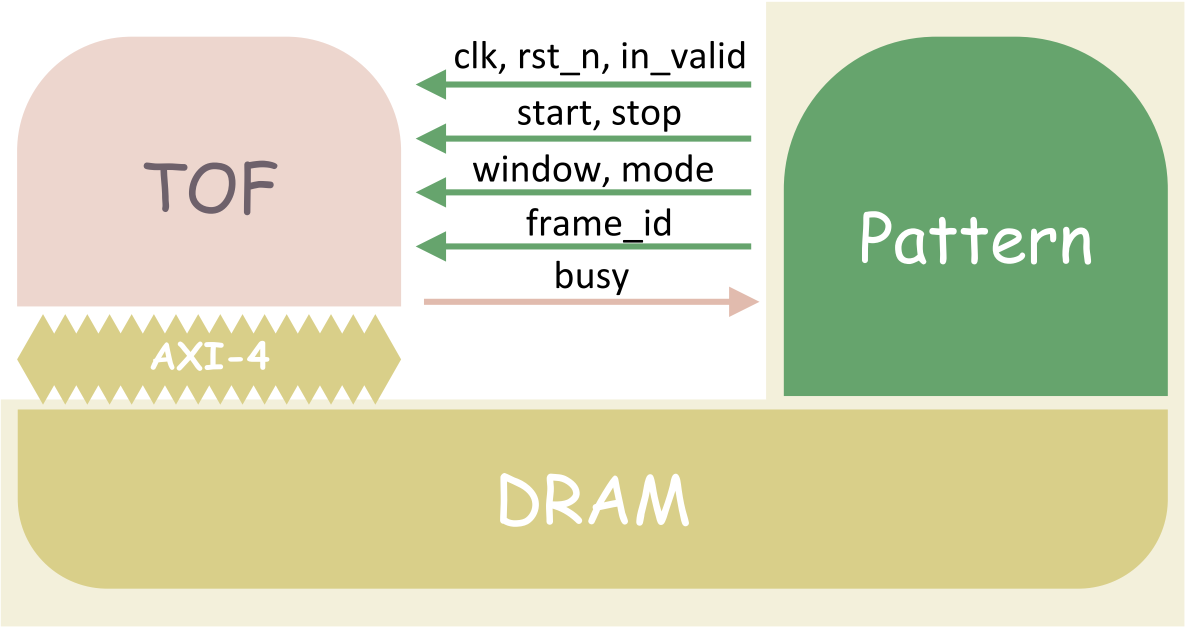
2022 Spring ICLAB Midterm Project

TA: Wen-Yue, Lin

Advisor: Prof. Chen-Yi, Lee

Overall System Block

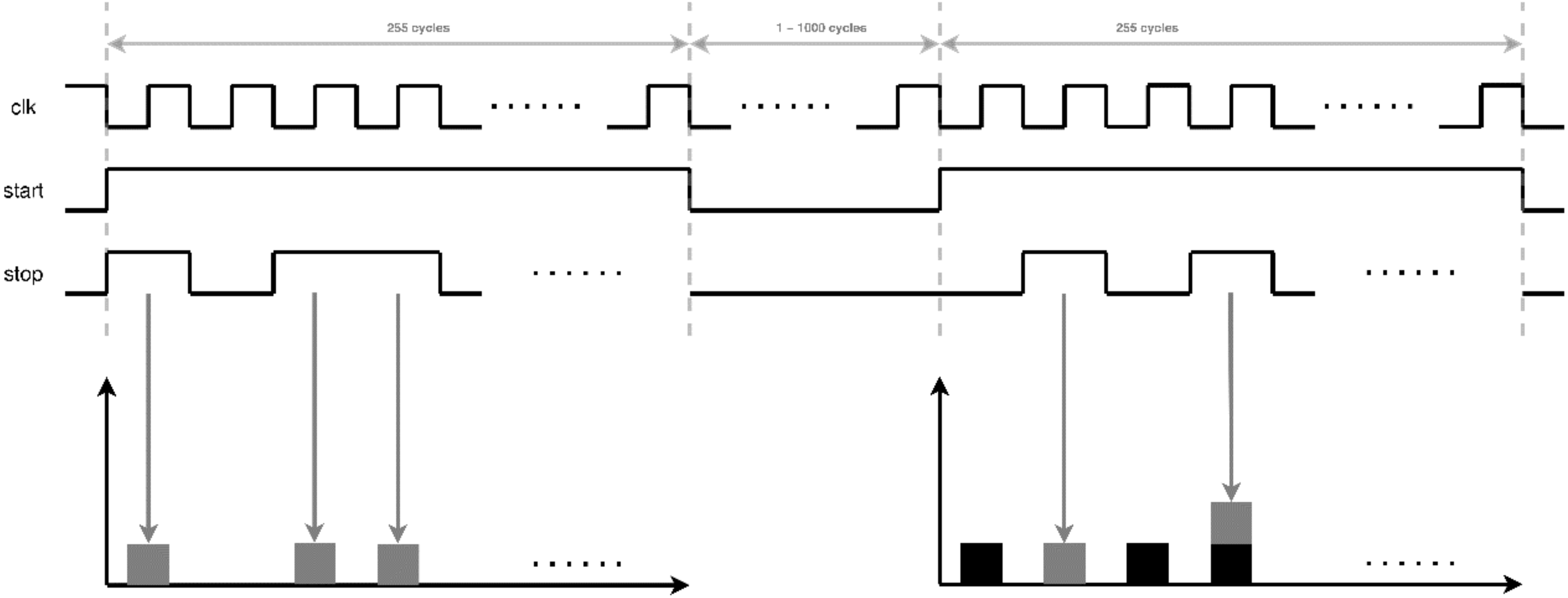


Input & Output Signals

Input signals	Width	Functional Description
rst_n	1	Asynchronous reset and active-low
clk	1	Clock for TOF chip
in_valid	1	When in_valid is high, all the other three input is valid
start	1	Start counting cycle
stop	16	Calculate the time between start and stop and store the time in histogram, 16 stops are independent
