Implementation of 4x1 mux in Arduino using Assembly

Sai Harshith Kalithkar
harshith.work@gmail.com
FWC22118
IIT Hyderabad-Future Wireless Communication Assignment

March 2023

Contents

1 Problem

(GATE EC-2022)

Q.19. Consider the 2-bit multiplexer(MUX) shown in the figure.For output to be the XOR of R and S,the values for W, X, Y and Z are ?

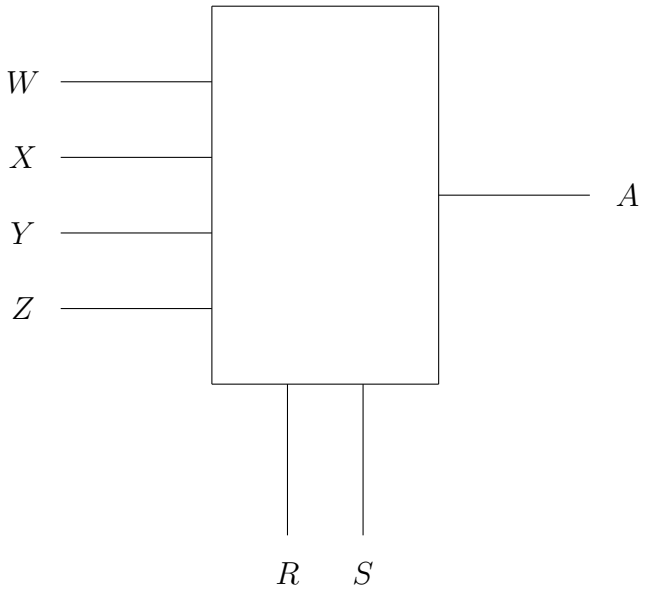

Figure 1: mux

1. $W = 0, X = 0, Y = 1, Z = 1$
2. $W = 1, X = 0, Y = 1, Z = 0$
3. $W = 0, X = 1, Y = 1, Z = 0$
4. $W = 1, X = 1, Y = 0, Z = 0$

2 Introduction

The above diagram is a 4:1 multiplexer where W, X, Y, Z are the inputs of the multiplexer and A is the output of the multiplexer.R, S are the select

lines of the multiplexer,which means:

1. For $R = 0, S = 0$,the first input line W is selected.
2. For $R = 0, S = 1$,the second input line X is selected.
3. For $R = 1, S = 0$,the third input line Y is selected.
4. For $R = 1, S = 1$,the fourth input line Z is selected.

Therefore,the resultant output expression of the multiplexer is $R'S'W + R'SX + RS'Y + RSZ$.

3 Components

COMPONENTS		
Component	Value	Quantity
Resistor	220 ohm	1
Arduino	UNO	1
Seven Segment Display		1
Jumper Wires	M-M	20
Breadboard		1

Table 1: contents

4 Hardware

1. Connect the COM of the seven-segment display to 5V and dot of the seven-segment to the ground.
2. Now connect any one of the pin of the seven-segment to pin no.2(digital).
3. Pin no.s 5,6,7,8 of the arduino should be initially connected to ground.
4. Now move pin no.s 5,6,7,8 accordingly and for the right combination the second pin of the arduino becomes high and the seven segement display glows.

Truth table		
R	S	A
0	0	0
0	1	1
1	0	1
1	1	0

Table 2: truth table

The K-map for this truth table will be a two variable K-map and it will be as follows:

		R	
		0	1
S	0	0	1
	1	1	0

Figure 2: k-map

So,the resultant expression of A is $A = R'S + RS'$.

5 Software

The Assembly code for the given circuit is

```
.include "/sdcard/codes/m328Pdef.inc"
ldi r16,0b11111100
out DDRD,r16
ldi r16,0b00000001
out DDRB,r16

loop:
 in r17,PIND

;Taking A
ldi r24,0b00000100
mov r18,r17
and r18,r24
ldi r25,0b00000010
loopa:
lsr r18
dec r25
brne   loopa
.DEF A =r18

;Taking B
ldi r24,0b00001000
mov r19,r17
and r19,r24
ldi r25,0b00000011
loopb:
lsr r19
dec r25
brne   loopb
.DEF B =r19

;Taking C
ldi r24,0b00010000
```

```
mov r20,r17
and r20,r24
ldi r25,0b00000100
loopc:
lsr r20
dec r25
brne    loopc
.DEF C =r20

;Taking D
ldi r24,0b00100000
mov r21,r17
and r21,r24
ldi r25,0b00000101
loopd:
lsr r21
dec r25
brne    loopd
.DEF D =r21

;Taking S0
ldi r24,0b01000000
mov r22,r17
and r22,r24
ldi r25,0b00000110
loope:
lsr r22
dec r25
brne    loope
.DEF S0 =r22

;Taking S1
ldi r24,0b10000000
mov r23,r17
and r23,r24
ldi r25,0b00000111
loopf:
lsr r23
```

```
dec r25
brne     loopf
.DEF S1 =r23

mov r24,r22
com r24
.DEF S_0 =r24
mov r25,r23
com r25
.DEF S_1 =r25

and A,S_0
and A,S_1
and B,S0
and B,S_1
and C,S_0
and C,S1
and D,S0
and D,S1
or A,B
or A,C
or A,D

out PORTB,A

rjmp loop

Start:
rjmp Start
```