

1-

MATLAB Bit accurate model is in MATLAB folder, consisting of two m files:

- a. `testvectorgenerator.m`: Generates S streams of 128x8 numbers line by line in two formats binary and decimal. The generation is such that there are on average thirty-one 1s inside a stream (the distribution follows a Gaussian function) then those streams that have more than thirty-one 1s are discarded. The parameter S should be determined for each examination. After running this m file, two files `input_decimal.txt` and `input_binary.bin` are generated
- b. `outputgenerator.m`: Takes in the file `input_decimal.txt` from the previous m file output, and then generates the locations of 1s in every stream, and their hamming weight and stores them respectively into `locations.txt` and `hamming_weights.txt`.

2-

RTL files are in HDL folder that includes:

```
top.v __
    |__ weight_locator.v
    |__ fifo648.v
    |    |__ dual_port_ram.v
    |__ fifo648.v
    |    |__ dual_port_ram.v
```

And two test benches that test `top.v` and `weight_locator.v`

When simulating with `top_tb.v` please make sure to assign the path to the three files that have been generated by MATLAB in step 1. Also, set S , number of streams, according to the S you set in the MATLAB files.

3-

I ran the test bench for automated MATLAB generated vectors of size 100 streams and another of size 1024 streams, and for a manually generated test vector as a corner case, where two consecutive streams are inserted into the system, the first stream ending with 32 1s and the second starting with 32 1s. All were verified. A Modelsim project is included in MODELSIM folder that runs on the test vector of size 100 streams. Before simulating, please modify the three paths inside the test bench, at lines 102 103 and 104, so that they point to the location of the MATLAB generated files. Also, make sure parameter S is set according to the S in MATLAB.

```

Transcript
# OK
# Location generated by HDL is 931
# Location validated by MATLAB is 931
# OK
# Break in Module top_tb at C:/Users/xmhos/Desktop/ChallengeProject/HDL/top_tb.v line 134







```

4-

FPGA Artix-7 (xc7a200tsbv484-a) chosen, clock constraint set to 20ns, synthesized and implemented in Vivado with no setup/hold/slack violation. Clock period = 20 ns thereby yielding a clock frequency of 50MHz and since the inputs are 128x8 per 129 clock cycle the throughput is:
Throughput = $8 \times (128) \times 50\text{MHz} / 129 = 49.6 \text{ Mbyte/sec} = 396.9 \text{ Mbps}$

| Design Timing Summary | | |
|--|----------------------------------|---|
| Setup | Hold | Pulse Width |
| Worst Negative Slack (WNS): 14.633 ns | Worst Hold Slack (WHS): 0.147 ns | Worst Pulse Width Slack (WPWS): 9.500 ns |
| Total Negative Slack (TNS): 0.000 ns | Total Hold Slack (THS): 0.000 ns | Total Pulse Width Negative Slack (TPWS): 0.000 ns |
| Number of Failing Endpoints: 0 | Number of Failing Endpoints: 0 | Number of Failing Endpoints: 0 |
| Total Number of Endpoints: 192 | Total Number of Endpoints: 192 | Total Number of Endpoints: 103 |
| All user specified timing constraints are met. | | |

With the following device utilization:

| Hierarchy | | | | | | | | | |
|---|----------------|-----|------------------------|-----------------------------|---------------------|---------------------|---------------|---------------------|------------------|
| | Name | ^ 1 | Slice LUTs (134600) | Slice Registers (269200) | F7 Muxes (67300) | F8 Muxes (33650) | DSPs (740) | Bonded IOB (285) | BUFGCTRL (32) |
|  | | | | | | | | | |
|  | | | | | | | | | |
|  | | | | | | | | | |
|  | | | | | | | | | |
|  | | | | | | | | | |
|  | | | | | | | | | |
| | top | | 593 | 1057 | 38 | 10 | 8 | 31 | 1 |
| | U1 (fifo648) | | 323 | 539 | 16 | 4 | 0 | 0 | 0 |
| | U2 (fifo648_0) | | 219 | 467 | 22 | 6 | 0 | 0 | 0 |

5-

In summary the system has two phases: **read-in** and **send-out**

Read-in: in this phase the stream of 128 bytes (=1024 bits) are read one byte at every clock. Meanwhile, the non-zero bytes as well as their coming time, clk_cycle_counter, are saved into two FIFOs at the next clock cycle upon their arrival. Since one of the constraint is that the hamming weight of the streaming input is less than 32, therefore, in a scenario in which 31 bytes of the whole coming byte has a hamming weight of 1, 31 locations will be filled in one episode (Where an Episode is from the begin to the end of when all 1024 bits have all been inserted.) However, The FIFO should be double the size 31*5 so that in case that instantly after the first Episode, in the 2nd Episode, four consecutive all-one bytes are coming in, they find a place in the FIFO to be stored. **The minimum requirement for the first FIFO size is 62x5 and for the second FIFO 62x6. However, we chose two FIFOs of sizes 64x8, because they abide to standard IPs, also to loosen up the range of hamming weight values.**

