

EJERCICIOS DEL CAPITULO 7:

EL MODELO DE PENTIUM PARA EL PROGRAMADOR DE APLICACIONES

EJERCICIO 1

Un Pentium 4 a 2 Ghz en un ambiente multitarea se halla ejecutando un programa de un segmento de código de una de dichas tareas que se compone de las siguientes instrucciones:

```
        mov  efds, F0F0 F0F0 H
        movi edx, 3F H
        in
rep     movs
        mov fs, dx
        movi eax, 6A H
        add eax, edx
```

- 1) ¿ Explicar con detalle la operación que realiza la instrucción in?
- 2) ¿ Qué condición se debe cumplir para ejecutar in? ¿ Se cumple?
- 3) ¿ En cuál de los 4 modos de trabajo posibles está funcionando el Pentium?
- 4) Al terminar la ejecución de la última de las instrucciones arriba indicadas, ¿ Cuánto vale el flag AF?
- 5) ¿ Y los flags ZF, SF y PF?, ¿ porqué?
- 6) ¿Este microprocesador es capaz de ejecutar la instrucción CUID? ¿ qué hace esa instrucción?
- 7) Explicar el desarrollo de la instrucción “rep movs”. ¿ Al trabajar con cadenas de caracteres las explota con post-incremento o post-decremento de las direcciones?
- 8) ¿ Este programa se ejecuta todo seguido o instrucción por instrucción?
- 9) ¿ Qué característica especial tiene la tarea que se está ejecutando?
- 10) Sin contar la instrucción “rep movs” ¿cuál de las otras es la que más duración ocupa y por qué?
- 11) Indicar como calcula el procesador la dirección de la Memoria Principal donde se encuentra la última instrucción del programa de arriba
- 12) Si se conoce que el CPI medio de este programa es 3,42 ¿ cuánto tiempo tarda en ejecutarse?
- 13) ¿ Cuántos MIPS nativos tiene rendimiento este procesador para este programa?

Resolución

1) ¿ Explicar con detalle la operación que realiza la instrucción in?

En este caso la operación que realiza es:

$in\ eax \leftarrow edx$

Introduce al acumulador lo que contenga la puerta de salida que yo use como referencia, en este caso es (3F).

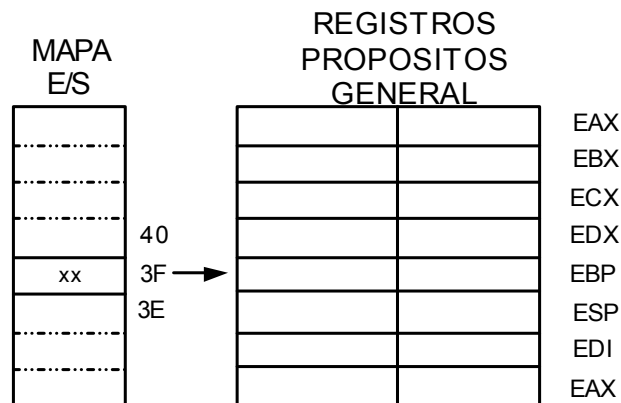


Figura 7.1. Se carga un valor de una posición del Mapa de E/S en EAX mediante la instrucción IN.

2) ¿ Qué condición se debe cumplir para ejecutar in? ¿ Se cumple?

Como es una instrucción protegida, se tendrá que ejecutar en un segmento de código cuyo nivel de privilegio (PL) sea igual o mayor que el valor de IOPL. En este caso como el valor del IOPL es 11 y el PL de valor 11 corresponde al mínimo nivel de privilegio, no habrá ningún problema para ejecutar esta instrucción en cualquiera de los niveles.

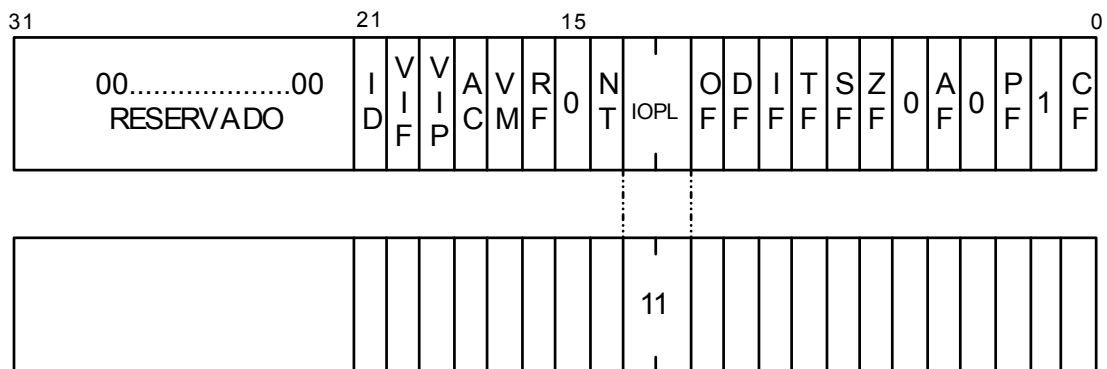


Figura 7.2. Registro Eflags con valor de privilegio 11.

3) ¿ En cuál de los 4 modos de trabajo posibles está funcionando el Pentium?

Como nos encontramos en ambiente multitarea, el bit VM contenido en EFLAGS está a 0, lo cual nos indica que no estamos en modo virtual. Por otro lado no tenemos ninguna instrucción que haya pasado de modo protegido a especial, por eso nos quedamos con el modo protegido.

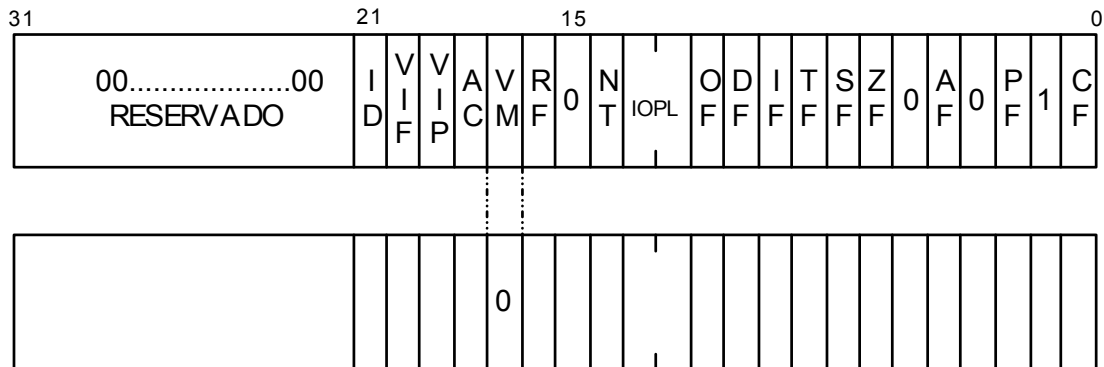


Figura 7.2. Registro Eflags con valor de modo virtual 0

4) Al terminar la ejecución de la última de las instrucciones arriba indicadas, ¿ Cuánto vale el flag AF?

El bit del flag AF tiene como objetivo indicarnos si hay acarreo en las operaciones de sumas. En este caso la instrucción realiza la siguiente operación:

$$\begin{array}{rcl}
 \text{EAX} & 6\text{A} & \\
 +\text{EDX} & \rightarrow 3\text{F} & \rightarrow \\
 & & \begin{array}{r}
 1 \\
 \swarrow \\
 \begin{array}{r}
 0110 1010 \\
 + 0011 1111 \\
 \hline
 1010 1001
 \end{array}
 \end{array}
 \end{array}$$

Como hay acarreo en el cuarto bit de la suma el flag AF se pondrá a 1.

5) ¿ Y los flags ZF, SF y PF?, ¿ porqué?

ZF→El flag ZF nos indica si el resultado de la operación es 0. En este caso como hemos podido comprobar en el apartado anterior, el resultado de la suma ha sido diferente a 0 por lo que ZF será igual a 0.

SF→El flag SF nos indica el signo. En el caso de trabajar sin signo, toma el valor mas alto del resultado. En este caso SF será igual a 0 porque estamos trabajando con 32 bits, ya que EAX y EDX son registros de 32 bits.

PF→El flag PF nos indica el bit de paridad. Para ello el numero de unos del resultado más el que contenga el flag PF ha de ser impar. Si contamos los unos del resultado del apartado anterior veremos que el numero de unos es igual a cuatro, en consecuencia PF será igual a 1.

6) ¿Este microprocesador es capaz de ejecutar la instrucción CUID? ¿ qué hace esa instrucción?

Para saber si nuestro microprocesador es capaz de utilizar esa instrucción deberá mirar el bit ID que viene dentro del EEFLAG. Como el valor del ID es igual a 1 si podrá ejecutarla. El que nuestro procesador pueda ejecutar la CUID conlleva que la CPU informe el software del modelo del microprocesador en que se esta ejecutando.

7) Explicar el desarrollo de la instrucción “rep movs”. ¿ Al trabajar con cadenas de caracteres las explota con post-incremento o post-decremento de las direcciones?

El prefijo *rep* se pone delante de ciertas instrucciones con la intención que se repitan tantas veces como el valor que contenga el registro ECX.
La instrucción *mov* realiza un movimiento de string. Los punteros de direcciones apuntan arriba o debajo de la dirección del string dependiendo que se vaya a hacer en post-incremento o post-decremento. Para saber cual de los dos se va a utilizar deberemos mirar el bit DF cuyo valor en este caso es 0.

8) ¿ Este programa se ejecuta todo seguido o instrucción por instrucción?

Para saber si un programa se va a ejecutar seguido o instrucción por instrucción tendremos que ir a mirar el bit TF dentro del EEFLAG. En este caso concreto el bit del TF tiene como valor 0. Este valor 0 significa que el programa se ejecutara todo seguido.

9) ¿ Qué característica especial tiene la tarea que se está ejecutando?

La tarea que se esta ejecutando tiene como característica especial que es una tarea anidada. Para poder saber si es anidada al igual que en los anteriores apartados habrá que mirar el EEFLAG, concretamente el bit NF cuyo valor para este caso será 1. Una tarea anidada se caracteriza por volver a la tarea inmediatamente anterior de donde se la llamo después de finalizar esta.

10) Sin contar la instrucción “rep movs” ¿cuál de las otras es la que más duración ocupa y por qué?

La instrucción mas larga es *mov fs, dx*. Fs y dx son dos registros de 16 bits que están en la CPU. FS es un registro del segmento de datos. Por lo cual automáticamente va a la tabla de descriptores y de ahí los mete a la memoria cache del registro asociado y lo carga en él. A partir de ahí actualizara lo que haya: base, limite y atributos.

11) Indicar como calcula el procesador la dirección de la Memoria Principal donde se encuentra la última instrucción del programa de arriba

Por un lado tenemos la base del segmento (los 14 bits de mas peso del valor del registro del segmento CS) al cual sumaremos el desplazamiento que contiene EIP con lo cual obtendremos la dirección donde esta la siguiente instrucción a ejecutar.

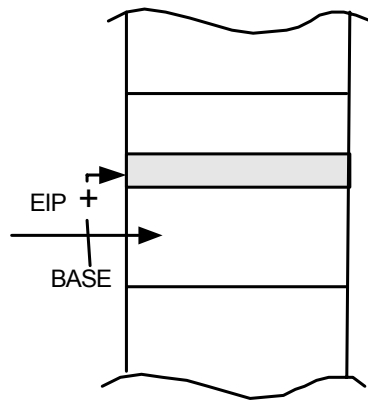


Figura7.3.

12) Si se conoce que el CPI medio de este programa es 3,42 ¿ cuánto tiempo tarda en ejecutarse?

Sabiendo que el CPI medio es igual a 3,42 , el CPI son los ciclos por instrucción de media que tarda en hacerse una instrucción. Por lo tanto partiendo de este valor:

$$T_{\text{programa}} = N^{\circ} \text{ Inst} * \text{CPI} * T_{\text{CLK}} = 7 * 3,42 * 0,5 = 11,97$$

$$T_{\text{clk}} = \frac{1}{f_{\text{clk}}} = \frac{1}{2 \text{ Ghz}} = 0,5 * 10^{-6}$$

13) ¿ Cuántos MIPS nativos tiene rendimiento este procesador para este programa?

Los MIPS nativos son los millones de instrucciones por segundo que realiza, por lo que:

$$T_{\text{inst}} = \text{CPI} * T_{\text{clk}} = 3,42 * 0,5 = 1,7$$

$$\text{MIPS nativos} = \frac{1 \text{ sg}}{T_{\text{inst}} * 10^6} = \frac{10^9}{1,7 * 10^6} = 588,23$$