



**THE UNIVERSITY
OF QUEENSLAND**
A U S T R A L I A

This exam paper must not be removed from the venue

Venue _____

Seat Number _____

Student Number

Family Name _____

First Name _____

School of Information Technology and Electrical Engineering

Semester One Examinations, 2022

CSSE2010/7201 Introduction to Computer Systems

This paper is for St Lucia Campus students.

Examination Duration: 120 minutes

Planning Time: 10 minutes

Exam Conditions:

- This is an Open Book examination
- Casio FX82 series or UQ approved or labelled calculator only
- During Planning Time - Students are encouraged to review and plan responses to the exam questions
- This examination paper will be released to the Library

Materials Permitted in the Exam Venue:

(No electronic aids are permitted e.g. laptops, phones)

* Open-book: Any additional written or printed material is permitted; material may also be annotated.

Materials to be supplied to Students:

Additional exam materials (e.g. answer booklets, rough paper) will be provided upon request.

None

Instructions to Students:

If you believe there is missing or incorrect information impacting your ability to answer any question, please state this when writing your answer.

For Examiner Use Only

Question Mark

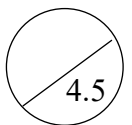
| | |
|----|--|
| Q1 | |
| Q2 | |
| Q3 | |
| Q4 | |
| Q5 | |
| Q6 | |
| Q7 | |

Total _____

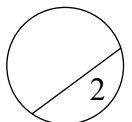
Question 1.**(12 marks)**

- (a) Fill in the following table to complete different number representations of a given number. Each row corresponds to one number which is represented across different formats in each column for that row. There are 9 blanks to be filled. (4.5 marks)

| Decimal | 8-bit signed binary in signed-magnitude | 8-bit signed binary in 2's complement | 8-bit signed binary in excess 128 |
|---------|--|--|--------------------------------------|
| | | | 00110110 |
| | 10011101 | | |
| | | 10100111 | |


 4.5

- (b) Express the 4-digit octal number 3475 in 6-digit quaternary (i.e., base 4) system without going through the decimal system. i.e., perform direct conversion from base-8 to base-4 via binary. Show your working. (2 marks)


 2

(c) Consider the binary two's complement signed numbers $A=11000111$ (given in **8 bits**) and $B=110011$ (given in **6 bits**).

(i) What is the result of $A+B$ in **8-bit binary two's complement** format? Show your working. Here, $+$ indicates addition. (2 marks)

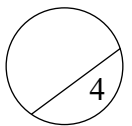
(ii) What are the values (0 or 1) of negative (N), zero (Z), carry (C) and overflow (V) flags after completing the above computation in (c)-(i)? (2 marks)

N:

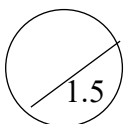
Z:

C:

V:



(d) What is the (decimal) interpretation of the IEEE single-precision floating point number $0xFF800000$? Show your working. (1.5 marks)



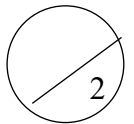
Question 2.**(14 marks)**

- (a) Consider a combinational logic circuit having three 1-bit inputs A, B, C and one 1-bit output F. The output F is high if and only if the number represented by ABC in unsigned binary format is greater than zero AND is exactly divisible by 2 or 3 (i.e., results in a non-zero quotient and 0 remainder). Here, A is the most significant bit and C is the least significant bit.

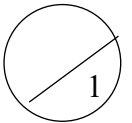
- (i) Complete the truth table below for this circuit.

