LABORATORY 02: Bitwise Logical Operations in C

CHRISTOPHER NIELSEN + CHRISTOPHER SHAMAH
KENNETH SHORT
Bench No: 17
ID#114211318
ID#112229076
Lab Section L01

Questions

1. You want to verify the implementation dependent effect(s) of the C programming language's >> operator on a signed value. Write a very short program to do this.

The following code will prove and display to the user what exactly the implantation dependant solution will do:

```
#include <stdio.h>

int main() {
   int signedValue = -10;

   // Right shift the signed value
   signedValue = signedValue >> 1;

   // Print the result
   printf("After right shift: %d\n", signedValue);

   return 0;
}
```

2. What is the difference between the & operator and the && operator in C?

The & operator will perform bitwise level comparisons on the logic levels of each bit of a sequence of data, while the && operator will perform a logic based comparison on the WHOLE sequence and return either a 1 or a 0 as the result based on either "all false" or "any bit true".

3. What is the difference between declaring a variable with uint16_t and unsigned int in a C program. How would you find the width of an unsigned int in our C compiler version?

A uint16_t is an unsigned 16-bit integer, and always will be the same size, compiler/implementation independent. However the unsigned int is “unsigned integer”, which has a size that is implementation dependent

In our compiler version we would use the libraries associated with our target microcontroller and C standard which would in its own appropriate header files contain the widths distinctly stated.

4. Compare the assembler code generated by the compiler using an optimization level of None (-O0) and of -Og for the first program you wrote for Task 1. Does one program result in more object code than the other?

The O0 results in FAR MORE object code being generated than the other in Og. nearly double the amount of code is created. The code for the same Task 1 program is shown below O0, followed by Og.