window	2	2’b00: window = 1, find the highest bin in histogram 2’b01: window = 2, find 2 adjacent bins that have maximum sum 2’b10: window = 4, find 4 adjacent bins that have maximum sum 2’b11: window = 8, find 8 adjacent bins that have maximum sum
mode	1	1’b0: generate histogram from input 1’b1: read histogram from DRAM
frame_id	5	Index of frame in DRAM, range in No. 0 ~ No. 31

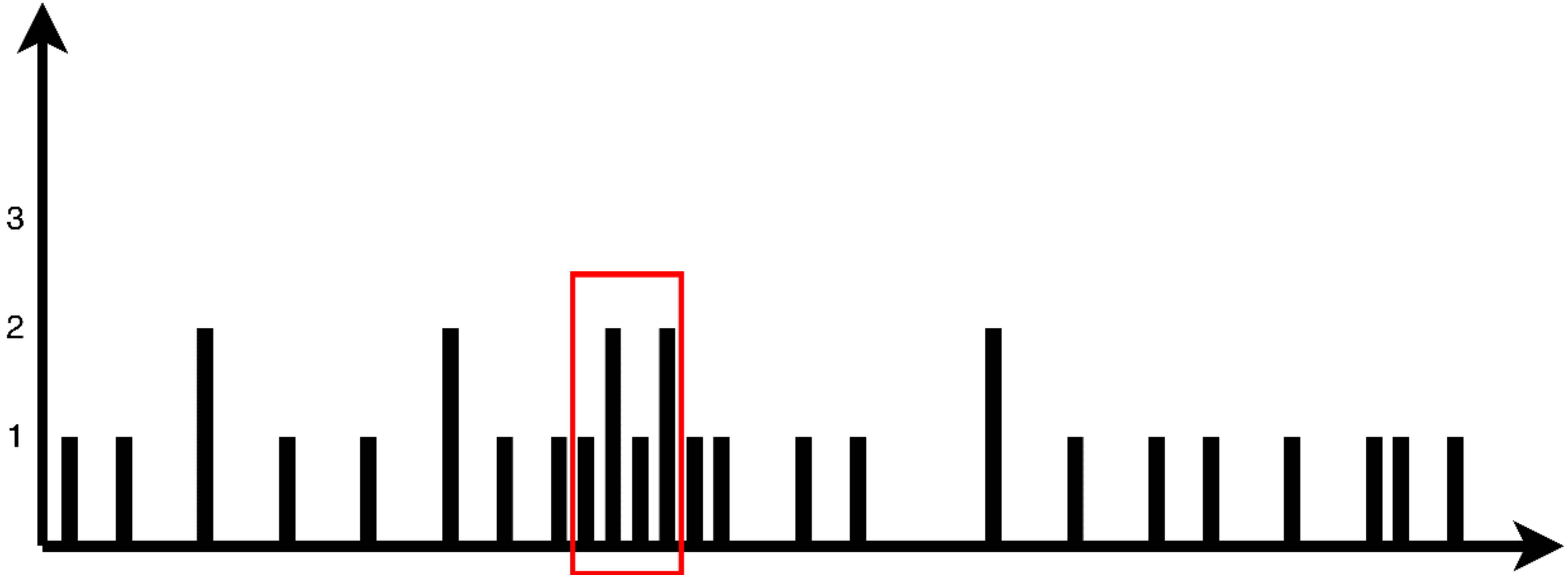
Output signal	Width	Functional Description
busy	1	Pulls low when design finish storing right data to DRAM

