Computer-Aided VLSI System Design
Homework 4: IoT Data Filtering

TA：朱怡蓁 r10943012@ntu.edu.tw **Due Tuesday, Nov. 22, 14:00**

TA：羅宇呈 f08943129@ntu.edu.tw

Data Preparation

1. Decompress 1111_hw4.tar with following command

```
tar -xvf 1111_hw4.tar
```

File/Folder	Description
IOTDF.v	Your design.
testfixture.v	Testbench for IOTDF.
pattern1.dat	Input IoT data for f1~f7
f1.dat ~ f7.dat	Output golden for function 1 ~ function 7
.synopsys_dc.setup	Configuration file for DC.
.synopsys_pt.setup	Configuration file for PrimeTime
IOTDF_DC.sdc	Constraint file for synthesis.
pt_script.tcl	PrimeTime power measurement script
runall_rtl	RTL simulation bash file
run_syn	Gate-level simulation bash file

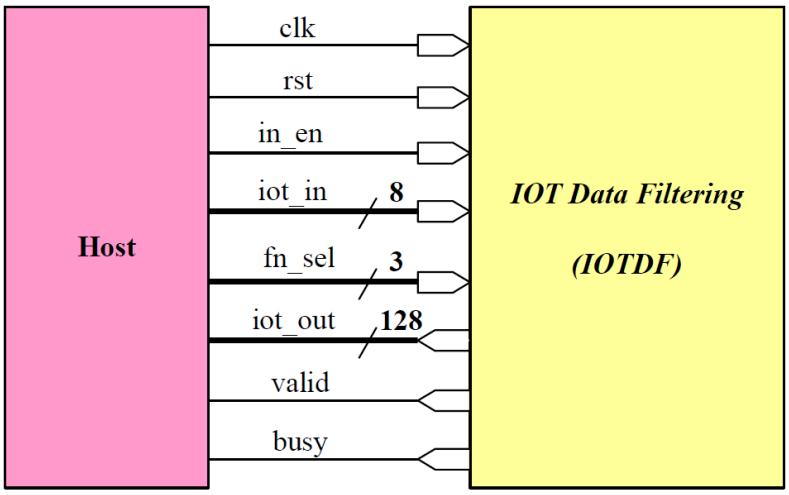
All libraries needed for synthesis, simulation can be found in previous homework.

Only worst-case library is used in this homework.

Introduction

In this homework, you are asked to design a **IoT Data Filtering (IOTDF)**, which can processor large IoT data from the sensors, and output the result in real-time.