(2 marks)

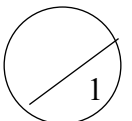
| A | B | C | F |
|---|---|---|---|
| 0 | 0 | 0 | |
| 0 | 0 | 1 | |
| 0 | 1 | 0 | |
| 0 | 1 | 1 | |
| 1 | 0 | 0 | |
| 1 | 0 | 1 | |
| 1 | 1 | 0 | |
| 1 | 1 | 1 | |



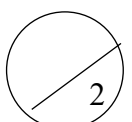
- (ii) Provide a sum-of-product (SOP) Boolean expression for F, where each product term contains all three literals A, B, and C (i.e., an un-simplified SOP expression). (1 mark)



- (iii) Simplify the above un-simplified SOP expression for F (i.e., obtain the minimal SOP) using Boolean algebra laws and show your steps for simplification. (1 mark)

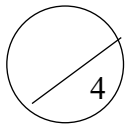


- (iv) Provide a logic diagram for the simplified Boolean expression for F obtained above in (a)-(iii) only using 2-input NOR gates. (2 marks)



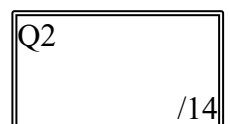
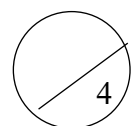
(b) The following questions are related to a full-adder circuit having 1-bit inputs A, B, C_{in} and 1-bit outputs S and C_{out} .

- (i) Provide a logic circuit for each output S and C_{out} **only using two 4:1 multiplexers and one 2:1 multiplexer** for each output. You cannot use any gates or other circuit blocks and you should use the standard multiplexer symbol. (4 marks)



- (ii) Provide a logic circuit for each output S and C_{out} **only using a 3:8 decoder and 2-input AND/OR gates**. Other circuit blocks cannot be used.

(4 marks)



Question 3.**(20 marks)**

- (a) Using the Moore type finite state machine (FSM) design approach, provide the design of a synchronous binary counter, which counts through the following sequences based on an external input E.

| Input E | Counting sequence |
|---------|------------------------------------|
| 1 | 0→2→4→6→0 (i.e., even up-counting) |
| 0 | 1→3→5→7→1 (i.e., odd up-counting) |

Following rules apply for this circuit.

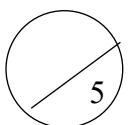
- The initial/reset state of the system is the state where all outputs are 0.
- From the initial state, the input E selects between even or odd counting. If the input E=0 at the initial state, in the next clock edge, the output becomes 1, thus starting the odd counting from the next clock edge. If E=1 at the initial state, in the next clock edge, the output becomes 2, thus continuing in the even counting.
- During counting, if the input E changes, the system goes to the first count in the appropriate sequence indicated by current value of E. For example, if E changes from 1 to 0 at count value 4 during even counting, in the next clock edge, the system's output should become 1, thus starting the odd counting.
- The system has an asynchronous reset which you do not need to show in the design.
- If you make any other assumptions state them explicitly.

Show all the design steps including state diagram, state table with suitable encoding, Boolean expressions (simplified) for next state and output logic and logic diagram. You should not use more than 3 flip flops. (15 marks)

- (b) Using a logic diagram, show the design of a 3-bit universal shift register which combines the following four modes of operations based on the control inputs F_1 (MSB) and F_0 (LSB).

| F_1 | F_0 | Mode of operation |
|-------|-------|-------------------------------|
| 0 | 0 | Parallel load |
| 0 | 1 | Stay same (no change) |
| 1 | 0 | Arithmetic shift to the right |
| 1 | 1 | Rotate to the left |

