eman ta zabal zazu
universidad euskal herriko
del país vasco unibertsitatea

departamento de electricidad y electrónica
elektrika eta elektronika saila

Examen de "S.E.T.I."

1er curso de Ingeniería Electrónica
1 de junio de 2000

Cuestiones teóricas. *(1 punto cada una)*

1 La Máquina Universal de Turing es (explique la respuesta)

A - una generalización de la Máquina de Turing
B - una máquina de Turing particular
C - ambas cosas
D - nada de lo anterior es correcto.

2 ¿No sería interesante simplificar la fórmula general de la representación en coma flotante del siguiente modo? : (razone la respuesta)

$(-1)^{signo}\ 2^{exponente-bias}\ mantisa$ si exponente $\neq 2^{be}-1$
∞ si exponente $= 2^{be}-1$ y mantisa$=0$
NaN si exponente $= 2^{be}-1$ y mantisa$\neq 0$

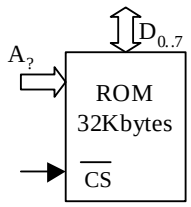
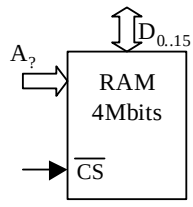
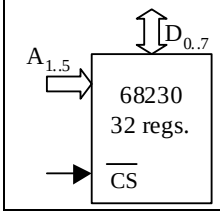
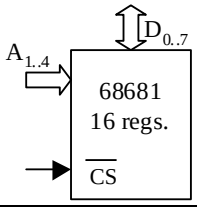
(be = número de bits del exponente)

3 Uno de los aspectos que teóricamente se contempla dentro de la arquitectura RISC consiste en primar las operaciones con operandos internos (registros del procesador) y limitar las capacidades de acceso externo (fuera del procesador, es decir a direcciones de memoria). Al proporcionar un elevado número de registros internos se considera que esta característica supone una ventaja importante sobre la aproximación CISC (con más capacidad de direccionamiento y menos registros). Explique porqué este enfoque permite confeccionar programas más rápidos (quizás un ejemplo sea aclaratorio).
(Se le ruega no explique otras características de la arquitectura RISC)

Cuestiones prácticas. *([3+1]+1+4 puntos)* [1]

4 Diseñe el circuito necesario para establecer el mapa de memoria de una computadora con 68000 según los dos requerimientos siguientes:

Requerimiento A:

<table>
<tr><td>A partir de 000000_{Hex}</td><td></td><td>A partir de F00000 _{Hex}</td><td></td></tr>
<tr><td> 
</td><td>64 Kbytes de memoria ROM utilizando circuitos de 32Kbytes (estructuradas en bytes 8 líneas de datos)</td><td> 
</td><td>1 Mbyte de memoria RAM utilizando circuitos de 4Mbits (estructuradas en words16 líneas de datos)</td></tr>
<tr><td>A partir de 010000 _{Hex} y en ningún caso fuera de 010000 _{Hex} -0100FF _{Hex}</td><td></td><td>A partir de 010100 _{Hex} y en ningún caso fuera de 010100 _{Hex} -0101FF _{Hex}</td><td></td></tr>
<tr><td> 
</td><td>Un circuito 68230</td><td> 
</td><td>Un circuito 68681</td></tr>
</table>

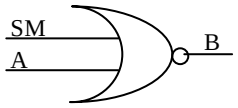
[1] La puntuación de las preguntas hace que se pueda llegar a obtener incluso un 12. Toda calificación que pase de 10 se dejará en 10, y las demás no serán alteradas.

Requerimiento B:
En función del estado de una línea que denominaremos SM (Selección de Modo) podremos optar por:

A.- Todo el mapa de memoria es accesible tanto en modo usuario como supervisor

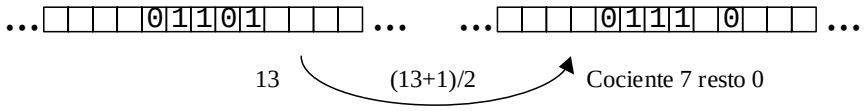
B.- La memoria ROM, 68230, 68681 y la memoria RAM con dirección más alta serán sólo accesibles por el supervisor, mientras que el resto de la RAM será accesible tanto en usuario como en supervisor.