Block Diagram

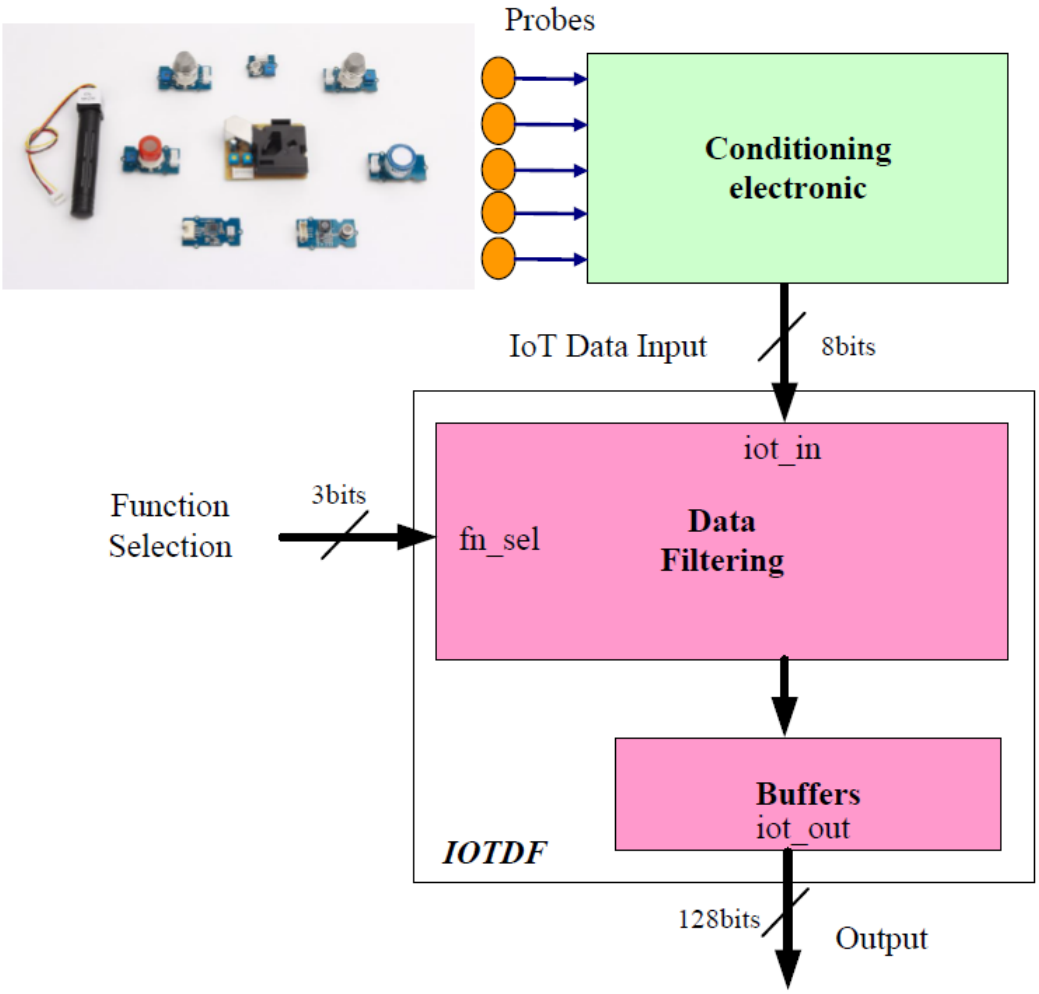


Specifications

1. Top module name: **IOTDF**
2. Input/output description:

Signal Name	I/O	Width	Simple Description
clk	I	1	Clock signal in the system (positive edge trigger). All inputs are synchronized with the **positive** edge clock. All outputs should be synchronized at clock **rising** edge
rst	I	1	Active **high** asynchronous reset.
in_en	I	1	Input enable signal. When **busy** is **low**, **in_en** is turned to **high** for fetching new data. Otherwise, **in_en** is turned to **low** if **busy** is **high**. If all data are received, **in_en** is turned to **low** to the end of the process.
iot_in	I	8	IoT input signal. Need **16** cycles to transfer one 128-bit data. The number of data is **96** in this homework.
fn_sel	I	3	Function Select Signal. There are 7 functions supported in IOTDF. For each simulation, only one function is selected for data processing.

iot_out	O	128	IoT output signal. One cycle for one data output.
busy	O	1	IOTDF busy signal (explained in description for in_en)
valid	O	1	IOTDF output valid signal Set **high** for valid output



Design Description

1. The sensor data is a 128-bit unsigned data, which is divided in 16 8-bit partial data for IOTDF fetching. The way for data transferring is as follow. Only 96 data are required to fetch for each function simulation.

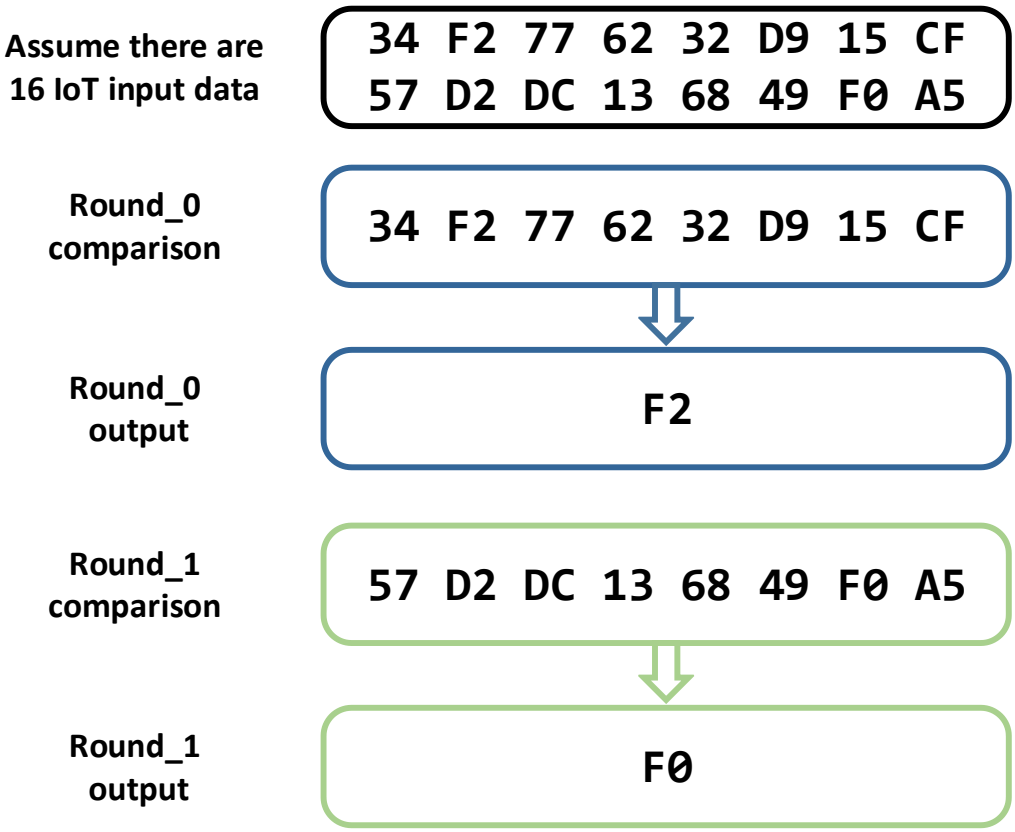
2. Seven functions are asked to design in this homework.

