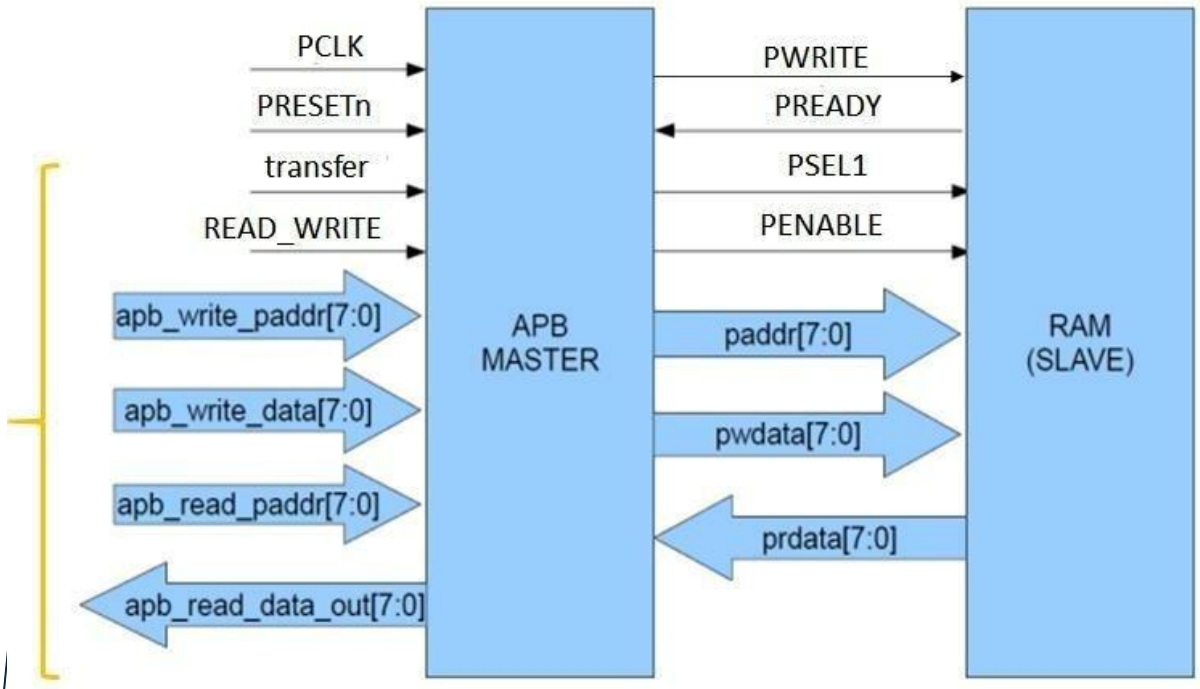
Implementation of

APB Interface

(Master/Slave)



Mounir Essam Mounir Ahmed.

Digital Logic Design.

ITI intake 46.

Introduction:

Implementation of APB Interface (Master/Slave) Using Combinational and Sequential Logic Circuits.

The Advanced Peripheral Bus (APB) is a simplified interface used for connecting peripherals to a system in embedded systems, particularly in SoCs (System-on-Chip). It is designed for low-bandwidth, low-latency communication, making it ideal for simpler, slower peripherals such as timers, UARTs, and GPIOs. The APB protocol consists of basic handshaking signals and provides simple, single-cycle access to devices.

The APB bus uses the following primary signals:

- PADDR: Address bus.
- PWDATA: Write data bus.
- PRDATA: Read data bus.
- PSEL: Select signal, used to identify which peripheral is being accessed.
- PWRITE: Write control signal (indicates write operation).
- PENABLE: Enables the transfer cycle.
- PREADY: Slave ready signal, indicating the completion of the operation.

