
REQUIREMENTS NOT MET

N/A. All requirements are met in this lab.

PROBLEMS ENCOUNTERED

For the homework 1, I did not encounter a problem when installing Atmel Studio but some issues did occur in the “Creating, Simulating, and Emulating in Atmel Studio” guide. One of the problems encountered, was in compiling the assembly code given where errors occurred. This was solved with me manually typing out the code rather than copying and pasting it, causing bits to randomly disappear or get corrupted. There were also issues in finding the buttons and shortcuts mentation in the guide as my toolbar was not initially installed so I had to manually bring up the toolbar and each of its components. This was not difficult but took some time to do. The final problem was that the processor status was not showing which through playing around in Atmel Studio, it was discovered that it appears when debugging is active and vanishes when its not.

HOMEWORK EXERCISES

1.1. Download and install Atmel Studio by following the aforementioned guide.

Seen being used in the screenshots below.

2.1. Obtain a screen shot from your computer of the results of step 12 (the simulation), also including your name in big letters on the same screen.

See appendix screenshot 2.1.

2.2. Obtain a screen shot from your computer of the results of step 14 (the emulation), again including your name in big letters on the same screen.

See appendix screenshot 2.2.

PSEUDOCODE/FLOWCHARTS

SECTION X (1, 2, etc.)

N/A. Code is given and written by Dr. Schwartz.

PROGRAM CODE

SECTION X (1, 2, etc.)

Code is given and written by Dr. Schwartz.

Section 2:

```
/*  
 * GPIO_Output.asm  
 *  
 * Modified: 22 May 19  
 * Author: Dr. Schwartz
```

```
  
This program shows how to initialize a GPIO port on the Atmel  
(Port D for this example) and demonstrates various ways to write to  
a GPIO port. The output will blink LEDs at the bottom left of the  
uPAD, labeled D4. PortD4, PortD5, and PortD6 are the red, green,  
and blue LEDs, respectively. Note that these LEDs are active-low.  
****/
```

```
;Definitions for all the registers in the processor. ALWAYS REQUIRED.  
;View the contents of this file in the Processor "Solution Explorer"  
; window under "Dependencies"  
.include "ATxmega128A1Udef.inc"
```

```
.equ BIT4 = 0b00010000  
.equ INV4 = 0b11101111  
.equ RED = INV4  
.equ BIT5 = 0b00100000  
.equ INV5 = ~BIT5  
.equ GREEN = ~(BIT5)  
.equ BIT6 = 0x40  
.equ BLUE = ~(BIT6)  
.equ BIT456 = 0x70  
.equ WHITE = ~(BIT456)  
.equ BIT64 = 0x50  
.equ PINK = ~(BIT64)  
.equ BLACK = 0xFF
```

```
.ORG 0x0000 ;Code starts running from address 0x0000.  
rjmp MAIN ;Relative jump to start of program.
```

```
.ORG 0x0100 ;Start program at 0x0100 so we don't overwrite  
; vectors that are at 0x0000-0x00FD
```