	Fn_sel	Functions
F1	3'b001	Max(N)
F2	3'b010	Min(N)
F3	3'b011	Avg(N)
F4	3'b100	Extract(low < data < high)
F5	3'b101	Exclude(data<low , high<data)
F6	3'b110	PeakMax(the data is larger than previous output data)
F7	3'b111	PeakMin(the data is smaller than previous output data)

The details of funstions are shown in the following: (8*bit for example)

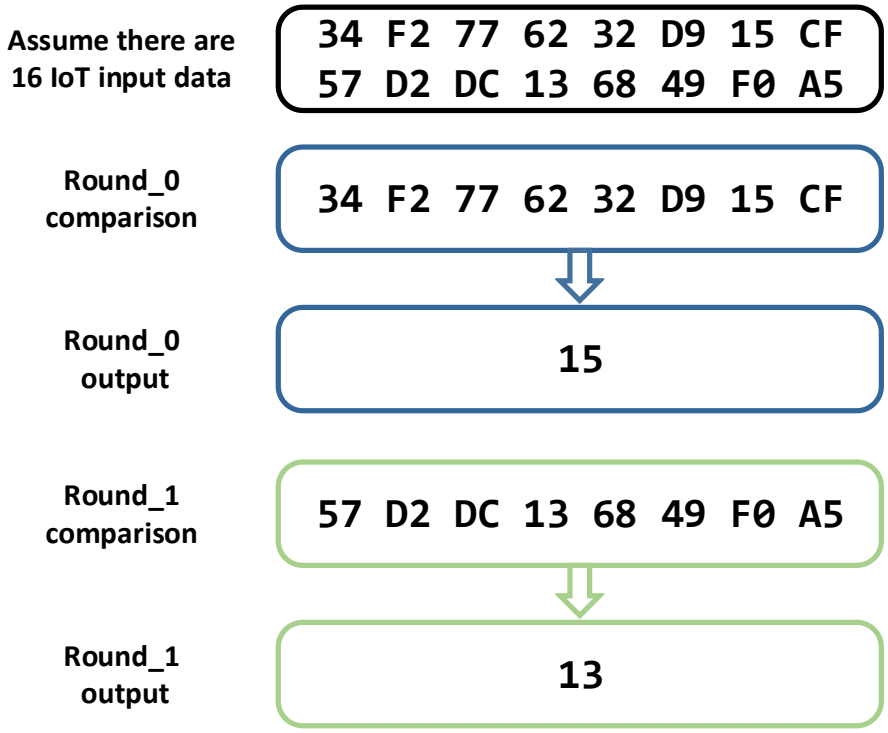
a. F1: Max(N)

- Find the largest data in 8 IoT data for each round.



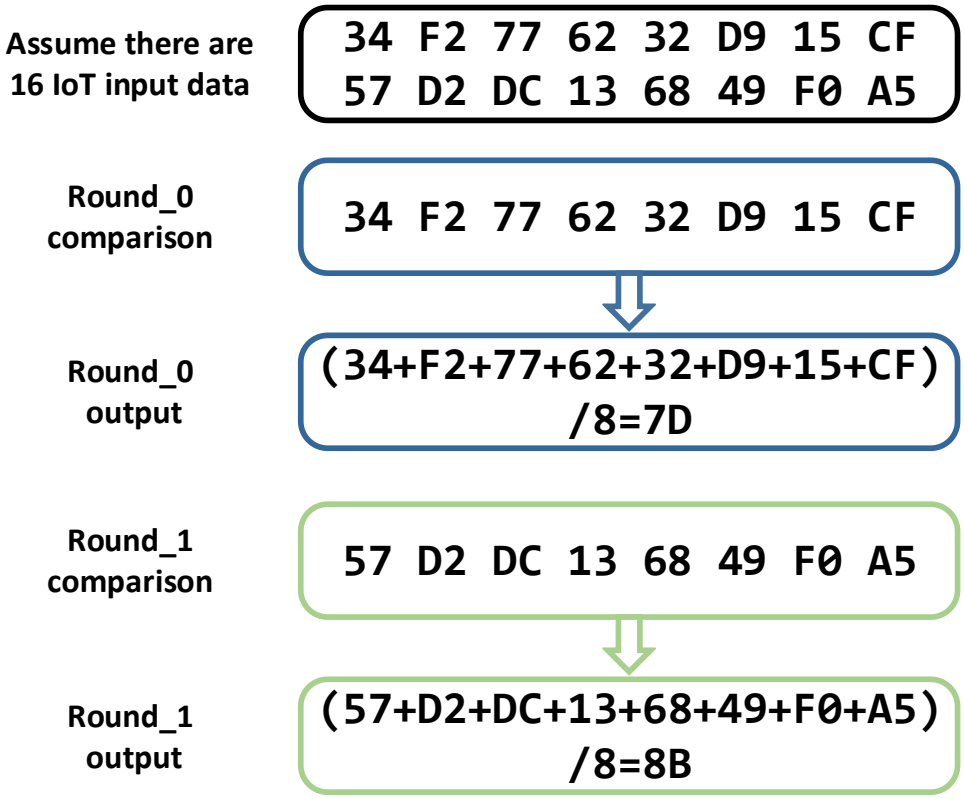
b. F1: Min(N)

- Find the smallest data in 8 IoT data for each round.



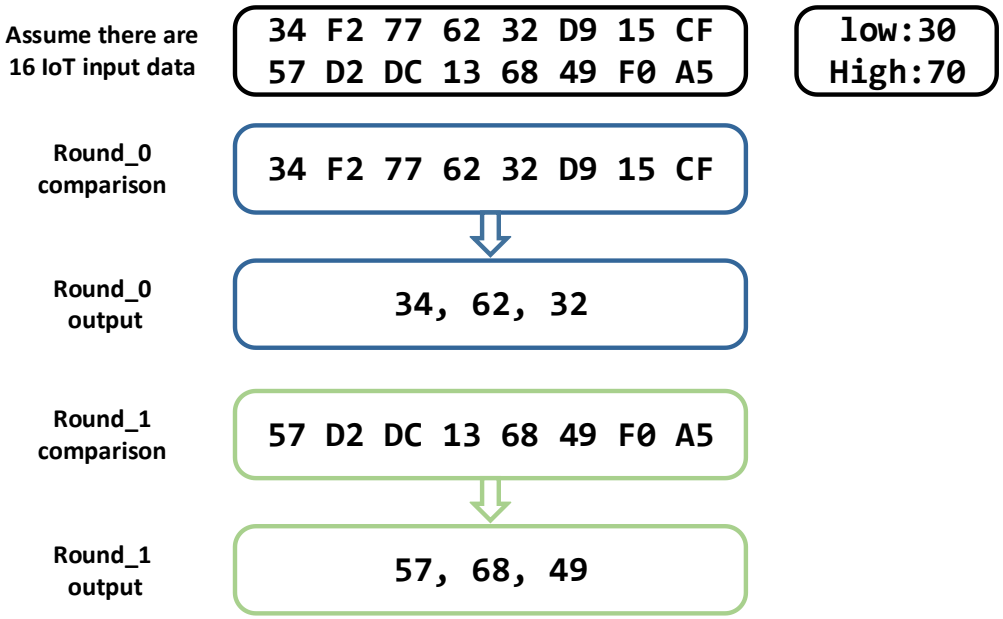
c. F3: Avg(N)

