

Apr 12, 13 20:59

adeja001\_lab2\_part1.c

Page 1/1

```

1  /*
2  * adeja001_lab2_part1.c - April 12, 2013
3  * Name: Ariana DeJaco E-mail:adeja001@ucr.edu
4  * CS Login: adeja001
5  * Partner Name: Joshua DeForest-Williams E-mail jdefo002@ucr.edu
6  * Lab Section: 022
7  * Assignment: Lab#2 Exercise#1
8  * Exercise Description: Count the number of 1s on ports A and B and output that
9  * number on port C
10 */
11
12 #include <avr/io.h>
13 #include <avr/sfr_defs.h>
14
15 // Bit-access function
16 unsigned char SetBit(unsigned char x, unsigned char k, unsigned char b) {
17     return (b ? x | (0x01 << k) : x & ~(0x01 << k));
18 }
19 unsigned char GetBit(unsigned char x, unsigned char k) {
20     return ((x & (0x01 << k)) != 0);
21 }
22
23 // Current Port Definitions
24 #define OUTPUT_DDR          DDRC
25 #define OUTPUT_INPORT       PINC
26 #define OUTPUT_OUTPORT     PORTC
27 #define COUNT_A_DDR        DDRA
28 #define COUNT_A_INPORT     PINA
29 #define COUNT_A_OUTPORT    PORTA
30 #define COUNT_B_DDR        DDRB
31 #define COUNT_B_INPORT     PINB
32 #define COUNT_B_OUTPORT    PORTB
33
34 // Additional macros not defines in sfr_defs.h
35 // #define SET_PORT_BIT(OUTPORT, BIT)    OUTPORT |= (1 << BIT)
36 // #define CLEAR_PORT_BIT(OUTPORT, BIT)  OUTPORT &= ~(1 << BIT)
37
38 // DDRA: Configures each of port A's physical pins to input (0) or output (1)
39 // PORTA: Writing to this register writes the port's physical pins (Write only)
40 // PINA: Reading this register reads the values of the port's physical pins (Read
41 // only)
42 int main(void)
43 {
44     OUTPUT_DDR = 0xFF; //Configures port C's 8 pins as outputs.
45     OUTPUT_OUTPORT = 0x00; //Initialize output on port C to 0x00;
46     COUNT_A_DDR = 0x00; // Configure Port A's 8 pins as inputs.
47     //COUNT_A_OUTPORT = 0xFF; // Configure Port A's 8 pins as inputs.
48     COUNT_B_DDR = 0x00; // Configure Port B's 8 pins as inputs.
49     //COUNT_B_OUTPORT = 0xFF; // Configure Port B's 8 pins as inputs.
50
51     char loop_counter;
52     char count;
53     while(1)
54     {
55         count = 0;
56         for (loop_counter=0; loop_counter<8; loop_counter++)
57         {
58             if (GetBit(COUNT_A_INPORT, loop_counter))
59             {
60                 count +=1 ;
61             }
62             if(GetBit(COUNT_B_INPORT, loop_counter))
63             {
64                 count +=1;
65             }
66         }
67         OUTPUT_OUTPORT = count;
68     }
69 }
70

