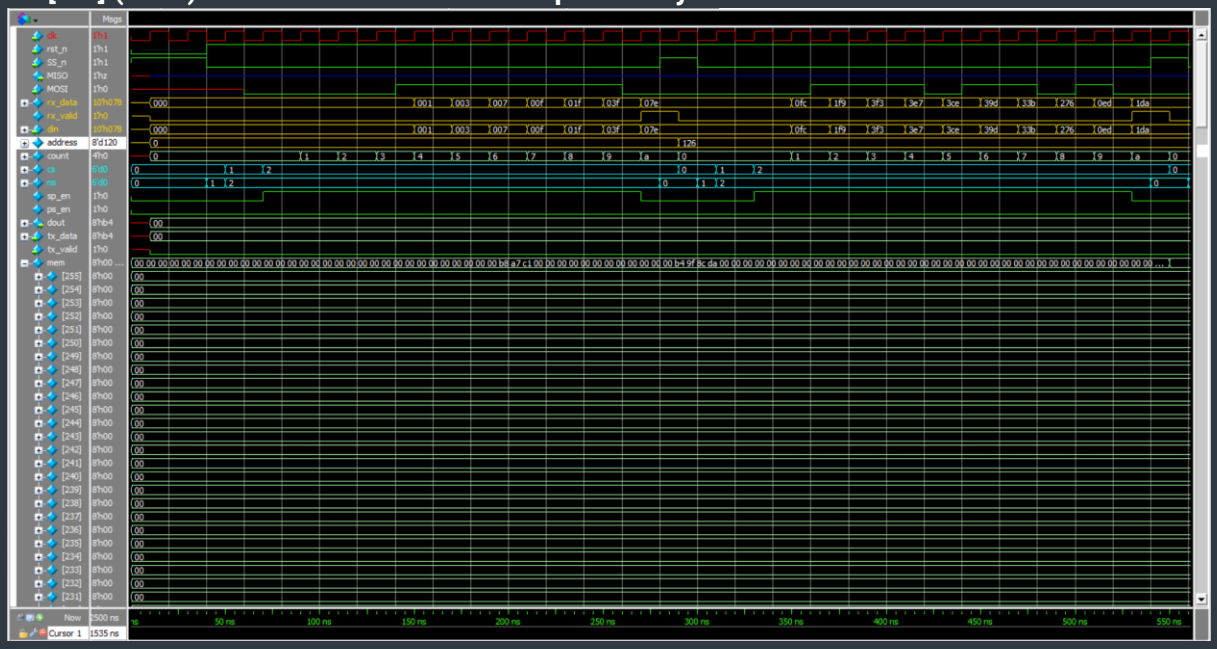
Project(2): SPI Slave with Single Port RAM
Team Name: Silicon Maestro
Team Member(solo): Amr Ayman Mohamed Abdo

1. QuestaSim Snippets

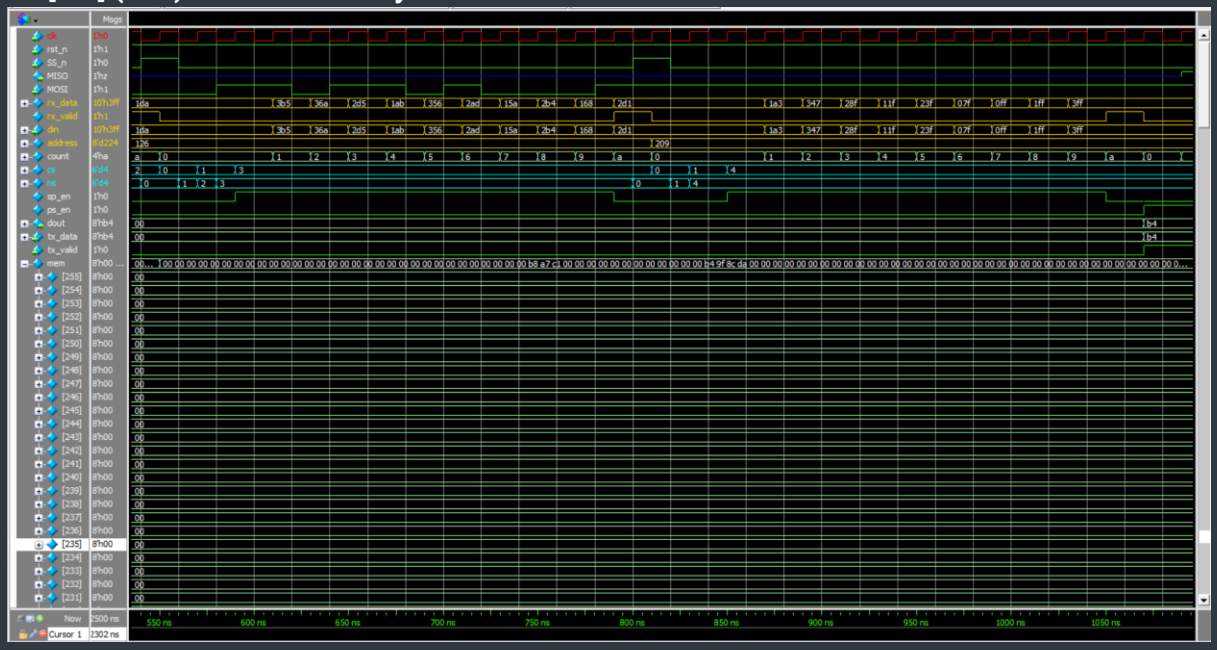
i. First Communication Cycle (cs = WRITE, din[9:8] = 00): MOSI is converted from serial to parallel and after 10 clock cycles rx_data (din) gets the parallel data, rx_valid is asserted and din[7:0] (126) is held internally in the RAM as a write address.

ii. Second Communication Cycle (cs = WRITE, din[9:8] = 01): MOSI is converted from serial to parallel and after 10 clock cycles rx_data (din) gets the parallel data, rx_valid is asserted and din[7:0] (0xda) is written into the address previously held in the address bus.

