

PRÀCTIQUES DE L'ASSIGNATURA

INTRODUCCIÓ ALS ORDINADORS



UNIVERSITAT DE BARCELONA



Índex

PRÀCTIQUES DE L'ASSIGNATURA.....	1
INTRODUCCIÓ ALS ORDINADORS.....	1
1. Normes de les Pràctiques.....	3
Avaluació.....	3
Sessions Pràctiques.....	3
Normativa.....	3
Pràctica 5: Introducció al simulador i8085.....	4
Objectius.....	4
Introducció Teòrica.....	4
Exercici I. Modes d'Adreçament del i8085.....	5
Exercici 2: Subrutines.....	5
Apunts.....	6
Informe.....	6
Pràctica 6: Memòries i subrutines.....	7
Part I: Objectiu.....	7
Cos de la Part I.....	7
Estudi de l'espai de memòries del microprocessador.....	8
Funcionament de la Pila.....	8
PART II: Objectiu.....	8
Cos de la Part II.....	8
Part III. Objectiu.....	9
Informe.....	9
Pràctica 7	10
Objectiu de la pràctica.....	10
1. Suma de dos valors introduïts per consola.....	10
2. Resta de dos valors introduïts per consola.....	10
3. Ensamblant el codi.....	10
Informe.....	11

1. Normes de les Pràctiques

Avaluació

Les pràctiques d'Introducció als Ordinadors són una part integrant de l'assignatura. Aquestes pràctiques es realitzen a l'aula IE de la facultat de Matemàtiques i comporten la realització de 10 pràctiques avaluable, un miniprojecte i un examen pràctic final. El resultat de l'avaluació d'aquestes pràctiques constitueix el 50% de la nota de l'assignatura. Per poder aprovar l'assignatura la nota mitjana de pràctiques ha de ser superior o igual a 5 i caldrà presentar-se a totes les sessions i entregar tota la documentació requerida.

Sessions Pràctiques

L'assignatura consta de un total de 10 pràctiques dividides en 10 sessions i un miniprojecte com a treball a realitzar per l'alumne a casa. S'establiran períodes d'entrega a través del Campus Virtual de l'assignatura. Si l'entrega no es realitza en el període establert, la pràctica constarà com suspesa.

Normativa

- Els alumnes treballaran a l'aula individualment (si hi ha ordinadors suficients). En cas de que es disposi d'un ordinador portàtil propi, pot portar-se a l'aula per a la realització de les pràctiques si el alumne així ho vol.
- Les pràctiques es realitzaran utilitzant el sistema operatiu Windows XP. La contrasenya d'entrada és l'assignada per accedir als ordinadors de la facultat
- Queda totalment prohibit instal·lar o utilitzar programes propis als ordinadors de l'aula
- No està permesa la utilització dels ordinadors per realitzar treballs d'altres assignatures durant les pràctiques.

Pràctica 5: Introducció al simulador i8085

1 sessió guiada. El professor introduirà el conjunt d'instruccions així com el seu format. Els alumnes aniran seguint els passos donats pel professor de pràctiques per mirar d'entendre el funcionament del simulador i8085.

Objectius

Conèixer els modes d'adreçament del 8085 i familiaritzar-se amb les seves instruccions.

Introducció Teòrica

El 8085 és un processador de 8 bits que consta d'un format d'adreçament de 16 bits. Per tal de fer això, el que fa és agrupar els registres en parelles.

El conjunt de registres és el següent

- Acumulador (8 bits) i Registre d'estat (8 bits) => En parella donen 16 bits que és el que es guarda en la pila quan sigui necessari.
- Registres de propòsit general:
 - Registre B (8 bits) i Registre C (8 bits)
 - Registre D (8 bits) i Registre E (8 bits)
 - Registre H (8 bits) i Registre L (8 bits)
- Registres que apunten a memòria:
 - Registre Program Counter (16 bits)
 - Registre Stack Pointer (16 bits)

A la figura 1 es mostra el diagrama de blocs del processador 8085 on es mostren els registres esmentats anteriorment.

