CPU Architecture

LAB1 preparation report

VHDL part1 – Concurrent code

Hanan Ribo

24/03/2022

Table of contents

1. Aim of the Laboratory

- Obtaining skills in VHDL part1 code, which contains Code Structure, Data Types, Operators and Attributes, Concurrent Code, Design Hierarchy, Packages and Components.
- Obtaining basic skills in ModelSim (multi-language HDL simulation environment).
- General knowledge rehearsal in digital systems.
- Proper analysis and understanding of architecture design.

2. System Design ISA

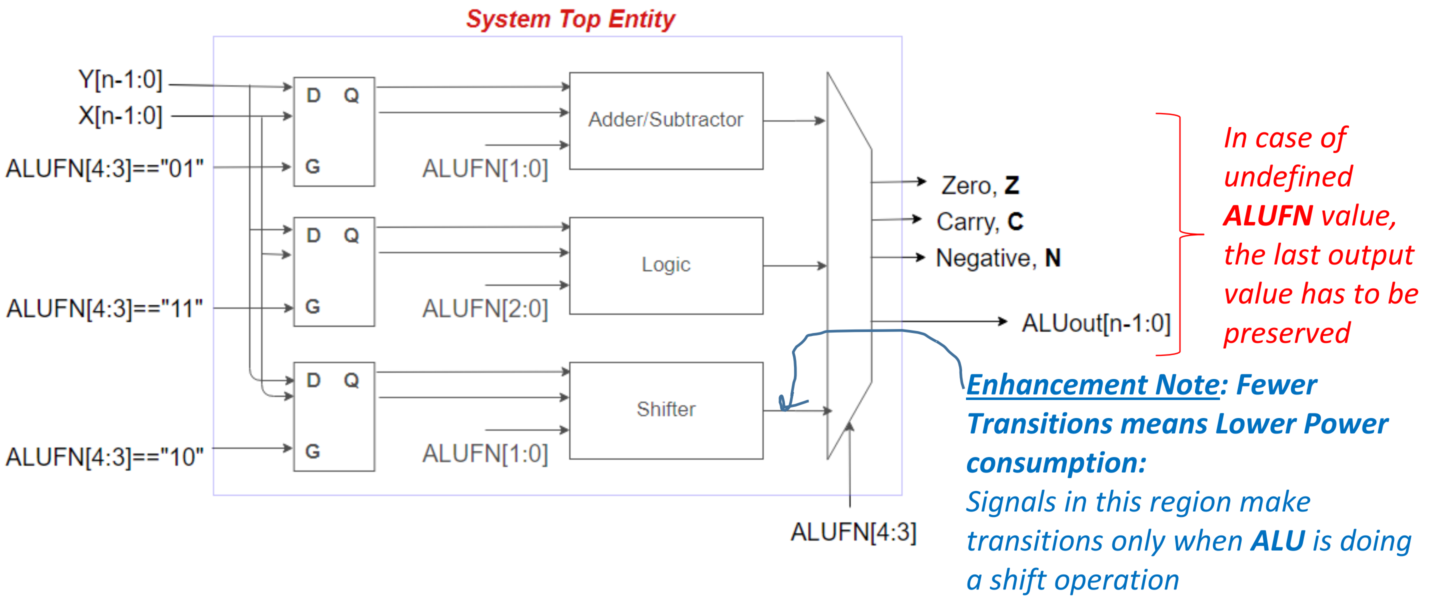
Function Kind	Decimal value	ALUFN	Operation	Note
Arithmetic	8	**01**000	Res=Y+X	
	9	**01**001	Res=Y-X	Used also for compare operation
	10	**01**010	Res=neg(X)	
Shift	16	**10**000	Res=SHL Y,X(k-1 to 0)	Shift Left Y of q≙X(k-1 ..0) times Res=Y(n-1-q…0)#(q@0) $\boldsymbol{When\ k = log_2 n}$
	17	**10**001	Res=SHR Y,X(k-1 to 0)	Shift Right Y of q≙X(k-1 ..0) times Res=(q@0)#Y(n-1...q) $\boldsymbol{When\ k = log_2 n}$
Boolean	24	**11**000	Res=not(Y)	
	25	**11**001	Res=Y or X	
	26	**11**010	Res=Y and X	
	27	**11**011	Res=Y xor X	
	28	**11**100	Res=Y nor X	
	29	**11**101	Res=Y nand X	