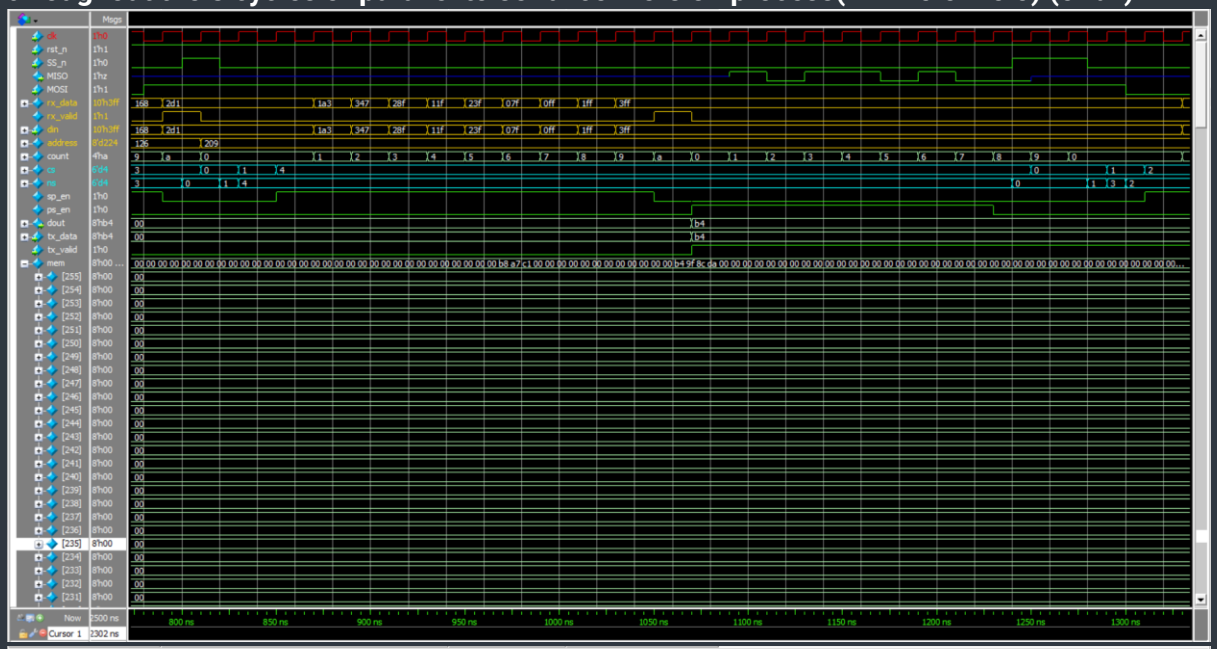


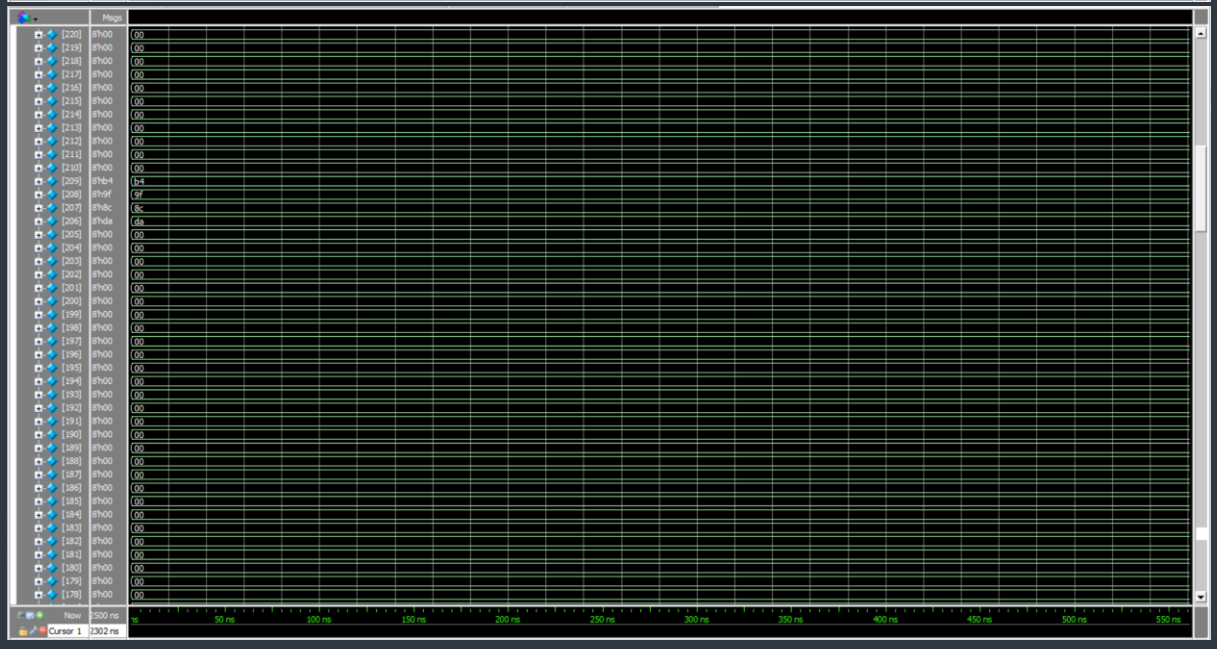
0xda after being written in address 126

iii. Third Communication Cycle (cs = READ_ADD, din[9:8] = 10): MOSI is converted from serial to parallel and after 10 clock cycles rx_data (din) gets the parallel data, rx_valid is asserted and din[7:0] (209) is held internally in the RAM as a read address.



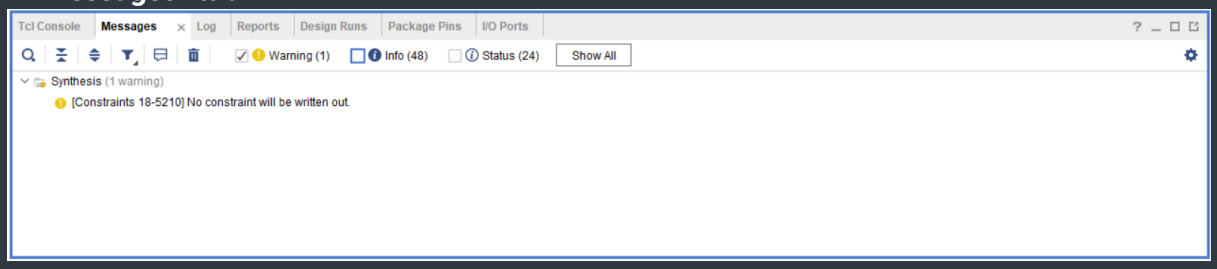
iii. Third Communication Cycle (cs = READ_DATA, din[9:8] = 11): MOSI is converted from serial to parallel and after 10 clock cycles rx_data (din) gets the parallel data, rx_valid is asserted and after another clock cycle the RAM sends the data read from address 209 held previously and tx_valid is asserted. Parallel to serial conversion takes place and MISO gets the tx_data in a serial manner throughout the 8 cycles of parallel to serial conversion process(1 1 1 0 0 1 0 0) (0xb4)