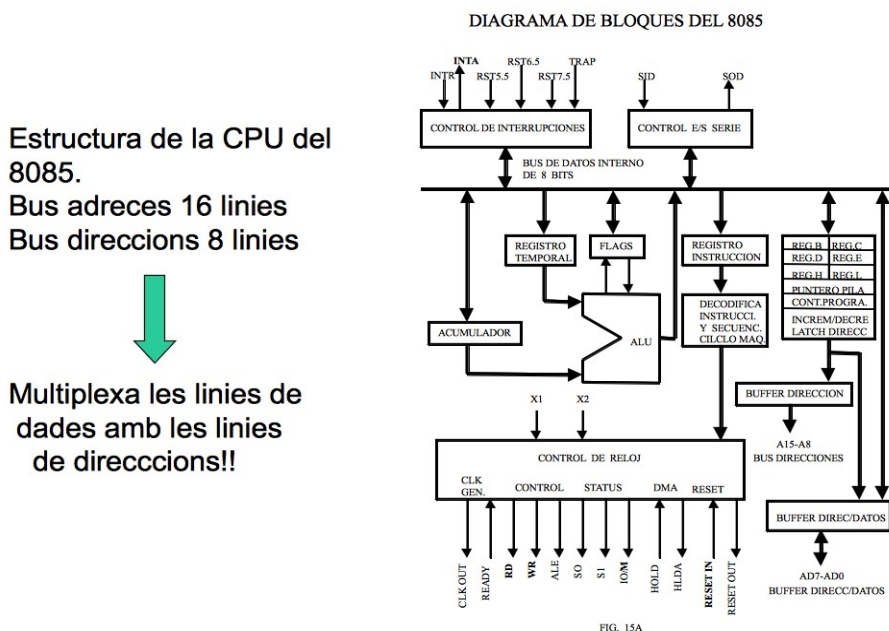


Figura 1. Diagrama de blocs del 8085

S'adjunta com a apèndix una presentació al campus virtual, on es proporciona als alumnes una

introducció al 8085. Els professors de pràctiques realitzaran una explicació a partir d'aquesta presentació.

Exercici I. Modes d'Adreçament del i8085

Escriuiu dos programes diferents per sumar dues matrius de 5x1 i desar el resultat en una tercera, podeu fer servir adreçament indirecte, per parella de registres o per registres. Penseu en dues hipòtesis, primera, les matrius d'entrada s'han de conservar, és a dir el resultat s'emmagatzema sobre mat3 i segona, desem el resultat sobre una de les dues matrius d'entrada, mat3 és irrellevant. El programa hauria d'utilitzar un comptador, instruccions d'increment, comparació i salt condicional. Feu la suma de números hexadecimals i sense tenir en compte números que pugui donar overflow.

Pregunta 1

En què simplificaria molt el codi del programa un dels modes d'adreçament del simulador Ripes?

L'inici del programa serà el següent:

```
.define
    num 5
.data 00h
    mat1: db 1,2,3,4,5
    mat2: db 6,7,8,9,0
    mat3: db 0,0,0,0,0
.org 100h
```

Calculeu les mides del codi del vostre programa i el nombre de cicles per a la seva execució.

Feu us de l'ajuda del programa simulador per conèixer les instruccions del 8085 i els seus modes d'adreçament, també ajudeu-vos de les explicacions de l'apartat de teoria. Tingueu en compte que no totes les instruccions suporten tots els modes d'adreçament. Feu la suma de números hexadecimals i sense tenir en compte números que pugui donar overflow.

Pregunta 2

Quants cicles de rellotge triga en executar-se una instrucció aritmètic – lògica qualsevol? Feu servir el fitxer adjunt on especifica el ISA del 8085. Indica quina és la mida mitjana de les teves instruccions. Calcula els cicles per instrucció mitjà per aquests codis.

Pregunta/Tasca 3

Pugeu el vostre codi i marqueu quina és la instrucció del vostre programa que triga més cicles en executar-se

Pregunta/Tasca 4

Traduiu el codi per fer-lo servir amb el simulador **Ripes**. Quants cicles triga en executar-se? Compareu els resultats (mida de codi, accessos a memòria i cicles promig per instrucció) amb els valors obtinguts per l'i8085

Exercici 2: Subrutines

Fent ús del programa anterior, realitzeu una subrutina que codifiqui una zona de memòria. La zona de memòria s'indicarà posant al registre doble HL l'adreça de començament de la zona de dades a codificar. Aquestes dades es consideren com els paràmetres d'entrada de la subrutina. La codificació es farà mitjançant una XOR entre cada byte de la zona de dades i la clau. Els valors dels registres no han de quedar afectats per la crida a la subrutina. Feu un programa que faci us

d'aquesta subrutina per provar-la amb diferents combinacions de valors de la clau i les dades per codificar.