Send-out: in this phase, the stored values are read upon the assertion of a start signal. Every byte is evaluated in terms of its hamming weight and, by means of a controlled rd_enable signal, is given a time span according to its hamming weigh. In the interim, a decoder (weight_locator) indicates the location of all possible 1s, if existing, in that read byte. A set of FSM-like signals pass these locations out by taking into account the cycle at which they were stored (fetched from the second FIFO) by the equation:

$$\text{Location_of_1_in_the_stream} = 8 \times \text{Stored_Clock_Cycle} + \text{Location_of_1_in_Byte}$$

For example, suppose 11010010 have been saved at Clock_Cycle = 10 in Read-in phase, and now at Send-out phase its value is read. Its hamming weight, 4, indicates that this byte requires 4 clock cycles to be thoroughly processed. In the interim, the weight_locator decoder indicates that this file has L0 = 1, L1 = 4, L2 = 6 and L3 = 7 (and the rest of the locations are “don’t care”). The state machine will finally send out 81, 84, 86 and 87, based on equation above. The next data from the FIFO will be read instantly and processed similarly. Note that upon the start of the Send-out, the system can instantly carry out another read-in phase at 129th clock cycle after assertion of a “start” signal without any interruption.

This system was designed incrementally, in the following steps:

- a. Writing and testing a small FIFO
- b. Writing a purely combination weight locator to export the location of 1s, if any, in a data byte, on 8 3-bit buses. Meanwhile it instantly exports the hamming weight, A.K.A the population count of the input on the PC output. An Example to illustrate how the weight_locator works is illustrated in the following picture. Note that in this example PC=5 distinguishes L0=0 as a 1-indicator from L5=0 as a non-1-indicator

| | | |
|-----|-------|--------|
| | ----- | |
| | | PC --5 |
| 1-- | | L0 --0 |
| 1-- | | L1 --1 |
| 0-- | | L2 --4 |
| 0-- | | L3 --6 |
| 1-- | | L4 --7 |
| 0-- | | L5 --0 |
| 1-- | | L6 --0 |
| 1-- | | L7 --0 |
| | | --0 |

- c. Combination of FIFO and weight locator was tested such that to certify if the read enable signal of the FIFO can be controlled by the decoding its data via weight locator module.
- d. A top module that integrated two FIFOs and the weight locator was written, and progressively upgraded with simple and manually generated test vectors.
- e. At a point where I realized the system is complete I tested it with the corner case issue, stated in this document. And I got the following error from my verification suit:

```

# Location generated by HDL is 6
# Location validated by MATLAB is 6
# OK
# Location generated by HDL is 8
# Location validated by MATLAB is 7
# ERROR. Display error specifications. i.e. the system state & at which clock cycle
# Location generated by HDL is 9
# Location validated by MATLAB is 8
# ERROR. Display error specifications. i.e. the system state & at which clock cycle
# Location generated by HDL is 10
# Location validated by MATLAB is 9
# ERROR. Display error specifications. i.e. the system state & at which clock cycle
# Location generated by HDL is 11

```

- f. Parsing into the timing signals and the code I realized a subtle issue in timing was neglected and fixed it by considering that state inside the state machine:

```

136 always @ (posedge clk)
137 begin
138     if (srst==1)
139         PC_out_temp <= 0;
140     else if (rd_en_pulse == 1)
141         PC_out_temp <= PC_out_temp; // just wait a cycle
142     else if ((PC_out_temp==0) && (PC_out!=0))
143         PC_out_temp <= PC_out - 1;
144     else if (PC_out_temp==0)
145         PC_out_temp <= 0;
146     else
147         PC_out_temp <= PC_out_temp - 1;
148 end

```

- g. And this corner case was resolved

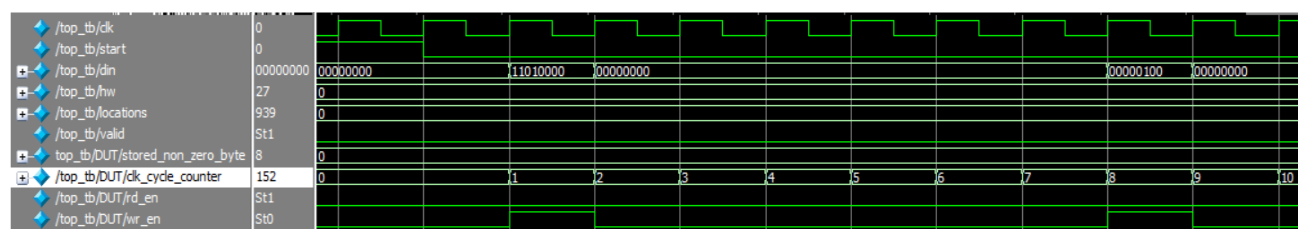
```

Transcript
# Location validated by MATLAB is 28
# OK
# Location generated by HDL is 29
# Location validated by MATLAB is 29
# OK
# Location generated by HDL is 30
# Location validated by MATLAB is 30
# OK
# Location generated by HDL is 31
# Location validated by MATLAB is 31
# OK
# Break in Module top_tb at C:/Users/xmhos/Desktop/Velodyne/HDL/top_tb.v line 87

