

Problem 3. (35 pts) The `Subr` subroutine below handles a circular queue, which occupies up to 20h bytes starting at offset 0300h in the Data Segment. Each queue item is one byte. The `Deq` subroutine dequeues the item at the front of the queue and places it into memory at byte `QData`. The `Enq` subroutine enqueues the item at byte `QData` onto the rear of the queue.

`Subr`:

	Offset in	Value
	Stack	of
	Segment	Word
<code>CMP WORD[QCount],0 ; Check for</code>		
<code>JE .done ; empty queue</code>		
<code>PUSH BP</code>		
<code>PUSH AX</code>		
<code>SUB SP,2 ; Space for local</code>	01C4	2B0F
<code>MOV BP,SP ; variable</code>	01C6	409A
<code>CALL Deq ; Remove first item</code>	01C8	1793
<code>MOV AL,[QData] ; from queue and</code>	01CA	01CE
<code>MOV [BP],AL ; save on stack</code>	01CC	2B0F
<code>CALL Subr ; Operate on rest of queue</code>	01CE	5093
<code>MOV AL,[BP] ; Append old first</code>	01D0	1788
<code>MOV [QData],AL ; item onto the</code>	01D2	01D6
<code>CALL Enq ; current queue</code>	01D4	2B0F
<code>ADD SP,2</code>	01D6	6088
<code>POP AX</code>	01D8	1776
<code>POP BP</code>	01DA	01F0
<code>.done:</code>	01DC	1406
<code>RET</code>	01DE	1867

(a) Suppose that just before the first call to `Subr`, `SP=01DE`, and the original front item of the queue is at offset 031Eh. Shown above are the hexadecimal values of some words in the Stack Segment immediately before the first execution of `RET`. At this time, `SP=01C4`, and an interrupt service routine for some device has just finished. Determine the following hexadecimal values; except for part (iv), write “Unknown” when there is insufficient information:

(i) Offset in the Code Segment of the 3-byte `CALL Subr` instruction in `Subr`: _____

(ii) Original value of word in memory labeled `QCount`: _____

(iii) Original values of bytes at offsets 031D, 031E, 031F, 0320 (hex) in the Data Segment:

DS:031D _____ DS:031E _____ DS:031F _____ DS:0320 _____

(iv) A **possible** value for the word at `SS:01C2` at this time (not “Unknown”): _____

(v) Values of bytes at offsets 0300, 0301, 0302, and 0303 (hex) in the Data Segment when the **original** call to `Subr` finishes (and `SP=01DE` again):

DS:0300 _____ DS:0301 _____ DS:0302 _____ DS: 0303: _____

(b) What undesirable behavior occurs if the first two instructions (`CMP` and `JE`) are omitted?

Problem 3, continued. (c) Explain clearly the overall effect of $Subr$ on the queue, in general.

Problem 4. (30 pts) Assume that the computer is connected to only three devices, D1, D2, and D3, which send interrupt signals periodically. There are no other external interrupt sources. All interrupt service routines are preemptive: very shortly after each routine begins, it executes an STI instruction. Assume that the average execution time for one instruction is 0.5 microsecond.

(a) Some information about the devices and their service routines is given below. Complete the table; write “Unknown” when there is insufficient information.

Device	D1	D2	D3
Interrupt type	0Ah (high priority)	0Bh	0Ch (low priority)
Service load	8 %	_____ %	12%
Interrupt rate	_____ Hz	50 Hz	_____ Hz
Service time (milliseconds)	_____ ms	4 ms	3 ms
Worst case interrupt latency	_____ ms	8 ms	_____ ms

(b) Determine the average number of instructions executed per second by background tasks (that is, outside the three interrupt service routines).

(c) You plan to attach a fourth device D4 whose interrupt rate is 400 Hz. Explain why D4 should **not** be assigned the lowest priority.

Problem 5. (30 pts) Write your answers in clear English.

(a) A student testing Machine Problem #3 types the input string “21/3)”, and the program outputs “-21838”. Based on this test, identify three possible logical errors in the MainLoop and Term subroutines.

1. _____

2. _____

3. _____

(b) Find two errors in the following implementation of the timer interrupt service routine in Machine Problem #4. For each error, explain why it is an error, and how you would correct it.

TimerInt:

```
    ADD WORD [Ticks], 1
    RET
```

1. _____

2. _____

(c) In Machine Problem #4, suppose Task1 does **not** set the memory word labeled `Ticks` to 0 when the user presses the space bar. Describe what unexpected behavior might happen, and why.

