2017 Digital IC Design
Homework 3: Approximate Average

1. Introduction

Please design a computational system whose transfer function is defined as follow. A series of 8-bit positive integer is generated as the input of the computational system by the test bench. The output value Y is a 10-bit positive integer, which is calculated according to equations (1), (2), (3) and (4).

$$Xavg_j = \left\lfloor \frac{\sum_{i=j}^{j+n-1} X_i}{n} \right\rfloor \quad \text{......} \quad (1)$$

where X_i is the value of the ith input data and $j \geq 1$.

$$XS = \{X_j, X_{j+1}, X_{j+2}, \ldots, X_{j+n-1}\} \quad \text{......} \quad (2)$$

$$Xappr_j = \begin{cases} Xappr_j = Xavg_j \\ \quad \text{if } Xavg_j \in XS \\ X_i \,\middle|\, (X_i \in XS) \textit{ and } (X_i < Xavg_j) \text{ and } (Xavg_j - X_i \text{ is minimal}) \\ \quad \text{if } Xavg_j \notin XS \end{cases} \quad \text{......} \quad (3)$$

where $Xappr_j$ is the value of the jth approximate average.

$$Y_j = \left\lfloor \frac{\sum_{i=j}^{j+n-1} (X_i + Xappr_j)}{n-1} \right\rfloor \quad \text{......} \quad (4)$$

where Y_j is the value of the jth output data.

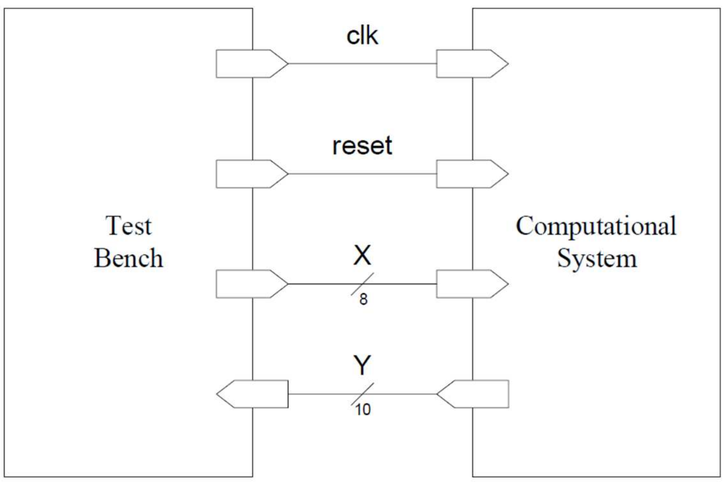
The computational system produces the output sequence according to the given input sequence. Each input and output data in the respective sequence is indexed. This index, in terms of hardware, is the relative time when the input data is given or the output data is ready. Thinking as a hardware designer, the approximate average is chosen from the last *n* input data which should be stored in the system. The system should be able to calculate the integral part of the real average of the last *n* input data first. And then if the integral part of the real average equals to any one of the last n input data, the approximate average is simply the integral part. Else the approximate average is the one which is one of the last n input data whose value is smaller than and closest to the integral part of the real average. The above descriptions stated the desired operations as those defined by equations (1), (2), and (3).

After the approximate average is obtained, the output value can be calculated according to equation (4). First, the last n input value is added by the corresponding approximate average. And then they are summed up and divided by $n-1$. The output value is the quotient after division.

For example, assume that n=4, X_1=3, X_2=24, X_3=16, X_4=8, and X_5=3. After the first 5 input items are given, the system should store them and calculate the output value. The average of the first 4 input values is 12(only the integral part is left). Since it is not in the set of {X_1, X_2, X_3, X_4}, the system selects one from {X_1, X_2, X_3, X_4} as the approximate average whose value is smaller than 12 and close to 12. In this case, the approximate average is 8. So the first output value is calculated n as $\lfloor[(3 + 8) + (24 + 8) + (16 + 8) + (8 + 8)] / [(4 - 1)]\rfloor = 27$. Similar to those described above, when the 5th input data item is given, the system should store X_2, X_3, X_4 and X_5 and calculate the corresponding output value. The 2nd output value should be the same as the first one because the values stored in the system is the same.