Table 1: Selected operations

3. System Design Micro-Architecture

In this laboratory you will design a module which contains the next three sub-modules:

- Generic Adder/Subtractor module between two vectors Y, X of size n-bit (the default is n=8).
- Generic Shifter module based on Barrel-Shifter n-bit size (the default is n=8).
- Boolean Logic operates bitwise.
- The generic n value can be 4,8,16,32 (set from ***tb.vhd*** file).
- You are required to design the whole system and make a test bench for testing.


System Top Entity
Y[n-1:0]
X[n-1:0]
ALUFN[4:3]=="01"
ALUFN[4:3]=="11"
ALUFN[4:3]=="10"
D Q
G
Adder/Subtractor
ALUFN[1:0]
Logic
ALUFN[2:0]
Shifter
ALUFN[1:0]
ALUFN[4:3]
Zero, Z
Carry, C
Negative, N
ALUout[n-1:0]
In case of undefined ALUFN value, the last output value has to be preserved
Enhancement Note: Fewer Transitions means Lower Power consumption:
Signals in this region make transitions only when ALU is doing a shift operation

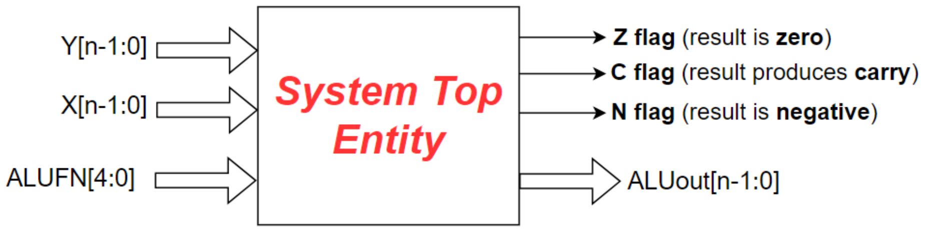
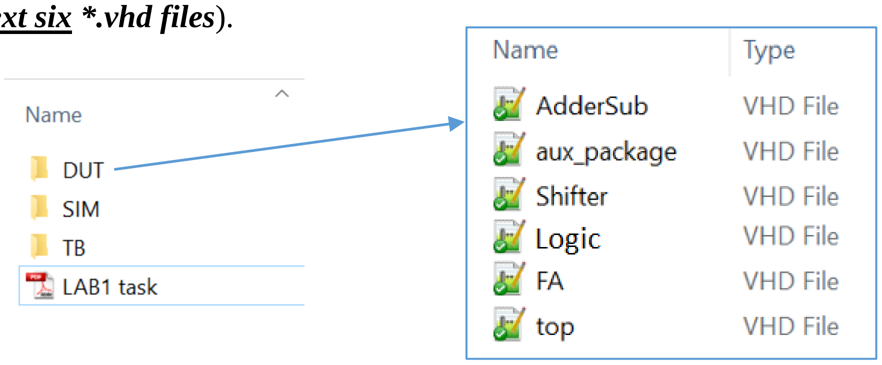


Figure 1 : System top level structure

- The Top Level design must be Structural (***your DUT must contain the <u>exact next six</u> *.vhd files***).



