Curs de Reavaluació d’IC – Sessió 6

# Activitat 1

Estudia el document corresponent al tema 13 de teoria que hi ha Atenea:

• “Teoría: El computador SISC Von Neumann”

Estudiar vol dir llegir els documents curosament i apuntar els dubtes que vagin sorgint per preguntar-los a la classe.

Activitat 2

Realitza els següents problemes i entrega les solucions en un document PDF a l’espai reservat a Atenea.

# Problema 1

Donat el següent codi en SISA.

.data v: .word 12,99,4,39,102,0,1,2,23,63 max: .word 0

.text

MOVI R0, lo(v)

MOVHI R0, hi(v)

LD R1, 0(R0)

MOVI R7,2

MOVI R5,20 while:

CMPLT R4,R7,R5

BZ R4, fi\_while

ADD R3,R7,R0

LD R2,0(R3)

CMPLTU R4,R1,R2

BZ R4, fi\_if

ADDI R1,R2,0 fi\_if:

ADDI R7,R7,2

BNZ R7, while fi\_while:

MOVI R3, lo(max)

MOVHI R3, hi(max)

ST 0(R3),R1

.end

1. Ompliu el contingut de les paraules de la zona de dades de la memòria i completeu el contingut de la zona d'instruccions de la memòria. En cas de no poder determinar el valor escriviu XXXX. Assumiu que la primera instrucció es guarda a la direcció 0x0000 de la memòria, i que el primer valor especificat al .data es posa a l'adreça 0x1000 de la memòria.

|  |  |
| --- | --- |
| @ | I-Mem[@] |
| 0x0000 | 0x9000 |
| 0x0002 | 0x9110 |
| 0x0004 | 0x3040 |
| 0x0006 | 0X9E02 |
| 0x0008 | 0x9A14 |
| 0x000A | 0x1F60 |
| 0x000C | 0X8807 |
| 0x000E | 0x0E1C |
| 0x0010 | 0X3680 |
| 0x0012 | 0x12A4 |
| 0x0014 | 0x8801 |
| 0x0016 | 0x2440 |
| 0x0018 | 0x2FC2 |
| 0x001A | 0X8FF7 |
| 0x001C | 0x9614 |
| 0x001E | 0x9710 |
| 0x0020 | 0X4E40 |

|  |  |
| --- | --- |
| @ | Data\_Mem[@] |
| 0x1000 | 000C |
| 0x1002 | 0063 |
| 0x1004 | 0004 |
| 0x1006 | 0027 |
| 0x1008 | 0066 |
| 0x100A | 0000 |
| 0x100C | 0001 |
| 0x100E | 0002 |
| 0x1010 | 0017 |
| 0x1012 | 003F |
| 0x1014 | 0000 |

|  |  |
| --- | --- |
|  |  |

1. Una vez ejecutado el programa en el SISC von Neumann ¿Cuál es la dirección de la memoria donde ha escrito la última instrucción (ST 0(R3), R1) y cuál es su contenido?

# Memw[0x1014 ] = 0x0006

c) ¿Cuántas instrucciones se ejecutan, cuántas son lentas y cuántas son rápidas en la Harvard multiciclo y cuantos ciclos tarda desde que se ejecuta la primera instrucción del código hasta que se ejecuta la última instrucción del código, ambas incluidas? ¿Cuánto tarda en ejecutarse el código en el Harvard uniciclo, en el Hardard multiciclo y en el Von Neumann suponiendo que los tiempos de ciclo son 3.000, 2000 y 1.000 u.t. respectivamente?

Nº de instruc. ejecutadas =32 Nº instr. lentas (H. multiciclo)= 6 Nº instr. rápidas (H. multiciclo)= 26

Nº de ciclos (H. uniciclo)= 32 Nº de ciclos (H. multiciclo)= 24+26\*3 Nº de ciclos (Von Neumann)= 24+26\*3

Tejec(Harvard uniciclo) =3.000\*32 Tejec(Harvard multiciclo) = 2000\*(24+26\*3)

Tejec(Von Neumann) = 1.000\*(24+26\*3)

## Problema 2

El siguiente programa en ensamblador se ha traducido a lenguaje máquina para ser ejecutado en el SISC Von Neumann, situando la sección .text a partir de la dirección 0x0100 de memoria y a continuación la sección .data.

