Estructura de Computadores

Grado de Ingeniería Informática

ETSINF

Tema 8: Ejercicios Interrupciones

Curso 2019-2020

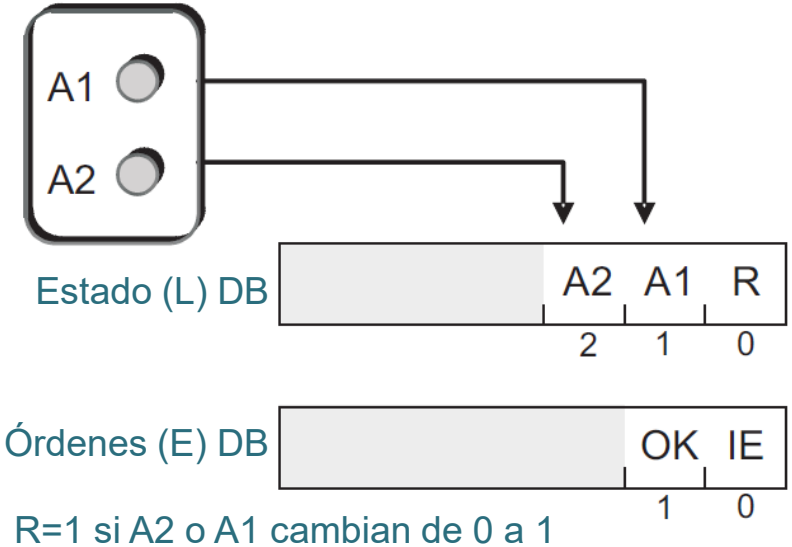
DISCA

Ejercicios de Interrupciones

- Ejercicio 6

Periférico A

DB:0xFFFFCC00 – Int3*

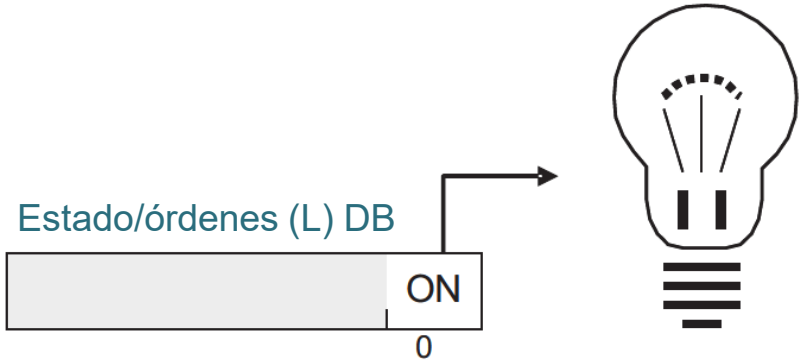


R=1 si A2 o A1 cambian de 0 a 1

Cancelación: Poner 1 en OK → R=0

Periférico B

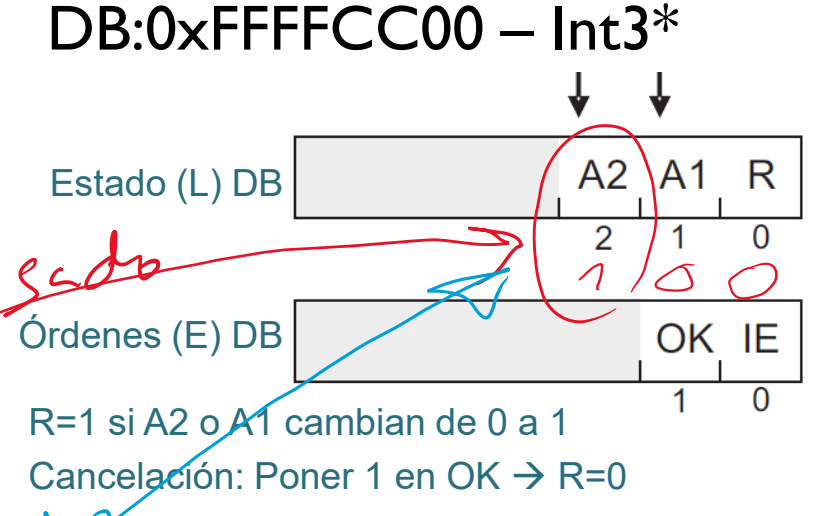
DB:0xFFFFDD00 – E/S Directa

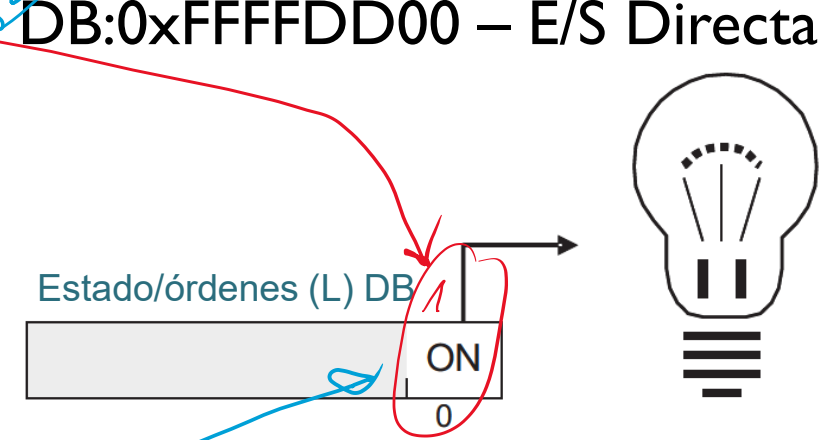


Ejercicio 6

1. Observa el programa siguiente y describe el comportamiento del sistema:

```
        la $t0,0xFFFFCC00
        la $t2,0xFFFFDD00
bucle1:lb $t1,0($t0)
        andi $t1,$t1,4
        beq $t1,$zero,bucle1
        li $t1,1
        sb $t1,0($t2)
buble2:lb $t1,0($t0)
        andi $t1,$t1,4
        bne $t1,$zero,bucle2
        sb $zero,0($t2)
        j bucle1
```



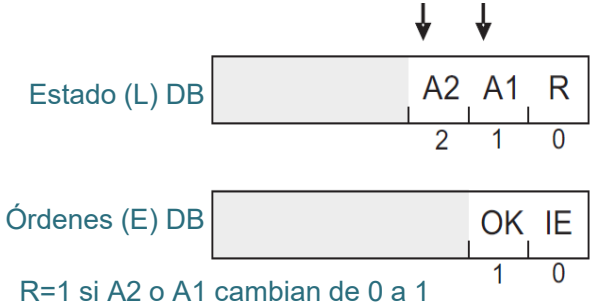


2. Escribe un programa que, por consulta de estado del bit *R*, controle los periféricos para que el sistema tenga el comportamiento siguiente:

Al pulsar el botón *A1*, la lámpara se ha de encender (esté como esté previamente) y ha de mantenerse encendida hasta que, al pulsar el botón *A2*, la lámpara se apague. Pulsar los dos botones al mismo tiempo no ha de tener ningún efecto.

```
          la $t0,0xFFFFCC00
          la $t2,0xFFFFDD00
bucle:    lb $t1,0($t0)
          andi $t3,$t1,1 #bit R
          beq $t3,$zero,bucle1
          andi $t1,$t1,6 #110 A2 y A1
          li $t3,2 #010 A1 pulsado
          beq $t3,$t1,enciende
          li $t3,4 #100 A2 pulsado
          beq $t3,$t1,apaga
cancela:  li $t1,2 #10 OK y no IE
          sb $t1,0($t0)
          b bucle
enciende:li $t1,1
          sb $t1,0($t2)
          b cancela
apaga:    sb $zero,0($t2)
          b cancela
```