Ajut: Crida a una subrutina en i8085:

```
call nom_subrutina
...
.org adreça_subrutina
PUSH...
/* Codi de la subrutina aquí
    TO DO...*/
POP...
RET
```

Ajut: Crida a una subrutina en Ripes:

```
call nom_subrutina
...
.text adreça_subrutina
addi sp,sp,-16 # Reservar stack frame
sw s1,4(sp)
sw s0,8(sp)
/* Codi de la subrutina aquí
    TO DO...*/
lw, s0, 8(sp)
lw s1, 4(sp)
addi sp, sp, 16
ret
```

No tots els registres utilitzats en una subrutina s'han de preservar. I8085 només preservarà els registres guardats explícitament en la pila, amb PUSH i POP. El mateix passa amb Ripes. En aquest cas, donat el nombre de registres de que disposa aquesta arquitectura, hi ha un registres més focalitzats en fer-se servir com a registres temporals i altres que requereixen ser preservats. Per exemple, els saved registers (s0-s11) es diuen així perquè la subrutina els ha de preservar, mentre que els temporary registers (t0-t6) no cal (el caller no pot assumir que es preservaran després d'una crida a funció).

Altres tipus de crides a subrutines: Tail calls

En informàtica, una crida de cua o tail call es una crida de subrutina realitzada com acció final de un procediment. Aquestes subrutines fan un ús molt més senzill i simplificat de la pila. Exemples de Tail calls són:

```
function foo(data) {
    a(data);
    return b(data);
}
```

Aquí, tant a(dada) com b(dada) són crides a subrutines, però b és l'últim que s'executa la funció abans de tornar, trobant-se així en la posició de la cua. b(dada) és un Tail call. No obstant això, no totes les trucades de cua es troben necessàriament a l'extrem sintàctic d'una subrutina

```
function bar(data) {
    if ( a(data) ) {
        return b(data);
    }
}
```

```

return c(data);
}

```

Aquí tant b(data) con c(data) són Tail calls.

Pregunta 4

Quina instrucció fem servir en tots dos casos per assignar la posició inicial al registre SP?

Pregunta 5

Quina es la instrucció utilitzada per guardar el PC en la pila quan treballem amb subrutines? I per recuperar de nou el valor del PC?

Tasca 2

Pujeu el codi creat amb les subrutines

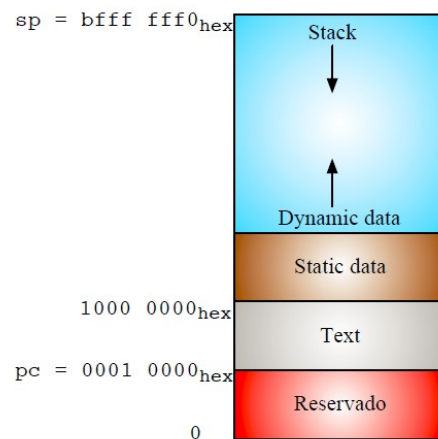


Figura 3.10: Reserva de memoria para el programa y datos en RV32I. Las direcciones más altas aparecen en la parte superior de la figura y las direcciones más bajas en la parte inferior. En esta convención del software de RISC-V, el **stack pointer** (sp) comienza en `bfff fff0_hex` y crece hacia abajo hacia el área de *Static data*. El área de *text* (código del programa) comienza en `0001 0000_hex` e incluye las librerías *linkeadas* estáticamente. El área de *Static data* comienza inmediatamente después del área de *text*; en este ejemplo, asumimos que es la dirección `1000 0000_hex`. Los datos dinámicos, reservados en C usando `malloc()`, están justo después del *Static data*. Llamado el *heap*, crece hacia el área de stack e incluye las librerías *linkeadas* dinámicamente.

Apunts

El programa que es farà servir és el Simulador del Microprocessador Intel 8085. Aquest programa inclou un editor de llenguatge ensamblador que es recomana fer servir per crear els programes (F2). Amb la tecla "F1" podeu obtenir ajuda en línia de la instrucció que es troba marcada amb el cursor. Des d'aquest editor podeu ensamblar el programa i passar al simulador. El simulador permet veure en tot moment el contingut de la memòria i els registre. Els registre apareixen agrupats de dos en dos format un sol registre de 16bits. El seu valor es pot veure en binari i hexadecimal. Cal treballar amb definició de pantalla de 800x600(o superior) i lletres petites.

