Nombre:	
DNI:	Grupo:

Examen de Problemas (3,0 p)

1. Ensamblador (0.7 puntos). Dado el siguiente programa escrito en C:

```
#include <stdio.h>

unsigned int x;
unsigned short y;
signed int z;

void main (void) {
if (x > (signed short) y)
    printf ("Hello ");
if (x > z)
    printf ("world");
}
```

a) Escriba la función **main()** en ensamblador de IA32. Para ello, tenga en cuenta estas aclaraciones:

1. En comparaciones con tamaños diferentes, se pasa el dato de menor tamaño al tamaño del mayor.
2. En comparaciones que implican **unsigned** y **signed**, se pasa el **signed** a **unsigned**.
3. Si hay que realizar un cambio de tamaño y de "signedness" (de **signed** a **unsigned** o viceversa), primero se realiza el cambio de tamaño manteniendo el signo y luego el de "signedness".

Ayuda:

- La instrucción **movzwl** ***src,dst*** pasa de 16 a 32 bits añadiendo ceros por la izquierda
- La instrucción **movswl** ***src,dst*** pasa de 16 a 32 bits extendiendo el signo
- Las instrucciones se salto condicional para **unsigned** usan los sufijos *a* (above) y *b* (below)

b) Suponiendo que las variables *x*, *y* y *z* estuvieran inicializadas así:

```
unsigned int x = 0xFFFFFFFF;
unsigned short y = 0xDEAD;
signed int z = -1;
```

¿qué imprimiría el programa?

2. Ensamblador. (0.4 puntos). La práctica "popcount" debía calcular la suma de bits (peso Hamming) de los elementos de un array. Un estudiante entrega la siguiente versión de popcount4:

```
int popcount4(unsigned* array, int len) {
    int i,j;
    unsigned x;
    int result = 0;

    for(i=0;i<len;i++){
        x=array[i];
        for(j=0;j<8;j++){
            result += x & 0x01010101;
            x>>=1;
        }
    }
    result += (result >> 16);
    result += (result >> 8);

    return result & 0xFF;
}
```

Esta función presenta varias diferencias con la versión "oficial" recomendada en clase, incluyendo la ausencia de una variable auxiliar **val** y la diferente posición de los desplazamientos >> y máscara **0xFF**.

Esta función popcount4 produce resultado 128 con el último ejemplo "grande" de la batería de tests, que inicializa a lista[i]=i un array de 2^{20} elementos pasado como argumento (`popcount4(lista, SIZE)`). Explicar por qué.

(Ayuda: calcular las sumas de bits verticalmente, en lugar de horizontalmente. Si el tamaño del array es potencia de dos, la mitad de los bits de cada columna son 0, y la mitad son 1).

3. Unidad de control (0.5 puntos). Suponga una unidad de procesamiento con un bus interno que está conectado a los registros PC, MAR, MDR, Y, Z, IR y R0...R7. También contiene una ALU con una entrada conectada a "4" o al registro Y por medio de un multiplexor, la otra entrada conectada directamente al bus, y la salida conectada al registro Z. El bus interno y todos los registros tienen un tamaño de 32 bits. El desplazamiento es un número con signo relativo al contador de programa que ocupa los 24 bits menos significativos del registro IR cuando la instrucción está en dicho registro, y la salida del registro IR hacia el bus interno es ese campo desplazamiento con el signo extendido a 32 bits. El código que implementa la ejecución de la instrucción de una palabra BRANCH *desplazamiento*, incluyendo la fase de captura de instrucción es el siguiente:

```
fetch:   MAR:=PC; Z:=PC+4;
         PC:=Z; Y:=Z; MDR:=M[MAR];
         IR:=MDR;
         goto f(IR);
branch:  Z:=Y+IR;
         PC:=Z; goto fetch;
```

