B i r z e i t U n i v e r s i t y

Faculty of Information Technology

Computer Systems Engineering

Department

Digital Lab ENCS211

PRE LAB EXP3

**Daoud Shaheen 1090395**

Design, construct, and test a circuit which uses an SN74151 to implement a sum-of-products

expression.

**(a)** Convert the following expression into summation form (i.e. F(A,B,C)= Σ(…)):



Y=AB’(C+C’)+(A+A’)B’C

=AB’C+AB’C’+AB’C+A’B’C

= AB’C+AB’C’+A’B’C

= Σ(1,4,5)

(b) Sketch on figure.6 the input connections necessary to implement the function in **a**

****

**Figure.6: 8-to-1 Multiplexer**

(c) Refer to function in (a) to fill in the table bellow (table.2)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **OUTPUTS** | | **INPUTS** | | |
| **Y'** | **Y** | **C** | **B** | **A** |
| **H** | **L** | **L** | **L** | **L** |
| **L** | **H** | **H** | **L** | **L** |
| **H** | **L** | **L** | **H** | **L** |
| **H** | **L** | **H** | **H** | **L** |
| **L** | **H** | **L** | **L** | **H** |
| **L** | **H** | **H** | **L** | **H** |
| **H** | **L** | **L** | **H** | **H** |
| **H** | **L** | **H** | **H** | **H** |

**Table.2**

**3.** Design, construct, and test a circuit which uses an SN74138 demultiplexer to implement a sumof-

products expression.

1. Convert the following expression into summation form (i.e. F(A,B,C)= Σ(…)):



Y=A’BC+(A+A’)BC’

=A'BC+ABC'+A'BC'

Y=F(A,B,C)= Σ(2,3,6)

**(b)** The demultiplexer output is selected, and will go low, by the address on inputs A, B, and C

when the IC is enabled. Therefore, we can create the output function Y by summing together the

outputs indicated by the summation form of Expression 1.2. Since the outputs of the

demultiplexer are active- low, this is done with a NAND gate. Connect each of the TRUE

minterm outputs of the demultiplexer in Figure.7 (indicated by the summation equation) to an

input of the NAND gate. Connect all unused NAND inputs to logic 1.

****

**Figure.7: 3-to-8 DeMUX**

**Part A:** Complete the following truth table for a priority encoder.

Assume the priority order (from highest to lowest) is IN2, IN0, IN3,

and IN1.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **VALID** | **OUT0** | **OUT1** | **IN3** | **IN2** | **IN1** | **IN0** |
| **0** | **X** | **X** | **0** | **0** | **0** | **0** |
| **1** | **0** | **0** | **X** | **0** | **X** | **1** |
| **1** | **1** | **0** | **0** | **0** | **1** | **0** |
| **1** | **0** | **1** | **X** | **1** | **X** | **X** |
| **1** | **1** | **1** | **1** | **0** | **X** | **0** |

**Part B:** Implement the following priority encoder using basic gates

(AND, OR, NAND, NOR, and NOT).Label all inputs and outputs.



to find out1:

m0 =X

mxxx1=0 🡪 m0001=m1

m0011=m3

m0101=m5

m1001=m9

m1011=m11

m1101=m13

m1111=m15

mxx10=0 🡪 m0010=m2

m0110=m6

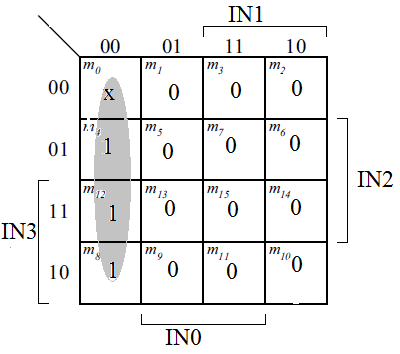
m1010=10

m1110=m14

mx100=1 -🡪 m0100=m4

m1100=m12

m8=1



OUT1= IN0’.IN1’

to find out0:

m0 =X

mxxx1=0 🡪 m0001=m1

m0011=m3

m0101=m5

m1001=m9

m1011=m11

m1101=m13

m1111=m15

mxx10=1 🡪 m0010=m2

m0110=m6

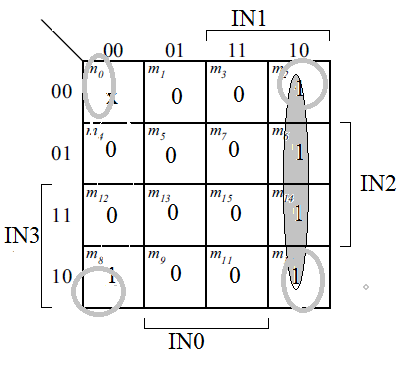
m1010=10

m1110=m14

mx100=0 -🡪 m0100=m4

m1100=m12

m8=1



OUT0=IN0’.IN1+IN0’.IN2’

=IN0’(IN1+IN2’)

VALID =IN0+ IN1+ IN2+ IN3



**Part C:** Design a Full Adder using a 3x8 decoder and any basic gates

needed.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inputs | | | Outputs | |
| A | B | Cin | S | Cout |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

S= Σ(1,2,4,7)

COUT= Σ(3,5,6,7)

