Group 2- Patrick Balcombe, Harry Austin, Devashish Bishnoi

**ECS615U-** **Laboratory Session 4**

# **Abstract**

**Method**

The first stage to create the devices was taken in ISE project navigator in which the VHDL code was written for the actual devices and their test benches. Through this we then also simulated the devices in ISim to gain timing diagrams for our test benches to confirm that the devices were working as expected according to the derived truth table. The constraints files were then created using a standardised UCF file created in a previous lab which were then altered for the I/O names of the specific devices in ISE. These constraints files map the inputs out outputs of the top level devise to the FPGM board that we later uploaded to the devices. We configure the FPGM board and upload the top level devises bit files, created in ISE, using iPACT after which the devices actual inputs and outputs where then tested.

The devices higher up in the design such as the Modulo-m counters use the functionality of the devises lower down in the design such as the half adder and d-flip flop to create a hierarchical design. Devices that are further down are port mapped, defining their connections within the device, into the higher level device so that it can be passed the signals necessary to carry out its function.

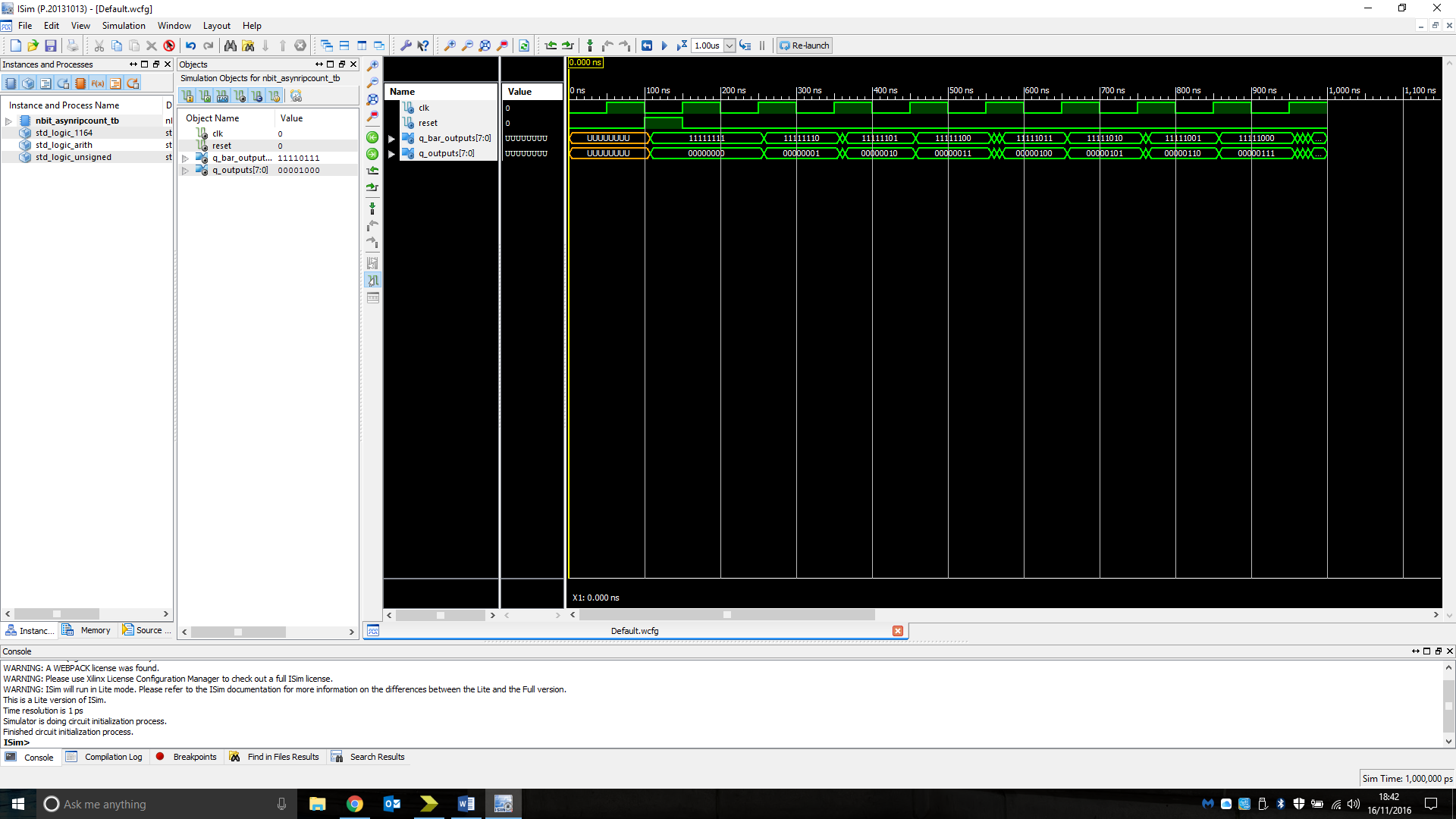
In this lab we created next state logic devices which in order to implement in VHDL logical equations had to be found. These logical equations were found by using k-maps for each one of the output bits from the given input bits. Once found the logical equations for a next state logic block were all implemented in a sub component that could be instantiated in a higher level device that had registers to store the current state.

