentioned above. **Do NOT rename the bit file** as then it will give a 'Permission Denied' error while copying it to the /tem/. folder on eagle25.

Use your UW login and make sure you are using this machine only in the time slot assigned to you. The eagle does not have a mechanism to reject multiple super user login; hence, use only when you are sure no one else is using it.

After downloading the input link data file, manipulate the inputs according to your link assignment. After saving the changed input file, you have to load it on the CTP7 again, using the upload command. Following this, download the output file and you can observe the links in the output text file.

In case you were unable to see the changes in the output file, check for the following:

1. Correct links were changed in the input file.
2. Input file was loaded correctly.
3. No one else is using the card.