Nota: Para aquellos alumnos que no hayan cursado la correspondiente asignatura, y en general como ayuda se muestra la siguiente puerta lógica que puede ser de utilidad:



SM	A	B
0	0	1
0	1	0
1	X	0

5 Escriba la tabla de una máquina de Turing que, dada una cantidad en binario (sea x) que se encuentra sobre una cinta en blanco, la altera dejando (x+1)/2 como cociente y resto separados por un espacio. En la posición inicial el cabezal de la máquina se encuentra sobre el bit menos significativo. (8 líneas son suficientes).



6 PROBLEMA DE PROGRAMACIÓN

La encriptación es una técnica mediante la cual nos es posible guardar de forma "segura" documentos aun cuando una tercera persona pueda tener acceso a los mismos. Se trata de codificar un fichero en base a algún algoritmo reversible, de tal forma que podamos más tarde desencriptarlo. Una estrategia sencilla para encriptar un archivo de texto consiste en desplazar una cantidad fija el código ascii de cada carácter en el texto:

$$y = x + d$$

ESTOESELTEXTOORIGINAL → (d=2) → GUVQGUGNVGZVQQTKIKPCN

En este caso hemos desplazado los códigos ascii dos posiciones. Sin embargo, existen técnicas algo más eficientes. Es deseable que un algoritmo de encriptación no use siempre las mismas variables (el desplazamiento de 2 en el caso anterior). Esto se resuelve mediante la obtención automática de los parámetros a través de una clave: cuando encriptemos un documento se nos pedirá una clave que servirá a su vez para desencriptarlo. Una solución sencilla para el caso anterior, pudiera ser tomar como parámetro de desplazamiento la suma de los caracteres de la clave (truncado a un byte):

ESTOESELTEXTOORIGINAL → (clave:"secret" → d=19) → XfgbXfX gXkgbbe\Z\aT

El algoritmo de encriptación que se pide realizar va a depender de dos parámetros, los cuales se obtendrán directamente de la clave (más adelante se explica cómo). El algoritmo tiene la siguiente fórmula:

$$y_n = x_n + a + nb$$

Lo que representa esta fórmula, es que a cada carácter se le suma un desplazamiento variable dependiendo de su índice, de tal forma que al primer carácter se le suma el desplazamiento (a+b), al segundo (a+2b)... . Para obtener los parámetros a y b, se utilizara la siguiente técnica:

- Se establece un valor UMBRAL dentro del rango [0,255] (byte).
- $a = \sum_{\forall c_i > UMBRAL} c_i$, siendo c_i los caracteres de la clave
- $b = \sum_{\forall c_i \leq UMBRAL} c_i$, siendo c_i los caracteres de la clave

Es decir, guardamos en "a" la suma de los caracteres que superiores al umbral, y en b la suma de los demás.

Los pasos a seguir serán:

1.- Obtener los parámetros a partir de la clave.
2.- Encriptar una cadena de caracteres.

Para ello, se deberán realizar las siguientes subrutinas y programa principal:

Subrutina SETPARAM

Esta subrutina deberá tomar la dirección de una cadena de caracteres (terminará con 0) y haciendo uso de la variable global UMBRAL (byte), devolver en las posiciones PARAM_A (byte) y PARAM_B (byte) (variables globales en posiciones fijas de memoria) los valores de los parámetros a y b. La subrutina podría tener en concreto la siguiente descripción:

```
; Subrutina:    SETPARAM
; Descripción:  Obtiene los parámetros PARAM_A(byte), PARAM_B(byte) a partir de
;               una cadena de caracteres.
; Entrada:      A0.L  direccion de la clave  (cadena de caracteres concluida en 0)
;               UMBRAL (byte) variable global valor umbral.
; Salida:       PARAM_A (byte) variable global parámetro "a".
;               PARAM_B (byte) variable global parámetro "b".
; Modifica:     CCR,AO,..... (lo que sea necesario)
```

Subrutina ENCRIPTCHAR

Esta subrutina deberá tomar como entrada un carácter a encriptar, así como su índice. Deberá aplicar la fórmula de encriptación $y = x + a + nb$, haciendo uso de las variables globales PARAM_A, PARAM_B. Asimismo, deberá devolver el carácter ya encriptado. La subrutina podría tener en concreto la siguiente descripción:

```
; Subrutina:    ENCRIPTCHAR
; Descripción:  Encripta un carácter
; Entrada:      D0.B      carácter
;               D1.W      índice n
;               PARAM_A (byte) variable global parámetro "a".
;               PARAM_B (byte) variable global parámetro "b".
; Salida:       D0.B      carácter encriptado
; Modifica:     CCR, D0, ... (lo que sea necesario)
```