As in this lab clocks were now required for the devices a signal had to be supplied to the FPGA to act as the clock. We used the variable clock that is available on the FPGA board and connected it with wire connecting it from its pin to one of the board’s general IO pins.

# Simulation and FPGA Testing

In this section we individually run the test bench files for each device in Modelsim and on the physical FPGM board to check the timing diagram as well as actual results of the I/O matches its truth table. The Truth tables lists possible combinations of inputs and the relevant outputs which are compared to the results gained from the simulation and tests on the board.

**n-bit asynchronous (ripple) counter:**

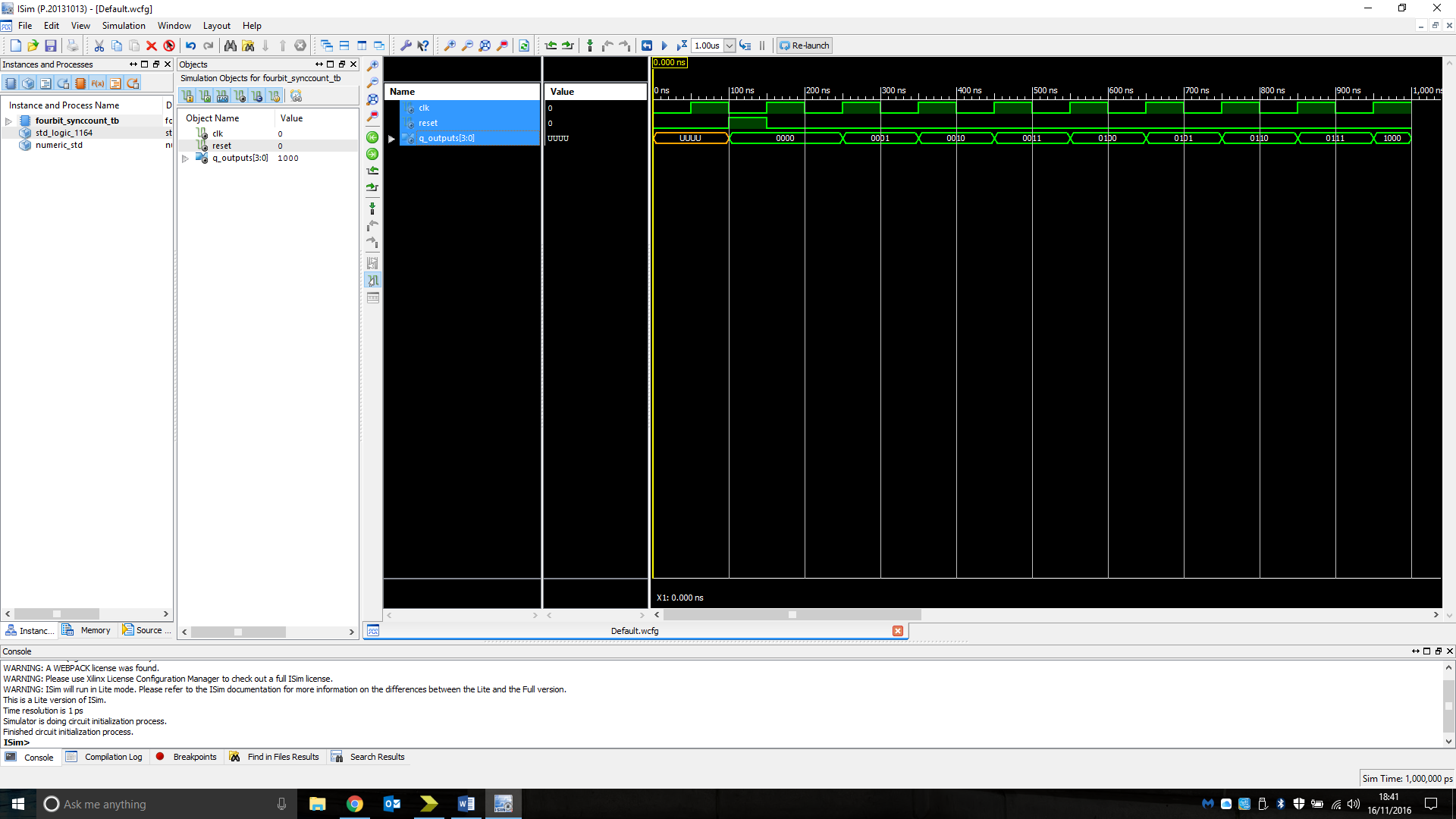
Truth table:

|  |  |  |  |
| --- | --- | --- | --- |
| Reset | CLK | Q\_outputs | Q\_bar\_outputs |
| 1 | 0 | 00000000 | 11111111 |
| 0 | 1 | 00000001 | 11111110 |
| 0 | 0 | 00000001 | 11111110 |
| 0 | 1 | 00000010 | 11111101 |
| 0 | 0 | 00000010 | 11111101 |
| 0 | 1 | 00000011 | 11111100 |
| 0 | 0 | 00000011 | 11111100 |
| 0 | 1 | 00000100 | 11111011 |

Figure: Simulation timing diagram of an n-bit asynchronous counter showing its behaviour when tested with a range of possible inputs.

Analysis: The output of the device is as expected for both the simulation and test on FPGM board.

**4-bit synchronous counter:**

Truth table:

|  |  |  |
| --- | --- | --- |
| Reset | CLK | Q\_outputs |
| 1 | 0 | 0000 |
| 0 | 1 | 0001 |
| 0 | 0 | 0001 |
| 0 | 1 | 0010 |
| 0 | 0 | 0010 |
| 0 | 1 | 0011 |
| 0 | 0 | 0011 |
| 0 | 1 | 0100 |
| 0 | 0 | 0100 |

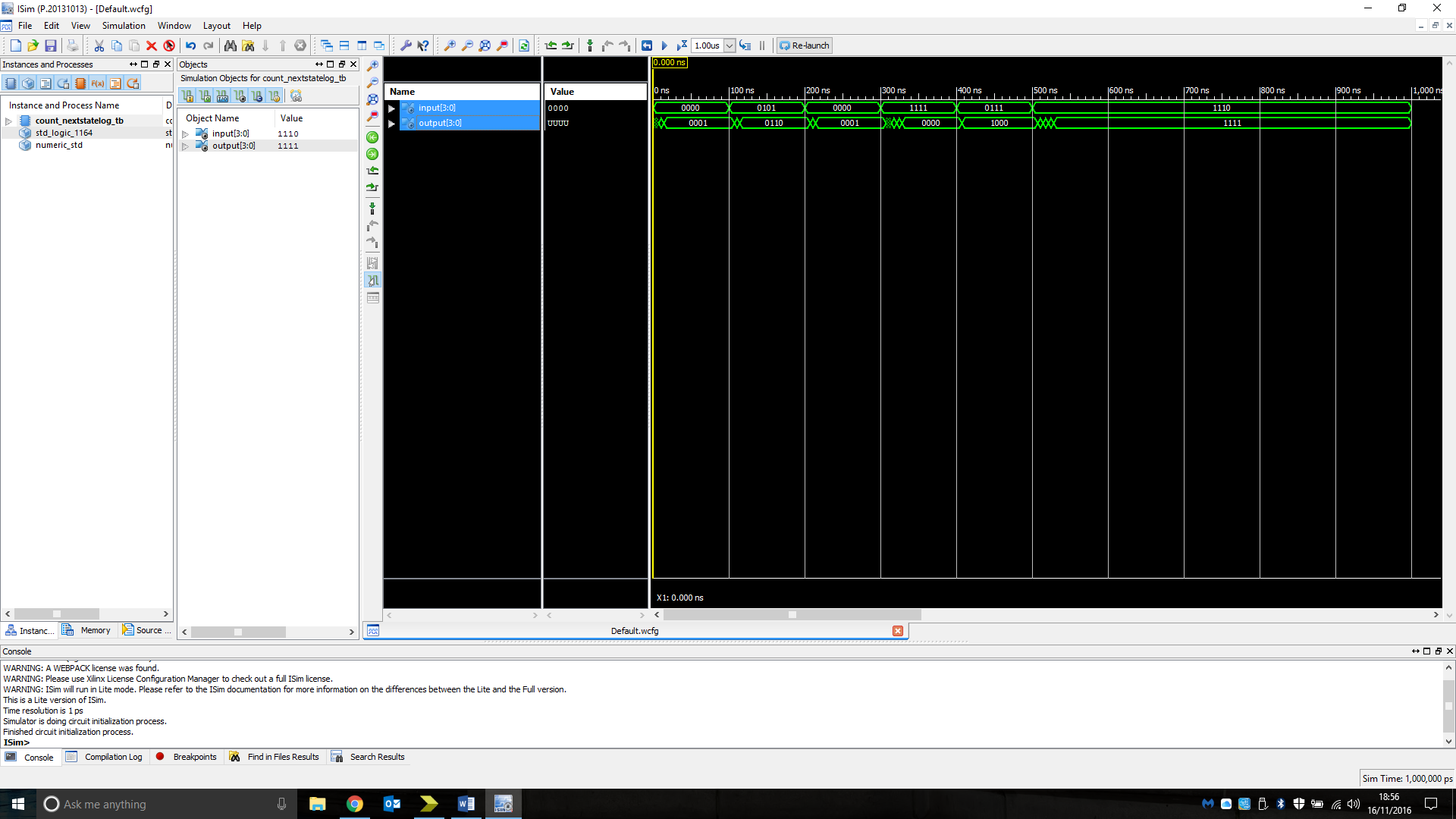
Figure: Simulation timing diagram of a 4-bit synchronous counter showing its behaviour when tested with a range of possible inputs.

Analysis: The output of the device is as expected for both the simulation and test on FPGM board.

**Synchronous counter next-state logic:**

Truth table:

|  |  |
| --- | --- |
| Input | Output |
| 0000 | 0001 |
| 0001 | 0010 |
| 0010 | 0011 |
| 0011 | 0100 |
| 0100 | 0101 |
| 0101 | 0110 |
| 0110 | 0111 |
| 0111 | 1000 |
| 1000 | 1001 |
| 1001 | 1010 |
| 1010 | 1011 |
| 1011 | 1100 |
| 1100 | 1101 |
| 1101 | 1110 |
| 1110 | 1111 |
| 1111 | 0000 |

Figure: Simulation timing diagram of a Synchronous counter next-state logic showing its behaviour when tested with a range of possible inputs.

Analysis: The output of the device is as expected for both the simulation and test on FPGM board.

**n-bit incrementer circuit:**

Truth table:

|  |  |  |  |
| --- | --- | --- | --- |
| Input | | Outputs | |
| InA | C\_in | Sum | C\_out |
| 0000 | 1 | 0001 | 1 |
| 0000 | 0 | 0000 | 0 |
| 1011 | 1 | 1100 | 0 |
| 1000 | 0 | 1000 | 0 |
| 1000 | 1 | 1001 | 0 |
| 1111 | 0 | 1111 | 0 |
| 1111 | 1 | 0000 | 1 |

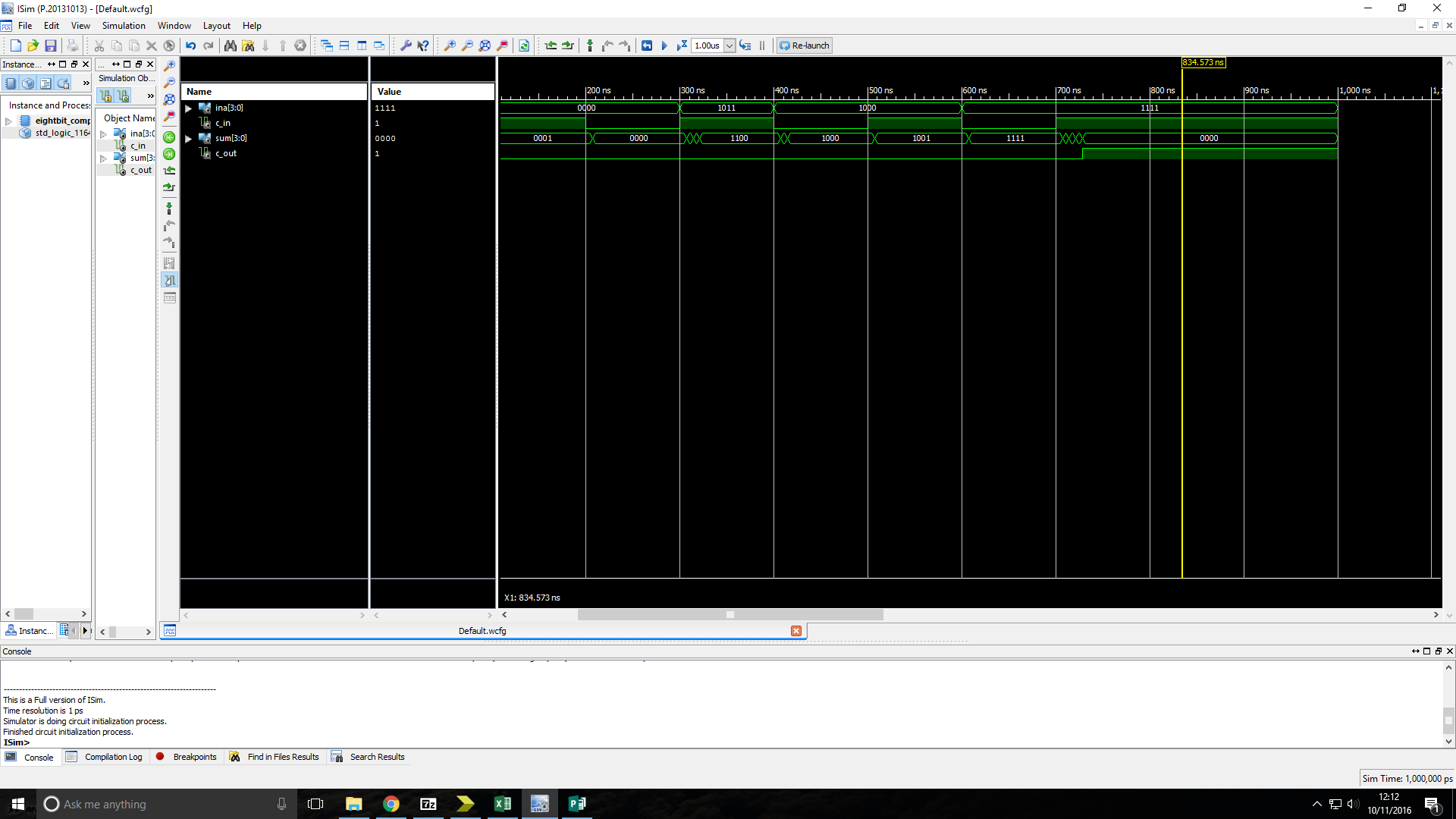


Figure: Simulation timing diagram of a n-bit incrementer circuit showing its behaviour when tested with a range of possible inputs.

Analysis: The output of the device is as expected for both the simulation and test on FPGM board.