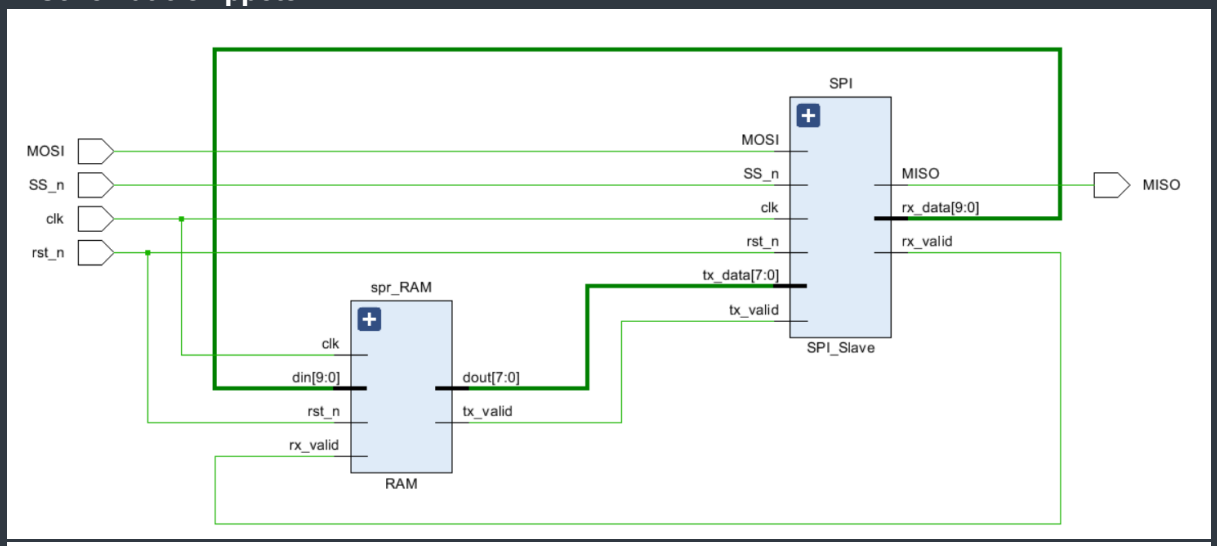


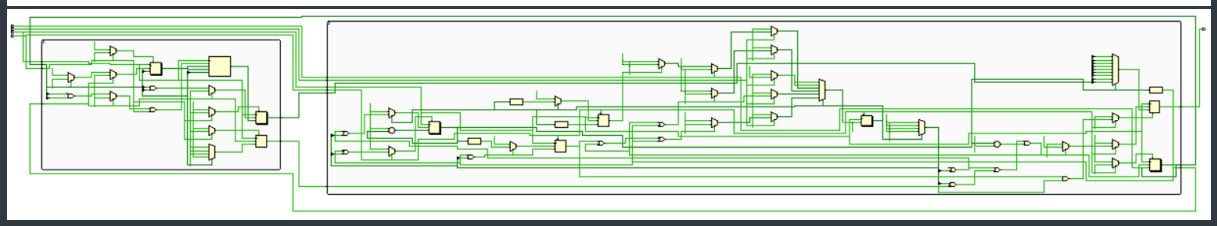
2. Elaboration

i. “Messages” tab



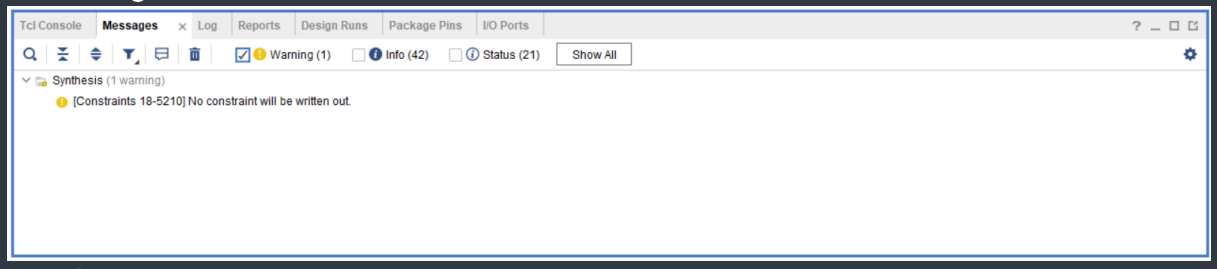
ii. Schematic snippets





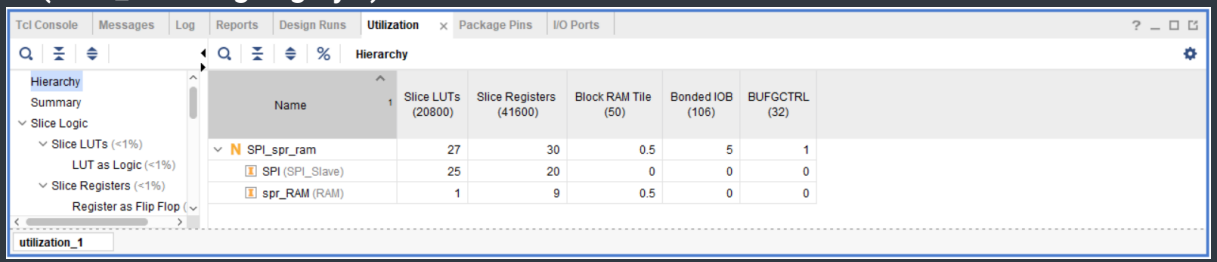
3. Synthesis

i. “Messages” tab

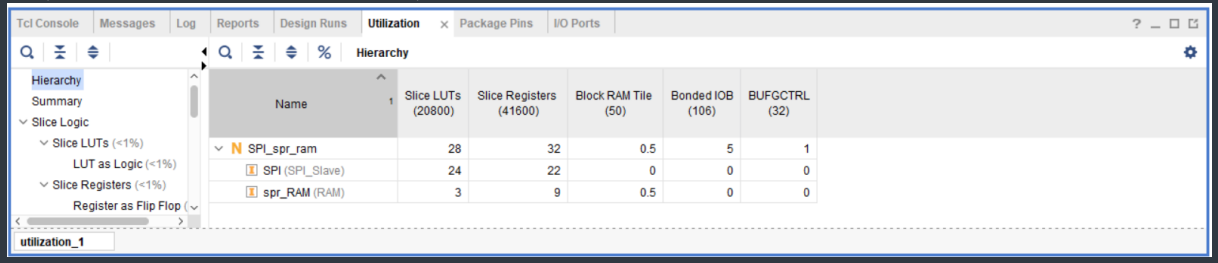


ii. Utilization report

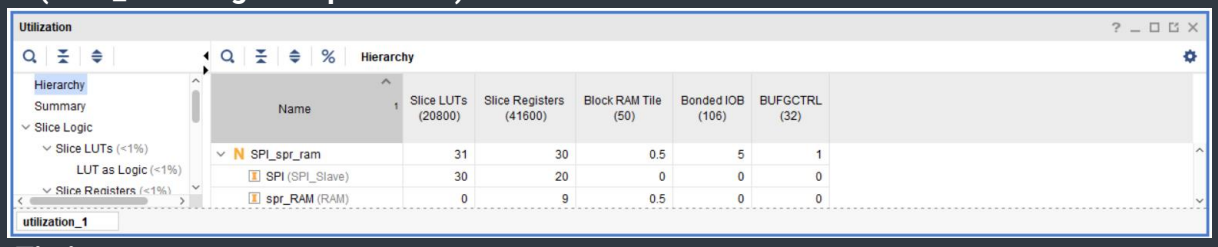
a. (*fsm_encoding = "gray"*)



b. (*fsm_encoding = "one_hot"*)

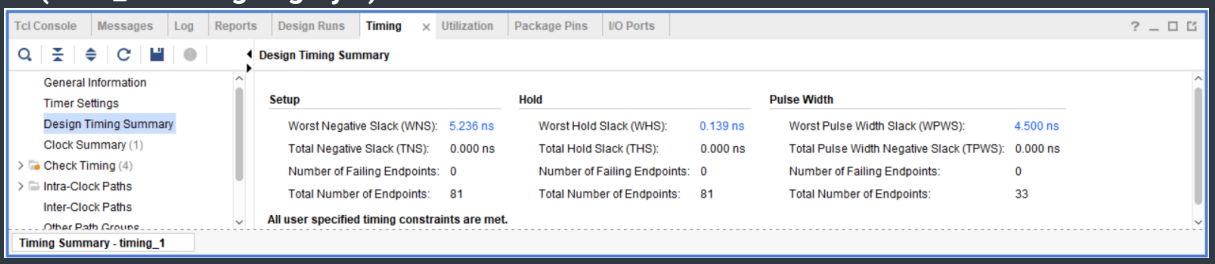


c. (*fsm_encoding = "sequential "*)

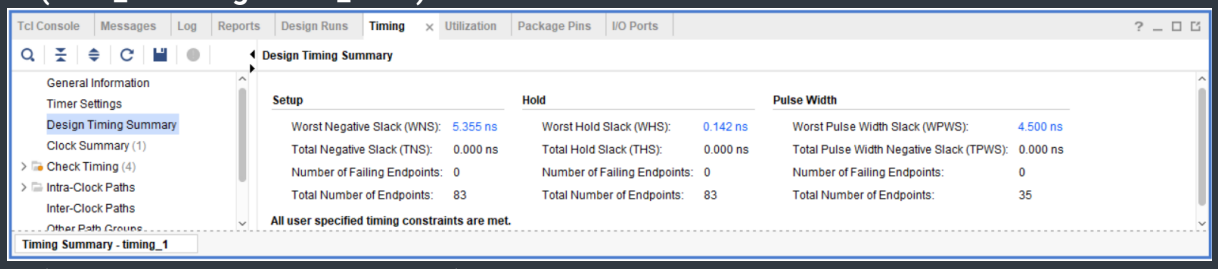


iii. Timing report

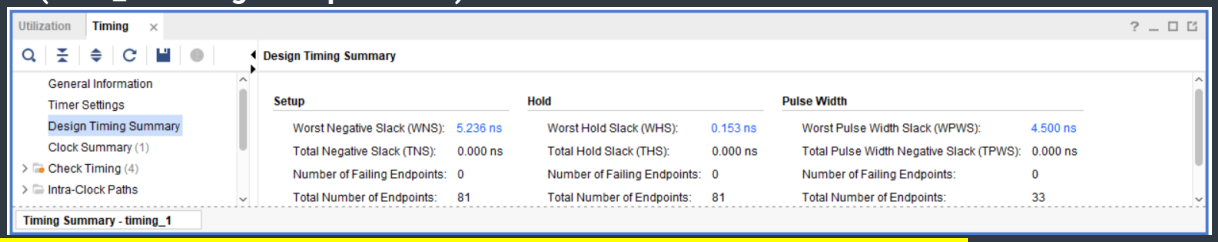
a. (*fsm_encoding = "gray"*)



b. (*fsm_encoding = "one_hot"*)



c. (*fsm_encoding = "sequential "*)



All of them are almost the same. We will proceed with sequential encoding.

iv. Synthesis report showing the encoding used

a. (*fsm_encoding = "gray"*)

```
INFO: [Synth 8-5534] Detected attribute (* fsm_encoding = "gray" *) [C:/Digital_Electronics/Kareem_Waseem_Digital_Design_Diploma/Project2_SPI_Slave/SPI_Slave.v:15]
INFO: [Synth 8-6155] done synthesizing module 'SPI_Slave' (1#1) [C:/Digital_Electronics/Kareem_Waseem_Digital_Design_Diploma/Project2_SPI_Slave/SPI_Slave.v:1]
INFO: [Synth 8-6157] synthesizing module 'RAM' [C:/Digital_Electronics/Kareem_Waseem_Digital_Design_Diploma/Project2_SPI_Slave/RAM.v:1]
```