2. Design Specifications

2.1 Block Overview

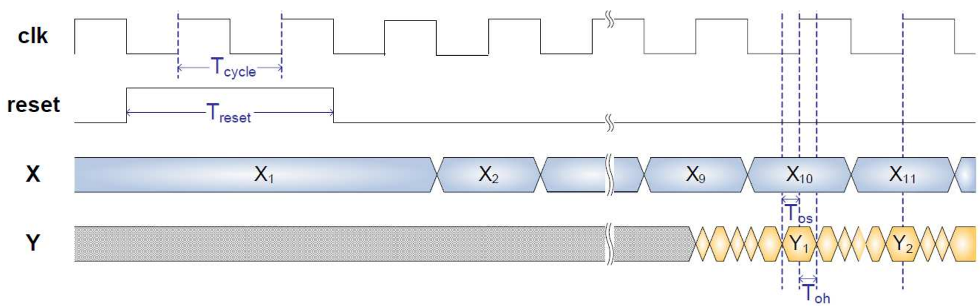


2.2 I/O Interface

Signal Name	I/O	width	Description
clk	I	1	clock for the computational system
reset	I	1	reset the state of the computational system when it asserts

X	I	8	input data of the computational system
Y	O	10	computed output

2.3 Timing Diagrams



Symbol	Description	Value
T_{cycle}	clock period	user defined
T_{reset}	reset pulse width	2 T_{cycle}
T_{os}	setup time from valid output to positive edge of *clk*	0.5ns
T_{oh}	hold time from positive edge of *clk* to invalid output	0.5ns

The I/O timing diagram is as shown above. For this homework, *n* in equations (1)-(4) are fixed to 9**.** The computational system is reset by asserting reset signal for 2 periods. The input X is changed to the next at the negative edges of the clock while output Y is checked by the test bench at positive edges of the clock. Note that the output should be stable around the positive edges of the clock. The setup and hold time requirements for the output are listed in Table. The first output data should be valid after the input data changes from 9th one to 10th and before the next positive clock edge. After that, the output should be changed to the next at the next positive clock edge and so on, that is to say, the test bench checks one output value per clock cycle.

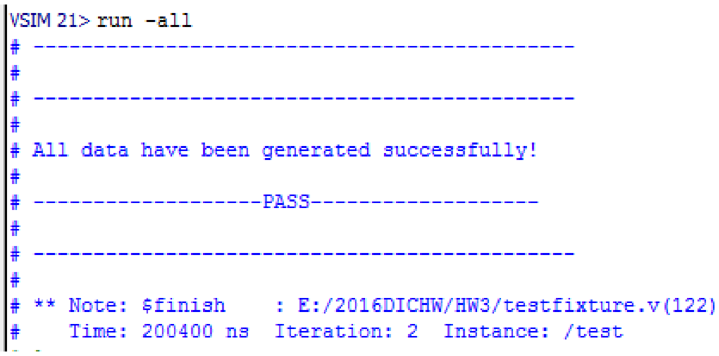
2.4 File Description

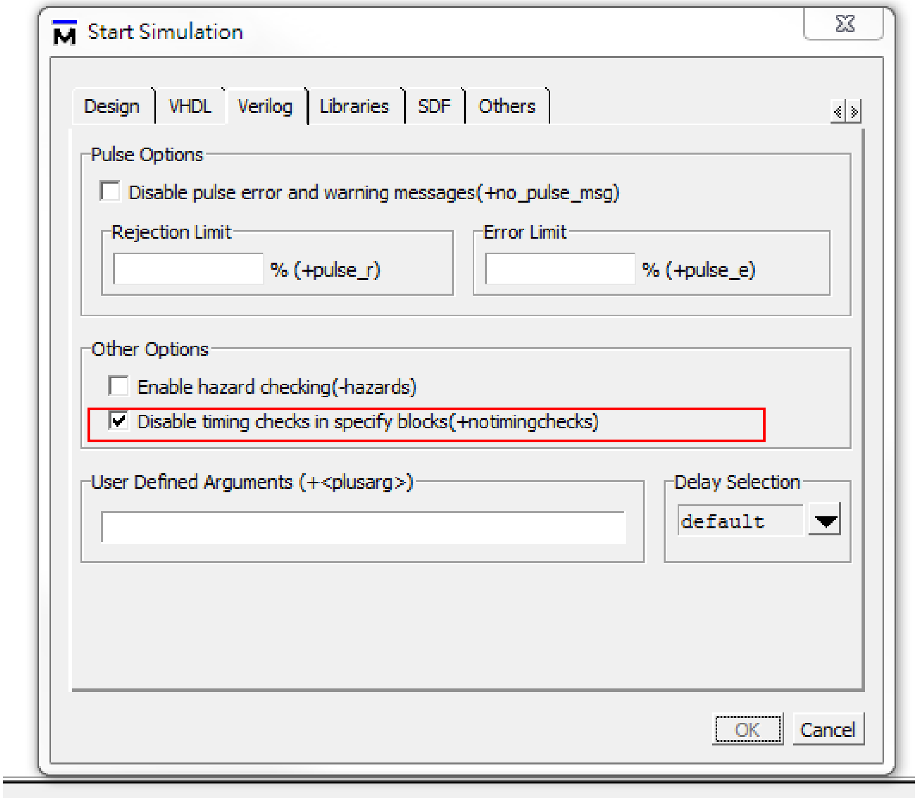
File Name	Description
CS.v	RTL code for using Verilog
testfixture.v	Test bench for verifying design
in.dat	Input patterns
out_golden.dat	Golden output patterns
cycloneii_atoms.v	Simulation library for gate-level simulation

