Adam Boros

Practice Assignment 3

Part A+B

```
-------------------
x: 0
y: 0
q0*: 0
q1*: 0
q2*: 0

z1: 1
z2: 0
state: A
```

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11
	S*				

```
-------------------
x: 1
y: 0
q0*: 1
q1*: 0
q2*: 0

z1: 1
z2: 0
state: B
```

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11
	S*				

```
-------------------
x: 0
y: 0
q0*: 1
q1*: 0
q2*: 0

z1: 1
z2: 0
state: B
```

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11
	S*				

```
-------------------
x: 1
y: 1
q0*: 1
q1*: 1
q2*: 0

z1: 0
z2: 0
state: D
```

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11
	S*				

```
-------------------
x: 1
y: 1
q0*: 0
q1*: 1
q2*: 0

z1: 1
z2: 0
state: C
```

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11
	S*				

```
-------------------
x: 0
y: 0
q0*: 0
q1*: 1
q2*: 0

z1: 1
z2: 0
state: C
```

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11
	S*				

```
x: 0
y: 1
q0*: 0
q1*: 1
q2*: 1

z1: 1
z2: 1
state: G
```

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11
	S*				

```
-------------------
x: 0
y: 1
q0*: 1
q1*: 1
q2*: 1

z1: 1
z2: 1
state: H
```

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11
	S*				

```
-------------------
x: 0
y: 0
q0*: 1
q1*: 1
q2*: 0

z1: 0
z2: 0
state: D * Terminal w
```

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11
	S*				

src\main.c

```c
/*
 * HelloWorld.c
 *
 * Created: 11/9/2023 10:43:27 AM
 * Author : Alin
 */



#include <stdio.h>
#include <avr/io.h>
#include <util/delay.h>

#include "usart.h"
#include "i2cmaster.h"

unsigned char x, y, z1, z2; //creating the variables
unsigned char q0, q1, q2, q0_next, q1_next, q2_next;

void read_xy_values(void); //checking which button is pressed and overriding the global variables x and y depending on which button is pressed
void show_output(void); //printing the state to the serial monitor
void state_transition(void); //changing states depending in the input


int main(void) {

  uart_init(); // open the communication to the microcontroller
  io_redirect(); // redirect input and output to the communication
  i2c_init(); //initializing the communication protocoll

  DDRC = 0b00110000; //setting the registers as inputs for the buttons (5 and 4 are for i2c)
  PORTC = 0b00111111; //setting the pullup resistors high for the buttons so we can check if they're pressed

  DDRD = 0b11110000; //setting the registers as outputs for the LEDS
  PORTD = 0b00000000; // turning the leds off

  q0 = 0; //setting the inputs initially to zero
  q1 = 0;
  q2 = 0;



  while(1) {
        read_xy_values(); //calling the xy_read value function
    state_transition(); //calling the state_transition function

    q0 = q0_next; //setting the current state equal to the next state, so the next state would become the current
    q1 = q1_next;
```

```
49      q2 = q2_next;
50
51      show_output(); //calling the show_output function
52      _delay_ms(1000);
53
54    }
55
56    return 0;
57  }
58
59  void read_xy_values(void){ //this function overrides the global variables x and y
    depending on which button is pressed
60
61    if (!(PINC & (1 << PC6)) && !(PINC & (1 << 7))){
62      x=0;
63      y=0;
64    }
65
66    if (!(PINC & (1 << PC2))){
67      x=1;
68      y=0;
69    }
70
71
72
73    if (!(PINC & (1 << PC3))){
74      x=0;
75      y=1;
76    }
77
78    if (!(PINC & (1 << PC2)) && !(PINC & (1 << PC3))){
79      x=1;
80      y=1;
81    }
82
83
84  }
85
86  void show_output(void){ //this function prints to the serial monitor, it prints the value
    of x and y and the current state
87    char disp = ' ';
88    printf("\n--------------------");
89    printf("\nx: %d", x);
90    printf("\ny: %d", y);
91
92    if (q2==0 && q1==0 && q0 ==0){
93      disp = 'A';
94    }
95    if (q2==0 && q1==0 && q0 ==1){
96      disp = 'B';
97    }
98    if (q2==0 && q1==1 && q0 ==0){
99      disp = 'C';
100   }
```

```
101     if (q2==0 && q1==1 && q0 ==1){
102       disp = 'D';
103     }
104     if (q2==1 && q1==0 && q0 ==0){
105       disp = 'E';
106     }
107     if (q2==1 && q1==0 && q0 ==1){
108       disp = 'F';
109     }
110     if (q2==1 && q1==1 && q0 ==0){
111       disp = 'G';
112     }
113     if (q2==1 && q1==1 && q0 ==1){
114       disp = 'H';
115     }
116 
117     z2 = (q2&&q1)||(q2&&(!q0));
118     z1 = (!q0)||(!q1)||(q2);
119 
120     printf("\nq0*: %d", q0_next);
121     printf("\nq1*: %d", q1_next);
122     printf("\nq2*: %d", q2_next);
123     printf("\n\nz1: %d\nz2: %d", z1,z2);
124     printf("\nstate: %c", disp);
125 }
126 
127 void state_transition(void){ //this function switches between states, depending on the x
    and y inputs
128     PORTD=0b00100000;
129 
130     q0_next = ((!q1) && x) || (q0 && (!x)) || (q2); //transition equations
131     q1_next = ((!q2) && q0 && x) || (q1 && (!x)) || (q2 && q1);
132     q2_next = (!q0) && (q2 || ((!x) && y));
133 
134     _delay_ms(100);
135     PORTD=0b00000000; //flashes the led
136 }
```

