El ensamblador para el ARC

Este archivo contiene las instrucciones para operar el ensamblador ARC

I Que es el ensamblador de ARC
El ensamblador de ARC es un programa traductor de lenguaje simbólico de ARC de dos pasadas.
Produciendo un archivo bin y un archivo listado.
Comentarios comienzan con el signo de admiración ! y terminan al finalizar la línea.
El programa realiza referencias simbólicas y cálculos aritméticos.

II Los siguientes códigos de operación y pseudo operaciones son reconocidas:

"nop"
“halt"
"sethi"
"be" "bcs" "bneg" "bvs" "ba" "bne" "bcc" "bpos" "bvc"
"call"
"jmpl"
"addcc" "andcc" "subcc" "orcc" "orncc" "xorcc"
"srl" "sll" "sra"
"add" "sub" "and" "or" "orn" "xor"
"ld" "st"
".dwb"
".begin"
".end"
".org"
".equ"

"nop" no realiza ninguna operación pero incrementa el contador de programa.

"halt" para el simulador.

"sethi" establece los 22 bits mas representativos y coloca ceros en los 10 bits menos significativos.
Si el operando es 0 y el registro %r0, entonces la instrucción se comporta como un no operación (nop).
Ejemplo de utilización: sethi 0x304F15, %r1. Dando: establecer los 22 bits mas representativos de %r1 a 0x304F15 y colocar los 10 bits de abajo a cero.

"be" saltar si igual a cero. Si el indicador Z es 1, entonces saltar a la dirección indicada por el rotulo que es el operando de la instrucción.
Ejemplo de utilización: be rotulo da: saltar al rotulo si Z es 1.

"bcs" saltar si C es 1. Si el indicador C es 1, entonces saltar a la dirección indicada por el rotulo que es el operando de la instrucción.
Ejemplo de utilización: bcs rotulo da: saltar al rotulo si C es 1.

"bcc" saltar si C es 0. Si el indicador C es 0, entonces saltar a la dirección indicada por el rotulo que es el operando de la instrucción.
Ejemplo de utilización: bcc rotulo da: saltar al rotulo si C es 0.

"bneg" saltar si negativo. Si el indicador N es 1, entonces saltar a la dirección indicada por el rotulo que es el operando de la instrucción.
Ejemplo de utilización: bneg rotulo da: saltar al rotulo si N es 1.

"bvs" saltar si hay rebalse (overflow). Si el indicador V es 1, entonces saltar a la dirección indicada por el rotulo que es el operando de la instrucción.
Ejemplo de utilización: bvs rotulo da: saltar al rotulo si V es 1.

"bvc" saltar si no hay rebalse (overflow). Si el indicador V es 0, entonces saltar a la dirección indicada por el rotulo que es el operando de la instrucción.
Ejemplo de utilización: bvs rotulo da: saltar al rotulo si V es 0.

"bne" saltar si no igual. Saltar a la dirección indicada por el rotulo que es el operando de la instrucción si no igual a cero.
Ejemplo de utilización: bne rotulo da: saltar al rotulo si no igual a cero.

"bpos" saltar si positivo. Saltar a la dirección indicada por el rotulo que es el operando de la instrucción si no igual a cero.
Ejemplo de utilización: bpos rotulo da: saltar al rotulo si positivo.

"ba" saltar siempre. Saltar a siempre a la dirección indicada por el rotulo que es el operando de la instrucción.
Ejemplo de utilización: ba rotulo da: saltar al rotulo siempre.

"call" llamar a subrutina y guardar la dirección de la instrucción actual en %r15. El operando de la instrucción es la dirección de la subrutina y se guarda como un desplazamiento de 30 bits en la instrucción
Ejemplo de utilización: call sub_r da: llamar a la subrutina ubicada en sub_r.

“addcc” y “add” sumar enteros, los operandos fuentes y guarda el resultado en el operando destino utilizando aritmética de complemento a dos. “addcc” cambia los bits de condición de acuerdo al resultado.
Ejemplo de utilización: add %r1, %r2, %r4 da %r4 ← %r1 + %r2.
Ejemplo de utilización: addcc %r1, 2, %r2 da %r2 ← %r1 + 2.

“subcc” y “sub” restar enteros, los operandos fuentes y guarda el resultado en el operando destino utilizando aritmética de complemento a dos. “subcc” cambia los bits de condición de acuerdo al resultado.
Ejemplo de utilización: sub %r1, %r2, %r4 da %r4 ← %r1 - %r2.
Ejemplo de utilización: subdcc %r1, 2, %r2 da %r2 ← %r1 - 2.