3. Scoring

3.1 Functional Simulation (pre-sim) **[60%]**

All of the result should be generated correctly, and you will get the following message in ModleSim simulation. You can turn off the timing check in pre-sim only.





3.2 Gate-Level Simulation (post-sim) **[20%]**

3.2.1 Synthesis

Your code should be synthesizable. After synthesizing in Quartus, the file named *CS.vo* and CS.sdo will be obtained.

Device : Cyclone II EP2C70F896C8

3.2.2 Simulation

All of the result should be generated correctly using *CS.vo* and *CS.sdo*, and you will get the following message in ModleSim simulation. (There should be no setup or hold time violations.)

```
VSIM 14> run -all
# ----------------------------------------------
#
# ----------------------------------------------
#
# All data have been generated successfully!
#
# ------------------PASS-------------------
#
# ----------------------------------------------
#
# ** Note: $finish    : E:/2016DICHW/HW3/gatelevel/testfixture.v(122)
#    Time: 84168 ns  Iteration: 2  Instance: /test
# 1
# Break in Module test at E:/2016DICHW/HW3/gatelevel/testfixture.v line 122

VSIM 15>
```

3.3 Performance **[20%]**

The performance is scored by the logic elements you used and the simulation time in post-sim. The scoring equation is *(Total logic elements + total memory bit + 9*embedded multiplier 9-bit element)* × *(total simulation time in ns)*. (The smaller the better).

E:/2016DICHW/HW3/gatelevel/testfixture.v - Default

```
`timescale 1ns/10ps
`define CYCLE  42            // Modify your clock period here

`define INFILE "in.dat"
`define OUTFILE "out_golden.dat"

module test;
parameter INPUT DATA = `INFILE;
```

Flow Summary	
Flow Status	Successful - Mon Nov 07 21:21:34 2016
Quartus II Version	10.0 Build 262 08/18/2010 SP 1 SJ Full Version
Revision Name	CS
Top-level Entity Name	CS
Family	Cyclone II
Device	EP2C70F896C8
Timing Models	Final
Met timing requirements	Yes
Total logic elements	559 / 68,416 (< 1 %)
Total combinational functions	559 / 68,416 (< 1 %)
Dedicated logic registers	73 / 68,416 (< 1 %)
Total registers	73
Total pins	20 / 622 (3 %)
Total virtual pins	0
Total memory bits	0 / 1,152,000 (0 %)
Embedded Multiplier 9-bit elements	0 / 300 (0 %)
Total PLLs	0 / 4 (0 %)

```
VSIM 14> run -all
# ---------------------------------------------
#
# ---------------------------------------------
#
# All data have been generated successfully!
# |
# -------------------PASS-------------------
#
# ---------------------------------------------
#
# ** Note: $finish    : E:/2016DICHW/HW3/gatelevel/testfixture.v(122)
#    Time: 84168 ns  Iteration: 2  Instance: /test
# 1
# Break in Module test at E:/2016DICHW/HW3/gatelevel/testfixture.v line 122

VSIM 15>
```

4. Submission

4.1 Submitted files

You should classified your files into three directories and compressed to .zip format. The naming rule is **HW3_*studentID_name_version*.zip**. The *vision* is v1 for the first submission, and v2, v3… for the revisions.

