מעבדת מאיצי חומרה – מעבדה 4

רועי שני 315995258
ינאי מייסון 206701591

Lab Objective

In this laboratory we learned to use the capabilities of the Quartus software, specifically to perform synthesis for models we developed previously in Lab 1, with the addition of an extra hardware component (PWM) that we implemented in this lab. The synthesis was carried out on an FPGA.

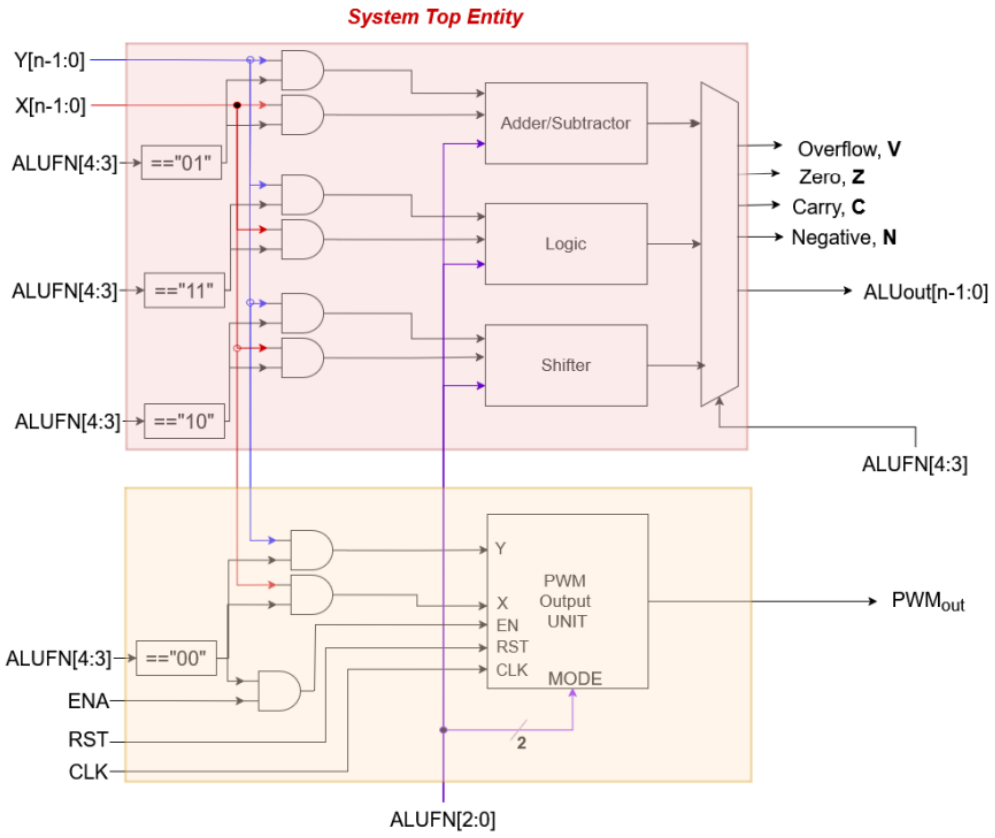

Figure 1 - system design overview

PWM Module

This module receives as inputs the clock output from the PLL (which is 2 MHz), an enable signal (en), a reset signal (rst), and the values of X and Y. When we feed the ALU with '00' on bits 3–4, the design enters the PWM mode.

The PWM block operates in three states: SET, RESET, and TOGGLE. Internally, a counter is initialized to 0. Inside a clocked process (triggered on the rising edge of the 2 MHz clock), the counter increments by 1 on each clock tick until it reaches the value of Y. When the counter value equals X, the output signal is changed according to the currently selected state, following the convention shown below:

- **SET**: When counter == X, set the PWM output high.
- **RESET**: When counter == X, set the PWM output low.
- **TOGGLE**: When counter == X, invert (toggle) the PWM output.

After reaching Y, the counter wraps around to 0 and begins counting again. In this way, the duty cycle of the PWM output is determined by the ratio of X to Y and by which state (SET/RESET/TOGGLE) is chosen.