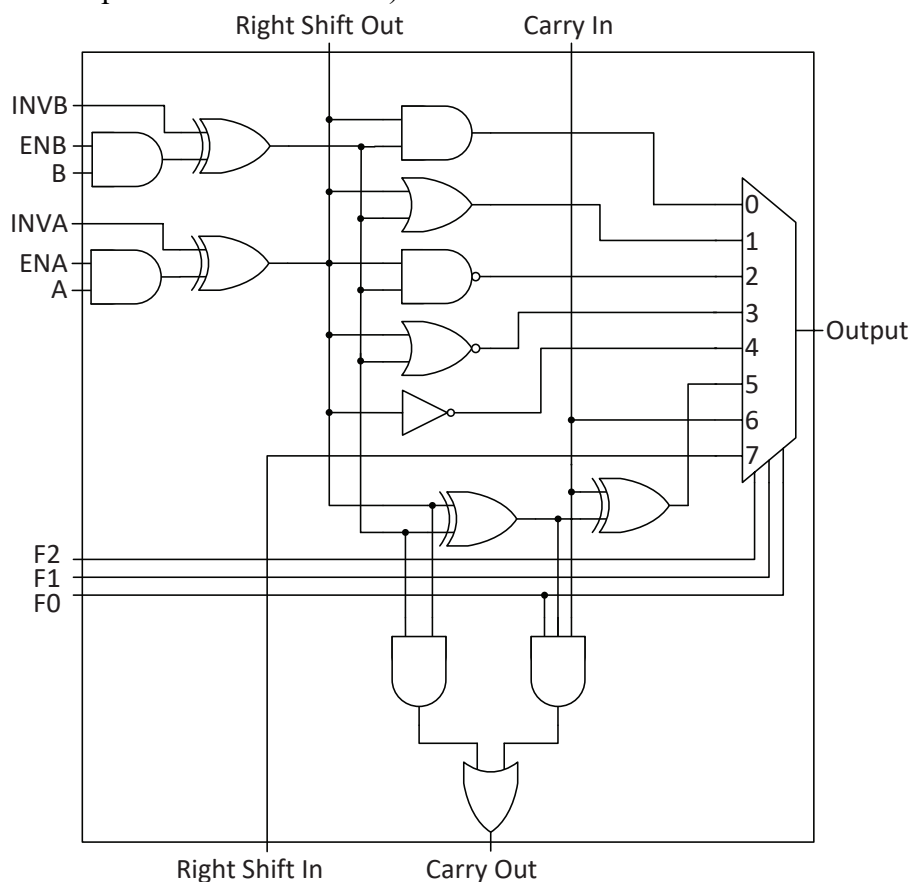
(5 marks)



Q3
/20

Question 4.**(8 marks)**

Consider the ALU bit slice shown below (note that for the output multiplexer on the right hand side, F2 is the most significant of the select bits, so for example, if F2,F1,F0 = 0,0,1 then input 1 of the multiplexer will be selected).



Consider 8 of these ALU bit slices put together to form an 8-bit ALU (i.e., with 8-bit data inputs A and B). Complete the following table to show the required control inputs for the ALU to perform the given functions. (Each control input will have value 0 or 1 or X (don't care)). The "carry in" control input applies only to the least significant bit, the "carry in" input of other bit slices comes from the "carry out" output of its neighbouring bit slice. The "right shift in" control input applies only to the most significant bit; the "right shift in" input of other bit slices comes from the "right shift out" output of its neighbouring bit slice.) If it is not possible to generate the given function, make a comment to this effect below the table. If there is more than one way to generate the given function, just show one way.

(2 marks each)

| Description of Function Output | ENA | INVA | ENB | INVB | Carry In | Right Shift In | F2 | F1 | F0 |
|---------------------------------------|-----|------|-----|------|----------|----------------|----|----|----|
| (i) Constant value (-128) | | | | | | | | | |
| (ii) A - B | | | | | | | | | |
| (iii) 2*B (multiplication by 2) | | | | | | | | | |
| (iv) $\bar{A} + \bar{B}$ (bitwise OR) | | | | | | | | | |

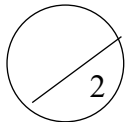
Q4

Question 5.**(16 marks)**

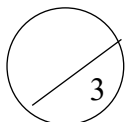
This question applies to the AVR ATmega324A microcontroller and associated assembly language instructions.

(a) Provide AVR assembly language code segments to implement the following operations with the given constraints. You can use any general-purpose register r0-r31 as needed.

