2º Grado Informática
Estructura de Computadores
23 de enero de 2017

Test de Teoría (3.0p)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
d	b	b	b	b	b	d	b	c	a	d	bc	d	a	a	c	c	b	a	a	d	d	a	a	d	d	c	d	a	c

cuentan como acierto b,c, como fallo a,d

Test de Prácticas (4.0p)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
a	a	c	c	c	d	c	c	b	b	a	d	d	a	d	c	a	d	d	c

Examen de Problemas (3.0p)

1. Ensamblador (0.7 puntos).

a) Se puntúa **0,6p** por el siguiente programa (**0,05p** por instrucción)

```
        movswl      y, %eax     // Cambio a int y luego a unsigned
        cmpl        x, %eax     // Compara y con x
        jae/jnb     .L2         // Comparación unsigned
        pushl       $hello
        call        printf
        addl        $4, %esp
.L2:
        movl        z, %eax     // Cambio a unsigned
        cmpl        %eax, x     // Compara x con z
        jbe/jna     .L1         // Comparación unsigned
        pushl       $world
        call        printf
        addl        $4, %esp
.L1:
```

b) Imprimiría "**Hello** " (se puntúa **0,1p** si la respuesta es correcta)
1ª comparación: 0xFFFF FFFF > 0xFFFF DEAD
2ª comparación: 0xFFFF FFFF == 0xFFFF FFFF

2. Ensamblador (0.4 puntos).

0.1p

Los dos bucles anidados calculan **4 sumas de bits independientes**, de los bits contenidos en el 1º, 2º, 3º y 4º LSByte.
Las máscaras y acumulación no se aplican tras procesar cada elemento, sino al final del todo, con lo cual no se garantiza que cada una de las 4 sumas quepa en el byte reservado para ella.

Si la suma no se hiciera horizontalmente (elemento a elemento), sino verticalmente (posición de bit a posición de bit) cada posición de bit contiene 2^{19} unos a lo largo del array (mitad ceros, mitad unos), y
cada 8 posiciones contendrían $2^{19}x2^3 = 2^{22}$ bits activados.

0.1p

La cuenta correspondiente al byte menos significativo **(1er LSByte) saldría 0x00 40 00 00**,
no cabe en el 1er LSByte, desborda al 3er LSbyte.
El sumando correspondiente al **2º LSByte** estaría desplazado 1B a la izquierda, **sería 0x40 00 00 00**
Los sumandos correspondientes al **3º y 4º LSByte se perderían** enteros (no caben en 32bits)

0.1p

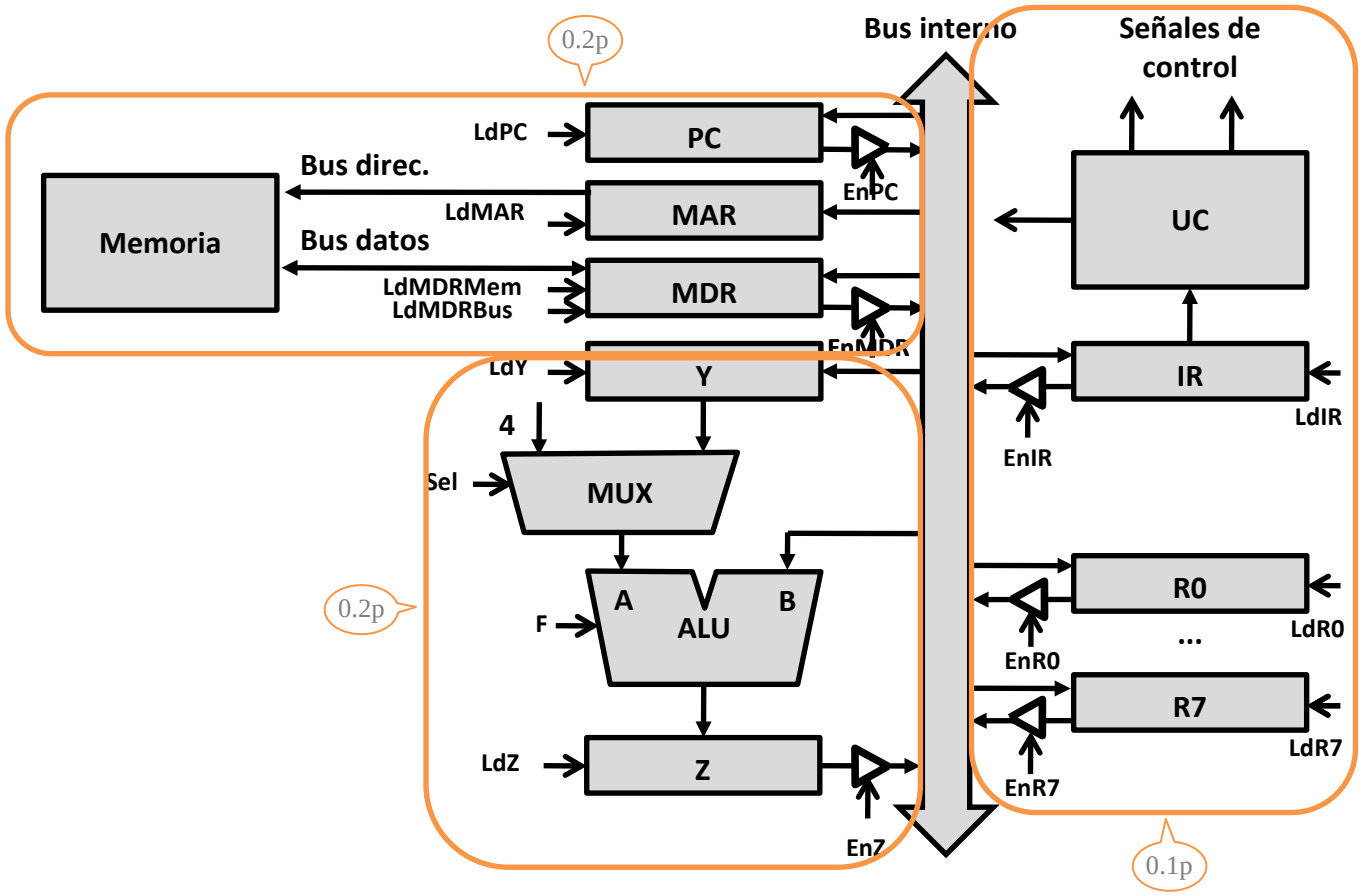
Aunque la suma se haga elemento a elemento (y no posición de byte a posición de byte), el total no cambia.
Cambia el orden en que van perdiéndose "acarreos", pero los acarreos que se pierden en total son los mismos.
Tampoco cambia porque result sea int en lugar de unsigned. Las sumas son las mismas, se crean con signo o sin signo.

El resultado del **bucle for saldría 0x40 40 00 00** (correspondiente al popcount del 1er y 2º bytes menos significativos)
y se transforma a **→ 0x4040 4040 → 0x4080 8080 → 0x0000 0080, es decir, 128**

0.1p

3. **Unidad de Control** (0.5 puntos).

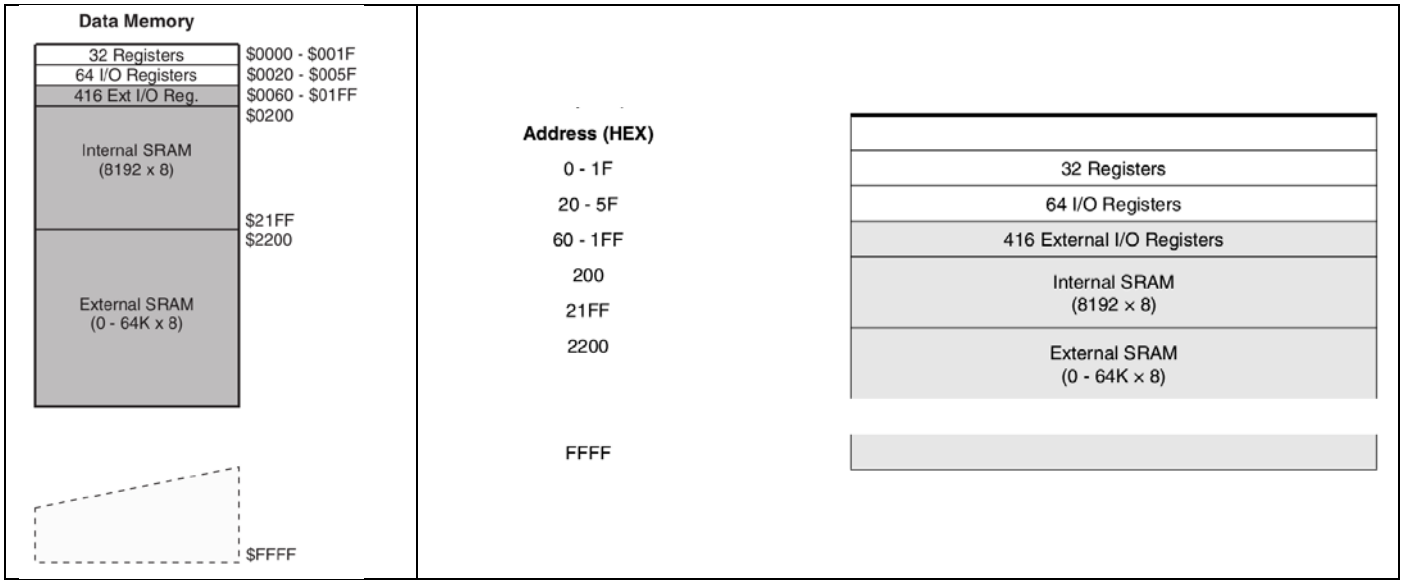
El problema 4 del examen de Septiembre 2016 daba el camino de datos y pedía el pseudocódigo para BRANCH. Esta pregunta es la inversa. La respuesta es aquél camino de datos.



Posiblemente se podría mejorar el dibujo desdoblando en MDR “Bus datos” igual que se desdobló “Bus interno”

4. **Diseño de E/S y memoria** (0.5 puntos).

a) Se puntúa **0,25p** por cualquier dibujo similar a los siguientes, **0,05p** por cada una de las 5 zonas



Referencias:

Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V datasheet:
http://www.atmel.com/Images/Atmel-2549-8-bit-AVR-Microcontroller-ATmega640-1280-1281-2560-2561_datasheet.pdf
ATmega1281/2561/V ATmega640/1280/2560/V datasheet:
http://pdf.datasheetcatalog.com/datasheets2/30/3055029_1.pdf

b) El chip dispone de patillas /CE y /OE ⇒ **SRAM (0,05 p.)**
A0-A18 (19 líneas de dirección) + I/O1-I/O8 (8 líneas de datos) ==> 2^{19} x 8 = **512 K x 8 (0,1 p.)**
2^{19} x 8 = 2^{19} x 2^3 = 2^{22} = **4 Mbits** (4194304 bits) **(0,1 p.)**

Referencias:
QuadRAM Shield for the Arduino Mega/Mega2560:
https://www.rugged-circuits.com/new-products/quadram
Arduino Mega 512K SRAM in shield format:
http://andybrown.me.uk/2013/05/05/512k-xmem-in-shield-format/
AS7C4096A - Alliance Memory:
http://www.alliancememory.com/pdf/sram/fa/as7c4096a_v1.2.pdf

5. **Memoria cache** (0.9 puntos).

a) L3 **(0,3p)**

MP: 64 GB = **2^{36} B**, L3: 8 MB = 2^{23} B, 64 B/línea = 2^6 B/línea, L3: 16 vías = 2^4 líneas/conjunto
L3: 2^{23} B / 2^6 B/línea = 2^{17} líneas, 2^{17} líneas / 2^4 líneas/conjunto = 2^{13} conjuntos
Dirección física de memoria principal desde el punto de vista de L3: **(0,15p, 0,05p cada campo)**

etiqueta (17)	**conjunto (13)**	**byte (6)**

Tamaño total en bits ocupado por todas las etiquetas en directorios L3: **(0,05p)**
1 cache · 16 vías/cache · 8K etiquetas/vía · 17 bits/etiqueta = **2^{17}x17 bits = 2 228 224 bits**
Tamaño total en bits ocupado por todos los datos/instrucciones en L3: **(0,05p)**
1 cache · 8MB · 8 bits/B = 2^{23}x2^3 bits = **2^{26} bits = 64Mbits = 67 108 864 bits**
Etiquetas / Datos = **3,32%** **(0,05p)**

b) L2 **(0,3p)**

L3: 8 MB = 2^{23} B, L2: 256 KB = 2^{18} B, 64 B/línea = 2^6 B/línea, L2: 4 vías = 2^2 líneas/conjunto
L2: 2^{18} B / 2^6 B/línea = 2^{12} líneas, 2^{12} líneas / 2^2 líneas/conjunto = 2^{10} conjuntos
Dirección física de memoria principal desde el punto de vista de una L2:

etiqueta (20)	**conjunto (10)**	**byte (6)**

Tamaño total en bits ocupado por todas las etiquetas en directorios L2:
4 caches • 4 vías/cache • 1K etiquetas/vía • 20 bits/etiqueta = **2^{14}x20 bits = 327 680 bits**
Tamaño total en bits ocupado por todos los datos/instrucciones en L2:
4 caches • 256KB/cache • 8 bits/B = 2^2x2^{18}x23 bits = **2^{23} bits = 8Mbits = 8 388 608 bits**
Etiquetas / Datos = **3,91%**

c) L1 **(0,3p)**

L2: 256 KB = 2^{18} B, L1: 32 KB = 2^{15} B, 64 B/línea = 2^6 B/línea, L2: 8 vías = 2^3 líneas/conjunto
L1: 2^{15} B / 2^6 B/línea = 2^9 líneas, 2^9 líneas / 2^3 líneas/conjunto = 2^6 conjuntos
Dirección física de memoria principal desde el punto de vista de una L1:

etiqueta (24)	**conjunto (6)**	**byte (6)**

Tamaño total en bits ocupado por totas las etiquetas en directorios L1:
8 caches • 8 vías/cache • 64 etiquetas/vía • 24 bits/etiqueta = **2^{12}x24 bits = 98 304 bits**
Tamaño total en bits ocupado por todos los datos/instrucciones en L1:
8 caches • 32KB/cache • 8 bits/B = 2^3x2^{15}x2^3 bits = **2^{21} bits = 2Mbits = 2 097 152 bits**
Etiquetas / Datos = **4,69%**

Referencias:
Intel Core i7-6700K specifications en CPU-World:
http://www.cpu-world.com/CPUs/Core_i7/Intel-Core%20i7-6700K.html
Cache: A Place for Concealment and Safekeeping:
http://duartes.org/gustavo/blog/post/intel-cpu-caches/