Part C

```
--- More detai
--- Quit: Ctrl

x: 0
y: 0
A_state

x: 0
y: 0
A_state

x: 1
y: 0
B_state

x: 0
y: 0
B_state

x: 0
y: 1
B_state

x: 0
y: 0
B_state
```

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11

S	XY 00	01	10	11	Z1 Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11

```
x: 1
y: 1
D_state
```

	XY				
S	00	01	10	11	Z1Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11

```
x: 0
y: 0
D_state
```

	XY				
S	00	01	10	11	Z1Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11

```
x: 1
y: 0
C_state
```

	XY				
S	00	01	10	11	Z1Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11

```
x: 0
y: 0
C_state
```

	XY				
S	00	01	10	11	Z1Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11

```
x: 0
y: 1
G_state
```

	XY				
S	00	01	10	11	Z1Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11

```
x: 0
y: 0
H_state
 *  Terminal will
```

	XY				
S	00	01	10	11	Z1Z2
A	A	E	B	B	10
B	B	B	D	D	10
C	C	G	A	A	10
D	D	D	C	C	00
E	F	F	F	F	11
F	B	B	B	B	10
G	H	H	H	H	11
H	D	D	D	D	11

src\main.c

```c
/*
 * HelloWorld.c
 *
 * Created: 11/9/2023 10:43:27 AM
 * Author : Alin
 */



#include <stdio.h>
#include <avr/io.h>
#include <util/delay.h>

#include "usart.h"
#include "i2cmaster.h"

typedef enum { //using enumrated types to represent the ststes
  A_state,
  B_state,
  C_state,
  D_state,
  E_state,
  F_state,
  G_state,
  H_state
} state;

unsigned char x, y, z1, z2; // creatung the variables

//X AND Y ARE USED AS GLOBAL VARIABLES!!!!

void read_xy_values(void); //checking which button is pressed and overriding the global variables x and y depending on which button is pressed
state state_transition(state); //changing states depending in the input
void print_state(state); //printing the state to the serial monitor

int main(void) {

  uart_init(); // open the communication to the microcontroller
  io_redirect(); // redirect input and output to the communication
  i2c_init(); //declaring the communication protocoll

  state current_state, next_state; //creating the state types
  current_state = A_state; //we start with state A

  DDRC = 0b00110000; //setting the registers as inputs for the buttons (5 and 4 are for i2c)
  PORTC = 0b00111111; //setting the pullup resistors high for the buttons so we can check if they're pressed

  DDRD = 0b11110000; //setting the registers as outputs for the LEDS
  PORTD = 0b00000000; // turning the leds off
```

```
50
51
52    while(1) {
53      read_xy_values(); //calling the xy_read value function
54
55      next_state = state_transition(current_state); //declaring the next state with the
    function state transition which takes the current state as an input parameter
56      current_state = next_state; //setting the current state equal to the next state, so
    the next state would become the current
57
58      printf("\nx: %d\ny: %d\n", x,y); //printing the x and y values for checking
59      print_state(current_state); //printing the state
60
61      _delay_ms(1000);
62
63    }
64    return 0;
65  }
66
67
68  void read_xy_values(void){ //this function overrides the global variables x and y
    depending on which button is pressed
69      if (!(PINC & (1 << PC6)) && !(PINC & (1 << 7))){
70        x=0;
71        y=0;
72      }
73      if (!(PINC & (1 << PC2))){
74        x=1;
75        y=0;
76      }
77      if (!(PINC & (1 << PC3))){
78        x=0;
79        y=1;
80      }
81      if (!(PINC & (1 << PC2)) && !(PINC & (1 << PC3))){
82        x=1;
83        y=1;
84      }
85    }
86
87
88  void print_state(state state_to_print){ //this function prints which state is the program
    in
89    switch(state_to_print){
90      case A_state: printf("A_state \n"); break;
91      case B_state: printf("B_state \n"); break;
92      case C_state: printf("C_state \n"); break;
93      case D_state: printf("D_state \n"); break;
94      case E_state: printf("E_state \n"); break;
95      case F_state: printf("F_state \n"); break;
96      case G_state: printf("G_state \n"); break;
97      case H_state: printf("H_state \n"); break;
98    }
99  }
```

```
100
101
102 state state_transition(state current_state){ //this function changes states depening on
    which state the program is in and the given x and y inputs
103   PORTD=0b00100000; //flashing one of the leds
104
105   switch (current_state) {
106     case A_state:
107         if (x == 0 && y == 0)
108             return A_state;
109         if (x == 0 && y == 1)
110             return E_state;
111         if (x == 1 && y == 0)
112             return B_state;
113         if (x == 1 && y == 1)
114             return B_state;
115         break;
116
117     case B_state:
118         if (x == 0 && y == 0)
119             return B_state;
120         if (x == 0 && y == 1)
121             return B_state;
122         if (x == 1 && y == 0)
123             return D_state;
124         if (x == 1 && y == 1)
125             return D_state;
126         break;
127
128     case C_state:
129         if (x == 0 && y == 0)
130             return C_state;
131         if (x == 0 && y == 1)
132             return G_state;
133         if (x == 1 && y == 0)
134             return A_state;
135         if (x == 1 && y == 1)
136             return A_state;
137         break;
138
139     case D_state:
140         if (x == 0 && y == 0)
141             return D_state;
142         if (x == 0 && y == 1)
143             return D_state;
144         if (x == 1 && y == 0)
145             return C_state;
146         if (x == 1 && y == 1)
147             return C_state;
148         break;
149
150     case E_state:
151         return F_state;
152         break;
```

```
153
154     case F_state:
155         return B_state;
156         break;
157
158     case G_state:
159         return H_state;
160         break;
161
162     case H_state:
163         return D_state;
164         break;
165
166   }
167
168   _delay_ms(100);
169   PORTD=0b00000000;
170
171   return current_state;
172 }
```