Fundamentos de Computadores 2
Examen Final – 16 septiembre 2020

Nombre:_____ DNI: _____

Apellidos: _____

Tiempo: 2.5 hours

Ejercicio 1 (3.5 puntos).

Queremos construir un programa tal que dada una lista de pares de números enteros (> 0) encuentre qué pares están compuestos de números coprimos (o mutuamente primos). Se entiende que dos números son coprimos si el único divisor común que tienen es el 1.

Asumimos que los datos de entrada están contenidos en una matriz, D, de la forma:

D=(x0, y0, c0, x1, y1, c1, ..., xN-1, yN-1, cN-1), donde cada triplete (xi, yi, ci) se interpreta de la siguiente manera: xi y yi representan un par de números, y ci es 0. Después de ejecutar el programa, el valor de cada ci debe haber sido modificado de tal manera que ci = 2, si xi y yi son coprimas; y ci = 1, en caso contrario.

Por ejemplo, si inicialmente D = (3,5,0, 6,18,0, 15,45,0, 13,10,0, 24,3,0, 24,35,0), el resultado final debe ser D = (3,5,2, 6,18,1, 15,45,1, 13,10,2, 24,3,1, 24,35,2).

a) (1,5 puntos) Escribe un programa en lenguaje ensamblador del ARM que recorre el array D y genera el resultado de acuerdo con la siguiente especificación. El programa llama a la subrutina "check_coprimos (int D[ ], int i)", cuyos argumentos de entrada son la dirección inicial de D y el número del par que queremos chequear (de 0 a M-1). La subrutina verifica si el par i-ésimo contenido en el array D está formado por coprimos y se encarga de almacenar el resultado en la posición de memoria correspondiente. El código debe respetar el estándar de llamadas a subrutinas estudiado en clase.

b) Escribe el código de la subrutina "check_coprimos" en lenguaje ensamblador del ARM, de acuerdo con la siguiente especificación. Suponemos que la subrutina "mcd(int a, int b)" ha sido implementada previamente, y devuelve el máximo común divisor (mcd) de los dos argumentos de entrada. Si el mcd es 1, entonces los números son coprimos. El código debe respetar el estándar de llamadas a subrutinas estudiado en clase.

```
#define M 6
int D[]= {una lista de M*3 valores
enteros}
void main ( ) {
        int i;
        for (i=0; i<M; i++)
                check_coprimos(D,i);
}
```

```
void check_coprimos(int A[ ], int pos) {
        int res;
        res= mcd( A[3*pos], A[(3*pos)+1] );
        if (res == 1)
                A[(3*pos)+2]=2;
        else
                A[(3*pos)+2]=1;
}
```

Ejercicio 2 (3,5 puntos). Estamos interesados en añadir al conjunto de instrucciones del MIPS la nueva instrucción

stradd rs,rt, Immed //Mem(BR(rs)+ SignExt(Immed)) ← (BR(rt) + (PC+4)); PC ← PC+4

La instrucción stradd (dirección de almacenamiento) implementa una operación de almacenamiento en la que la dirección de memoria efectiva se obtiene añadiendo el contenido de un registro base, especificado en el campo rs de la instrucción, más un offset que se especifica en el campo Immed. La dirección del operando que se almacena es el resultado de sumar la dirección de la siguiente instrucción secuencial (PC+4) y el contenido de un registro especificado en el campo rt de la instrucción. Por supuesto, como en cualquier instrucción secuencial, el PC se incrementa en cuatro. El opcode para stradd es '000101'.

a) (1,5 puntos) Modificar el camino de datos para que el computador pueda ejecutar la nueva instrucción.
b) (1 punto) Modificar el Diagrama de Estado para incluir la nueva instrucción.
c) (1 punto) Modificar la Tabla de Verdad de la Unidad de Control para incluir la nueva instrucción (añadir todas las filas y columnas que pueda necesitar para completar el diseño).
nota: Las respuestas sin razonamiento serán ignoradas.

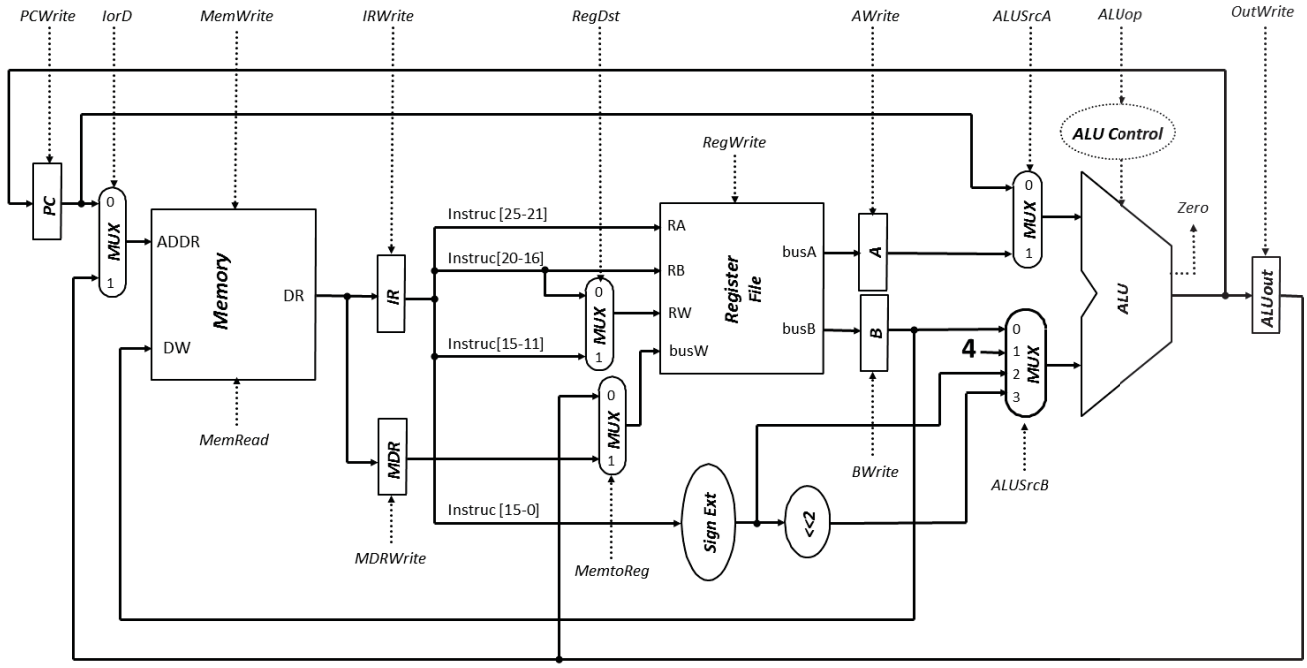
Ejercicio 3 (3 puntos).
Supongamos un computador con un sistema de memoria direccionable por bytes que tenga las siguientes características:
- El procesador genera direcciones de 32 bits.
- El tamaño del bloque es de 64 bytes.
- Hay una caché de 32 Kbytes mapeada directamente.

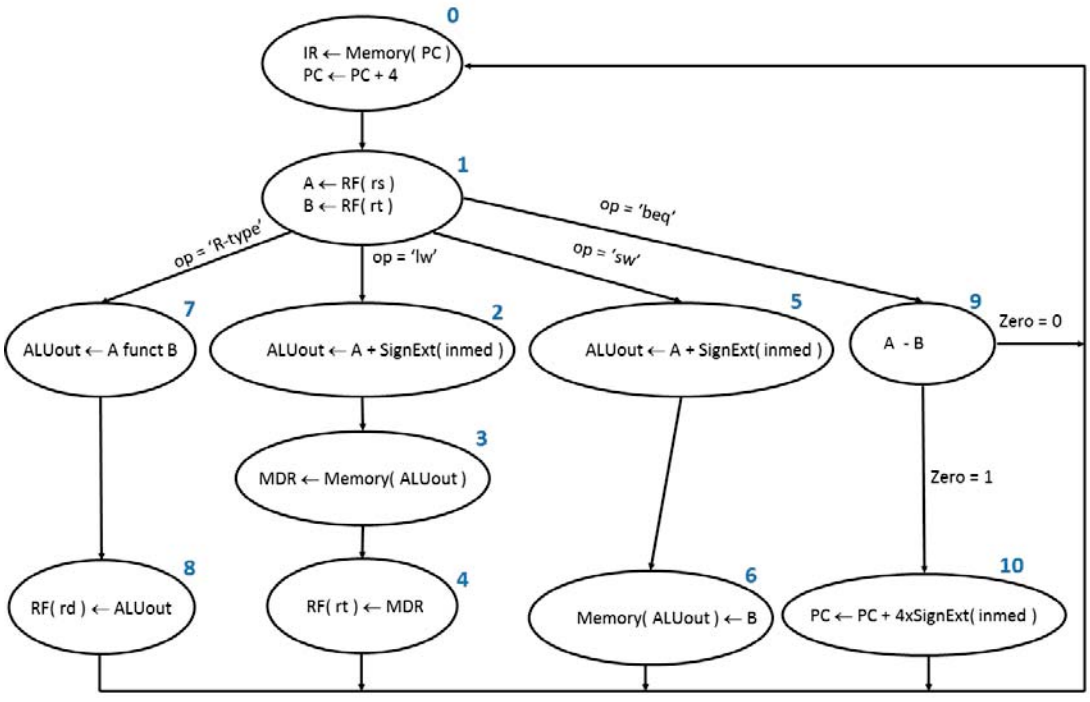
Responda a las siguientes preguntas, explicando los pasos que conducen a cada solución. Las respuestas sin un razonamiento adecuado serán descartadas.
a) (0,25 puntos) Mostrar el formato de dirección para la Memoria Principal (MP) y la caché.
b) (0,5 puntos) Supongamos que el procesador genera la dirección 0x1A0F0846, y la etiqueta correspondiente al bloque $(33)_{10}$ de la caché es actualmente 0x0341E. Explique si habrá un error, un acierto o si no hay suficiente información para dar una respuesta.
c) (1,5 puntos) Suponiendo que la caché esté inicialmente vacía, y que el procesador genera una secuencia de referencias de memoria que va de 0x800 a 0x182F, ambos inclusive, seguido de una secuencia que va de 0x1400 a 0x23B0, nuevamente ambos inclusive, encuentre el número total de fallos de la caché.
d) (0,75 puntos) Para la secuencia completa de referencias definida en la pregunta anterior, encuentre el número total de referencias de memoria y la tasa de aciertos.

Fundamentos de Computadores 2
Examen Final – 16 septiembre 2020

Nombre:_____ DNI: _____

Apellidos: _____





Current state		op	Zero	Next state	IRWrite	PCWrite	AWrite	BWrite	ALUSrcA	ALUScrB	ALUOp	OutWrite	MemWrite	MemRead	IorD	MDRWrite	MemtoReg	RegDest	RegWrite
0000	(fetch)	XXXXXX	X	0001	1	1			0	01	00 (add)		0	1	0				0
0001	(deco)	100011 (lw)	X	0010															
0001	(deco)	101011 (sw)	X	0101															
0001	(deco)	000000 (R-type)	X	0111	0	0	1	1					0	0					0
0001	(deco)	000100 (beq)	X	1001															
0010	(ex-lw)	XXXXXX	X	0011	0	0			1	10	00 (add)	1	0	0					0
0011	(mem-lw)	XXXXXX	X	0100	0	0							0	1	1	1			0
0100	(wb-lw)	XXXXXX	X	0000	0	0							0	0			1	0	1
0101	(ex-sw)	XXXXXX	X	0110	0	0		0	1	10	00 (add)	1	0	0					0
0110	(wb-sw)	XXXXXX	X	0000	0	0							1	0	1				0
0111	(ex-R)	XXXXXX	X	1000	0	0			1	00	10 (funct)	1	0	0					0
1000	(wb-R)	XXXXXX	X	0000	0	0							0	0			0	1	1
1001	(ex-beq)	XXXXXX	0	0000	0	0			1	00	01 (sub)		0	0					0
1001	(ex-beq)	XXXXXX	1	1010															
1010	(wb-beq)	XXXXXX	X	0000	0	1			0	11	00 (add)		0	0					0