.data

N = 6 ; tiene que ser <= ; que 10 y > que 0.

A: .byte 10

.even

V1: .word 2,-5,264,-63,23

.word 58,-64,32,0,-7

.even

V2: .space 20

.text

L1: MOVI R0, lo(V1)

MOVHI R0, hi(V1)

MOVI R1, N

MOVI R2, 0

L2: LD R7, 0(R0)

ADD R2, R2, R7

ADDI R1,R1,-1

ADDI R0,R0,2

BNZ R1, L2

L3: MOVI R7, lo(V2)

MOVHI R7, hi(V2)

ST 4(R7), R2 .end

1. Una vez cargado el programa en memoria:
   * ¿A qué dirección de memoria corresponde la etiqueta, o dirección simbólica, V2?

V2 = 0x0136

* + ¿Cuál es la dirección de memoria y su contenido donde han quedado almacenadas cada una de las siguientes instrucciones?

MOVI R0, lo(V1) => Memw[0x0100] = 0x9022

LD R7, 0(R0) => Memw[0x0108 ] = 0x31C0

BNZ R1, L2 => Memw[0x0110] = 0x83FB

1. Una vez ejecutado el programa en el computador SISC Von Neumann ¿Cuál es la dirección de memoria donde ha escrito la instrucción (ST 4(R7),R2) y cuál es su contenido?

Memw[0x013A] = 0x0117

1. ¿Cuánto tarda en ejecutarse el código en las tres versiones de los computadores SISC? Suponed que se han fabricado los computadores con una tecnología tal que el tiempo de ciclo del Harvard uniciclo, el Harvard multiciclo y el Von Neumann es de 3.500, 900 y 1.500 u.t. respectivamente.

Tejec(Harvard uniciclo) = Tejec(Harvard multiciclo) = Tejec(Von Neumann) =

## Problema 3

Cada una de las filas de la tabla se pregunta sobre un ciclo concreto de la ejecución de una instrucción en el SISC Von Neumann. indicad el contenido del registro IR. Poned XXXX si no se puede saber.

|  |  |  |
| --- | --- | --- |
| Nodo / Estado  (Mnemo Salida) | Instrucción en IR  (en ensamblador) | Valor del IR  (en hexadecimal) |
| D | ADD R1,R2,R3 | 0x 04CC |
| Movhi | MOVHI R6,2 | 0x9002 |
| Out | OUT 73, R2 | 0xA549 |
| Addr | ST 4(R7), R1 | 0x4E44 |
| F | SUB R3,R2,R1 | 0xXXXX |
| Bz | BZ R6, -3 | 0x8CFD |

## Problema 4

Cada una de las filas de la tabla se pregunta sobre un ciclo concreto de la ejecución de una instrucción en el SISC Von Neumann. Indicad el contenido de la ROM\_OUT en hexadecimal usando las conexiones en el orden que están en el chuletario. Utilizad el valor 0 para los bits que sean x (solo para este apartado).

|  |  |  |
| --- | --- | --- |
| Nodo / Estado  (Mnemo Salida) | Instrucción en IR  (en ensamblador) | Contenido ROM\_OUT  (en hexadecimal) |
| D | ADD R1,R2,R3 | 0x 0020B0 |
| Movhi | MOVHI R6,2 | 0x04026A |
| Out | OUT 73, R2 | 0x080000 |
| Addr | ST 4(R7), R1 | 0x000030 |
| St | ST 4(R7), R1 | 0x208000 |
| Bz | BZ R6, -3 | 0x400220 |
| Al | SHA R5,R6,R0 | 0x041000 |
| F | SUB R3,R2,R1 | 0XC260F0 |
| Addi | ADDI R6, R7, -1 | 0x040031 |