- Find the average in 8 IoT data for each round.
- Round down the output if the result is not integer



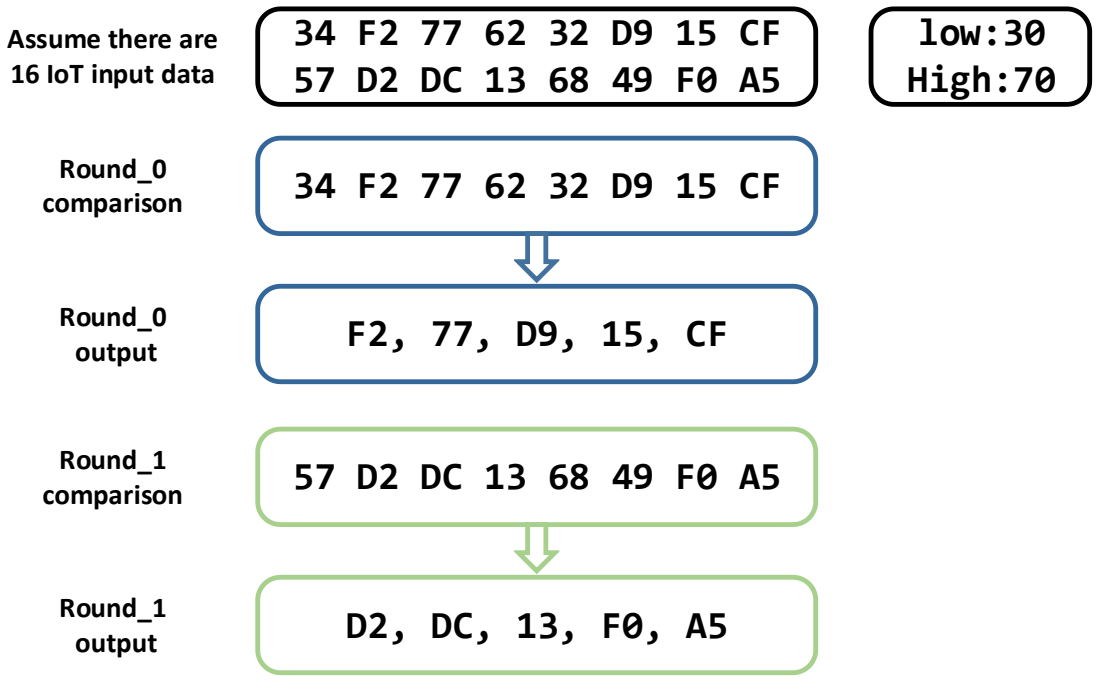
d. F4: Extract(low < data < high)

- Find the data between the known “low” value and the known “high” value.
- For the home work, the “low” and “high” value are set as follow:
  - Low: 128’h 6FFF_FFFF_FFFF_FFFF_FFFF_FFFF_FFFF_FFFF
  - High: 128’h AFFF_FFFF_FFFF_FFFF_FFFF_FFFF_FFFF_FFFF



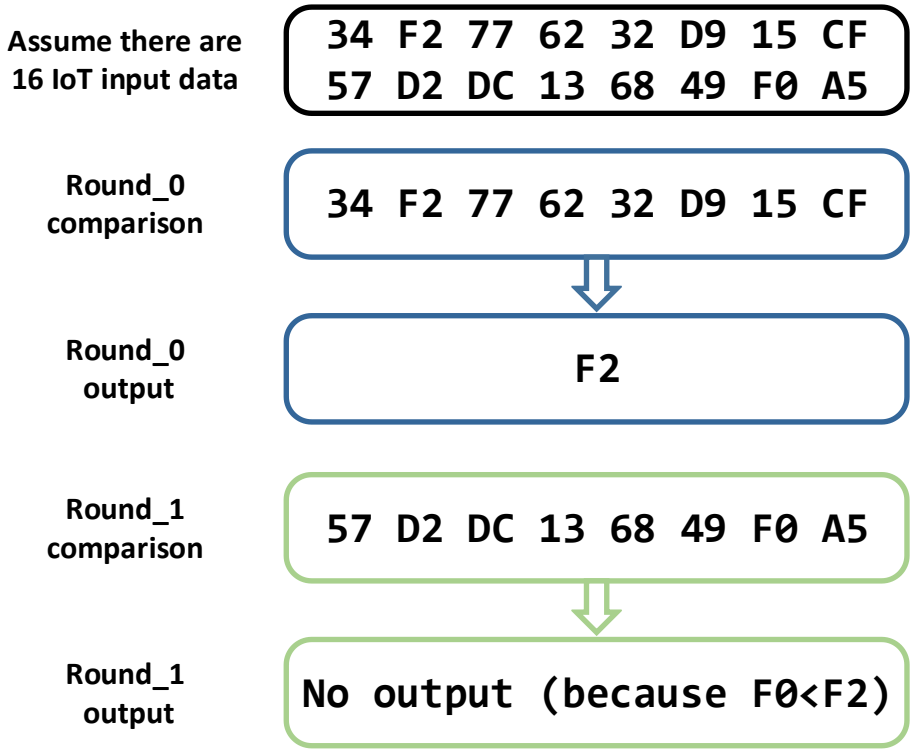
e. F5: Exclude(data<low , high<data)

