## Randomizer

|  |  |
| --- | --- |
| **Main File Name** | **Randi.vhd** |
| **Testbench File Name** | **Randi\_tb.vhd** |

### Ports:

|  |  |  |  |
| --- | --- | --- | --- |
| Signals | In/Out | Type | Width |
| Clk\_50MHz | In | std\_logic | 1 |
| Reset | In | std\_logic | 1 |
| randi\_input\_data | In | std\_logic | 1 |
| randi\_input\_ready | In | std\_logic | 1 |
| randi\_output\_valid | Out | std\_logic | 1 |
| randi\_output\_data | Out | std\_logic | 1 |
|  |  |  |  |

### Block Diagram Sketch:

A black background with a black square

Description automatically generated with medium confidence

### Testing and Functionality

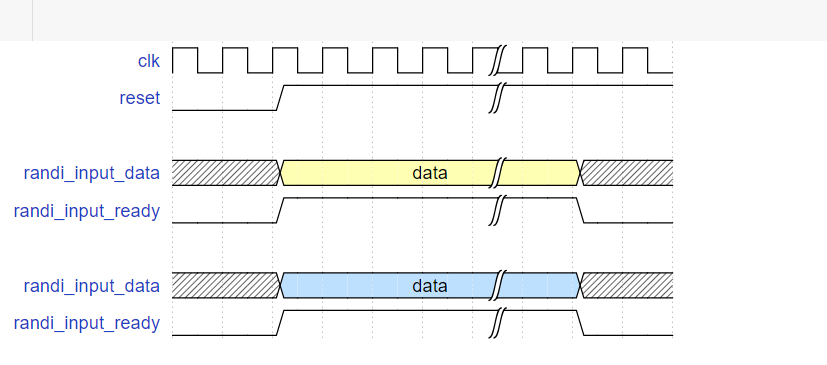
* We have 96 cycles, with a period of 40 ns = (3640 ns runtime in testbench).
* No Setup nor warm-up cycles
* Handshakes (ready and valid signals) are in phase with inputs/outputs.