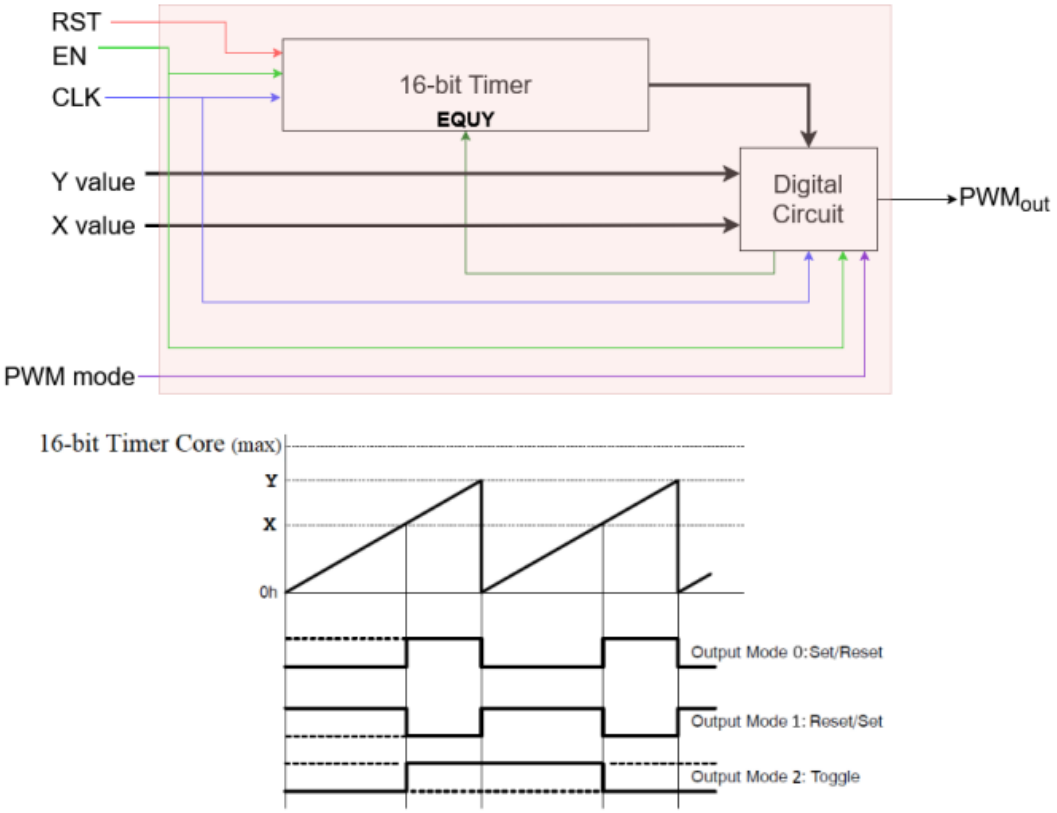

Figure 2 - PWM module design

Logical Simulation

First, we verified that the system performs logical operations as defined by running the testbench (TB) in ModelSim. We saw that our code works exactly as expected, and we corrected logical errors in the design.

