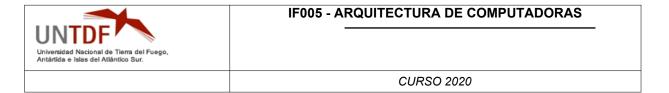
# UNTOF Universidad Nacional de Tierra del Fuego, Antártida e Islas del Atlántico Sur. CURSO 2020

### Práctica 07 - Tema: Unidad de Control

- 1. En el microprograma del Anexo II, asumiendo que un ciclo de memoria dura dos microinstrucciones.
  - a) Qué condición se debe mantener en dos microinstrucciones consecutivas al leer o escribir la memoria?
- 2. Se dice que la mayor parte del tiempo de ejecución de cada instrucción de máquina convencional está dedicado a decodificarla bit por bit. En el microprograma del Anexo II:
  - a) Identifique las sentencias que forman parte del proceso de decodificación de la instrucción
  - b) Para cada instrucción de máquina convencional indique la secuencia de microinstrucciones que forman parte de su proceso de decodificación
  - c) Para cada instrucción de máquina convencional indique la secuencia de microinstrucciones que forman parte de su proceso de ejecución
  - d) Construya una tabla como la que se muestra en el anexo III
- 3. Si las lecturas o escrituras en la memoria principal necesitaran tres ciclos de microinstrucción en lugar de dos.
  - a) Se vería afectado el tamaño del microprograma del Anexo II?
  - b) Si es así, en cuanto?
- 4. Se desea agregar a la máquina convencional, definida en el Anexo I, las instrucciones dadas en el anexo IV. Para ello se le pide que:
  - a) Asigne un código de operación conveniente (debe ser compatible con los que ya están asignados)
  - b) Modifique el microprograma para que acepte estas nuevas instrucciones.
- 5. Para el set de instrucciones de maguina convencional dado en el Anexo I
  - a) Determine que cantidad de instrucciones de maquina convencional, de los siguientes tipos, puede soportar:
    - i) Instrucciones con un operando en memoria
    - ii) Instrucciones con un operando en registro
    - iii) Instrucciones con dos operando en registros
- 6. Determine cuántas instrucciones de los tipos dados en el punto anterior se pueden agregar al repertorio ya existente.
- 7. Agregue al repertorio, una instrucción que permita mover el contenido del acumulador a los registros A al F, sin modificar la secuencia de microinstrucciones del microprograma original.



### Anexo I

Set de Instrucciones de Maquina Convencional

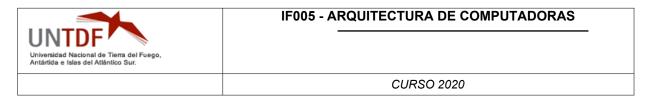
CODIGO DE OPERACIÓN					Descripción	Significado
Binario		Mnem				
0000	xxxx	xxxx	xxxx	LODD	Carga directa	ac:=m[x]
0001	xxxx	xxxx	xxxx	STOD	Almacena directo	m[x]:= ac
0010	xxxx	xxxx	xxxx	ADDD	Suma directo	ac:= ac + m[x]
0011	xxxx	xxxx	xxxx	SUBD	Resta directo	ac:= ac - m[x]
0100	xxxx	xxxx	xxxx	JPOS	Salta si positivo	if ac >= 0 then pc := x
0101	xxxx	xxxx	xxxx	JZER	Salta si cero	if ac = 0 then pc := x
0110	xxxx	xxxx	xxxx	JUMP	salta	pc:= x
0111	xxxx	xxxx	xxxx	LOCO	Carga constante	ac:= x
1000	xxxx	xxxx	xxxx	LODL	Carga local	ac:=m[sp+x]
1001	xxxx	xxxx	xxxx	STOL	Almacena local	m[sp+x]:= ac
1010	xxxx	xxxx	xxxx	ADDL	Suma local	ac:= ac + m[sp+x]
1011	xxxx	xxxx	xxxx	SUBL	Resta local	ac:= ac - m[sp+x]
1100	xxxx	xxxx	xxxx	JNEG	Salta si negativo	if ac < 0 then pc := x
1101	xxxx	xxxx	xxxx	JNZE	Salta si no cero	if ac <> 0 then pc := x
1110	xxxx	xxxx	xxxx	CALL	llama a rutina	sp:= sp -l; m[sp]:= pc; pc:= x
1111	0000	0000	0000	PSHI	Apila indirecto	sp := sp -l; m[sp]:= m[ac]
1111	0010	0000	0000	POPI	Desapila indirecto	m[ac]:= m[sp]; sp:= sp -1
1111	0100	0000	0000	PUSH	Apila	sp:= sp -l; m[sp]:= ac
1111	0110	0000	0000	POP	Desapila	ac:= m[sp]; sp:= sp +l
1111	1000	0000	0000	RETN	Retorno	pc := m[sp]; sp := sp + 1
1111	1010	0000	0000	SWAP	intercambia	tmp := ac; ac := sp; sp := tmp
1111	1100	уууу	уууу	INSP	Incrementa SP	sp := sp + y
1111	1110	уууу	уууу	DESP	Decrementa SP	sp:sp-y

# UNTOF Universidad Nacional de Tierra del Fuego, Antárida e Islas del Atlántico Sur. CURSO 2020

# Anexo II

## Microprograma

0:	mar := pc ; rd ;	40:	tir := Ishift(tir); if n then goto 46;
1:	pc := pc + 1 ; rd ;	41:	alu := tir ; if n then goto 44 ;
2:	ir := mbr ; if n then goto 28 ;	42:	alu := ac ; if n then goto 22 ;
3:	tir := Ishift (ir+ir) ; if n then goto 19 ;	43:	goto 0;
4:	tir := Ishift(tir) ; if n then goto 11 ;	44:	alu := ac ; if z then goto 0 ;
5:	alu := tir ; if n then goto 9 ;	45:	pc := band(ir,amask) ; goto 0 ;
6:	mar := ir ; rd ;	46:	tir := Ishift(tir); if n then goto 50;
7:	rd;	47:	sp := sp+(-1);
8:	ac := mbr ; goto 0 ;	48:	mar := sp ; mbr := pc ; wr ;
9:	mar := ir ; mbr := ac ; wr ;	49:	pc := band(ir,amask) ; wr ; goto 0 ;
10:	wr; goto 0;	50:	tir := lshift(tir); if n then goto 65;
11:	alu := tir ; if n then goto 15 ;	51:	tir := lshift(tir); if n then goto 59;
12:	mar := ir ; rd ;	52:	alu := tir ; if n then goto 56 ;
13:	rd;	53:	mar := ac ; rd ;
14:	ac := mbr+ac ; goto 0 ;	54:	sp := sp+(-1) ; rd ;
15:	mar := ir ; rd ;	55:	mar := sp ; wr ; goto 10 ;
16:	ac := ac+1 ; rd ;	56:	mar := sp ; sp := sp +1 ; rd ;
17:	a := inv(mbr) ;	57:	rd;
18:	ac := ac+a ; goto 0 ;	58:	mar := ac ; wr ; goto 10 ;
19:	tir := Ishift(tir) ; if n then goto 25 ;	59:	alu := tir ; if n then goto 62 ;
20:	alu := tir ; if n then goto 23 ;	60:	sp := sp+(-1);
21:	alu := ac ; if n then goto 0 ;	61:	mar := sp ; mbr := ac ; wr ; goto 10 ;
22:	pc := band(ir,amask) ; goto 0 ;	62:	mar := sp ; sp := sp +1 ; rd ;
23:	alu := ac ; if z then goto 22 ;	63:	rd;
24:	goto 0;	64:	ac := mbr ; goto 0 ;
25:	alu := tir ; if n then goto 27 ;	65:	tir := lshift(tir); if n then goto 73;
26:	pc := band(ir,amask) ; goto 0 ;	66:	alu := tir ; if n then goto 70 ;
27:	ac := band(ir,amask) ; goto 0 ;	67:	mar := sp ; sp := sp +1 ; rd ;
28:	tir := Ishift (ir+ir) ; if n then goto 40 ;	68:	rd;
	+		1



29:	tir := lshift(tir) ; if n then goto 35 ;	69:	pc := mbr ; goto 0
30:	alu := tir ; if n then goto 33 ;	70:	a := ac ;
31:	a := ir + sp ;	71:	ac := sp ;
32:	mar := a ; rd ; goto 7 ;	72:	sp := a ; goto 0 ;
33:	a := ir + sp ;	73:	alu := tir ; if n then goto 76 ;
34:	mar := a ; mbr := ac ; wr ; goto 10 ;	74:	a := band(ir,amask) ;
35:	alu := tir; if n then goto 38;	75:	sp := sp + a ; goto 0
36:	a := ir + sp ;	76:	a := band(ir,amask) ;
37:	mar := a ; rd ; goto 13 ;	77:	a := inv(a) ;
38:	a := ir + sp ;	78:	a := a +1 ; goto 75 ;
39:	mar := a ; rd ; goto 16 ;		

## Anexo III

Instrucción de máquina convencional	Números de microinstrucciones de decodificación	Números de microinstrucciones de ejecución
LODD		
JZER		
RETN		

### **Anexo IV**

Operación	Descripcion	significado	Observaciones
INAC	Incrementa el acumulador	ac:= ac + 1	
DEAC	Decrementa el acumulador	ac:= ac - 1	
ADDC	Suma <b>y</b> al acumulador	ac := ac + y	donde <b>y</b> son los 8 bits menos significativos de la instrucción
SUBC	Resta <b>y</b> al acumulador	ac := ac - y	donde <b>y</b> son los 8 bits menos significativos de la instrucción