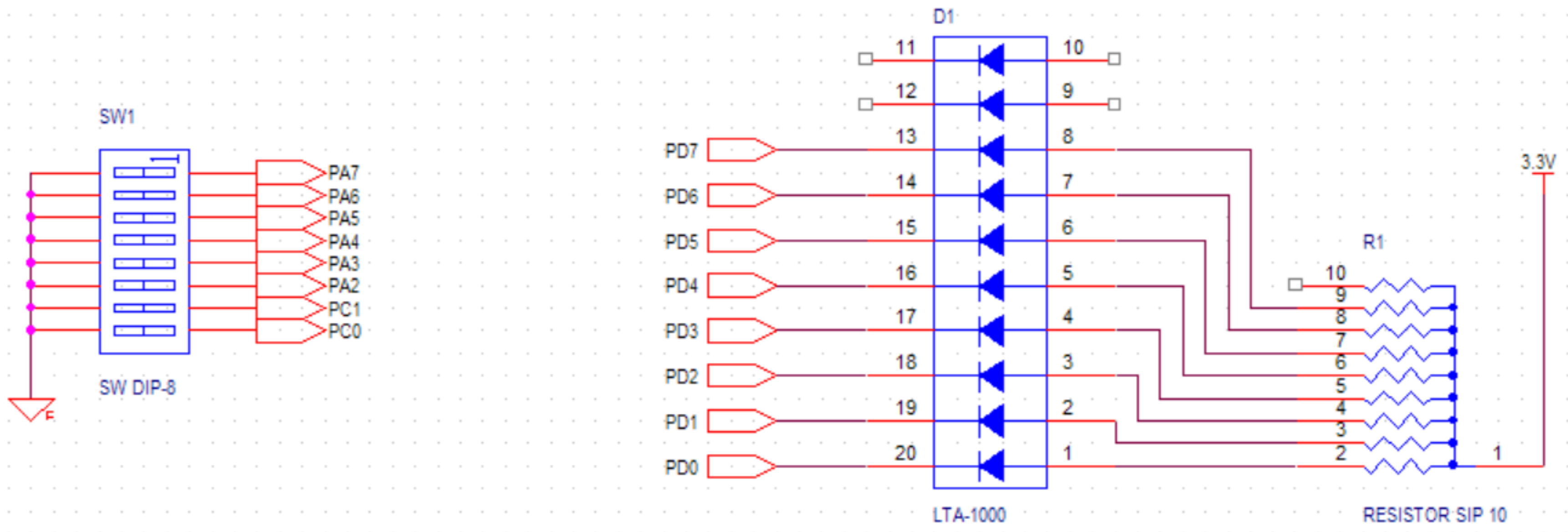
```
-O0
—-------------------------------------------------------------------------------------------

int main(void)
{
 10c:	cf 93    	push	r28
 10e:	df 93    	push	r29
 110:	cd b7    	in	r28, 0x3d	; 61
 112:	de b7    	in	r29, 0x3e	; 62

	//INPUT Pin Configuration for Dip Switches
	//flat
	PORTA_PIN7CTRL = 0x08;  //enable pullup resistor for PA7
```

```
114:	87 e1	ldi	r24, 0x17	; 23
116:	94 e0	ldi	r25, 0x04	; 4
118:	28 e0	ldi	r18, 0x08	; 8
11a:	fc 01	movw	r30, r24
11c:	20 83	st	Z, r18
		PORTA_PIN6CTRL = 0x08;  //enable pullup resistor for PA6
11e:	86 e1	ldi	r24, 0x16	; 22
120:	94 e0	ldi	r25, 0x04	; 4
122:	28 e0	ldi	r18, 0x08	; 8
124:	fc 01	movw	r30, r24
126:	20 83	st	Z, r18
		PORTA_PIN5CTRL = 0x08;  //enable pullup resistor for PA5
128:	85 e1	ldi	r24, 0x15	; 21
12a:	94 e0	ldi	r25, 0x04	; 4
12c:	28 e0	ldi	r18, 0x08	; 8
12e:	fc 01	movw	r30, r24
130:	20 83	st	Z, r18
		PORTA_PIN4CTRL = 0x08;  //enable pullup resistor for PA4
132:	84 e1	ldi	r24, 0x14	; 20
134:	94 e0	ldi	r25, 0x04	; 4
136:	28 e0	ldi	r18, 0x08	; 8
138:	fc 01	movw	r30, r24
13a:	20 83	st	Z, r18
		PORTA_PIN3CTRL = 0x08;  //enable pullup resistor for PA3
13c:	83 e1	ldi	r24, 0x13	; 19
13e:	94 e0	ldi	r25, 0x04	; 4
140:	28 e0	ldi	r18, 0x08	; 8
142:	fc 01	movw	r30, r24
144:	20 83	st	Z, r18
		PORTA_PIN2CTRL = 0x08;  //enable pullup resistor for PA2
146:	82 e1	ldi	r24, 0x12	; 18
148:	94 e0	ldi	r25, 0x04	; 4
14a:	28 e0	ldi	r18, 0x08	; 8
14c:	fc 01	movw	r30, r24
14e:	20 83	st	Z, r18
		//PORTA_PIN1CTRL = 0x08;  //enable pullup resistor for PA1 // I DONT WANT THIS TO FLOAT AND CAUSE BAD BEHAVIOR
		//PORTA_PIN0CTRL = 0x08;  //enable pullup resistor for PA0 // I DONT WANT THIS TO FLOAT AND CAUSE BAD BEHAVIOR

		PORTC_PIN1CTRL = 0x08;  //enable pullup resistor for PC1
150:	81 e5	ldi	r24, 0x51	; 81
152:	94 e0	ldi	r25, 0x04	; 4
154:	28 e0	ldi	r18, 0x08	; 8
156:	fc 01	movw	r30, r24
158:	20 83	st	Z, r18
		PORTC_PIN0CTRL = 0x08;  //enable pullup resistor for PC0
15a:	80 e5	ldi	r24, 0x50	; 80
15c:	94 e0	ldi	r25, 0x04	; 4
15e:	28 e0	ldi	r18, 0x08	; 8
160:	fc 01	movw	r30, r24
162:	20 83	st	Z, r18
		//OUTPUT Port config
		//flat
		PORTD_DIR = 0xFF;      //set Port D as an output for led bar graph
164:	80 e6	ldi	r24, 0x60	; 96
166:	94 e0	ldi	r25, 0x04	; 4
168:	2f ef	ldi	r18, 0xFF	; 255
16a:	fc 01	movw	r30, r24
16c:	20 83	st	Z, r18
  while (1)
  {
                    //flat
                    PORTD_OUT = ((VPORTA_IN & 0b11111100) | (VPORTC_IN & 0b00000011));
 16e:     84 e6     ldi       r24, 0x64   ; 100
 170:     94 e0     ldi       r25, 0x04   ; 4
 172:     22 e0     ldi       r18, 0x02   ; 2
 174:     30 e0     ldi       r19, 0x00   ; 0
 176:     f9 01     movw      r30, r18
 178:     20 81     ld        r18, Z
 17a:     42 2f     mov       r20, r18
 17c:     4c 7f     andi      r20, 0xFC   ; 252
 17e:     2a e0     ldi       r18, 0x0A   ; 10
 180:     30 e0     ldi       r19, 0x00   ; 0
 182:     f9 01     movw      r30, r18
 184:     20 81     ld        r18, Z
