INEL 4206. EXAMEN 1. 10 DE MARZO DE 2011
Soluciones

Notación: Para efectos de este examen, nos referimos a un registro y su contenido con la forma "*Nombre de registro = Contenido de Registro*"; por ejemplo SP = 0506h, R5=0x32AF. La referencia a memoria como "[*Dirección*] = *Contenido de memoria*"; por ejemplo [0xF2FE] = 0A234h.

Problema 1. Para un cierto microcontrolador, la sección RAM en la memoria de datos inicia con la dirección 0x0300. La capacidad RAM es de 512 Bytes.

a. Especifique el rango de direcciones cubierta por el RAM.

b. En el lenguaje ensamblador ("assembly language"), el stack se define iniciando el "Stack Pointer" (SP) mediante una instrucción del tipo **mov #___, SP**, donde el espacio faltante corresponde a un número que se guardará en SP. El programador decide inicializar SP para que apunte a una dirección fuera del RAM de tal forma que el stack propriamente hablando quede en la sección "alta" de la memoria RAM. Complete la instrucción con un número adecuado.

c. Justifique su contestación anterior o refute el razonamiento del programador.

Solución:

a. Las direcciones del RAM son, sucesivamente 0300h, 0301h, 0302h, ... hasta cubrir las 512 posiciones. El problema consiste entonces encontrar cuál es el mayor número que se suma a 0300h. Como $512 = 2^9$, este número en binario es 1 1111 1111B (nueve bits), es decir 1FFh (=511), por lo que la dirección última del RAM es 0300h + 1FFh = 04FFh. **El rango de direcciones es 0300h − 04FFh**.

b. Cuando se inicializa el SP, no ha ocurrido ninguna operación de "push". Al ocurrir la primera, el SP se disminuye en dos unidades y debe de estar apuntando a la palabra que ocupa los dos bytes con las mayores direcciones dentro del RAM, 04FEh y 04FFh. Por lo tanto se ha inicializado en 04FEh + 2h = 0500h, fuera del RAM. La instruccón es entonces **mov #0500h, SP**.

c. La justificación se ha dado en la respuesta del inciso anterior.

Problema 2. Se está ejecutando un programa en lenguaje de ensamblaje ("assembly language"). Se tiene SP = 0340h y, como parte del programa, se ejecutaron las instrucciones **push R5** y **push R6** en momentos en que R5 = 3A2Fh y R6= 1234h, respectivamente. Posteriormente se invoca ("fetch") la instrucción **call SRUT1**. Justo después de decodificar ("decode") esta instrucción se tiene PC = 0xE2F4. Ilustre mediante un dibujo cómo se aprecia el stack inmediatamente después de que se ejecuta el retorno de la subrutina SRUT1, suponiendo que ésta ha sido correctamente escrita. Su respuesta debe incluir las direcciones y contenidos de memoria, así como los valores de SP y PC.

Solución: Para ilustrar el proceso tenemos las siguientes etapas:

SP=0340h.
Memory

[0340h]=XXXX ⇐ SP
[033Eh]=XXXX
[033Ch]=XXXX
[033Ah]=XXXX
[0338h]=XXXX

Paso 0: Antes del primer "push"

SP=033Eh.
Memory

[0340h]=XXXX
[033Eh]=3A2F ⇐ SP
[033Ch]=XXXX
[033Ah]=XXXX
[0338h]=XXXX

Paso 1: después de ""**push R5**"

SP=033Ch
Memory

[0340h]=XXXX
[033Eh]=3A2F
[033Ch]=1234h ⇐ SP
[033Ah]=XXXX
[0338h]=XXXX

Paso 2: después de ""**push R6**"

SP=033Ah,
PC=XXXX
Memory

[0340h]=XXXX
[033Eh]=3A2F
[033Ch]=1234h
[033Ah]=E2F4h ⇐ SP
[0338h]=XXXX

Paso 3: después de "**call SRUT1**". PC tiene ahora la dirección de SRUT1.

