

Experiment Name:     Module 0: ESE 381 Design Environment

Experiment Number:   #0

Lab Participant 1:    Ben Jarnagin, 107692807

Lab Participant 2:    Alex Tkatchev, 106014516

Lab Section:          #1

Bench:                #3

```
#####
#
# IAR C/C++ Compiler V6.70.1.929 for Atmel AVR      29/Jan/2016 02:06:45
# Copyright 1996-2015 IAR Systems AB.
# Standalone license - IAR Embedded Workbench 4K Kickstart edition for Atmel AVR 6.70
#
# Source file = G:\labs\0\2-PRELIMINARY\T4\sws_and.c
# Command line =
#   G:\labs\0\2-PRELIMINARY\T4\sws_and.c --cpu=m128 -ms -o
#   G:\labs\0\2-Preliminary\T4\Debug\Obj -ICN
#   G:\labs\0\2-Preliminary\T4\Debug\List -y --initializers_in_flash
#   --no_cse --no_inline --no_code_motion --no_cross_call --no_clustering
#   --no_tbaa --debug -DENABLE_BIT_DEFINITIONS -e --eeprom_size 4096
#   --clib -On
# List file = G:\labs\0\2-Preliminary\T4\Debug\List\sws_and.lst
# Object file = G:\labs\0\2-Preliminary\T4\Debug\Obj\sws_and.r90
#
#####
```

G:\labs\0\2-PRELIMINARY\T4\sws\_and.c

```
1  /*
2  *   title:      sws_and.c
3  *   description: performs AND bitwise between PD7-PD4 and PD3-PD0. Output
4  *               to PB3-PB0 as active low. PB7-PB4 are kept constant high.
5  *   target:     ATMEGA128
6  */
7
8  #include <iom128.h>

\           In segment ABSOLUTE, at 0x30
\ union <unnamed> volatile __io _A_PIND
\   _A_PIND:
\ 00000000      DS8 1

\           In segment ABSOLUTE, at 0x31
\ union <unnamed> volatile __io _A_DDRD
\   _A_DDRD:
\ 00000000      DS8 1

\           In segment ABSOLUTE, at 0x32
\ union <unnamed> volatile __io _A_PORTD
\   _A_PORTD:
\ 00000000      DS8 1

\           In segment ABSOLUTE, at 0x37
\ union <unnamed> volatile __io _A_DDRB
\   _A_DDRB:
\ 00000000      DS8 1

\           In segment ABSOLUTE, at 0x38
\ union <unnamed> volatile __io _A_PORTB
\   _A_PORTB:
\ 00000000      DS8 1
```

```

\           In segment CODE, align 2, keep-with-next
10   int main(void){
\       main:
11       //setup input and output pins
12       DDRD = 0x00;
\ 00000000 E000      LDI   R16, 0
\ 00000002 BB01      OUT   0x11, R16
13       PORTD = 0xFF; //switches will use internal PUN
\ 00000004 EF0F      LDI   R16, 255
\ 00000006 BB02      OUT   0x12, R16
14       DDRB = 0xFF;
\ 00000008 EF0F      LDI   R16, 255
\ 0000000A BB07      OUT   0x17, R16
15
16       //define variables
17       char tempA,tempB;
18
19       //while loop for AND process
20       while(1){
21         tempA = PIND;      //tempA = 0bABCDEFGH
\         ??main_0:
\ 0000000C B300      IN    R16, 0x10
\ 0000000E 2F10      MOV   R17, R16
22         tempB = tempA>>4; //tempB = 0b0000ABCD
\ 00000010 2F01      MOV   R16, R17
\ 00000012 9502      SWAP  R16
\ 00000014 700F      ANDI  R16, 0x0F
\ 00000016 2F20      MOV   R18, R16
23
24         //PORTB = ~( PIND & (PIND>>4) ) //Volatile solution
25
26         PORTB = ~( tempA & tempB ); //AND values and invert for active low
\ 00000018 2F01      MOV   R16, R17
\ 0000001A 2302      AND   R16, R18
\ 0000001C 9500      COM   R16
\ 0000001E BB08      OUT   0x18, R16
\ 00000020 CFF5      RJMP  ??main_0
\ 00000022          REQUIRE _A_DDRD
\ 00000022          REQUIRE _A_PORTD
\ 00000022          REQUIRE _A_DDRB
\ 00000022          REQUIRE _A_PIND
\ 00000022          REQUIRE _A_PORTB
27     }
28 }

```

Maximum stack usage in bytes:

RSTACK Function

-----

2 main

Segment part sizes:

Bytes Function/Label

```
-----  
1  _A_DDRB  
1  _A_DDRD  
1  _A_PIND  
1  _A_PORTB  
1  _A_PORTD  
34 main
```

5 bytes in segment ABSOLUTE

34 bytes in segment CODE

34 bytes of CODE memory

0 bytes of DATA memory (+ 5 bytes shared)

Errors: none

Warnings: none

```
#####
#
# IAR C/C++ Compiler V6.70.1.929 for Atmel AVR      29/Jan/2016 02:06:45
# Copyright 1996-2015 IAR Systems AB.
# Standalone license - IAR Embedded Workbench 4K Kickstart edition for Atmel AVR 6.70
#
# Source file = G:\labs\0\2-PRELIMINARY\T5\sws_alu.c
# Command line =
#   G:\labs\0\2-PRELIMINARY\T5\sws_alu.c --cpu=m128 -ms -o
#   G:\labs\0\2-Preliminary\T5\Debug\Obj -ICN
#   G:\labs\0\2-Preliminary\T5\Debug\List -y --initializers_in_flash
#   --no_cse --no_inline --no_code_motion --no_cross_call --no_clustering
#   --no_tbaa --debug -e --eeprom_size 4096 --clib -On
# List file = G:\labs\0\2-Preliminary\T5\Debug\List\sws_alu.lst
# Object file = G:\labs\0\2-Preliminary\T5\Debug\Obj\sws_alu.r90
#
#####
```

G:\labs\0\2-PRELIMINARY\T5\sws\_alu.c

```
1  /*
2  *   title:      sws_alu.c
3  *   description: performs AND, OR, XOR or ~ bitwise between PD5-PD3 and
4  *                PD2-PD0 based on selection by PD7-PD6. Output to
5  *                PB2-PB0 as active low. PB7-PB3 are kept constant high.
6  *   target:    ATMEGA128
7  */
8
9  #include <iom128.h>

\           In segment ABSOLUTE, at 0x30
\ union <unnamed> volatile __io _A_PIND
\   _A_PIND:
\ 00000000      DS8 1

\           In segment ABSOLUTE, at 0x31
\ union <unnamed> volatile __io _A_DDRD
\   _A_DDRD:
\ 00000000      DS8 1

\           In segment ABSOLUTE, at 0x32
\ union <unnamed> volatile __io _A_PORTD
\   _A_PORTD:
\ 00000000      DS8 1

\           In segment ABSOLUTE, at 0x37
\ union <unnamed> volatile __io _A_DDRB
\   _A_DDRB:
\ 00000000      DS8 1

\           In segment ABSOLUTE, at 0x38
\ union <unnamed> volatile __io _A_PORTB
\   _A_PORTB:
\ 00000000      DS8 1
```

```

\          In segment CODE, align 2, keep-with-next
11      int main(void){
\          main:
12          //setup input and output pins
13          DDRD = 0x00;
\ 00000000 E000      LDI  R16, 0
\ 00000002 BB01      OUT  0x11, R16
14          PORTD = 0xFF;
\ 00000004 EF0F      LDI  R16, 255
\ 00000006 BB02      OUT  0x12, R16
15          DDRB = 0xFF;
\ 00000008 EF0F      LDI  R16, 255
\ 0000000A BB07      OUT  0x17, R16
16
17          //define variables
18          char op, tempA, tempB;
19
20          while(1){
21              tempA = PIND;      //tempA = 0bABCDEFGH
\          ??main_0:
\ 0000000C B300      IN   R16, 0x10
\ 0000000E 2F20      MOV  R18, R16
22              tempB = tempA>>3;    //tempB = 0b000ABCDE
\ 00000010 2F02      MOV  R16, R18
\ 00000012 9506      LSR  R16
\ 00000014 9506      LSR  R16
\ 00000016 9506      LSR  R16
\ 00000018 2F30      MOV  R19, R16
23              op = tempB>>3;    //op = 0b000000AB
\ 0000001A 2F03      MOV  R16, R19
\ 0000001C 9506      LSR  R16
\ 0000001E 9506      LSR  R16
\ 00000020 9506      LSR  R16
\ 00000022 2F10      MOV  R17, R16
24
25              switch(op){
\ 00000024 2F01      MOV  R16, R17
\ 00000026 5000      SUBI  R16, 0
\ 00000028 F039      BREQ  ??main_1
\ 0000002A 950A      DEC  R16
\ 0000002C F039      BREQ  ??main_2
\ 0000002E 950A      DEC  R16
\ 00000030 F039      BREQ  ??main_3
\ 00000032 950A      DEC  R16
\ 00000034 F039      BREQ  ??main_4
\ 00000036 C007      RJMP  ??main_5
26              case 0:    //AND
27                  tempA &= tempB;
\          ??main_1:
\ 00000038 2323      AND  R18, R19
28              break;

```

```

\ 0000003A C005      RJMP  ??main_5
29
30      case 1:  //OR
31      tempA |= tempB;
\      ??main_2:
\ 0000003C 2B23      OR    R18, R19
32      break;
\ 0000003E C003      RJMP  ??main_5
33
34      case 2:  //XOR
35      tempA ^= tempB;
\      ??main_3:
\ 00000040 2723      EOR    R18, R19
36      break;
\ 00000042 C001      RJMP  ??main_5
37
38      case 3:  //COMP
39      tempA = ~tempA;
\      ??main_4:
\ 00000044 9520      COM    R18
40      break;
41      }
42
43      PORTB = ~( tempA & 0x03 );
\      ??main_5:
\ 00000046 2F02      MOV    R16, R18
\ 00000048 7003      ANDI   R16, 0x03
\ 0000004A 9500      COM    R16
\ 0000004C BB08      OUT    0x18, R16
\ 0000004E CFDE      RJMP  ??main_0
\ 00000050      REQUIRE _A_DDRD
\ 00000050      REQUIRE _A_PORTD
\ 00000050      REQUIRE _A_DDRB
\ 00000050      REQUIRE _A_PIND
\ 00000050      REQUIRE _A_PORTB
44      }
45      }

```

Maximum stack usage in bytes:

RSTACK Function

```

-----
2  main

```

Segment part sizes:

Bytes Function/Label

```

-----
1  _A_DDRB
1  _A_DDRD
1  _A_PIND
1  _A_PORTB

```

1 \_A\_PORTD  
80 main

5 bytes in segment ABSOLUTE  
80 bytes in segment CODE

80 bytes of CODE memory  
0 bytes of DATA memory (+ 5 bytes shared)

Errors: none

Warnings: none