Histogram Generate



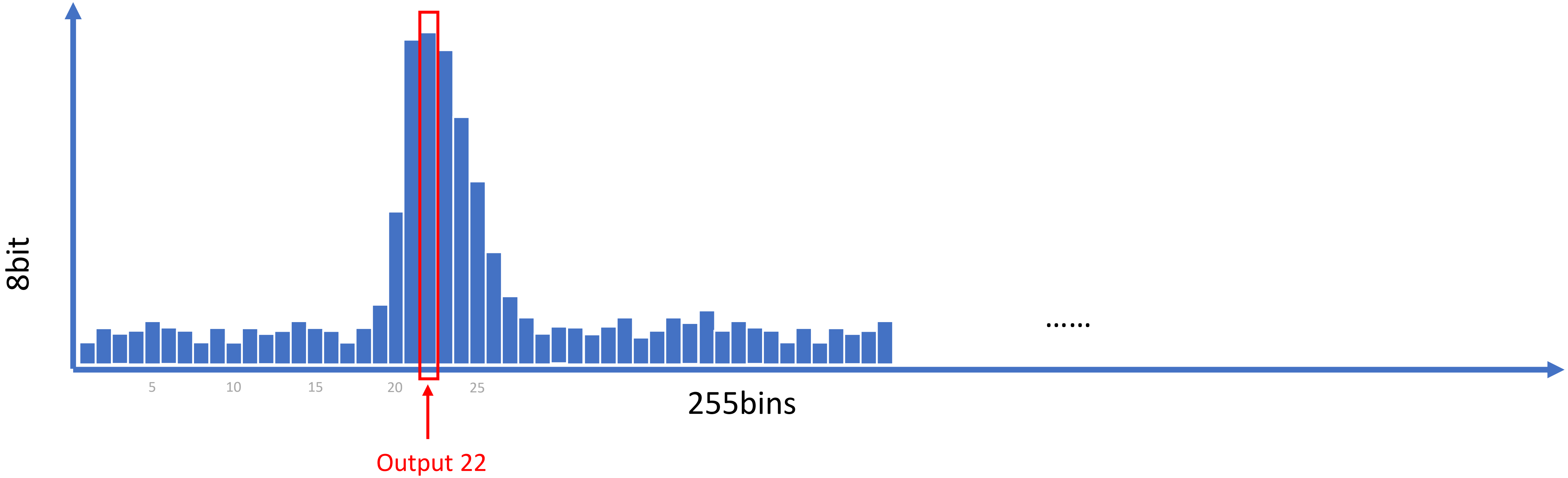
Window Method

- For low count
- Ex. window = 4



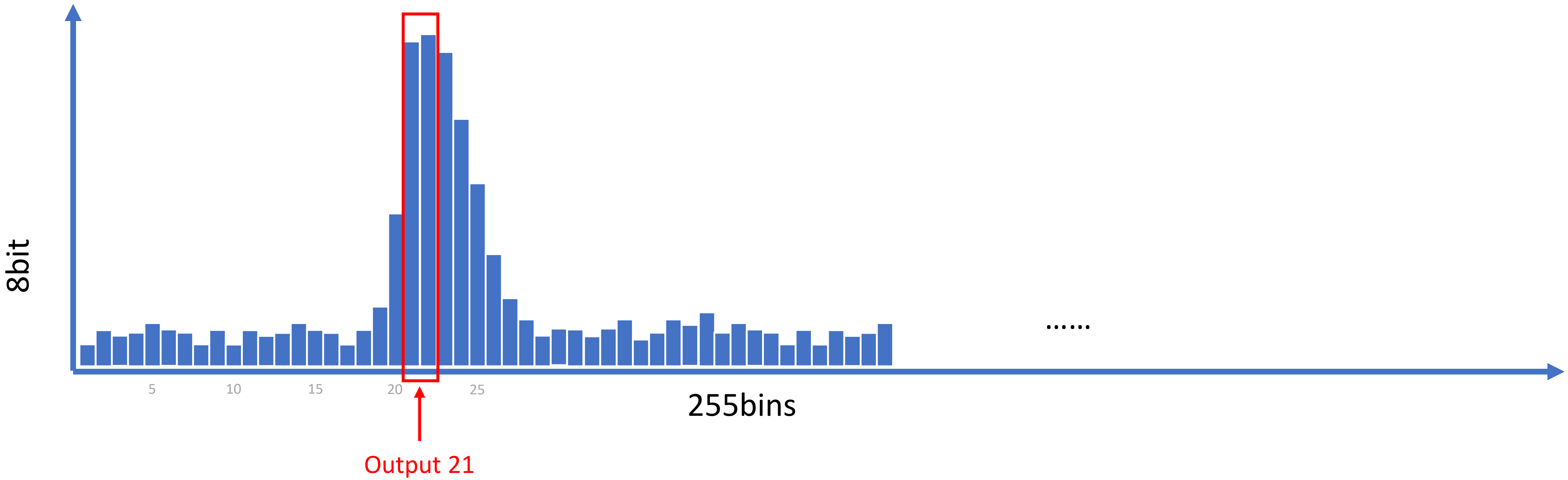
Window Method

- Window = 1: find the highest bin
- If two bins have the same height → output first bin