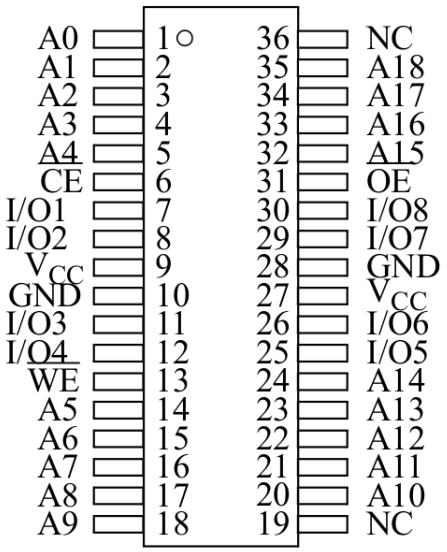
Dibuje el camino de datos de esta unidad de procesamiento, incluyendo registros, ALU, multiplexor, buses, posibles buffers triestado, y señales de control etiquetadas.

4. Diseño de E/S y memoria (0.5 puntos). El microcontrolador ATmega2560 (utilizado en algunas placas Arduino Mega) tiene una arquitectura Harvard (espacios de memoria de programa y de datos separados). El espacio de memoria de datos y E/S mapeada en memoria es direccionable por bytes y se conecta al microcontrolador a través de un bus de direcciones de 16 bits y un bus de datos de 8 bits. El mapa de memoria-E/S incluye las siguientes zonas, en este orden:

- Registros de trabajo de propósito general: 32 registros de 8 bits accesibles a través de las 32 primeras direcciones de memoria
- E/S: 64 direcciones de puertos de E/S, accesibles por instrucciones in/out y load/store
- E/S ampliada: 416 direcciones de puertos de E/S, accesibles por instrucciones load/store
- SRAM interna: 8 KB
- Espacio de memoria SRAM externa opcional: resto del mapa de memoria (55.5 KB)

a. (0.25 puntos) Dibuje el mapa de memoria-E/S, indicando en hexadecimal la primera y la última dirección de cada zona.

b. (0.25 puntos) Existen ampliaciones de memoria externa de un tamaño mayor que el que permite directamente el microcontrolador gracias a la técnica de conmutación de bancos. Algunas de estas ampliaciones usan el chip de memoria AS7C4096A de la figura. Indique el tipo de memoria de este chip (SRAM o DRAM), su configuración expresada como n.º de direcciones x n.º de datos, y su tamaño total en bits.



5. Memoria cache (0.9 puntos). En la web CPU-World podemos encontrar las siguientes características sobre la jerarquía de memoria del microprocesador i7-6700K, de 4 núcleos (tamaño de línea 64 B):

Level 1 cache size	4 x 32 KB 8-way set associative instruction caches 4 x 32 KB 8-way set associative data caches
Level 2 cache size	4 x 256 KB 4-way set associative caches
Level 3 cache size	8 MB 16-way set associative shared cache
Physical memory	64 GB

Indique el nombre y tamaño en bits de los campos de dirección usados para la política de correspondencia, así como el tamaño total en bits de todas las memorias de etiquetas, tamaño total en bits de todas las memorias de datos/instrucciones, y porcentaje de espacio de etiquetas respecto a datos/instrucciones para cada uno de los tres casos siguientes:

a) Dirección física de memoria principal desde el punto de vista de L3:

Tamaño total en bits ocupado por todas las etiquetas en directorios L3:
Tamaño total en bits ocupado por todos los datos/instrucciones en L3:
Porcentaje de espacio ocupado por etiquetas respecto a datos/instrucciones en L3:

b) Dirección física de memoria principal desde el punto de vista de una L2:

Tamaño total en bits ocupado por todas las etiquetas en directorios L2:
Tamaño total en bits ocupado por todos los datos/instrucciones en L2:
Porcentaje de espacio ocupado por etiquetas respecto a datos/instrucciones en L2:

c) Dirección física de memoria principal desde el punto de vista de una L1:

Tamaño total en bits ocupado por totas las etiquetas en directorios L1:
Tamaño total en bits ocupado por todos los datos/instrucciones en L1:
Porcentaje de espacio ocupado por etiquetas respecto a datos/instrucciones en L1: