

CSS332: Microcontrollers and Applications

Midterm Mock Exam

curated by The Peanuts

Name.....ID.....Section.....Seat No.....

Conditions: Semi-Closed Book

Directions:

1. This exam has 15 pages (including this page).
2. If you finish early, stare at your paper like it's a masterpiece.
3. Write your name clearly, even microcontrollers need proper initialization.
4. Reading the problem is optional but highly recommended.
5. Answers must be written in a structured format. Random noise (scribbles) will not be processed correctly.
6. Going to the toilet may deduct your score, hold it like your grades depend on it.

For solution, click here.

Problem 1

Answer the following questions:

Convert binary 1101 1011 to decimal

Convert decimal 123 to binary and hexadecimal

Convert hexadecimal 0xA7 to binary and decimal

Problem 2

For the addition of 8-bit unsigned numbers: $0xF7 + 0x19$, determine:

- a) The result in hexadecimal
- b) The state of the Carry (C) flag
- c) The state of the Zero (Z) flag
- d) The state of the Half-carry (H) flag

For the subtraction of signed 8-bit numbers: $0x65 - 0x8A$, determine:

- a) The result in hexadecimal
- b) The state of the Negative (N) flag
- c) The state of the Overflow (V) flag
- d) Is this result correct from a signed arithmetic perspective? Explain.

Problem 3

Consider the following AVR assembly program:

```
.ORG 0
LDI R16, HIGH(RAMEND)
OUT SPH, R16
LDI R16, LOW(RAMEND)
OUT SPL, R16

LDI R20, 0x5A
LDI R21, 0x3F
LDI R22, 0x81
PUSH R20
PUSH R21
PUSH R22
POP R23
POP R24
POP R25
```

- a) What values will be stored in registers R23, R24, and R25 after executing this program?

- b) Show the values stored and the stack pointer position each time at *three* points: after all three PUSH instructions, after the first POP instruction, and after all three POP instructions.

| | | | | | |
|--------|--|--------|--|--------|--|
| 0x08FB | | 0x08FB | | 0x08FB | |
| 0x08FC | | 0x08FC | | 0x08FC | |
| 0x08FD | | 0x08FD | | 0x08FD | |
| 0x08FE | | 0x08FE | | 0x08FE | |
| 0x08FF | | 0x08FF | | 0x08FF | |

- c) What would happen if the PUSH and POP instructions were executed without initializing the stack pointer (the first four instructions)? Explain.

Problem 4

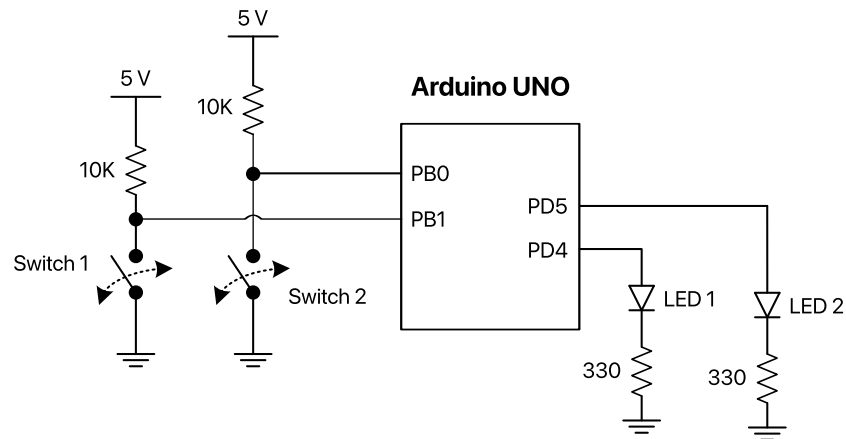
Calculate the total time delay (in milliseconds) for the delay subroutine, assuming the microcontroller is running at 16MHz:

```
DELAY:
    LDI R20, 160
L1:
    LDI R21, 200
L2:
    LDI R22, 250
L3:
    NOP
    NOP
    DEC R22
    BRNE L3
    DEC R21
    BRNE L2
    DEC R20
    BRNE L1
    RET
```

Show your calculations step by step, counting instruction cycles accurately.

Problem 5

The circuit below shows an *Arduino UNO* connected to two switches and two LEDs:



Write a complete assembly program that implements the following functionality:

- Correctly configures the I/O pins (PB0, PB1 as inputs with pull-up resistors; PD4, PD5 as outputs)
- When Switch 1 is pressed (closed state, LOW signal), LED 1 turns ON
- When Switch 1 is not pressed (open state, HIGH signal), LED 1 turns OFF
- Similarly, when Switch 2 is pressed (closed state, LOW signal), LED 2 turns ON
- When Switch 2 is not pressed (open state, HIGH signal), LED 2 turns OFF

This page is intentionally left blank for writing code

Problem 6

Using the 7-segment display pinout provided in the appendix:

Write a complete assembly program that displays your student ID one digit at a time on a common cathode 7-segment display connected to PORTD.

For example, if your student ID is *6622772422*, the program should display each digit sequentially with a delay between digits and loop continuously.

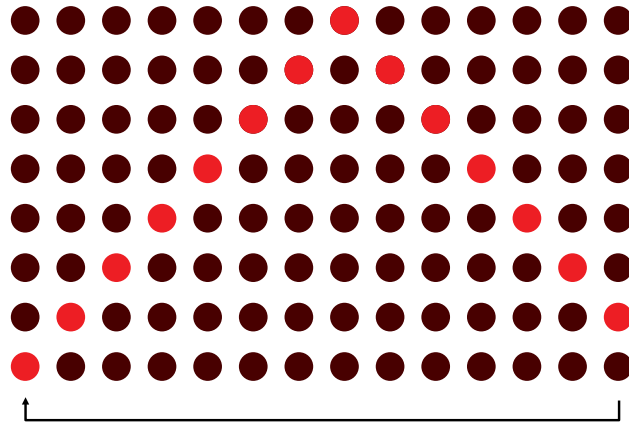


Your student ID has 10 digits. If you store these digits in program memory using the `.DB` directive, explain whether or not the padding effect will occur. Would you need to add an extra byte? Why or why not?

This page is intentionally left blank for writing code

Problem 7

Write a complete assembly program using macros to create a LED sequence on 8 LEDs connected to PORTD (PD0-PD7). The LEDs should light up in a pattern where a single LED appears to move back and forth across the row of LEDs.



Use the following macros in your program:

```
.MACRO DELAY
    LDI R20, @0
L1: LDI R21, @1
L2: LDI R22, @2
L3: NOP
    NOP
    DEC R22
    BRNE L3
    DEC R21
    BRNE L2
    DEC R20
    BRNE L1
.ENDMACRO

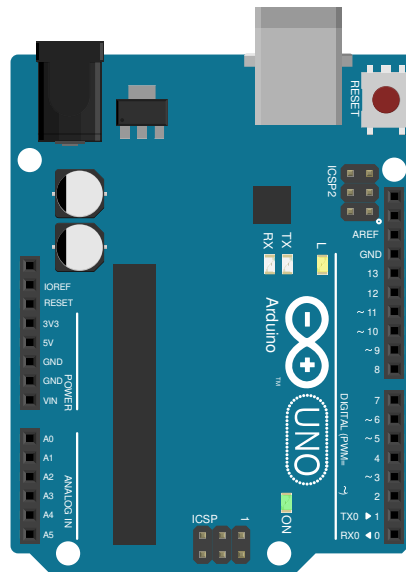
.MACRO SETLED
    LDI R16, @0
    OUT PORTD, R16
.ENDMACRO
```

This page is intentionally left blank for writing code

Appendix

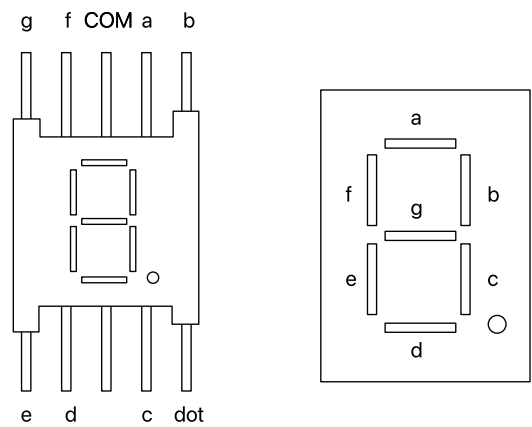
Arduino UNO Pin Layout

| Label | Port |
|-------|-----------|
| A0 | PC0(ADC0) |
| A1 | PC1(ADC1) |
| A2 | PC2(ADC2) |
| A3 | PC3(ADC3) |
| A4 | PC4(ADC4) |
| A5 | PC5(ADC5) |



| Label | Port |
|-------|---------------|
| SCL | PC5(ADC5/SCL) |
| SDA | PC4(ADC4/SDA) |
| AREF | AREF |
| GND | GND |
| 13 | PB5(SCK) |
| 12 | PB4(MISO) |
| 11 | PB3(MOSI) |
| 10 | PB2(OC1B) |
| 9 | PB1(OC1A) |
| 8 | PB0 |
| 7 | PD7 |
| 6 | PD6 |
| 5 | PD5 |
| 4 | PD4 |
| 3 | PD3(INT1) |
| 2 | PD2(INT0) |
| 1 | PD1(TXD) |
| 0 | PD0(RXD) |

7-Segment Display Pinout



| Number | g | f | e | d | c | b | a |
|--------|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| 2 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 3 | 1 | 0 | 0 | 1 | 1 | 1 | 1 |
| 4 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 5 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 6 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 7 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 8 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 9 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |

| Mnemonics | Operands | Description | Operation | Flags | #Clock |
|--|----------|--------------------------|----------------------------------|-----------|--------|
| Arithmetic and Logic Instructions | | | | | |
| ADD | Rd, Rr | Add two registers | $Rd \leftarrow Rd + Rr$ | Z,C,N,V,H | 1 |
| ADC | Rd, Rr | Add with carry | $Rd \leftarrow Rd + Rr + C$ | Z,C,N,V,H | 1 |
| ADIW | Rdl, K | Add immediate to word | $Rdh:Rdl \leftarrow Rdh:Rdl + K$ | Z,C,N,V,S | 2 |
| SUB | Rd, Rr | Subtract registers | $Rd \leftarrow Rd - Rr$ | Z,C,N,V,H | 1 |
| SUBI | Rd, K | Subtract immediate | $Rd \leftarrow Rd - K$ | Z,C,N,V,H | 1 |
| SBC | Rd, Rr | Subtract with carry | $Rd \leftarrow Rd - Rr - C$ | Z,C,N,V,H | 1 |
| SBCI | Rd, K | Subtract with carry imm. | $Rd \leftarrow Rd - K - C$ | Z,C,N,V,H | 1 |
| SBIW | Rdl, K | Subtract imm. from word | $Rdh:Rdl \leftarrow Rdh:Rdl - K$ | Z,C,N,V,S | 2 |
| AND | Rd, Rr | Logical AND | $Rd \leftarrow Rd \times Rr$ | Z,N,V | 1 |
| ANDI | Rd, K | AND with immediate | $Rd \leftarrow Rd \times K$ | Z,N,V | 1 |
| OR | Rd, Rr | Logical OR | $Rd \leftarrow Rd \vee Rr$ | Z,N,V | 1 |
| ORI | Rd, K | OR with immediate | $Rd \leftarrow Rd \vee K$ | Z,N,V | 1 |
| EOR | Rd, Rr | Exclusive OR | $Rd \leftarrow Rd \oplus Rr$ | Z,N,V | 1 |
| COM | Rd | One's complement | $Rd \leftarrow 0xFF - Rd$ | Z,C,N,V | 1 |
| NEG | Rd | Two's complement | $Rd \leftarrow 0x00 - Rd$ | Z,C,N,V,H | 1 |
| INC | Rd | Increment | $Rd \leftarrow Rd + 1$ | Z,N,V | 1 |
| DEC | Rd | Decrement | $Rd \leftarrow Rd - 1$ | Z,N,V | 1 |
| Branch Instructions | | | | | |
| RJMP | k | Relative jump | $PC \leftarrow PC + k + 1$ | None | 2 |
| JMP | k | Direct jump | $PC \leftarrow k$ | None | 3 |
| RCALL | k | Relative call | $PC \leftarrow PC + k + 1$ | None | 3 |
| CALL | k | Direct call | $PC \leftarrow k$ | None | 4 |
| RET | | Return from subroutine | $PC \leftarrow STACK$ | None | 4 |