```

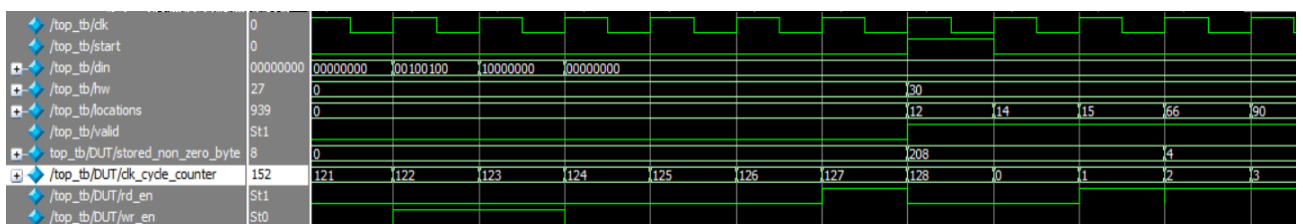
- h. And then so did every other automated test vectors I tried. Two samples of sizes 100 and 1024 streams are inside the folders and can be used for verification.
- i. A snapshot of the first two out of 100 streams of a sample test vector are attached below:

Start of arrival of stream 1:



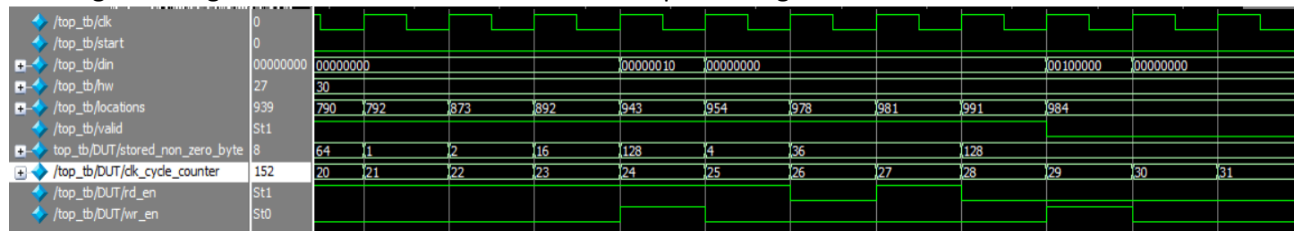
...

End of stream 1 and start generating locations for stream 1. Instantly, arrival of stream 2



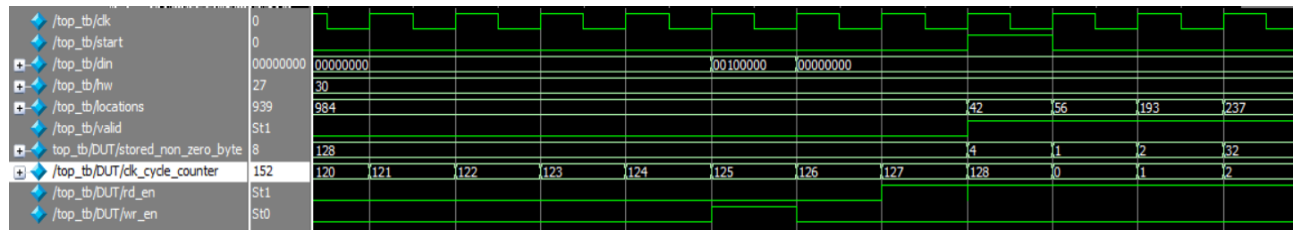
...

End of generating locations for stream 1. Stream 2 still processing



...

End of stream 2 and start of generating locations for stream 2. Instantly, start of arrival of stream 3



... and alike to the last stream.

Design summary is reflected in section 4.

6-

Time assigned to this project:

I started to get my hand on this project from early Saturday 2/17/2018 until late Monday 2/19/2018, and devoted an average time of 30 hours on the following basis:

| Project Phase | Time assigned (hour) |
|--|----------------------|
| testing a small FIFO | 1 |
| Writing a purely combination weight locator | 7 |
| Combination of FIFO and weight | 3 |
| A simple top module that integrated simple modules and tested with a few vectors | 10 |
| troubleshooting | 4 |
| MATLAB test vector and output validation generation | 5 |
| This document | 5 |
| Total | 35 |