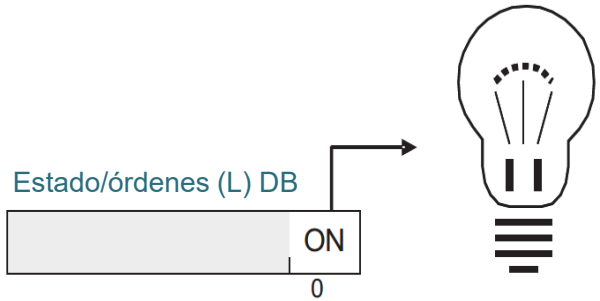
DB:0xFFFFCC00 – Int3*



R=1 si A2 o A1 cambian de 0 a 1

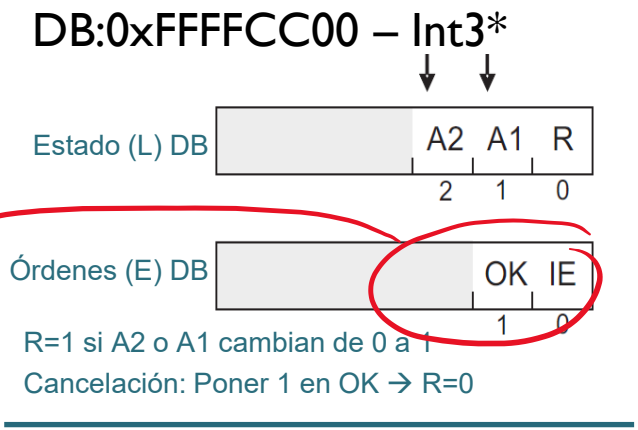
Cancelación: Poner 1 en OK → R=0

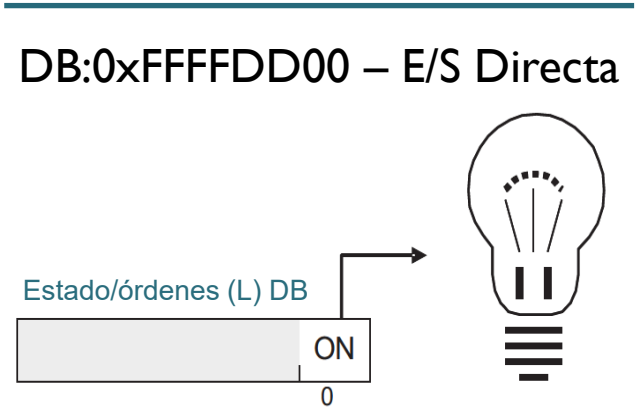
DB:0xFFFFDD00 – E/S Directa



3. Considera ahora que las interrupciones están habilitadas en la interfaz de *A*. Escribe el tratamiento que, dentro del manejador de excepciones, ha de aplicarse a la interrupción *Int3**. El comportamiento ha de ser el descrito en el apartado 2. Puedes utilizar los registros *$t0* a *$t3*.

```
int3:   la $t0,0xFFFFCC00
        la $t2,0xFFFFDD00
bucle:  lb $t1,0($t0)
        andi $t3,$t1,1 #bit R
        beq $t3,$zero,bucle1
        andi $t1,$t1,6 #110 A2 y A1
        li $t3,2 #010 A1 pulsado
        beq $t3,$t1,enciende
        li $t3,4 #100 A2 pulsado
        beq $t3,$t1,apaga
cancela: li $t1,2 #10 OK y no IE
        sb $t1,0($t0)
        b bucle
enciende:li $t1,1
        sb $t1,0($t2)
        b cancela
apaga:  sb $zero,0($t2)
        b cancela
```



