EE346 Final Exam - Practice Problems

WARNING

No questions contained in this document, or any other worksheet, may be reproduced on your page of notes. Use the page of notes to record data about the ATmega Microcontroller and other class related material.

In addition to the material presented here, another great source of practice problems can be found in the Review Material folder.

Contents

The following questions cover the following worksheets.

ATmega328P Subsystems

- ATmega328P Peripherals
- ATmega328P Serial Communications
- ATmega328P Timers and Interrupts
- ATmega328P External Interrupts

AVR Microprocessor

- AVR Bits and Bytes
- AVR SREG
- AVR Load-Store Programming
- AVR Addressing Modes II
- Indirect Addressing Mode Questions.
- AVR Branching

Lecture material not explicitly covered by this document include the following.

- AVR Assembly I — Fundamental material required to answer many if not all of the questions.
- AVR Microcontroller — Fundamental material required to answer many of the questions.
- AVR Subroutines — Required to answer many of the questions.

ATmega328P Subsystems

The following questions are based on the Arduino Proto-Shield and the following table.

GPIO Ports

You may assume the PUD in MCUCR is cleared (normal operating mode). Review ATmega328P GPIO lecture notes for help in answering these questions.

DDxn	PORTxn	PUD (in MCUCR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	Pxn will source current if ext. pulled low.
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	X	Output	No	Output Low (Sink)
1	1	X	Output	No	Output High (Source)

1. Using two single bit instructions, configure Port D bit 2 as an input without a pull-up resistor.

```
cbi    DDRD, 2
cbi    PORTD, 2
```

2. Using two single bit instructions, configure and set to one (1) Port D bit 5.

```
sbi    DDRD, 5
sbi    PORTD, 5
```

3. Using six byte instructions, configure Port C bits 5 to 0 as inputs with pull-up resistors. Do not modify bits 7 to 6. Use register 16 and 17 as working (temporary) registers.

```
in     r16, DDRC
in     r17, PORTC
cbr    r16, 0b00111111
sbr    r17, 0b00111111
out    DDRC, r16
out    PORTC, r17
```

4. Using six byte instructions, configure Port D bits 7 and 6 as inputs with pull-up resistors. Do not modify bits 5 to 0. Use register 16 and 17 as working (temporary) registers.

```
in     r16, DDRD
in     r17, PORTD
cbr    r16, 0b11000000
sbr    r17, 0b11000000
out    DDRD, r16
out    PORTD, r17
```

SPI Interface

Review ATmega328P SPI Serial Communications lecture notes for help in answering these questions.

5. Assume the SPI subsystem of the ATmega328P is configured as the master, and outputting to an 8-bit Shift Register with Output Latches (74HC595) as shown in the Arduino Proto-Shield schematic. How many pins are needed to implement this interface? 3

6. Assume the SPI subsystem of the ATmega328P is configured as the master, and outputting to an 8-bit Shift Register with Output Latches (74HC595) as shown in the Arduino Proto-Shield schematic. Plus, I add an additional 8-bit Parallel In Serial Out Shift Register to read the 8 switches.

I wire the output of this register to the Master In Serial Out (MISO) pin of the Arduino. How many pins are needed to implement this interface? 4

7. The SPI interface is implemented with a maximum of four(4) pins (MISO, MOSI, SCK, and SS). On which of these pins would you see the serial data output? MOSI

8. The SPI interface is implemented with a maximum of four(4) pins (MISO, MOSI, SCK, and SS). On which of these pins would you see the serial clock? SCK

9. Using three byte instructions, configure Port B bits 5, 3, and 2 (SCK, MOSI, SS) as outputs. Do not modify bits 5 to 0. Use register 16 as a working (temporary) register.

```
in    r16,DDRB
sbr   r16,0x2C
out   DDRB,r16
```

10. Using two byte instructions, set SPCR Enable (SPE) bit 6, Master (MSTR) bit 4, clock rate fck/16 (SPR1 = 0, SPR0 = 1)of the configure the SPI Control Register (SPCR). Set all other bits to zero (0).

```
ldi   r16,0x51
out   SPCR,r16
```