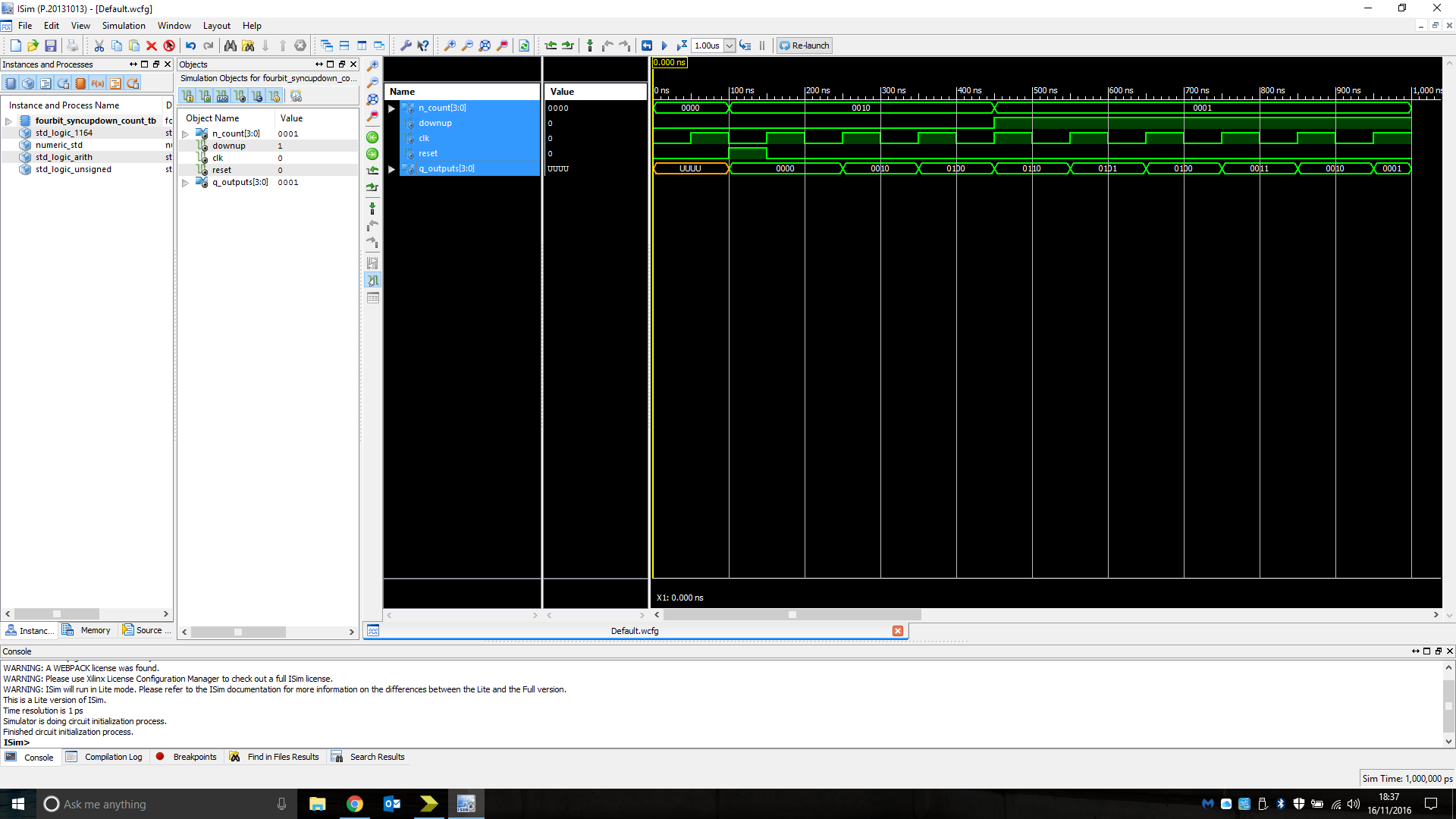
**n-bit synchronous counter with parallel load input:**

Truth table:

Figure: Simulation timing diagram of a n-bit synchronous counter with parallel load input showing its behaviour when tested with a range of possible inputs.

Analysis: The output of the device is as expected for both the simulation and test on FPGM board.

**4-bit synchronous up/down counter:**

Truth table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| N\_count | Down/Up | Reset | CLK | Q\_outputs |
| 0010 | 0 | 1 | 0 | 0000 |
| 0010 | 0 | 0 | 1 | 0010 |
| 0010 | 0 | 0 | 0 | 0010 |
| 0010 | 0 | 0 | 1 | 0100 |
| 0001 | 1 | 0 | 0 | 0110 |
| 0001 | 1 | 0 | 1 | 0101 |
| 0001 | 1 | 0 | 0 | 0101 |
| 0001 | 1 | 0 | 1 | 0100 |
| 0001 | 1 | 0 | 0 | 0100 |
| 0001 | 1 | 0 | 1 | 0011 |

Figure: Simulation timing diagram of a 4-bit synchronous up/down counter showing its behaviour when tested with a range of possible inputs.

Analysis: The output of the device is as expected for both the simulation and test on FPGM board.

**Q-** When counting upwards with increments of “1111” the synchronous up/down counter appears to count down as when adding 1111 to 1111 this gives a value of 1110 (one number bellow 1111). This is as there is a carry from the first column into the second and second to the third but obviously no carry into the first column.

**8-bit Comparator:**

Truth table:

|  |  |  |
| --- | --- | --- |
| InA | InB | output |
| 00000000 | 00000001 | 0 |
| 00000000 | 00000000 | 1 |
| 11111111 | 11111111 | 1 |
| 11110111 | 10111111 | 0 |
| 01010101 | 10101010 | 0 |
| 01010101 | 01010101 | 1 |