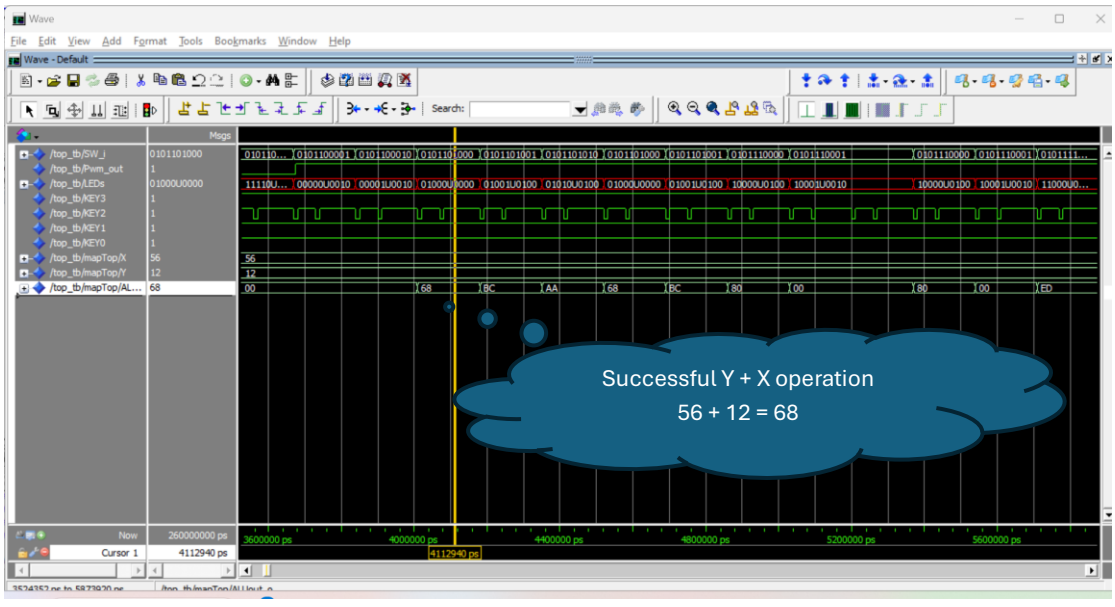

Figure 3 -successful add simulation

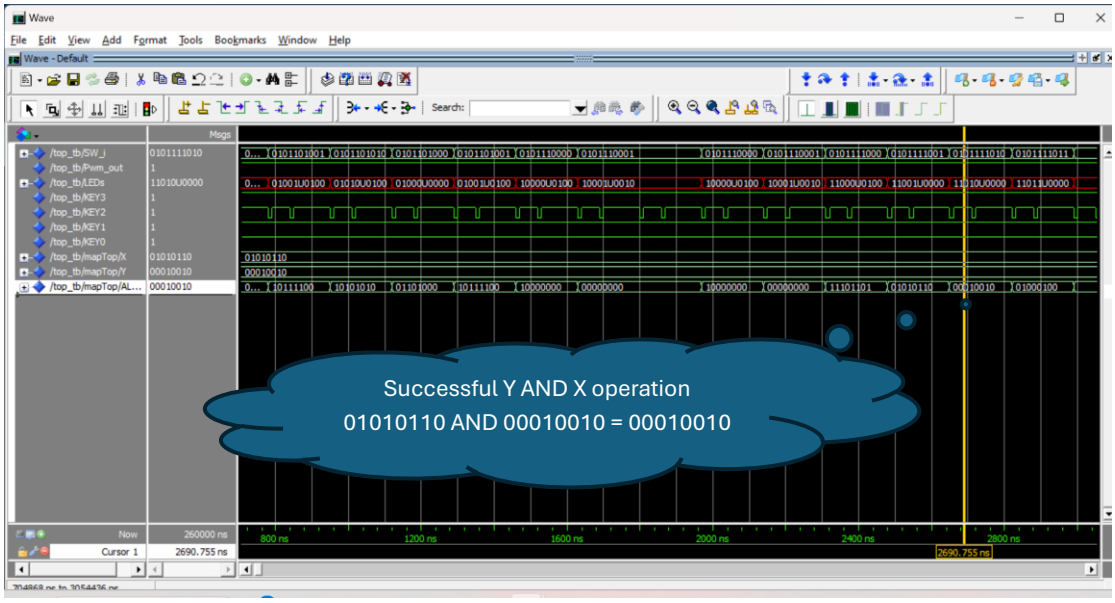

Figure 4 – successful AND operation

System Validation

Design

We inspected the register-transfer-level implementation using Quartus's RTL Viewer to verify that the top-level module and its four submodules (ADDERSUB, LOGIC, SHIFTER, PWM) were instantiated correctly and connected as intended.

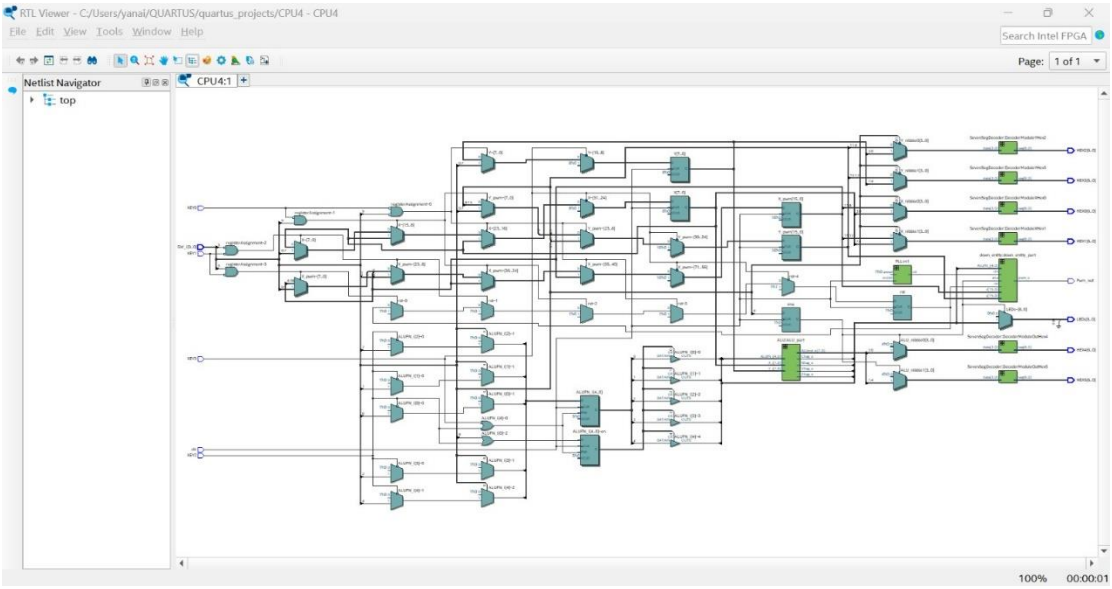

Figure 5- RTL system view

Next, we used the Technology Map Viewer to confirm that the synthesized netlist matched our design intent and to visualize how logic cells were allocated in the FPGA fabric.