```

Apr 12, 13 20:59

adeja001\_lab2\_part2.c

Page 1/2

```

1  /*
2  * adeja001_lab2_part2.c - April 8, 2013
3  * Name: Ariana DeJaco E-mail:adeja001@ucr.edu
4  * CS Login: adeja001
5  * Partner Name:Joshua DeForest-Williams E-mail jdefo002@ucr.edu
6  * Lab Section: 022
7  * Assignment: Lab#1 Exercise#2
8  * Exercise Description: A car has a fuel-level sensor that sets PA3..PA0 to a v
9  alue between 0 (empty) and 15 (full).
10 * A series of LEDs connected to PC5..PC0 should light to graphically indicate t
11 he fuel level. If the fuel level is 1 or 2,
12 * PC5 lights. If the level is 3 or 4, PC5 and PC4 light. Level 5-6 lights PC5..
13 PC3. 7-9 lights PC5..PC2. 10-12
14 * lights PC5..PC1. 13-15 lights PC5..PC0. Also, PC6 connects to a "Low fuel" ic
15 on, which should light if the level is 4 or less.
16 * (The example below shows the display for a fuel level of 3).
17 */
18
19 #include <avr/io.h>
20 #include <avr/sfr_defs.h>
21
22 // Bit-access function
23 unsigned char SetBit(unsigned char x, unsigned char k, unsigned char b) {
24     return (b ? x | (0x01 << k) : x & ~(0x01 << k));
25 }
26
27 unsigned char GetBit(unsigned char x, unsigned char k) {
28     return ((x & (0x01 << k)) != 0);
29 }
30
31 // Current Port Definitions
32 #define LED_DDR DDRC
33 #define LED_INPORT PINC
34 #define LED_OUTPORT PORTC
35 #define SENSOR_DDR DDRA
36 #define SENSOR_INPORT PINA
37 #define SENSOR_OUTPORT PORTA
38
39 // Additional macros not defines in sfr_defs.h
40 #define SET_PORT_BIT(OUTPORT, BIT) OUTPORT |= (1 << BIT)
41 #define CLEAR_PORT_BIT(OUTPORT, BIT) OUTPORT &= ~(1 << BIT)
42
43 //DDRA: Configures each of port A's physical pins to input (0) or output(1)
44 //PORTA: Writing to this register writes the port's physical pins
45 // (Write only)
46 //PINA: Reading this register reads the values of the port's physical pins
47 // (Read only)
48 int main(void)
49 {
50     SENSOR_DDR = 0x00;
51     SENSOR_OUTPORT = 0xFF; // Configure port A's 8 pins as inputs
52     LED_DDR = 0xFF;
53     LED_OUTPORT = 0x00; // Configure port C's 8 pins as outputs,
54     const unsigned char Low_level = 0;
55     const unsigned char Sec_low_level = 2;
56     const unsigned char Mid_low_level = 4;
57     const unsigned char Mid_Hi_level = 6;
58     const unsigned char Sec_Hi_level = 9;
59     const unsigned char High_level = 12;
60
61     while(1)
62     {
63         char led = 0;
64         if (SENSOR_INPORT > High_level)
65         {
66             SET_PORT_BIT (led,5);
67         }
68         if (SENSOR_INPORT > Sec_Hi_level)
69         {
70             SET_PORT_BIT (led,4);
71         }
72         if (SENSOR_INPORT > Mid_Hi_level)
73         {
74

```

Apr 12, 13 20:59

adeja001\_lab2\_part2.c

Page 2/2

```

70         SET_PORT_BIT(led,3);
71     }
72     if (SENSOR_INPORT > Mid_low_level)
73     {
74         SET_PORT_BIT(led,2);
75     }
76     if (SENSOR_INPORT > Sec_low_level)
77     {
78         SET_PORT_BIT(led,1);
79     }
80     if (SENSOR_INPORT > Low_level )
81     {
82         SET_PORT_BIT(led,0);
83     }
84     if (SENSOR_INPORT <= Mid_low_level)
85     {
86         SET_PORT_BIT(led,6);
87     }
88     LED_OUTPORT = led;
89 }
90
91
92
93
94
95
96

```

Apr 12, 13 21:01

adeja001\_lab2\_part3.c

Page 1/2

```

1  /*
2  * adeja001_lab2_part3.c - April 8, 2013
3  * Name: Ariana DeJaco E-mail:adeja001@ucr.edu
4  * CS Login: adeja002
5  * Partner Name: Joshua DeForest-Williams E-mail jdefo002@ucr.edu
6  * Lab Section: 022
7  * Assignment: Lab#1 Exercise#3
8  * Exercise Description: A car has a fuel-level sensor that sets PA3..PA0
9  * to a value between 0 (empty) and 15 (full). A series of LEDs connected
10 * to PC5..PC0 should light to graphically indicate the fuel level.
11 * If the fuel level is 1 or 2, PC5 lights. If the level is 3 or 4,
12 * PC5 and PC4 light. Level 5-6 lights PC5..PC3. 7-9 lights PC5..PC2.
13 * 10-12 lights PC5..PC1. 13-15 lights PC5..PC0. Also, PC6 connects to a
14 * "Low fuel" icon, which should light if the level is 4 or less.
15 * (The example below shows the display for a fuel level of 3).
16 * In addition to the above, PA4 is 1 if a key is in the ignition,
17 * PA5 is 1 if a driver is seated, and PA6 is 1 if the driver's seatbelt
18 * is fastened. PC7 should light a "Fasten seatbelt" icon if a key is in
19 * the ignition, the driver is seated, but the belt is not fastened.
20 */
21
22
23 #include <avr/io.h>
24 #include <avr/sfr_defs.h>
25
26 // Bit-access function
27 unsigned char SetBit(unsigned char x, unsigned char k, unsigned char b) {
28     return (b ? x | (0x01 << k) : x & ~(0x01 << k));
29 }
30 unsigned char GetBit(unsigned char x, unsigned char k) {
31     return ((x & (0x01 << k)) != 0);
32 }
33
34 // Current Port Definitions
35 #define LED_DDR          DDRC
36 #define LED_INPORT       PINC
37 #define LED_OUTPORT      PORTC
38 #define SENSOR_DDR      DDRA
39 #define SENSOR_INPORT    PINA
40 #define SENSOR_OUTPORT   PORTA
41
42 // Additional macros not defines in sfr_defs.h
43 #define SET_PORT_BIT(OUTPORT, BIT)    OUTPORT |= (1 << BIT)
44 #define CLEAR_PORT_BIT(OUTPORT, BIT)  OUTPORT &= ~(1 << BIT)
45
46 //DDRA: Configures each of port A's physical pins to input (0) or output(1)
47 //PORTA: Writing to this register writes the port's physical pins
48 // (Write only)
49 //PINA: Reading this register reads the values of the port's physical pins
50 // (Read only)
51 int main(void)
52 {
53     SENSOR_DDR = 0x00;
54     SENSOR_OUTPORT = 0xFF; // Configure port A's 8 pins as inputs
55     LED_DDR = 0xFF;
56     LED_OUTPORT = 0x00; // Configure port C's 8 pins as outputs,
57     const unsigned char Low_level = 0;
58     const unsigned char Sec_low_level = 2;
59     const unsigned char Mid_low_level = 4;
60     const unsigned char Mid_Hi_level = 6;
61     const unsigned char Sec_Hi_level = 9;
62     const unsigned char High_level = 12;
63
64     while(1)
65     {
66         char SENSOR = SENSOR_INPORT & 0x0F;
67         char ignition = (SENSOR_INPORT >> 4) & 0x01;
68         char seated = (SENSOR_INPORT >> 5) & 0x01;
69         char seatbelt = (SENSOR_INPORT >> 6) & 0x01;
70         char led = 0;
71         if (SENSOR > High_level)
72         {
73             SET_PORT_BIT (led,0);