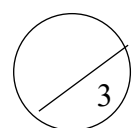
- (i) Initialise a 16-bit signed two's complement variable stored in the X register to (-6) in 2 CPU cycles or less. Here, r27 stores the most significant byte. (2 marks)



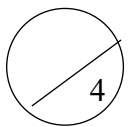
- (ii) Two 16-bit signed two's complement numbers A and B have been initialised and stored in r3:r4 and r9:r10, respectively. Compute $(A + 2*B)$ in 4 CPU cycles or less. Here, r3 contains the most significant byte of A and r9 contains the most significant byte of B and the 16-bit result is stored back in r3:r4. (3 marks)



- (iii) Two 8-bit signed two's complement numbers A and B are stored in r16 and r17, respectively. If A is greater than or equal to B then make r18=0, otherwise make r18=0xFF. (3 marks)



- (iv) Two 8-bit signed two's complement numbers A and B are stored in r5 and r6, respectively. Initialise the 8-bit number C stored in r8 such that the least significant 4 bits of C become bits 1-4 of A and the most significant 4 bits of C become bits 3-6 of B, where bits are numbered from 0-7. That is, if $A = a_7a_6a_5a_4a_3a_2a_1a_0$ and $B = b_7b_6b_5b_4b_3b_2b_1b_0$ then make $C = b_6b_5b_4b_3a_4a_3a_2a_1$. Original A and B should not be changed. (4 marks)



- (b) Consider the assembly language code segment given below

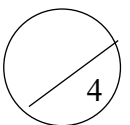
```

        .def index=r16
        .def maxindex=r17
        .def sum=r20
        .def avg=r21
loop:   ldi maxindex,17
        ldi index,1
        clr sum
        clr avg
        cp index, maxindex
        brge exit
        add sum, index
        inc index
        jmp loop
exit:   mov avg, sum
        asr avg
        asr avg
        asr avg
        asr avg

```

- (i) How many **bytes** in program memory does the above code occupy? (1 mark)
- (ii) How many clock cycles (CPU cycles) does the above code take to execute? (2 marks)

- (iii) How can you improve the execution time (i.e., reduce the number of CPU cycles) of the above code? Provide one suggestion. (1 mark)

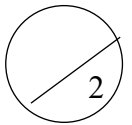


Question 6.**(15 marks)**

- (a) For each of the following C statements for the Atmel AVR ATmega324A, write down the assembly language equivalent. (You may assume that definitions in the m324Adef.inc file are available. Several instructions may be required. You can freely use general purpose registers).

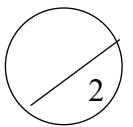
(i) `DDRC |= 0x5E;`

(2 marks)



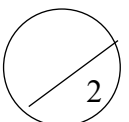
(ii) `PORTB = TCNT2;`

(2 marks)



(iii) `OCR1B = OCR1A;`

(2 marks)



The following is extracted from Section 15.9 of the ATmega324A datasheet. You will need to refer to this information when answering this question.

15.9.1 TCCR0A – Timer/Counter Control Register A

| | | | | | | | | | |
|---------------|--------|--------|--------|--------|---|---|-------|-------|--------|
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| 0x24 (0x44) | COM0A1 | COM0A0 | COM0B1 | COM0B0 | – | – | WGM01 | WGM00 | TCCR0A |
| Read/Write | R/W | R/W | R/W | R/W | R | R | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

• Bits 7:6 – COM0A1:0: Compare Match Output A Mode

These bits control the Output Compare pin (OC0A) behavior. If one or both of the COM0A1:0 bits are set, the OC0A output overrides the normal port functionality of the I/O pin it is connected to. However, note that the Data Direction Register (DDR) bit corresponding to the OC0A pin must be set in order to enable the output driver.

When OC0A is connected to the pin, the function of the COM0A1:0 bits depends on the WGM02:0 bit setting. Table 15-2 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to a normal or CTC mode (non-PWM).