- Find the data which is smaller than the known “low” value, or larger than the known “high” value.
- For the home work, the “low” and “high” value are set as follow:
  - Low: 128’h 7FFF_FFFF_FFFF_FFFF_FFFF_FFFF_FFFF_FFFF
  - High: 128’h BFFF_FFFF_FFFF_FFFF_FFFF_FFFF_FFFF_FFFF



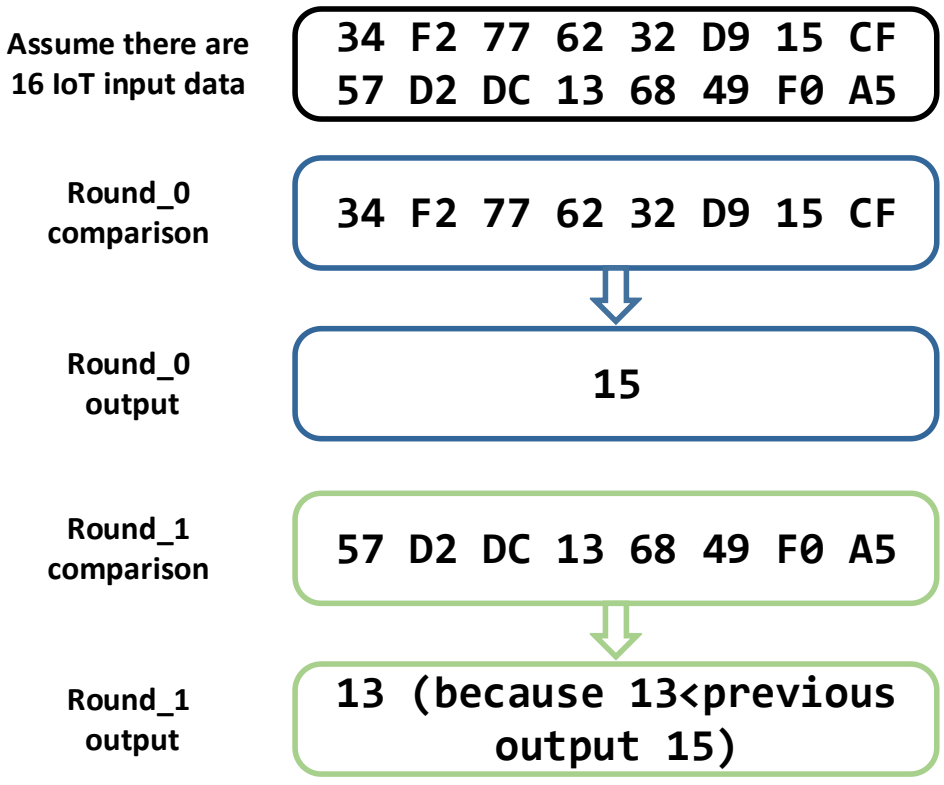
f. F6: PeakMax(the data is larger than previous output data)

- Find the largest data in round_0 first. For the rest of the round, output the data which is larger than previous output data.