11. Using a single I/O instruction, output register r8 to the SPI by writing its contents to the SPI Data Register (SPDR).

```
out   SPDR, r8
```

12. After a byte is written to the SPI Data Register, we can poll the SPI interrupt flag (bit) in the SPI status register (SPSR) to find out when the byte has been transmitted. Write a subroutine to wait for the end of transmission by implementing the following comments.

```
spiTxWait:
; Wait for transmission complete
      in    r16,SPSR    // input to register r16 the contents of SPSR
      bst   r16,SPIF    // save bit 7 (SPIF) to the T-bit in SREG
      brtc  spiTxWait   // continue polling the SPIF if T-bit is clear
      ret               // return
```

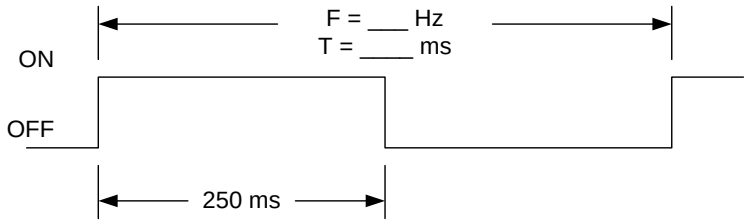
Timer Subsystem

Review ATmega328P Timers lecture notes for help in answering these questions.

13. A sinusoid signal repeats itself 60 times each second. Waht is the period of this signal?

    16.67 ms

14. What is the frequency, period, and duty cycle of the following waveform.



F = 2 Hz, T = 500 ms, duty cycle = 50%

15. What is the maximum delay that can generated by our 16-bit Timer 1 with a prescale value of 64 and a system clock frequency of 16 MHz.

    262.14 msec

16. A prescaler of 1024 (clk/1024) is applied to our 8-bit time 0, while a prescaler of 256 (clk/256) is applied to our 8-bit time 2. Finally, a prescaler of 1 (clk) is applied to our 16-bit time 1. Assuming a system clock frequency of 16 MHz, which timer(s) will generate a delay of 4.096 msec.

    Timer 0 = 16.384 msec, Timer 1 = 4.096 msec, Timer 2 = 4.096 msec

    Answer is Timer 1 and Timer 2.

17. What value would you load into the TCNT1H and TCNT1L register pair to generate a delay of 250 msec.

    TCNT1H = 0x0B and TCNT1L = 0xDC

Interrupts

Review Timer Interrupts lecture notes for help in answering these questions.

18. In which register can you find the global interrupt enable bit? SREG
19. Where can you find the Interrupt Vector Table (IVT)? Flash Program Memory
20. How many words (16-bits) are reserved for each entry in the IVT? 2
21. When an interrupt is triggered, what register is placed on the stack? PC
22. Why is this register saved? So the currently running process can continue execution when the processor is completes running the ISR.
23. Why is the first instruction executed, at address 0x0000, always a jump? So the AVR processor does not accidently execute an ISR.
24. If the instruction at address 0x0000 is not a jump what ISR will be executed? External Interrupt Request 0 INT0 at address 0x0002.
25. What immediately happens when an interrupt occurs? The microcontroller completes the current instruction, stores the address of the next instruction on the stack, and clears the SREG I-bit.

26. What is instruction is required to return from an Interrupt? `reti`

27. In what way is a `ret` instruction different from an `reti` instruction? An `reti` instruction sets the global interrupt enable flag bit I in SREG.

28. What is one of the last things and ISR does before it returns control to the interrupted program? Restores the Program Counter and Enables the global interrupt flag bit I.

29. What is the first and second to last thing your ISR should do? Save and restore the SREG register.

30. What is the last thing your ISR should do? Execute an reti instruction.

31. Where should your ISR save the SREG register? One of the 32 general purpose registers.

32. Where should your ISR save general purpose registers modified by the ISR? Registers modified by the ISR should be temporarily placed on the stack.

33. Upon return from an ISR and enabling the global interrupt flag ; the AVR processor finds another interrupt waiting to be executed. What will happen next? The main program will execute one more instruction before any pending interrupt is run.

Timer Interrupts

Review Timer Interrupts lecture notes for help in answering these questions.

34. Just before a Timer/Counter Overflow Interrupt is run, what IVT address needs to be placed in the program counter (PC)? 0x001A

35. How many bits need to be set for a Timer/Counter 1 Overflow interrupt to be triggered? 3 (SREG I, TOIE1, TOV1)

36. What triggers a Timer/Counter 1 Overflow Interrupt? Counter goes from 0xFFFF to 0x0000

External Interrupts

Review ATmega328P External Interrupts lecture notes for help in answering these questions.

37. While the global interrupt SREG bit I is cleared, both the Timer/Counter 1 Overflow bit is set (IVT Address IVT 0x000D) Timer1 OVF and an external interrupt is received (IVT Address 0x0001) INT0. Assuming the interrupt enable bit for both interrupts is set, what will happen when the global interrupt bit is set (enabled)? The External Interrupt Request 0 will be run.

AVR Microprocessor

Introduction to Microcontrollers including History

See Lecture Notes and Quiz 1 Review Material

38. What memory model is used by the EDVAC? Princeton

39. ISA is an abbreviation for what? Instruction Set Architecture

40. How many general purpose registers does the AVR processor have? 32

41. What is the mnemonic for the last AVR general purpose register? r31

Status Register (SREG)

Review AVR SREG lecture notes for help in answering these questions.

42. Using a single bit instructions, disable all interrupts.

```
cli
```

43. What is wrong with this instruction `push SREG`? You can only push a general purpose register onto the stack.

44. Assume the subtract instruction `sub r16, r17` has just been run by the AVR microprocessor. Complete the table provided. The "difference" column should reflect the contents of register r16 after the subtraction operation (leave the answer in 2's complement form) and not the actual difference (i.e., if done using your calculator). Use the AVR Studio simulator to verify your answers.

			signed	unsigned						
r16	r17	difference	relationship	relationship	H	S	V	N	Z	C
3B	3B	00	+ = +	=	0	0	0	0	1	0
3B	15	26	+ > +	>	0	0	0	0	0	0
15	3B									
F9	F6									
F6	F9									
15	F6									
F6	15									
68	A5									
A5	68									

Addressing Modes

Direct and Immediate Addressing Modes

Review "AVR Load-Store Programming"

45. You can find the operand for the immediate addressing mode in what type of memory? Flash Program Memory

46. You can find the operand for the direct addressing mode of an `lds` and `sts` instruction in what type of memory? SRAM Data Memory

47. You can find the operand for the direct addressing mode of an `in` and `out` instruction in which address space(s)? SRAM Data Memory and I/O Register Memory

48. The address space of which two addressing modes overlap? SRAM Data Direct and I/O Register Memory Direct

49. What register is both a source and a destination for the instruction `add r16, r17`? R16

50. What addressing mode is used for the *destination* operand address field of the instruction `lds r16, 0x33`? Register Direct

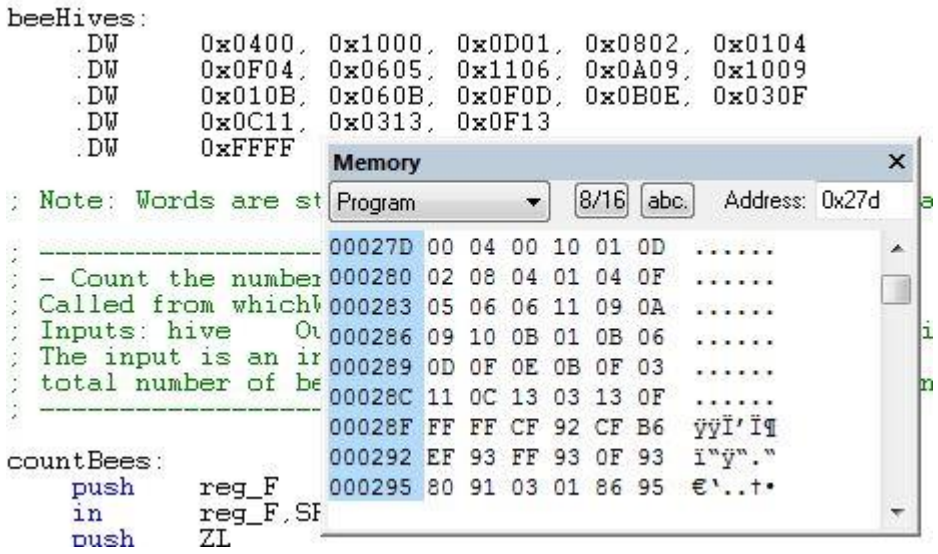
Indirect Addressing Modes

Review "AVR Indirect Addressing", "Application of the Indirect Addressing Mode", and "Indirect Addressing Mode Questions" lecture notes for help in answering these questions.

51. What addressing mode should you use if you want to look up a pre-defined value in a table (data is known at assembly time)? Flash Program Indirect
52. What addressing mode should you use if you want to look up a value in a table (data is known at run time)? SRAM Data Indirect
53. What addressing mode is used for the source operand of an `lpm` instruction? Flash Program Indirect
54. What register pair is found in the source operand address field of an `lpm` instruction? Z
55. What register numbers correspond to pre-defined mnemonics ZH:ZL? R31:r30
56. What two addressing mode should you use if you want to work with a table of data located in SRAM (data is known at run time)? 1. SRAM Data Indirect and 2. SRAM Data Indirect with Displacement
57. What addressing mode is used for the source operand of an `ld` instruction? SRAM Data Indirect
58. Which three register pairs may be found in the source operand address field of an `ld` instruction? X, Y, and Z
59. What two 8-bit register mnemonics are used to define the X register pair? XH:XL and R27:r26
60. What addressing mode is used for the *destination* operand address field of the instruction `lpm r16, Z`?The key to this question is in italics "destination." The destination operand uses the Register Direct addressing mode, the source operand is register indirect.
61. Write a code snip-it to load the 3rd byte (index = 2) of data from a table (label = TABLE) located in Flash Program memory. *Answer left up to the student.*
62. What is wrong with this instruction `lds r16, low(Table << 1)`? The source operand address field is of the immediate addressing mode type. The `ldi` instruction should have been used in place of the `lds` instruction.
63. I want to load the number 33_{16} into register 16. Why can I not use the instruction `lds r16, 0x33` to do this? The source operand address field should be immediate , not SRAM Data direct. The `lds` instruction would load r16 with the contents of SRAM memory at location 0x33. The `ldi` instruction should have been used in place of the `lds` instruction.
64. The AVR processor saves bytes of data in Flash Program Memory using what memory byte ordering? Little Endian

65. Big Endian saves what half of a 2 byte (16-bit) word in the first byte (lowest address)? The most significant (Big) byte.

66. Each entry (.DW) in the following table contains two bytes (1 16-bit word). These two bytes provide the row and column of a room containing bees. For example with respect to the maze, the room in row 00 column 04 contains 1 bee. If we look at the first entry we see it contains 0x0400. Comparing this with the corresponding Program Memory Window in AVR Studio the least significant byte is saved in the lowest order byte; so 0x0400 would be save as bytes 0x00 and 0x40. What form of Byte ordering (Big or Little Endian) does this represent? Little Endian



67. What addressing mode is great for implementing look-up tables in Flash Program Memory? Program Memory Indirect

68. Build a program to convert a 4-bit gray-code number into binary. *Solution left to the student*

69. What instruction is used to divide a register by two? `lsr`

70. Write a program to set a 32 byte buffer located in SRAM to the blank ‘’ ASCII character. *Solution left to the student*.