### RTL (Quartus)

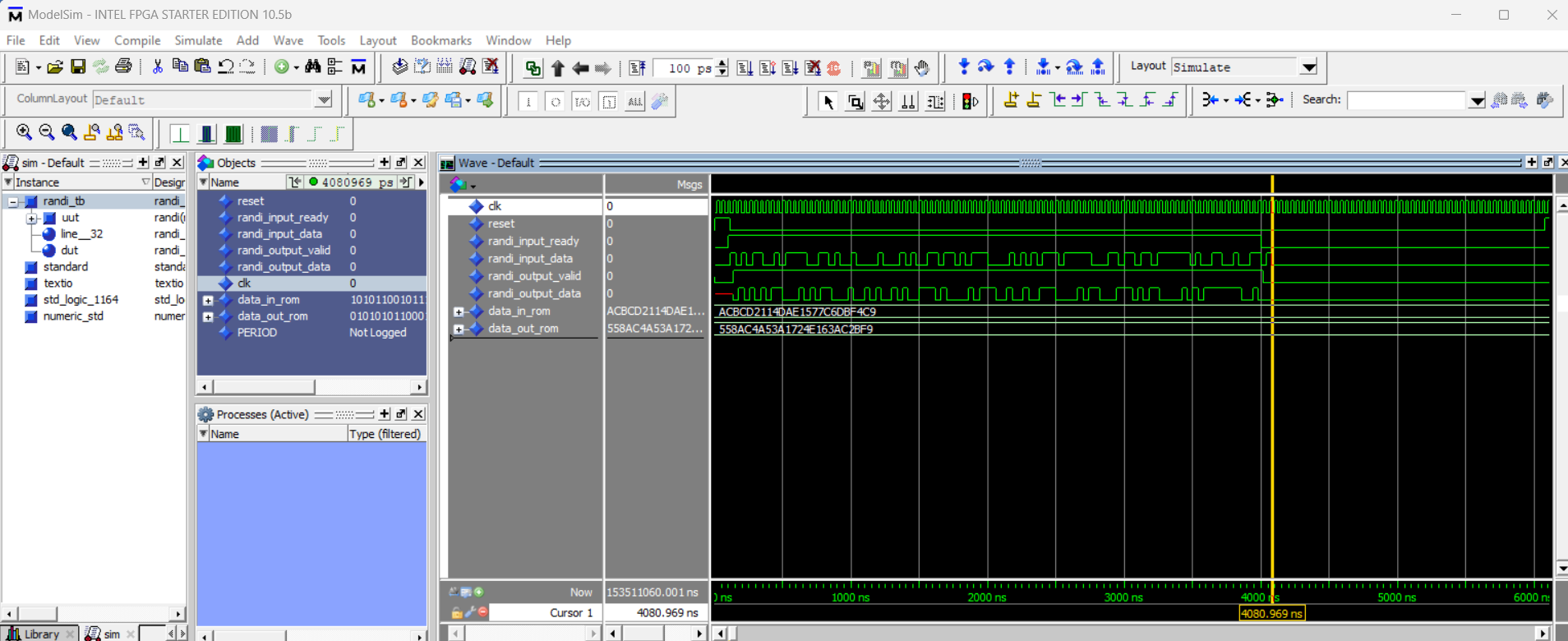
A diagram of a network

Description automatically generated

### Waveform (Sketch Expect)



### Results (ModelSim Altera)



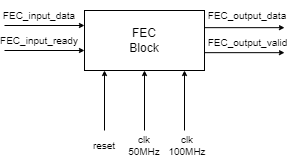
## FEC

|  |  |
| --- | --- |
| **Main File Name** | **FEC.vhd** |
| **Testbench File Name** | **FEC\_tb.vhd** |
| **Extra File** | **PPL\_BLOCK.vhd** |

### Ports:

|  |  |  |  |
| --- | --- | --- | --- |
| Signals | In/Out | Type | Width |
| clk\_50MHz | In | STD\_LOGIC | 1 |
| clk\_100MHz | In | STD\_LOGIC | 1 |
| reset | In | STD\_LOGIC | 1 |
| FEC\_input\_data | In | STD\_LOGIC | 1 |
| FEC\_input\_ready | In | STD\_LOGIC | 1 |
| FEC\_output\_valid | Out | STD\_LOGIC | 1 |
| FEC\_output\_data | Out | STD\_LOGIC\_VECTOR | 1 |

