PAC1

Estructura de Computadors

Grau en Enginyeria Informàtica

feb18-jun18

Estudis d'informàtica, multimèdia i comunicació

EIMT.UOC.EDU

Universitat Oberta de Catalunya

Presentació

La present PAC1 conté 4 preguntes i representa el 50% de la nota de l'avaluació contínua.

Com podreu veure, els exercicis són molt semblats als quals heu fet durant aquests dies, en els quals a més heu pogut donar les solucions, comentar-les i plantejar dubtes en el fòrum. Aquesta PAC és **individual**, **avaluable** i per tant no pot comentar-se.

Competències

Les compentències específiques que persegueix la PAC1 són:

- [13] Capacitat per identificar els elements de l'estructura i els principis de funcionament d'un ordinador.
- [14] Capacitat per analitzar l'arquitectura i organització dels sistemes i aplicacions informàtics en xarxa.
- [15] Conèixer les tecnologies de comunicacions actuals i emergents i saber-les aplicar convenientment per dissenyar i desenvolupar solucions basades en sistemes i tecnologies de la informació.

Objectius

Els objectius de la següent PAC són:

- Conèixer el joc d'instruccions de la màquina CISCA.
- Conèixer els modes d'adreçament de la màquina CISCA.
- Traduir petits programes a assemblador.
- Comprendre la codificació interna de les instruccions d'assemblador.
- Descriure les microoperacions involucrades en l'execució d'una instrucció d'assemblador.

Enunciat

Respondre cada pregunta o apartat en el requadre corresponent.

Recursos

Podeu consultar els recursos disponibles a l'aula, però no fer ús del fòrum.

Criteris de valoració

La **puntuació** de cada pregunta i els **criteris d'avaluació** els trobareu a cada pregunta.

Format i data de lliurament

- La PAC1 podeu lliurar-la a l'apartat de **lliurament d'activitats** amb el nom **cognom1_cognom2_nom_PAC1 (pdf / odt / doc / docx).**

- La data **límit** de lliurament és el **23/03/2018.**

Enunciat

Pregunta 1 (2 punts)

Suposeu que l'estat inicial del computador (el valor que contenen els registres, posicions de memòria i bits de resultat just abans de començar l'execució de cada fragment de codi, de cada apartat) és el següent:

R0= 400h R1= 300h R2= 200h R3= 100h R4= 500h	M(00000100h)=FFFFF000h M(00000200h)=0000FFFFh M(00000300h)=11110000h M(00000400h)=00001111h M(00000500h)=F000000Fh

- Bits de resultat del registre d'estat: Z=0, S=0, C=0, V=0

- Registres especials: suposem que el PC apunta a l'inici del fragment de codi de cada apartat.

Quin serà l'estat del computador després d'executar cadascun dels següents fragments de codi? Indiqueu solament el contingut (**en hexadecimal**) dels registres i posicions de memòria que s'hagin modificat com a resultat de l'execució del codi. Indiqueu el valor final de tots els bits de resultat. (No us demanem que indiqueu el valor del PC després d'executar el codi i per això no us hem donat el valor inicial del PC, on comença cada fragment de codi).

Suposeu que l'adreça simbòlica A val 00000500h.

Important: deixeu clar al final de cada apartat els registres, adreces de memòria i bits d'estat que es modifiquen, amb el seu valor final.

a)

```
      SUB R1, R0
      MOV [R3], R1
XOR [R3], R4
```

Solució:

R1:= 300h – 400h = FFFFFF00h
[00000100h]= FFFFFF00h
[00000100h]= FFFFFF00h XOR 00000500h = FFFFFA00h

R1= FFFFFF00h
[00000100h]= FFFFFA00h
Z= 0 , S= 1 , C= 0 , V= 0

b)

```
      NOT R1
      ADD R1, [R2]
      SAR R1, 10h
      SUB R1, [A]
```

Solució:

R1= NOT(00000300h)= FFFFFCFFh
R1= FFFFFCFFh + 0000FFFFh = 0000FCFEh
R1= 00000000h
R1= 00000000h – F000000Fh = 0FFFFFF1h

R1= 0FFFFFF1h
Z= 0 , S= 0 , C= 1 , V= 0

c)

```
PLUS:CMP R3,R1
     JE  END
     ADD R0, [R3]
     ADD R3, 100h
     JMP PLUS
END: ADD R0, [A]
```

Solució:

CMP 100h, 300h
R0:= 400h + FFFFF000h = FFFFF400h
R3:= 200h

CMP 200h, 300h
R0:= FFFFF400h + 0000FFFFh = 0000F3FFh
R3:= 300h

CMP 300h, 300h

R0:= 0000F3FFh + F000000Fh= F000F40Eh

R3= 300h
R0= F000F40Eh
Z= 0 , S= 1 , C= 0 , V= 0

Criteris de valoració. Els apartats a) i b) valen 0,5 punts cadascun i el c) val 1 punt. La valoració de cada apartat és del 100% si no hi ha cap error en la solució de l'apartat, és del 50% si el contingut de les posicions de memòria i registres modificats és correcte però hi ha algun error en el contingut de un o varis dels bits de resultat, i és del 0% si hi ha algun error en alguna posició de memòria o registre modificat.

Pregunta 2 (2 punts)

En el codi d'alt nivell M és una variable de tipus vector de dues dimensions (matriu) de 10 per 10 elements. Cada element de la matriu és un enter de 32 bits. A el programa assemblador la matriu es troba emmagatzemada a partir de l'adreça simbòlica M, per files i en posicions consecutives (M[0][0], en l'adreça simbòlica M, el següent, M[0][1], en l'adreça M+4, …, M[1][0] en l'adreça M+40, M[1][1] en l'adreça M+44, etc.).

El programa ha de multiplicar per 2 els números senars que trobi en les columnes senars, tenint en compte que les columnes van de la 0 a la 9, tractem les columnes 1, 3, 5, ..., 9).

Nota: *Els nombres, representat en binari, son senars si acaben en 1 i parells si acaben en 0.*

Al programa assemblador R1 adreça I, R2 s'usa adreça J.

Exemple:
Suposant una matriu 10x10 amb les següents dades inicials (només apareixen les files 0,1 i 9 a nivell d'exemple):

2	3	4	3	2	9	8	9	6	5
12	9	7	87	15	4	0	4	2	9
…									
2	7	6	4	9	8	5	0	6	2

Segons l'exemple anterior i el que es demana, en negreta apareixen els números que s'han dividit (per simplificar només apareixen les files 0, 1 i 9):

2	**6**	4	**6**	2	**18**	8	**18**	6	**10**
12	**18**	7	**174**	15	4	0	4	2	**18**
…									
2	**14**	6	4	9	8	5	0	6	2

```
I:= 0;
 J:= 0;
 WHILE I<10 {
    J:= 1;
    WHILE J<10 {
            IF ((M[I][J] % 2)==1) M[I][J]= M[I][J] * 2;
            J:= J+2;
    }
    I:= I+1;
 }
```

```
Solució:
            MOV R1, 0
            MOV R2, 0
WI:         CMP R1, 400
            JGE ENDWI
            MOV R2, 4
WJ:         CMP R2, 36      ; CMP R2, 40
            JG  ENDWJ       ; JGE ENDWJ
            MOV R3, R1
            ADD R3, R2
            MOV R4, [M+R3]
            MOV R5, R4
            AND R4, 1
            JE  NO_MUL
            MUL R5, 2
            MOV [M+R3], R5
NO_MUL:     ADD R2, 8
            JMP WJ
ENDWJ:      ADD R1, 40
            JMP WI
ENDWI:
```

Criteris de valoració.

2 punts. Es perden 0,5 punts per cada instrucció incorrecta.