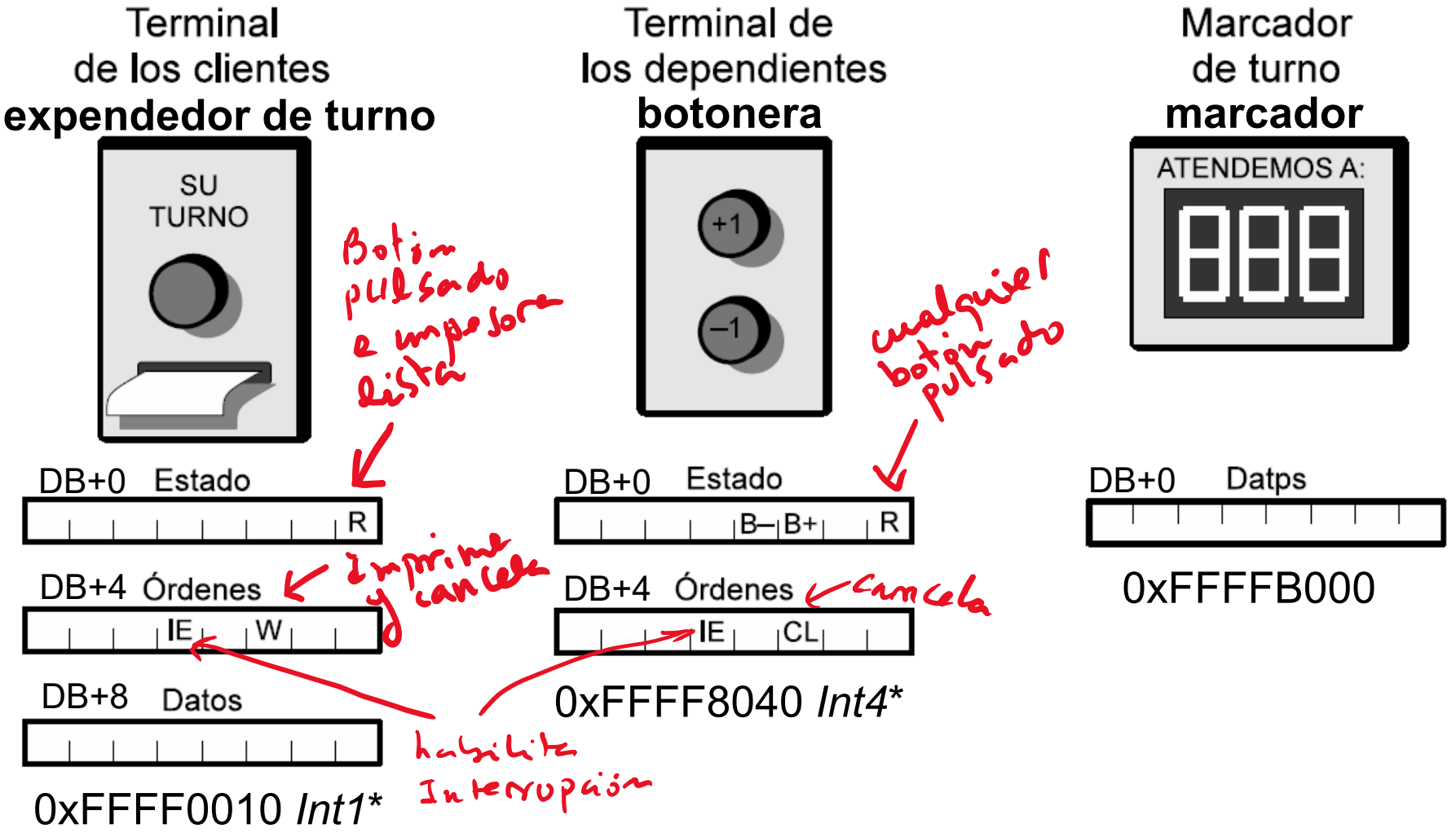


4. Se ha incluido en el manejador las funciones 3297 y 3298 que pueden utilizar los programas de usuario por medio de llamada al sistema. La función 3297 permite a los programas encender y apagar la lámpara *B*. La función 3298 permite a los programas consultar el estado de la lámpara. Su especificación es la siguiente:

Función	Índice ($v0)	Parámetros entrada	Parámetros de salida
Encender_lamp	3297	$a0 = 1:encendido 0:apagado	—
Estado_lamp	3298	—	$v0 = 1:encendido 0:apagado

```
# llama a la función que consulta el estado
  li $v0,3298
  syscall
# ahora tiene en $v0 el estado de la luz
# calcula el parámetro para invertir el estado
  xori $a0,$v0,1
# llama a la función que fija nuevo estado
  li $v0,3297
  syscall
```

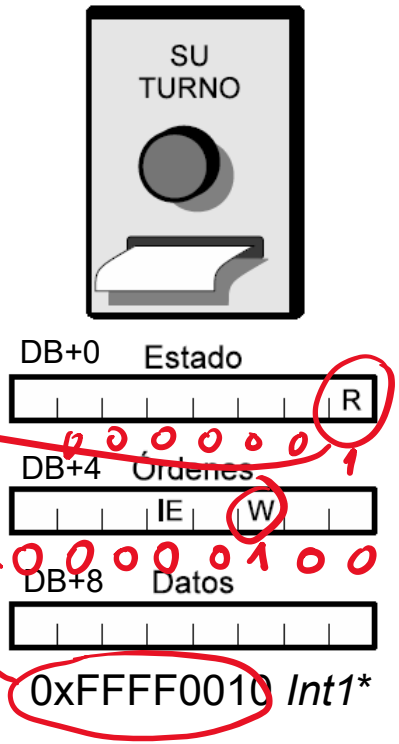
Ejercicio 10 – Punto de venta



1. Escriba un programa de prueba del expendedor que se limite a esperar por consulta de estado que se pulse el botón de pedir turno e imprima un tiquet con el número “255”.