### Block Diagram Sketch:



### Testing and Functionality

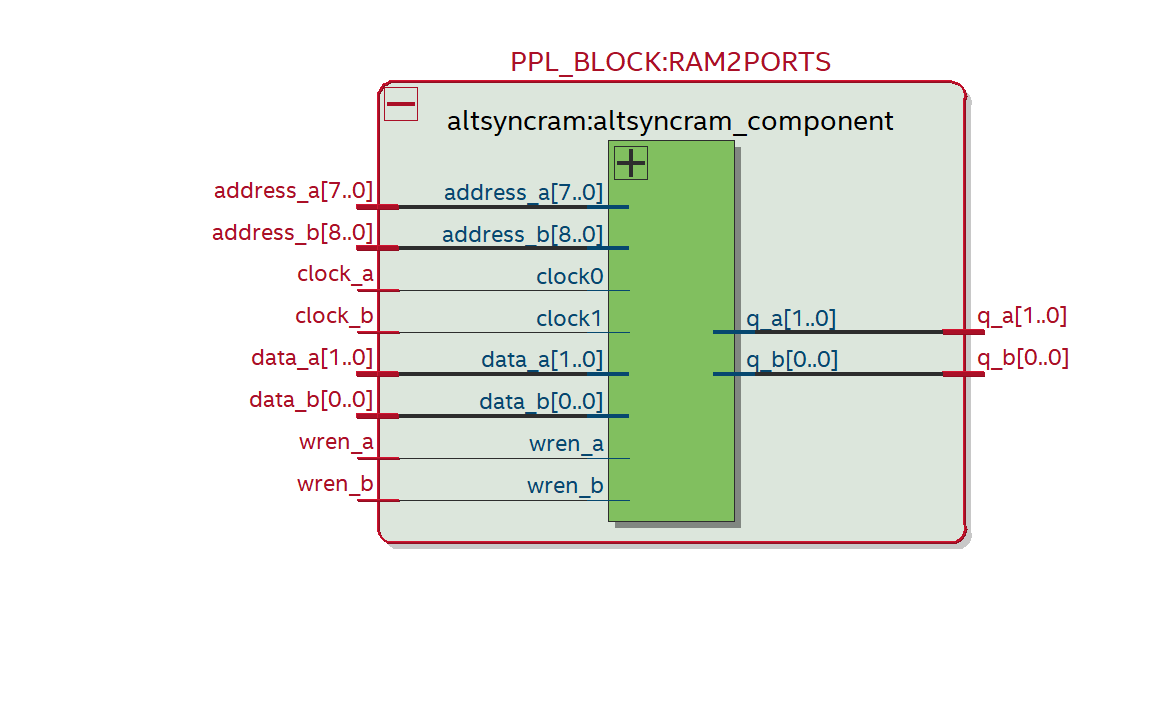
1. We receive 96 data bits at 50 MHz, at period 20 ns. They are processed in parallel and sent to our PLL\_BLOCK (RAM).
2. Then we transmit 192 bits of data a faster rate of 100 MHz.
3. We need to either send or receive not both. And repeats the cycle again.

### RTL (Quartus)

A drawing of a circuit board

Description automatically generated

### Memory Element RTL (2Ports) (Explained further in Next section)

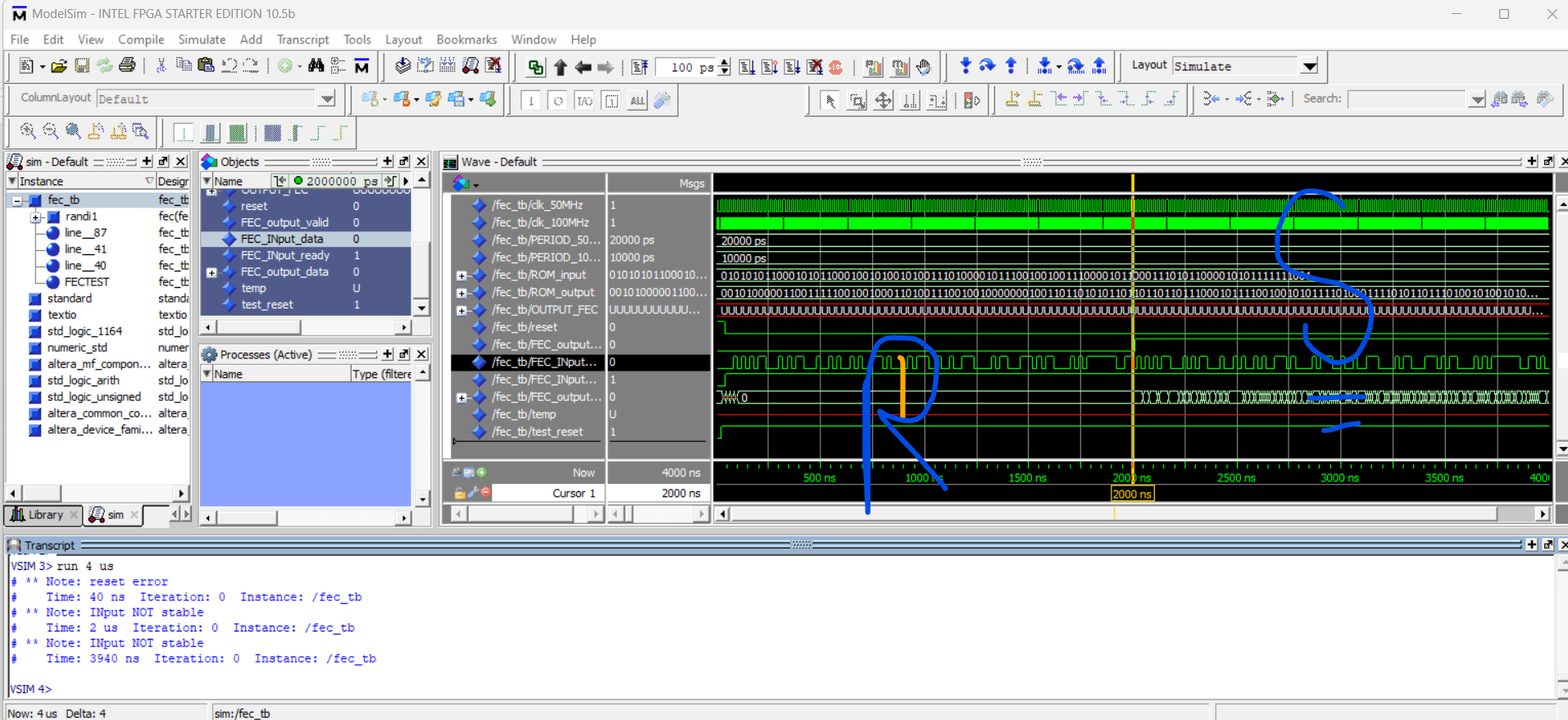


### Waveform (Sketch Expect)

A diagram of a program

Description automatically generated with medium confidence

### Results (ModelSim Altera)



## Dual Port Memory

## (We would change its name from PPL\_BLOCK to 2PORTSRAM in next phase).

|  |  |
| --- | --- |
| **Main File Name** | **PPL\_BLOCK** |

## A diagram of a dual port ram Description automatically generated

Fig.1

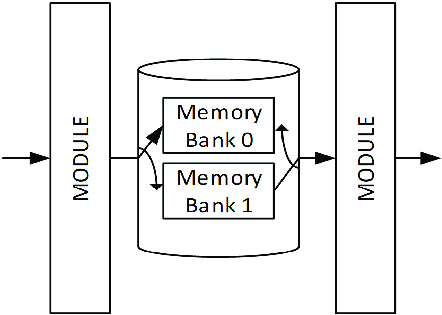


Fig 2.

|  |  |  |  |
| --- | --- | --- | --- |
| Signals | In/Out | Type | Width |
| clk\_50MHz | In | STD\_LOGIC | 1 |
| data\_in\_portA | In | STD\_LOGIC\_VECTOR | 95 |
| reset | In | STD\_LOGIC | 1 |
| data\_in\_portB | In | STD\_LOGIC\_VECTOR | 95 |
| w\_en\_portA | In | STD\_LOGIC | 1 |
| w\_en\_portB | In | STD\_LOGIC | 1 |
| address\_portA | in | STD\_LOGIC\_VECTOR | 95 |
| Address\_portB | In | Std\_logic\_vector | 95 |
| r\_en\_portA | in | Std\_logic | 1 |
| r\_en\_portB | In | Std\_logic | 1 |
| data\_out\_portA | Out | Std\_logic\_vector | 95 |
| data\_out\_portB | out | Std\_logic\_vector | 95 |

|  |  |  |  |
| --- | --- | --- | --- |
| Signals | In/Out | Type | Width |
| clk\_100MHz | In | STD\_LOGIC | 1 |
| data\_in\_portA | In | STD\_LOGIC\_VECTOR | 95 |
| reset | In | STD\_LOGIC | 1 |
| data\_in\_portB | In | STD\_LOGIC\_VECTOR | 95 |
| w\_en\_portA | In | STD\_LOGIC | 1 |
| w\_en\_portB | In | STD\_LOGIC | 1 |
| address\_portA | in | STD\_LOGIC\_VECTOR | 95 |
| Address\_portB | In | Std\_logic\_vector | 95 |
| r\_en\_portA | in | Std\_logic | 1 |
| r\_en\_portB | In | Std\_logic | 1 |
| data\_out\_portA | out | Std\_logic\_vector | 95 |
| data\_out\_portB | out | Std\_logic\_vector | 95 |

**Functionality**

The dual port ram takes two input banks named port A and port B , then reads and writes bit vectors from and to the memory (referred to by the address pin in fig.1) . The data packets are 192 bits wide and of length 1 bit .The dual-port memory implements a ping pong buffer which reads data from a bank while writing in another bank within the same clock cycle. The aim of the buffer is to allow for the smooth flow of bits in order for the upstream and downstream modules to correctly handle the input bits and output bits.

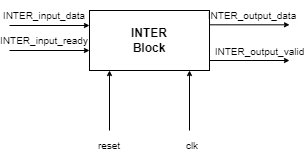
## Interleaver

|  |  |  |
| --- | --- | --- |
| **Main File Name** |  | **INTER.vhd** |
| **Testbench File Name** |  | **INTER\_tb.vhd** |
| **Extra File 1** |  | **subSIPO.vhd** |
| **Extra File 1 TESTBENCH** |  | **subSIPO\_tb.vhd** |
| **Extra File 2** |  | **subPISO.vhd** |
| **Extra File 2 TESTBENCH** |  | **subPISO\_tb.vhd** |

### Ports:

|  |  |  |  |
| --- | --- | --- | --- |
| Signals | In/Out | Type | Width |
| INTER\_INput\_data | **In** | std\_logic | 1 |
| INTER\_INput\_ready | **In** | std\_logic | 1 |
| Clk | **In** | std\_logic | 1 |
| reset | **In** | std\_logic | 1 |
| INTER\_OUTput\_data | **Out** | std\_logic | 1 |
| INTER\_OUTput\_valid | **out** | std\_logic | 1 |