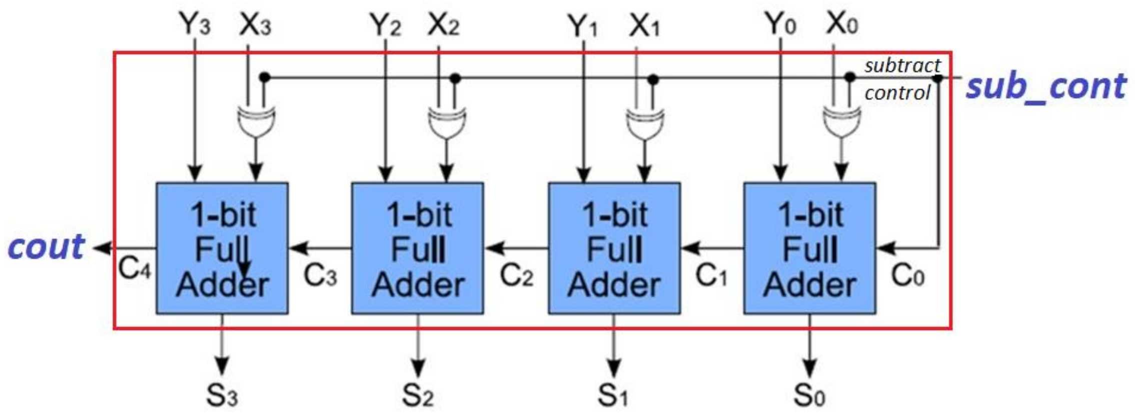
Name
DUT
SIM
TB
LAB1 task

Name	Type
AdderSub	VHD File
aux_package	VHD File
Shifter	VHD File
Logic	VHD File
FA	VHD File
top	VHD File

- **Note:** in the given two files **top.vhd**, **aux_package.vhd (**you can only add code to these files, ***you are not allowed to erase anything*****).**
- The submitted project must be compiled using the given **tb_ref1.vhd** file (otherwise the submitted project will be considered as a failure).
- The submitted assignments get through copy checking machine, in this case, both sides' assignments will be disqualified.

4. Generic Adder/Subtractor module based on a single ripple carry adder:

- You are required to design a Generic Adder/Subtractor between two vectors Y, X of size n-bit (the default is n=8), using the next diagram. The design must be Structural of a least two levels.
- In order to do so, you are asked to use the Generic Adder code that was given in Moodle.



Note: this figure illustrates $Y \pm X$ operation

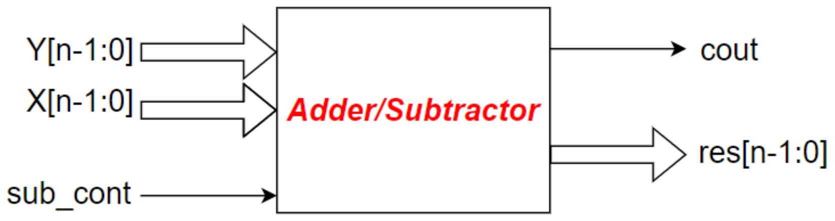
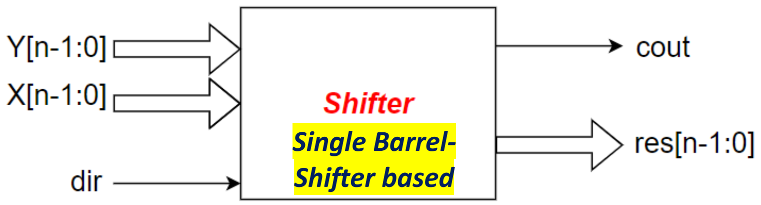


Figure 2: Generic Adder/Subtractor (based on a single ripple carry adder)

5. Shifter Module based on a single Barrel-Shifter n-bit:

Note: using **sll, srl** operators are forbidden and causes to disqualification of this clause.
Hint: In order to meet the requirements you must use ***generate*** concurrent statement



Note: this figure illustrates the operation ***Shift*** *X, Y (2* ***downto*** *0)*

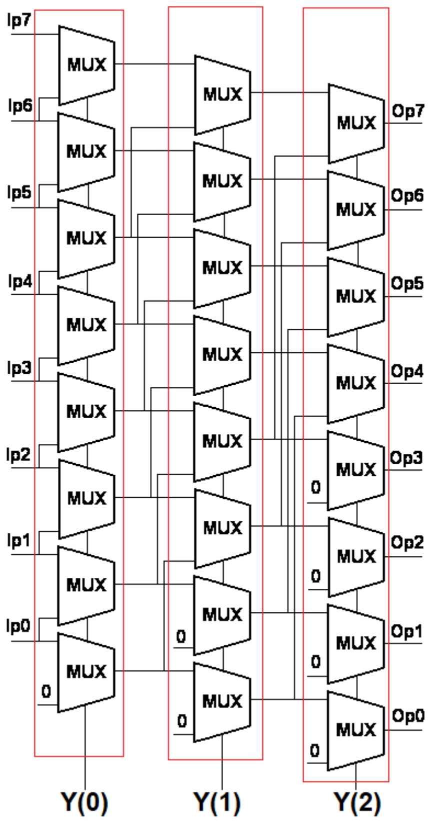


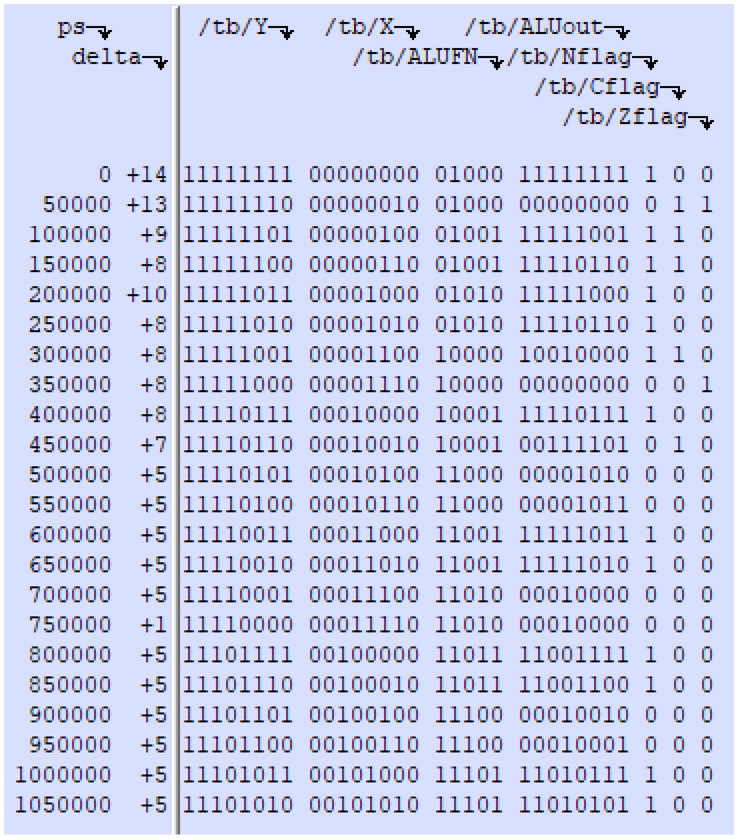
Figure 3: Example of 8-bit Barrel Shifter

6. Test and Timing:

- Design a test bench which tests all the system.
- Analyze the results by zooming on the important transactions in the waveforms. explain these (input/output/internal signals of the system).
- You are welcome to use the Internet also as reference.
- **Good tip for beginners**: Build a test bench for each module you are designing for easy debugging, otherwise you will waste a lot of time for whole system debug.
- The timing of the system will be ideal (means a functional simulation).

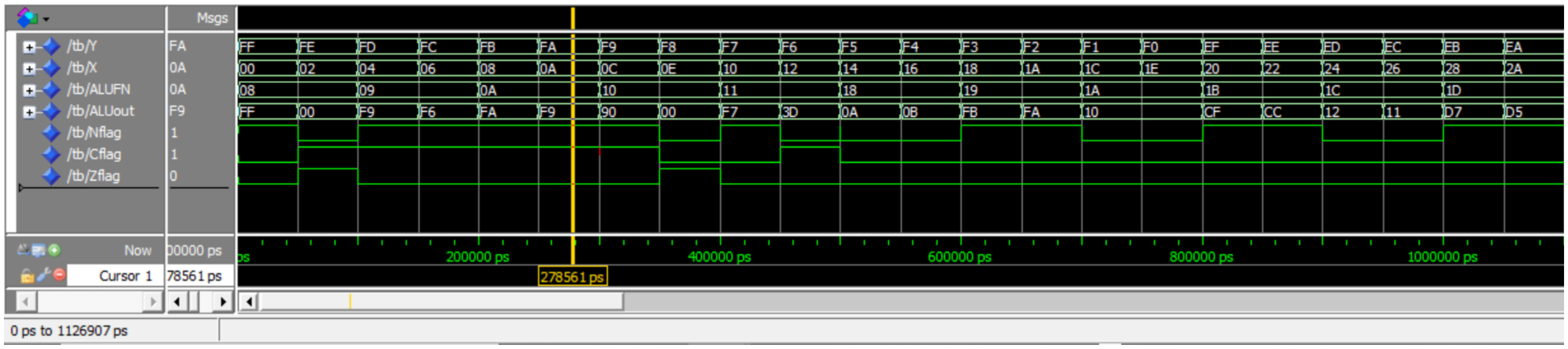
7. System Output Example (for n=8)

Signal vectors are shown in binary format



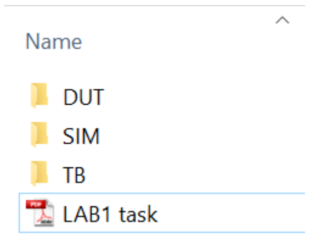
```
     ps        /tb/Y     /tb/X     /tb/ALUout
   delta                /tb/ALUFN  /tb/Nflag
                                     /tb/Cflag
                                       /tb/Zflag

      0 +14 11111111 00000000 01000 11111111 1 0 0
  50000 +13 11111110 00000010 01000 00000000 0 1 1
 100000  +9 11111101 00000100 01001 11111001 1 1 0
 150000  +8 11111100 00000110 01001 11110110 1 1 0
 200000 +10 11111011 00001000 01010 11111000 1 0 0
 250000  +8 11111010 00001010 01010 11110110 1 0 0
 300000  +8 11111001 00001100 10000 10010000 1 1 0
 350000  +8 11111000 00001110 10000 00000000 0 0 1
 400000  +8 11110111 00010000 10001 11110111 1 0 0
 450000  +7 11110110 00010010 10001 00111101 0 1 0
 500000  +5 11110101 00010100 11000 00001010 0 0 0
 550000  +5 11110100 00010110 11000 00001011 0 0 0
 600000  +5 11110011 00011000 11001 11111011 1 0 0
 650000  +5 11110010 00011010 11001 11111010 1 0 0
 700000  +5 11110001 00011100 11010 00010000 0 0 0
 750000  +1 11110000 00011110 11010 00010000 0 0 0
 800000  +5 11101111 00100000 11011 11001111 1 0 0
 850000  +5 11101110 00100010 11011 11001100 1 0 0
 900000  +5 11101101 00100100 11100 00010010 0 0 0
 950000  +5 11101100 00100110 11100 00010001 0 0 0
1000000  +5 11101011 00101000 11101 11010111 1 0 0
1050000  +5 11101010 00101010 11101 11010101 1 0 0
```

Signals ***Y, X, ALUout*** are shown in HEX format (same exact example as above).