```

Apr 12, 13 21:01

adeja001\_lab2\_part3.c

Page 2/2

```

74     }
75     if (SENSOR > Sec_Hi_level)
76     {
77         SET_PORT_BIT (led,1);
78     }
79     if (SENSOR > Mid_Hi_level)
80     {
81         SET_PORT_BIT(led,2);
82     }
83     if (SENSOR > Mid_low_level)
84     {
85         SET_PORT_BIT(led,3);
86     }
87     if (SENSOR > Sec_low_level)
88     {
89         SET_PORT_BIT(led,4);
90     }
91     if (SENSOR > Low_level )
92     {
93         SET_PORT_BIT(led,5);
94     }
95     if (SENSOR <= Mid_low_level)
96     {
97         SET_PORT_BIT(led,6);
98     }
99     if (ignition && seated && !seatbelt)
100    {
101        SET_PORT_BIT(led, 7);
102    }
103    LED_OUTPORT = led;
104 }
105 }
106
107
108
109
110

```

Apr 12, 13 21:01

adeja001\_lab2\_part4.c

Page 1/1

```

1  /*
2  * adeja001_lab2_part4.c - April 8, 2013
3  * Name: Ariana DeJaco E-mail:adeja001@ucr.edu
4  * CS Login: adeja002
5  * Partner Name: Joshua DeForest-Williams E-mail jdefo002@ucr.edu
6  * Lab Section: 022
7  * Assignment: Lab#1 Exercise#4
8  * Exercise Description: (Challenge): Read an 8-bit value on PA7..PA0
9  * and write that value on PB3..PB0PC7..PC4. That is to say,
10 * take the upper nibble of PINA and map it to the lower nibble of PORTB,
11 * likewise take the lower nibble of PINA and map it to the upper
12 * nibble of PORTC (PA7 -> PB3, PA6 -> PB2, M-^E PA1 -> PC5, PA0 -> PC4).
13 */
14
15 #include <avr/io.h>
16 #include <avr/sfr_defs.h>
17
18 // Bit-access function
19 unsigned char SetBit(unsigned char x, unsigned char k, unsigned char b) {
20     return (b ? x | (0x01 << k) : x & ~(0x01 << k));
21 }
22 unsigned char GetBit(unsigned char x, unsigned char k) {
23     return ((x & (0x01 << k)) != 0);
24 }
25
26 // Current Port Definitions
27 #define HIGH_NIB_DDR          DDRC
28 #define HIGH_NIB_INPORT      PINC
29 #define HIGH_NIB_OUTPORT     PORTC
30 #define FULL_NIB_DDR         DDRA
31 #define FULL_NIB_INPORT      PINA
32 #define FULL_NIB_OUTPORT     PORTA
33 #define LOW_NIB_DDR          DDRB
34 #define LOW_NIB_INPORT       PINB
35 #define LOW_NIB_OUTPORT      PORTB
36
37 // Additional macros not defines in sfr_defs.h
38 //#define SET_PORT_BIT(OUTPORT, BIT)    OUTPORT |= (1 << BIT)
39 //#define CLEAR_PORT_BIT(OUTPORT, BIT)  OUTPORT &= ~(1 << BIT)
40
41 //DDRA: Configures each of port A's physical pins to input (0) or output (1)
42 //PORTA: Writing to this register writes the port's physical pins (Write only)
43 //PINA: Reading this register reads the values of the port's physical pins (Read
44 only)
45 int main(void)
46 {
47     HIGH_NIB_DDR = 0xFF; //Configures port C's 8 pins as outputs.
48     HIGH_NIB_OUTPORT = 0x00; //Initialize output on port C to 0x00;
49     FULL_NIB_DDR = 0x00; // Configure Port A's 8 pins as inputs.
50     LOW_NIB_OUTPORT = 0xFF; // Configure Port A's 8 pins as inputs.
51     LOW_NIB_DDR = 0x00; // Configure Port B's 8 pins as inputs.
52     LOW_NIB_OUTPORT = 0xFF; // Configure Port B's 8 pins as inputs.
53     while(1)
54     {
55         //PORTC (PA7 -> PB3, PA6 -> PB2, M-^E PA1 -> PC5, PA0 -> PC4).
56         char temp_low_nib = (FULL_NIB_INPORT >> 4);
57         LOW_NIB_OUTPORT = temp_low_nib;
58         char temp_high_nib = (FULL_NIB_INPORT & 0x0f) << 4;
59         HIGH_NIB_OUTPORT = temp_high_nib;
60     }