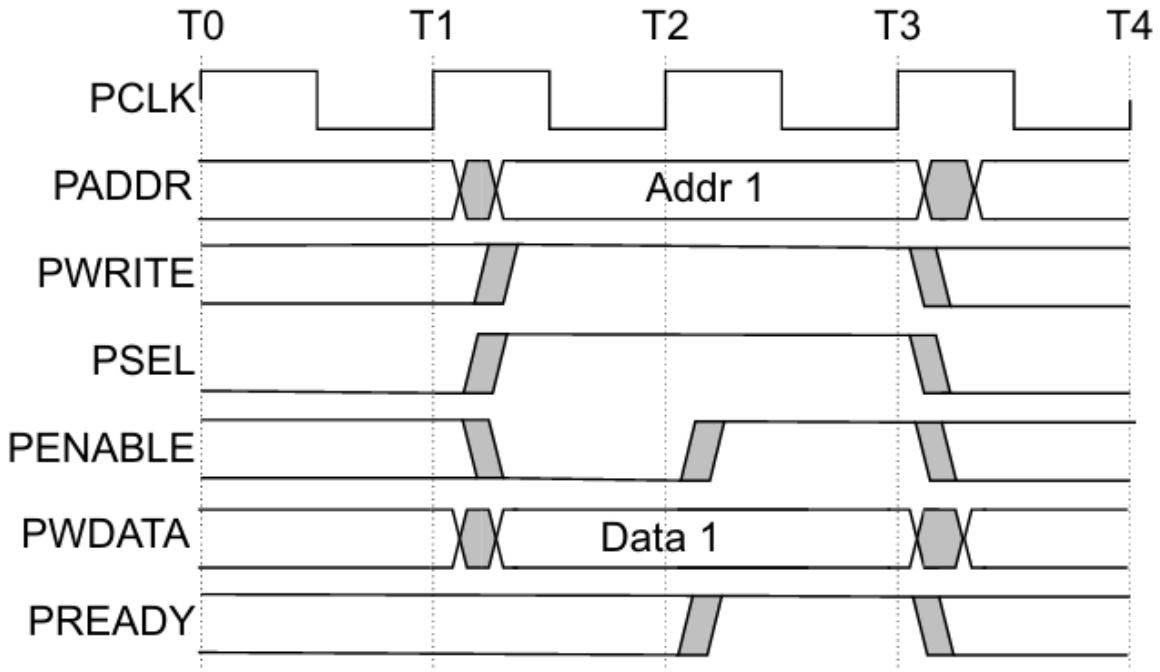


Figure 1: Write transfer with no wait states

Project Milestones:

Milestone 1: Design and Implement the Master Interface

1: FSM

I started to calculate an FSM for the APB interface according to the standard.

I choose the first type of APB Write and Read with no wait states.

I use this FSM diagram:

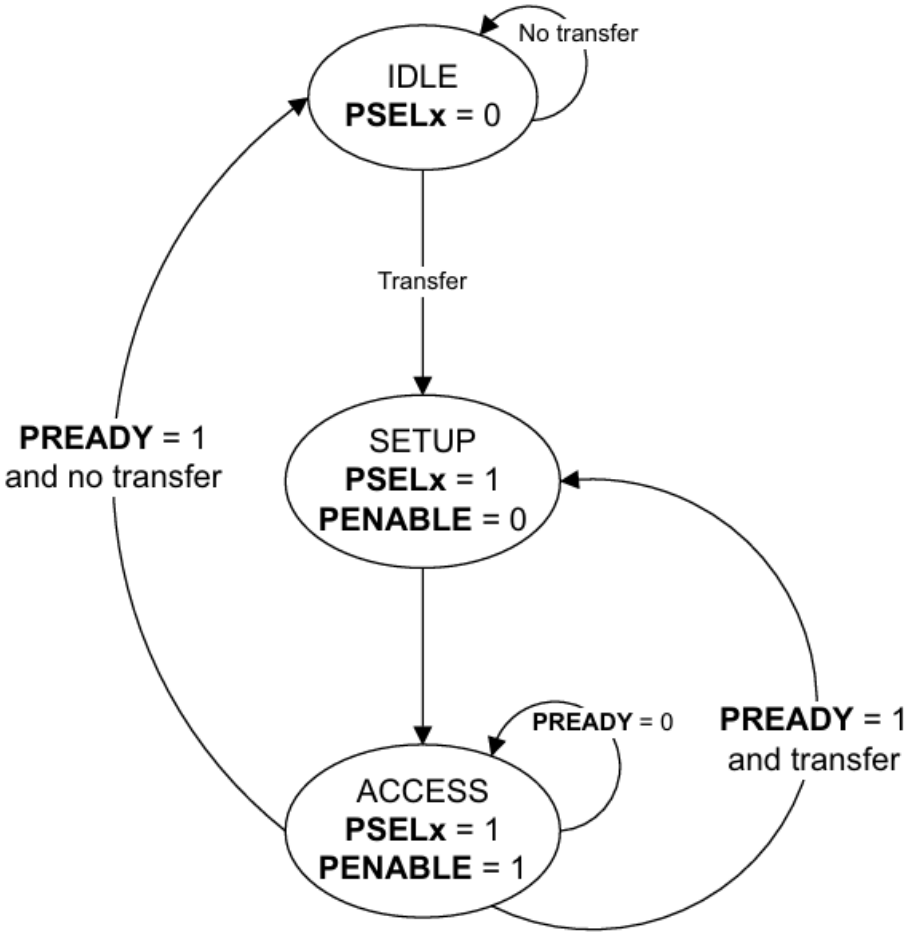


Figure -2 State diagram.

