Brandon Pollack

Microprocessor lab1

Brandon Pollack

Lab1 Summary

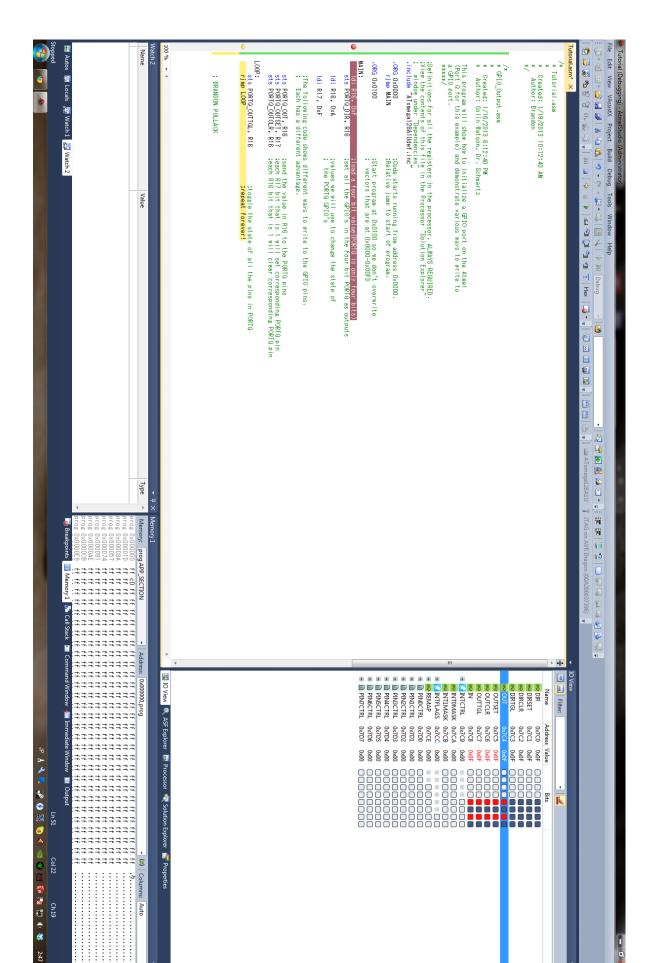Prelab

1: RAM starts at address 2000 so that is where any written memory should begin.

2: Program memory is a separate section of the memory than the data memory. Program memory is where the program (application) and other things are stored on non volatile Flash memory. Data memory has some memory mapped EEPROM, SRAM, IO memory, and external memory if it is available.

3. You can use XYZ registers to point to data in the program memory, these are stored in regiesters 26-31. All other registers can be loaded data, but only registers 16 and up can be used to compare immediate data.
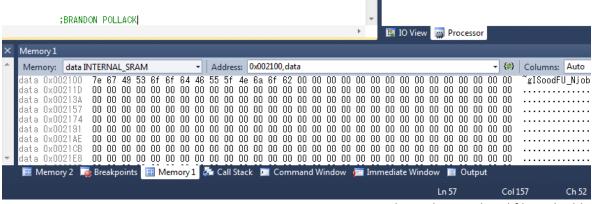
Problems encountered:

For some reason I don't yet understand, the assembler shifts all ORGS left one bit, I compensated by simply shifting back right.

Future application:

I know assembly a little better now and can manipulate data well.

Tutorial.asm* X

```
/*
 * Tutorial.asm
 *
 * Created: 1/19/2013 10:12:40 AM
 * Author: Brandon
 */

/*
 * GPIO_Output.asm
 *
 * Created: 1/16/2013 6:12:40 PM
 * Author: Colin Watson, Dr. Schwartz
 */

This program will show how to initialize a GPIO port on the Atmel
(Port Q for this example) and demonstrate various ways to write to
a GPIO port.
*****/

;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"

.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:
    ldi  R16, 0x0F      ;load a four bit value [PORTQ is only four bits]
    sts  PORTQ_DIR, R16 ;set all the GPIO's in the four bit PORTQ as outputs

    ldi  R16, 0xA       ;values we will use to change the state of
    ldi  R17, 0xF       ; the PORTQ GPIO's

;The following code shows different ways to write to the GPIO pins.
; Each has a different advantage.

    sts  PORTQ_OUT, R16    ;send the value in R16 to the PORTQ pins
    sts  PORTQ_OUTSET, R17 ;each R17 bit that is 1 will set corresponding PORTQ pin
    sts  PORTQ_OUTCLR, R16 ;each R16 bit that is 1 will clear corresponding PORTQ pin

LOOP:
    sts  PORTQ_OUTTGL, R16 ;toggle the state of all the pins in PORTQ
    rjmp LOOP             ;repeat forever!

; BRANDON POLLACK
```

