Estructura de los computadores

Memoria de las practicas 10 y 11

Práctica 10:

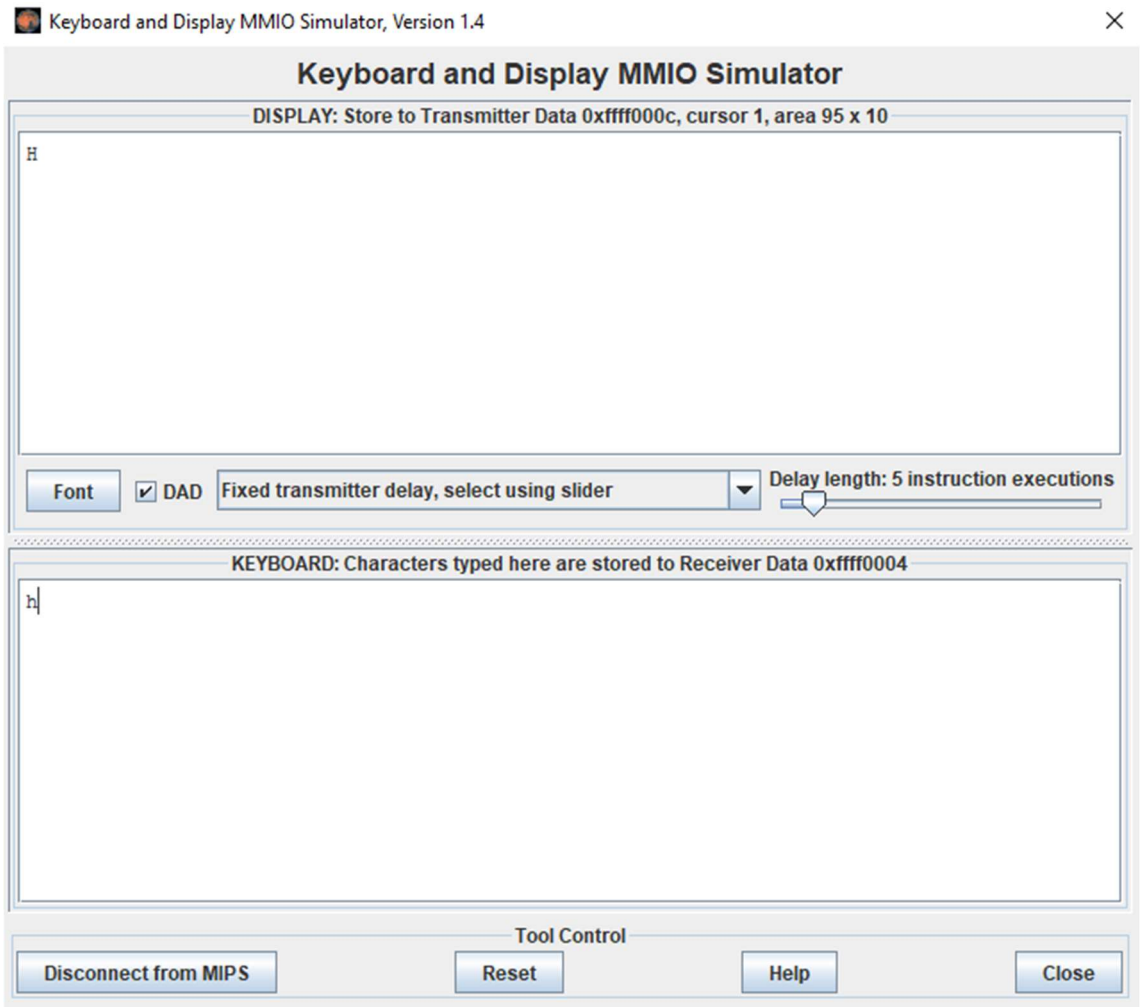
Cuestión 4:

Transforma el programa echo de la cuestión 3 en el programa caps que muestra por la consola la mayúscula del carácter introducido por el teclado. Supón que todos los caracteres introducidos están en minúscula.

```
.text
main:
        li $t3,'\n'             #caracter salto de linea

        jal getc                #lee caracter por teclado

        move $s0, $v0           #mueve el caracter guardado a s0 para comparar condicion de echo

        beq $s0,$t3,end         #comprueba si el caracter es salto de linea para terminar el programa

        move $a0, $v0           #mueve el caracter guardado a a0 para escribirlo

        subi $a0,$a0,32         #trasnformo caracter a mayuscula

        jal putc                #escribe el caracter por teclado

        j main                  #vuelve al principio del programa

getc:

        lui $t0, 0xffff         #Direccion de registro de control por teclado
        li $t1, 0               #Iniciar contador de espera

        b_espera_g:
        lw $t2, ($t0)           #Lee registro control del teclado

        andi $t2, $t2, 1        #Extrae el bit de ready
        addiu $t1, $t1, 1       #Incrementa el contador
                                #(cuenta las iteraciones)

        beqz $t2, b_espera_g    #Si !=0 entonces se ha detectado caracter

        lw $v0, 4($t0)          #Lee registro de datos del teclado
                                #caracter guardado en $v0

        jr $ra                  #vuelve al programa principal
 putc:
        lui $t0, 0xffff         #Direccion de registro de control por teclado

        b_espera_p:
```

```
        lw $t1, 8($t0)          #Lee registro control del teclado
        andi $t1, $t1, 0x0001   #Extrae el bit de ready
        beqz $t1, b_espera_p    #Si =0 entonces sigue esperando
        sw $a0, 12($t0)         #Escribe en la consola
        jr $ra                  #vuelve al programa principal
end:
        li $v0,10
        syscall                 #fin programa
```

<u>Ejemplo de ejecucion:</u>



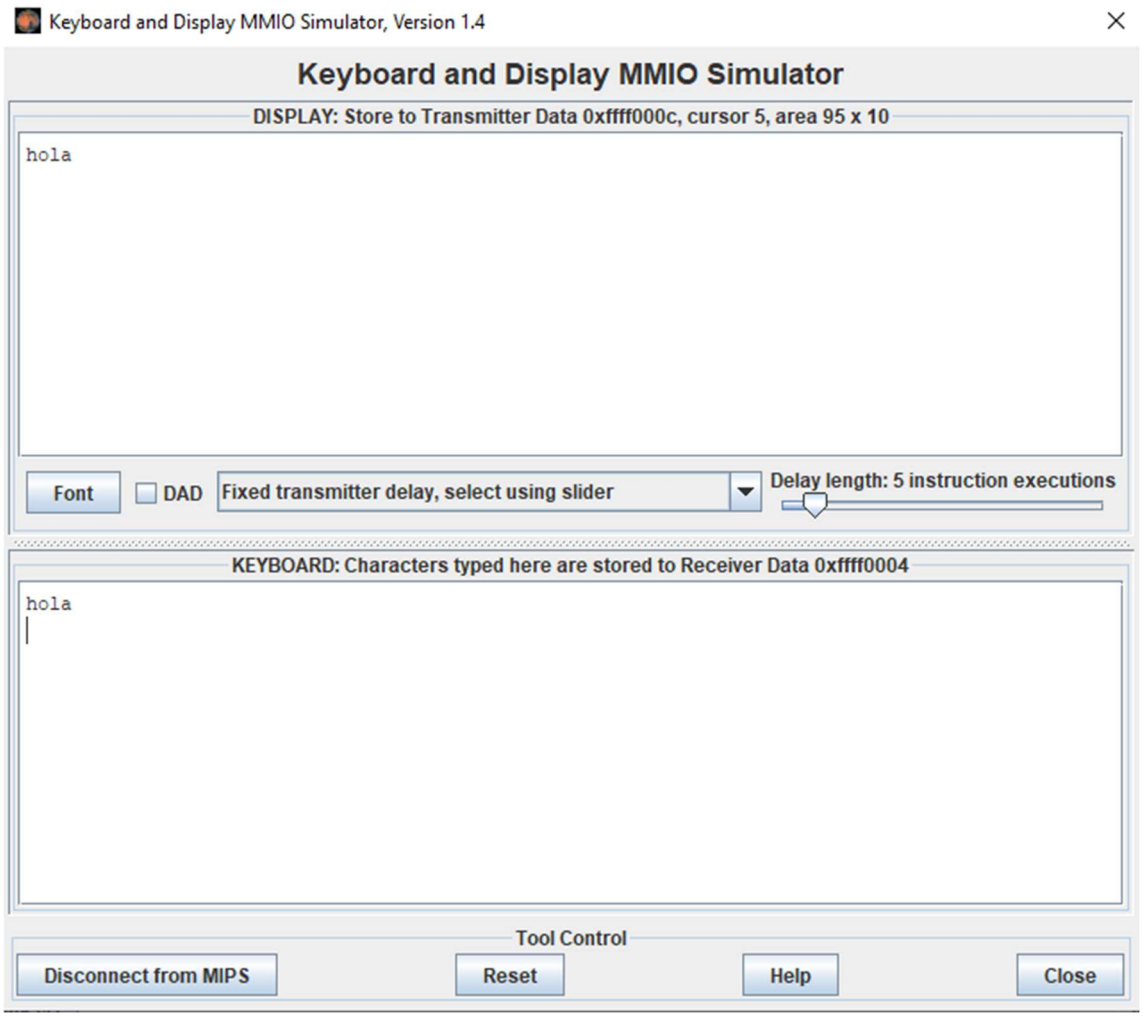
Cuestión 5:

Compléta el código escribiendo la función read_string. Esta función tiene que leer del teclado la cadena de caracteres que introduzca el usuario y tiene que almacenarla en un buffer denominado cadena. La cadena finaliza cuando el usuario teclee un salto de línea. Posteriormente el programa muestra la cadena en la consola. Al escribir la función read_string no olvidéis meter en el buffer el carácter de salto de línea.

```
.data

cadena: .space 32
        .eqv ControlTeclado 0
        .eqv BufferTeclado 4
        .eqv ControlDisplay 8
        .eqv BufferDisplay 12

.text

        la $a0,cadena
        jal read_string
        la $a0,cadena
        jal print_string

        li $v0,10
        syscall

print_string:
        la $t0,0xFFFF0000
sync:
        lw $t1, ControlDisplay($t0)
        andi $t1,$t1,1
        beqz $t1,sync

        lbu $t1,0($a0)
        beqz $t1,final
        sw $t1, BufferDisplay($t0)
        addi $a0,$a0,1
        j sync

final:
        jr $ra

read_string:
        la $t0, 0xffff0000              #direccion de registro de control por teclado
        li $t3,0

sync_read:
        lw $t1,ControlTeclado($t0)      #lee registro
        andi $t1,$t1,1                  #extrae bit ready

        beqz $t1,sync_read              #si ready==0, entra en bucle

        lw $t2,BufferTeclado($t0)       #carga datos a t2
        sb $t2,cadena($t3)              #almacena t2 en cadena,posicion t3

        addi $t3,$t3,1                  #aumenta la posicion en 1

        beq $t2,'\n',fin_read           #comprueba que el ultimo caracter sea \n

        j sync_read
```

```
fin_read:
        jr $ra
```

Ejemplo de ejecucion:



Práctica 11:

Cuestión 7:

Supón que el contenido del registro Cause ($13) tiene los siguientes valores después de haberse producido una excepción. Rellena la tabla 3 indicando cual ha sido la causa que ha provocado la excepción en cada caso.

Cause	Fuente de la excepción
0x00000000	(Int) Interrupción(Hardware)
0x00000020	(SyS0) Excepcion syscall
0x00000024	(Bp) Excepción por punto de ruptura(breakpoint)
0x00000028	(RI) Excepción por instrucción reservada
0x00000030	(Ov) Excepción por desbordamiento aritmético