Este programa, útil para el diagnóstico del sistema, se ejecutará en modo supervisor y podrá acceder sin restricciones a la interfaz de los periféricos.

```
          la $t0,0xFFFF0010
consulta: lb $t1,0($t0)
          andi $t1,$t1,1
          beqz $t1,consulta
          li $t1,255
          sb $t1,8($t0)
          li $t1,0x04
          sb $t1,4($t0) # bit W:escritura y cancelación
          j consulta
```



2. Observe este tratamiento de la interrupción *Int4** dentro del manejador. Explique qué hace.

```
int4: la $t0,0xFFFF8040
      li $t1,0x14
      sb $t1,4($t0)
      li $t2,0xD
      lb $t1,0($t0)
      bne $t1,$t2,L0
      sb $zero,clientes
      sb $zero,servicio
      la $t0,0xFFFFB000
      sb $zero,0($t0)
      j retexc
  L0: # aquí ampliará más adelante
      j retexc
```

Nota: el programa del enunciado tiene dos errores, el primero es que utiliza un valor 0xD que asume que el bit R está activo, cuando las instrucciones previas lo han cancelado y el segundo es que no hace un andi para poner a cero los bits indeterminados del registro de estado, asume que estos son cero. He corregido el programa, cambiando el orden entre la lectura del registro de estado y la cancelación, para que si contenga el bit R activo y por tanto sea válido el valor 0xD y he añadido el andi que faltaba:

```
int4: la $t0,0xFFFF8040
      lb $t1,0($t0)
      li $t2,0x14
      sb $t2,4($t0)
      li $t2,0xD
      andi $t1,$t1,$t2
      bne $t1,$t2,L0
      sb $zero,clientes
      sb $zero,servicio
      la $t0,0xFFFFB000
      sb $zero,0($t0)
      j retexc
  L0: # aquí ampliará más adelante
      j retexc
```

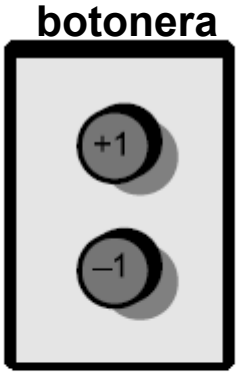
2. Observe este tratamiento de la interrupción *Int4** dentro del manejador. Explique qué hace.

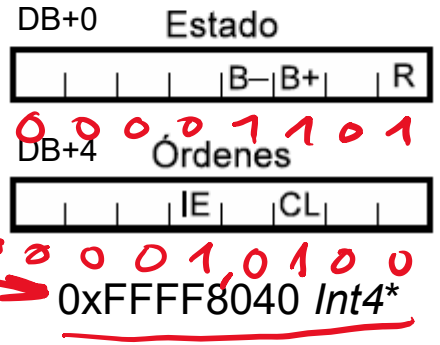
```
int4: la $t0,0xFFFF8040
      lb $t1,0($t0)
      li $t2,0x14
      sb $t2,4($t0)
      li $t2,0xD
      andi $t1,$t1,$t2
      bne $t1,$t2,L0
      sb $zero,clientes
      sb $zero,servicio
      la $t0,0xFFFFB000
      sb $zero,0($t0)
      j retexc
 L0: # aquí ampliará más adelante
      j retexc
```

no

¿Los dos botones?

cancelo + Int habilitada





Cuando se pulsan los 2 botones a la vez pone a cero las variables clientes y servicio y pone un 0 en el display

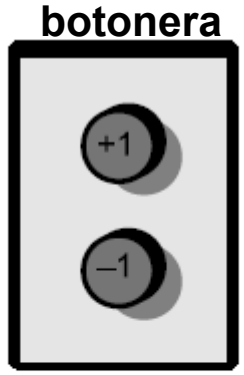
3. Complete el tratamiento del apartado 2 a partir de la etiqueta L0 para que, al pulsar uno de los dos botones del terminal, se incremente o decremente la variable servicio y su valor resultante se muestre en el marcador. Ignore la posibilidad de desbordamiento, porque nunca vendrán más de 50 clientes en un día cualquiera, y por la noche se apaga el sistema.

lo que tengo en $t1 es 0000B-B+01

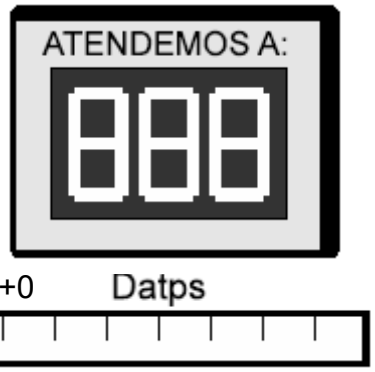
Luego si estuviera B- solo sería 00001001

0x09

```
L0: li $t2,0x9 # B-?
    beq $t1,$t2,L1
    # tratamiento de B+
    lb $t1,servicio
    addi $t1,$t1,1
    sb $t1,servicio #servicio++
    la $t0,0xffffb000
    sb $t1,0$(t0) #escribo en display
    j retexc #termino
L1: # tratamiento de B-
    lb $t1,servicio
    addi $t1,$t1,-1
    sb $t1,servicio #servicio--
    la $t0,0xffffb000
    sb $t1,0$(t0) #escribo en display
    j retexc #termino
```



