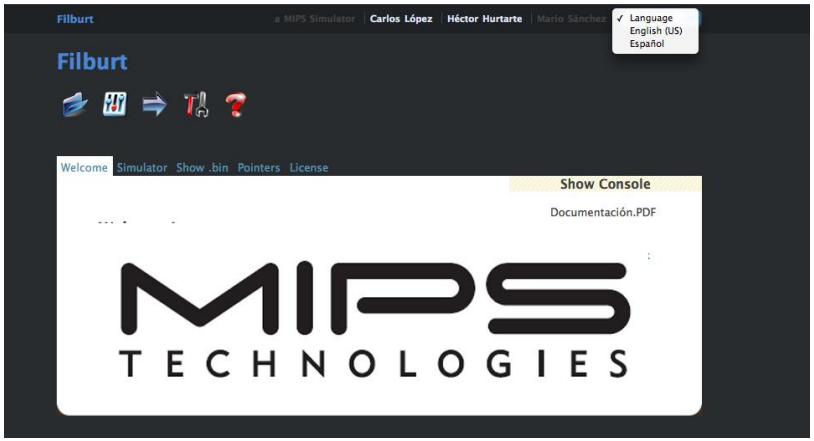
Universidad del Valle de Guatemala

Proyecto Final

Organización de las Computadoras

CC3001



MANUAL DE USUARIO

Carlos Eduardo López Camey (#08107)

Héctor Hurtarte (#08119)

Mario Sánchez (#06089)

Manual de Usuario
FILBURT

Introducción

A continuación se presenta el manual de usuario de MIPS, en el encontrara cada una de las instrucciones utilizadas, y en cada una de ellas un ejemplo, para que sea de mejor entendimiento.
Así como utilizar MIPS, interfaz grafica; cargar archivos, ejecutar programa, debuggear y reinicializar maquina.

MIPS

- **Ejecutar programa: Si desea que el programa empiece a ejecutarse el usuario debe de seleccionar el link anterior, en ese momento el programa empezara a ejecutarse sin ningún problema. Este proceso será de corrido ejecutara todo el programa de un solo sin interrupciones.**

- **Cargar Archivos: En la interfaz encontrara este icono, el cual sirve para cargar archivos únicamente con formato .asm, cualquier archivo que no cumpla con este formato no lo reconocerá como un archivo ejecutable.**

- **Reinicializar maquina: Si en algún momento el usuario desea reiniciar la maquina, únicamente haciendo click al siguiente icono podrá reiniciar todo de nuevo.**

- **Debugg: Este icono sirve para ir ejecutando paso por paso, a diferencia de ejecutar, uno puede observar el proceso del programa.**

Conjunto de instrucciones del lenguaje ensamblador

ADD
Descripción: Suma el contenido de dos registros y almacena el resultado en otro registro.
Sintaxis:
ADD $sd $rs $rt
Operación:
$rd = $rs + $rt
Ejemplo:
ADD $s0 $s1 $s2
Codificado del ejemplo:
0000 0001 0000 1001 0101 0000 0010 0000

ADDI
Descripción: Suma el contenido de un registro con un valor inmediato de 16 bits (IMM) y lo almacena en otro registro.
Sintaxis:
ADDI $rt $rs IMM(16bits)
Operación:
$rt = $rs + IMM(16bits)
Ejemplo en HEX:
ADDI $s0 $s1 xFFFF
Ejemplo en DEC:
ADDI $s0 $s1 %-1
Codificado del ejemplo:
0010 0001 0000 1001 1111 1111 1111 1111

AND

Descripción: Operación binaria que hace “AND” del contenido de dos registros y almacena el resultado en otro registro.
Sintaxis:
AND $rd $rs $rt
Operación:
$rd = $rs AND $rt
Ejemplo:
AND $s0 $s1 $s2
Codificado del ejemplo:
0000 0001 0000 1001 0101 0000 0010 0100

ANDI

Descripción: Operación binaria que hace “AND” del valor contenido en un registro con un valor inmediato (IMM) y almacena el resultado en otro registro.
Sintaxis:
ANDI $rt $rs IMM(16bits)
Operación:
$rt = $rs AND IMM(16bits)
Ejemplo en HEX:
ANDI $s0 $s1 xFFFF
Ejemplo en DEC:
ANDI $s0 $s1 %-1
Codificado del ejemplo:
0011 0001 0000 1001 1111 1111 1111 1111

BEQ

Descripción: Bifurcación en el caso que el contenido de dos registros sean iguales (el offset puede estar definido por una etiqueta).
Sintaxis:
BEQ $rs $rt offset(16bits)
Operación:
if (rs == rt) then PC = PC + offset(32bits)
else PC++
Ejemplo:
BEQ $s0 $s1 xFFFF
Codificado del ejemplo:
0001 0001 0000 1001 1111 1111 1111 1111

BGEZ

Descripción: Bifurcación en el caso que el contenido de un registro sea mayor o igual que cero (el offset puede estar definido por una etiqueta).
Sintaxis:
BGEZ $rs offset(16bits)
Operación:
if ($rs >= 0) then PC = PC + offset(32bits)
else PC++
Ejemplo:
BGEZ $s0 xFFFF
Codificado del ejemplo:
0000 0101 0000 0001 1111 1111 1111 1111

BGTZ
Descripción: Bifurcación en el caso que el contenido de un registro sea mayor que cero (el offset puede estar definido por una etiqueta).
Sintaxis:
BGTZ $rs offset(16bits)
Operación:
if (rs > 0) then PC = PC + offset(32bits)
else PC++
Ejemplo:
BGTZ $s0 xFFFF
Codificado del ejemplo:
0001 1101 0000 0000 1111 1111 1111 1111

BLEZ
Descripción: Bifurcación en el caso que el contenido de un registro sea menor o igual que cero.
Sintaxis:
BLEZ $rs offset(16bits)
Operación:
if (rs <= 0) then PC = PC + offset(32bits)
else PC++
Ejemplo:
BLEZ $s0 xFFFF
Codificado del ejemplo:
0001 1001 0000 0000 1111 1111 1111 1111

BLZ
Descripción: Bifurcación en el caso que el contenido de un registro sea menor que cero (el offset puede estar definido por una etiqueta).
Sintáxis:
BLTZ $rs offset(16bits)
Operación:
if (rs < 0) then PC = PC + offset(32bits)
else PC++
Ejemplo:
BLTZ $s0 xFFFF
Codificado del ejemplo:
0001 1001 0000 0000 1111 1111 1111 1111

BNE
Descripción: Bifurcación en el caso que el contenido de dos registros sean diferentes.
Sintáxis:
BNE $rs, $rt, offset(16bits)
Operación:
if (rs <> rt) then PC = PC + offset(32bits)
else PC++
Ejemplo:
BNE $s0 $s1 xFFFF
Codificado del ejemplo:
0001 1001 0000 1001 1111 1111 1111 1111

J
Descripción: Salta a la dirección especificada por una dirección inmediata o en base a la dirección de una etiqueta.
Sintáxis:
J instr_index (26bits)
Operación:
PC = instr_index (32bits)
Ejemplo:
J <label>
Si <label> está relacionada con la posición de memoria X, entonces los bits instr_index, contienen el offset para llegar a ésta dirección.
Codificado del ejemplo:
0000 0100 0000 0000 0000 0000 0000 0000

JR
Descripción: Salta a la dirección contenida en un registro.
Sintaxis:
JR $rs
Operación:
PC = $rs
Ejemplo:
J $s0
Codificado del ejemplo:
0000 0001 0000 0000 0000 0000 0000 1000

JAL
Descripción: Salta a la dirección especificada por una dirección inmediata o en base a la dirección de una etiqueta. Almacena en el registro $ra la dirección siguiente (a donde se debe retornar).
Sintáxis:
JAL instr_index (26bits)
Operación:
$ra = PC
PC = instr_index (32bits)
Ejemplo:
J <label>
Si <label> está relacionada con la posición de memoria X, entonces los bits instr_index, contienen el offset para llegar a ésta dirección.
Codificado del ejemplo:
0000 0100 0000 0000 0000 0000 0000 0000

JALR
Descripción: Salta a la dirección contenida en un registro. Almacena en el registro $ra la dirección siguiente (a donde se debe retornar).
Sintaxis:
JR $rs
Operación:
$ra = PC
PC = $rs
Ejemplo:
J $s0
Codificado del ejemplo:
0000 0001 0000 0000 0000 0000 0000 1000

LW

Descripción: Una palabra (32 bits) es cargada en un registro con el contenido de la dirección de memoria especificada como PC + offset(32bits). Este offset puede ser calculado en base a una etiqueta.

Sintaxis:

LW $rt, offset(16)

Operación:

$rt = Mem[PC+ offset(32)]

Ejemplo en HEX:

LW $s0, %3

Codificado del ejemplo:

1000 1101 0010 1000 0000 0000 0000 0011

SW

Descripción: Almacena una palabra en una dirección de memoria calculada como PC+offset(32bits). Este offset puede ser calculado en base a una etiqueta.

Sintaxis:

SW $rt offset(16bits)

Operación:

Mem[PC + offset(32bits)] = rt

Ejemplo en HEX:

SW $s0, %4($s1)

Codificado del ejemplo:

1010 1101 0010 1000 0000 0000 0000 0100

Etiquetas

Las etiquetas en la MIPS a implementar poseen un requerimiento, que el nombre de la etiqueta este separada por un espacio y que el contenido tiene que está entre comillas "" si es texto, porcentaje % si es decimal y x si es entero.

.DATA *dirección:* indica que lo siguiente son datos, por default inicia en la siguiente localidad de memoria disponible. Si el parámetro opcional *dirección* está presente, entonces los datos comienzan en esa dirección

.TEXT *dirección:* indica que lo siguiente son necesariamente instrucciones. Y pasa lo mismo que con .DATA con respecto al parámetro opcional *dirección*.

.ASCII *string*: ensambla el *string* siguiente, un carácter en cada localidad de memoria, no lo termina con un caracter *null*.

.ASCIIZ *string*: ensambla el *string* siguiente, un carácter en cada localidad de memoria, si lo termina con un caracter *null*.

.SPACE *num*: reserva las siguientes *num* localidades de memoria.

.WORD *num*: llena la localidad de memoria con el contenido definido por el parámetro que debe de ser un número decimal o hexadecimal

Registros

Nombre simbólico	Numero	Uso
$v1	2-3	Registro de resultado
$a0-$a3	4-7	Registros de parámetro 1…..4
$t0-$t9	8-15 , 24-25	Registros temporales 0…..9
$s0-$s7	16-23	Guarda los registros 0….7
$ra	31	Regresa la dirección