```

Apr 12, 13 21:02

adeja001\_lab2\_part5.c

Page 1/2

```

1  /*
2  * adeja001_lab2_part5.c - April 8, 2013
3  * Name: Ariana DeJaco E-mail:adeja001@ucr.edu
4  * CS Login: adeja001
5  * Partner Name: Joshua DeForest-Williams E-mail jdefo002@ucr.edu
6  * Lab Section: 022
7  * Assignment: Lab#1 Exercise#5
8  * Exercise Description: (Challenge): A car's passenger-seat weight
9  * sensor outputs a 9-bit value (ranging from 0 to 511) and connects
10 * to input PD7..PD0PB0 on the microcontroller. If the weight is equal
11 * to or above 70 pounds, the airbag should be enabled by setting PB1 to
12 * 1. If the weight is above 5 but below 70, the airbag should be
13 * disabled and an "Airbag disabled" icon should light by setting PB2 to
14 * 1. (Neither B0 nor B1 should be set if the weight is 5 or less,
15 * as there is no passenger).
16 */
17
18 #include <avr/io.h>
19 #include <avr/sfr_defs.h>
20
21 // Bit-access function
22 unsigned char SetBit(unsigned char x, unsigned char k, unsigned char b) {
23     return (b ? x | (0x01 << k) : x & ~(0x01 << k));
24 }
25 unsigned char GetBit(unsigned char x, unsigned char k) {
26     return ((x & (0x01 << k)) != 0);
27 }
28
29 // Current Port Definitions
30 #define WEIGHT_DDR          DDRD
31 #define WEIGHT_INPORT       PIND
32 #define WEIGHT_OUTPORT      PORTD
33 #define AIRBAG_DDR          DDRB
34 #define AIRBAG_INPORT       PINB
35 #define AIRBAG_OUTPORT      PORTB
36
37 // Additional macros not defines in sfr_defs.h
38 #define SET_PORT_BIT(OUTPORT, BIT)      OUTPORT |= (1 << BIT)
39 #define CLEAR_PORT_BIT(OUTPORT, BIT)    OUTPORT &= ~(1 << BIT)
40
41 //DDRA: Configures each of port A's physical pins to input (0) or output (1)
42 //PORTA: Writing to this register writes the port's physical pins (Write only)
43 //PINA: Reading this register reads the values of the port's physical pins (Read
44 only)
45 int main(void)
46 {
47     long weight;
48     char airbag = 0;
49     WEIGHT_DDR = 0x00;
50     WEIGHT_OUTPORT = 0xFF;
51     AIRBAG_DDR = 0xFE;
52     AIRBAG_OUTPORT = 0x01;
53     while(1)
54     {
55         airbag = 0;
56         weight = (WEIGHT_INPORT << 1);
57         weight = weight | (AIRBAG_INPORT & 0x01);
58         if (weight >= 70)
59         {
60             SET_PORT_BIT(airbag, 1);
61         }
62         else if (weight < 70 && weight > 5)
63         {
64             CLEAR_PORT_BIT(airbag, 1);
65             SET_PORT_BIT(airbag, 2);
66         }
67         else if (weight < 5)
68         {
69             CLEAR_PORT_BIT(airbag, 0);
70             CLEAR_PORT_BIT(airbag, 1);
71         }
72         AIRBAG_OUTPORT = airbag;
73     }
74 }

```

Apr 12, 13 21:02

adeja001\_lab2\_part5.c

Page 2/2

```

73     }
74 }

```