## Problema 5

Cada una de las filas de la tabla se pregunta sobre un ciclo concreto de la ejecución de una instrucción en el SISC Von Neumann. Escribid el valor de los bits de la palabra de control que genera el bloque SISC CONTROL UNIT durante el ciclo a que hace referencia cada apartado. Poned x siempre que no se pueda saber el valor de un bit (ya que no sabemos cómo se han implementado las x en la ROM\_OUT). Para cada apartado/fila se indica el nodo/estado de la UC en ese ciclo y la instrucción (en ensamblador) que está almacenada en el IR en ese ciclo. Suponed que el contenido de todos los registros, Rk para k=0,...7, antes de ejecutarse cada instrucción es 0.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nodo / Estado  (  Mnemo Salida) | Instrucción en IR  (en ensamblador) | @A | @B | Pc/Rx | Ry/N | OP | F | P/I/L/A | @D | WrD | Palabra de Control  Wr-Out | Rd-In | Wr-Mem | LdIr | LdPc | Byte | Alu/R@ | R@/Pc | N  (hexa) | ADDR-IO  (  hexa) |
| F | JALR R1,R2 | 011 | 010 | 1 | 0 | 10 | 000 | XX | XX | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | X | XXXX | XX |
| Bz | BZ R6, -2 | 110 | 011 | 0 | X | 10 | 001 | XX | XX | 0 | 0 | 0 | 0 | X | 1 | 0 | 0 | X | FFFE | FE |
| Addr | LDB R2,3(R1) | 001 | 010 | 0 | 0 | 00 | 100 | XX | 010 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | X | X | 0003 | 04 |
| St | ST 4(R7), R1 | 111 | 001 | X | X | 00 | 100 | XX | XXX | 0 | 0 | 0 | 1 | X | 0 | 0 | X | 1 | 0004 | 04 |
| Al | SHA R5,R6,R0 | 110 | 000 | 0 | 1 | 00 | 100 | XX | 101 | 1 | 0 | 0 | 0 | X | 0 | 0 | X | X | XXXX | AE |
| F | SUB R3,R2,R1 | 010 | 001 | 1 | 0 | 00 | 101 | XX | 110 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | X | XXXX | 9D |
| Addi | ADDI R6,R7,-1 | 111 | 110 | 0 | 0 | 00 | 100 | 00 | 110 | 1 | 0 | 0 | 0 | X | 0 | 0 | X | X | FFFE | FE |

## Problema 6

Especificad el camino crítico (indicando la suma ordenada de los tiempos de propagación de los bloques por los que pasa) y calculad el tiempo de ciclo mínimo para que el computador SISC Von Neumann pueda ejecutar correctamente el tipo de instrucción SISA que se indica en cada apartado (este sería el tiempo de ciclo mínimo del computador si solo ejecutara instrucciones como la indicada u otras que requieran menor tiempo). No tenéis que añadir ningún porcentaje de seguridad en el cálculo del tiempo de ciclo mínimo. Suponed que los tiempos de propagación de los bloques que forman el computador son los siguientes:

Tp(ROM\_Q+) = 60 u.t.

Tp(ROM\_OUT) = 70 u.t.

Tp(MUX-2-1) = 40 u.t.

Tp(MUX-4-1) = 80 u.t.

Tp(REG) = 100 u.t. // Tiempo de propagación de un registro.

Tp(REGFILE) = 150 u.t. // Tiempo de lectura del banco de registros

Tp(ALU-slow) = 600 u.t. // Tp de la ALU para las operaciones/funciones lentas: ADD, SUB, CMP\*.

Tp(ALU-quick) = 400 u.t. // Tp de la ALU para las operaciones/funciones rápidas: cualquier otra distinta de ADD, SUB, CMP\*. Tacc(MEMORY) = 800 u.t. // Tiempo de acceso (para la lectura o escritura) a la memoria Tp(AND-2) = Tp(OR-2) = 20 u.t.

Tp(NOT) = 10 u.t.