SP=033Ch, PC=E2F4
Memory

[0340h]=XXXX
[033Eh]=3A2F
[033Ch]=1234h ⇐ SP
[033Ah]=E2F4h
[0338h]=XXXX

Respuesta: después del retorno. PC tiene ahora la dirección anterior al "call".

El paso 3 es la situación en el stack justo después que se ejecuta la invocación a la sub rutina,"**call SRUT1**". Si la programación es correcta, esta debe ser la misma posición que tenga SP justo antes de retornar. La ejecución del retorno cargará en el PC el dato señalado por SP, en este caso el mismo contenido que tenía PC justo antes de la invocación. Después de cargar el PC, SP es aumentado en 2. De esa manera, llegamos a la respuesta.

Problema 3. Para cada una de las instrucciones que se muestran en la columna izquierda, la situación antes de la ejecución de la misma, en registros y memoria, es como sigue:

R6 = 0358h	[0624h]=4254h	[035Ch]=0001h
R7 = 02F4h	[0622h]=36FFh	[035Ah]=00AAh
SP = 0622h	[0620h]=0ABCh	[0358h]=44FFh
PC = 0E464h	[061Eh]=2FEEh	[0356h]=9A8Ch

A la derecha de cada instrucción escriba los resultados de la misma después de ser ejecutadas. Para simplificar, escriba *solamente* el o los registros o los resultados en memoria que cambiaron. Incluya todos los cambios habidos. En caso de no tener suficiente información para indicar el nuevo contenido, escriba XXXX y justifique su respuesta. Las instrucciones con dos operandos suponen la fuente ("source") primero.

a. **mov R6, R7**
b. **mov R7, 35Ah**
c. **push R6**
d. **mov @R6, R6**
e. **pop R7**
f. **mov #356h, 0356h**
g. **call SUB1**
h. **mov R7, 4(R6)**
i. **mov 4(R6), R7**
j. **ret**

Solución: Los resultados se muestran a continuación:

a. **mov R6, R7** ⇒ R7=0358h
b. **mov R7, 35Ah** ⇒ [035A]=02F4h
c. **push R6** ⇒ SP=0620h, [0620h]=0358h
d. **mov @R6, R6** ⇒ R6=44FFh
e. **pop R7** ⇒ SP=0624h, R7=36FFh
f. **mov #356h, 0356h** ⇒ [0356h]=0356h
g. **call SUB1** ⇒ SP=0620h, [0620h]=E464h, PC=XXXX (no conocemos la dirección de la primera instrucción de SUB1)
h. **mov R7, 4(R6)** ⇒ [035Ch]=02F4h (pues 0358h+4h=035Ch)
i. **mov 4(R6), R7** ⇒ R7= 0001h
j. **ret** ⇒ PC=36FFh, SP=0624h

Problema 4. Un microcontrolador hipotético tiene un bus de direcciones ("address bus") con un ancho de seis bits. Los seis bits del bus se denominan, del más al menos significativo, **A5, A4, A3, A2, A1** y **A0**. El bus de control ("control bus") contiene una señal **MEM** que se activa (=1) cuando la instrucción requiere trabajar con memoria o con puerto I/O. En la figura P. 4 se muestra un periferal que contiene cuatro registros PR0, PR1, PR2 y PR3, junto con un decodificador ("decoder") interno usado para seleccionar el registro que se activa. Para cada registro existe una única combinación de valores de los bits del'bus de direcciones ("address bus") que lo activa cuando MEM=1. Esta combinación recibe el nombre de *dirección del registro* ("register address").

a. Establezca la dirección de cada uno de los cuatro registros del periferal en notación hexadecimal.

b. Escriba la instrucción necesaria para copiar en el registro R4 del CPU el contenido del registro PR1 del periferal