Informe

Entregueu a l'informe el següent:

- Objectius assolits en fer aquesta pràctica
- Resposta de les preguntes i qüestions que es presenten al guió
- Diagrames de blocs comentats dels diferents programes i explicacions sobre el codi.
- Conclusions

Pràctica 6: Pila i entrada/sortida

(Part I: sessió guiada per part dels professors de pràctiques, Part II i III: treball a casa)

Part I

Objectius:

- Estudi de l'espai de memòries del microprocessador
- Comprendre el funcionament de la pila
- Veure un exemple d'utilització de subrutines.

Codi en i8085

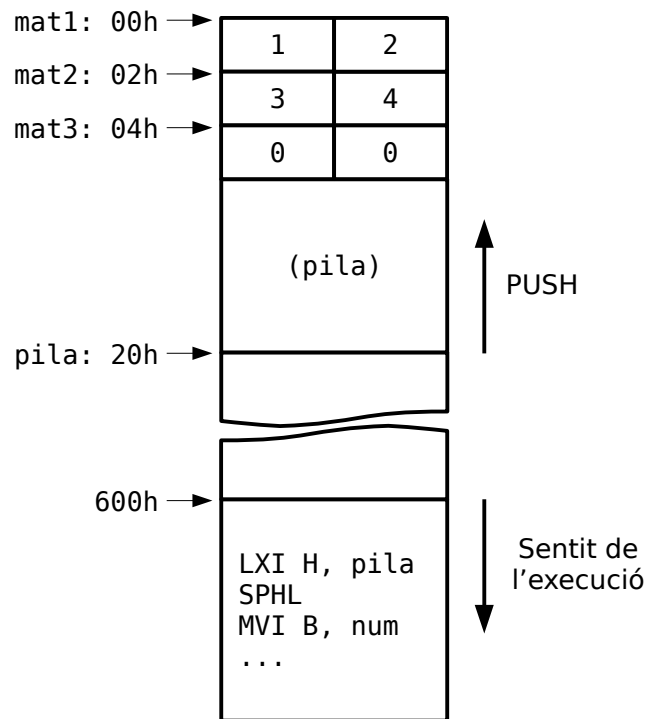
Executeu pas a pas el següent programa:

i8085	RISC V (com a ajuda)
<pre>.define num 02h .data 00h mat1: db 1,2 mat2: db 3,4 mat3: db 0,0 .data 20h pila: .org 600h LXI H, pila SPHL MVI B, num LXI D, mat1 LXI H, mat2 loop: CALL suma DCR B JNZ loop NOP HLT suma: PUSH PSW LDAX D ADD M STAX D INX H INX D POP PSW RET</pre>	<pre>.data mat1: .byte 1, 2 mat2: .byte 3, 4 mat3: .byte 0, 0 .text li s0, 2 la a1, mat1 la a2, mat2 loop: jal suma addi s0, s0, -1 bgtz s0, loop j end suma: addi sp, sp, -12 sw s1, 12(sp) sw s2, 8(sp) sw s3, 4(sp) lb s1, 0(a1) lb s2, 0(a2) add s3, s1, s2 sb s3, 0(a1) addi a1, a1, 1 addi a2, a2, 1 lw s2, 4(sp) lw s2, 8(sp) lw s1, 12(sp) addi sp, sp, -12 ret end:</pre>

Com es faria per desar el resultat en mat3? (guiat, no cal entregar)

Preguntes (cal entregar)

1. L'adreçament de la instrucció LXI és:
 - a) directe
 - b) indirecte
 - c) immediat
 - d) implícit
2. Quina instrucció guarda el PC a la Pila?
 - a) PUSH PC
 - b) POP PC
 - c) CALL
 - d) MOV M, PC
3. Quin espai ocupa en memòria la subrutina 'suma'?
4. Quants cicles triga en executar-se la subrutina 'suma'?



Estudi de l'espai de memòries del microprocessador

TASCA 1 (cal entregar)

Dibuixeu el mapa de memòria de dades: direccions i contingut. Indiqueu les instruccions que modifiquen les dades de la memòria. En cadascuna d'elles, indiqueu quines modificacions es produeixen.

Situeu la memòria de programa. Dins d'ella, localitza el sub-bloc que pertany a la subrutina 'suma'.

Funcionament de la Pila

TASCA 2 (cal entregar)

Indiqueu el començament de la pila en l'espai de memòria de dades del microprocessador.

