

Arquitectura de computadoras



2020

Explicación Práctica 6

Puerto de E/S mapeado en Memoria

CONTROL y **DATA** son direcciones del puerto de E/S y son **fijas**

CONTROL: .word32 0x10000

DATA: .word32 0x10008

Salida de datos – Pantalla alfanumérica

Pantalla alfanumérica

Si queremos imprimir un número

DATA recibe un “*dato*”
CONTROL recibe **1** → la salida será un *entero sin signo*
recibe **2** → la salida será un *entero con signo*
recibe **3** → la salida será un *punto flotante*

Si queremos imprimir un string

DATA recibe una “*dirección*”
CONTROL recibe **4** → la salida será el string (asciiz) apuntado por la “*dirección*”

Si queremos limpiar pantalla

CONTROL recibe **6** → *limpia la pantalla alfanumérica*

Salida de datos – Pantalla gráfica

Pantalla gráfica

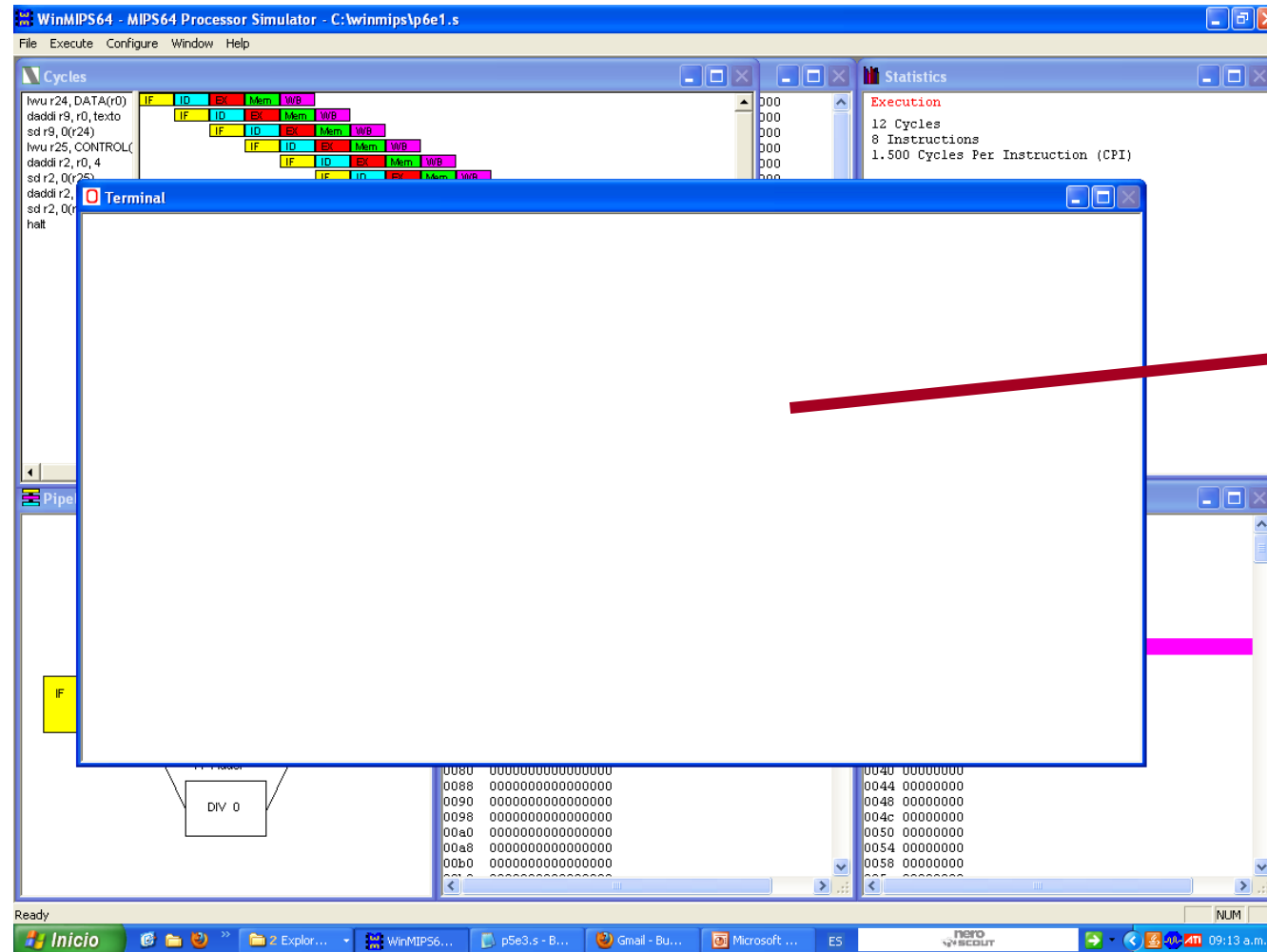
Si queremos pintar un píxel

DATA recibe el color RGB (en 4 bytes)
DATA+4 coordenada Y
DATA+5 coordenada X
CONTROL recibe **5** → *la salida será un píxel de la pantalla gráfica*