El tiempo de propagación de un bloque combinacional (Tp) y el tiempo de acceso a memoria para realizar una lectura (Tacc) es el tiempo desde que están estables todas las entradas necesarias hasta que se estabilizan las salidas requeridas al valor correcto para las entradas aplicadas. Desconocemos como se han implementado internamente los bloques (y podría ser de forma diferente a los vistos en clase). Recordad que un registro con señal de carga (Ld), REGwLd, está construido con un REG y un MUX-2-1 (no os damos el esquema interno del REGwLd, porque lo tenéis que saber).

a) Tc correspondiente al nodo de D (decode). 850UT

1. Tc correspondiente al nodo de Bnz. 700UT
2. Tc correspondiente al nodo de addr. 850UT
3. Tc correspondiente al nodo de Ldb. 1010UT
4. Tc correspondiente al nodo de St. 1010UT
5. Tc correspondiente al nodo de Movi. 730UT

## Problema 7

Modificad el SISC Von Neumann para que sea capaz de ejecutar la nueva instrucción ADD3, además de las 25 instrucciones SISA originales, efectuando las mínimas modificaciones del hardware.

Sintaxis ensamblador: ADD3 Rd, Ra, Rb

Semántica: Rd = Rd + Ra + Rb;

Formato: 1011 aaa bbb ddd xxx

F IR 🡨 MEMw[Pc]// PC 🡨 PC+2

D RX 🡨 RA // RY 🡨RB

ADD31 RB 🡨 RX+RY

ADD32 RZ 🡨 RB // RY🡨 RB

ADD33 RD 🡨 RX + RY

## Problema 8

Modificad el SISC Von Neumann para que sea capaz de ejecutar la nueva instrucción ADD4, además de las 25 instrucciones SISA originales, efectuando las mínimas modificaciones del hardware.

Sintaxis ensamblador: ADD4 Rd, Ra, Rb, Rc

Semántica: Rd = Rd + Ra + Rb + Rc;

Formato: 1011 aaa bbb ddd ccc

F IR 🡨 MEMw[PC]// PC 🡨 PC+2

D RX 🡨 RA // RY 🡨 RB

ADD41 RB 🡨 RX + RY

ADD42 RX 🡨 RA // RY 🡨 RC

ADD43 RB 🡨 RX+RY

ADD44 RX 🡨 RD // RY 🡨 RB

ADD45 RD 🡨 RX+RY

## Problema 9

Modificad el SISC Von Neumann para que sea capaz de ejecutar la nueva instrucción MOVM, además de las 25 instrucciones SISA originales, efectuando las mínimas modificaciones del hardware.

Sintaxis ensamblador: MOVM Rd, Ra, Rb

|  |  |  |
| --- | --- | --- |
| Semántica: |  | RbMEMw[Ra] ; MEMw[Rd]Rb  Copia la palabra de memoria direccionada por Ra a la posición de memoriaque indica Rd, usando Rb como registro temporal. |
| Formato: |  | 1011 aaa bbb ddd xxx |

## Problema 10 (examen E4-17-18-Q2)

Una de las sucesiones más simples y famosas de las matemáticas es la sucesión de Fibonacci (a veces mal llamada serie de Fibonacci) que es la sucesión infinita de números naturales como la siguiente: 0,1,1,2,3,5,8,13,21,34,55,89,144,233,377, … La sucesión comienza con los números 0 y 1, (f0=0 y f1=1) y a partir de estos, cada elemento de la sucesión es la suma de los dos anteriores (fn=fn-1 + fn-2). A los elementos de esta sucesión se les llama números de Fibonacci.

1. Completad el código SISA para que calcule 500 elementos de la sucesión de Fibonacci y los almacene en memoria. Suponed que en la posición de memoria 12346 ya se encuentra el primer valor de la sucesión (f0) y en la siguiente posición, la 12348, el segundo valor (f1). El código debe rellenar las siguientes 500 posiciones de memoria con los siguientes valores de la sucesión. Los valores son números naturales de 16 bits. En caso de que el valor de la sucesión calculado exceda los 16 bits el programa se queda con los 16 bits de menor peso.