Subrutina ENCRIPTSTRING

Esta subrutina deberá tomar como entrada la dirección de la cadena que se desea encriptar, y deberá encriptarla haciendo uso de la rutina ENCRIPTCHAR. Se entiende por encriptar el cambiar cada carácter de la cadena por el correspondiente encriptado. La subrutina podría tener en concreto la siguiente descripción:

```
; Subrutina:     ENCRIPTSTRING
; Descripción:   Encripta una cadena de caracteres
; Entradas:      A0.L       Dirección de la cadena
; Modifica:      CCR, A0, ...(lo que sea necesario)
```

Programa principal

El programa principal deberá coordinar todas las acciones. Deberá declarar las variables PARAM_A(Byte), PARAM_B(Byte), el valor UMBRAL(Byte), la clave y la propia cadena a encriptar. Se ocupará de inicializar los parámetros mediante la subrutina GETPARAM, y después deberá llamar a la subrutina ENCRIPTSTRING. El programa principal podría tener en concreto las siguientes partes:

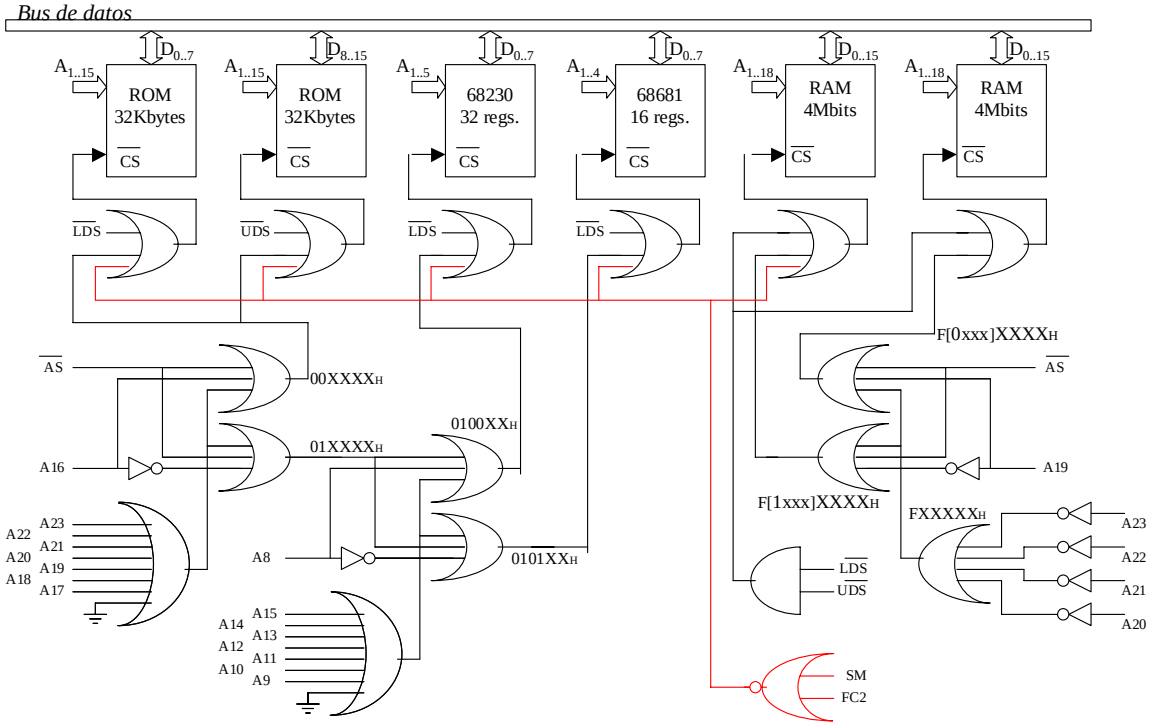
```
INCLUDE /usr/68k/semones.i

SECTION 0
Inicialización de los parámetros de entrada antes de la llamada a subrutina
Llamada a SETPARAM
Inicialización de los parámetros de entrada antes de la llamada a subrutina
Llamada a ENCRIPTSTRING
Final

SECTION 1
Declaración de todas las variables globales y cadenas de texto.
```

Examen de "S.E.T.I."