Watch 2

Name	Value	Type

Memory 1

Memory:	prog APP_SECTION	Address:	0x0000000.prog

```
prog 0x000000  ff c0 ff ff ff ff ff ff ff ff ff ff ff ff ff
prog 0x00001D  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
prog 0x00003A  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
prog 0x000057  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
prog 0x000074  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
prog 0x000091  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
prog 0x0000AE  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
prog 0x0000CB  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
prog 0x0000E8  ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff
```

IO View

Name	Address	Value	Bits
DIR	0x7C0	0x0F	
DIRSET	0x7C1	0x0F	
DIRCLR	0x7C2	0x0F	
DIRTGL	0x7C3	0x0F	
OUT	0x7C4	0x0F	
OUTSET	0x7C5	0x0F	
OUTCLR	0x7C6	0x0F	
OUTTGL	0x7C7	0x0F	
IN	0x7C8	0x0F	
INTCTRL	0x7C9	0x00	
INT0MASK	0x7CA	0x00	
INT1MASK	0x7CB	0x00	
INTFLAGS	0x7CC	0x00	
REMAP	0x7CE	0x00	
PIN0CTRL	0x7D0	0x00	
PIN1CTRL	0x7D1	0x00	
PIN2CTRL	0x7D2	0x00	
PIN3CTRL	0x7D3	0x00	
PIN4CTRL	0x7D4	0x00	
PIN5CTRL	0x7D5	0x00	
PIN6CTRL	0x7D6	0x00	
PIN7CTRL	0x7D7	0x00	

Stopped   Ln 51   Col 22   Ch 19

IO View | Processor

Memory 1

| Memory: | data INTERNAL_SRAM | ▼ | Address: | 0x002100,data | ▼ | {#} | Columns: | Auto |

```
data 0x002100  7e 67 49 53 6f 6f 64 46 55 5f 4e 6a 6f 62 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ~gISoodFU_Njob
data 0x00211D  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...............
data 0x00213A  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...............
data 0x002157  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...............
data 0x002174  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...............
data 0x002191  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...............
data 0x0021AE  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...............
data 0x0021CB  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...............
data 0x0021E8  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...............
```

📋 Memory 2 🔴 Breakpoints 📊 Memory 1 📊 Call Stack ▶ Command Window 📑 Immediate Window 📄 Output

Ln 57 | Col 157 | Ch 52

This is the Simulated filtered table

IO View | Processor

Memory 1

| Memory: | data INTERNAL_SRAM | ▼ | Address: | 0x002100,data | ▼ | {#} | Columns: | Auto |

```
data 0x002100  7e 67 49 53 6f 6f 64 46 55 5f 4e 6a 6f 62 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ~gISoodFU_Njob..............
data 0x00211D  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...........................
data 0x00213A  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...........................
data 0x002157  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...........................
data 0x002174  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...........................
data 0x002191  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...........................
data 0x0021AE  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...........................
data 0x0021CB  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...........................
data 0x0021E8  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...........................
```

📋 Memory 2 🔴 Breakpoints 📊 Memory 1 📊 Call Stack ▶ Command Window 📑 Immediate Window 📄 Output

Ln 55 | Col 1 | Ch 1 | INS

JP A 🔧 🔵 💾 ⚡ 📛 ✅ ◀ 💾 📥 💿 🌱 Ⅱ SW 📋 ▣ 🔊 ❄ 11:00 AM

This is emulated directly with the AVR CPU

Pseudo code
Load zero to R3
Load table values for input table
Pointer Z to table
Pointer X to SRAM

loop   load R0 with Z and inc X
        Compare values
        BRLT   dontwrite
        Write to X and inc
        jump back loop
dontwrite   compare R3 to R0
             BREQ done
             jump loop

done   jump done