Nota: Para saber qué acciones concretas hace el algoritmo os puede ayudar leer el apartado c) de este ejercicio

|  |  |  |
| --- | --- | --- |
| @Mem |  |  |
| 0x0000 | .data  .text            bucle:              .end | MOVIR5, lo(12350) ; R5=primera posición de memoria  MOVHI R5, hi(12350)  MOVI R6, lo(12350) ; R6=última posición de memoria  MOVHIR6, hi(12350)  LD R0, -4(R5) ; primer valor de la sucesión  LD R1, -2(R5) ; segundo valor de la sucesión  ADD R2, R0, R1  ST 0(R5),R2  ADDI R0, R1, 0  ADDI R1, R2, 0  ADDI R5, R5, 2  CMPLE R7, R5, R6  BNZ R7, -7 |
| 0x0001 |
| 0x0002 |
| 0x0003 |
| 0x0004 |
| 0x0005 |
| 0x0006 |
| 0x0007 |
| 0x0008 |
| 0x0009 |
| 0x000A |
| 0x000B |
| 0x000C |
| 0x000D |
| 0x000E |

1. ¿Cuántos ciclos y unidades de tiempo tarda en ejecutarse el código anterior en Harvard uniciclo y en el Von Neumann, considerando que el tiempo de ciclo del Harvard uniciclo es de 4000 u.t. y el del V.Neumann es de 1000 u.t.?

Código en el Harvard uniciclo: Número de ciclos= 3500 Tejec= 1402000

Código en el Von Neumann: Número de ciclos= 11020 Tejec= 11020000

1. Queremos acelerar la ejecución del código. Para ello vamos a modificar el computador Von Neumann para añadirle una nueva instrucción que haga el cálculo de los números de Fibonacci como la siguiente:

Binario: 1011 aaa bbb ddd xxx

Ensamblador: FIB Rd, Ra, Rb

Semántica: Rd = Ra + Rb ; Ra = Rb ; Rb = Rd

Completad el diseño del SISC Von Neumann para que pueda ejecutar, además de las 25 instrucciones originales SISA, la nueva instrucción FIB, que tiene el formato y codificación, la sintaxis ensamblador y la semántica anteriores.

Para su implementación, además de modificar el contenido de la ROM\_OUT y de la ROM\_Q+, cosa imprescindible para añadir una nueva instrucción, se le añade a la unidad de control un MUX-2-1 (con señal de selección Mx, que genera la ROM\_OUT, y que solamente valdrá 1 en alguna de las fases de ejecución de FIB). La ejecución de la instrucción FIB solamente requiere 5 ciclos de ejecución, nodos F, D, Fib1, Fib2 y Fib3. Todos los registros (Ra, Rb y Rd) deben ser distintos para que la instrucción se ejecute correctamente en 5 ciclos. Se pide:

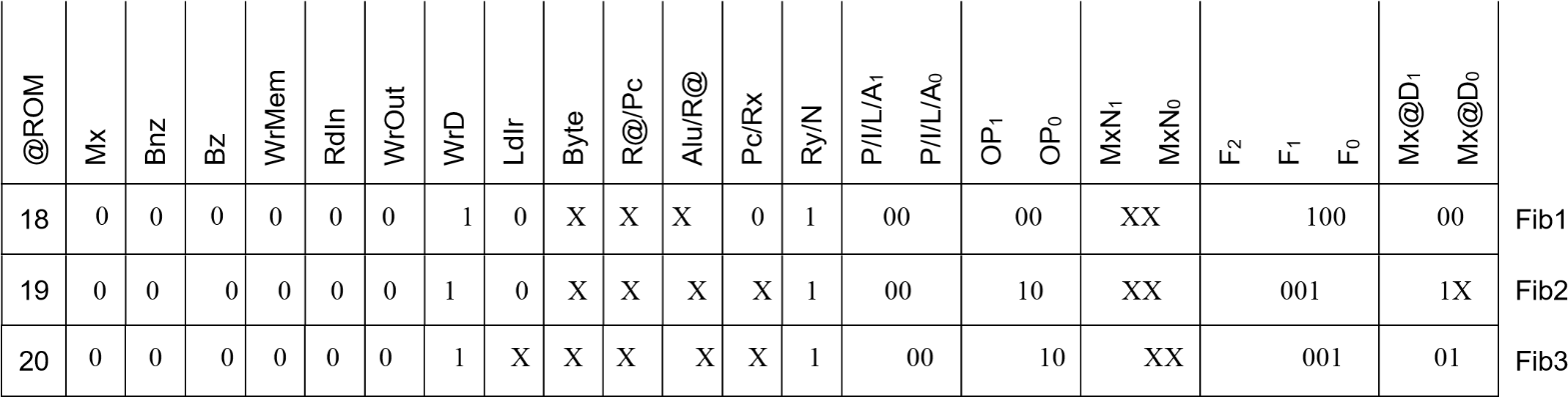
c1) Indica claramente, con texto y/o con un dibujo, donde se conectan las entradas de datos 0 y 1 del MUX-2-1 y donde se conecta su salida, para el correcto funcionamiento del computador con la nueva instrucción FIB. Si se responde incorrectamente este apartado, el resto del apartado (subapartados c2, c3 y c4) se considerará incorrecto.