Msgs
/tb/Y FA
/tb/X 0A
/tb/ALUFN 0A
/tb/ALUout F9
/tb/Nflag 1
/tb/Cflag 1
/tb/Zflag 0
FF FE FD FC FB FA F9 F8 F7 F6 F5 F4 F3 F2 F1 F0 EF EE ED EC EB EA
00 02 04 06 08 0A 0C 0E 10 12 14 16 18 1A 1C 1E 20 22 24 26 28 2A
08 09 0A 10 11 18 19 1A 1B 1C 1D
FF 00 F9 F6 FA F9 90 00 F7 3D 0A 0B FB FA 10 CF CC 12 11 D7 D5
Now 00000 ps
Cursor 1 78561 ps
200000 ps
400000 ps
600000 ps
800000 ps
1000000 ps
278561 ps
0 ps to 1126907 ps

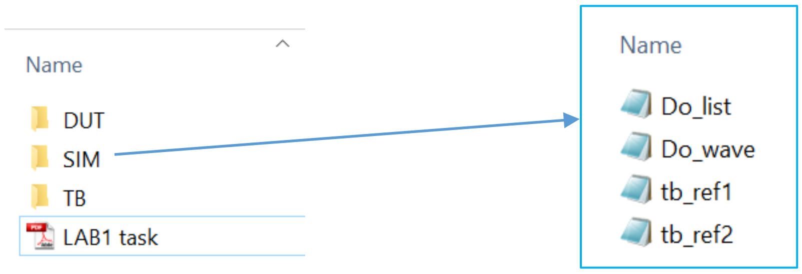
In addition, you are given two test bench files ***tb_ref1.vhd, tb_ref2.vhd*** (in TB folder) and their associate do and list files (in SIM folder).



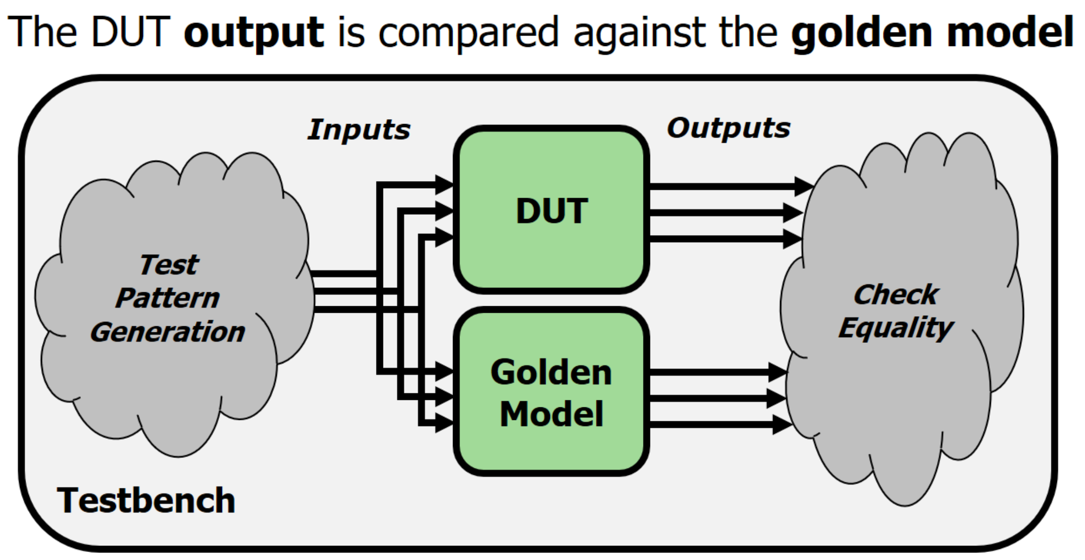
Name

DUT
SIM
TB
LAB1 task

In order to use golden model based functional verification, you are given TextDiff application (download and double click the ***TextDiff.exe*** file) in order to compare your developing design results to the golden model results as part of design developing chain.