"andcc" y "and" AND bit a bit los operandos fuentes y guardar en la dirección destino. "andcc" cambia los bits de condición N y Z de acuerdo al resultado. Ejemplo de utilización: and %r1, %r2, %r4. Da: %r4 ← %r1 AND %r2

"orcc" y "or" OR bit a bit los operandos fuentes y guardar en la dirección destino. "orcc" cambia los bits de condición N y Z de acuerdo al resultado.
Ejemplo de utilización: or %r1, %r2, %r4. Da: %r4 ← %r1 OR %r2
Ejemplo de utilización: orcc %r1, 1, %r2. Da: %r4 ← %r1 OR 1

"orncc" y "orn" NOR bit a bit los operandos fuentes y guardar en la dirección destino. "orncc" cambia los bits de condición N y Z de acuerdo al resultado.
Ejemplo de utilización: or %r1, %r2, %r4. Da: %r4 ← %r1 NOR %r2
Ejemplo de utilización: orcc %r1, 1, %r2. Da: %r4 ← %r1 NOR 1

"xor" y "xorcc" XOR bit a bit los operandos fuentes y guardar en la dirección destino. "xorcc" cambia los bits de condición N y Z de acuerdo al resultado.
Ejemplo de utilización: %r1, %r0, %r1. Da: %r1 ← %r1 XOR %r0

"srl" desplaza a la derecha los bits del registro, de 0 a 31bits, los bits de la izquierda se llenan con ceros.
Ejemplo de utilización: srl %r1, 3, %r4. Da: desplazar %r1 3 bits a la derecha y guardar en %r4.
Ejemplo de utilización: srl %r1, %r4, %r5. Da desplazar el registro %r1 a la derecha tantos lugares como el contenido de %r4 y guardar en %r5.

"sll" desplaza a la izquierda los bits del registro, de 0 a 31bits, los bits de la derecha se llenan con ceros.
Ejemplo de utilización: sll %r1, 3, %r4. Da: desplazar %r1 3 bits a la izquierda y guardar en %r4.
Ejemplo de utilización: srl %r1, %r4, %r5. Da desplazar el registro %r1 a la izquierda tantos lugares como el contenido de %r4 y guardar en %r5.

"sra" desplaza a la derecha los bits del registro, de 0 a 31bits, el bit de signo se replica a medida que se desplaza a la derecha.
Ejemplo de utilización: sra %r1, 3, %r4. Da: desplazar %r1 3 bits a la derecha y guardar en %r4, replicando el bit de signo.
Ejemplo de utilización: sra %r1, %r4, %r5. Da desplazar el registro %r1 a la derecha tantos lugares como el contenido de %r4 y guardar en %r5, replicando el bit de signo.

"ld" cargar un registro de la memoria central. La dirección de memoria debe estar alineada a palabra.
Ejemplo de utilización: ld [x], %r1
ld [x], %r0, %r1
ld %r0+x, %r1
Da: copiar los contenidos de la dirección x de memoria en el registro %r1. (%r0
es siempre cero)

"st" guardar el contenido del registro en la memoria central. La dirección de memoria debe estar alineada a palabra.
Ejemplo de utilización: st %r1, [x]
st %r1, %r0 + x
st %r1, %r0, x
Da: guardar los contenidos del registro %r1 en la dirección x de memoria %r1. (%r0
es siempre cero)

"jmpl" incondicional, transferencia de control indirecta por registro. Saltar a una nueva dirección y guardar la dirección de la instrucción actual en el registro destino.
Ejemplo de utilización:: jmpl %r15 + 4, %r2
Da: cargar en el contador de programa los contenidos de %r15 + 4. La dirección actual se guarda en %r2.

Existen las siguientes pseudo operaciones:

.dwb n. Define Word Block: reservar espacio para un bloque de n palabras.
.begin, .end: comenzar y terminar de ensamblar respectivamente.
Símbolo .equ valor iguala el valor al Símbolo en la tabla de símbolos.

Símbolo no debe continuarse con dos puntos (☺ porque no es un rotulo del programa
Ejemplo de utilización
WS .equ 32

.org símbolo cambiar el contador de locaciones por la dirección especificada por símbolo.
Ejemplo de utilización:
.org 0x1000 la siguiente instrucción se traducirá en la ubicación 0x1000.
Start .equ 0x2000
.org Start ! la siguiente instrucción se ubicará en 0x2000
Comentarios comienzan luego del símbolo ! y continua hasta el fin de la línea.
Ejemplo de utilización:
nop ! esta es la instrucción de no operar.

III. Este traductor funciona en cualquier computadora que ejecute aplicaciones de Java 1.1. Se ha probado en Macintosh, Windows NT y Solaris.

IV. Como traduzco un programa en ARC. Para utilizar el traductor utilizando el simulador de ARC.
Los archivos fuentes se pueden nombrar de la forma que acepta el sistema operativo que se esta utilizando. Pero la ex tensión .asm debe estar incluida en el nombre, por ejemplo programa2.asm. El traductor producirá dos archivos: programa2.tst, el archivo con el listado y si el resultado es exitoso el archivo binario programa2.bin .

Ejecutar el simulador y seleccionar el botón **Edit**. Se habilita el editor de texto, si se cargo un archivo binario al simulador se abrirá el archivo correspondiente .asm. Si no hay nada cargado se debe cargar un archivo o editar uno nuevo. Seleccionando el botón **Assemble** invoca el traductor y este o colocará los errores encontrados o el programa traducido.