|  |
| --- |
|  |

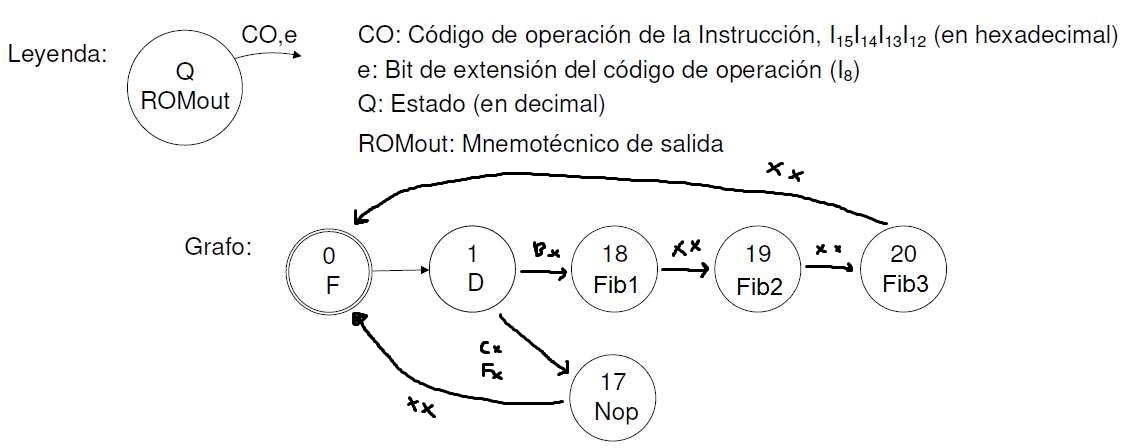
c2) Completad el contenido de la tabla que indica, mediante una fila para cada nodo, la acción o acciones en paralelo que se realiza en el computador en cada uno de los ciclos/nodos que requiere la ejecución, propiamente dicha, de la nueva instrucción (Fib1, Fib2 y Fib3). Usad el mismo lenguaje de transferencia de registros que en la documentación.

|  |  |  |
| --- | --- | --- |
| Nodo / Estado Número | Mnem. | Acciones |
| 18 | Fib1 | Rd 🡨 Rx+Ry // Ry 🡨 Rb |
| 19 | Fib2 | Ra 🡨 Ry // Rx 🡨 Rd |
| 20 | Fib3 | Rb 🡨 Rx |

c3) Completad (poniendo 0, 1 o x en cada bit) las tres filas de la tabla que especifican el contenido de la ROM\_OUT para las direcciones 18 (Fib1) y 19 (Fib2) y 20 (Fib3). Poned x siempre que el valor de un bit no importe.



c4) Completad el fragmento del grafo de estados del circuito secuencial de la unidad de control de la figura. Se da la leyenda del grafo y todos los nodos necesarios para ejecutar las dos nuevas instrucciones, pero faltan arcos y etiquetas. Dibujad todos los arcos que faltan y todas las etiquetas. Os pedimos que no dibujéis ningún otro nodo.



1. Completad la dirección o contenido, según corresponda, de la ROM\_Q+ del SISC Von Neumann.

ROM\_Q+[0x0AC]=0x09 ROM\_Q+[0x 31]=0x0C

1. Reescribe el código correspondiente al bucle del apartado a) que haga la misma tarea con menos instrucciones y usando la nueva instrucción FIB.
2. ¿Cuántos ciclos y unidades de tiempo tarda ahora en ejecutarse el código completo con la modificación escrita en el apartado e) en el nuevo SISC Von Neumann (Tc=1000 u.t.)?

Número de ciclos= 9020 Tejec= 9020000UT