MAIN:

```
; initialize the data direction of the three LEDs as outputs (PD4-6)  
ldi R16, BIT456  
sts PORTD_DIRSET, R16
```

```
; Notice that the 3 LEDs (RED, GREEN, and BLUE) are all now on, creating white
```

```
; The following code shows different ways to write to the GPIO pins.
```

```
; Turn on each of the primary colored LEDs in turn, then use some combinations  
; These instructions sends the value in R16 to the PORTD pins.  
; Since the LEDs are wired as active-low, an R16 = RED = 0xFE = 0b1111 1110  
; will turn the RED LED on.
```

```
    ldi    R16, RED
    sts    PORTD_OUT, R16                ;send the value in R16 to the PORTD pins

    ldi    R16, GREEN
    sts    PORTD_OUT, R16
    ldi    R16, BLUE
    sts    PORTD_OUT, R16
    ldi    R16, WHITE
    sts    PORTD_OUT, R16
    ldi    R16, PINK
    sts    PORTD_OUT, R16

    ldi    R16, BLACK
    sts    PORTD_OUT, R16

    ldi    R16, BLUE
    sts    PORTD_OUT, R16

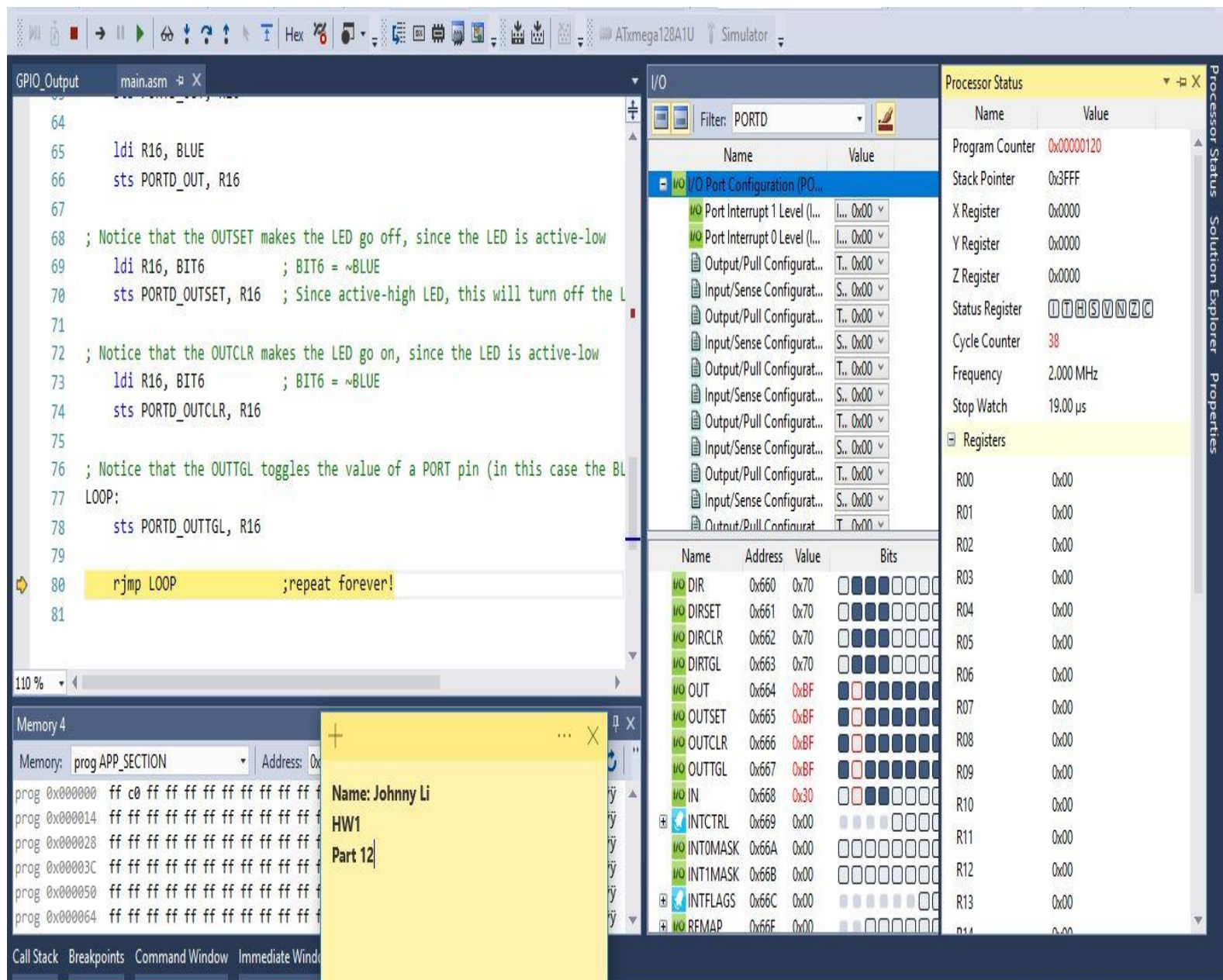
; Notice that the OUTSET makes the LED go off, since the LED is active-low
    ldi    R16, BIT6                    ; BIT6 = ~BLUE
    sts    PORTD_OUTSET, R16           ; Since active-high LED, this will turn off the LED

; Notice that the OUTCLR makes the LED go on, since the LED is active-low
    ldi    R16, BIT6                    ; BIT6 = ~BLUE
    sts    PORTD_OUTCLR, R16

; Notice that the OUTTGL toggles the value of a PORT pin (in this case the BLUE LED)
LOOP:
    sts    PORTD_OUTTGL, R16

    rjmp   LOOP                        ;repeat forever!
```

APPENDIX



2.1: Screenshot of part 12.

The screenshot displays the Atmel Studio IDE interface for an ATmega128A1U microcontroller. The main window shows assembly code for a GPIO output program. The code includes comments about LED states and active-low signals. The I/O window shows the configuration of various I/O registers, including PORTD and its associated control registers. The Processor Status window shows the current state of the processor, including the Program Counter, Stack Pointer, and various registers. A yellow sticky note is placed over the memory window, displaying the name 'Johnny Li' and 'HW1'.

Assembly Code (main.asm):

```

64
65 ldi R16, BLUE
66 sts PORTD_OUT, R16
67
68 ; Notice that the OUTSET makes the LED go off, since the LED is active-low
69 ldi R16, BIT6 ; BIT6 = ~BLUE
70 sts PORTD_OUTSET, R16 ; Since active-high LED, this will turn off the L
71
72 ; Notice that the OUTCLR makes the LED go on, since the LED is active-low
73 ldi R16, BIT6 ; BIT6 = ~BLUE
74 sts PORTD_OUTCLR, R16
75
76 ; Notice that the OUTTGL toggles the value of a PORT pin (in this case the BL
77 LOOP:
78 sts PORTD_OUTTGL, R16
79
80 rjmp LOOP ;repeat forever!
81

```

I/O Registers:

Name	Value
PORTD	0x00
PORTD_OUT	0x00
PORTD_OUTSET	0x00
PORTD_OUTCLR	0x00
PORTD_OUTTGL	0x00
PORTD_IN	0x00
PORTD_INTCTRL	0x00
PORTD_INTOMASK	0x00
PORTD_INT1MASK	0x00
PORTD_INTFLAGS	0x00
PORTD_REMAP	0x00

Processor Status:

Name	Value
Program Counter	0x0000120
Stack Pointer	0x3FFF
X Register	0x0000
Y Register	0x0000
Z Register	0x0000
Status Register	0x00000000
Cycle Counter	0
Frequency	
Stop Watch	
Registers	
R00	0x00
R01	0x00
R02	0x00
R03	0x00
R04	0x00
R05	0x00
R06	0x00
R07	0x00
R08	0x00
R09	0x00
R10	0x00
R11	0x00
R12	0x00
R13	0x00
R14	0x00

Memory Window:

Address	Value
0x000000	ff c0 ff ff ff ff ff ff ff ff
0x000014	ff ff ff ff ff ff ff ff ff ff
0x000028	ff ff ff ff ff ff ff ff ff ff
0x00003C	ff ff ff ff ff ff ff ff ff ff
0x000050	ff ff ff ff ff ff ff ff ff ff
0x000064	ff ff ff ff ff ff ff ff ff ff

Sticky Note:

Name: Johnny Li
HW1

2.2: Screenshot of part 14