Si queremos limpiar pantalla

CONTROL recibe **7** → *se limpia la pantalla gráfica*

WinMIPS – Terminal



Salidas

Ejemplo 1 – Pantalla alfanumérica

.data

```
texto:      .asciiz    "Hola, Mundo!"    ; El mensaje a mostrar
CONTROL:    .word32    0x10000           ; Dirección de mapeo
DATA:       .word32    0x10008           ; Dirección de mapeo
```

.text

```
lwu    $s0, DATA($zero)    ; $s0 = dirección de DATA
daddi   $t0, $zero, texto    ; $t0 = dirección del mensaje a mostrar
sd      $t0, 0($s0)          ; DATA recibe el puntero al comienzo del mensaje
lwu    $s1, CONTROL($zero)   ; $s1 = dirección de CONTROL
daddi   $t0, $zero, 6        ; $t0 = 6 -> función 6: limpiar pantalla alfanumérica
sd      $t0, 0($s1)          ; CONTROL recibe 6 y limpia la pantalla
daddi   $t0, $zero, 4        ; $t0 = 4 -> función 4: salida de una cadena ASCII
sd      $t0, 0($s1)          ; CONTROL recibe 4 y produce la salida del mensaje
halt
```

Ejemplo 2 – Pantalla gráfica

.data

```
coorX:      .byte  24           ; coordenada X de un punto
coorY:      .byte  24           ; coordenada Y de un punto
color:      .byte  255, 0, 255, 0 ; color: máximo rojo + máximo azul => magenta
CONTROL:    .word32 0x10000      ; Dirección de mapeo
DATA:       .word32 0x10008      ; Dirección de mapeo
```


.text

```
lwu  $s6, CONTROL($zero)      ; $s6 = dirección de CONTROL
lwu  $s7, DATA($zero)        ; $s7 = dirección de DATA
daddi $t0, $zero, 7           ; $t0 = 7 -> función 7: limpiar pantalla gráfica
sd   $t0, 0($s6)              ; CONTROL recibe 7 y limpia la pantalla gráfica
lbu  $s0, coorX($zero)        ; $s0 = valor de coordenada X
sb   $s0, 5($s7)              ; DATA+5 recibe el valor de coordenada X
lbu  $s1, coorY($zero)        ; $s1 = valor de coordenada Y
sb   $s1, 4($s7)              ; DATA+4 recibe el valor de coordenada Y
lwu  $s2, color($zero)        ; $s2 = valor de color a pintar
sw   $s2, 0($s7)              ; DATA recibe el valor del color a pintar
daddi $t0, $zero, 5           ; $t0 = 5 -> función 5: salida gráfica
sd   $t0, 0($s6)              ; CONTROL recibe 5 y produce el dibujo del punto
halt
```

Entrada de datos

Si **CONTROL** recibe 8,

DATA poseerá un “*número*” ingresado por teclado (entero o pto flotante.)


- 
- Muestra el carácter presionado en la terminal.
 - **Termina de leer cuando se presiona enter.**
 - Si el dato ingresado no es un número, *guarda 0*

Si **CONTROL** recibe 9,

ASCII



DATA poseerá un “*byte*” ingresado por teclado

- 
- **No** muestra el carácter presionado en la terminal
 - Solo se lee un carácter. **No espera un enter.**

Ejemplo 3 - Entrada de datos (número)

Escriba un programa que solicite el ingreso por teclado de un **número entero mayor a cero**. Se debe verificar el valor ingresado, y si es correcto se debe guardar en **NRO**. Realizar usando la **función de lectura de un número entero**.

.data

```
CONTROL: .word32 0x10000  
DATA:    .word32 0x10008  
NRO:     .word    0
```

.text

```
lwu      $s0, CONTROL($zero)  
lwu      $s1, DATA($0)  
daddi    $s2, $zero, 8  
sd        $s2, 0($s0)  
ld        $s2, 0($s1)  
beqz     $s2, FIN  
sd        $s2, NRO($zero)  
FIN:     halt
```

Ejemplo 4 - Entrada de datos (carácter)

Escriba un programa que solicite el ingreso por teclado de un **número entero mayor a cero de un sólo dígito**. Se debe verificar el valor ingresado, y si es correcto se debe guardar en **NRO**. Realizar usando la **función de lectura de un carácter**.

```
.data
CONTROL: .word32 0x10000
DATA:    .word32 0x10008
NRO:     .word   0
```

```
.text
lwu $s0, CONTROL($zero)
lwu $s1, DATA($0)
daddi $s2, $zero, 9
sd $s2, 0($s0)
lbu $s2, 0($s1)
slti $s3, $s2, 0x31
bnez $s3, FIN
slti $s3, $s2, 0x3A
beqz $s3, FIN
daddi $s2, $s2, -0x30
sb $s2, NRO($zero)
FIN: halt
```