Table 15-2. Compare Output mode, non-PWM mode

| COM0A1 | COM0A0 | Description |
|--------|--------|--|
| 0 | 0 | Normal port operation, OC0A disconnected |
| 0 | 1 | Toggle OC0A on Compare Match |
| 1 | 0 | Clear OC0A on Compare Match |
| 1 | 1 | Set OC0A on Compare Match |

Table 15-3 shows the COM0A1:0 bit functionality when the WGM01:0 bits are set to fast PWM mode.

Table 15-3. Compare Output mode, Fast PWM mode ⁽¹⁾

| COM0A1 | COM0A0 | Description |
|--------|--------|--|
| 0 | 0 | Normal port operation, OC0A disconnected. |
| 0 | 1 | WGM02 = 0: Normal Port Operation, OC0A Disconnected. WGM02 = 1: Toggle OC0A on Compare Match. |
| 1 | 0 | Clear OC0A on Compare Match, set OC0A at BOTTOM, (non-inverting mode). |
| 1 | 1 | Set OC0A on Compare Match, clear OC0A at BOTTOM, (inverting mode). |

• Bits 5:4 – COM0B1:0: Compare Match Output B mode

These bits control the Output Compare pin (OC0B) behavior. If one or both of the COM0B1:0 bits are set, the OC0B output overrides the normal port functionality of the I/O pin it is connected to. However, note that the Data Direction Register (DDR) bit corresponding to the OC0B pin must be set in order to enable the output driver.

When OC0B is connected to the pin, the function of the COM0B1:0 bits depends on the WGM02:0 bit setting. Table 15-2 on page 109 shows the COM0A1:0 bit functionality when the WGM02:0 bits are set to a normal or CTC mode (non-PWM).

Table 15-5. Compare Output mode, non-PWM mode

| COM0B1 | COM0B0 | Description |
|--------|--------|---|
| 0 | 0 | Normal port operation, OC0B disconnected. |
| 0 | 1 | Toggle OC0B on Compare Match |
| 1 | 0 | Clear OC0B on Compare Match |
| 1 | 1 | Set OC0B on Compare Match |

Table 15-6 shows the COM0B1:0 bit functionality when the WGM02:0 bits are set to fast PWM mode.

Table 15-6. Compare Output mode, Fast PWM mode ⁽¹⁾

| COM0B1 | COM0B0 | Description |
|--------|--------|--|
| 0 | 0 | Normal port operation, OC0B disconnected. |
| 0 | 1 | Reserved |
| 1 | 0 | Clear OC0B on Compare Match, set OC0B at BOTTOM, (non-inverting mode). |
| 1 | 1 | Set OC0B on Compare Match, clear OC0B at BOTTOM, (inverting mode). |

• **Bits 1:0 – WGM01:0: Waveform Generation mode**

Combined with the WGM02 bit found in the TCCR0B Register, these bits control the counting sequence of the counter, the source for maximum (TOP) counter value, and what type of waveform generation to be used, see [Table 15-8 on page 111](#). Modes of operation supported by the Timer/Counter unit are: Normal mode (counter), Clear Timer on Compare Match (CTC) mode, and two types of Pulse Width Modulation (PWM) modes (see "Modes of Operation" on page 127).

Table 15-8. Waveform Generation mode bit description

| Mode | WGM2 | WGM1 | WGM0 | Timer/Counter mode of operation | TOP | Update of OCRx at | TOV Flag set on ^{(1)/(2)} |
|------|------|------|------|---------------------------------|------|-------------------|------------------------------------|
| 0 | 0 | 0 | 0 | Normal | 0xFF | Immediate | MAX |
| 1 | 0 | 0 | 1 | PWM, Phase Correct | 0xFF | TOP | BOTTOM |
| 2 | 0 | 1 | 0 | CTC | OCRA | Immediate | MAX |
| 3 | 0 | 1 | 1 | Fast PWM | 0xFF | BOTTOM | MAX |
| 4 | 1 | 0 | 0 | Reserved | – | – | – |
| 5 | 1 | 0 | 1 | PWM, Phase Correct | OCRA | TOP | BOTTOM |
| 6 | 1 | 1 | 0 | Reserved | – | – | – |
| 7 | 1 | 1 | 1 | Fast PWM | OCRA | BOTTOM | TOP |

