Assignment

Hardware Design, Simulation, Synthesis and Implementation

Objective: In this assignment, you will be designing, simulating and synthesizing various digital blocks including adders, multiplexers, decoder, parity generator, flipflops and an updown counter. A hardware software co-design approach is also introduced where the software part involves the processor being used solely to provide test inputs.

Tools used: Modelsim, Xilinx Vivado, Python, Synopsys

Hardware used: Zybo board, Digilent Discovery kit.

Guidelines to be followed:

- **Naming convention**: File name should match with the module name (eg: if the module is named "full-adder", the design file name should be named as "full-adder.v" and the testbench should be named as "full-adder_tb.v")
- **Formatting**: Each Verilog file should have a proper header: giving the name, description, version history as given in the sample file. Comments should be given carefully so that program is understandable without too much clutter (You may refer the uploaded sample file)
- **Submission**: The completed word document in the given template (converted to pdf) and uploaded in AUMS. The same pdf along with all Verilog design and testbench files should be submitted in the given one drive link ESLD-Assignment.
  You can create a folder structure for yourself (XXX stands for the 3 digits of your roll number)
  - "ECE19xxx" -> Assignment 1A, Assignment 1 B, 1C, 1D, 1E

Part D- Hardware Design using FPGA with a hardware-software approach (Digilent Discovery kit)

1. Synthesize the up-down counter on an FPGA and simulate using test vectors generated from the ARM processor. The output bits need to be monitored on screen using Digilent Discovery Kit to verify the functionality

Date : 17/10/23

Hardware used : Pynq Z1 & Digilent Digital Discovery kit

Procedure :

I. Generate Inputs using Python from the ARM processor.
II. Monitor the outputs using a logical analyser (Digital Discovery kit) to verify its functionality.

Pin Configuration :

##ChipKit Digital I/O On Outer Analog Header

##NOTE: These pins should be used when using the analog header signals A0-A5 as digital I/O (Chipkit digital pins 14-19)

set_property -dict { PACKAGE_PIN Y11 IOSTANDARD LVCMOS33 } [get_ports { ck_io[14] }]; #IO_L18N_T2_13 Sch=ck_a[0]

set_property -dict { PACKAGE_PIN Y12 IOSTANDARD LVCMOS33 } [get_ports { ck_io[15] }]; #IO_L20P_T3_13 Sch=ck_a[1]

set_property -dict { PACKAGE_PIN W11 IOSTANDARD LVCMOS33 } [get_ports { ck_io[16] }]; #IO_L18P_T2_13 Sch=ck_a[2]

set_property -dict { PACKAGE_PIN V11 IOSTANDARD LVCMOS33 } [get_ports { ck_io[17] }]; #IO_L21P_T3_DQS_13 Sch=ck_a[3]

set_property -dict { PACKAGE_PIN T5 IOSTANDARD LVCMOS33 } [get_ports { ck_io[18] }]; #IO_L19P_T3_13 Sch=ck_a[4]

set_property -dict { PACKAGE_PIN U10 IOSTANDARD LVCMOS33 } [get_ports { ck_io[19] }]; #IO_L12N_T1_MRCC_13 Sch=ck_a[5]

Implementation :