### Block Diagram Sketch:

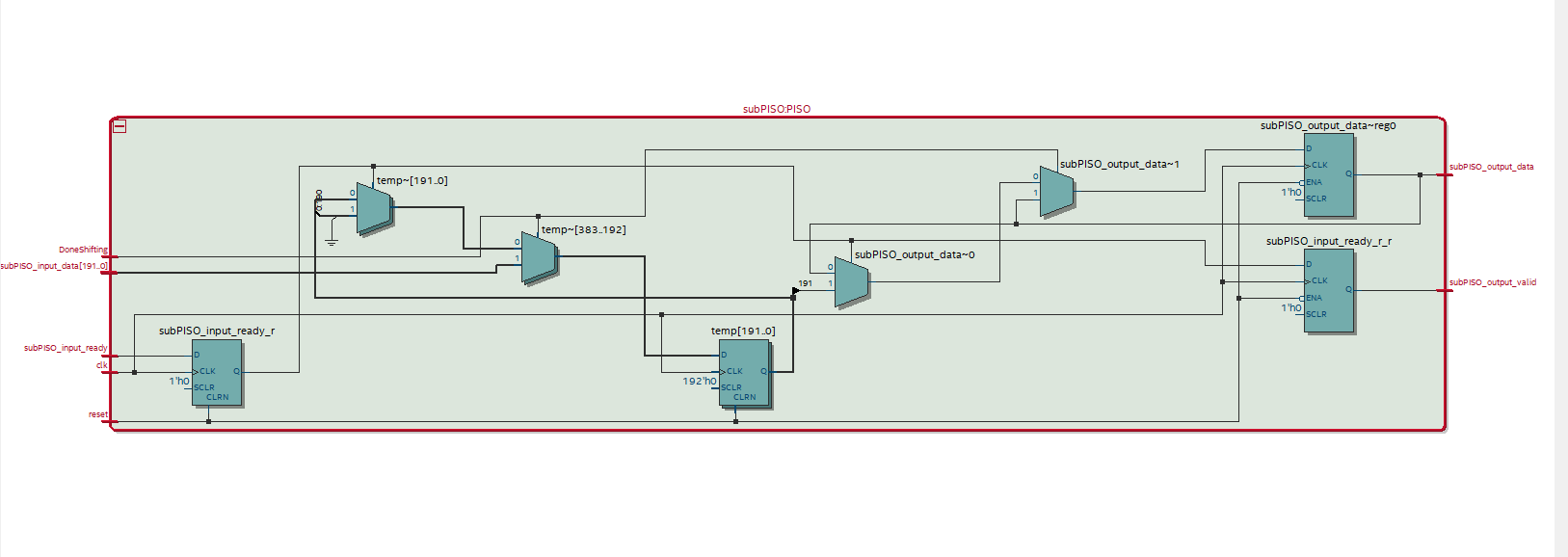


### Testing and Functionality

* We did Testing on two stages, testing PISO and SIPO blocks first, then testing the whole INTER Block.
* For SIPO, we entered a 192- bits in parallel , and checked the output serial.
* For PISO, we entered 192 bits in series, and cheked the pareleh output signal.
* For INTER, input data is Parellel, then wires permeate the input, then output it serially
* Total delay of the block is n+2 (where n is the number of cycles) … and this two is mainly from shifting and loading.

### RTL (Quartus)

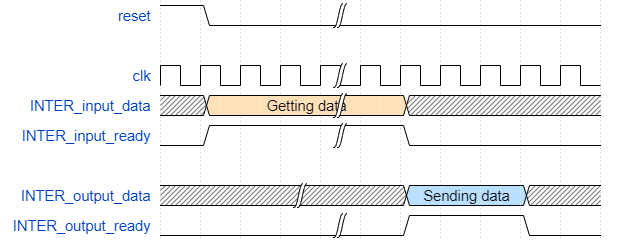
1. **RTL OF INTER (MAIN BLOCK)  
   A computer graphic of a circuit

   Description automatically generated with medium confidence**
2. **RTL OF subPISO (SUB BLOCK)   
     