Indiqueu quines instruccions modifiquen la pila. Per cadascuna, indiqueu:

Instrucció	Descripció	Canvi en la pila
PUSH PSW	Afegeix PSW a la pila.	Augmenta en 2 bytes.
...		

PART II

Objectius:

- Comprendre l'adreçament a ports E/S.

El simulador permet comunicar el processador amb una sèrie de ports E/S:

- Ports d'entrada: interruptors i teclat.
- Ports de sortida: panell de LEDs, *display* 7 segments, *display* 15 segments.

Per accedir a aquests recursos, haurem d'accedir a les direccions adients dels ports.

Codi

Executeu pas a pas el següent programa:

```
.data 100h
      pila:
.org 24h
      JMP ports
.org 500h
      LXI H, pila
      SPHL
      CALL ports
      NOP
      HLT
ports:
      PUSH PSW
      IN 04h
      ANI 00000001
      OUT 05h
      POP PSW
      RET
```

TASCA 3 (cal entregar)

Què fa la subrutina 'ports'?

Per això, introduïu dades amb els interruptors o amb el teclat; observeu en un port de sortida el resultat de la subrutina.

Part III

Objectius:

- Dissenyeu un programa *assembler* que representi en un *display* de 7 segments els números del 0 al 5. Els números s'introduiran a partir del teclat. A més, ha de permetre l'opció d'esborrar el *display*; per això, en prémer la lletra 'c', es produirà un CLEAR del *display*.

Propostes de millora:

El simulador 8085 té una part de la memòria de dades que implementa una 'Pantalla de text'.

Modifiqueu el programa per escriure la lletra que vulgueu a la pantalla. Al menú "Opciones" → "de interrupciones" podeu habilitar que es generi una interrupció TRAP en prémer qualsevol tecla del teclat. Això produirà el salt de l'execució del programa a la subrutina que hi hagi a la direcció de memòria 0024h. Aproveiteu aquesta possibilitat per reproduir el funcionament continu del teclat i la pantalla.

TASCA 4 (cal entregar)

Pujeu un fitxer amb:

- Explicació breu de l'algorisme triat.
- Programa assemblador comentat.

Informe

Entregueu a l'informe el següent:

- Objectius assolits en fer aquesta pràctica.
- Tasques i resposta de les preguntes que es presenten al guió i marcades com «cal entregar».
- Diagrames de blocs comentats dels diferents programes i explicacions sobre el codi.
- Conclusions.

Pràctica 7

(1 sessió + treball a casa)

Objectiu de la pràctica

Programar en ensamblador del 8085 varies aplicacions, demostrant els coneixements adquirits en teoria.

1. Suma de dos valors introduïts per consola

Dissenyeu una subrutina que a partir de dos nombres introduïts pel teclat del simulador i8085 faci la suma i presenti el resultat en la pantalla de text del i8085. Feu servir els adreçaments directe i indirecte i indiqueu al codi on tenim aquests adreçaments.

Tasca 1. Pugeu el codi. Com gestioneu el problema del signe? Com gestioneu el problema del overflow?

2. Resta de dos valors introduïts per consola

Dissenyeu una subrutina que a partir de dos nombres introduïts pel teclat del simulador i8085 faci la resta i presenti el resultat en la pantalla de text del i8085. Feu servir els adreçaments directe i indirecte i indiqueu al codi on tenim aquests adreçaments.

Tasca 2. Pugeu el codi de la resta. Com gestioneu el problema del signe? Com gestioneu el problema del carry?

3. Ensamblant el codi

A partir dels codis generats en els apartats 1 i 2, feu un programa capaç de fer sumes, restes, AND's i OR's

Tasca 3. Pugeu el codi final.

Questió 1:

Quina diferència hi ha entre la suma i la OR?

- i) són iguals
- ii) la OR és una operació lògica i la suma és una operació aritmètica.
- iii) La OR és una operació aritmètica i la suma és una operació lògica
- iv) cap de les anteriors és correcta.

Questió 2:

La instrucció STA 1234h

- i) és una operació que carrega el contingut de la posició de memòria 1234h en l'acumulador
- ii) fa servir adreçament directe

- iii) fa servir adreçament immediat
- iv) totes són certes

Informe

Entregueu a l'informe el següent:

- Objectius assolits en fer aquesta pràctica
- Resposta de les preguntes i qüestions que es presenten al guió
- Diagrames de blocs comentats dels diferents programes i explicacions sobre el codi.
- Conclusions