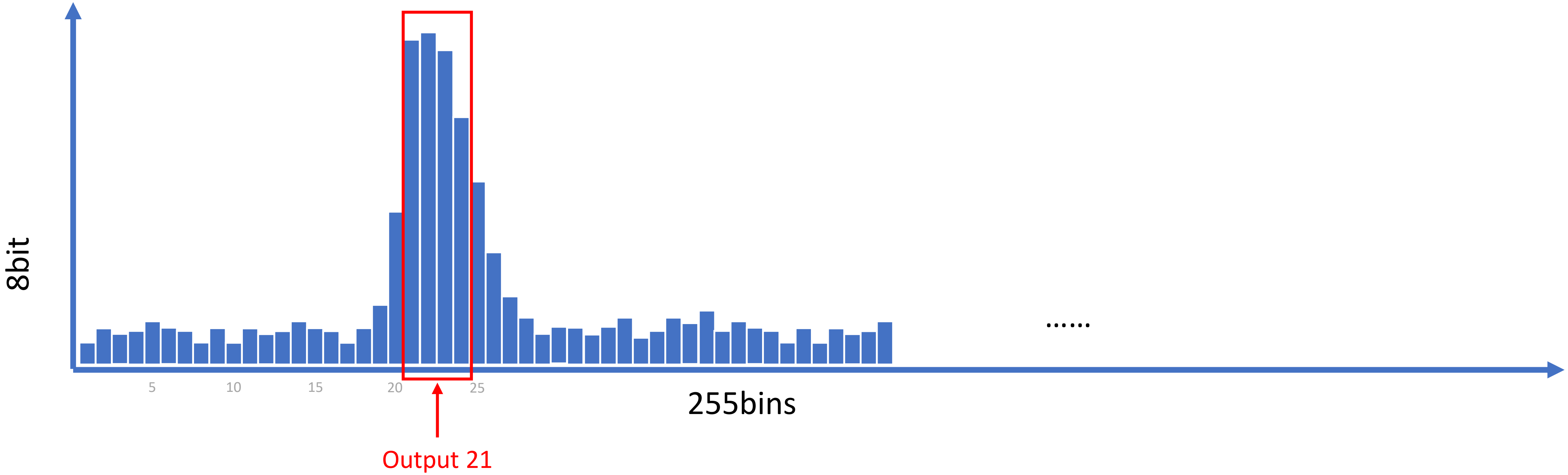
Window Method

- Window = 2: find two adjacent bins that have maximum sum
- If two bins have the same height → output first bin
- Output the first location of the bin group (21 in this case)



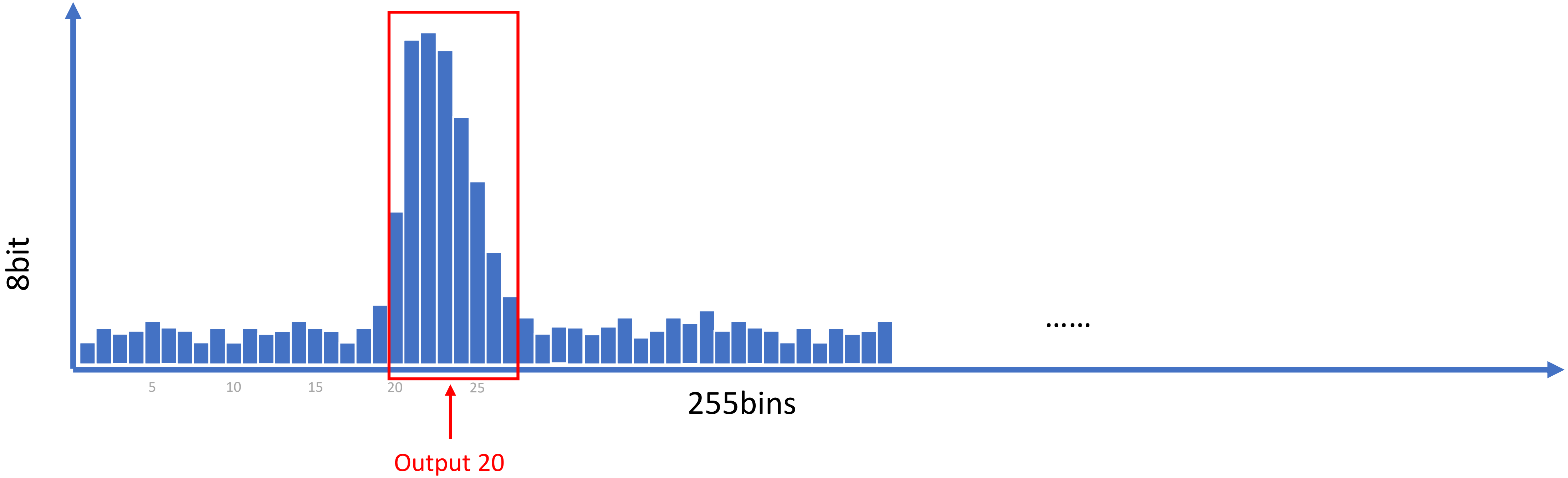
Window Method

- Window = 4: find 4 adjacent bins that have maximum sum
- If two bins have the same height → output first bin
- Output the first location of the bin group (21 in this case)

