ORG test 2 13 NOV 85 by _____

1-4 10 pts each 5-8 & 9 15 pts each Good Luck!

1. In an 8-bit machine $3C_{16}$ is subtracted from BD_{16}. Give the contents of the flags C, S, V, Z and PE (parity even) after this operation.

2. AB. Formulate a mapping process from (macro) op-codes to (μ) control memory addresses given that μ-memory has 1K words, op-codes are 6 bits wide and 4 control words are needed for each macro-op

C. Illustrate your map with the op-code 42_{10}, give your address in binary.

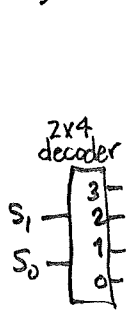
D. What is the size of the CAR?

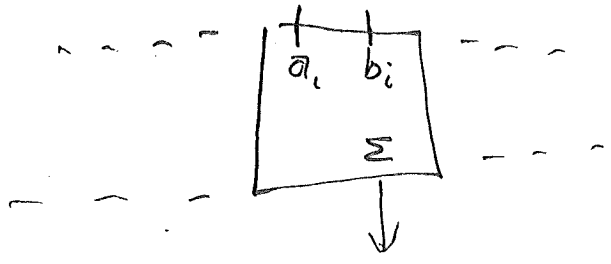
3. AB. The text divides all interrupts into 3 classes. Give these 3 classes and an example of each.

C. What is an interrupt vector and how is it used?

4. Design a circuit for ALU which yields (Actually we need only one "stage" of the logic which is "fed" into a parallel adder.) The decoder will ease the pain

S_1S_0	$C_{in}=0$	$C_{in}=1$
00	A	A+1
01	A+B	A+B+1
10	A+$\overline{B}$	A+$\overline{B}$+1
11	B-1	B





5 (μ-code) The Jazzed Up Chapter 8 computer is improved to include SP (stack pointer register which points to TOS and "grows" toward low memory) and the μ-code format is expanded to include F4 (see →) Op codes 9 and 12 are for CALL and RETURN. Write these in the Ch.8 style, give ORG and worry about effective addresses

A CALL

B RETURN

F4	
0	NOP
1	SPTAR
2	INCSP
3	DECSP
4	BRTSP
5	PCTSP
6	SPTPC
7	———

6. (addressing modes) The test 2 computer is a 13-bit machine with four 13-bit registers (cleverly named) R0, R1, R2 & R3 The general format for an instruction is

op-code 7 = move	source mode	source reg.	destination mode	dest. reg.
3	3	2	3	2

op-code '7' is for "move" or more correctly "copy". The "modes" are as follows note for mode = 0 the reg. field doesn't pick a reg but the reg field picks one of the "ext. modes"

reg (memory reference)	
0	immediate
1	direct
2	indirect
3	relative

mode	
0	memory reference
1	reg direct
2	reg indirect
3	index
4	pre-decrement
5	post-increment

Given the instruction at M[0] is the one given below and M[1] contains 'x', M[2] contains 'y' (where addition constants and/or addresses are found) Write the effect of each instruction "á la" M[M[R0+x+M[4]]] ← R2+y"

	OP	SM	SR	DM	DR
A	7	0	0	1	3
B	7	2	2	0	1
C	7	0	3	4	1
D	7	5	0	0	2
E	7	3	2	3	3

(μ-ops) ORG/T2/P3

7.1 This version of the chapter 5 computer has both an SP (points to TOS, grows toward low memory) and an FP (a "frame pointer"). The LINK ass-instruction reserves some stack space (for subroutine variables). The amount of space is given by the address field (it's a memory reference instruction) It does PUSH FP, FP←SP, SP←SP+constant. UNLINK undoes this (but it doesn't know the "constant" (nor does it need it)) Write the execution cycles for these with controls given they have "op-codes" r & s respectively. (Additional Adders are availible)

A. LINK

B UNLINK

8. (parameter passing via the stack) [SP points to TOS, grows toward low memory]

A compiler would generate the code

```
push 10
push 4
push 1
call P
```

for a procedure call P(1,4,10) if the procedure body is as in the box.

```
Procedure P (A,B,C: integer);
[Local variables]
 begin
  :
 end
```

A. Suppose integers and addresses fit into 1 word of memory. At the begining of P the variables A,B,C are easy to obtain via using the SP as an index register. What are the offsets from SP for Var A? Var B? Var C?

B. If Procedure P has local variables it will call LINK (Prob 7) to create stack space for them. The variables A,B,C are still easy to obtain but this time via indexing from the FP. What are the offsets from FP for Var A? Var B? Var C?

CDE. This stack use allows for procedures to be called with a different number of parameters each time it is called. i.e. push the variables onto the stack in backwards order (right to left) then push the size of the variables onto the stack (i.e. reals take 2 words) then push the number of variables onto the stack and call the routine. Like writeln(integer, real, integer)

C. what is the offset of the first variable from SP in general?

D. what is the offset of the second variable from SP?

E the offset for 3?