   **
3. **RTL OF subSIPO (SUB BLOCK)**

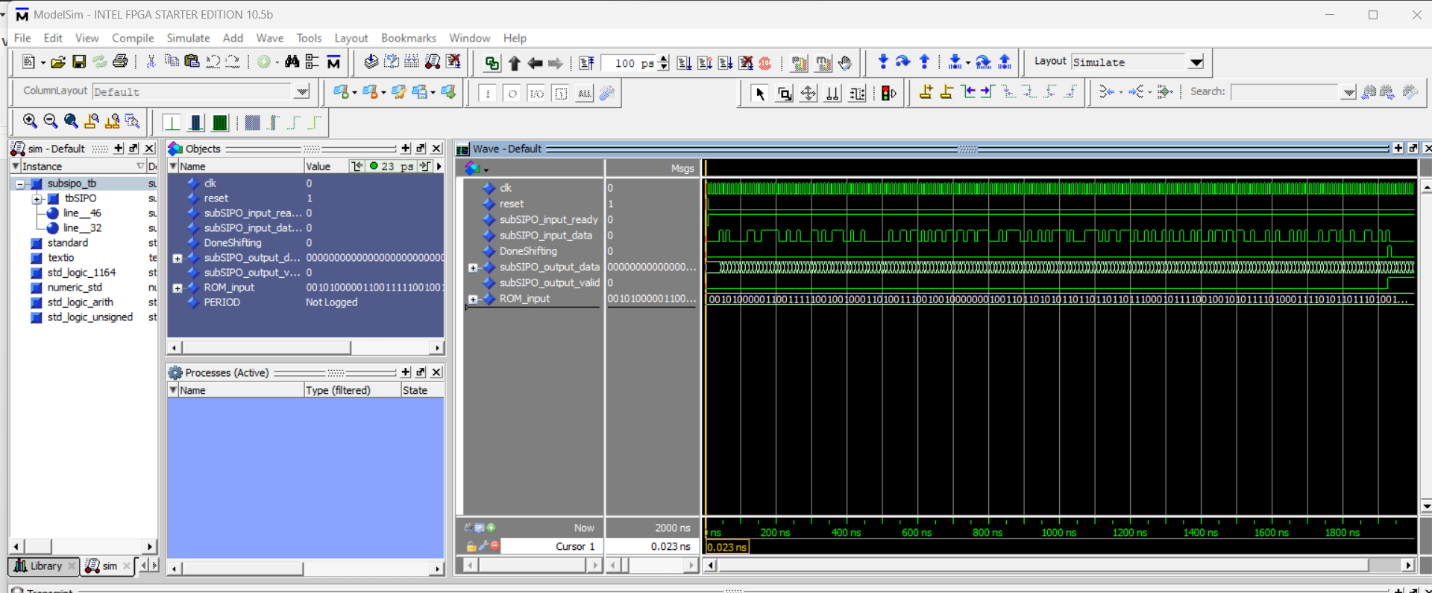
**A computer screen shot of a circuit board

Description automatically generated**

### WaveForm (Sketch Expect)



### Results (ModelSim Altera)

1. **RESULTS OF PISO (SUB BLOCK)  
   **
2. **RESULTS OF SIPO (SUB BLOCK)** A screenshot of a computer

   Description automatically generated
3. **RESULTS OF INTER (MAIN BLOCK).** A screenshot of a computer

   Description automatically generated

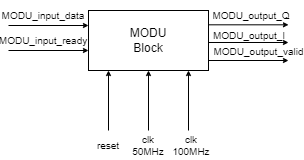
## Modulation

|  |  |
| --- | --- |
| **Main File Name** | **MODU.vhd** |
| **Testbench File Name** | **MODU\_tb.vhd** |

### Ports:

|  |  |  |  |
| --- | --- | --- | --- |
| Signals | In/Out | Type | Width |
| MODU\_input\_data | In | std\_logic | 1 |
| MODU\_input\_ready | In | std\_logic | 1 |
| clk\_100MHz | In | std\_logic | 1 |
| clk\_50MHz | In | std\_logic | 1 |
| reset | In | std\_logic | 1 |
| MODU\_output\_valid | Out | std\_logic | 1 |
| MODU\_output\_Q | Out | std\_logic\_vector | 16 |
| MODU\_output\_I | Out | std\_logic\_vector | 16 |

### Block Diagram Sketch:



### Testing and Functionality

* ⁠Again, we use 100MHz for output and 50MHz for input.  
    
  - For handshaking, we have some delay to allow data to arrive as serial inputs (let’s say late by x cycles).   
  - The total time needed is n +x ( where n might be calculated as 192\*8).