State	New Encoding	Previous Encoding
IDLE	001	000000
CHK_CMD	000	000001
READ_DATA	011	000100
READ_ADD	010	000011
WRITE	111	000010

b. (*fsm_encoding = "one_hot"*)

```
INFO: [Synth 8-5534] Detected attribute (* fsm_encoding = "one_hot" *) [C:/Digital_Electronics/Kareem_Waseem_Digital_Design_Diploma/Project2_SPI_Slave/SPI_Slave.v:15]
INFO: [Synth 8-6155] done synthesizing module 'SPI_Slave' (1#1) [C:/Digital_Electronics/Kareem_Waseem_Digital_Design_Diploma/Project2_SPI_Slave/SPI_Slave.v:1]
```

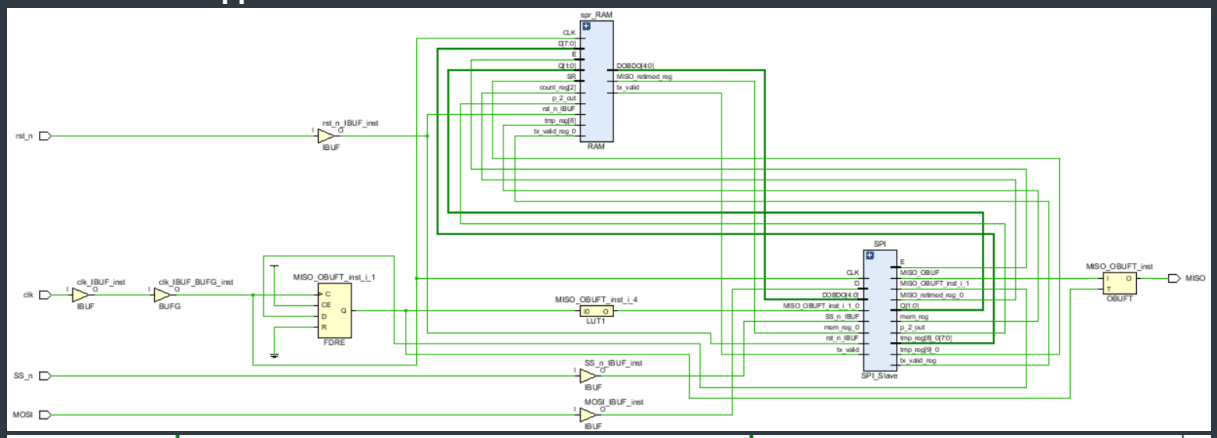
State	New Encoding	Previous Encoding
IDLE	00010	000000
CHK_CMD	00001	000001
READ_DATA	00100	000100
READ_ADD	01000	000011
WRITE	10000	000010

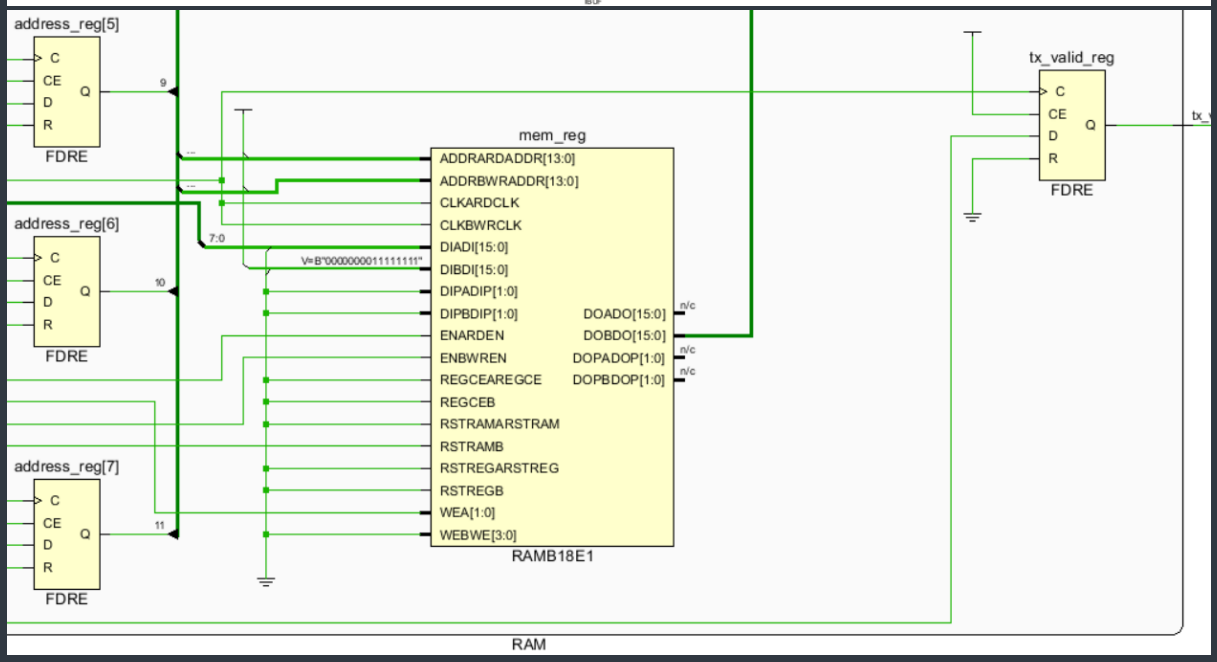
c. (*fsm_encoding = "sequential "*)

```
INFO: [Synth 8-5534] Detected attribute (* fsm_encoding = "sequential" *) [C:/Digital_Electronics/Kareem_Waseem_Digital_Design_Diploma/Project2_SPI_Slave/SPI_Slave.v:15]
INFO: [Synth 8-6155] done synthesizing module 'SPI_Slave' (1#1) [C:/Digital_Electronics/Kareem_Waseem_Digital_Design_Diploma/Project2_SPI_Slave/SPI_Slave.v:1]
```

State	New Encoding	Previous Encoding
IDLE	001	000000
CHK_CMD	000	000001
READ_DATA	010	000100
READ_ADD	011	000011
WRITE	100	000010

