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
19 de septiembre de 2008

Ejercicio práctico de Teoría de la Información — 1,75 puntos

1 Recordemos a nuestro alumno en la discoteca:

> Un alumno de SETI se encuentra en una discoteca, y no puede evitar formarse un modelo de la secuencia de tipos de canciones que "pincha" el discjockey. Cuando lleva escuchadas 34 tiene detectado que éste sigue una secuencia cíclica que es la siguiente: HCDD seguido de R o B donde R es el doble de frecuente que B. …
>
> (por si sirve de contextualización, diremos que H es un Hit de actualidad, C un Clásico, D es música Disco, R un Rock suave, y B una Balada)

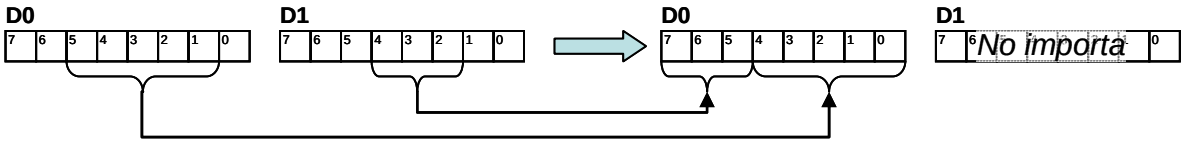
Al comentar cómo se había formado este modelo le ofrecieron sustituir al discjockey en sus vacaciones de verano, con la condición de que siguiera su mismo comportamiento. Se le planteó qué hacer en el momento de decidir si pinchaba R o B: o bien ponía sistemáticamente dos veces R y una B (xRxRxBxRxRxB con x=HCDD) o bien utilizaba un dado para determinar qué poner. Indique qué consecuencias sobre la entropía tiene cada una de las dos opciones y cómo llevaría usted a cabo la segunda (qué haría con el dado). ¿Podría utilizar una moneda en vez de un dado?

Ejercicio práctico de codificación — 1,75 puntos

2 Suponiendo una representación de reales a 10 bits (1,4,5), determine el código de π (≈3,141592653589…)

Programación del µP68000: operación simple con pocas instrucciones — 1 punto

3 Escriba la secuencia de instrucciones en ensamblador del 68000 para llevar a cabo la transformación planteada en la figura.



Programación del µP68000: codificación de rutina a partir de su algoritmo — 1,5 puntos

4 Escriba el código 68000 correspondiente a la siguiente rutina (elija cómo pasar los parámetros y el retorno):

```
Rutina máximo (array_de_bytes ListaDeNumeros, word long_lista) devuelve byte
Principio
  vTemp=ListaDeNumeros[1];
  Para i=2 hasta long_lista hacer
     Si ListaDeNumeros[i] > vTemp entonces vTemp:= ListaNumeros[i];
  Devolver vTemp
Fin
```

Programación del 68000: problema completo 4 puntos

5 Dadas dos series de puntos en un espacio bidimensional (2 words como coordenadas X e Y) almacenados en memoria de modo contiguo, con una word inicial indicando el número de puntos, escribir las siguientes rutinas:

Subrutina CENTROIDE

Esta subrutina deberá calcular un punto que tenga por valor X la media de todas las X de todos los puntos (aproximada a un entero), y de igual modo con la coordenada Y. La subrutina ***podría*** tener en concreto la siguiente descripción:

```
; Subrutina:   CENTROIDE
; Descripción: Busca el punto de coordenadas medias (valores enteros)
; Entrada:     A0.L  dirección de la lista de coordenadas
; Salida:      D0.L  Parte alta coordenada X, parte baja coordenada Y
; Modifica:    (rellenar debidamente)
```

Subrutina DISTANCIA1

Esta subrutina deberá calcular la distancia perpendicular entre dos puntos (la suma de sus distancias en cada eje |X1-X2|+|Y1-Y2|).

```
; Subrutina:   DISTANCIA1
; Descripción: calcula la distancia perpendicular
; Entrada:     D0.L, D1.L los dos puntos (Parte alta coordenada X, parte baja coordenada Y)
; Salida:      D2.W  distancia
; Modifica:    (rellenar debidamente)
```

Programa principal

El programa principal deberá definir las dos listas de puntos, calcular los dos centroides y hallar la distancia perpendicular entre ellos para dejarla en una posición de memoria. El programa principal ***podría*** tener en concreto las siguientes partes:

```
org     $1000
…
Llamada a CENTROIDE para una de las listas de puntos
…
Llamada a CENTROIDE para la otra lista de puntos
…
Llamada a DISTANCIA1
Guardar el resultado
Final.

org     $2000
; Declaración de todos los datos
```

8
23,87
7676,76
2656,6522
2,76
876,236
28123,32
343,6
23,7787

Ejemplo de lista de puntos

Examen de S.E.T.I.

1er curso de Ingeniería Electrónica
19 de septiembre de 2008

SOLUCIONES Y COMENTARIOS

1 En el caso de "pinchar" sistemáticamente dos R y una B cada vez que le toca el turno de esta elección, implica que la secuencia está totalmente determinada (es un ciclo de período 15 símbolos). En consecuencia, sabido esto, nuestro modelo tiene entropía nula. En ningún instante hay ninguna duda sobre cual será la próxima emisión.

En el caso de utilizar un dado, la naturaleza intrínsecamente aleatoria del proceso de arrojar el dado y decidir en función de su resultado, hace que nuestro modelo siga presentando la misma incertidumbre que se presentaba originalmente (ejercicio del examen anterior) arrojando un valor de entropía idéntico.

Lo anterior es cierto, naturalmente, si se utiliza bien el dado. Como se trata de obtener una distribución de probabilidad de P(B)=1/3, P(R)=2/3, bastará con asignar 2 y 4 de los 6 posibles valores del dado respectivamente a cada símbolo (p.ej. si el dado muestra 1 o 2 se emitirá B, si muestra 3,4,5, o 6 se emitirá R).

Con una moneda no podremos generar esta distribución de probabilidad en una sola tirada, pero podemos utilizar una doble tirada y actuar como sigue (p.ej.) cara+cara=B, cara+cruz=R, cruz+cara=R, cruz+cruz=volver a tirar. De este modo, desechando la combinación cruz-cruz, atendemos a tres posibles configuraciones equiprobables y asociamos una con B y dos con R

2 El bit de signo será 0 (positivo)

El factor exponencial ha de ser 2 (2^1) puesto que es un número entre 2 y 4. Como disponemos de 4 bits para el exponente, al bías será 7 (0111_2) y el exponente 1 requerido se obtiene como 1=E-bias=E-7, es decir E=8 (1000_2).

En cuanto a la mantisa, dado que $\pi = 2^1 (1+M)$, entonces $M=\pi/2-1=0{,}5707963267945$

Disponemos de 5 bits que representan los valores 1/2,1/4,1/8,1/16,1/32 y con ellos hemos de determinar la mejor aproximación a M. Poniendo esto en 32–avos (16/32,8/32,4/32,2/32,1/32) como M= 18,265482457424 / 32, la mejor aproximación será la correspondiente a la configuración binaria del 18 (10010_2)

$\pi =$ | 0 | 1000 | 10010 | $= 3{,}125$

3

```
LSR.B #1,D0
LSL.B #3,D1
AND.B #%00011111,D0
AND.B #%11100000,D1
OR.B  D1,D0
```

4 Si no nos atenemos a la literalidad del algoritmo mostrado en el enunciado, puede realizarse la misma función más eficientemente, dado que el ensamblador nos permite ajustarnos a las capacidades de programación de nuestro procesador concreto. Los alumnos han tenido tendencia a no implementar una traducción directa por este motivo, lo que ha sido dado por bueno. La rutina mostrada a continuación es aproximadamente una versión literal del algoritmo del enunciado.

```
; Subrutina:      MAXIMO
; Descripción:    Obtiene el valor máximo de una lista de bytes
; Entrada: A0.L lista de bytes
;                 D0.W número de elementos en la lista
; Salida:  D1.B El máximo
; Modifica: CCR
MAXIMO:  MOVE.W      D2,-(SP)        ;protege D2 que usa como índice del PARA
         MOVE.B      (A0),D1         ;vTemp=ListaDeNumeros[1];
         MOVE.W      #2,D2           ;Para i=2
FOR:     CMP.B       D0,D2           ; hasta long_lista
         BHI.S       FIN             ;(sale si el contador se pasa)
         CMP.B       -1(A0,D2),D1    ;Si ListaDeNumeros[i] > vTemp
         BHI.S       MAYOR           ;
         MOVE.B      -1(A0,D2),D1    ;vTemp:= ListaNumeros[i]
MAYOR:   ADD.B       #1,D2           ;(índice del Para i=i+1)
         BRA.S                       FOR
FIN:     MOVE.W      (SP)+,D2        ;recupera contenido previo de D2
         RTS
```

5

```
            ORG       $1000
            LEA       LISTA1,A0
            BSR       CENTROIDE
            MOVE.L    D0,D1
            LEA       LISTA2,A0
            BSR       CENTROIDE
            BSR       DISTANCIA1
            MOVE.W    D2,RESULTADO
            SERMON    FPROG

LISTA1:     DC.W      ((LISTA2-LISTA1)/2-1)/2   ,8,3  ,4,56  ,5,23  ,34,433  ,23,23
LISTA2:     DC.W      ((RESULTADO-LISTA2)/2-1)/2 ,12,87 ,35,76 ,34,8 ,456,5 ,22,98
RESULTADO:  DS.W      1

; Subrutina:       CENTROIDE
; Descripción:     Busca el punto de coordenadas medias (valores enteros)
; Entrada:  A0.L  dirección de la lista de coordenadas
; Salida:   D0.L  Parte alta coordenada X, parte baja coordenada Y
; Modifica: CCR
; Observaciones: Ambas versiones obvian el rebose en la suma. La de la derecha admite
;                sumas mayores antes de producir errores
```

```
CENTROIDE:  MOVEM.L   A0/D1-D3,-(SP)
            MOVE.W    (A0)+,D1
            MOVE.W    D1,D2
            CLR.W     D0
            CLR.W     D3
            BRA.S     ITERA
CICLO:      ADD.W     (A0)+,D0
            ADD.W     (A0)+,D3
ITERA:      DBRA.S    D1,CICLO
            DIVU      D2,D0
            DIVU      D2,D3
            SWAP      D0
            MOVE.W    D3,D0
            MOVEM.L   (SP)+,A0/D1-D3
            RTS
```

```
CENTROIDE:  MOVEM.L   A0/D1-D4,-(SP)
            MOVE.W    (A0)+,D2
            MOVE.W    D2,D3
            CLR.L     D0
            CLR.L     D1
            CLR.L     D4
            BRA.S     ITERA
CICLO:      MOVE.W    (A0)+,D4
            ADD.L     D4,D0
            MOVE.W    (A0)+,D4
            ADD.L     D4,D1
ITERA:      DBRA.S    D2,CICLO
            DIVU      D3,D0
            DIVU      D3,D1
            SWAP      D0
            MOVE.W    D1,D0
            MOVEM.L   (SP)+,A0/D1-D4
            RTS
```

```
; Subrutina:       DISTANCIA1
; Descripción:     calcula la distancia perpendicular
; Entrada:  D0.L, D1.L los dos puntos (Parte alta coordenada X, parte baja coordenada Y)
; Salida:   D2.W  distancia
; Modifica: CCR
; Observaciones: Se entiende que los números son enteros(signed).Se obvian los reboses.

DISTANCIA1: MOVEM.L   D0/D1,-(SP)
            SUB.W     D0,D1
            BHI       POSITIVO1
            NEG.W     D1
POSITIVO1:  MOVE.W    D1,D2
            SWAP      D0
            SWAP      D1
            SUB.W     D0,D1
            BHI       POSITIVO2
            NEG.W     D1
POSITIVO2:  ADD.W     D1,D2
            MOVEM.L   (SP)+,D0/D1
            RTS
```