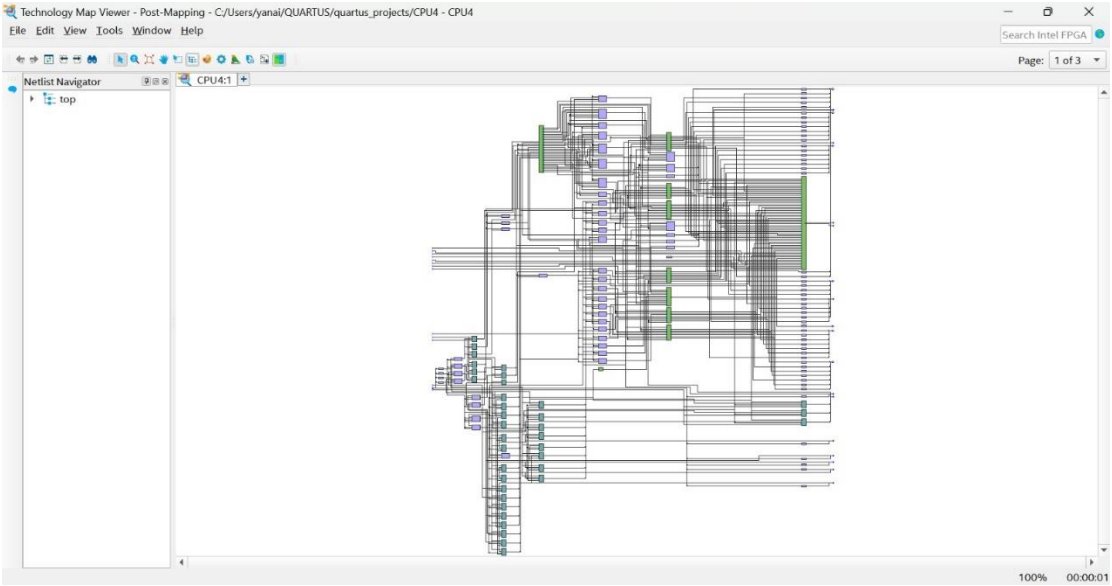

Figure 6- Technology viewer system design

Finding Fmax

In order to obtain the critical path the system was encapsulated in a constraint file as described in the Quartus guide. The following result was obtained:

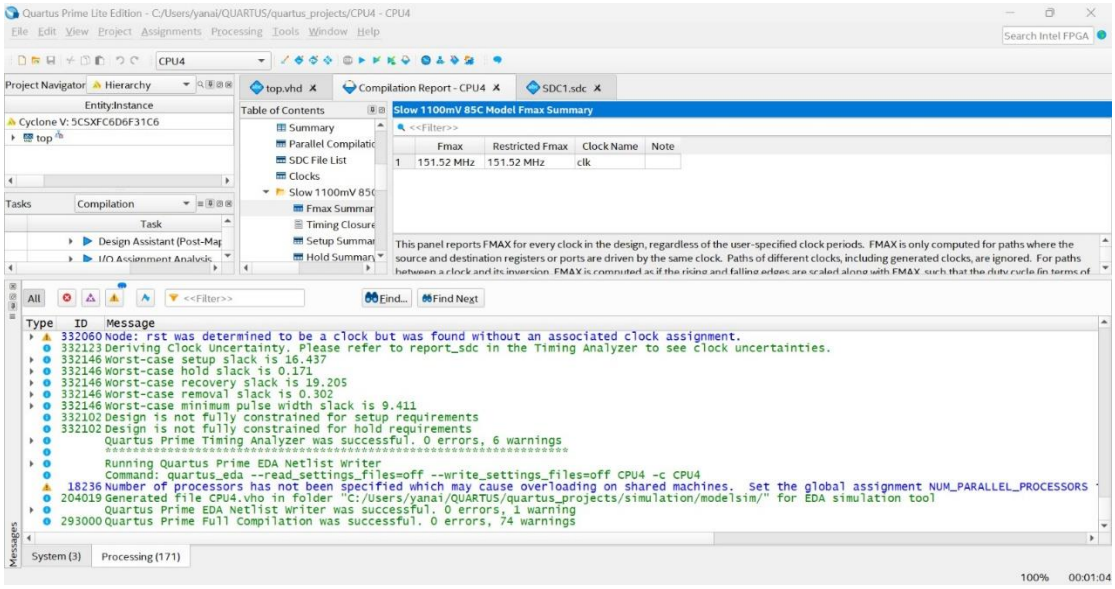

Figure 7- Finding Fmax

In order to identify the critical path the Timing analyzer was deployed, which yielded the following results:

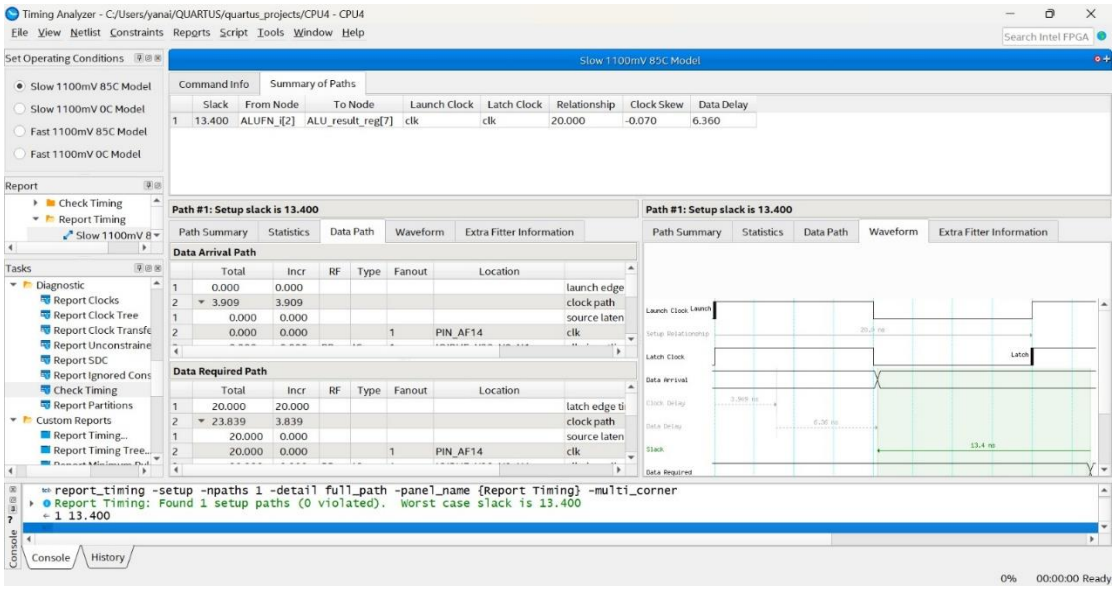

Figure 8- finding critical path

The critical path was identified as the one that passes through the ALU into the Shifter, as seen in the technology map viewer below:

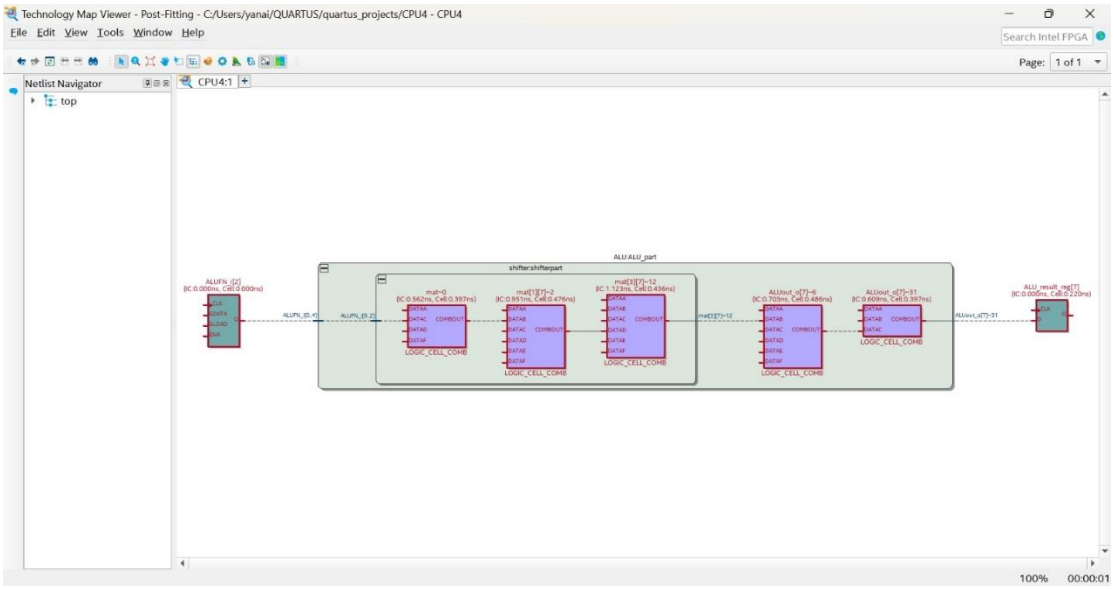

Figure 9- post-fitting technology map viewer of the critical path

The resource property editor indicates that indeed the critical path is that identified:

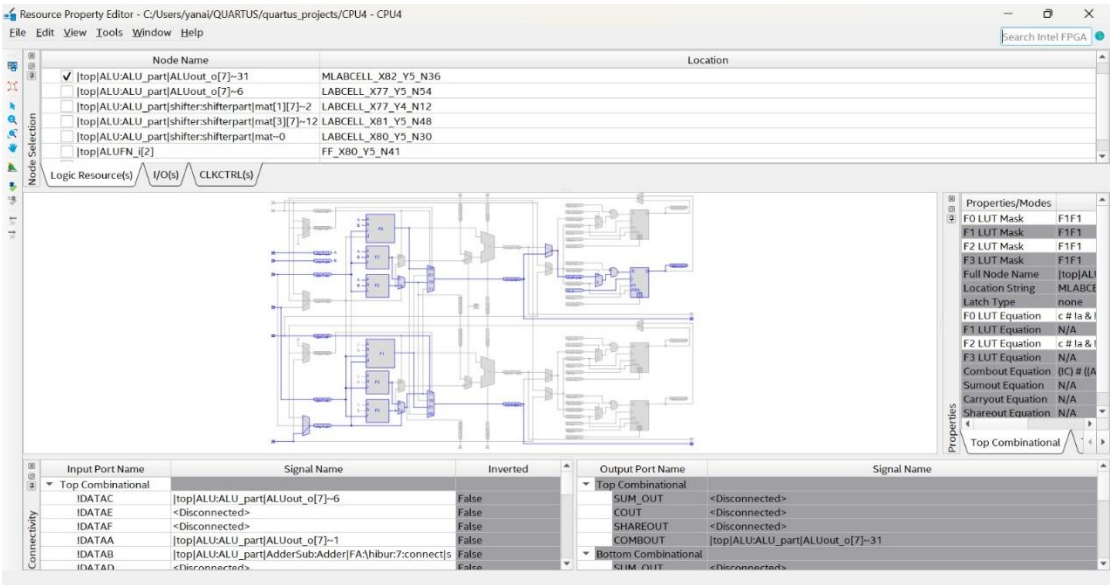

Figure 10 - resource property editor display

The critical path can also be viewed in the chip viewer:

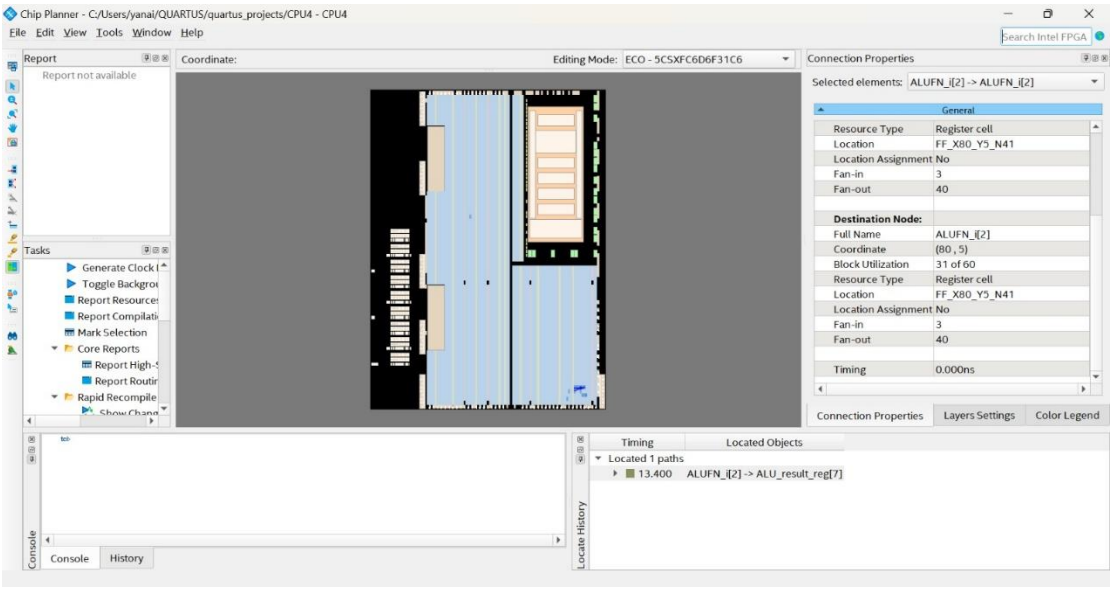

Figure 11 - Chip planner view of critical path

Zooming in provides a better understanding of the critical path:

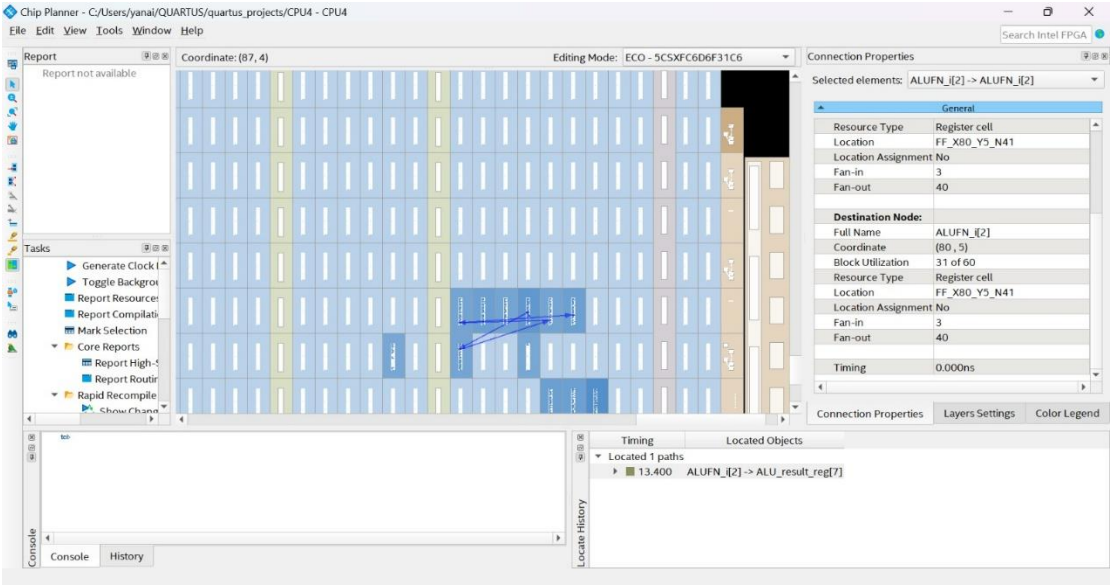

Figure 12 - zoomed in chip planner view of critical path

Running the program

The pin planner was assigned as described in the DE-10 standard user manual:

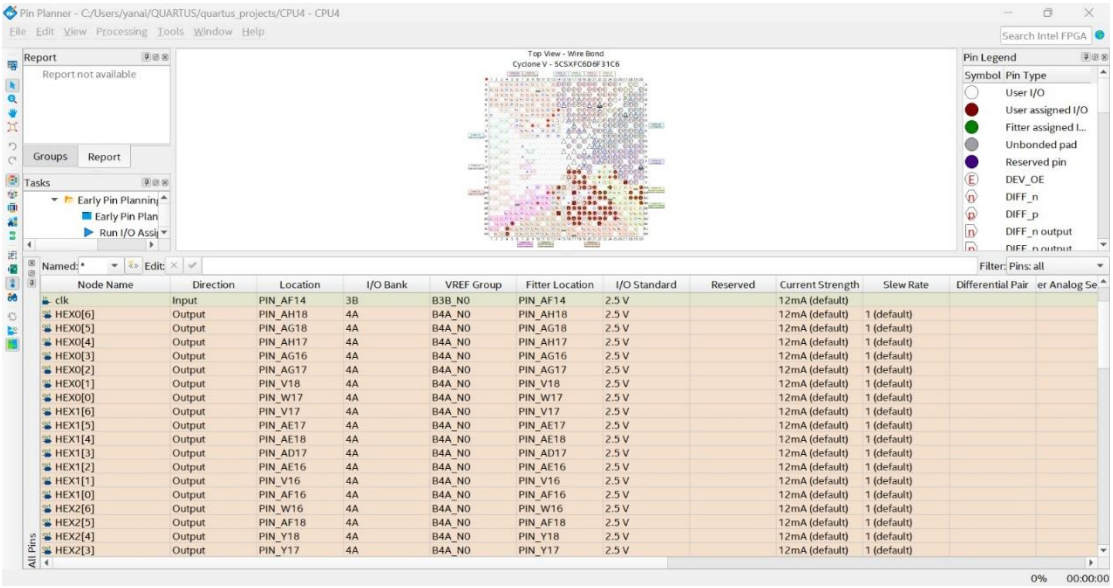

Figure 13 - pin planner example

The program was successfully compiled:

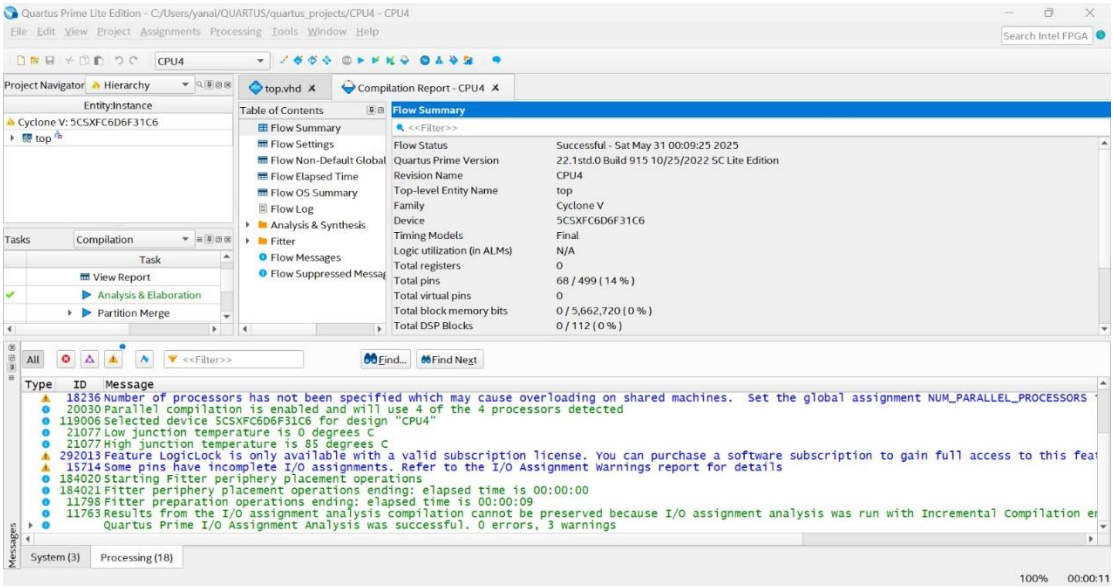

Figure 14 - successful compilation on quartus software

The program was then loaded successfully onto the FPGA:

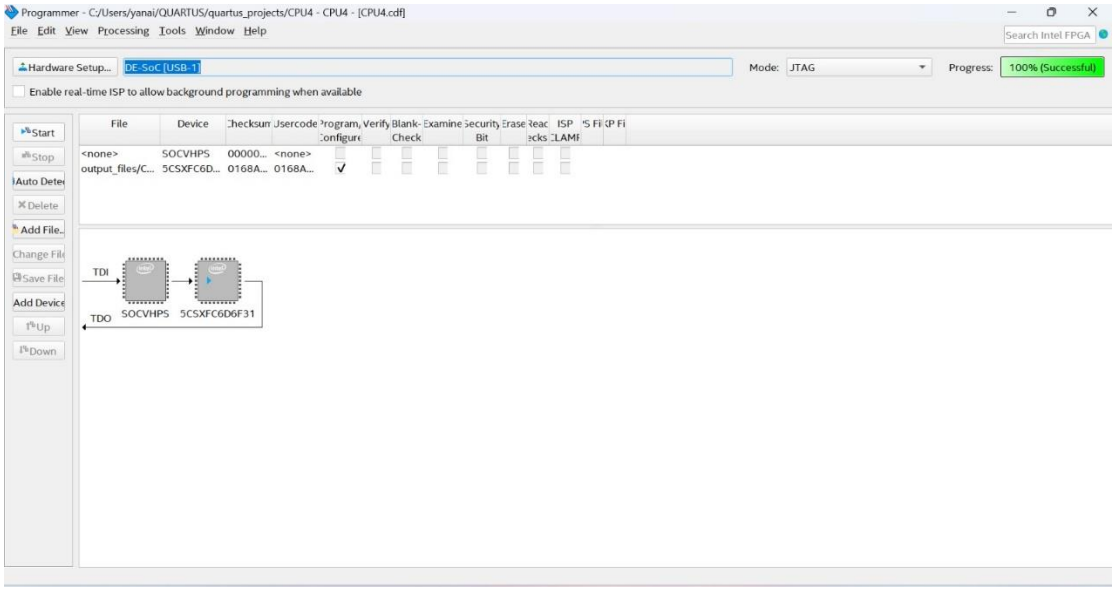

Figure 15 - successful loading of program to hardware

In order to perform verification of the program, the signal tap was utilized:

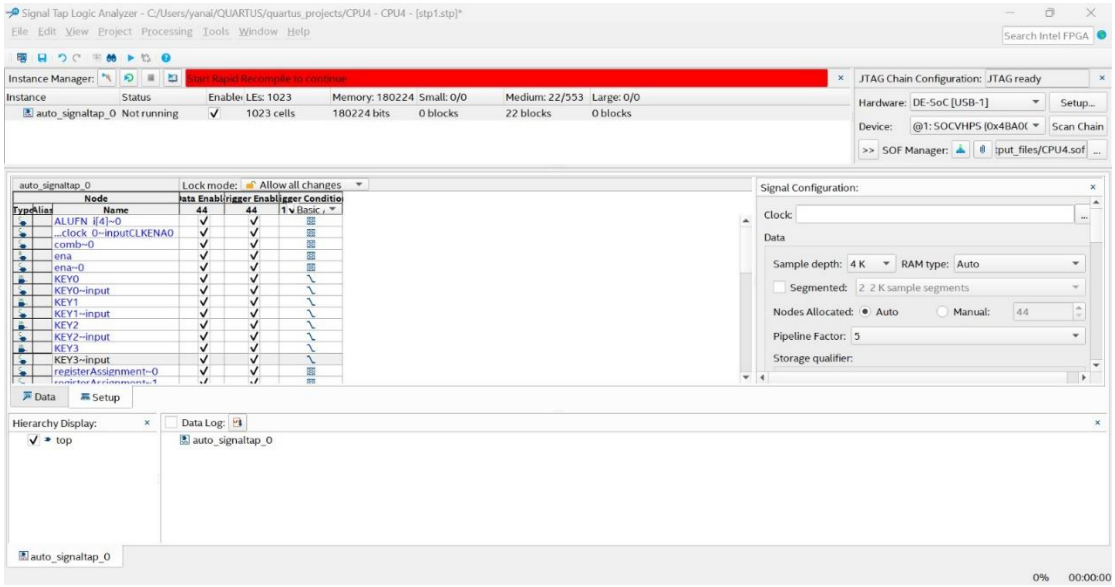

Figure 16 - signal tap initialization

The following functionalities were measured:

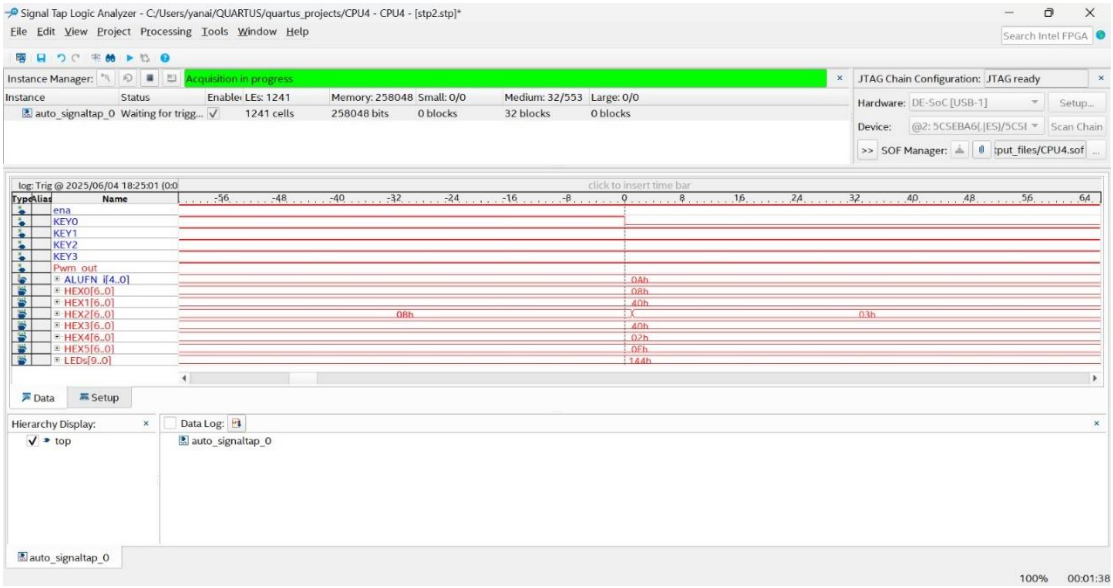

Figure 17 - Pressing KEY0 correctly loaded a new X value into the ALU

Figure18 - successful operation of PWM

All measured functionality matched our design specifications, confirming that the system works correctly on the target FPGA.