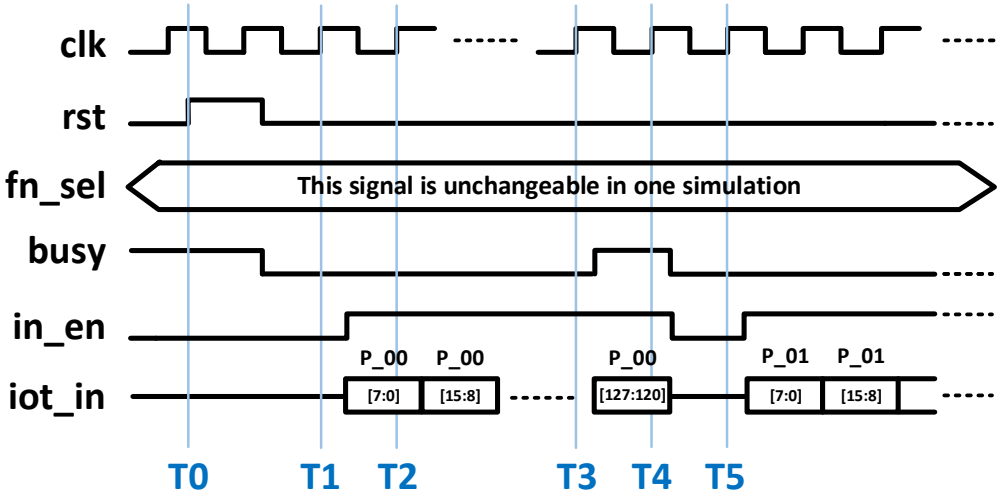


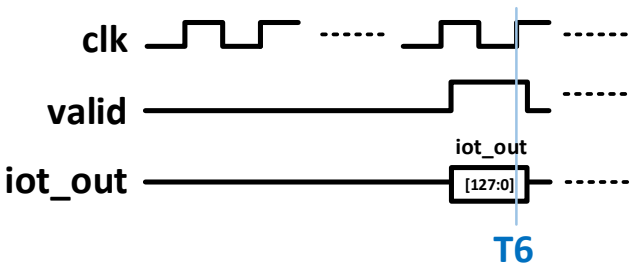
g. F7: PeakMin(the data is smaller than previous output data)

- Find the smallest round_0 first. For the rest of the round, output the data which is smaller than previous output data.



Timing Diagram





1. IOTDF is initialized between T0~T1.
2. in_en is set to high and start to input IoT data P_00[7:0] if busy is low at T1.
3. in_en is kept to high and input IoT data P_00[15:8] if busy is low at T2.
4. in_en is kept to high and input IoT data P_00[127:120] if busy is low at T3.
5. in_en is set to low and IoT data is set to 0 (stop streaming in data) if busy is high at T4.
6. There are 16 cycles between T1~T4 for one IoT data. You can set busy to high to stop steaming in data if you want.
7. You have to set valid to high if you want to output iot_out.

Hint

1. Clock gating can help reduce the power.
2. With registers optimization/sharing, the area will be much lower.
3. Pipeline can help cut down on process time.

Submission

1. Create a folder named **studentID_hw4**, and put all below files into the folder
   - **IOTDF.v**
   - **IOTDF_syn.v**
   - **IOTDF_syn.sdf**
   - **IOTDF_syn.area**
   - **F1_7.power (power report)**
   - **report.pdf**
   - **all other design files** included in your design for rtl simulation (optional)

   Note: Use **lower case** for the letter in your student ID. (Ex. r06943027_hw4)
2. Compress the folder **studentID_hw4** in a **tar file** named **StudentID_hw4_v*k*.tar (*k* is the number of version, *k* =1,2,…)**

```
tar -cvf StudentID_hw4_vk.tar StudentID_hw4
```

   TA will only check the last version of your homework.
3. Submit to NTU cool

Grading Policy

1. TA will use **runall_rtl** and **runall_syn** to run your code at RTL and gate-level simulation.
2. TAs will score you design with **Power**, **Time** and **Area**
   - Area: area from synthesis report (ex. 33988μm^2 below)

```
****************************************
Report : area
Design : IOTDF
Version: R-2020.09-SP5
Date   : Sun Oct 30 09:16:01 2022
****************************************

Library(s) Used:

    slow (File: /home/raid7_2/course/cvsd/CBDK_IC_Contest/CIC/SynopsysDC/db/slow.db)

Number of ports:                          201
Number of nets:                          3668
Number of cells:                         3347
Number of combinational cells:           2904
Number of sequential cells:               424
Number of macros/black boxes:               0
Number of buf/inv:                        467
Number of references:                      92

Combinational area:              20385.773806
Buf/Inv area:                     1699.097375
Noncombinational area:           13602.963221
Macro/Black Box area:                0.000000
Net Interconnect area:      undefined  (No wire load specified)

Total cell area:                 33988.737026
Total area:                 undefined
1
```

- Time: processing time from simulation (ex. 16590ns below)

```
ncsim> run
FSDB Dumper for IUS, Release Verdi3_L-2016.06-SP1-1, Linux, 09/27/2016
(C) 1996 - 2016 by Synopsys, Inc.
*Verdi3* : Create FSDB file 'IOTDF_F3.fsdb'
*Verdi3* : Begin traversing the scopes, layer (0).
*Verdi3* : End of traversing.
*Verdi3* : Begin traversing the MDAs, layer (0).
*Verdi3* : Enable +mda and +packedmda dumping.
*Verdi3* : End of traversing the MDAs.
----------------------------------------------------

Start to Send IOT Data & Compare ...


P00:  ** Correct!! **
P01:  ** Correct!! **
P02:  ** Correct!! **
P03:  ** Correct!! **
P04:  ** Correct!! **
P05:  ** Correct!! **
P06:  ** Correct!! **
P07:  ** Correct!! **
P08:  ** Correct!! **
P09:  ** Correct!! **
P10:  ** Correct!! **
P11:  ** Correct!! **

----------------------------------------------------

Congratulations! All data have been generated successfully!

-------------------------PASS-------------------------

Simulation complete via $finish(1) at time 16590 NS + 0
./testfixture.v:227       #(`CYCLE/2); $finish;
```

- Power: Use below command to analyze the power. (Need to source the following .cshrc file first!) (ex. 0.8994 mW below)

```
Unix% source /usr/cad/synopsys/CIC/primetime.cshrc
Unix% pt_shell -f ./pt_script.tcl | tee pp.log
```

```
                    Internal  Switching  Leakage    Total
Power Group         Power     Power      Power      Power   (     %)  Attrs
--------------------------------------------------------------------------------
clock_network       5.462e-04 1.347e-04 1.455e-06 6.824e-04 (75.88%)  i
register            9.190e-05 4.246e-05 2.589e-05 1.603e-04 (17.82%)
combinational       1.589e-05 3.080e-05 1.001e-05 5.670e-05 ( 6.30%)
sequential             0.0000    0.0000    0.0000    0.0000 ( 0.00%)
memory                 0.0000    0.0000    0.0000    0.0000 ( 0.00%)
io_pad                 0.0000    0.0000    0.0000    0.0000 ( 0.00%)
black_box              0.0000    0.0000    0.0000    0.0000 ( 0.00%)

  Net Switching Power  = 2.080e-04   (23.13%)
  Cell Internal Power  = 6.540e-04   (72.72%)
  Cell Leakage Power   = 3.736e-05   ( 4.15%)
                         ---------
Total Power            = 8.994e-04  (100.00%)

X Transition Power     = 1.331e-05
Glitching Power        = 1.149e-06

Peak Power             =    0.0857
Peak Time              = 15665.999
```

3. Different score for different level design
   - **Level A**: achieve following request and get **90pts**
     a. Function work, and pass all patterns in RTL-simulation
     b. Finish synthesis, and pass all patterns in gate-level simulation
     c. Your performance score is less than $\mathbf{10^9}$
     d. Cycle time for gate-level simulation is less than **100ns**

     Score for performance to be considered:

     Score = (Power1 × Time1 + … + Power7 × Time7) * Area

     Unit: power(mW), Time(ns), Area(um^2)

   - **Level B**: achieve following request and get **80pts**
     a. Function work, and pass all patterns in RTL-simulation
     b. Finish synthesis, and pass all patterns in gate-level simulation
     c. Your performance score is less than $\mathbf{2*10^9}$
     d. Cycle time for gate-level simulation is less than **100ns**

   - **Level C**: achieve following request and get **70pts**
     a. Function work, and pass all patterns in RTL-simulation
     b. Finish synthesis, and pass all patterns in gate-level simulation
     c. Your performance score is more than $\mathbf{2*10^9}$
     d. Cycle time for gate-level simulation is less than **100ns**

   - **Level D**: achieve following request and get **60pts**
     a. Function work, and pass all patterns in RTL-simulation
     b. Finish synthesis, but fail at least one pattern in gate-level simulation

   - **Level E**: achieve following request and get **50pts**
     a. Fail at least one pattern in RTL-simulation

4. Report: **10pts**

Reference

[1] IC Design Contest, 2019.