First, I put my truth table and detect the current state and next state and detect the output according to change in the input. I have used a MOORE FSM .

index	Current State		Inputs		Next State		Outputs	
	Q_1	Q_O	Transfer	PREADY	Q_1+	Q_O+	PSEL	PENABLE
0	0	0	0	0	0	0	0	X
1	0	0	0	1	0	0	0	X
2	0	0	1	0	0	1	0	X
3	0	0	1	1	0	1	0	x
4	0	1	0	0	1	0	1	0
5	0	1	0	1	1	0	1	0
6	0	1	1	0	1	0	1	0
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	0	0	1	1
10	1	0	1	0	1	0	1	1
11	1	0	1	1	0	1	1	1
12	1	1	0	0	X	X	X	X
13	1	1	0	1	X	X	X	X
14	1	1	1	0	X	X	X	X
15	1	1	1	1	x	x	x	x

Then make a K-Map and get the equations of Q_1+ , Q_O+ , PSEL and PENABLE.

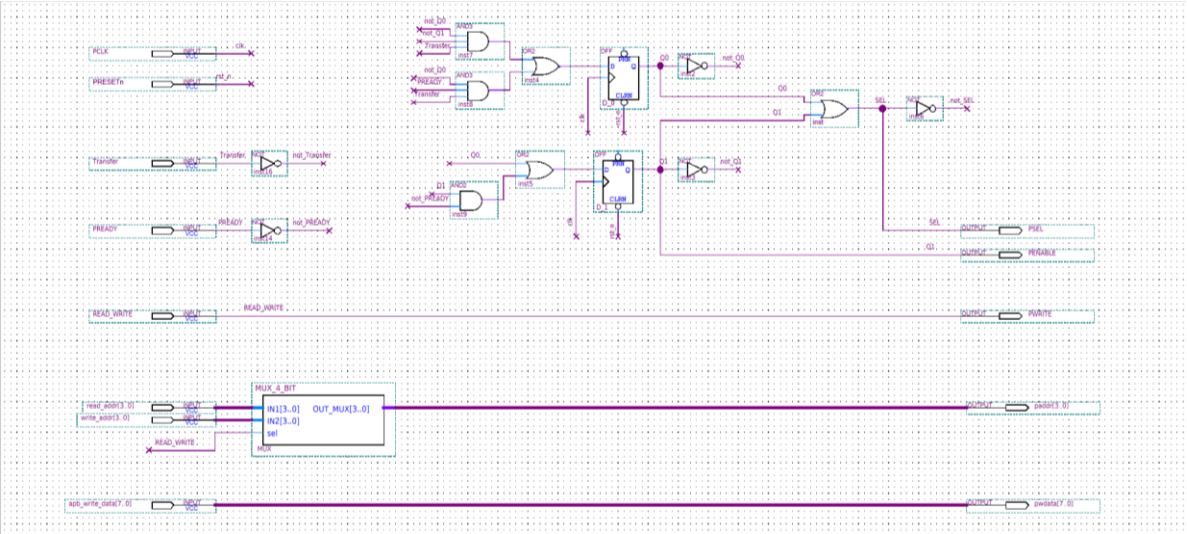


Figure -3 Master Logic Gates.

And implement it using logic gates and flip flops at Quartus.

2: MUX

Then I have started to make a MUX that take an input with 4-bits to choose what will be the address should I take:

If I want to write I send a signal (READ_WRITE) with (1) so choose (Write_addr [3:0]).

If I want to read, I send a signal (READ_WRITE) with (0) so choose (read_addr [3:0]).

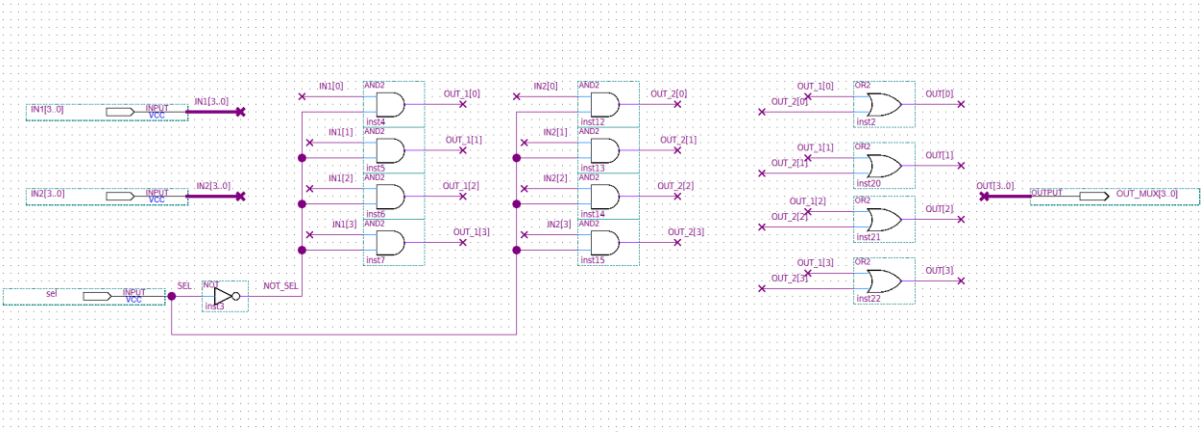


Figure -4 MUX 4-bit Logic Gates.