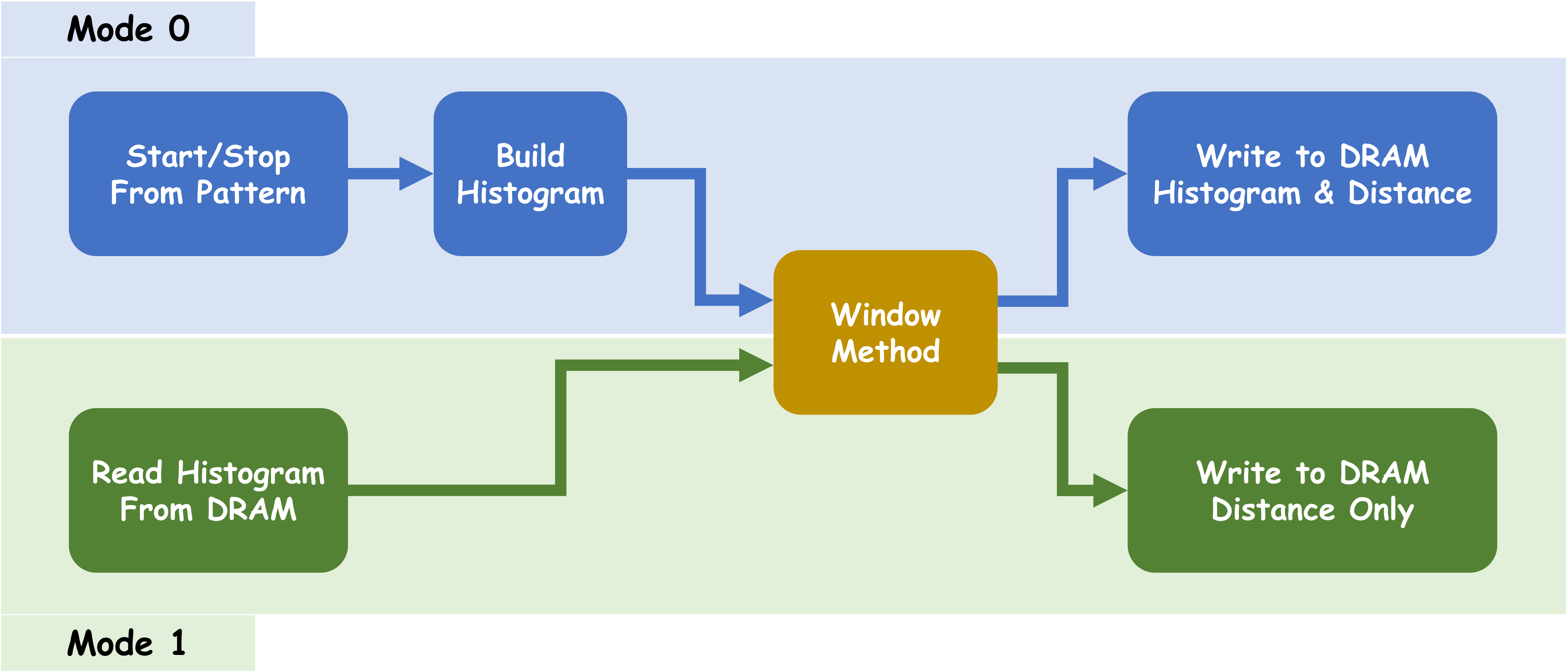


Window Method

- Window = 8: find 8 adjacent bins that have maximum sum
- If two bins have the same height → output first bin
- Output the first location of the bin group (20 in this case)



System Flow



Mode 0
Start/Stop From Pattern
Build Histogram
Window Method
Write to DRAM Histogram & Distance
Read Histogram From DRAM
Write to DRAM Distance Only
Mode 1

DRAM Provided by TA

Address Mapping

DRAM

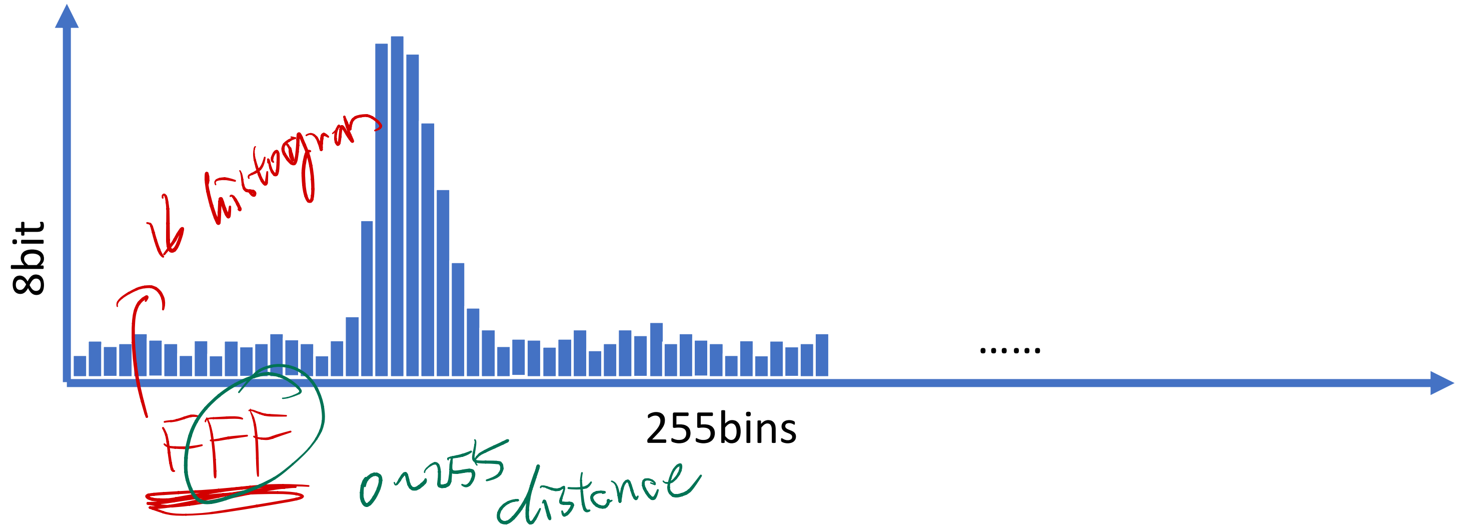
From : 0x00000000
To : 0x0000FFFF
Kernel Not Accessible

From : 0x00010000
To : 0x0002FFFF
Frame : NO.0 - NO.31

0000
~
FFFF

共32個

10
~
2F
fram



0x00010000-0x000100FE: Histogram
0x000100FF: Distant for that pixel
0x00010?XX: 16 Histograms form 1 frame
0x0001?XXX: 16 Frames
0x0002?XXX: another 16 Frames

DRAM Provided by TA

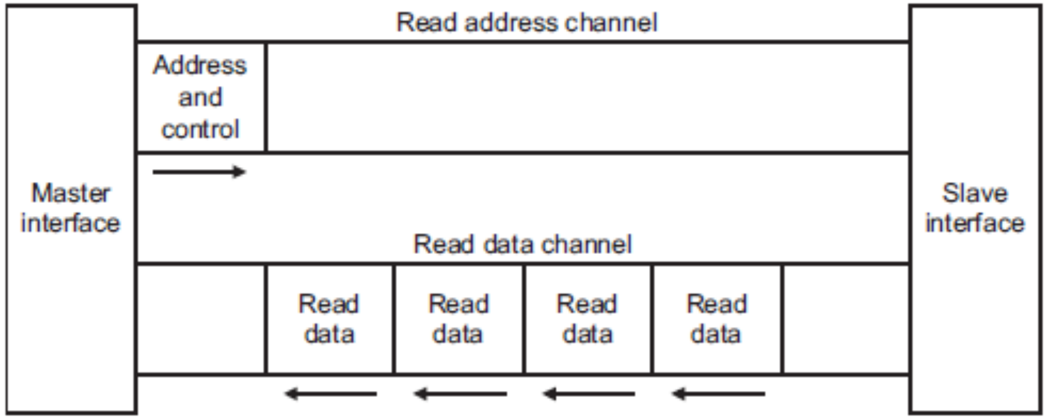
Fetch DRAM in your Design

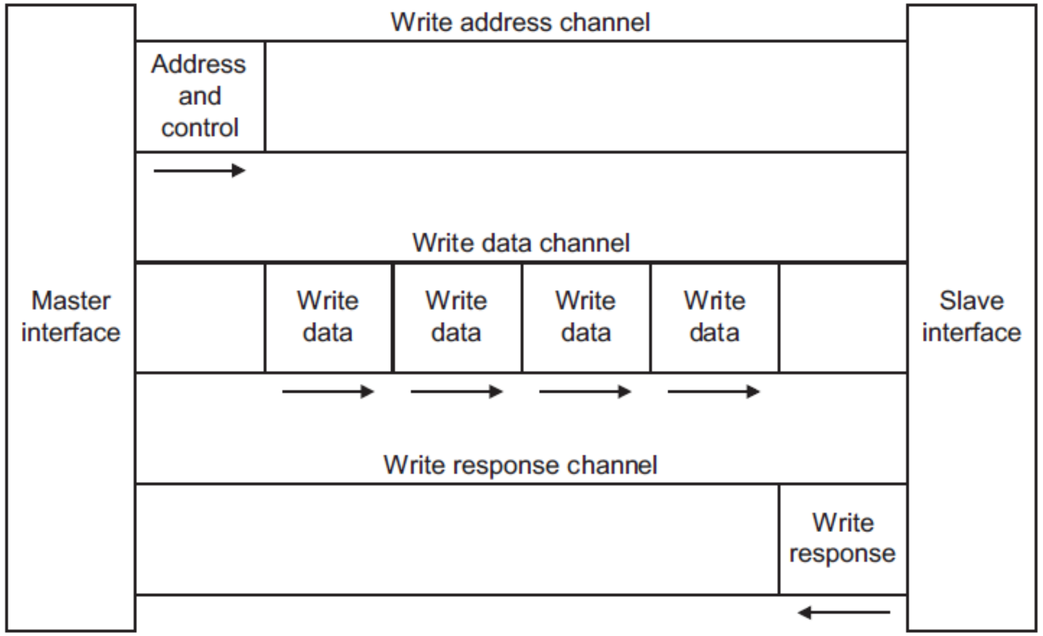
DRAM

From : 0x00000000
To : 0x0000FFFF
Kernel Not Accessible

From : 0x00010000
To : 0x0002FFFF
Frame : NO.0 - NO.31

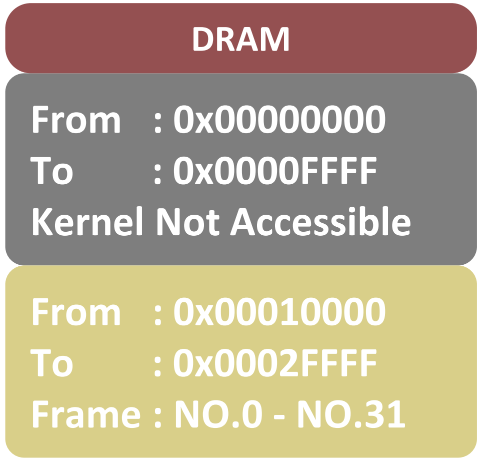
- AXI4 Protocol: Refer to Midterm Project Pre-released (AXI4)





DRAM Provided by TA

PATTERN Check Method



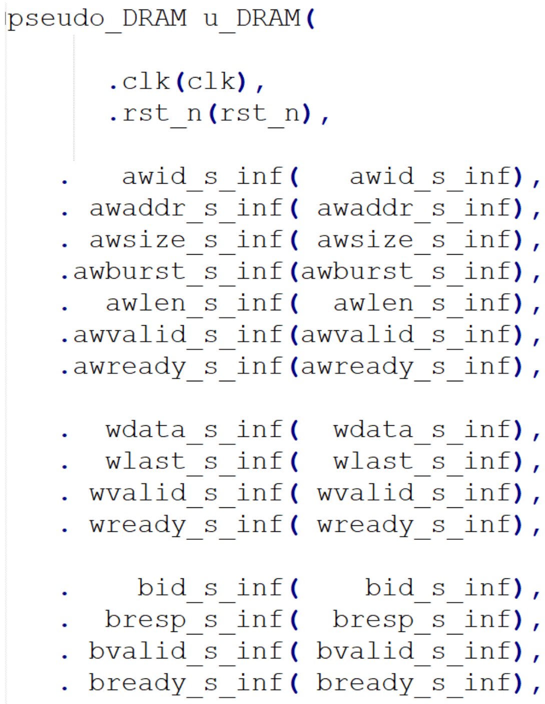
DRAM

From : 0x00000000
To : 0x0000FFFF
Kernel Not Accessible

From : 0x00010000
To : 0x0002FFFF
Frame : NO.0 - NO.31

- How to directly access DRAM in pattern.

Declare DRAM in PATTERN



```
pseudo_DRAM u_DRAM(

        .clk(clk),
        .rst_n(rst_n),

    .   awid_s_inf(   awid_s_inf),
    . awaddr_s_inf( awaddr_s_inf),
    . awsize_s_inf( awsize_s_inf),
    .awburst_s_inf(awburst_s_inf),
    .  awlen_s_inf(  awlen_s_inf),
    .awvalid_s_inf(awvalid_s_inf),
    .awready_s_inf(awready_s_inf),

    .  wdata_s_inf(  wdata_s_inf),
    .  wlast_s_inf(  wlast_s_inf),
    . wvalid_s_inf( wvalid_s_inf),
    . wready_s_inf( wready_s_inf),

    .    bid_s_inf(    bid_s_inf),
    .  bresp_s_inf(  bresp_s_inf),
    . bvalid_s_inf( bvalid_s_inf),
    . bready_s_inf( bready_s_inf),
```

Note: You should declare DRAM in pattern not design, and if your design contains DRAM unit, you will fail demo, i.e. you can only access DRAM data by AXI4 protocol in you design.

