



Universidad
de Huelva

*Departamento de Ingeniería Electrónica, de Sistemas
Informáticos y Automática*

INFORMÁTICA INDUSTRIAL II

PRÁCTICAS DE LABORATORIO

CURSO 2001-2002

PRÁCTICAS de INFORMÁTICA INDUSTRIAL II

ASISTENCIA

Las prácticas tendrán lugar en el Laboratorio de Arquitectura y Redes (Sótano, pasillo de la izquierda del edificio V.R.C) siendo obligatoria la asistencia, permitiéndose la ausencia justificada hasta a un 20 % de las mismas, las cuales serán recuperadas con posterioridad.

Se irán entregando los enunciados de las prácticas a realizar a medida que se vayan terminando. El alumno deberá realizarlas a lo largo de las sesiones de prácticas. Una vez realizadas serán evaluadas por el profesor de prácticas, que resolverá cualquier duda presentada en la práctica. Se aconseja traer la práctica preparada de casa, como mínimo su diseño.

EVALUACIÓN

a) Las prácticas han de ser aptas para aprobar la asignatura.

Las prácticas de laboratorio se calificarán aptas, y muy bien en función de:

- Nivel técnico y calidad de redacción de la memoria de prácticas.
- Revisión del trabajo de practicas funcionando por el profesor de prácticas.

b) Memoria de prácticas

Se entregará una memoria impresa de las prácticas realizadas, junto con un disco que contendrá los ficheros fuente de los diseños o programas si ha lugar. En la memoria debe aparecer claramente el nombre del alumno. Cada memoria de practicas debe estar redactada mediante procesador de textos o equivalente y contendrá como mínimo los siguientes apartados:

- 1.- Portada donde aparezca claramente nombre, curso y fecha.
- 2.- Índice del documento.
- 3.- Un epígrafe para cada práctica, donde aparezca:
 - Enunciado.
 - Diseño del circuito con esquemas.
 - Comentarios sobre el montaje y/o problemas presentados y como se han resuelto.
 - Respuesta a las cuestiones planteadas en el enunciado.
 - Posibilidades de ampliación/mejora si ha lugar.
- 4.- Anexos donde aparezcan fotocopias y/o documentación técnica si se considera necesario. Bibliografía que se ha empleado.

Todos los programas y diseños han de aparecer comentados, tanto los del disco como los de la memoria. Se valorará la claridad en este aspecto. Por imperativos legales la memoria no se devolverá al alumno.

MATERIAL USADO

Para la realización de las prácticas se hará uso del siguiente material:

1. Manuales específicos de circuitos integrados.
2. Equipo de desarrollo ALTAIR 537.
3. Entorno de desarrollo KEIL.
4. Monitor de depuración PAULMON2.

Manuales específicos de circuitos integrados:

Se trata de unos libros técnicos editados por los fabricantes, en los que podemos encontrar las características de los distintos circuitos integrados, tales como su función, su patillaje, su circuitería interna, su tabla de funcionamiento, sus características eléctricas y térmicas, etc.

Para la consulta en dichos manuales se dispone de varios índices:

- *Un índice alfanumérico*, donde se pueden encontrar los chips ordenados alfabéticamente según su nomenclatura y acompañados de la sección y página en la que se encuentran.
- *Un índice funcional*, en el cual los circuitos se ordenan según la función que realizan.

Para buscar un circuito integrado en un manual, la forma más sencilla consiste en buscar en el índice funcional hasta encontrar el chip adecuado a nuestras necesidades, averiguando su nombre. Una vez hecho esto nos iremos al índice alfanumérico, donde hallaremos la página en que se encuentra.

Equipo de desarrollo ALTAIR 537:

Es una placa equipada con el procesador 80C537 de Infineon, que no es más que un equivalente al 8051 con más periféricos. En el anexo se ofrecen los esquemas del circuito y toda la información auxiliar.

