Experiment: #9:

 Magnitude Comparator

Marlon Myers

Digital Control

EMT1250

Professor: Prof. Patrick

Table of Contents:

|  |  |
| --- | --- |
| Objective | 3 |
| Abstract | 3 |
| Schematic | 4 |
| VHDL code  | 5 |
|

|  |  |
| --- | --- |
| Waveform |  |

 | 9 |
| Data Results | 10 |
|  |  |

Objective:

* To build 1 - bit and 2 - bit Magnitude Comparator circuit s using the Quartus II development software with the DE - 2 board.
* To test the design by downloading the file into the DE - 2 board, exercising the inputs with toggle switches and observing 3 individual LED’s.

Abstract:

A Magnitude Comparator compares one and two binary numbers and indicates whether the numbers are equal or which number is larger. The Comparator will take inputs from 4 toggle switches ( SW[ 0] ~SW[3] ) representing two 2 bit binary numbers and activate LED G[ 0] ~LEDG[ 3 ] in the DE - 2 board . The input numbers will be labeled A0, A1 and B0, B1 and the outputs will be labeled AeqB, AltB, and AgtB. The LED’s are activated under the following condition.

|  |  |  |  |
| --- | --- | --- | --- |
| Input | AgtB | AeqB | AltB |
| A>B | On | Off | Off |
| A=B | Off | On | Off |
| A<B | Off | Off | On  |

|  |  |  |
| --- | --- | --- |
| Name | Pin Number | Switches |
| A1 | Pin\_AE14 | SW[3] |
| A0 | PIN\_P25 | SW[2] |
| B1 | PIN\_N26 | SW[1] |
| B0 | PIN\_N25 | SW[0] |

|  |  |  |
| --- | --- | --- |
| Name | Pin Number | LED |
| AgtB | Pin\_W19 | LEDG[2] |
| AeqB | Pin\_AF22 | LEDG[1] |
| AltB | Pin\_AE22 | LEDG[0] |

Schematic:

VHDL Code:

Library ieee;

Use ieee.std\_logic\_1164.All;

Entity XNORckt is

Port(

A,B : In std\_logic;

AXnorB : out std\_logic);

End XNORckt;

ARCHITECTURE behavior OF XNORckt IS

Begin Process(A ,B )

Begin

AXnorB <= A XNOR B;

END PROCESS;

End Behavior;

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

Library ieee;

Use ieee.std\_logic\_1164.All;

Entity andckt IS

Port(

X,Y: IN std\_logic;

XandY : OUT std\_logic);

END andckt;

ARCHITECTURE behavior OF andckt IS

BEGIN

PROCESS(X,Y)

BEGIN

XandY <= X AND Y; --and gate operation

END PROCESS;

END behavior;

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

Library ieee;

Use ieee.std\_logic\_1164.All;

Entity Notckt IS

Port(

A1 : IN std\_logic;

A1Not : Out std\_logic);

End Notckt;

ARCHITECTURE behavior of Notckt IS

Begin Process(A1)

Begin

A1Not <= Not A1;

End Process;

End Behavior;

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

Library ieee;

Use ieee.std\_logic\_1164.All;

Entity NANDckt Is

Port(

A,B : IN std\_logic;

ANANDB : Out std\_logic);

End NANDckt;

ARCHITECTURE behavior of NANDckt IS

Begin Process(A,B)

Begin

ANANDB <= A NAND B;

End Process;

End Behavior;

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

Library ieee;

Use ieee.std\_logic\_1164.All;

Entity NANDckt2 IS

Port(

A,B,C : IN std\_logic;

ANANDBNANDC4 : OUT std\_logic);

End NANDckt2;

ARCHITECTURE behavior of NANDckt2 IS

Begin Process(A,B,C)

Begin

ANANDBNANDC4 <= (A NAND B) NAND C;

End Process;

End Behavior;

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

Library ieee;

Use ieee.std\_logic\_1164.All;

Entity NotNotckt IS

Port(

A1 : IN std\_logic;

A1NotNot : Out std\_logic);

End NotNotckt;

ARCHITECTURE behavior of NotNotckt IS

Begin Process(A1)

Begin

A1NotNot <= Not (Not A1);

End Process;

End Behavior;

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

Library ieee;

Use ieee.std\_logic\_1164.All;

Entity NORCKTOUT IS

Port(

D, E : IN std\_logic;

DNORE : OUT std\_logic);

End NORCKTOUT;

ARCHITECTURE behavior of NORCKTOUT IS

Begin Process(D,E)

Begin

DNORE <= D NOR E;

End Process;

End Behavior;

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

Library ieee;

Use ieee.Std\_logic\_1164.All;

USE Work.All; -- Top level Entity

ENTITY Lab9\_3 IS

Port(

Input\_A0, Input\_B0, Input\_A1, Input\_B1 : IN std\_logic;

GtOutput,LtOutput,EqOutput : Out std\_logic);

END Lab9\_3;

ARCHITECTURE struct OF Lab9\_3 IS

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

COMPONENT XNORckt IS

Port(

A,B : In std\_logic;

AXnorB : out std\_logic);

End COMPONENT;

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

COMPONENT andckt IS

Port(

X,Y : IN std\_logic;

XandY : OUT std\_logic);

END COMPONENT;

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

COMPONENT Notckt IS

Port(

A1 : IN std\_logic;

A1Not : Out std\_logic);

End COMPONENT;

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

COMPONENT NANDckt Is

Port(

A,B : IN std\_logic;

ANANDB : Out std\_logic);

End COMPONENT;

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

COMPONENT NANDckt2 IS

Port(

A,B,C : IN std\_logic;

ANANDBNANDC4 : OUT std\_logic);

End COMPONENT;

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

COMPONENT NotNotckt IS

Port(

A1 : IN std\_logic;

A1NotNot : OUT std\_logic);

End COMPONENT;

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

COMPONENT NORCKTOUT IS

Port(

D,E : IN std\_logic;

DNORE : OUT std\_logic);

End COMPONENT;

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

SIGNAL wire\_1, wire\_2, wire\_4, wire\_5: std\_logic;

SIGNAL wire\_6, wire\_7, wire\_8: std\_logic;

SIGNAL wire\_A, wire\_B : std\_logic;

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

Begin

Gate1: XNORckt port map (A => Input\_A0, B => Input\_B0, AXnorB => wire\_1);

Gate2: XNORckt port map (A => Input\_A1, B => Input\_B1, AXnorB => wire\_2);

Gate3: andckt port map (X => wire\_1, Y => wire\_2, XandY => wire\_A);

Gate4: NotNotckt port map (A1 => wire\_A, A1NotNot => EqOutput);

Gate5: Notckt port map (A1 => Input\_A1, A1Not => wire\_4);

Gate6: Notckt port map (A1 => Input\_A0, A1Not => wire\_5);

Gate7: NANDckt port map (A => Input\_B1, B => wire\_4, ANANDB => wire\_6);

Gate8: NANDckt2 port map (A => wire\_4, B => wire\_5, C => Input\_B0, ANANDBNANDC4 => wire\_7);

Gate9: NANDckt2 port map (A => wire\_5, B => Input\_B1, C => Input\_B0, ANANDBNANDC4 => wire\_8);

Gate10: NANDckt2 port map (A => wire\_6, B => wire\_7, C => wire\_8, ANANDBNANDC4 => wire\_B);

Gate11: NotNotckt port map (A1 => wire\_A, A1NotNot => LtOutput);

Gate12: NORCKTOUT port map (D => wire\_A, E => wire\_B, DnorE => GtOutput);

END Architecture struct

Wave Form:

Data Results:

|  |  |
| --- | --- |
| Input | Output |
| A1 | A0 | B1 | B0 | AgtB | AeqB | AltB |
| Off | Off | Off | Off | 0 | 1 | 0 |
| Off | Off | Off | On | 0 | 0 | 1 |
| Off | Off | On | Off | 0 | 0 | 1 |
| Off | Off | On | On | 0 | 0 | 1 |
| Off | On | Off | Off | 1 | 0 | 0 |
| Off | On | Off | On | 0 | 1 | 0 |
| Off | On | On | Off | 0 | 0 | 1 |
| Off | On | On | On | 0 | 0 | 1 |
| On | Off | Off | Off | 1 | 0 | 0 |
| On | Off | Off | On | 1 | 0 | 0 |
| On | Off | On | Off | 0 | 1 | 0 |
| On | Off | On | On | 0 | 0 | 1 |
| On | On | Off | Off | 1 | 0 | 0 |
| On | On | Off | On | 1 | 0 | 0 |
| On | On | On | Off | 1 | 0 | 0 |
| On | On | On | On | 0 | 1 | 0 |