3: Simulation

Then, I have started to make a simulation using the waveform at Quartus.

I added signals manually and put also inputs manually and then see the expected output.

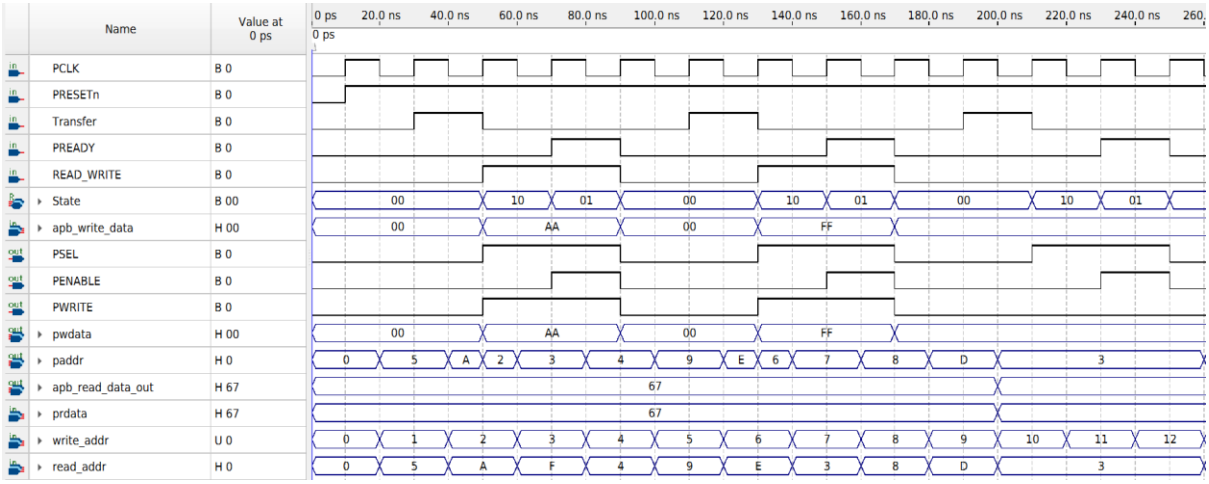


Figure -5 Master Waveform.

Milestone 2: Design and Implement the Slave Interface

1: RAM

First, I have started to make a RAM to store the Write Data and its size is 16 Rows x 8 Cols.

So, I have started to make a 8-bits Register and instantiate it 16 times to make a RAM.

