1)

```
.include "macros.s"
.include "crt0.s"

.data
        w: .word 0xF0E0

.text
main:
   $movei  R1, w
   $movei  R0, 16
   $movei  R2, 0
   add R3, R1, R0      ;final
   $movei  R5, 1
   $movei  R6, 0       ;numero a imprimir

bucle:
   LD  R4, R2(R1)
   $CMPEQ  R4, R4, R5
   addi    R6, R6, 1          ;hasta 16
   BZ R4, suma1
   $CMPEQ R4, R6, R0
   BZ R4, bucle
   BNZ R4, salirBucle
suma1:
   addi    R1, R1, 1
   $CMPEQ R4, R6, R0
   BZ R4, bucle

salirBucle:
   $movei R1, 0x8000          ;Pongo el bit de puesta en marcha
   OUT Rfil_pant, R6          ;le indico la fila donde se va a mostrar el valor en la pantalla
   OUT Rcol_pant, R6          ;le indico la columna en la cual debe mostrar el valor por pantalla
   OUT Rdat_pant, R6          ;Cargo en el registro de datos de la pantalla el dato.
   ;Rdat_pant ya contiene el carácter a mostrar, a continuación le doy la orden para que lo muestre
por pantalla
   OUT Rcon_pant, R1          ;El valor aparecera por pantalla
```

2)

```
.include "macros.s"
.include "crt0.s"
.data
   w: .word 0xF0E0
   ticks: .word 0 ; variable global para acumulador de ticks
   final: .byte 0 ; variable global indica que se alcanzó los 10 s
      .balign 2
   tecla: .byte 0 ; variable global que guarda la tecla pulsada
      .balign 2
.text
main:
```

```
   $movei  R1, w
   $movei  R0, 16      ;fin
   $movei  R2, 0
   $movei R3,  0       ;numero de 0
   $movei  R5, 1       :comp 1
   $movei  R6, 0       ;numero a imprimir

bucle:
   LD  R4, R2(R1)
   $CMPEQ  R4, R4, R5
   addi    R6, R6, 1
   BZ R4, suma1
   BNZ R4, suma0
   $CMPEQ R4, R6, R0
   BZ R4, bucle
   BNZ R4, salirBucle
suma1:
   addi    R1, R1, 1
   $CMPEQ R4, R6, R0
   BZ R4, bucle
suma0:
   addi R1, R1, 1
   $CMPEQ R4, R3, R0
   BZ R4, bucle

salirBucle:
   ;R6 contiene el valor a mostar por pantalla
   $MOVEI R0, interrupts_vector          ;Cargo la dirección del vector de interrupciones, para que
a partir de este vector de interrupciones pueda colocar la dirección en la cual empieza la rutina de
interrupción, que es la rutina de clock.
   $MOVEI R1, clock                      ;Cargo la dirección donde se encuentra la rutina de atención a
la interrupción al reloj, que he llamado clock.
   ST 0(R0), R1                          ;Cargo esta dirección a partir de la posición 0 del vector de
interrupciones.
   MOVI R2, 1
   OUT Rcon_rel, R2                      ;Habilito las interrupciones desde reloj, colocando en 1 el bit
menos siginificativo.
   EI                                    ;Habilito que el procesador acpte interrupciones. (EI coloca el registro
S7 al bit 1 en 1.)
   $MOVEI R0, final                      ;Cargo la dirección de la variable final en el registro R0

bucle2:                                  ;En este bucle el main espera a las interrupciones que va a generar el
reloj.
   LDB R2, 0(R0)                         ;Cargo el contenido que se encuentra en final en R2
   BZ R2, bucle2                         ;Vuelve a bucle hasta que final no sea 1

   ;Despues de 10 segundos muestro el valor en pantalla
   $movei R1, 0x8000                     ;Pongo el bit de puesta en marcha
   OUT Rfil_pant, R6                     ;le indico la fila donde se va a mostrar el valor en la pantalla
   OUT Rcol_pant, R6                     ;le indico la columna en la cual debe mostrar el valor por
pantalla
   OUT Rdat_pant, R6                     ;Cargo en el registro de datos de la pantalla el dato.
```

```
   ;Rdat_pant ya contiene el carácter a mostrar, a continuación le doy la orden para que lo muestre por pantalla
   OUT Rcon_pant, R1                ;El valor aparecera por pantalla

   movi R0, 0
   OUT Rcon_tec, R0                 ;Le indico al dispositivo que la sincronización es por interrupción.
   $MOVEI R1, interrupts_vector     ;Cargo la dirección del vector de interrupciones, para que a partir de este vector de interrupciones pueda colocar la dirección en la cual empieza la rutina de interrupción, que es la rutina de teclat.
   $MOVEI R2, teclat
   ST 0(R1), R2                     ;Cargo la dirección donde se encuentra la rutina de atención a la interrupción al teclado, que he llamado teclat.
   MOVI R4, 1
   OUT Rcon_tec, R4                 ;Habilito las interrupciones desde el teclado, colocando en 1 el bit menos siginificativo.
   EI                               ;Habilito que el procesador acpte interrupciones.
   $MOVEI R3, 0
   $MOVEI R5, final                 ;Cargo la dirección de la variable final en el registro R3
   STB 0(R5), R3                    ;Vuelvo a poner a 0 final.

   HALT

clock:
   $MOVEI R0, ticks                 ;Cargo la dirección de ticks en R0.
   LD R1, 0(R0)                     ;Cargo el contenido de ticks en R1
   ADDI R1, R1, 1
   ST 0(R0), R1                     ;ticks++
   MOVI R2, 10*10
   $CMPGE R2, R1, R2                ;ticks >= 10 seg
   BZ R2, fiClock                   ;Si no es igual a 10seg salta a fiClock

   ;cuando ha llegado a 10seg continua
   MOVI R1, 1                       ;R1=1
   $MOVEI R0, final                 ;Cargo la dirección de final en R0
   STB 0(R0), R1                    ;final=true
fiClock:
   JMP R6                           ;Vuelve a la Rutina de servicios generales, que restaura los registros que guardo en la pila y vuelve a la rutina principal del programa y continua ejecutando el programa donde lo dejo cuando fue interrupido por el reloj.

teclat:
   in R1, Rdat_tec                  ;R1=codigo de rastreo
   $MOVEI R3, tteclat               ;Cargo la dirección tteclat en R3
   add R1, R3, R1                   ;R1=&tteclat[codigoderastreo]
   $movei R3, tecla
   STB 0(R3), R1                    ;tecla=teclapresionada

   ;cuando ha acabado
   MOVI R1, 1                       ;R1=1
   $MOVEI R0, final                 ;Cargo la dirección de final en R0
   STB 0(R0), R1                    ;final=true
```

fiTeclat:
JMP R6 ;Vuelve a la Rutina de servicios generales, que restaura los registros que guardo en la pila y vuelve a la rutina principal del programa y continua ejecutando el programa donde lo dejo cuando fue interrupido por el teclado.

3) Donat un processador d'arquitectura de 32 bits en un sistema amb memòria principal de 16 MB, amb una memòria cache directa de 16 KB, i de 4 words per bloc.

a) Donada una direcció a memòria originada des del processador ¿quants bits seran de byte offset, word offset, index i tag?

Tamañobloque/tamañocahce = 64 Bloques.
Log2(64) = 8 bits de indexados

6 bits de index