Notes: 1. MAX = 0xFF
2. BOTTOM = 0x00

15.9.2 TCCR0B – Timer/Counter Control Register B

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---------------|-------|-------|---|---|-------|------|------|------|--------|
| 0x25 (0x45) | FOC0A | FOC0B | – | – | WGM02 | CS02 | CS01 | CS00 | TCCR0B |
| Read/Write | W | W | R | R | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

Table 15-9. Clock Select bit description

| CS02 | CS01 | CS00 | Description |
|------|------|------|---|
| 0 | 0 | 0 | No clock source (Timer/Counter stopped) |
| 0 | 0 | 1 | clk _{IO} /(No prescaling) |
| 0 | 1 | 0 | clk _{IO} /8 (From prescaler) |
| 0 | 1 | 1 | clk _{IO} /64 (From prescaler) |
| 1 | 0 | 0 | clk _{IO} /256 (From prescaler) |
| 1 | 0 | 1 | clk _{IO} /1024 (From prescaler) |
| 1 | 1 | 0 | External clock source on T0 pin. Clock on falling edge. |
| 1 | 1 | 1 | External clock source on T0 pin. Clock on rising edge. |

15.9.3 TCNT0 – Timer/Counter Register

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---------------|------------|-----|-----|-----|-----|-----|-----|-----|-------|
| 0x26 (0x46) | TCNT0[7:0] | | | | | | | | TCNT0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

The Timer/Counter Register gives direct access, both for read and write operations, to the Timer/Counter unit 8-bit counter. Writing to the TCNT0 Register blocks (removes) the Compare Match on the following timer clock. Modifying the counter (TCNT0) while the counter is running, introduces a risk of missing a Compare Match between TCNT0 and the OCR0x Registers.

15.9.4 OCR0A – Output Compare Register A

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---------------|------------|-----|-----|-----|-----|-----|-----|-----|-------|
| 0x27 (0x47) | OCR0A[7:0] | | | | | | | | OCR0A |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

The Output Compare Register A contains an 8-bit value that is continuously compared with the counter value (TCNT0). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC0A pin.

15.9.5 OCR0B – Output Compare Register B

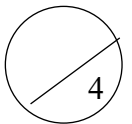
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---------------|------------|-----|-----|-----|-----|-----|-----|-----|-------|
| 0x28 (0x48) | OCR0B[7:0] | | | | | | | | OCR0B |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W | |
| Initial Value | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

The Output Compare Register B contains an 8-bit value that is continuously compared with the counter value (TCNT0). A match can be used to generate an Output Compare interrupt, or to generate a waveform output on the OC0B pin.

(b) Using the above information for Timer/Counter 0 of Atmega324A, answer the following questions.

- (i) Using the OC0A pin (Port B pin 3) under clear timer on compare match (CTC) mode, provide a C code segment to configure the relevant GPIO port and the Timer/Counter 0 module, to toggle Port B pin 3 at every 10 milliseconds (i.e., is ON for 10ms and then OFF for 10ms and so on). The microcontroller clock frequency is 8MHz.

(4 marks)

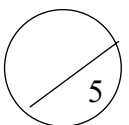


- (ii) It is required to control the average voltage delivered to a load using PWM with the following specifications. The microcontroller clock frequency is 8 MHz.

- Fast PWM mode of Timer/Counter 0 with PWM frequency of 31.25 KHz
- Supply voltage (V_{cc}) is 5V and the average voltage to be delivered to the load is 3.75 V (i.e., 75% of V_{cc})
- You can use either OC0A (PB3) or OC0B (PB4) as the PWM output pin

Provide a C code segment to configure the necessary GPIO port pin and Timer/Counter 0 module to obtain a PWM signal to achieve the above specifications. Your C code segment should include comments to explicitly mention details such as PWM mode, pre-scalar value, output compare register values, GPIO pin used.