Automatic Testbench



Note: in comparison between the given ***tb_ref1.lst, tb_ref2.lst*** files and yours, you should ignore the ***delta cycle*** column

8. Requirements

1. The lab assignment is in pairs (as shown in the inlay file).
2. The design must be well commented.
3. **Important:** For each of two submodules:
   - Graphical description (a square with ports going in and out) and short descriptions.
4. Elaborated analysis and wave forms:
   - Remove irrelevant signals.
   - Zoom on regions of interest.
   - Draw clouds on the waveform with explanations of what is happening (Figure 4).
   - Change the waveform colors in ModelSim for clear documentation **(Tools->Edit Preferences->Wave Windows).**
5. A ZIP file in the form of **id1_id2.zip** (where id1 and id2 are the identification number of the submitters, and id1 < id2) *must be upload to Moodle only by student with id1* (any of these rules violation disqualify the task submission).
6. The **ZIP** file will contain (***only the exact next sub folders***):

Directory	Contains	Comments
DUT	Project VHDL files	**Only VHDL files of DUT**, excluding test bench **Note: your project files must be well compiled without errors as a basic condition before submission**
TB	***Four*** VHDL files that are used for test bench	AdderSub, Logic, shifter, System (top)
SIM	***Four*** ModelSim DO files (wave, list)	AdderSub, Logic, shifter, System (top)
DOC	Project documentation	• ***readme.txt*** (list of the DUT **.vhd* files with their brief functional description) • ***pre1.pdf*** (report file that includes brief explanation of the four modules with their wave diagrams as shown in figure 4)

Table 2: Directory Structure

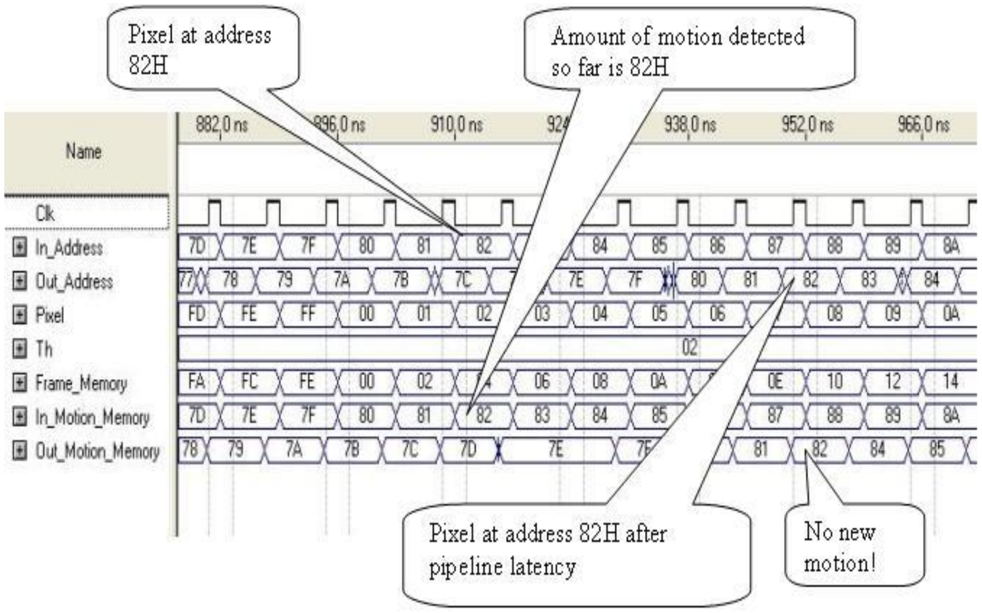


Figure 4: Clouds over the waveform

9. Grading Policy

Weight	Task	Description
10%	Documentation	The "clear" way in which you presented the requirements and the analysis and conclusions on the work you've done
90%	Analysis and Test	The correct analysis of the system (under the requirements)

Table 1 : Grading

Under the above policies you'll be also evaluated using common sense:

- Your files will be compiled and checked, the system must work.
- Your design and architecture must be intelligent, minimal, effective and well organized.

For a late submission the penalty is 2^{days}