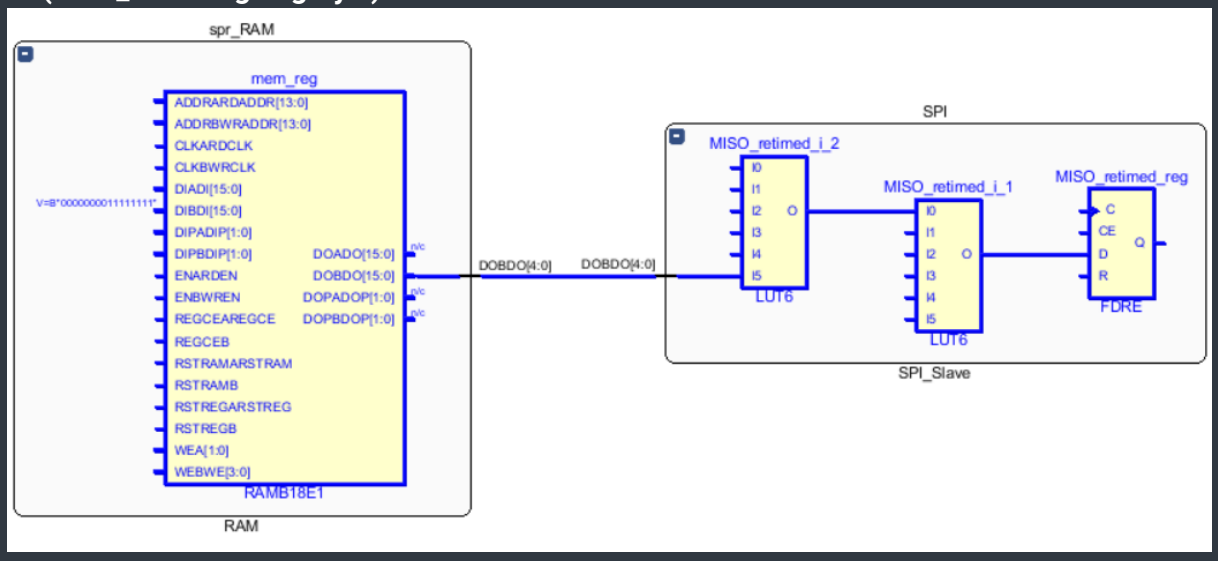
v. Schematic snippets



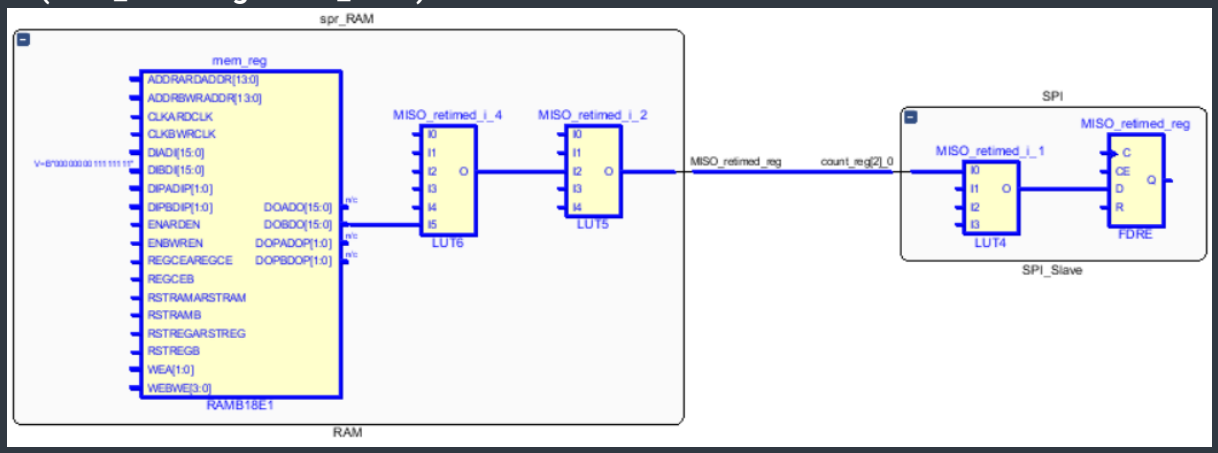


vi. Critical path snippets

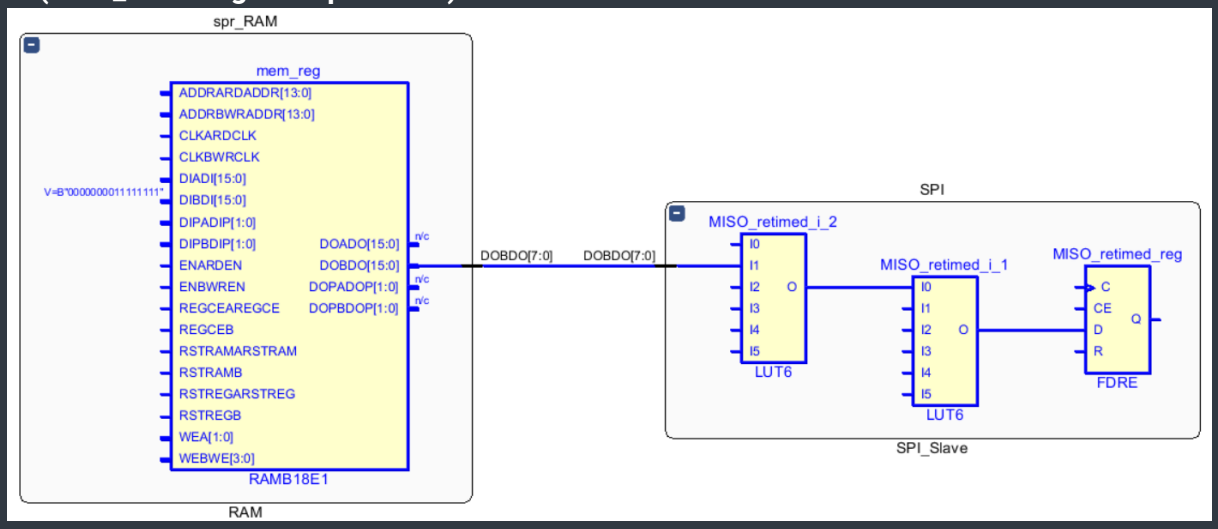
a. (*fsm_encoding = "gray"*)



b. (*fsm_encoding = "one_hot"*)