 186:     23 70     andi      r18, 0x03   ; 3
 188:     24 2b     or        r18, r20
 18a:     fc 01     movw      r30, r24
 18c:     20 83     st        Z, r18
  }
—-------------------------------------------------------------------------------------------


-Og


0000010c <main>:

{



                    //INPUT Pin Configuration for Dip Switches

                    //flat

                    PORTA_PIN7CTRL = 0x08;  //enable pullup resistor for PA7

 10c:     88 e0     ldi       r24, 0x08   ; 8

 10e:     80 93 17 04         sts       0x0417, r24         ; 0x800417 <__TEXT_REGION_LENGTH__+0x7e0417>

                    PORTA_PIN6CTRL = 0x08;  //enable pullup resistor for PA6

 112:     80 93 16 04         sts       0x0416, r24         ; 0x800416 <__TEXT_REGION_LENGTH__+0x7e0416>

                    PORTA_PIN5CTRL = 0x08;  //enable pullup resistor for PA5

 116:     80 93 15 04         sts       0x0415, r24         ; 0x800415 <__TEXT_REGION_LENGTH__+0x7e0415>

                    PORTA_PIN4CTRL = 0x08;  //enable pullup resistor for PA4

 11a:     80 93 14 04         sts       0x0414, r24         ; 0x800414 <__TEXT_REGION_LENGTH__+0x7e0414>

                    PORTA_PIN3CTRL = 0x08;  //enable pullup resistor for PA3

 11e:     80 93 13 04         sts       0x0413, r24         ; 0x800413 <__TEXT_REGION_LENGTH__+0x7e0413>

                    PORTA_PIN2CTRL = 0x08;  //enable pullup resistor for PA2
```

```
122:    80 93 12 04         sts     0x0412, r24         ; 0x800412 <__TEXT_REGION_LENGTH__+0x7e0412>
                //PORTA_PIN1CTRL = 0x08;  //enable pullup resistor for PA1 // I DONT WANT THIS TO FLOAT AND CAUSE BAD BEHAVIOR
                //PORTA_PIN0CTRL = 0x08;  //enable pullup resistor for PA0 // I DONT WANT THIS TO FLOAT AND CAUSE BAD BEHAVIOR


                PORTC_PIN1CTRL = 0x08;  //enable pullup resistor for PC1
126:    80 93 51 04         sts     0x0451, r24         ; 0x800451 <__TEXT_REGION_LENGTH__+0x7e0451>
                PORTC_PIN0CTRL = 0x08;  //enable pullup resistor for PC0
12a:    80 93 50 04         sts     0x0450, r24         ; 0x800450 <__TEXT_REGION_LENGTH__+0x7e0450>
                //OUTPUT Port config
                //flat
                PORTD_DIR = 0xFF;     //set Port D as an output for led bar graph
12e:    8f ef   ldi     r24, 0xFF   ; 255
130:    80 93 60 04         sts     0x0460, r24         ; 0x800460 <__TEXT_REGION_LENGTH__+0x7e0460>
  while (1)
  {
                //flat
                PORTD_OUT = ((VPORTA_IN & 0b11111100) | (VPORTC_IN & 0b00000011));
134:    92 b1   in      r25, 0x02   ; 2
136:    8a b1   in      r24, 0x0a   ; 10
138:    9c 7f   andi    r25, 0xFC   ; 252
13a:    83 70   andi    r24, 0x03   ; 3
13c:    89 2b   or      r24, r25
13e:    80 93 64 04         sts     0x0464, r24         ; 0x800464 <__TEXT_REGION_LENGTH__+0x7e0464>
142:    f8 cf   rjmp    .-16        ; 0x134 <main+0x28>
—-------------------------------------------------------------------------------------------
```