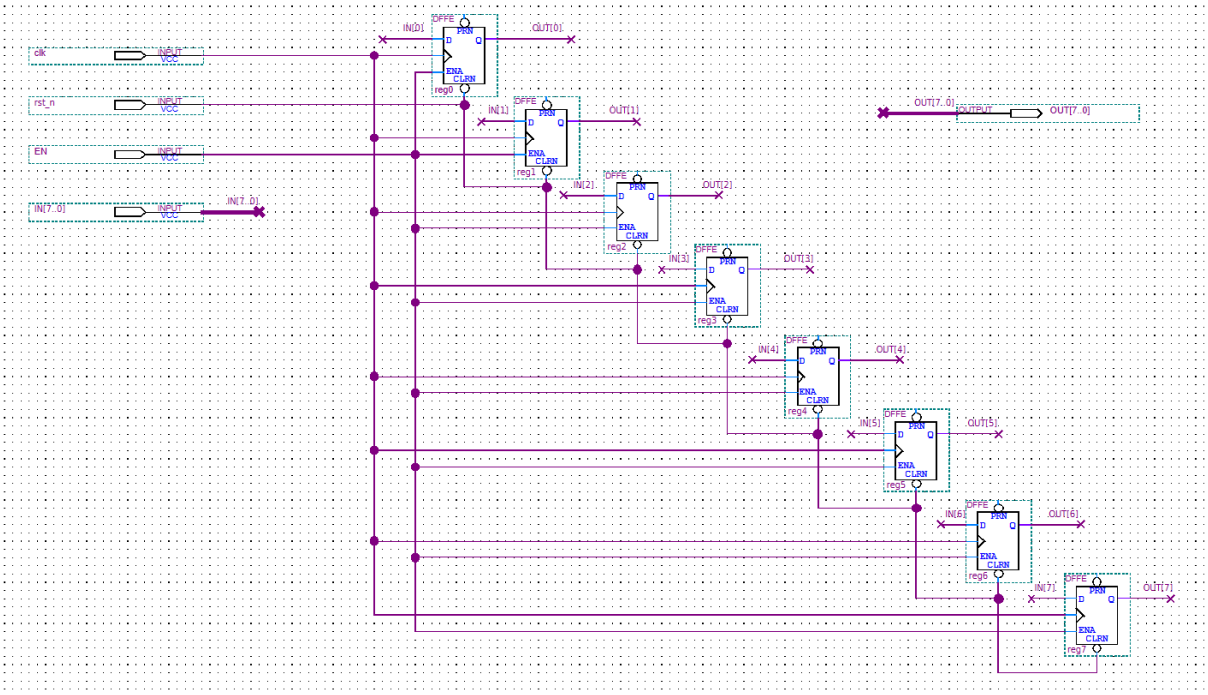


Figure -6 Register store 8-bits data.

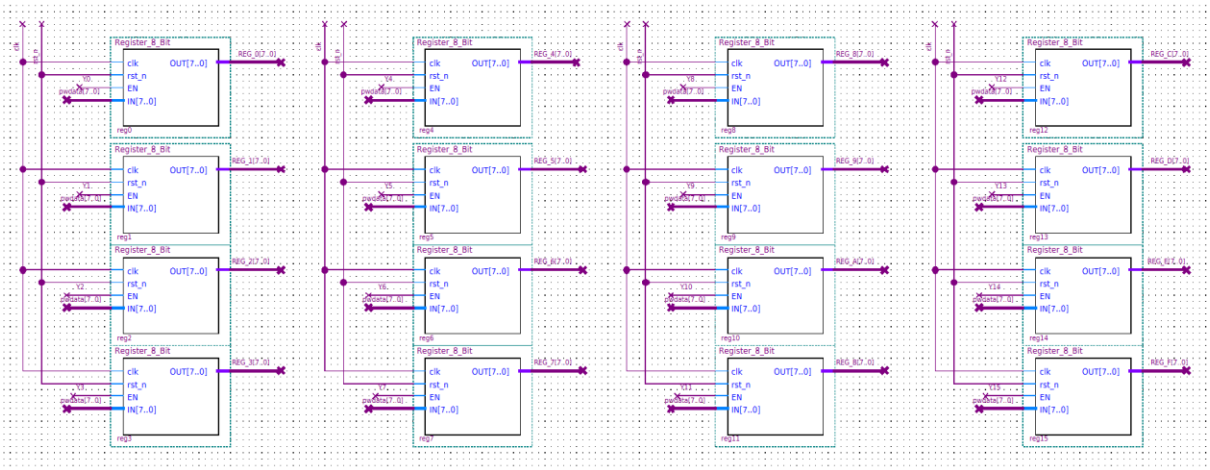


Figure -7 RAM with 16 (8-bits Register).

2: Decoder

Then I have started to make a 4-to-16 Decoder to enable how to detect which Register I should store in it according to the address.

First, I made a decoder 2-to-4 then make the biggest one 4-to-16 from it.

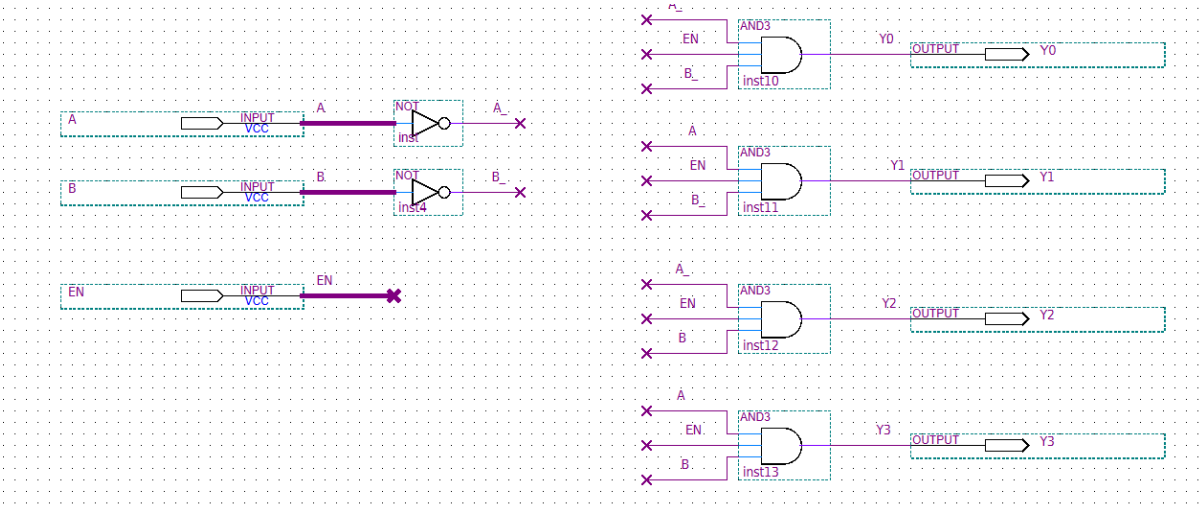


Figure -8 Decoder 2-to-4.