Variable in pseudo_DRAM.v (do not modify)

reg [7:0] DRAM_r [0:196607]; (Address from 00000000 to 0002FFFF)

Access submodule element (Pattern may use it to check data store in DRAM is correct or not)

```
u_DRAM.DRAM_r[temp_addr]
```

DRAM Provided by TA

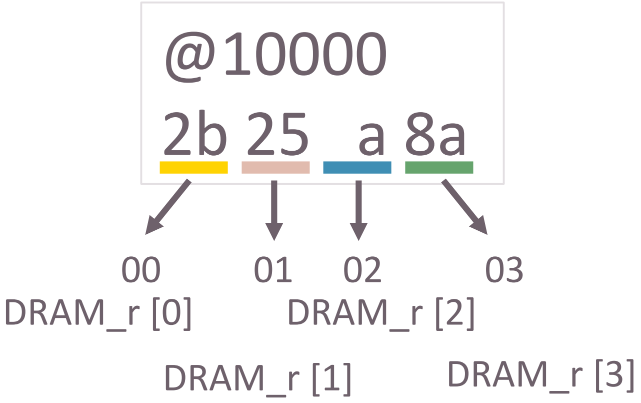
.dat file example

../00_TESTBED/DRAM/DRAM.dat

```
@10000
2b 25 a 8a
@10004
2b a8 29 34
@10008
bf 8a 5 79
@1000c
6d c5 29 a
@10010
73 a7 a 94
@10014
85 62 42 a4
@10018
ea dd 80 8b
@1001c
26 45 4a 1c
```

Variable in pseudo_DRAM.v

reg [7:0]   DRAM_r   [0:196607]; (Address from 00000000 to 0002FFFF)



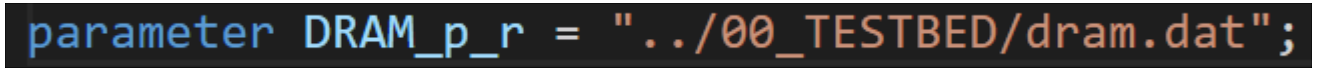
DRAM_r		
Address	[7:4]	[3:0]
[0]	2	B
[1]	2	5
[2]	0	A
[3]	8	A
[4]	2	B
[5]	A	8
[6]	2	9
[7]	3	4

DRAM Provided by TA

NOTE: YOU MAY USE

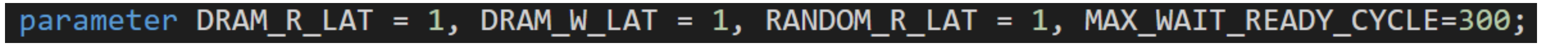
- You may modify the following part in ../00_TESTBED/pseudo_DRAM.v.

.dat file path →



```
parameter DRAM_p_r = "../00_TESTBED/dram.dat";
```

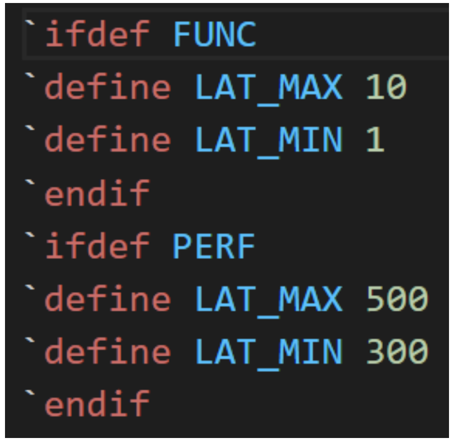
DRAM latency →



```
parameter DRAM_R_LAT = 1, DRAM_W_LAT = 1, RANDOM_R_LAT = 1, MAX_WAIT_READY_CYCLE=300;
```

- If you want to refresh dram, you may use the following code.

```
$readmemh("../00_TESTBED/dram.dat", u_DRAM.DRAM_r);
```



```
`ifdef FUNC
`define LAT_MAX 10
`define LAT_MIN 1
`endif
`ifdef PERF
`define LAT_MAX 500
`define LAT_MIN 300
`endif
```

Grading in Midterm

$$\boldsymbol{Score} = +\boldsymbol{Functionality}(\mathbf{60\%}) + \boldsymbol{Performacne}(\mathbf{40\%})$$

$$\boldsymbol{Performacne}(\mathbf{40\%}) = \boldsymbol{Rank}(\boldsymbol{Area} \ * \ \boldsymbol{Latency} \ * \ \boldsymbol{clock\,period})$$

03/23: Release Midterm Project
04/26: 1st Demo
05/03: 2nd Demo

Q&A