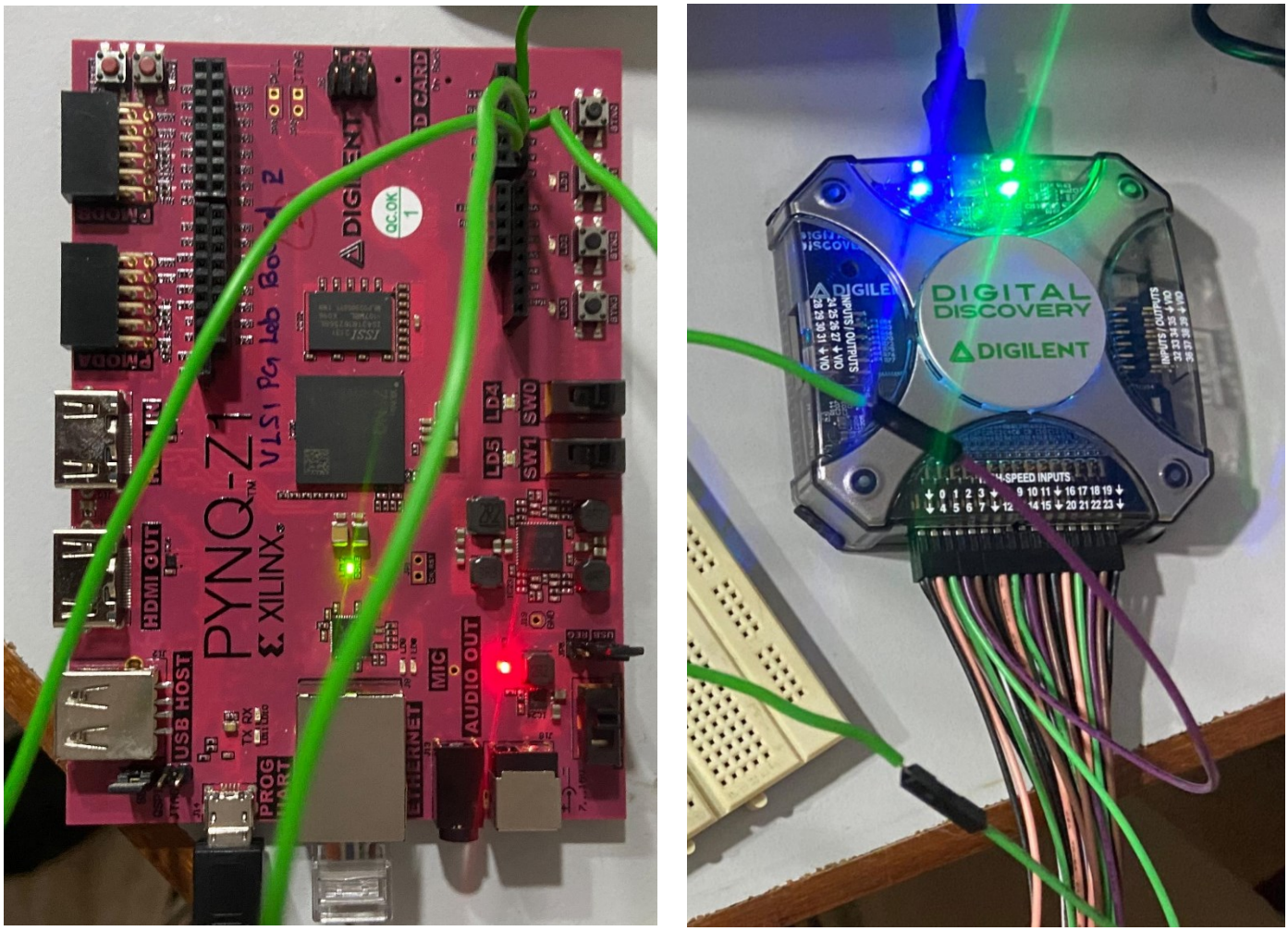

Fig 1. Left : Pynq Z1 board, the outputs A0-A3 are configured as output ports.

Right : Digilent Digital Discovery kit ports 0,1,2,3 receives input from Pynq through jumper cables.

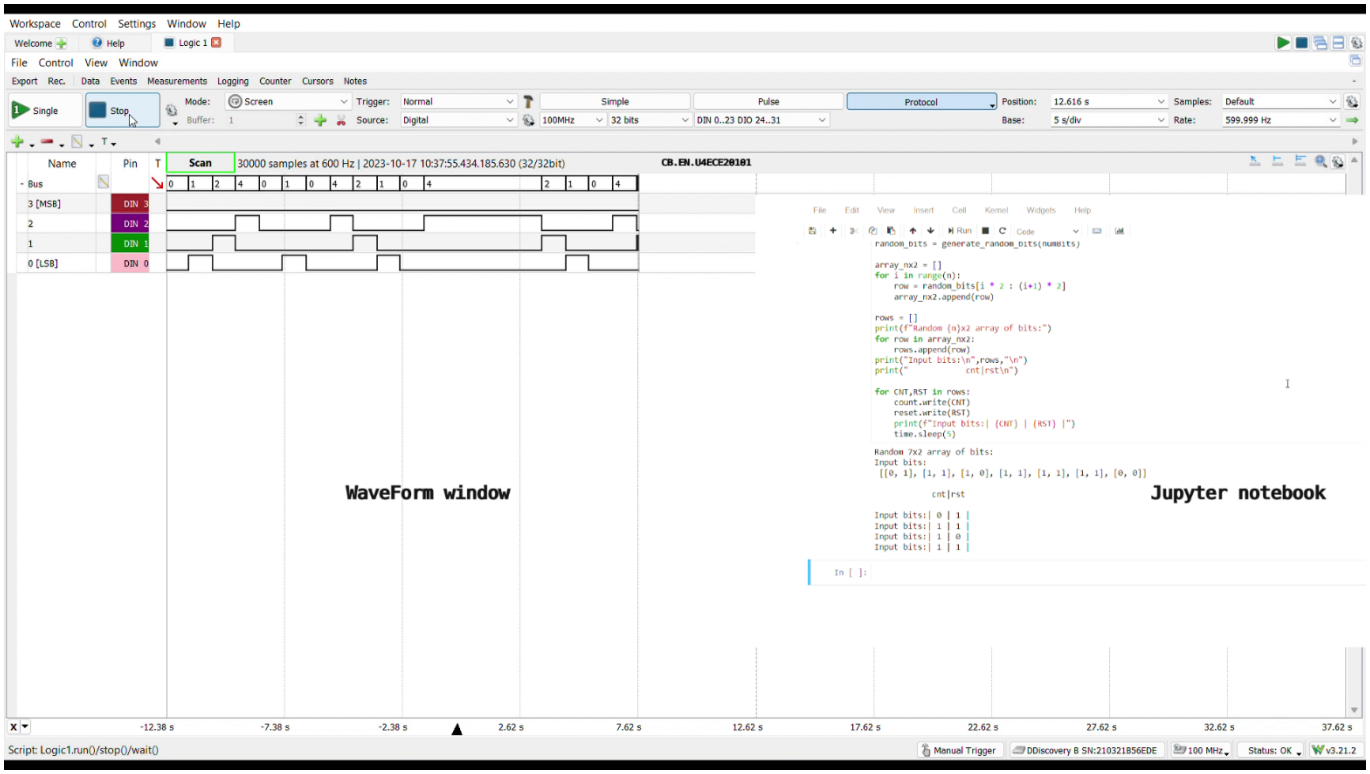

Fig 2. Left : Waveform Window – Logic analyser (Digital Discovery kit).

Right : Jupyter Notebook – Generate Random Inputs from the available ARM processor using Python.

Video can be accessed here.

In this section of the assignment, we are specifically analysing the logic generated in our previous assignment (Part 1C). The codes remain unchanged, and the same codebase is used, with modifications only made to the port's constraint. You can access the Part 1C Jupyter notebook here.