Organização de Computadores Digitais I

1ª. Questão: Implemente um programa em linguagem montadora com as instruções da Tabela I que realize o algoritmo de um calendário digital.

a) Deve ter dia , mês e ano.
b) Deve imprimir na Tela na linha 5: D: xx M: xx A: xx
c) O calendário deve começar em D: 01 M: 01 A: 01 e seguir marcando o tempo até que o processador seja desligado.
d) Considere a utilização de uma rotina **outascii** (em anexo) que trabalha com R0 e R1 sem alterá-los e imprime os dois dígitos menos significativos de R1 em decimal da seguinte maneira:
OUTPUT R0 R1(digito 1)
OUTPUT R0+1 R1(digito 2)

Tabela I – Instruções do processador

LOAD Rx #Nr	LOAD Rx END	LOAD Rx Ry	MOV Rx Ry
CMP	STORE END Rx	STORE Ry Rx	ADD Rz Rx Ry
SUB Rz Rx Ry	MUL Rz Rx Ry	DIV Rz Rx Ry	AND Rz Rx Ry
OR Rz Rx Ry	XOR Rz Rx Ry	NOT Rz Rx	NOP
JMP END	CALL END	RTS	
PUSH Rx	POP Rx	CLRC	SETC
INC Rx	DEC Rx	INPUT Rx	OUTPUT Rx Ry
SHIFT Rx	ROTATE Rx		

2ª. Questão: Monte as seguintes 7 linhas de programa. (3bit – registrador)

a) Considere que cada instrução é representada por 6 bits: cmp = 01; jmp = 02; load = 03; input = 04, add = 05, call = 06, and = 07, halt = 08, output =09.
b) Considere que a condição tem 4 bits: **eg** (equal or greater) = 07.
c) Considere que o programa começa na linha 0008 da memória (16 bits).

```
            add r4 r2 r3
            cmp r4 r5
            ceg grande
            jmp digito1
grande:     load r5 #37
digito1:    output r0 r1 r4
            halt
```

3ª. Questão: Explique o algoritmo abaixo

```
; Imprime os dois dígitos menos
; significativos de r2 em HEX
; Trabalha com r0 r1 r2 sem altera-los
; OUTPUT R0   R1 R2(0)
; OUTPUT R0+1 R1 R2(1)
outascii:
                push r0
                push r1
                push r2
                push r3
                push r4
                push r5
                push fr
                ; Digito r2(0) --> HEX
                load r3 #000F
                and r4 r2 r3
                load r5 #000A
                cmp r4 r5
                jeg grande
                load r5 #0030  ;r2 = (0-9)
                bra digito1
grande:         load r5 #0037 ;r2 = (A-F)
digito1:        add r4 r4 r5
                output r0 r1 r4
                ; Digito r2(1) --> HEX
                sr0 r2
                sr0 r2
                sr0 r2
                sr0 r2
                load r3 #000F
                and r4 r2 r3
                load r5 #000A
                cmp r4 r5
                jeg grande2
                load r5 #0030  ;r2 = (0-9)
                bra digito2
grande2:        load r5 #0037 ;r2 = (A-F)
digito2:        add r4 r4 r5
                inc r0
                output r0 r1 r4
                pop fr
                pop r5
                pop r4
                pop r3
                pop r2
                pop r1
                pop r0
                rts
```

Tabela ASCII

Dec	Hex	Char
48	30	0
49	31	1
50	32	2
51	33	3
52	34	4
53	35	5
54	36	6
55	37	7
56	38	8
57	39	9
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F

4ª. Questão: Desenhe um processador de 8 bits, contendo 4 registradores de 8 bits, ULA de 8 bits e memória de 64K palavras de 8 bits, com o mínimo de estruturas necessárias para executar só e somente só as instruções da Tabela abaixo. Considerando um conjunto de 14 instruções, mostre como será a representação de cada instrução na memória. Desenhe o circuito deste processador para as 4 instruções da tabela. Mostre os sinais que a Máquina de Controle deve enviar para os dispositivos para executar as instruções da Tabela nos ciclos de Busca, Decodificação e Execução (adicione mais ciclos se achar necessário).

Instrução	Ação
LOAD Rx Ry	Rx ← MEM[Ry]
PUSH Rx	Rx → PILHA
STORE END Rx	MEM[END] ← Rx
CALLC END	Goto END Se Carry = 1

5ª. Questão: Desenhe um processador de 32 bits, contendo 4 registradores de 32 bits, ULA de 32 bits e memória de 64K palavras de 32 bits, com o mínimo de estruturas necessárias para executar só e somente só a instrução CALL END.