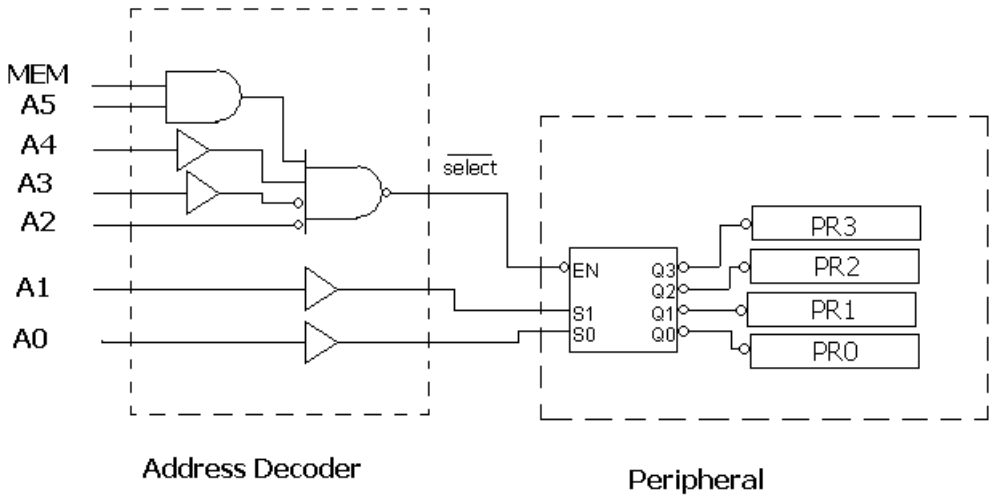

Figure P. 4: Address decoding.

Solución **(a)** Para que un registro se active, deben suceder dos cosas: 1) que el enable se active (cuando la salida del NAND sea 0) y 2) que el decodificador escoja al registro con la combinación adecuada de A1 y A0 (00 para PR0, 01 para PR1, 10 para PR2 y 11 para PR3).

La salida del NAND es $\overline{MEM \cdot A5 \cdot A4 \cdot \overline{A3} \cdot \overline{A2}}$, y es 0 solamente para MEM=1, A5=A4=1, A3=A2=0. Como la dirección de un registro corresponde a la combinación A5-A4-A3-A2-A1-A0 que lo activa al registro, se tiene:

PR0: 110000B=30h PR1: 110001B=31h PR2: 110010B=32h PR3: 110011B=33h.

(b) mov 31h,R4 o bien **mov &31h, R4**

Problema 5. Se tiene un dispositivo I/O con cinco registros de un byte cada uno. En el mapa de memoria se encuentran varios espacios sin utilizar. El primer espacio disponible comienza en la localización o dirección 0100h y termina en 0103h. El segundo espacio disponible comienza en 0200h y termina en 02FFh. El tercer espacio comienza 0450h y termina en 0457h. El último espacio disponible comienza en 070Ah y termina en 070Fh. El I/O se activa mediante una señal de selección.

a. Seleccione un espacio disponible y justifique su decisión.

b. Especifique cuáles líneas del bus de direcciones ("address bus") utilizaría para activar el I/O y cuáles para seleccionar los registros.

c. Tomando en cuenta las dos respuestas anteriores, ¿quedan direcciones inutilizables por lo que hizo? (por inutilizables se quiere decir que no podrán ser asignadas a otro dispositivo o celda) Si la respuesta es afirmativa, indique cuáles y porqué. Si la respuesta es negativa, justifique la misma.

Solución: Este problema tiene al menos dos soluciones posibles. Se muestra una de ellas y la otra se deja como ejercicio al lector. El problema está relacionado al anterior. Se puede decir que el problema 4 es de análisis mientras que este problema es su contraparte.