71. To modify the seven segment display on the proto-shield you must write to register 8 and do what? `call spiTx`

Instructions Encoding

Be able to encode instructions

72. `ldi` Load Sore Programming and AVR Addressing Indirect Lectures

73. `jmp and call` AVR Branching and AVR Stack Operations

74. `rjmp and rcall` AVR Branching and AVR Stack Operations

75. `ret` AVR Stack Operations

Programming, Labs, and Using the AVR Studio Simulator

Assembly Directives

Load-Store Programming

See Addressing Modes

Arithmetic and Logic Instructions

ALU Instructions

To be generated…

Bits and Bit-test Instructions

Review AVR Bits and Bytes lecture notes for help in answering these questions.

76. What instruction(s) could you use to clear the carry bit?

    clc, bclr SREG_C, bclr 0

77. What instruction would you use to clear PORT D bit 3?

    cbi PORTD,3 or cbi 0x0B,3

78. What is wrong with this the instruction `cbi TIMSK1, 0`?

    TIMSKI is located in the extended I/O address space of the ATmega328P microcontroller and is therefore not accessible.

79. Write an instruction to clear bit 4, 2, and 0 in register r16. `cbr r16, 0b0001 0101`

80. Write an instruction to clear bit 4, 2, and 0 in register r16 without using the cbr instruction.

    andi r16, 0b1110 1010 or **andi r16, 0xEA**

81. Write an instruction to set bit 4, 2, and 0 in register r16. `sbr r16, 0b0001 0101`

82. Write an instruction to set bit 4, 2, and 0 in register r16 without using the sbr instruction

    ori r16, 0b0001 0101 or and r16, 0x15

83. What is wrong with this instruction to toggle bit 4, 2, and 0 in register 16? `eor r16, 0b0001 0101`

    The eor instruction works with two registers.

84. Write an instruction sequence to toggle bit 4, 2, and 0 in register 16.

    ```
    ldi..r17, 0b0001 0101
    eor r16, r17
    ```

85. Write an instruction sequence to set bits 4, 2, and 0 to 101_2 in register 16, without modifying any other bits.

```
cbr   r16, 0b0000 0100
sbr   r16, 0b0001 0001
```

86. Write an instruction to set bits 7, 6, 5, 4, 3, 2, 1, 0 in register 16, without using the sbr or or instruction

```
ser   r16
```

87. Write a program to create a "software wire" between switch 0 to the decimal point of the 7-segment display on the proto-shield. *Solution left to the student*

88. Pulse Clock input of Proto-Shield Debounce D Flip-flop (PORTD bit 5). Assume currently at logic 0.

```
sbi   PORTD, 5
cbi   PORTD, 5
```

89. What logical instruction is implemented by the tst instruction? and

90. What is the difference between the and `and` the `tst` instruction?

    The `tst` instruction performs the `and` operation without modifying the destination operand.

91. What does the `cp` instruction have in common with the `tst` instruction?

    Both do not modify the register(s) in the operand field. Both are used to set flag bits in SREG.

92. Assuming register 16 contains 0x00. What would the Z-bit be set/cleared to after executing the instruction `tst r16`? Z = 1

93. What instruction is used to disable all interrupts? `cli`

94. How many operand(s) does the `tst` instruction have? 1

95. Write the Boolean instrucion that would be functionally equivalent to the `tst r16` instruction.

```
and   r16, r16
```

96. What instruction would be used to implement the logic operation represented by the question mark (?) in Figure 1? `eor`

97. What phrase best describes the operation shown in Figure 1 box A? Don't Change

98. What word best describes the operation shown in Figure 1 box B? Toggle

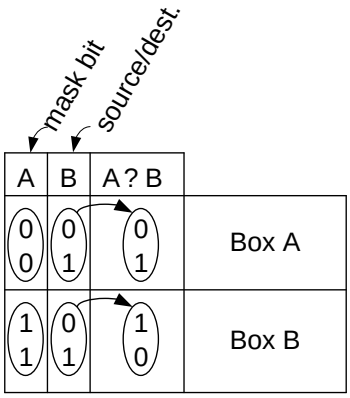
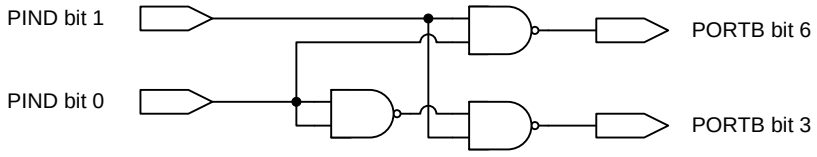


Figure 1 Truth Table of the Exclusive OR operator

99. Write the code needed to implement the following circuit.



100. Using just two instructions, test the contents of register16, if all the bits are cleared (equal to zero) branch to the label is_zero.

```
tst   r16
breq  is_zero
```

101. Write a subroutine to test if a bit set in byte variable imageD corresponds to a bit set in byte variable imageR. If the corresponding bit is set return a non-zero value in register r24.

```
hitWall:
  lds   r24,imageD
  lds   r16,imageR
  and   r24, r16
  ret
```

102. Given that only one bit is set in register r16, write a subroutine to if this bit is at either edge of the register (bits 7 and 0). If it isn't then branch to label contScan.

```
ldi   r19, 0b100000001
and   r19,r16     // test if LED hit is at an edge
breq  contScan    // continue scan if z = 0
```

103. Using the exclusive or instruction, write a subroutine to test if byte variable row contains -1 (0xFF). If it does return zero in register r24.

```
lds   r24, row
ldi   r16, 0xFF
eor   r24, r16
ret
```

104. In ASCII, the character 'H' is encoded as 0x48 and the character A is 0x41. Assume that register r16 contains 'H' or 'I.' We want to write a subroutine that when called will convert 'H' to 'I' and 'I' to 'H.' One way to accomplish this to toggle bits 3 and 0 in register r16 each time the subroutine is called. Can you write the subroutine?

```
ldi   r17, 0b00001001
eor   r16, r17
ret
```

105. Write an instruction to divide a signed number in register r16 by 2? `asr r16`

106. Write an instruction to multiply an unsigned number in register r16 by 2? `lsl r16`

107. Each entry in a table contains 16 bits (2 bytes). The address of this table is saved in the register pair ZH:ZL. In my program I need to access bytes within the table. To do this I need to change the word address in Z into a byte address. This is done by shifting register Z to the left by one bit. Write the code needed to shift the register pair ZH:ZL to the left.

```
lsl   ZL    // carry-in is zero
rol   ZH    // shift in the carry-out from ZL
```

Control Transfer

Branching

Review AVR Branching lecture notes for help in answering these questions.

108. All unconditional jump instructions using the relative addressing mode, utilize 12 bits to encode the distance the program is to jump relative to the program counter (PC). Given that this 12 bit number is saved using 2's complement notation, what is the range in words (16-bits) that the AVR processor can jump for this type of instruction? Branch relative to PC + ($-2^{k-1} \Rightarrow 2^{k-1}-1$, where k = 12) + 1 ⇨PC-2048 to PC+2047, within 16 K word address space of ATmega328P

109. All conditional branch instructions using the relative addressing mode, utilize 7 bits to encode the distance the program is to branch relative to the program counter (PC). Given that this 7 bit number is saved using 2's complement notation, what is the range in words (16-bits) that the AVR processor can branch for this type of instruction? All branch relative to PC + ($-2^{k-1} \Rightarrow 2^{k-1}-1$, where k = 7) + 1 ⇨PC-64 to PC+63, within 16 K word address space of ATmega328P

110. Which instructions do you typically find before a relative branch instruction? Compare and test instructions? `cp, cpc, cpi, tst, bst`

111. Compare and test instructions like `cp, cpc, cpi, tst, bst` do not modify any of the 32 general purpose registers. What register(s) do they modify? SREG

112. What type of instruction typically follows a compare and test instruction, like `cp, cpc, cpi, tst, bst`? A conditional control transfer instruction.

113. Why does a compare or test instruction, typically precede a conditional branch instruction? The compare or test instruction sets or clears flag bits within the status register (SREG) used by the conditional branch instruction to make a decision.

114. What is wrong with the following two sequential instructions? `cpi r16, 0x33,` followed by `rjmp there` The first instruction is setting and clearing SREG bits to be used by a conditional branch instruction. The second instruction is an unconditional jump instruction which doesn't need to make any decision (i.e., it is always going to jump).

115. Why is the following two sequential instructions silly? `clr r16,` followed by `ldi r16, 0x33` The second instruction will write 33_{16} to r16, so there is no reason to clear it.

116. What bit(s) within SREG are never modified by a compare or test instruction? I (global interrupt enable), T (bit copy storage).

117. What is wrong with the following two sequential instructions? `cpi r16, 0x33,` followed by `brts there` The first instruction is setting and clearing SREG bits associated with an ALU instruction. The second instruction is a conditional branch instruction which tests to see if the T-bit is set. The T-bit is not modified by a compare instruction.

118. Assuming that r16 contains the value 0x33. What value would be in r16 after the instruction `cpi r16, 0x33` is executed? R16 = 0x33

119. Assuming that r16 contains the value 0x33. What value would be in r16 after the instruction `subi r16, 0x33` is executed? R16 = 0x00

120. Assuming that r16 contains the value 0x33. What 1-bit value would be contained in the Z-bit in SREG after the instruction `cpi r16, 0x33` is executed? Z = 1

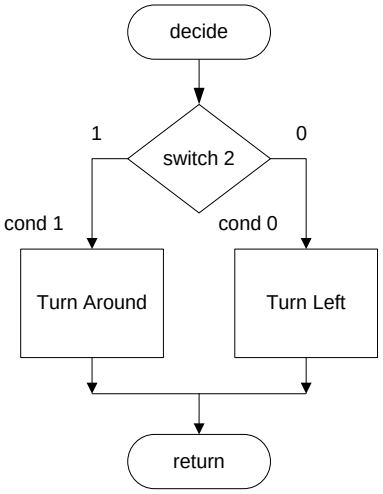
121. What type of instruction never modify bits within the SREG register? Data Transfer

122. What hex value encodes the opcode (`rcall`). D

123. Write the machine code instruction that encodes the assembly instruction `rjmp loop`. The instruction is located at address 0x215. The label `loop` is located at address 0x01E2. *Solution left to the student*

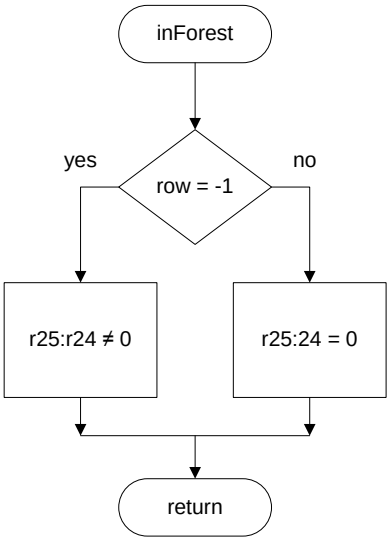
124. The instruction `rjmp end_switch` is located at address 0x01ED and is encoded as 0xC020. At what address would you find the label `end_switch`? *Solution left to the student*

125. Write the code to implement the following flow-chart



Solution left to the student

126. Write the code to implement the following flow-chart



Solution left to the student

Looping

Review AVR Looping lecture notes for help in answering these questions.

Subroutines

The Basics

Review AVR Subroutine Basics lecture notes for help in answering these questions.

Stack Operations

Review AVR Stack Operations lecture notes for help in answering these questions.

Introduction to C++

Review C++ Introduction plus "Test Your Knowledge" questions.