5. What is the common name for the function implemented in Task 3?

Its is a multiplexer with A as the Select input, B as the select low, and C as the select High.



Title	Lab 2 - T1-4 Hardware	
Size A	Document Number <Doc>	Rev .1
Date: Sunday, February 04, 2024	Sheet 1 of 1	

```c
/*
 * read_modify_write_sftw_sw0.c
 *
 * Created: 2/3/2024 7:00:29 PM
 * Author : MysticOwl
 */

#include <avr/io.h>

int main(void)
{

    //INPUT Pin Configuration for Dip Switches

    //flat
    PORTA_PIN7CTRL = 0x08;  //enable pullup resistor for PA7
    PORTA_PIN6CTRL = 0x08;  //enable pullup resistor for PA6
    PORTA_PIN5CTRL = 0x08;  //enable pullup resistor for PA5
    PORTA_PIN4CTRL = 0x08;  //enable pullup resistor for PA4
    PORTA_PIN3CTRL = 0x08;  //enable pullup resistor for PA3
    PORTA_PIN2CTRL = 0x08;  //enable pullup resistor for PA2
    //PORTA_PIN1CTRL = 0x08;  //enable pullup resistor for PA1 // I DONT WANT THIS ⮒
       TO FLOAT AND CAUSE BAD BEHAVIOR
    //PORTA_PIN0CTRL = 0x08;  //enable pullup resistor for PA0 // I DONT WANT THIS ⮒
       TO FLOAT AND CAUSE BAD BEHAVIOR

    PORTC_PIN1CTRL = 0x08;  //enable pullup resistor for PC1
    PORTC_PIN0CTRL = 0x08;  //enable pullup resistor for PC0

    //OUTPUT Port config

    //flat
    PORTD_DIR = 0xFF;      //set Port D as an output for led bar graph

    //pushbutton code
    VPORTB_DIR = 0b11111011;      //set everything but B2 as an output so they    ⮒
      stay zero.
    PORTB_PIN2CTRL = 0x08;  //enable pullup resistor for SW0 for pushbutton

        //flat
    PORTD_OUT = ((VPORTA_IN & 0b11111100) | (VPORTC_IN & 0b00000011));


    while (1)
    {

        VPORTB_OUT = VPORTB_IN << 1;

        if  ( (VPORTB_IN & 0b00000100) == 0b00000000) { // if the whole port is   ⮒
```

```
          ZERO, then the pushbutton is pressed which means OUTPUT.


            VPORTD_OUT = (VPORTD_OUT&0b11000111)|((((VPORTA_IN & 0b00000100) |
              (VPORTC_IN & 0b00000011)) << 3) & (0b00111000));
        }

        else {

        }



    }

}
```

```c
/*
 * simple_comb_functon_nb.c
 *
 * Created: 2/3/2024 7:05:04 PM
 * Author : MysticOwl
 */

#include <avr/io.h>

typedef union{

    uint8_t byte;

    struct {

        uint8_t bit0 : 1;
        uint8_t bit1 : 1;
        uint8_t bit2 : 1;
        uint8_t bit3 : 1;
        uint8_t bit4 : 1;
        uint8_t bit5 : 1;
        uint8_t bit6 : 1;
        uint8_t bit7 : 1;

    }bvals;
} Named_bits;

int main(void)
{

    volatile Named_bits data;
//  volatile uint8_t
//  volatile uint8_t

    //flat
    PORTA_PIN7CTRL = 0x08;  //enable pullup resistor for PA7
    PORTA_PIN6CTRL = 0x08;  //enable pullup resistor for PA6
    PORTA_PIN5CTRL = 0x08;  //enable pullup resistor for PA5
    PORTA_PIN4CTRL = 0x08;  //enable pullup resistor for PA4
    PORTA_PIN3CTRL = 0x08;  //enable pullup resistor for PA3
    PORTA_PIN2CTRL = 0x08;  //enable pullup resistor for PA2
    PORTA_PIN1CTRL = 0x08;  //enable pullup resistor for PA1 // I DONT WANT THIS 
      TO FLOAT AND CAUSE BAD BEHAVIOR
    PORTA_PIN0CTRL = 0x08;  //enable pullup resistor for PA0 // I DONT WANT THIS 
      TO FLOAT AND CAUSE BAD BEHAVIOR

    //OUTPUT Port config

    //flat
    PORTD_DIR = 0xFF;       //set Port D as an output for led bar graph



    while (1)
    {

        data.byte = VPORTA_IN;
        data.bvals.bit7 = ( (!data.bvals.bit7&data.bvals.bit6&!data.bvals.bit5) |
          (!data.bvals.bit7&data.bvals.bit6&data.bvals.bit5) | (data.bvals.bit7&!
          data.bvals.bit6&data.bvals.bit5) |
          (data.bvals.bit7&data.bvals.bit6&data.bvals.bit5));
        data.byte &= 0b10000000;
        VPORTD_OUT = ~data.byte;


    }
}
```