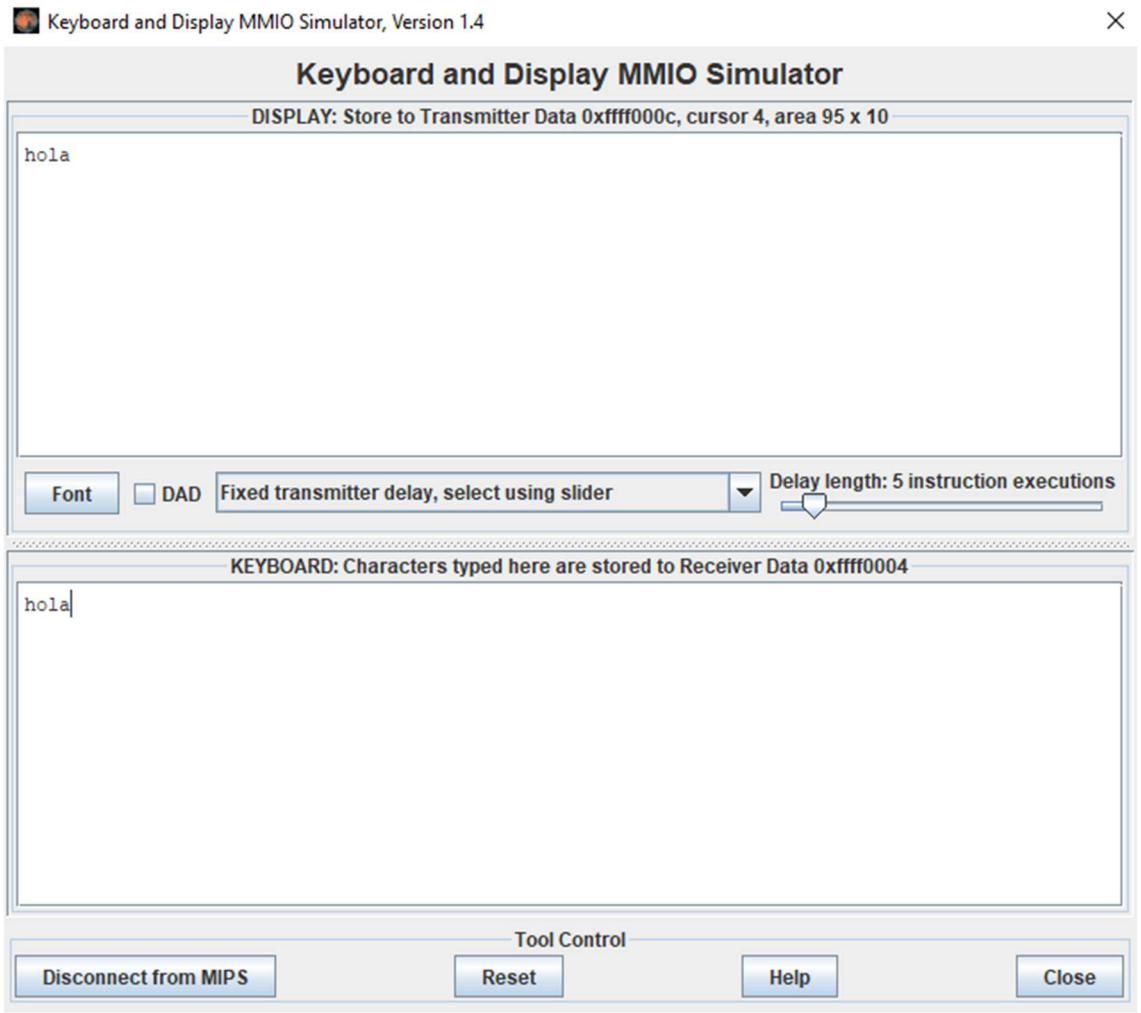
Cuestión 11:

Modifica la rutina de tratamiento de interrupciones para que escriba en el display del transmisor el carácter leído en el receptor. Haz que guarde en el registro $v0 el carácter leído. Escribe un programa principal apropiado para hacer pruebas que finalice cuando en el receptor se pulse un salto de línea

```
# Reserva de espacio para guardar registros en kdata

.kdata

contexto:  .word 0,0,0,0              # espacio para alojar cuatro registros

.ktext 0x80000180                     # Dirección de comienzo de la rutina

 # Guardar registros a utilizar en la rutina.
 la $k1, contexto
 sw $at, 0($k1)                       # Guardamos los registros
 sw $t0, 4($k1)
 sw  $v0, 8($k1)
 sw $a0, 12($k1)

 #Comprobación de si se trata de una interrupción
 mfc0 $k0, $13                        # Registro Cause
 srl $a0, $k0, 2                      # Extraemos campo del código
 andi $a0, $a0, 0x1f
 bne $a0, $zero, acabamos              # Sólo procesamos aquí E/S

 #Tratamiento de la interrupción
lui $t0, 0xffff                       #Direccion de registro de control por teclado
lw $v0, 4($t0)                        # Lee carácter del teclado
sw $v0, 12($t0)                       #Escribe en la consola

 acabamos:
 mtc0 $0, $13                         # Iniciar registro Cause
 mfc0 $k0, $12                        # Leer registre Status
 andi $k0, 0xfffd                     # Iniciar bit de excepción
 ori  $k0, 0x11                       # Habilitar interrupciones
 mtc0 $k0, $12                        # reescribir registre Startus

 # Restaurar registros (menos v0)
  lw $at, 0($k1)                      # Recupero $at
  lw $t0, 4($k1)
  lw $a0, 12($k1)
```

```
# Devolver en el programa de usuario
eret

 .text
main:
 lui $t0,0xffff                  # Dirige del registro de control
 lw $t1,0($t0)                   # Registre de control del receptor
 ori $t1,$t1,0x0002              # Habilitar interrupciones del teclado
 sw $t1,0($t0)                   # Actualizamos registro de control

 mfc0 $a0, $12                   # leer registre Status
 ori $a0, 0xff11                 # Habilitar todas las interrupciones
 mtc0 $a0, $12                   # reescribir el registro status

 beq $v0,'\n',fin                #comprobacion salto de linea

 j main

 fin:
 li $v0, 10
 syscall                         # syscall 10 (exit)
```

<u>Ejemplo de ejecucion:</u>



Cuestión 12:

Escribe una rutina general de tratamiento de excepciones que permita tratar excepciones por desbordamiento aritmético, error por lectura al intentar el acceso a una dirección no alineada e interrupciones de teclado. En los tres casos se tiene que escribir un mensaje en la consola del MARS de la excepción tratada. Escribe el programa de prueba apropiado para probar los tres casos.

```
# Reserva de espacio para guardar registros en kdata
.kdata

contexto:  .word 0,0,0,0,0,0,0                                  # espacio para alojar cuatro registros
error_t: .asciiz "\n ATENCION:Interrupcion por teclado \n \n"
error_d: .asciiz "\n ATENCION: Excepcion por direccion erronea (Numero 4) \n \n"
error_a: .asciiz "\n ATENCION: Excepcion por desbordamiento aritmetico \n \n"

.ktext 0x80000180                                               # Dirección de comienzo de la rutina

 # Guardar registros a utilizar en la rutina.
 la $k1, contexto
 sw $at, 0($k1)                                                 # Guardamos $at
 sw $t0, 4($k1)
 sw  $v0, 8($k1)
 sw $a0, 12($k1)
 sw $t1,16($k1)
 sw $t2,20($k1)
 sw $t3,24($k1)

 #Cargar mensajes de error
 la $t1, error_t
 la $t2, error_d
 la $t3, error_a
 #Comprobación de si se trata de una interrupción
 mfc0 $k0, $13                                                  # Registro Cause

 beq $k0, 256,teclado                                           #Interrupcion por teclado



 beq $k0, 16,direccion                                          #Excepcion por direccion erronea (4)


 beq $k0, 48, desbord                                           #desbordamiento aritmetico

 j acabamos

  #Tratamiento de la interrupción
 teclado:

move $a0,$t1
li $v0,4
syscall
lui $t0, 0xffff                                                 #Direccion de registro de control por teclado
lw $v0, 4($t0)                                                  # Lee carácter del teclado
 j acabamos

direccion:
move $a0,$t2
```

```
li $v0,4
syscall
mfc0 $k0, $14                    # $k0 <= EPC
addiu $k0, $k0, 4                # Incremento de $k0 en 4
mtc0 $k0, $14                    # Ahora EPC apunta a la siguiente instrucción
j acabamos

desbord:
move $a0,$t3
li $v0,4
syscall
mfc0 $k0, $14                     # $k0 <= EPC
addiu $k0, $k0, 4                # Incremento de $k0 en 4
mtc0 $k0, $14                     # Ahora EPC apunta a la siguiente instrucción
 j acabamos
 acabamos:


 # Restaurar registros (menos v0)

 lw $at, 0($k1)                  # Recupero $at
 lw $t0, 4($k1)
 lw $a0, 12($k1)
 lw $t1, 16($k1)
 lw $t2, 20($k1)
 lw $t3, 24($k1)

 # Devolver en el programa de usuario siguiente instruccion

 eret

 .data
elige_1: .asciiz "1-Teclado (entra en bucle hasta detectar interrupcion por teclado)\n"
elige_2: .asciiz "2-Direccion erronea \n"
elige_3: .asciiz "3-Desbordamiento aritmetico\n"
elige_4: .asciiz "4-Salir\n"
elige: .asciiz "Elige la excepcion a probar(1-4): "
vector: .word 1, 3, 5, 7, 11, 13

 .text

main:
 la $t1,elige_1
 la $t2,elige_2
 la $t3,elige_3
 la $t4,elige_4
 la $t5,elige
 #Imprime el menu:
 move $a0,$t1
 li $v0,4
 syscall

 move $a0,$t2
 li $v0,4
 syscall

 move $a0,$t3
 li $v0,4
 syscall
```

```
move $a0,$t4
li $v0,4
syscall

move $a0,$t5
li $v0,4
syscall

li $v0,5
syscall                                  #Elige el nº de opcion


lui $t0,0xffff                           # Dirige del registro de control
lw $t1,0($t0)                            # Registre de control del receptor
ori $t1,$t1,0x0002                       # Habilitar interrupciones del teclado
sw $t1,0($t0)                            # Actualizamos registro de control

mfc0 $a0, $12                            # leer registre Status
ori $a0, 0xff11                          # Habilitar todas las interrupciones
mtc0 $a0, $12                            # reescribir el registro status

beq $v0,1,test_teclado
beq $v0,2,test_direccion
beq $v0,3,test_desbord
beq $v0,4,fin

j main

test_teclado:
 lui $t0,0xffff                          # Dirige del registro de control
lw $t1,0($t0)                            # Registre de control del receptor
ori $t1,$t1,0x0002                       # Habilitar interrupciones del teclado
sw $t1,0($t0)                            # Actualizamos registro de control

mfc0 $a0, $12                            # leer registre Status
ori $a0, 0xff11                          # Habilitar todas las interrupciones
mtc0 $a0, $12                            # reescribir el registro status
beq $v0,'\n',main                        #comprobacion salto de linea
j test_teclado

test_direccion:
la $t0, vector
lw $t0, 3($t0)
j main

test_desbord:
li $t1, 0x7FFFFFFF
addiu $t2, $t1,1                         #Se ignora el desbordamiento
addi $t3, $t1, 1                         #Detecta el desbordamiento
j main

fin:
li $v0, 10
syscall                                  # syscall 10 (exit)
```

Ejemplo de ejecucion:

```
1-Teclado (entra en bucle hasta detectar interrupcion por teclado)
2-Direccion erronea
3-Desbordamiento aritmetico
4-Salir
Elige la excepcion a probar(1-4): 3

 ATENCION: Excepcion por desbordamiento aritmetico

1-Teclado (entra en bucle hasta detectar interrupcion por teclado)
2-Direccion erronea
3-Desbordamiento aritmetico
4-Salir
Elige la excepcion a probar(1-4): 2

 ATENCION: Excepcion por direccion erronea (Numero 4)

1-Teclado (entra en bucle hasta detectar interrupcion por teclado)
2-Direccion erronea
3-Desbordamiento aritmetico
4-Salir
Elige la excepcion a probar(1-4): 1

 ATENCION:Interrupcion por teclado


 ATENCION:Interrupcion por teclado

1-Teclado (entra en bucle hasta detectar interrupcion por teclado)
2-Direccion erronea
3-Desbordamiento aritmetico
4-Salir
Elige la excepcion a probar(1-4): 4

-- program is finished running --
```