### RTL (Quartus)

A diagram of a computer program

Description automatically generated

### Waveform (Sketch Expect)

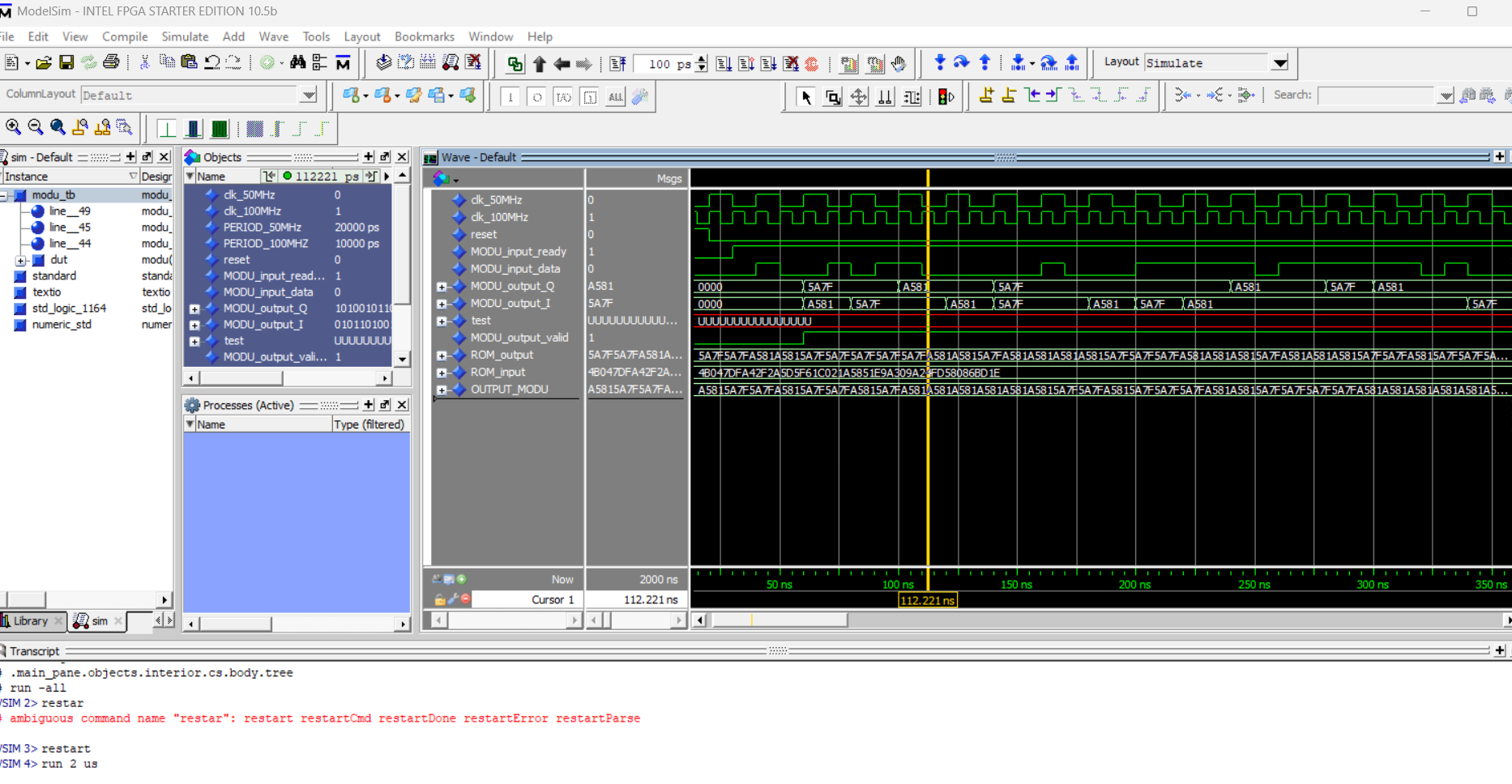
A diagram of a data processing process

Description automatically generated with medium confidence

### Results (ModelSim Altera)

A screenshot of a computer

Description automatically generated



**Extra File 1**

## PPL (Next Phase)

|  |  |
| --- | --- |
| **Main File Name** | **Different CLOCK Frequencies** |

### Ports:

|  |  |  |  |
| --- | --- | --- | --- |
| Signals | In/Out | Type | Width |
| defulatclk | In | std\_logic | 1 |
| Reset | In | std\_logic | 1 |
| Clk\_valid | OUT | std\_logic | 1 |
| Clk1 | Out | std\_logic | 1 |
| Clk2 | Out | std\_logic | 1 |
|  |  |  |  |

A diagram of a block

Description automatically generated