Entorno de desarrollo KEIL:

Integra un editor, ensamblador, compilador de C y simulador del 8051. Lo empleamos para el desarrollo de nuestros programas. En el anexo se acompaña un manual. Existen varias versiones, aunque todas ellas se basan en el mismo juego de directivas de compilación, solo cambia la facilidad de uso.

Monitor de depuración PAULMON2:

Consiste en un pequeño programa que se ejecuta en la placa ALTAIR 537 y nos permite descargar los programas realizados con el ensamblador o compilador en la placa y probar y depurar nuestros programas. Se acompaña un pequeño manual en el anexo.

PRÁCTICA Nº 1

ENTORNO DE DESARROLLO

OBJETIVOS

Introducir el manejo práctico en laboratorio del entorno de edición, ensamblado y depuración de programas KEIL. Introducir el uso del monitor de depuración PAULMON2.

CONCEPTOS BÁSICOS

Documentación teórica: Listado de instrucciones del procesador 8051.

MATERIAL

PC con entorno KEIL
Placa de desarrollo ALTAIR 537

PROCEDIMIENTO

PARTE1 - Simulador

La primera práctica de laboratorio está diseñada para familiarizarnos con el lenguaje ensamblador, herramientas de software empleadas para escribir programas en ensamblador del 8051. Comenzaremos la práctica usando el software para introducir, ensamblar y simular un corto programa, que se da más abajo. Luego tendremos que escribir un corto programa y demostrar su funcionamiento empleando el simulador.

El siguiente trozo de código se ha diseñado para emplearlo en un dispositivo electrónico de consumo, en el que el usuario puede editar parámetros usando un juego de pulsadores y displays de siete segmentos colocados en el panel frontal. Cuando se desconecta la alimentación del sistema, la circuitería externa y una rutina de interrupción (que no necesitamos conocer) pasa el procesador a un modo de bajo consumo (power-down), donde la RAM interna se mantiene mediante una batería. Sin embargo, la batería puede agotarse.

Esta rutina es llamada inmediatamente después que se hayan cambiado cualquier parámetro y cuando se restaura la alimentación. Si la memoria se ha corrompido, la rutina pone a uno el bit de acarreo, para indicar que hay que inicializar los parámetros a valores por defecto, evitando que el dispositivo funcione con parámetros aleatorios.

Hay 15 parámetros de usuario, cuyos valores se almacenan en las direcciones de memoria interna 30H a 3EH. Un byte de checksum se almacena en la dirección 3FH. La rutina primero calcula el nuevo checksum. Luego se compara con el antiguo valor almacenado en la posición 3FH. Si son iguales, se pone a cero el bit de acarreo, en otro caso se pone a uno. Finalmente, el valor en la posición 3FH se reemplaza con el nuevo checksum.

Esta rutina puede servir tanto para verificar la integridad de los datos del usuario como para actualizar el byte de checksum inmediatamente después de que los datos se modifican por el usuario. Cuando se emplea al comienzo, el valor devuelto por el bit de carry es importante, pero cuando se emplea para actualizar el checksum en memoria, el bit de acarreo no se emplea.

```
comp:      PUSH  ACC           ;salva el acumulador
           MOV   A, R0         ;y también R0
           PUSH  ACC
           CLR   A             ; pone ACC a cero
           MOV   R0, #30h      ; inicializa R0 con 30H
comp1:     ADD   A, @R0         ; suma cada byte
           INC   RO            ; incrementa el puntero
           CJNE  R0, #3Fh, comp2 ; bucle si no hemos acabado
           MOV   R0, A         ; guarda ACC en R0 por ahora
           CLR   C             ; Carry=0 para resta
           SUBB  A, 3Fh        ; resta antiguo checksum
           MOV   3Fh, R0       ; almacena nuevo checksum
           CLR   C
           JZ    comp3         ; chequeo fue ok si ACC=0
           SETB  C             ; carry=1 si fallo chequeo
comp3:     POP   ACC           ; restaura R0 y ACC
           MOV   R0, A         ; Carry y 3Fh es lo único
           POP   ACC           ; que se modifica
           RET                ; y eso es todo!
```

Introducir el código anterior empleando el editor del entorno KEIL, habiendo creado antes un proyecto. Ensamblar el código y finalmente emplear el simulador incluido en el entorno (d-Scope) para verificar que el código funciona.

PARTE2 – Placa de desarrollo

Desarrollar un programa en ensamblador que saque un mensaje por pantalla empleando las rutinas proporcionadas por el monitor PAULMON2.

Descargar el programa a la placa y probar su correcto funcionamiento.

PRÁCTICA N° 2

ARITMÉTICA MULTIBYTE

OBJETIVOS

Simulación de programas aritméticos.

CONCEPTOS BÁSICOS

Documentación teórica: Listado de instrucciones del procesador 8051.

MATERIAL

PC con entorno KEIL
Placa de desarrollo ALTAIR 537

PROCEDIMIENTO

El 8051 solo puede realizar aritmética de 8 bits, sin embargo realizando múltiples operaciones de 8 bits, se pueden manejar cantidades mayores. Recientemente un grupo de diseño ha desarrollado software empleando enteros sin signo de 16 bits. Ellos necesitan que escribas una rutina para calcular el producto de dos números de 16 bits (sin signo) para usarla en su programa.

El programa colocará dos cantidades de 16 bits en las posiciones de memoria 20h y 21h, así como direcciones 22h y 23h donde el lugar inferior de memoria almacena el byte menos significativo. La rutina debe colocar el resultado, que será por supuesto un valor de 32 bits, en las direcciones de memoria 24h a 27h, como indica la tabla siguiente:

Dirección de Memoria	Valor
20h	Valor de entrada, Byte Menos Significativo
21h	Valor de entrada, Byte Más Significativo
22h	Valor de entrada, Byte Menos Significativo
23h	Valor de entrada, Byte Más Significativo
24h	Valor de salida, Byte Menos Significativo
25h	Valor de salida
26h	Valor de salida
27h	Valor de salida, Byte Más Significativo

El código puede usar cualquier registro y direcciones de memoria que necesite. Recordar que una rutina de multiplicación completa no es necesaria, solo hay que hacer cuadrado de un valor simple de 16 bits. La rutina no necesita optimizarse en velocidad, pero es crucial que el valor resultado del producto final devuelto por la rutina sea el cuadrado correcto de 32 bits del valor de entrada.

Algoritmo sugerido: Consideremos la tarea de multiplicar dos números ordinarios, como por ejemplo 2382 y 5347. Simplemente introduciéndolos en la calculadora obtenemos un producto de 12736554. Si la calculadora solo pudiera aceptar dos dígitos como entrada, el algoritmo sería el mismo.

Los números deben considerarse para realizar la operación con solo dos dígitos a la vez:

$$2382 \times 5347 = ((23 \times 100) + 82) \times ((53 \times 100) + 47) = ?$$

Factorizando, se obtiene la siguiente expresión.

$$(23 \times 53 \times 100 \times 100) + (23 \times 47 \times 100) + (82 \times 53 \times 100) + (82 \times 47) = ?$$

Ahora el problema consiste en cuatro multiplicaciones que se pueden realizar, seguidas por una gran suma.

$$(1219 \times 10000) + (1081 \times 100) + (4346 \times 100) + 3854 = 12,736,554$$

Aunque existen otras técnicas para multiplicar números, este algoritmo está bien indicado para multiplicar enteros sin signo de 16 bits en el 8051, puesto que emplea multiplicación de 8 bits que devuelve un valor de 16 bits. Recomendando que implementes un algoritmo similar a este.

De nuevo, introducir el código en el editor de texto, ensamblarlo y simularlo. Crear una corta lista de números (algunos mayores de 255, por favor) y ejecutar el programa con cada uno de ellos para verificar que calcula de forma precisa el cuadrado de cada número. Asegurarse que el programa funciona a vuestra satisfacción, demostradlo al profesor de prácticas para que tome nota del grupo.