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
24 de enero de 2008

Ejercicio de C.	**PARTE 1**	1 punto

1 Escríbase un programa que imprima los números primos existentes en el rango [1,MAX], utilizando para ello el algoritmo conocido como criba de Eratóstenes.

Dicho algoritmo parte de un conjunto formado por todos los números enteros de dicho rango, del cual se van eliminando sucesivamente los múltiplos de 2, 3,. . . continuando por orden con los números que van quedando en el conjunto. Así, tras eliminar los múltiplos de 2 y de 3, a continuación se eliminarán los múltiplos de 5, ya que el número 4 habrá sido eliminado previamente al ser múltiplo de 2.

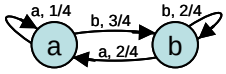
Sugerencias de implementación:

- Defínase MAX como una macro.
- Para representar el conjunto de números, y puesto que el lenguaje C no cuenta con un tipo conjunto, utilícese un array de char (que por ejemplo podríamos llamar cedazo), donde cada elemento tomará sólo los valores booleanos 1 o 0 para indicar respectivamente la presencia o ausencia del número correspondiente. Así, si cedazo[n] contiene 1, ello significa que n está en el conjunto.
- Puesto que C indexa los arrays a partir de cero, y para evitar confusiones, defínase el array cedazo con MAX+1 elementos, dejando sin uso el elemento cedazo[0].

Cuestiones.	**PARTE 2**	1,25 + 1,25 puntos

2 Imagine que tiene que dar una charla extremadamente corta sobre "Situación actual de la codificación de documentos" (ojo, no confundir con encriptación). Ponga por escrito lo que diría.

3 Dada la fuente de la figura, podemos determinar cómo será la fuente extensión de orden 2 correspondiente. Dibuje su grafo.



Estructura de un microprocesador.	1,25 punto

4 Nuestro computador virtual es extremadamente simple. Entre otras muchas capacidades necesarias en un procesador real, carece de la posibilidad de atender a interrupciones. Explique qué cambios serían necesarios para introducir dicha capacidad (instrucciones, registros, estructuras, circuitos,… todo lo que estime necesario). No es preciso diseñar un nuevo sistema, sino explicar, con un nivel de detalle razonablemente suficiente, las acciones necesarias para añadir la capacidad pedida.

(Entienda que esto puede tener sentido incluso para un simulador, como es el caso. Podríamos considerar p.ej. el teclado como un periférico que genera interrupciones y por tanto atenderlo con la rutina correspondiente en caso de pulsar una tecla, mientras se simula un programa)

Mapeo de memoria.	1,25 puntos

5 Disponemos de una vieja placa de computador con 68000 con un conector para extensión con todos los buses, y de cuatro circuitos integrados de memoria RAM que vamos a aprovechar para añadirla al sistema mediante una pequeña placa de circuito impreso. Estos circuitos son: 2 RAM de 256Kbytes cada una, estructuradas en niblas, y 2 RAM de 256Kbytes cada una, estructuradas en bytes

El sistema dispone de 1MByte de ROM en las posiciones iniciales, 1MByte de RAM en las finales, y pastillas de periféricos ocupando las zonas 80XXXX y 81XXXX.

Debemos "mapear" nuestras memorias del modo que nos parezca más conveniente.

Dibuje cómo quedará el mapa de memoria con la inclusión que decida realizar y el circuito correspondiente.

Programación del 68000.	**PARTE 3**	4 puntos (1 + 1 + 1 + 1)

6 En una estructura en memoria se situa una lista de notas obtenidas por todos los alumnos de una asignatura para cada pregunta de un examen del modo que se muestra en el siguiente ejemplo:

```
Evaluacion:    dc.b    (Fin_evaluacion-Evaluacion)/10-1; número de alumnos
               dc.b    7,3,6,4,3,5,2,5,9,8    ; notas de las 10 preguntas para el primer alumno 1
               …       …                      ; …
               dc.b    3,6,4,8,4,6,8,5,3,3    ; notas de las 10 preguntas para el último alumno
Fin_Evaluacion:
```

Se pide que se escriban las siguientes subrutinas y programa principal:

```
;Subrutina NOTA_ALUMNO
;Calcula la nota de un alumno como la media de sus calificaciones para las 10 preguntas del examen.
;El resultado será un byte donde los 4 bits más altos contendrán la parte entera de la nota y los 4
;más bajos un decimal.
; (p. ej. 0,0 [0000 0000 = 00hex]; 5,2 [0101 0010 = 52hex]; 10.0 [1010 0000 = A0hex])
;Entrada:    A0.L  dirección de su lista de notas por pregunta
;Salida:     D0.B  nota obtenida (según formato indicado)
;Modifica:   CCR…

;Subrutina NOTAS_FINALES
;Calcula y genera en memoria la lista de notas finales para todos los alumnos con el formato
;devuelto por la rutina NOTAS_ALUMNO
;Entrada:    A0.L  dirección de la lista de notas pormenorizada de todos los alumnos
;            A1.L  dirección de la lista de notas finales.
;Modifica:   CCR…

;Subrutina HISTOGRAMA
;Situa en 10 posiciones de memoria consecutivas, asociadas a los rangos [0-1),[1-2),..,[9-10),
;[10-10] de las notas obtenidas por los alumnos, la cuenta del número de ellos que han obtenido nota
;en dicho rango
;Entrada:    A0.L  dirección de la lista de notas finales de todos los alumnos.
;            A1.L  dirección del histograma
;Modifica:   CCR…
```

Programa principal

En la zona correspondiente al programa de prueba deberá definirse la tabla de notas, reservarse espacio para la tablas de notas finales y el histograma y escribirse el programa principal que ejecutará el cálculo de las notas finales y del histograma .

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
24 de enero de 2008

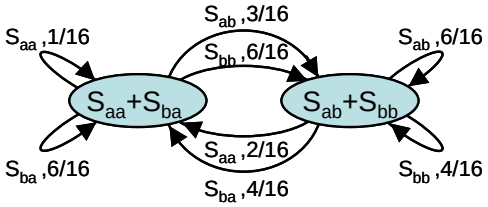
SOLUCIONES Y COMENTARIOS

1

```
#include <stdio.h>
#define MAX 100
char cedazo[MAX+1];

main()
{       int n, m;

        /* llenamos el cedazo con los números a cribar */
        for (n = 1; n <= MAX; n++) cedazo[n] = 1;

        /* la criba propiamente dicha */
        for (n = 2; n <= MAX/2; n++) if (cedazo[n]) for (m = 2; m <= MAX/n; m++) cedazo[n*m] = 0;

        /* imprimimos los números que han quedado */
        for (n = 1; n <= MAX; n++) if (cedazo[n]) printf("%d ", n);
        printf("\n");
}
```

2 *La respuesta a esta pregunta debía reflejar la situación actual, es decir la de existencia de una codificación capaz de dar satisfacción a todas las necesidades "razonables" (UNICODE y/o la versión ISO más compleja) pero sin ser asumida de modo unánime. Con esto sería razonable comentar las desventajas de las codificaciones que no se terminan de abandonar (ASCII y extensiones), así como los motivos para que esto suceda (la relación entre lenguajes que sufren dichas desventajas y su presencia en el mundo de la computación)*

3 El alfabeto de la nueva fuente será $\{S_{aa}, S_{ab}, S_{ba}, S_{bb}\}$. En principio podemos plantear un grafo con cuatro estados asociando cada uno de ellos a un símbolo (puesto que seguiremos teniendo una fuente de orden 1). No obstante, como ya se ha mencionado en clase, "de donde no hay no sale información" y nuestra fuente extendida no puede tener más de dos estados. Esto resulta evidente si pensamos que no disponemos de información diferenciada tras la emisión de S_{aa} o S_{ba} (idem. para S_{ab} o S_{bb}). Es decir, un estado estará asociado a $\{S_{aa}+S_{ba}\}$ y otro a $\{S_{ab}+S_{bb}\}$. Que la fuente extendida se encuentre en un momento dado en uno de estos estados es equivalente a que la fuente original se encuentre en el etiquetado con el último simbolo: $\{S_{aa}+S_{ba}\}\equiv\{a\}$ y $\{S_{ab}+S_{bb}\}\equiv\{b\}$. A partir de cada uno de estos estados tendremos transiciones por los cuatro símbolos cuya probabilidad puede obtenerse por la concatenación de dos transiciones en el modelo original.



$$P(S_{ab}|\{S_{aa}+S_{ba}\}) = P(a|\{a\})^* P(b|\{a\}) = 1/4 *3/4 = 3/16$$
$$P(S_{ab}|\{S_{ab}+S_{bb}\}) = P(a|\{b\})^* P(b|\{a\}) = 2/4 *3/4 = 6/16$$
Etc.

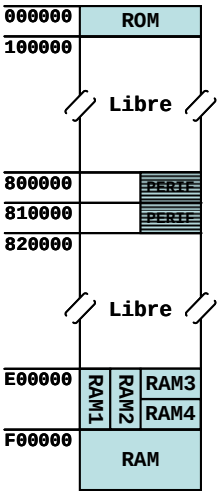
4 Aceptar interrupciones supone que una nueva entrada al procesador ha de ser tenida en cuenta por su Control. Las interrupciones se habrán de tener en cuenta al comenzar la Fase Fetch, de modo que el Paso 0 ya no es dependiente únicamente del contador sino también de esta nueva señal. En caso de que la señal de interrupción este activa, se ejecutará un Procesamiento de Excepción, compuesto por una serie de pasos que habremos de determinar.
Un planteamiento correcto de las interrupciones supone disponer de una pila que se usará para almacenar la dirección de retorno. De modo que comenzaríamos por introducir este elemento (esto se hizo ya en el ejercicio 4 del examen de septiembre de 2005).
Una vez que disponemos de la pila, como sabemos por el conocimiento de este proceso en el 68000, será preciso guardar también el estado del procesador (el SR en el 68000), que en el caso que nos ocupa no consiste en otra cosa que el registro de 1 bit Flag. Esto requiere también una alteración del hardware puesto que en la arquitectura actual dicho registro no puede llevarse al exterior.
Con todo ello hemos planteado los cambios necesarios a falta de determinar exactamente cómo se lleva a cabo paso a paso el procesamiento de excepción (y en todo caso detallar gráficamente las alteraciones de hardware) y de especificar

cómo introducir una nueva instrucción similar al RTE del 68000. Esto último se encuentra resuelto para introducir la RTS en el ejercicio 4 del examen de febrero de 2005.
En cuanto a la determinación exacta de los pasos del procesamiento de excepción, se trata de algo mecánico: con la señal de interrupción el Paso cero ya no implica el comienzo del fetch, sino que comenzará el envío del PC a la pila (con el número de pasos que se precisen), para seguir con el envío del Flag a la pila y la carga de la dirección de la rutina de atención a la interrupción en el PC.
[Nota: como se comentó en el examen, planteamos el caso más simple en que sólo hay una interrupción posible (no hay niveles, y una rutina de atención (no hay vectores; en todo caso la vectorización podria simularse por software desde dentro de la rutina de atención a la interrupción haciendo que esta "consulte" todos los posibles origenes: teclado, timers, etc.)]

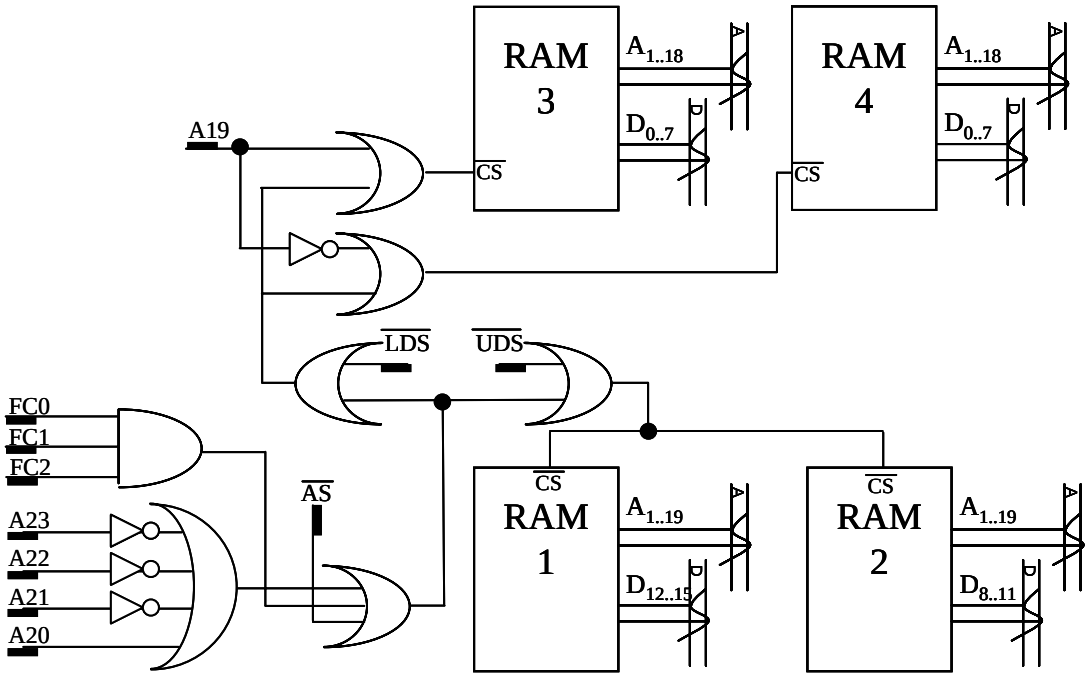
Un detalle final: la señal de interrupción no podrá considerarse directamente en la función de control. Habrá de atacar a un Flip-Flop Set-Reset en la entrada Set, de modo que cuando es atendida (en el último paso del Proceso de Excepción) se ataque a su entrada Reset y se prosiga normalmente con la ejecución de instrucciones de la rutina de atención a la interrupción.

5 Tenemos:
2 RAM de 256Kbytes, estructuradas en niblas
→ 512 Kniblas → 2^{19} elementos direccionables (niblas)
2 RAM de 256Kbytes, estructuradas en bytes
→ 2^{18} elementos direccionables (bytes)

Es decir, las pastillas en niblas son el doble de "largas" que las de bytes, y la mitad de "anchas" (tienen la misma capacidad como dice el enunciado), de modo que podemos formar un bloque de memoria perfectamente coherente situando las de bytes de modo consecutivo en la parte baja y las de niblas en paralelo en la parte alta (o viceversa). Por otro lado el lugar más adecuado será el que forme un espacio contiguo con la RAM preexistente.



Esto se consigue con el siguiente circuito:



6

```
                ;Pondremos el programa de test en 1000 hex.
                ORG       $1000

                ;Llamada al calculo de las notas con sus parametros
                LEA       Evaluacion,A0
                LEA       Notas,A1
                BSR       NOTAS_FINALES
                ;Llamada a la obtencion del histograma
                LEA       Notas,A0
                LEA       Histograma_notas,A1
                BSR       HISTOGRAMA
                ;Parada del simulador/monitor
                TRAP      #15
                DC.W      0

                ;La tabla con datos de origen a continuacion del test
Evaluacion:     dc.b      (Fin_evaluacion-Evaluacion)/10-1; número de alumnos
                dc.b      7,3,6,4,5,5,2,5,9,2             ; 10 notas del alumno 1
                dc.b      1,3,6,6,3,8,2,10,9,8            ; 10 notas del alumno 2
                dc.b      4,10,6,4,3,5102,5,9,8           ; 10 notas del alumno 3
                dc.b      10,10,10,10,10,10,10,10,10,10   ; 10 notas del alumno 4
                dc.b      7,6,4,8,4,6,8,5,3,3             ; 10 notas del alumno 5
Fin_Evaluacion:

                ;Los resultados -lista de notas e histograma- en 1100 hex.
                ORG       $1100
Histograma_notas: DS.B    11              ;10 intervalos para [0-9.9] + 1 para el 10.0
Notas:
                ;Las rutinas en 2000 hex
                ORG       $2000

;Subrutina NOTA_ALUMNO
;=============================================================================================
;Calcula la nota de un alumno como la media de sus calificaciones para las 10 preguntas del examen.
;El resultado será un byte donde los 4 bits más altos contendrán la parte entera de la nota y los 4
;más bajos un decimal.
; (p. ej. 0,0 [0000 0000 = 00hex]; 5,2 [0101 0010 = 52hex]; 10.0 [1010 0000 = A0hex])
;Entrada:       A0.L  dirección de su lista de notas por pregunta
;Salida:        D0.B  nota obtenida (según formato indicado)
;Modifica:      CCR

NOTA_ALUMNO:    MOVEM.L   A0/D1-D2,-(SP)
                CLR.L     D0              ; En D0 haremos la suma de las notas
                MOVE.W    #10-1,D1        ; D1 contador del ciclo (9..0)
C_alumno:       ADD.B     (A0)+,D0        ; Se suman todas las notas
                DBRA      D1,C_alumno
                DIVU      #10,D0          ; al dividir tenemos el resultado en Long (y del reves)
                MOVE.l    D0,D1           ; copiamos para procesar cada parte
                LSL.B     #4,D0           ; la parte entera a los 4 bits superiores del byte
                SWAP      D1              ; la parte decimal abajo del todo
                OR.B      D1,D0           ; se juntan ambas y ya esta.
                MOVEM.L   (SP)+,A0/D1-D2
                RTS

;Subrutina NOTAS_FINALES
;=============================================================================================
;Calcula y genera en memoria la lista de notas finales para todos los alumnos con el formato
;devuelto por la rutina NOTAS_ALUMNO
;Entrada:       A0.L  dirección de la lista de notas pormenorizada de todos los alumnos
;               A1.L  dirección de la lista de notas finales.
;Modifica:      CCR

NOTAS_FINALES:  MOVEM.L   A0-A1/D0-D1,-(SP)
                CLR.W     D1              ;Usaremos D1 como contador del ciclo (W)
                MOVE.B    (A0)+,D1        ;que es el número de alumnos
                MOVE.B    D1,(A1)+        ;La tabla final también lo llevará
                BRA.S     FC_notas        ;y vamos a construirla
C_notas:        BSR.S     NOTA_ALUMNO     ;calculando cada nota...
                MOVE.B    D0,(A1)+        ;... poniendola en la tabla
                LEA       10(A0),A0       ;... y pasando a las notas del siguiente alumno
FC_notas:       DBRA      D1,C_notas
                MOVEM.L   (SP)+,A0-A1/D0-D1
                RTS
```

```
;Subrutina HISTOGRAMA
;=============================================================================================
;Situa en 10 posiciones de memoria consecutivas, asociadas a los rangos [0-1),[1-2),..,[9-10),
;[10-10] de las notas obtenidas por los alumnos, la cuenta del número de ellos que han obtenido nota
;en dicho rango
;Entrada:       A0.L  dirección de la lista de notas finales de todos los alumnos.
;;              A1.L  dirección del histograma
;Modifica:      CCR

HISTOGRAMA:     MOVEM.L   A0/D0-D1,-(SP)
                MOVE.W    #11-1,D1        ;Para las 11 casillas (contador a 10)
A_cero:         CLR.B     (A2,D1.W)       ;Comenzaremmos por poner histograma a cero (por si acaso)
                DBRA      D1,A_cero
                CLR.W     D0              ;Usaremos D0 como indice (W) en función de cada nota
                CLR.W     D1              ;Usaremos D1 como contador del ciclo (W)
                MOVE.B    (A0)+,D1        ;que es el número de alumnos
                BRA.S     FC_histograma   ;y vamos a hacer el histograma
C_histograma:   MOVE.B    (A0)+,D0        ; tomamos cada nota
                LSR.B     #4,D0           ; nos quedamos con la parte entera
                ADD.B     #1,(A1,D0.W)    ; y sumamos 1 al segmento correspondiente del histograma
FC_histograma:  DBRA      D1,C_histograma
                MOVEM.L   (SP)+,A0/D0-D1
                RTS
```