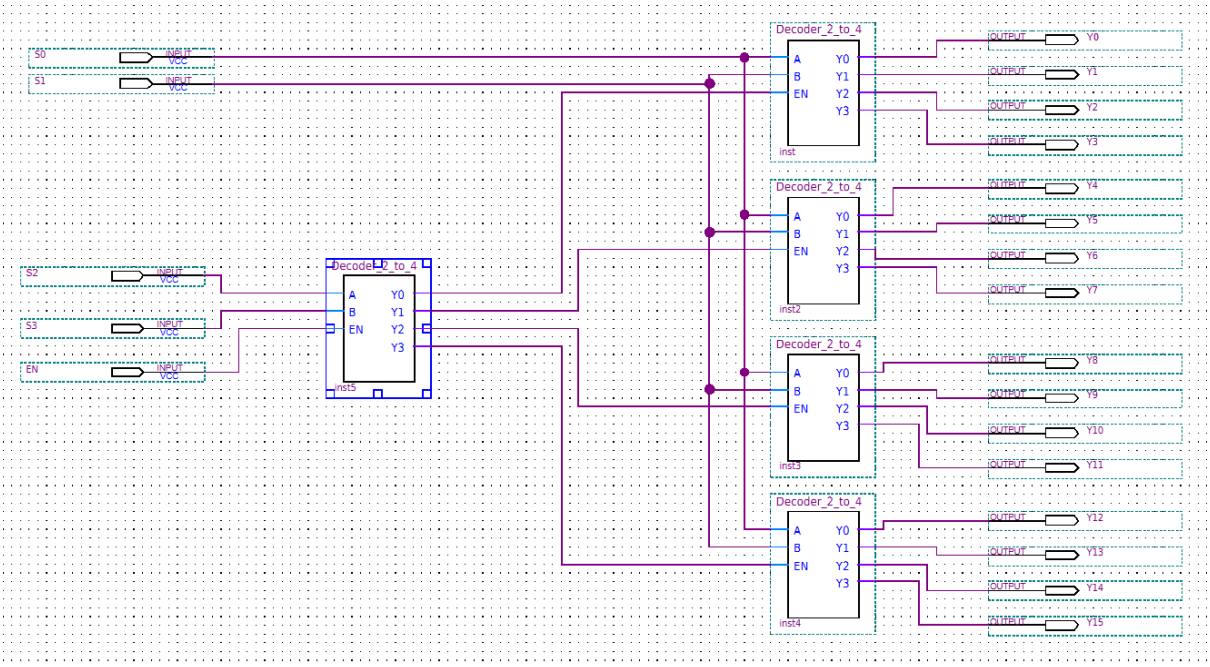


Figure -9 Decoder 4-to-16.

3:MUX

Then I have started to implement a MUX to select which data should be go through the output of the Slave according to address and read enable.

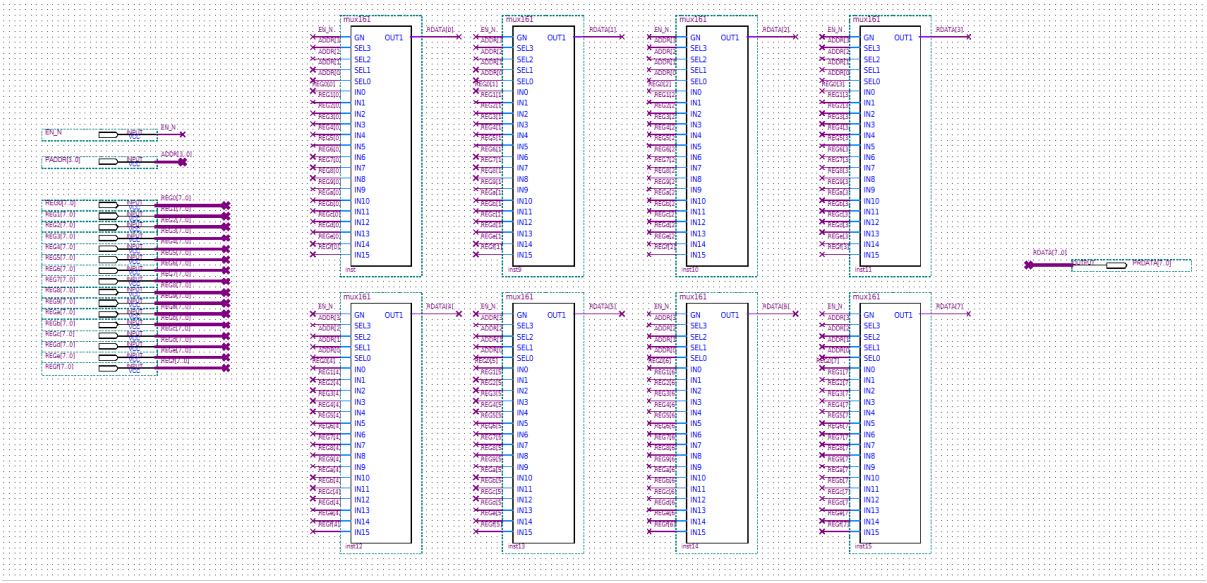


Figure -10 Mux to select between 16 data out each one with 8-bits.

Then I have integrated all blocks of the design to make a Slave.

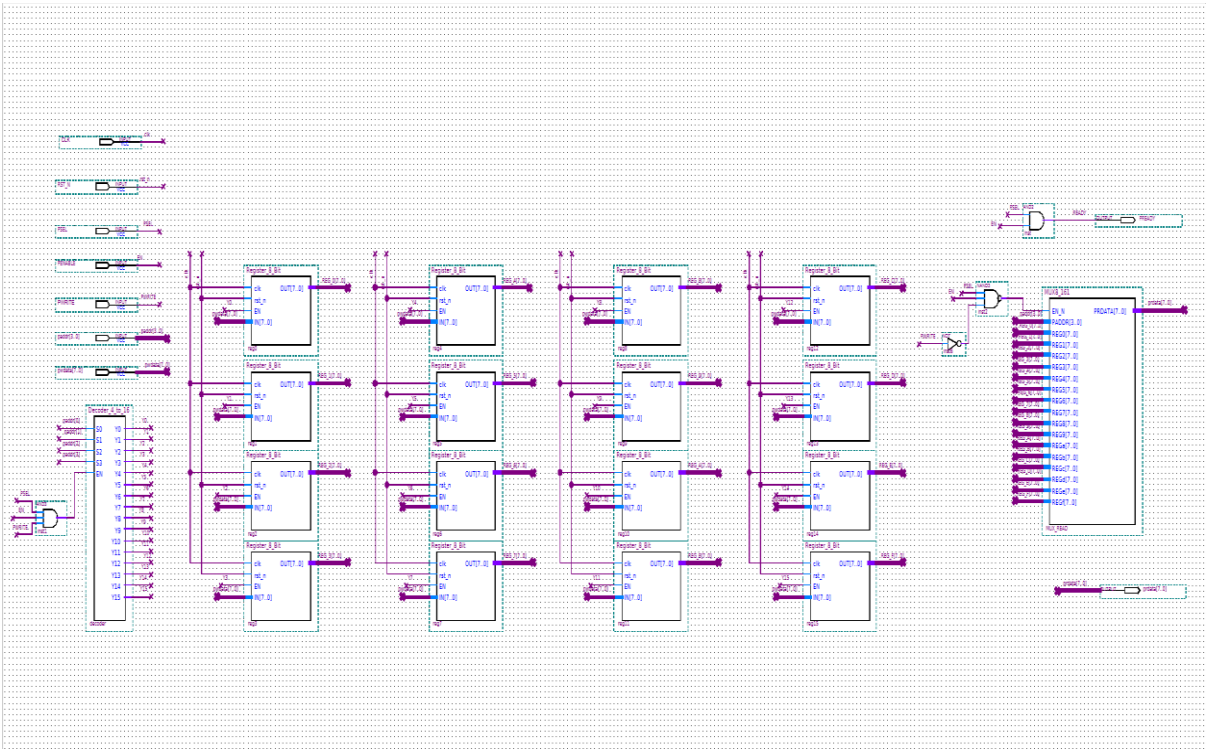


Figure -11 Slave Logic Design.

4:Simulation

Then, I have started to make a simulation using the waveform at Quartus.

I added signals manually and put also inputs manually and then see the expected output.

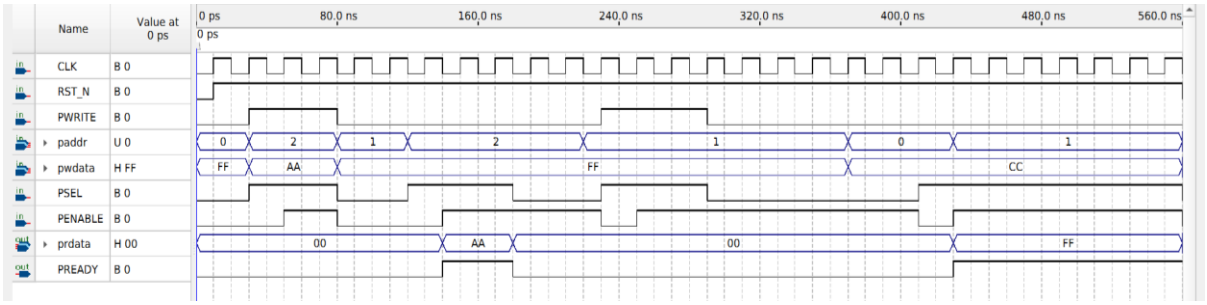


Figure –12 Slave Waveform.

Milestone 3: Full System Integration and Final Simulation

After all of this, I made the whole system and integrate the master and slave.

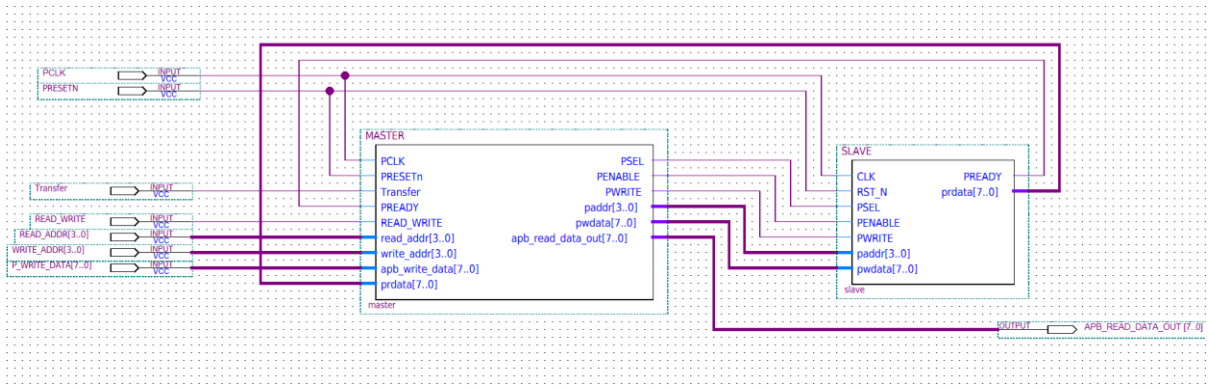


Figure –13 APB Interface.

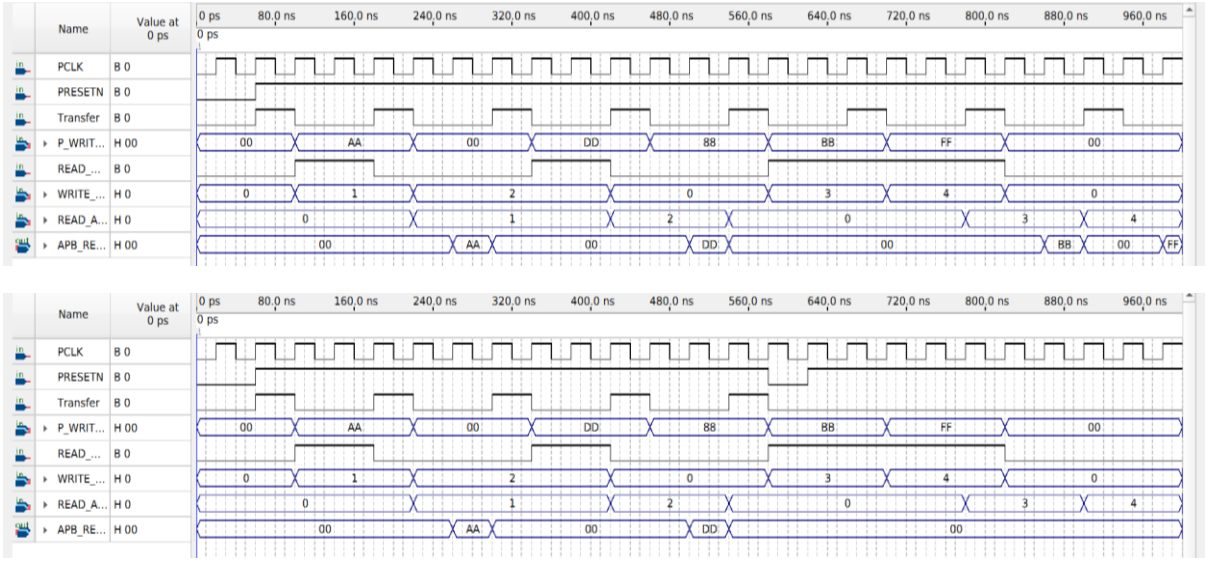


Figure –14 APB Waveform.