Pregunta 3 (3 punts)

Donat el següent fragment de codi d'un programa en llenguatge assemblador CISCA:

```
PLUS: CMP R10,0
      JE END
      ADD R0, [R4]
      XOR R10, [V]
      JMP PLUS
END:
```

Traduïu-lo a llenguatge màquina i expresseu-lo en la següent taula. Suposeu que la primera instrucció del codi s'assembla a partir de l'adreça **003FCF00h** (que és el valor del PC abans de començar l'execució del fragment de codi). Suposeu que l'adreça simbòlica V val **00124000h**. En la següent taula useu una fila per codificar cada instrucció. Si suposem que la instrucció comença en l'adreça @, el valor Bk de cadascun dels bytes de la instrucció amb adreces @+k per a k=0, 1,… s'ha d'indicar en la taula en hexadecimal en la columna corresponent (recordeu que els camps que codifiquen un desplaçament en 2 bytes o un immediat o una adreça en 4 bytes ho fan en format little endian, això cal tenir-ho en compte escrivint els bytes de menor pes, d'adreça més petita, a l'esquerra i els de major pes, adreça major, a la dreta). Completeu també la columna 'Adreça' que indica per a cada fila l'adreça de memòria del byte B0 de la instrucció que es codifica en aquesta fila de la taula.

		Bk per a k=0..10										
Adreça	**Assemblador**	0	1	2	3	4	5	6	7	8	9	10
003FCF00h	CMP R10,0	26	1A	00	00	00	00	00				
003FCF07h	JE END	41	60	10	00							
003FCF0Bh	ADD R0, [R4]	20	10	34								
003FCF0Eh	XOR R10, [V]	32	1A	20	00	40	12	00				
003FCF15h	JMP PLUS	40	00	00	CF	3F	00					
003FCF1Bh												

Criteris de valoració. Cada instrucció assemblada incorrectament resta 0.5 punts. Una instrucció està incorrectament assemblada si no s'escriu el valor o s'escriu un valor incorrecte de un o varis dels dígits hexadecimal que codifiquen la instrucció en llenguatge màquina. A més es resta 0.5 punts si hi ha algun error en la columna que indica les adreces de memòria en les quals comença cada instrucció.

Pregunta 4 (2 punts)

El *cicle d'execució* de una instrucció es divideix en 3 fases principals
1) Lectura de la instrucció
2) Lectura dels operands font
3) Execució de la instrucció i emmagatzematge de l'operant destinació

Donar la seqüència de micro-operacions que cal executar en cada fase per a les següents instruccions del codi codificat en la pregunta anterior.

ADD R0,[R4]

Fase	Micro-operacions
1	MAR ← 003FCF0Bh, READ ; Contingut del PC al registre MAR MBR ← 341020h ; Llegim instrucció PC ← 003FCF0Eh ; Incrementem PC en 3 unitats. IR ← MBR ; Carreguem instrucció a registre IR
2	MAR ← contingut de IR registre (R4), READ MBR ← memòria
3	R0 ← R0 + MBR

JMP PLUS

Fase	Micro-operacions
1	MAR ← 003FCF15h, READ ; Contingut del PC al registre MAR MBR ← 003FCF000040h ; Llegim instrucció PC ← 003FCF1Bh ; Incrementem PC en 6 unitats IR ← MBR ; Carreguem instrucció a IR
2	L'operant font és a la instrucció mateixa
3	PC ← 003FCF00h ; Actualitzem PC amb l'adreça de salt

Criteris de valoració. Si tot està correcte s'obté 2 punts. Cada instrucció és independent i val 1 punt. Es resta 0.5 per cada fallada dins de la mateixa instrucció.

Pregunta 5 (1 punt)

Respon a les següents preguntes:

Pregunta 1. Què entenem per «segmentació de les instruccions»?

La segmentació de les instruccions (pipeline) consisteix en dividir el cicle d'execució de les instruccions en un conjunt d'etapes. Aquestes etapes poden coincidir, o no, amb les fases del cicle d'execució de les instruccions.

Pregunta 2. En què consisteix l'adreçament “relatiu a PC”?

L'adreçament relatiu a PC és equivalent al relatiu a registre base, amb la diferència que utilitza el registre comptador de programa (PC) de manera implícita a la instrucció.

Criteris de valoració. Cada pregunta individual val 0,5 punts si està correcta. 0 si no ho està o és incompleta.