RTL category	
*.v	All of your verilog RTL code
Gate-Level category	
*.vo	Gate-Level netlist generated by Quartus
*.sdo	SDF timing information generated by Quartus
Documentary category	
*.pdf	The report file of your design (in pdf).

4.2 Report file

You have to describe how the circuit is designed as detailed as possible, and the flow summary result and simulation results are necessary. **Please follow the specification in appendix.**

4.3 Please submit your .zip file to folder HW3 in the ftp site.

Deadline: 2017-11-28 23:59

ftp : 140.116.245.92

Usermame : ic_design

Password : icdesign

5. If you have any problem, please contact the TA by email:

weiting84610@gmail.com

p78031175@mail.ncku.edu.tw

Appendix

(Hexadecimal number)

index i	X_i	$Xavg_j$	$Xappr_j$	$\sum_{i=j}^{j+n-1}(X_i + Xappr_j)$	Y_j	index j
1	8f					
2	0b					
3	5d					
4	20					
5	f3					
6	3e					
7	e5					
8	03					
9	0c	5c	3e	056a	00ad	1
10	74	59	3e	054f	00a9	2
11	79	65	5d	06d4	00da	3
12	01	5b	3e	0561	00ac	4
13	30	5c	3e	0571	00ae	5
14	2e	46	3e	04ac	0095	6
15	a4	52	30	0494	0092	7
16	76	45	30	0425	0084	8
17	84	54	30	04a6	0094	9
18	51	5b	51	0614	00c2	10
19	d6	66	51	0676	00ce	11
20	70	65	51	066d	00cd	12
21	35	6b	51	06a1	00d4	13
22	10	68	51	0681	00d0	14
23	23	66	51	0676	00ce	15
24	e7	6e	51	06b9	00d7	16
25	3b	67	51	067e	00cf	17
26	6d	65	51	0667	00cc	18
27	34	61	3b	0584	00b0	19
28	61	54	3b	050f	00a1	20
29	89	57	3b	0528	00a5	21
30	bf	67	61	0708	00e1	22
31	dc	7d	6d	0840	0108	23
32	d3	91	89	09ec	013d	24
33	9c	88	6d	08a5	0114	25
34	8f	92	8f	0a2b	0145	26

(Decimal number)

index i	X_i	$Xavg_j$	$Xappr_j$	$\sum_{i=j}^{j+n-1}(X_i + Xappr_j)$	Y_j	*index j*
1	143					
2	11					
3	93					
4	32					
5	243					
6	62					
7	229					
8	3					
9	12	92	62	1386	173	1
10	116	89	62	1359	169	2
11	121	101	93	1748	218	3
12	1	91	62	1377	172	4
13	48	92	62	1393	174	5
14	46	70	62	1196	149	6
15	164	82	48	1172	146	7
16	118	69	48	1061	132	8
17	132	84	48	1190	148	9
18	81	91	81	1556	194	10
19	214	102	81	1654	206	11
20	112	101	81	1645	205	12
21	53	107	81	1697	212	13
22	16	104	81	1665	208	14
23	35	102	81	1654	206	15
24	231	110	81	1721	215	16
25	59	103	81	1662	207	17
26	109	101	81	1639	204	18
27	52	97	59	1412	176	19
28	97	84	59	1295	161	20
29	137	87	59	1320	165	21
30	191	103	97	1800	225	22
31	220	125	109	2112	264	23
32	211	145	137	2540	317	24
33	156	136	109	2213	276	25
34	143	146	143	2603	325	26

2017 Digital IC Design Homework 3: Approximate Average

NAME	
Student ID	

<table>
<tr><th colspan="6">Simulation Result</th></tr>
<tr><td>Functional simulation</td><td>Pass or Fail</td><td>Gate-level simulation</td><td>Pass or Fail</td><td>Gate-level simulation time</td><td>simulation time (ns)</td></tr>
<tr><td colspan="3">(your pre-sim result)</td><td colspan="3">(your post-sim result)</td></tr>
</table>

<table>
<tr><th colspan="2">Synthesis Result</th></tr>
<tr><td>Total logic elements</td><td></td></tr>
<tr><td>Total memory bit</td><td></td></tr>
<tr><td>Embedded multiplier 9-bit element</td><td></td></tr>
<tr><td colspan="2">(your flow summary)</td></tr>
</table>

Description of your design

*Scoring = (Total logic elements + total memory bit + 9*embedded multiplier 9-bit element) × (gate-level simulation time in ns)*