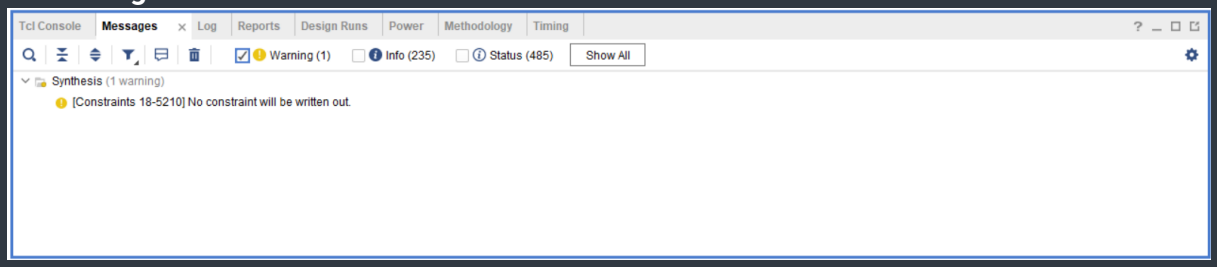


c. (*fsm_encoding = "sequential"*)

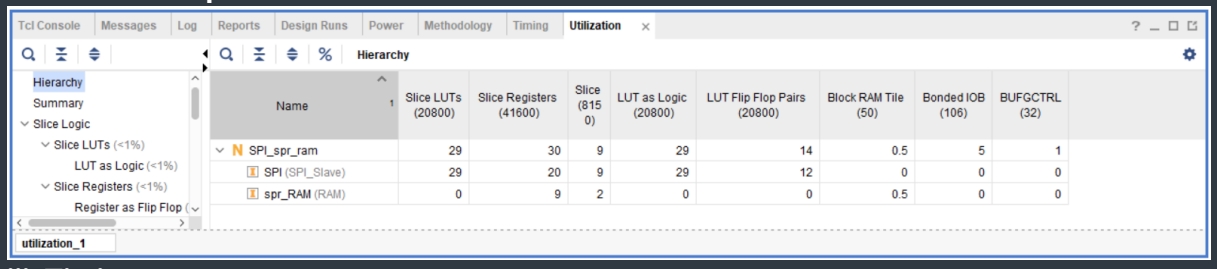


4. Implementation

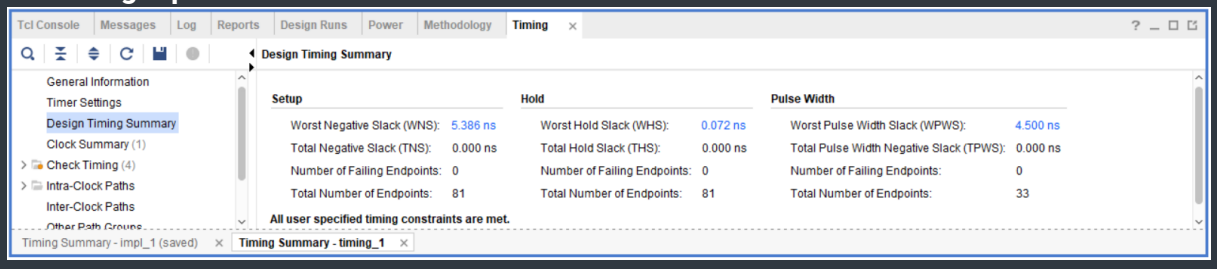
i. “Messages” tab