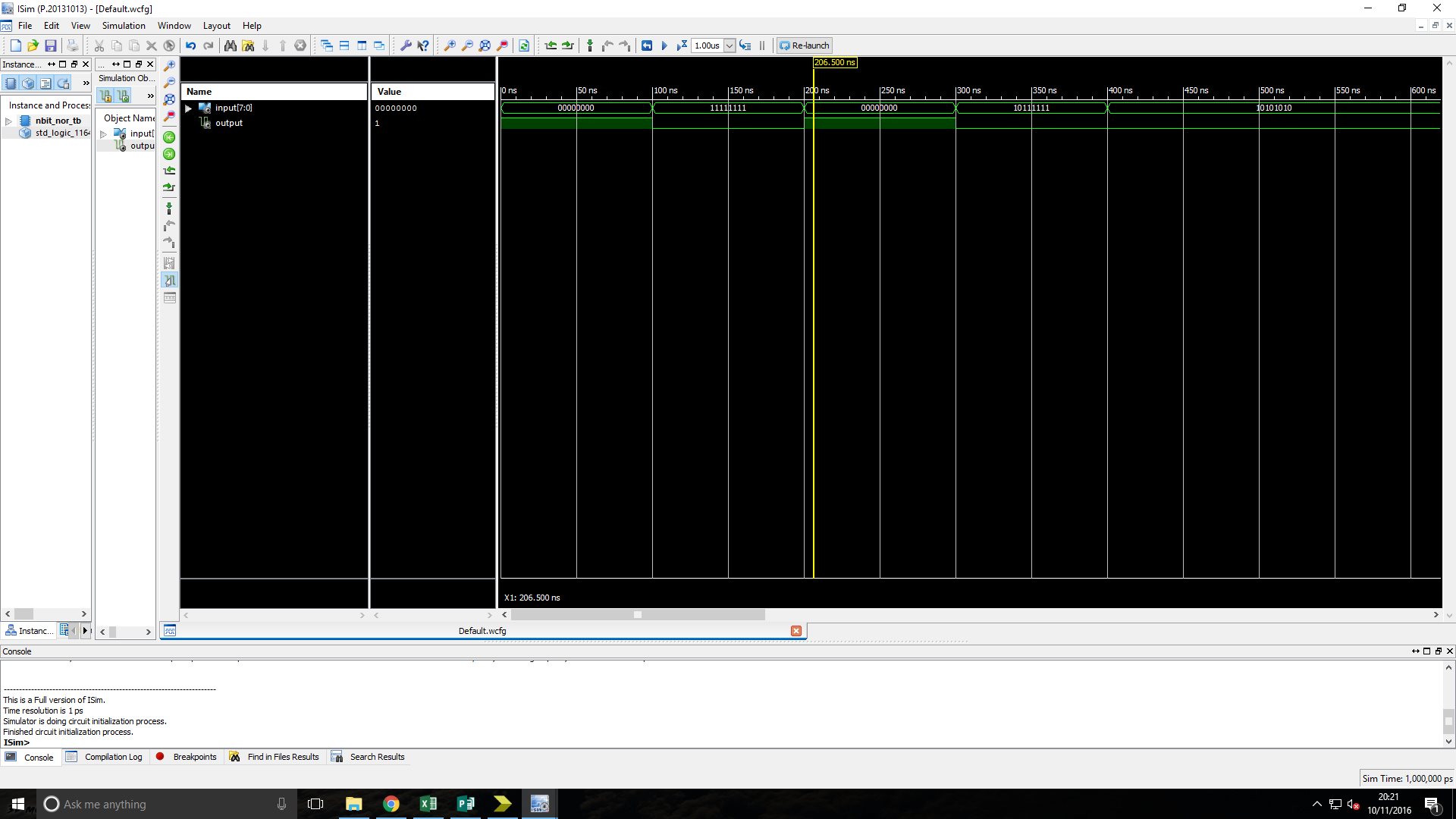
Figure: Simulation timing diagram of a 8-bit Comparator showing its behaviour when tested with a range of possible inputs.

Analysis: The output of the device is as expected for both the simulation and test on FPGM board.

**8bit NOR:**

 Truth table:

|  |  |
| --- | --- |
| input | output |
| 00000000 | 11111111 |
| 11111111 | 00000000 |
| 00000000 | 11111111 |
| 10111111 | 00000000 |
| 10101010 | 00000000 |

Figure: Simulation timing diagram of a 8-bit Comparator showing its behaviour when tested with a range of possible inputs.

Analysis: The output of the device is as expected for both the simulation and test on FPGM board.

**modulo-m counter with asynchronous reset:**

Truth table:

Figure: Simulation timing diagram of a 8-bit Comparator showing its behaviour when tested with a range of possible inputs.

Analysis: The output of the device is as expected for both the simulation once a small error due to inconsistencies in the naming of the incrementor file was rectified. The test on FPGM board produced no errors.

**Q-** The counter with enable only increments when the enable input is high, once the enable button is released it stopes counting and continues to output the value that it had reached.

**modulo-m counter with synchronous reset:**

Truth table:

Figure: Simulation timing diagram of a modulo-m counter with synchronous reset showing its behaviour when tested with a range of possible inputs.

Analysis: The output of the device is as expected for both the simulation and test on FPGM board.

**Washer controller output logic:**

Truth table:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | Outputs | | | | |
| Function | State | Door Lock | Water Pump | Soap | Rotate Drum | Drain |
| Load clothes | 000 | 0 | 0 | 0 | 0 | 0 |
| 000 | 0 | 0 | 0 | 0 | 0 |
| 001 | 0 | 0 | 0 | 0 | 0 |
| Wash fill | 010 | 1 | 1 | 1 | 0 | 0 |
| Wash spin | 011 | 1 | 0 | 0 | 1 | 0 |
| Drain | 100 | 1 | 0 | 0 | 0 | 1 |
| Rinse fill | 101 | 1 | 1 | 0 | 0 | 0 |
| Rinse spin | 110 | 1 | 0 | 0 | 1 | 0 |
| Drain | 000 | 1 | 0 | 0 | 0 | 1 |
| 111 | 1 | 0 | 0 | 0 | 1 |
| Spin dry | 000 | 1 | 0 | 0 | 1 | 1 |



Figure: Simulation timing diagram of a washer controller output logic showing its behaviour when tested with a range of possible inputs.

Analysis: The output of the device is as expected for both the simulation and test on FPGM board.

**Washer controller next state logic:**

Truth table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | | Inputs | |
| Function | State | Next State | Door Open | Control |
| Load clothes | 000 | 000 | 1 | X |
| 000 | 000 | 0 | 0 |
| 001 | 000 | 0 | 1 |
| Wash fill | 010 | 001 | X | X |
| Wash spin | 011 | 010 | X | X |
| Drain | 100 | 011 | X | X |
| Rinse fill | 101 | 100 | X | X |
| Rinse spin | 110 | 101 | X | X |
| Drain | 000 | 110 | X | 0 |
| 111 | 110 | X | 1 |
| Spin dry | 000 | 111 | X | X |

Figure: Simulation timing diagram of a washer controller next state logic showing its behaviour when tested with a range of possible inputs.

Analysis: The output of the device is as expected for both the simulation and test on FPGM board.

**Washer controller(provided):**

Truth table:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | | Inputs | | Outputs | | | | |
| Function | State | Next State | Door Open | Control | Door Lock | Water Pump | Soap | Rotate Drum | Drain |
| Load clothes | 000 | 000 | 1 | X | 0 | 0 | 0 | 0 | 0 |
| 000 | 000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 001 | 000 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Wash fill | 010 | 001 | X | X | 1 | 1 | 1 | 0 | 0 |
| Wash spin | 011 | 010 | X | X | 1 | 0 | 0 | 1 | 0 |
| Drain | 100 | 011 | X | X | 1 | 0 | 0 | 0 | 1 |
| Rinse fill | 101 | 100 | X | X | 1 | 1 | 0 | 0 | 0 |
| Rinse spin | 110 | 101 | X | X | 1 | 0 | 0 | 1 | 0 |
| Drain | 000 | 110 | X | 0 | 1 | 0 | 0 | 0 | 1 |
| 111 | 110 | X | 1 | 1 | 0 | 0 | 0 | 1 |
| Spin dry | 000 | 111 | X | X | 1 | 0 | 0 | 1 | 1 |

Figure: Simulation timing diagram of the Washer controller showing its behaviour when tested with a range of possible inputs.

Analysis: The output of the device is as expected for both the simulation and test on FPGM board.

**Discussion**

All devices that were tested matched their truth tables for all stimuluses’ input. In the case of the N-bit synchronous counter with enable it was able to be simulated once an inconsistency in the name of the incrementer given by the engineer and that provided code was rectified. In future labs further efforts should be made to check for possible inconsistencies when defining the name of the files at the beginning of the implementation process. The two mod-m counters both count up from zero while the enable is high until they reach the same level as the m\_value and revert to “0000”. As soon as the asynchronous counter reaches the value of m\_value it resets however the synchronous counter waits till the next clock pulse this is because the asynchronous counters comparator is connected directly to the reset input which is not clock dependant unlike the sync counter that has it connected to the load/count input.

# Conclusion