1er curso de Ingeniería Electrónica
1 de junio de 2000

Soluciones a las cuestiones prácticas

4 El enunciado plantea un requerimiento incorrecto en lo referente a la RAM. No es posible utilizar RAM de ancho 16 bits con un 68000 porque no se ejecutarán correctamente las escrituras de byte, no obstante el "mapeo" pedido se obtendría del siguiente modo:



5

Estado	Lee	Escribe	Movimiento	Nuevo estado
0	0	#	D	0
0	#	1	I	H
0	1	#	D	1
1	#	0	I	2
2	#	#	I	3
3	0	1	I	H
3	1	0	I	3
3	#	1	I	H

espacio en blanco; D.- derecha; I.- Izquierda; H.- parada (halt)

Estado 0: Elimina el bit menos significativo dejando un blanco. Si era 0 va a la derecha y pone 1 (el resto) y termina (el cociente ya es correcto). Si era 1 va a poner resto 0 (estado 1).
Estado 1: Pone 0 (el resto) y va a localizar el cociente para incrementarlo (estado 2).
Estado 2: Salta el espacio en blanco y se sitúa en el bit menos significativo del cociente para incrementarlo (estado3).
Estado 3: Si lee 0 o # pone uno (el incremento) y termina. Si lee 1 pone 0 y continua hacia la izquierda (hay llevada).

SOLUCIONES AL PROBLEMA DE PROGRAMACIÓN

Subrutina SETPARAM

```
; Subrutina:      SETPARAM
; Descripción:    Obtiene los parámetros PARAM_A(byte), PARAM_B(byte) a partir de
;                 una cadena de caracteres.
; Entrada:        A0.L  direccion de la clave  (cadena de caracteres concluida en 0)
;                 UMBRAL          valor umbral para contribución a "a" o "b"
; Salida:         PARAM_A         parámetro a
;                 PARAM_B         parámetro  b
; Modifica:       CCR,AO,D0,D1,D2
```

```
SETPARAM  CLR.B   D0            ;  d0=a=0
          CLR.B   D1            ;  d1=b=0
          MOVE.B  UMBRAL,D2     ;  d2=umbral
          BRA.S   WHILE         ;  mientras c ≠ 0
DO        CMP.B   (A0),D2       ;      si ( c ≤ umbral )
          BLO.S   SUMA_A        ;
SUMA_B    ADD.B   (A0)+,D1      ;           entonces   b += c
          BRA.S   WHILE         ;
SUMA_A    ADD.B   (A0)+,D0      ;           sino       a += c
WHILE     TST.B   (A0)          ;
          BNE.S   DO            ;
          MOVE.B  D0,PARAM_A    ;  PARAM_A=d0
          MOVE.B  D1,PARAM_B    ;  PARAM_B=d1
          RTS                   ;  termina
```

Subrutina ENCRIPTCHAR

```
; Subrutina:      ENCRIPTCHAR
; Descripción:    Encripta un caracter
; Entrada:        D0.B      carácter
;                 D1.W      índice n
;                 PARAM_B a
;                 PARAM_B b
; Salida:         D0.B      carácter encriptado
; Modifica:       CCR, D0, D2
```

```
ENCRIPTCHAR  CLR.W   D2           ; dejo a cero la WORD d2
             MOVE.B  PARAM_B,D2   ; d2.W = b
             MULU    D1,D2        ; d2 = n x b
             ADD.B   PARAM_A,D2   ; d2 = a + n x b
             ADD.B   D2,D0        ; c += d2
             RTS                  ; termina
```

Subrutina ENCRIPTSTRING

; Subrutina: ENCRIPTSTRING
; Descripción: Encripta una cadena de caracteres
; Entradas: A0.L Dirección de la cadena
; Modifica: CCR, A0,

```
ENCRIPTSTRING CLR.W   D1              ; n = 0
              BRA.S   WHILE           ; mientras  c ≠ 0
DO            ADDQ.W  #1,D1           ;     n++
              MOVE.B  (A0),D0         ;     d0 = c
              BSR.S   ENCRIPTCHAR     ;     d0 = encriptchar(c)
              MOVE.B  D0,(A0)+        ;     c = d0
WHILE         TST.B   (A0)            ;
              BNE.S   DO              ;
              RTS                     ; termina
```

Programa principal

```
              INCLUDE /usr/68k/semones.i

              SECTION 0
              LEA     CLAVE,A0        ; cargo en A0 la dirección de la clave
              BRA.S   SETPARAM        ; obtengo los parámetros a y b
              LEA     CADENA,A0       ; cargo en A0 la dirección de la cadena
              BRA.S   ENCRIPTSTRING   ; encripto la cadena
              SERMON  FPROG           ; termina el programa


              SECTION 1
CADENA        DC.B    'Esta es la cadena a encriptar',0
CLAVE         DC.B    'miclave',0
PARAM_A       DS.B    1
PARAM_B       DS.B    1
UMBRAL        DC.B    58
```