(5 marks)



Question 7.**(15 marks)**

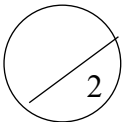
- (a) Consider the following assembly language code for Atmega324A and answer the questions below in relation to the two-pass assembly process

```
jmp RESET
jmp ISR1
jmp ISR2

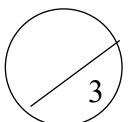
.dseg
var1: .BYTE 2
var2: .BYTE 2
var3: .BYTE 2

.cseg
.def temp=r16
table1: .DB 10,115,75,141,0
table2: .DB 60,35,86,8,39,46
table3: .DW var1, var2, var3
RESET:
ldi temp, low(RAMEND)
out SPL, temp
ldi temp, high(RAMEND)
out SPH, temp
```

- (i) What is the code and data segment sizes in **bytes** for the above module? (2 marks)



- (ii) If the code segment and data segment location counters start from 0 and 256, respectively, provide a symbol table to show the memory addresses corresponding to the labels RESET, var1, var2, var3, table1, table2, table3 (3 marks)

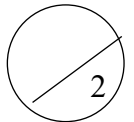


(b) A Western Digital Ultrastar hard drive has the following specifications:

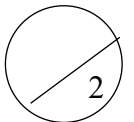
| | |
|------------------------------|------------------------------|
| Nominal Capacity: | 4TB (1TB = 10^{12} bytes) |
| Sector size: | 4096 bytes |
| Rotational speed: | 7200 RPM |
| Max Sustained transfer rate: | 255MB/s (1MB = 10^6 bytes) |
| Average seek time (typical): | 8.2ms (read) / 8.6ms (write) |

The hard drive is formatted with a file system that has a block size of 16kB (16,384 bytes) and currently contains 1,200,000 files of various sizes which occupy 100,000,000 data blocks.

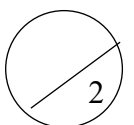
(i) What is the average access time when reading from the disk? (Show your working.)
(2 marks)



(ii) How much total space within the 100,000,000 allocated data blocks is likely to be wasted (i.e., not used for storing data)? (Show your working. Express your answer in kB, where 1kB=1024 bytes).
(2 marks)



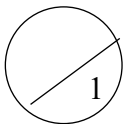
(c) In a UART communication scenario where a transmitter sends 7 bits of data, with 1 start bit, 1 stop bit, odd parity bit at baud rate of 19,200 bauds/sec, what is the associated bit rate in this serial communication? Show your working
(2 marks)



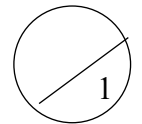
- (d) In a certain system a CPU is interfaced with a memory having 2^{24} bytes with an addressable cell size of 8 bits. A selected content of memory is shown in the table below

| Address in decimal | Content in Hex |
|--------------------|----------------|
| 400 | 3A |
| 401 | 23 |
| 402 | 5A |
| 403 | 3D |
| 404 | 2F |
| 405 | 1E |

- (i) If the data bus is 16 bits wide what is the size of the address bus? (1 mark)

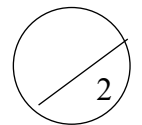


- (ii) If 2 bytes are read at address 402 in little-endian convention and the resulting 16-bit quantity is stored at address 404 in big-endian convention, provide the content of memory locations 400-405 above after these operations. (1 marks)



- (iii) If the memory content in the above table refers to a stack where the stack pointer (SP) is 400 and the stack grows towards decreasing memory addresses, what is the **value of the SP and the content of the memory cell at the top of the stack** after the following operations are carried out. Note that the statements below are not given in correct assembly syntax and just indicate the operations. (2 marks)

PUSH 0x22; comment: push value 0x22 on to the stack
 PUSH 0x56; comment: push value 0x56 on to the stack
 POP



For scratch work

For scratch work

END OF EXAMINATION