El dispositivo I/O tiene una seal "select"'que debe activarse mediante una combinación de bits del bus de direcciones ("address bus"), y unos selectores para escoger el registro correspondiente. Con esto en mente, procedemos a resolver el problema

a. Se necesitan al menos cinco direcciones, lo que elimina al espacio comenzando en 0100h, que tiene cuatro. Los otros espacioes tienen 256 direcciones, ocho y seis, respectivamente. Sin embargo, se van a utilizar tres bits del bus de direcciones, para un total de 8 combinaciones con esos tres bits. Eso significa que en realidad, necesitamos ocho direcciones libres, lo que elimina al espacio comenzando en 070Ah. Los otros dos espacios son buenos para usarlos. Empleamos el que empieza en 0450h, que provee el mínimo requerido para cubrir al puerto.

b. Para seleccionar los registros, usamos A2, A1 y A0. El resto de los bits del bus de direcciones los usamos para activar el I/O.

c. Como los tres bits A2, A1, A0 permiten ocho alternativas, y solamente se tienen cinco registros, hay tres alternativas que no llevan a ningún lado pero que tampoco pueden usarse. *Dependiendo* del diseño que se haya hecho del hardware, serán las direcciones inutilizables. Por ejemplo, si las combinaciones 000, 001, 010, 011 y 100 son válidas para los cinco registros, entonces las direcciones 0450h, 0451h, 0452h, 0453h y 0454h han sido utilizadas mientras que 0455h, 0456h y 0457h quedan inutilizables. (El total de combinaciones posibles es 56.)

Nota: Si se considera el rango de direcciones 0200h - 02FFh, la respuesta c) debe especificar en qué dirección se inicia el I/O.

Bono 1. En el problema 2, la sub rutina SRUT1 tiene un error de programación ("bug") que consiste en lo siguiente: se ejecuta la instrucción **pop R4** justo antes del retorno, sin antes haber programado un push, y sin modificar manualmente (es decir, mediante alguna instrucción) el registro SP.

a. Ilustre mediante un dibujo cómo se aprecia el stack inmediatamente después de que se ejecuta el retorno de la subrutina SRUT1. Su respuesta debe incluir las direcciones y contenidos de memoria, así como los valores de SP y PC.

b. Explique porqué se dice que se ha cometido un error de programación.

Solución: Recordemos que al retorno de una sub rutina, PC debe cargarse con el mismo contenido que tenía antes de empezar la misma. Por ello tenemos:

a. La operación de pop provoca que el resultado del problema Prob:probAnterior se convierta en la situación justo **antes** del retorno. Al retorno, lo que va a PC es lo que ahora está siendo señalado por SP, lo que provoca que tenemos finalmente ⇒

SP=033Eh h,
PC=1234h,
R4=E2F4h

Memory

[0340h]=XXXX
[033Eh]=3A2F ⇐ SP
[033Eh]=1234h
[033Ah]=E2F4h
[0338h]=XXXX

b. Es un error porque PC no queda con el contenido que tenía antes de llamar a la sub rutina, y por lo tanto ha cambiado la instrucción a llamar en el siguiente ciclo.

Bono 2. Supongamos que un microcontrolador hipotético no permite usar el PC como operando en ninguna instrucción pero sí permite usar el registro SP. Tampoco tiene saltos incondicionales (jmp). Por razones que no vienen al caso, un programador necesita que inmediatamente después de ejecutar una sub-rutina SRUT1 (invocada con **call SRUT1**), la siguiente instrucción a ejecutarse sea una que se encuentra cuatro direcciones de memoria, direcciones de byte, después de la que normalmente se ejecutaría.

El programador decide entonces que una solución a este problema puede es insertar una instrucción al final de SRUT1 tipo **add**. ¿Qué instrucción puede ser y por qué resuelve el problema?

Solución: "**add #4, 0(SP)**" o "**add #4, @SP**" resuelve el problema por la razón siguiente: Al retornar, el contenido de memoria señalada por SP se carga a PC, convirtiéndose así en la dirección de la instrucción a ejecutarse después de regresar de la sub rutina. La dirección original había sido guardada justamente en este lugar de memoria al invocar la sub rutina. Por ello, sumando cuatro a este contenido, se resuelve el problema.