El lenguaje ensamblador no dispone de estructuras de control de flujo de programa definidas, que permitan decidir entre dos (o varios) caminos de ejecución de instrucciones distintos (por ejemplo la sentencia if de otros lenguajes de programación como C/C++, JAVA, etc.). Normalmente para implementar cualquier estructura de este tipo es necesario evaluar previamente una condición, simple o compuesta. El camino que seguirá la ejecución del programa dependerá del resultado de esta evaluación.

En éste capítulo se describe como se evalúan las condiciones en ensamblador del MIPS R2000. En primer lugar se realiza un breve repaso al conjunto de instrucciones y pseudoinstrucciones que tiene el MIPS R2000 para realizar comparaciones y control del flujo de programa. A continuación, se proponen varios ejemplos que implementan fragmentos de código para evaluar diversas condiciones

simples y compuestas.

**INSTRUCCIONES DE SALTO CONDICIONAL:**

**BÁSICAS:**

**beq rs, rt , etiqueta y bne rs, rt** , etiqueta  **à**  comparan el contenido de los registros rs y rt y según su igualdad o no, saltan a la instrucción referenciada por etiqueta o no.

**RESPECTO A CERO:**

**[bgez, bgtz, blez, bltz] rs etiqueta**  **à**  comparan el contenido de rs con cero y saltan a la instrucción referenciada por etiqueta si rs >= 0 (bgez), rs > 0 (bgtz), rs <= 0 (blez) y rs < 0 (bltz).

**PSEUDOINSTRUCCIONES:**

[**bge, bgt, ble, blt] rs, rt, etiqueta**  **à**  comparan el contenido de los registros rs y rt y según su relación saltan a la instrucción referenciada por etiqueta si rs >= rt (bge), si rs > rt (bgt), si rs <= rt (ble) y si rs < rt (blt).

**INSTRUCCIONES DE COMPARACIÓN:**

**BÁSICA:**

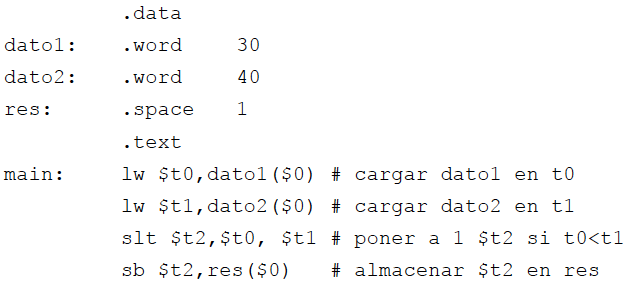
**slt rd, rs , rt à** el registro rd se pondrá a 1 si el contenido del registro rs es menor que el contenido del registro rt y se pondrá a 0 en caso contrario.

**PSEUDOINSTRUCCIONES:**

**[sge, sgt, sle, sne, seq] rd, rs, rt à** comparan el contenido de los registros rs y rt y según su relación ponen un 1 o un 0 en rd. En concreto: poner 1 si rs >= rt (sge), poner 1 si rs > rt (sge), poner 1 si rs <= rt (sle), poner 1 si rs distinto de rt (sne) y poner 1 si rs == rt (seg).

**Evaluación de condiciones simples: mayor, menor, mayor o igual, menor o igual**

**1.-** Crea un fichero con el siguiente código que compara las variables dato1 y dato2 y deja el resultado en la variable booleana res:



**Responde a las siguientes cuestiones:**

A) ¿Qué valor se carga en la posición de memoria res?

-Se carga un 1.Ya que 30 es menor que 40 y la instrucción slt ,regula que si 30 es menor que 40 la etiqueta res tenga un valor de 1,ese 1 viene del registro $t2.

B) Inicializa las posiciones de memoria dato1 y dato2 con los valores 50 y 20 respectivamente. ¿Qué valor se carga ahora en la posición de memoria res?

-Se carga un 1.Ya que 50 es menor que 20y la instrucción slt ,regula que si 50 es menor que 20 la etiqueta res tenga un valor de 1.Pero como es menor,el valor del registro $t2,que es 0 se guarda en res.

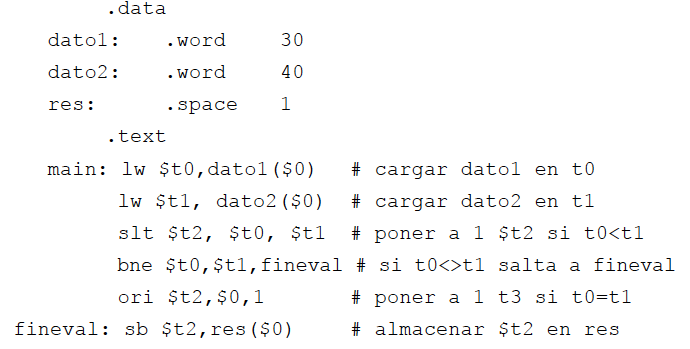
C) ¿Qué comparación se ha evaluado entre dato1 y dato2?

-Se ha usado la comparación slt rd, rs, rt,activa si menor,que pone el registro rd a 1 si rs es menor que rt y a 0 en caso contrario

D) Modifica el código para evaluar la condición res(1)  (dato1 == dato2).

E) Resuelve la cuestión anterior utilizando la pseudoinstrucción sge.

**2.-** Crea un fichero con el siguiente código que compara las variables dato1 y dato2 y deja el resultado en la variable booleana res:



**Responde a las siguientes cuestiones:**

A) ¿Qué valor se carga en la posición de memoria res?

-1 se carga en res.

B) Inicializa las posiciones de memoria dato1 y dato2 con los valores 50 y 20 respectivamente. ¿Qué valor se carga ahora en la posición de memoria res?

-Se carga el valor 0 ya que 50 y 20 son distintos entonces en la bne salta a fineval y queda con el valor 0.

C) Inicializa las posiciones de memoria dato1 y dato2 con los valores 20 y 20 respectivamente. ¿Qué valor se carga ahora en la posición de memoria res?

-Se carga 1,ya que la instrucción ori dice que si los dos datos son iguales se pone a 1; es 20 y 20,entonces devuelve un 1.

D) ¿Qué comparación se ha evaluado entre dato1 y dato2?

Como dato1 es menor o igual que dato2, cambia a uno, y si es mayor, se cambia a cero. Comparacion menor igual.

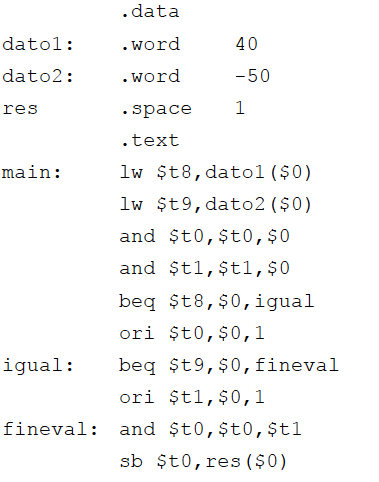
E) Evalúa esta comparación utilizando pseudoinstrucciones.

F) Modifica el código para evaluar la condición res(1) ß (dato1 >= dato2) sin utilizar pseudoinstrucciones.

G) Modifica el código anterior utilizando pseudoinstrucciones.

**Evaluación de condiciones compuestas por operadores lógicos “and” y “or”:**

**3.-** Crea un fichero con el siguiente código que compara las variables dato1 y dato2 y deja el resultado en la variable booleana res:



A) ¿Qué valor se carga en la posición de memoria res?

Se carga 1.

B) Inicializa las posiciones de memoria dato1 y dato2 con los valores 0 y 20 respectivamente. ¿Qué valor se carga ahora en la posición de memoria res?

-Se carga 0.

C) Inicializa las posiciones de memoria dato1 y dato2 con los valores 20 y 0 respectivamente. ¿Qué valor se carga ahora en la posición de memoria res?

-Se carga 0.

D) Inicializa las posiciones de memoria dato1 y dato2 con los valores 0 y 0 respectivamente. ¿Qué valor se carga ahora en la posición de memoria res?

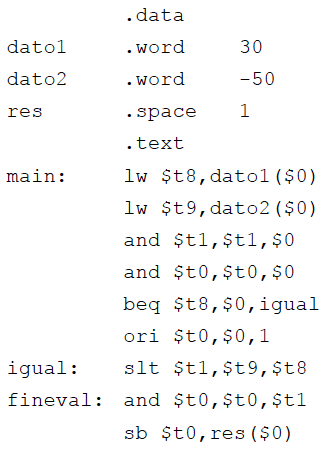
-Se carga 0.

E) ¿Qué comparación compuesta se ha evaluado entre dato1 y dato2?

Se compara si son iguales.

F) Modifica el código anterior para que la condición evaluada sea res(1) ß ((dato1 <> 0) and (dato1 <> dato2))

**4.-** Crea un fichero con el siguiente código que compara las variables dato1 y dato2 y deja el resultado en la variable booleana res:



A) ¿Qué valor se carga en la posición de memoria res?

-Se carga 1.

B) Inicializa las posiciones de memoria dato1 y dato2 con los valores 10 y 20 respectivamente. ¿Qué valor se carga ahora en la posición de memoria res?

-Se carga 0.

C) Inicializa las posiciones de memoria dato1 y dato2 con los valores 0 y -20 respectivamente. ¿Qué valor se carga ahora en la posición de memoria res?

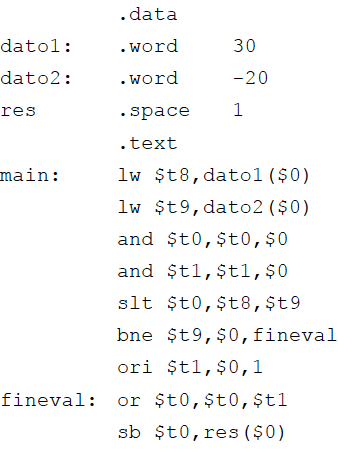
-Se carga 0.

D) ¿Qué comparación compuesta se ha evaluado entre dato1 y dato2?

E) Modifica el código anterior para que la condición evaluada sea res(1) ß ((dato1 <> dato2) and (dato1 <= dato2)).

F) Modifica el código anterior utilizando la pseudoinstrucción sle.

**5.-** Crea un fichero con el siguiente código que compara las variables dato1 y dato2 y deja el resultado en la variable booleana res:



A) ¿Qué valor se carga en la posición de memoria res?

B) Inicializa las posiciones de memoria dato1 y dato2 con los valores -20 y 10 respectivamente. ¿Qué valor se carga ahora en la posición de memoria res?

C) Inicializa las posiciones de memoria dato1 y dato2 con los valores 10 y 0 respectivamente. ¿Qué valor se carga ahora en la posición de memoria res?

D) Inicializa las posiciones de memoria dato1 y dato2 con los valores 20 y 10 respectivamente. ¿Qué valor se carga ahora en la posición de memoria res?

E) ¿Qué comparación compuesta se ha evaluado entre dato1 y dato2?

F) Modifica el código anterior para que la condición evaluada sea res(1) ß ((dato1 <= dato2) and (dato1 <= 0))

G) Modifica el código anterior utilizando la pseudoinstrucción sle.

**6.-** Escriba un programa mayor.s que lea dos números por consola y muestre por pantalla el más elevado de los dos.

**Recordatorio:** las llamadas al sistema que permiten la lectura y escritura de información aparecen en la siguiente tabla:

