DEPARTAMENTO DE TECNOLOGÍA ELECTRÓNICA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Iniciación al microcontrolador ATMEGA328P

1. Introducción y objetivos

- Presentar el entorno de programación y depuración de microcontroladores de ATMEL llamado Atmel-Studio.

- Simular y depurar algunos programas muy simples que implican movimiento de datos entre registros o entre registros y memoria, escritos para el microcontrolador ATMEGA328P.

- Atmel-Studio puede descargarse gratuitamente desde las páginas de Microchip en: https://www.microchip.com/mplab/avr-support/avr-and-sam-downloads-archive

2. Estudio experimental

Bloque I:

1) Siga los pasos mostrados en la "guía de uso del programa" para crear un nuevo proyecto. Elija como carpeta de trabajo la carpeta **practicas** y asigne como nombre del proyecto **pract_inic**.

Continúe con los pasos mostrados en la guía de programa. Sustituya el contenido del archivo **main.asm** que se ha abierto en la ventana principal por el siguiente programa.

```
ldi r16,25
ldi r17,1
mov r0,r16
add r0,r16
add r0,r17
subi r16,1
ldi r30,0x00
ldi r31,0x01
st z,r0
fin: rjmp fin
```

2) Realice el ensamblado del programa como se indica en el apartado *"Ensamblado y emulación de un programa"* del manual y compruebe en la ventana inferior que no hay errores. Si los hubiera repase la escritura y corríjalos. Una vez que el ensamblado sea correcto, procederemos a la simulación del mismo como se indica en el mismo apartado del manual.

En el manual se indican las diversas formas de ejecución del programa (algunas de ellas también están disponibles en iconos de la barra de herramientas) y las diferentes ventanas del entorno donde encontrar información relevante sobre la que se le pregunta en el punto siguiente y que debe manejar con soltura.

3) Debe comprobar si:

a) en la memoria de código aparece el programa en código máquina,

b) inicialmente todos los registros contienen el valor 0,

c) al ejecutar paso a paso el programa los registros modifican adecuadamente su valor en consonancia con lo programado (recuerde que la representación de datos en el entorno se realiza en base 16),

d) el valor de la pareja de registros R30 y R31 aparece en dos sitios: junto con el resto de los registros y también en la parte superior de la ventana como puntero Z (lo mismo sucede para los registros R26:29 y los punteros X e Y),

e) el valor del contador de programa se va incrementando al ejecutar instrucciones,

f) la memoria de datos se modifica adecuadamente tras realizar la instrucción *st z,r0*,

g) al incorporar al programa otras dos instrucciones (mostradas a continuación), se modifican los bits del registro de estado SREG (los valores sombreados indican valor lógico 1), explique los resultados obtenidos,

```
ldi r18,0xff
add r18,r17
```

h) al incorporar al programa dos instrucciones más se modifican nuevamente los bits del registro de estado SREG, explique los resultados obtenidos.

```
ldi r18,0x7f
add r18,r17
```

4) Haga otros cambios en el programa y compruebe su efecto.

Bloque II:

En este bloque incorporaremos un nuevo programa al proyecto. Para ello, siga los pasos indicados en la sección *"Añadir más ficheros al proyecto"* del manual. Con el comado *Add Existing Item*, añada el programa *"intercambia.asm"*.

En este programa, inicializamos dos tablas en la memoria de datos y posteriormente las intercambiamos entre sí. El programa es similar al realizado en la práctica del CS2 pero utilizamos el modo de direccionamiento con postincremento. También utilizamos la instrucción de salto condicional para realizar un bucle en lugar de repetir las instrucciones.

1) Realice el ensamblado del programa como se indica en el manual (*"Ensamblado y emulación de un programa"*). Tenga en cuenta que al haber ahora dos programas cargados debe indicar cuál de ellos quiere simular. Para ello colóquese sobre la ventana *Solution Explorer* y picando sobre el fichero con el botón derecho seleccione la opción *"Set as entry file"*. Esto hace que aparezca una flecha roja en el icono a la izquierda del nombre. Una vez que el ensamblado sea correcto, procederemos a la simulación paso a paso del mismo.

2) Debe comprobar que:

a) en la memoria de código aparece el programa en código máquina,

b) en la memoria de datos se inicializan los valores en consonancia con lo programado (recuerde que la representación de datos en el entorno se realiza en base 16),

c) los valores de las tablas se intercambian durante la ejecución del programa,

d) los punteros Y y Z se autoincrementan durante la ejecución de las instrucciones que los usan para recorrer las tablas,

e) el contador de programa va repitiendo una serie de valores cada vez que se ejecuta el bucle,

f) el registro de estado SREG permanece inalterado hasta el momento de salir del bucle (cuando el registro r16 se hace 0) que es indicado poniendo el banderín Z activo (aparecerá sombreado).

Bloque III:

En este bloque incorporaremos un nuevo programa al proyecto. Para ello proceda de nuevo como en el bloque II y cargue el fichero *calculacubo.asm* que se le ha suministrado. En este programa calculamos el cubo de un dato almacenado en un registro (R21) utilizando dos subrutinas anidadas (subrutina *cubo* y subrutina *mult*)

1) Realice el ensamblado del programa como se indica en el manual. Tenga en cuenta que al haber ahora tres programas cargados debe indicar cuál de ellos quiere simular. Para ello colóquese sobre la ventana *Solution Explorer* y picando sobre el fichero seleccione la opción *"Set as entry file"*. Una vez que el ensamblado sea correcto, procederemos a la simulación paso a paso del mismo.

2) Debe comprobar que:

a) el puntero de pila modifica su valor decrementándose cada vez que se salta a subrutina e incrementándose en los retornos de subrutina,

b) en la memoria de datos se almacenan los valores del contador de programa (ya incrementados) cuando se salta a subrutina,

c) el contador de programa toma los valores de las direcciones de inicio de la subrutina al ejecutar la instrucción CALL y recupera su valor (incrementado) al ejecutar RET,

d) el valor que se ha elegido para ilustrar el programa es 3, pruebe otros valores y determine a partir de qué valor se produce desbordamiento.

3) Modifique el programa de modo que en lugar de realizar el cubo de un único elemento recorra una tabla de 4 elementos (que usted debe inicializar en la memoria de datos) y obtenga otra tabla con los 4 elementos elevados al cubo.