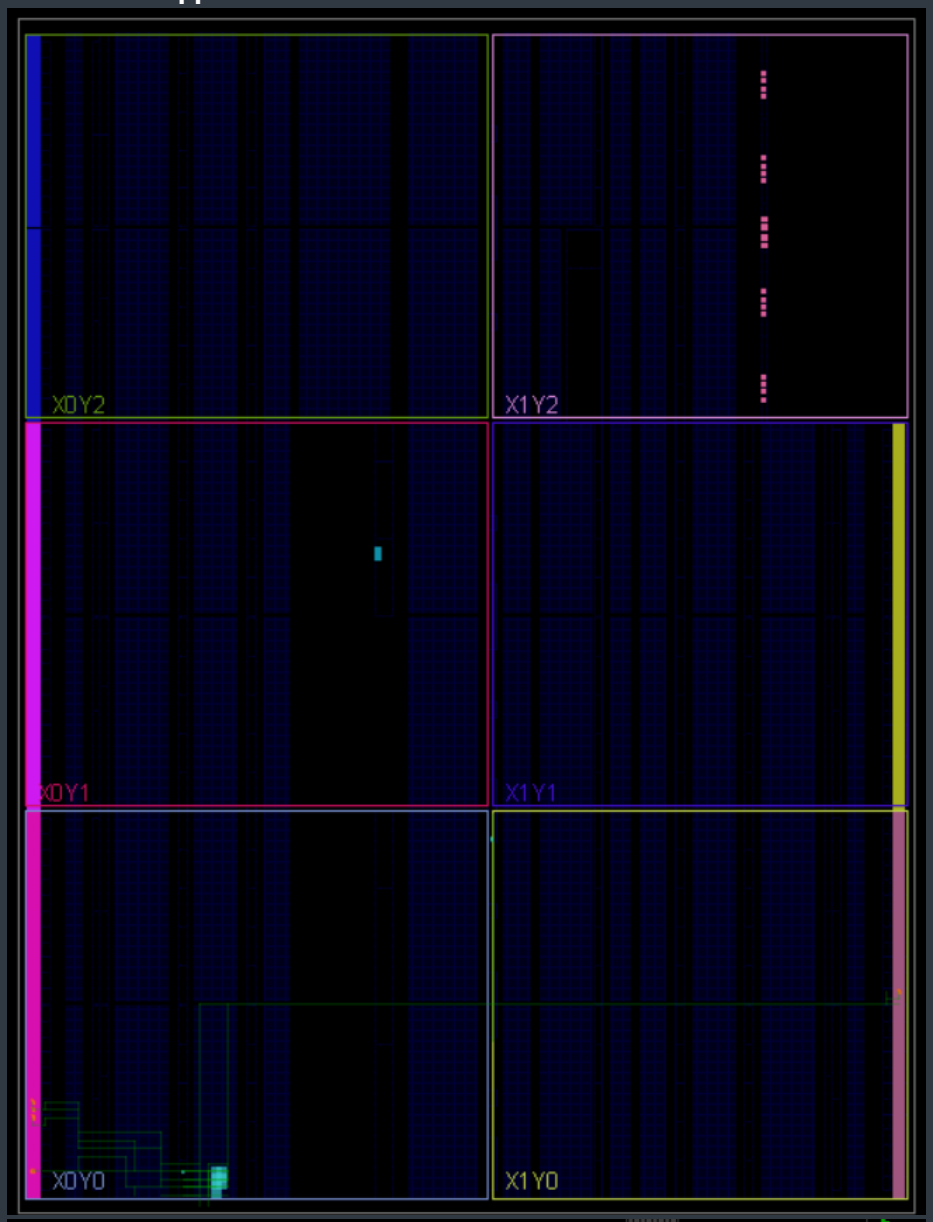
ii. Utilization report

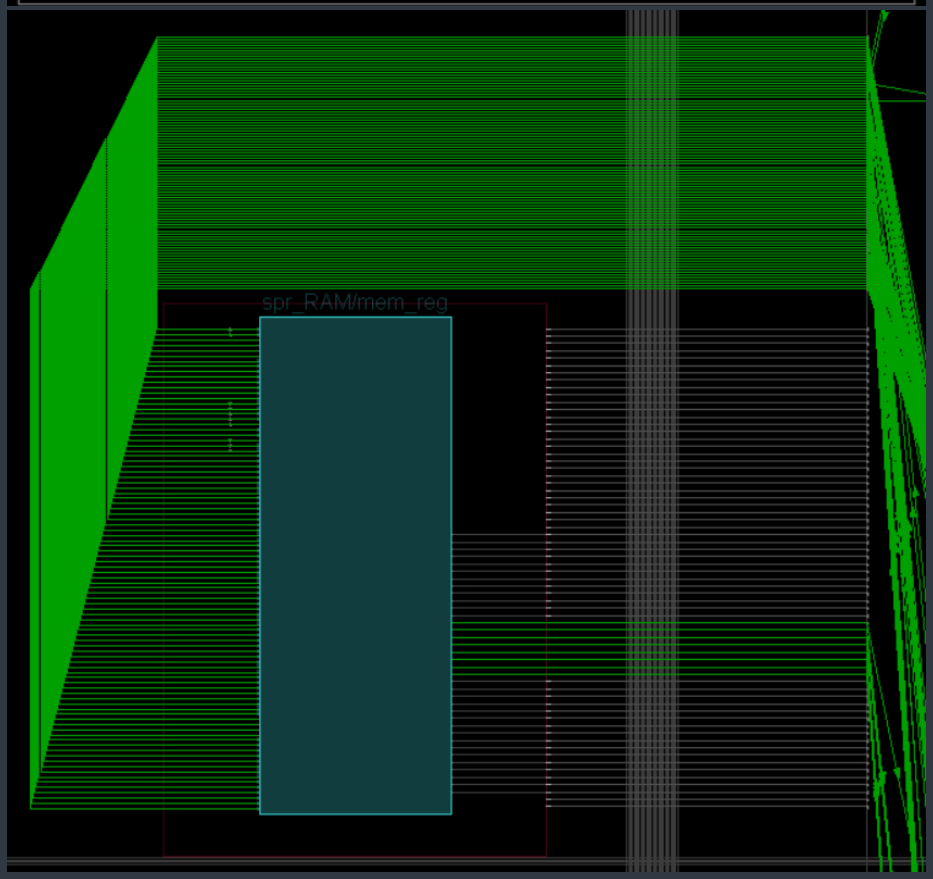


iii. Timing report

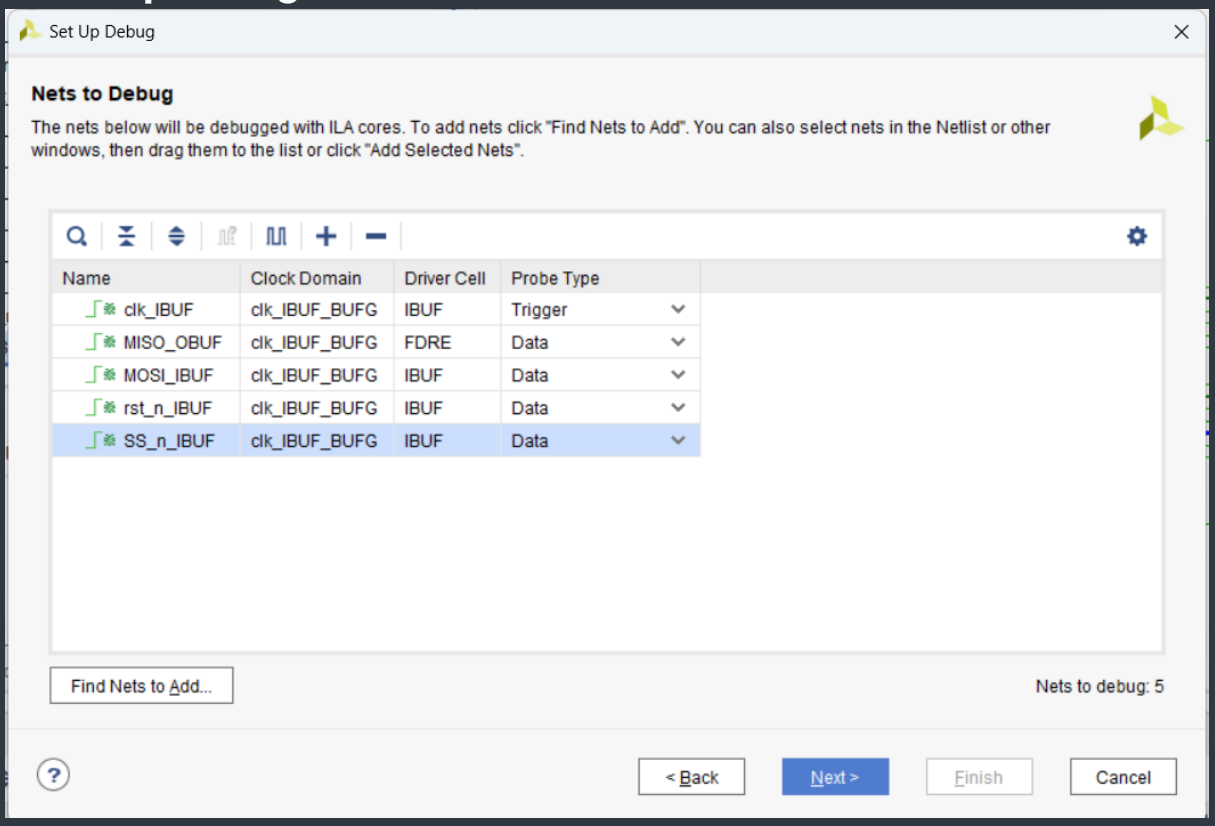


iv. Device snippets





5. Set Up Debug



6. Bitstream file