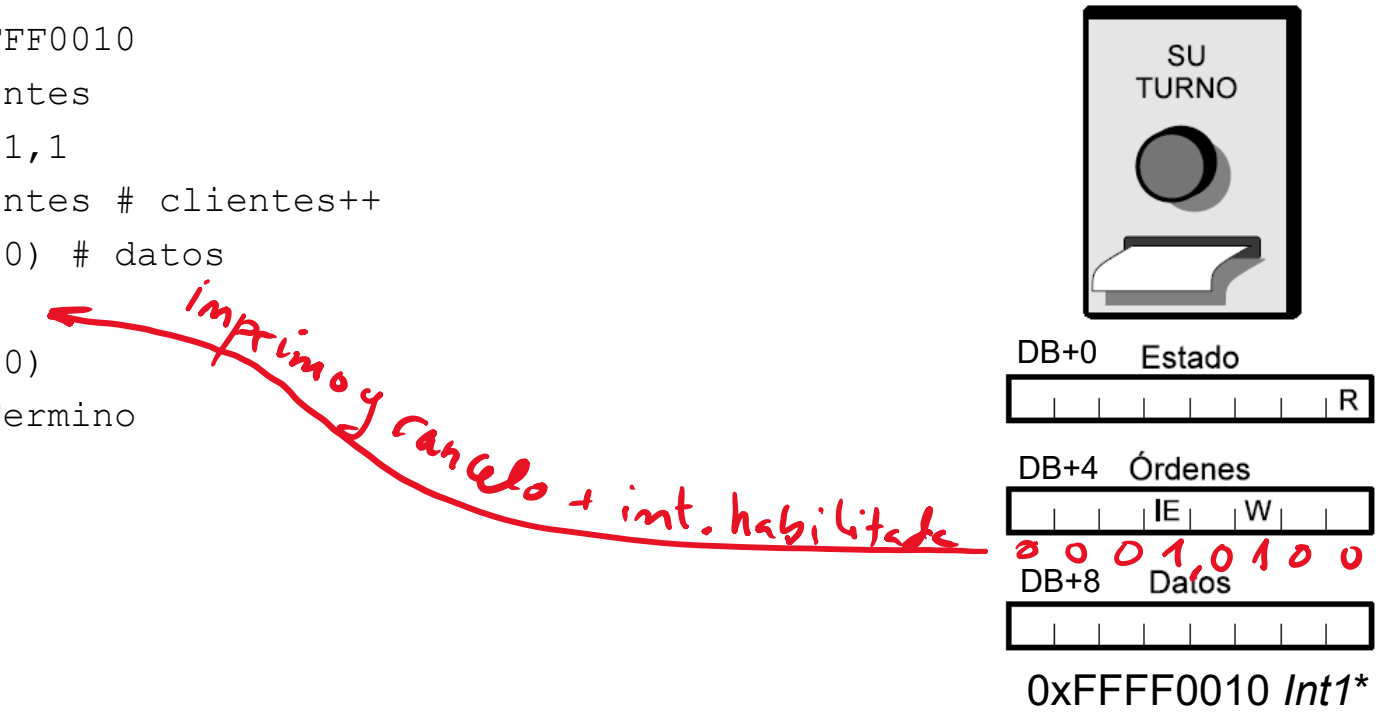
0xFFFF8040 *Int4**



0xFFFFB000

4. Escriba el tratamiento correspondiente a la interrupción Int1. Cuando un cliente pulse el botón del expendedor, hay que incrementar la variable clientes e imprimir en el ticket su valor.

```
int1: la $t0,0xFFFF0010
      lb $t1,clientes
      addi $t1,$t1,1
      sb $t1,clientes # clientes++
      sb $t1,8($t0) # datos
      li $t1,0x14
      sb $t1,4($t0)
      j retxec #Termino
```



5. Escriba el tratamiento de la función de sistema get_clients que tiene este perfil:

Servicio	Índice	Parámetros de salida
`get_clients`	$v0 = 666	$v0 = Número de turnos dados

Suponga que el manejador salta a la etiqueta fun666 cuando la causa de excepción es la instrucción syscall y $v0 = 666. Sólo ha de escribir el código apropiado a partir de esta etiqueta.

```
fun666: lb $v0,clientes
        j retxec
```