Microelectrónica, 4º Curso, P94

Implementación de un microcontrolador sencillo

M. L. López Vallejo, P. Ituero y C. López Barrio

Abril 2009

Índice

1. Introducción

La gran variedad de materias que se cubren en este curso tiene como objetivo proporcionarle todo lo que necesita para diseñar y llevar a la práctica un circuito integrado CMOS. La razón del proyecto final es, justamente, la de ayudarle a profundizar en estas materias y darle un mejor conocimiento de cómo es el proceso de diseño.

Su trabajo consistirá en diseñar un modesto sistema CMOS hasta el punto de obtener un plano de base detallado. Partiendo de las especificaciones que se le entregan, y trabajando en grupos de dos, debe lleva a cabo el diseño y entregar una memoria de gran calidad. Procure finalizar el diseño, puesto que un diseño a medias será calificado pero y es menos instructivo que uno completo, a la vez que resultará mucho más difícil de escribir la memoria final.

En este proyecto se va a realizar la implementación de un microcontrolador PIC simplificado. De hecho, las simplificaciones van a ser tantas que lo vamos a denominar MINI-PIC. El resto de este documento está dedicado a la descripción del microcontrolador objetivo del proyecto y sus principales componentes.

La calificación del proyecto estará basada fundamentalmente en la calidad de su diseño - su simplicidad, corrección y grado de finalización-. Conforme a esto, debe buscar como meta principal un diseño correcto, con un trazado *(layout)* simple y bien estructurado. Las consideraciones de tamaño, consumo y velocidad son secundarias.

Lea atentamente las instrucciones detalladas que se dan a continuación.

2. Objetivos y requisitos del proyecto

El objetivo del proyecto es diseñar un circuito integrado CMOS funcionalmente equivalente a un pequeño PIC de la familia PIC16F83 de Microchip. Para ello debe adaptar las especificaciones a nuestro estilo de diseño:

1. Utilice CMOS de cualquiera de los tipos estudiados (dinámico, estático, dominó, etc.).
2. Elija una estrategia de temporización. Si es con reloj de dos fases siga la disciplina de temporización (con extensiones) que se estudia en clase.
3. Suponga, si lo desea, que las señales de reloj (*clk*, o $\Phi 1$ y $\Phi 2$) le vienen dados desde fuera de la pastilla. Por contra, también puede partir de una única fase externa y generar internamente las dos fases.
4. Plantee, si es posible, una estrategia de reducción de consumo.
5. Añada elementos que faciliten el test del circuito.

Puede realizar las suposiciones que considere necesarias, señalándolas y justificándolas convenientemente, así como ignorar detalles tales como asignación de las patillas de salida, tiempos de conmutación, etc. Sus metas son un diseño regular, correcto, que pueda ser comprobado (test) y con mínimo cableado.

El proyecto se ralizará en parejas utilizando la herramienta de diseño *full-custom* MicroWind, presentada en clase. Esta es una herramienta especialmente diseñada para el entorno académico, por lo que el trabajo que se haga con ella se verá facilitado en gran medida.

Para consultar cualquier duda que surja respecto al proyecto debe ponerse en contacto con Marisa López Vallejo (e-mail: marisa@die.upm.es, despacho C-230) o Pablo Ituero Herrero (correo electrónico: pituero@die.upm.es, despacho C-226). Siempre se solicitará cita previa vía correo electrónico.

Fechas importantes

Fecha recomendada de presentación del anteproyecto: 8 de Mayo de 2009.

Fecha recomendada de presentación del trabajo: (para que pueda ser incluída la nota en la convocatoria de junio) 29 de Junio de 2009.

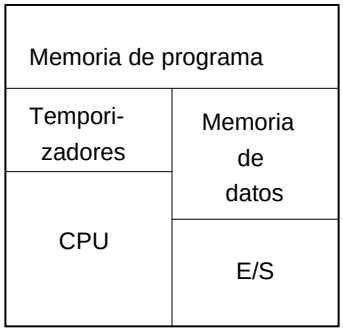

Figura 1: Elementos principales en la arquitectura de un PIC.

3. Especificaciones del diseño

3.1. Arquitectura del PIC

Los PIC son una serie de microcontroladores muy sencillos fabricados por Microchip[1]. Como todos los microcontroladores es un microprocesador de baja complejidad con facilidades de entrada/salida (periféricos incorporados en el chip). La arquitectura del PIC presenta dos peculiaridades claras:

- El uso de un *conjunto reducido de instrucciones*. Es decir, es una máquina RISC, con un pequeño conjunto de instrucciones sencillas (en nuestro caso son 35), lo mínimo necesario para hacer un programa que funcione. Las instrucciones de este tipo de máquinas se suelen ejecutar en un solo ciclo de reloj.
- La utilización de una *arquitectura Harvard*. Este tipo de arquitecturas se caracteriza por diponer de memorias separadas para datos y para programa. La separación es total, por lo que no se comparte el bus de acceso a las mismas. De esta forma se puede acceder simultáneamente al programa y a los datos, siendo posible un *pipeline* más claro y sencillo de las instrucciones.

Estos dos aspectos son clave para entender el buen funcionamiento de un dispositivo tan simple. En la figura 1 se pueden ver las cinco partes principales integrantes de un PIC:

- CPU
- Memoria de programa
- Memoria de datos
- Temporizadores
- Dispositivos de Entrada/Salida

Dependiendo de qué microcontrolador se trate en particular, se tienen diferentes configuraciones de tamaño de las memorias, puertos de entrada y salida, etc. Nosotros nos centraremos en la familia PIC16F83, cuyas características básicas son:

[1] `www.microchip.com`

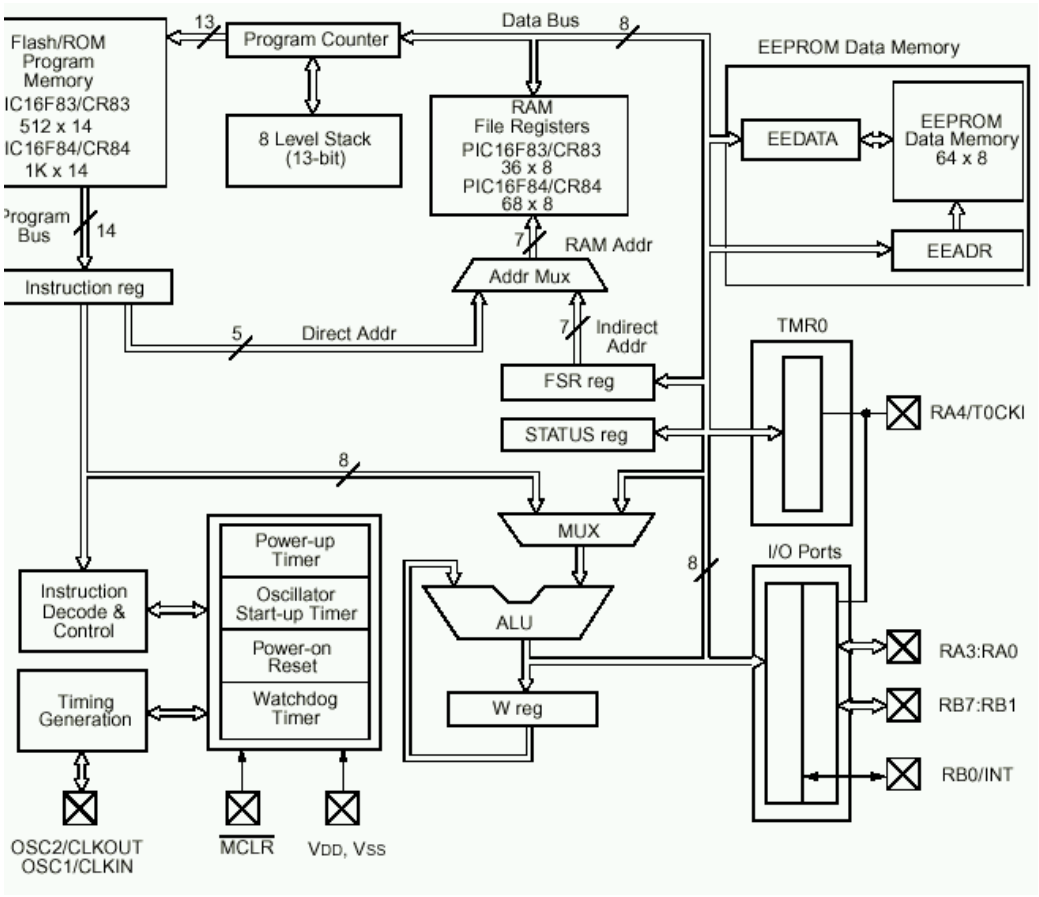

Figura 2: Detalle de la arquitectura de un PIC.

- Las instrucciones son de 14 bits
- El datapath es de 8 bits
- La memoria de programa disponible en el chip es 1 Kword ($1K \times 14bits$)
- La memoria de datos es de 128 + 4 bytes

Otras características de un PIC de la serie en que nos vamos a centrar nosotros se detallan en lo sucesivo.

En la figura 2 se puede ver un diagrama de bloques de la arquitectura de los PIC16F83. Observando este diagrama vemos cómo el bus de datos es diferente del de programa, como corresponde a una arquitectura Harvard. Siendo este microcontrolador de 8 bits, el bus de datos es de este tamaño, pero el de programa es de 14, que coincide como es lógico con el tamaño de instrucción del PIC. Otras familias tienen instrucciones de diferente tamaño, pero siguen considerándose microcontroladores de 8 bits, al ser ésta la unidad básica utilizada en el procesamiento de la CPU.

En las siguientes secciones se trata con algo más de detalle los principales elementos del PIC.

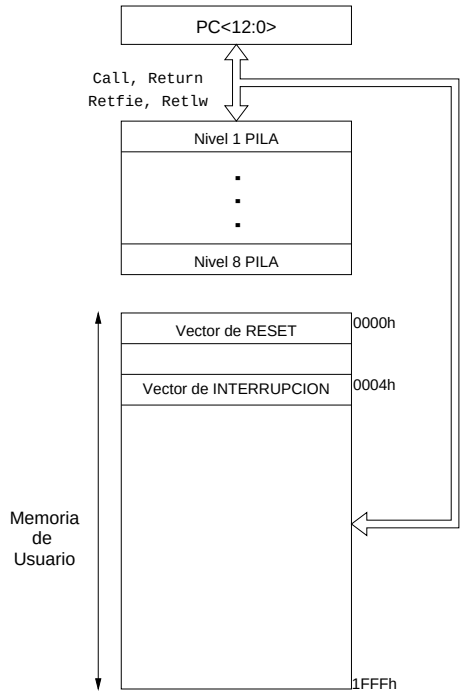

Figura 3: Mapa de memoria de programa y pila

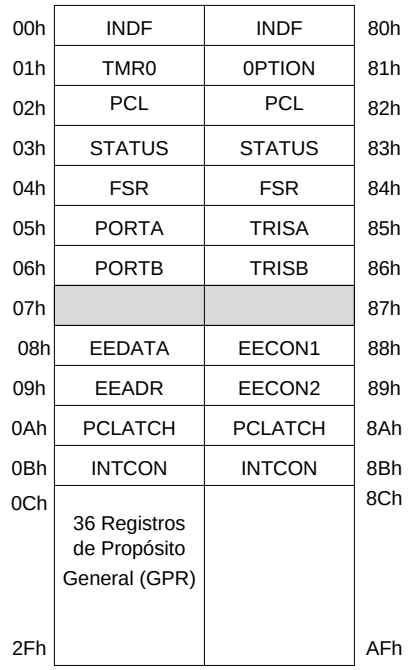

Figura 4: Mapa de memoria de datos

3.1.1. Organización de la memoria

En los PIC hay dos bloques de memoria, una de programa y otra de datos.

Memoria de programa Incluye el *contador de programa* (**PC**), registro de 13 bits que permite acceder a $8K \times 14bits$ posiciones del espacio de la memoria de programa. En la figura 3 se puede ver la relación entre estos elementos para un PIC16F83. El vector de RESET está en la posición `0000h`, mientras que el vector de interrupción está en la posición `0004h`.

La pila del PIC es muy sencilla. Son 8 registros de 13 bits (tamaño del bus de direcciones de la memoria de programa) que almacenan direcciones de la memoria de programa para volver de llamadas a subrutinas o de una interrupción. Al ser sólo 8 registros, podemos controlar cuál es la siguiente posición libre simplemente con 3 bits (del registro TOS, que se verá posteriormente).

Como se puede ver, ésta no es una pila en el sentido amplio, sino que está muy limitada. El programador debe tener en cuenta que hay un límite al número de subrutinas e interrupciones que puede anidar. El funcionamiento básico de la pila es el siguiente. Cuando se produce una llamada a subrutina (instrucción `call`) el contenido del PC se debe llevar a la pila (push) y la dirección de la subrutina se lleva al PC. Si se termina la ejecución de la subrutina (instrucción `return`) se debe llevar el contenido de la pila al PC (pop). La instrucción `retlw` hace lo mismo que `return k` pero lleva antes el literal k al registro W.

En resumen, cuando se realiza una operación *push* en la pila se debe hacer:

```
Stack[TOS] <- PC
TOS <- TOS + 1
```

```
PC(12:11) <- PCLATH, PC(10:0) <- k
```

El funcionamiento del contador de programa se detalla en la siguiente sección. Cuando se realiza una operación *pop* en la pila se debe hacer:

```
TOS <- TOS - 1
PC <- Stack[TOS]
```

Memoria de datos El bloque de datos se descompone a su vez en RAM de propósito general, *Registros de propósito general (General Purpose Register, GPR)* y en *Registros con función especial (Special Function Register, SFR)*, todos de 8 bits. Esta memoria (GPR y SFR) está distribuida en dos bancos, por lo que es necesario controlar a qué banco se accede mediante bits de control. Estos bits están en el registro de estado (status<7..5>). En la figura 4 se pueden ver qué posiciones ocupan los registros especiales en el fichero de registros (Register File, RF en lo sucesivo), así como la estructura que sigue esta memoria.

Toda la memoria de datos se puede recorrer utilizando la dirección absoluta de cada registro del RF, o se puede direccionar de modo indirecto utilizando el registro FSR (*File Select Register*). Los registros más importantes para el funcionamiento del microcontrolador serán explicados con mayor detalle en la próxima sección.

3.1.2. Registros

Pasemos ahora a ver con un poco de detalle los registros principales de la CPU del PIC.

Registro de trabajo (Reg W) La primera característica que llama la atención es que el PIC tiene sólo un *registro de trabajo (working register, reg W)*, que interviene en casi todas las instrucciones aritméticas y tiene un comportamiento similar al de un acumulador. Este registro es de 8 bits y no se puede direccionar. Para su implementación se debe tener en cuenta que este registro es de lectura y escritura, y que se debe cargar de forma condicional. Es decir, dependiendo de la instrucción en particular que se esté ejecutando será necesario cargar un nuevo valor en W o no.

Registro de instrucción (Reg INST) Es un registro de 14 bits que no tiene ninguna particularidad. Como es lógico se puede escribir desde el bus de programa y se puede leer. La lectura la realiza el decodificador de instrucción, que se encarga de activar las señales necesarias para ejecutar la instrucción que se está decodificando.

Registro Contador de Programa (Reg PC) Es un registro de 13 bits que se puede leer y escribir. Hay que tener en cuenta que este registro se debe cargar de forma controlada desde diferentes fuentes, por lo que necesita una señal de control de carga (`load`). La selección de las fuentes de datos para el PC se puede hacer con un multiplexor sencillo, teniendo muy claro de dónde puede venir la información. Por ejemplo, el PC se puede cargar desde la memoria de programa o desde la pila. Pero también puede simplemente incrementarse para pasar a la siguiente instrucción.

La implementación real de este registro en el PIC es un tanto sofisticada. El byte bajo del PC es el registro PCL, un registro que se puede leer y escribir. El "byte" alto PC<12:8> no se

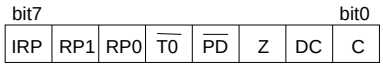

Figura 5: Registro STATUS de un PIC.

puede leer o escribir directamente, sino que procede del registro PCLATH. El contenido de este último registro se transfiere al byte alto del PC cuando el PC se carga con un nuevo valor (en las instrucciones `call, goto` o al escribir en PCL). En las notas de aplicación de Microchip se puede ver este comportamiento en detalle.

Registro de Estado (Reg STATUS) El registro de Estado contiene los bits que indican el estado de la ALU, del RESET y además los bits de control de los bancos de memoria de datos. En la figura 5 se pueden ver los 8 bits que componen este registro. Su significado es el siguiente (de más significativo a menos):

IRP: Bit de selección de banco de registros. Se utiliza para direccionamiento indirecto. Si está a 0 se seleccionan los bancos 0 y 1 (00h-FFh). Si está a 1 se seleccionan los bancos 2 y 3 (100h-1FFh). NO se utiliza en los PIC 16F8X, por lo que no lo utilizaremos (se mantiene siempre a 0).

RP1:RP0 Bits de selección de registros, se utilizan en direccionamiento directo de la siguiente forma:

- 00 → banco 0 (00h-7Fh)
- 01 → banco 1 (80h-FFh)
- 10 → banco 2 (100h-17Fh)
- 11 → banco 3 (180h-1FFh)

En los PIC 16F8X sólo se utiliza RP0, por lo que nosotros ignoraremos el otro bit manteniéndolo a 0.

$\overline{TO}$: Bit de *time-out*. Avisa poniédose a 0 que se ha producido un final de cuenta en el temporizador WDT. Se pone a 1 al arrancar o tras ejecutar las instrucciones `clrwdt` o `sleep`.

$\overline{PD}$: Bit de *power-down*. Avisa poniédose a 0 que se ha detenido el funcionamiento del microcontrolador. Esto sucede tras la instrucción `sleep`. Se pone a 1 al arrancar o tras ejecutar la instrucción `clrwdt`.

Z: Bit de Cero. Se pone a 1 cuando se ha producido un resultado de 0 en la ALU, y a 0 si el resultado no es 0.

DC: Acarreo de dígito (*digit-carry*). Refleja el acarreo si se produce en medio byte (*nibble*) poniéndose a 1. Este acarreo se produce del bit 3 al 4 en un byte. Si no hay acarreo es 0.

C: Acarreo. Refleja el acarreo que se produce en el bit más significativo del byte poniéndose a 1. Si no hay acarreo es 0.

Evidentemente, este registro puede ser leído para conocer el valor de estos flags. También puede ser escrito, aunque de una forma un tanto peculiar. Los bits Z, DC y C no se pueden modificar. Lo mismo sucede con los bits $\overline{TO}$ y $\overline{PD}$. El programador debe modificar los bits de este registro mediante las instrucciones `bcf, bsf, swapf` y `movwf`, que no afectan al registro Status en su ejecución.

Registro de interrupciones (Reg INTCON) Registro de 8 bits de lectura y escritura que habilita o deshabilita las diferentes fuentes de interrupción que hay en el microcontrolador.

Registros de direccionamiento indirecto (Regs INDF y FSR) El registro INDF no existe físicamente. Cuando se accede a este registro en realidad se accede al registro al que apunta FSR (*File Selection Register*), realizando de este modo el direccionamiento indirecto.

3.1.3. Puertos de Entrada/Salida

El PIC16F83 tiene dos puertos de entrada/salida, PORTA y PORTB. PORTA es de 5 bits de entrada/salida cuya dirección se configura mediante el registro TRISA. En este registro se debe poner el bit correspondiente a 1 para configurarlo como entrada y a 0 para que sea una salida. El pin RA4 lo utiliza también el temporizador TMR0 para la entrada de reloj (TOCK1). Los tres bits más significativos no se utilizan.

El PORTB es un puerto bidireccional de 8 bits, cuya configuración se realiza de forma naáloga en el registro TRISB. El pin RB0 se puede utilizar como fuente de interrupción externa. Estos puertos pueden tener otras funciones especiales en las que no vamos a entrar en este documento.

3.1.4. Unidad Aritmético-Lógica (ALU)

Siguiendo la figura 2, la ALU es de 8 bits, y puede realizar las siguientes operaciones:

- Suma
- Resta
- Desplazamiento
- Operaciones lógicas

Todas las operaciones aritméticas funcionan en complemento a 2. La ALU es completamente combinacional, por lo que las operaciones se realizan en menos de un ciclo de reloj. Cuando las operaciones son entre dos operandos, típicamente uno de los operandos es el *registro de trabajo (reg W)* y el otro es uno de los registros del *fichero de registros* o una constante cargada en modo inmediato.

Las operaciones de la ALU pueden afectar a los siguientes bits del registro de estado: C, DC y Z.

3.2. Conjunto de instrucciones

Como ya se ha dicho, todas las instrucciones del PIC son de 14 bits. En cada instrucción hay un campo **OPCODE** que indica el tipo de instrucción, y uno o más operandos. El tamaño de este código depende del tipo de operación, variando de un mínimo de 3 bits a un máximo de 6. En la descripción que damos a continuación de las diferentes instrucciones utilizaremos lo

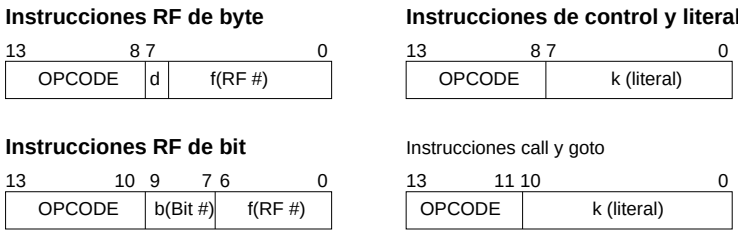

Figura 6: Campos de las instrucciones dependiendo del tipo.

siguiente: `f` se refiere a un registro del Fichero de Registros (RF); `b` será la posición del bit en un byte; `k` es un *literal*, o expresión, y `d` representará la selección de destino (si `d=0` es el registro W y si `d=1` es el registro `f` del RF.).

Hay cuatro tipos básicos de instrucciones. En cada grupo se mantienen en común los dos bits más significativos del código de operación, como a continuación se detalla:

- Relacionadas con el RF a nivel de byte (00): operan con los 8 bits de un byte de la memoria de datos.
- Relacionadas con el RF a nivel de bits (01): operan un bit específico de un byte de la memoria de datos.
- Instrucciones de control de programa (10): especifican una posición de la memoria de programa.
- Instrucciones "literales" (11): el operando aparece explícitamente (como literal) en la instrucción.

Hay algunas instrucciones que no siguen esta regla, como las instrucciones de retorno de subrutina o de *sleep*, que tienen identificador 00.

En la figura 6 se puede ver cómo se organiza la información de la instrucción en campos dependiendo del tipo de instrucción de que se trate. El conjunto total de instrucciones del PIC se puede ver en la tabla 1.

Nemónico, operandos	**Descripción**	**Código operación**	**Reg. Estado**
Operaciones a nivel de byte en el RF			
ADDWF f, d	Suma W y f	00 0111 dfff ffff	C,DC,Z
ANDWF f, d	AND W con f	00 0101 dfff ffff	Z
CLRF f	Pone a 0 f	00 0001 lfff ffff	Z
CLRW -	Pone a 0 W	00 0001 0xxx xxxx	Z
COMF f, d	Complemento de f	00 1001 dfff ffff	Z
DECF f, d	Decrementa f	00 0011 dfff ffff	Z
DECFSZ f, d	Decrementa f, salta si 0*	00 1011 dfff ffff	
INCF f, d	Incrementa f	00 1010 dfff ffff	Z
INCFSZ f, d	Incrementa f, salta si 0*	00 1111 dfff ffff	
IORWF f, d	OR W con f	00 0100 dfff ffff	Z
MOVF f, d	Mueve f	00 1000 dfff ffff	Z
MOVWF f	Mueve W to f	00 0000 lfff ffff	
NOP -	No Operación	00 0000 0xx0 0000	
RLF f, d	Rota Izq. f a través de Carry*	00 1101 dfff ffff	C
RRF f, d	Rota Drcha. f a través de Carry*	00 1100 dfff ffff	C
SUBWF f, d	Resta W de f	00 0010 dfff ffff	C,DC,Z
SWAPF f, d	Intercambia nibbles en f	00 1110 dfff ffff	
XORWF f, d	OR Exclusivo W con f*	00 0110 dfff ffff	Z
Operaciones a nivel de bit en el RF			
BCF f, b	Bit Clear f	01 00bb bfff ffff	
BSF f, b	Bit Set f	01 01bb bfff ffff	
BTFSC f, b	Bit Test f, Skip if Clear*	01 10bb bfff ffff	
BTFSS f, b	Bit Test f, Skip if Set*	01 11bb bfff ffff	
Operaciones de control y literales			
ADDLW k	Suma literal y W	11 111x kkkk kkkk	C,DC,Z
ANDLW k	AND literal y W	11 1001 kkkk kkkk	Z
CALL k	Llama subrutina**	10 0kkk kkkk kkkk	
CLRWDT -	Clear Watchdog Timer*	00 0000 0110 0100	TO,PD
GOTO k	Ir a dirección**	10 1kkk kkkk kkkk	
IORLW k	OR literal con W*	11 1000 kkkk kkkk	Z
MOVLW k	Mueve literal a W	11 00xx kkkk kkkk	
RETFIE -	Retorno de interrupción**	00 0000 0000 1001	
RETLW k	Retorno con literal en W**	11 01xx kkkk kkkk	
RETURN -	Returno de Subrutina**	00 0000 0000 1000	
SLEEP -	Pasa a modo standby	00 0000 0110 0011	TO,PD
SUBLW k	Resta W de un literal	11 110x kkkk kkkk	C,DC,Z
XORLW k	OR Exclusivo de literal con W*	11 1010 kkkk kkkk	Z

Cuadro 1: Conjunto de instrucciones.*No las vamos a implementar. **Instrucciones que requieren 2 ciclos de reloj.

3.3. Nuestro diseño: el MINI-PIC

En esta sección se van a especificar los mínimos que debe cumplir el diseño del MINI-PIC que se hace en el proyecto de la asignatura. Ha de quedar claro desde el principio que el proyecto tiene un margen de libertad bastante grande. Es decir, NO hay que hacer un PIC idéntico a uno de la familia PIC16F83, sino que se deben adaptar las especificaciones del mismo para poder utilizar las técnicas vistas en clase. A continuación se enumeran los puntos que son necesarios para tener un diseño final que funcione:

- Conjunto básico de registros:
  - Contador de programa (PC). No hace falta que sea de 13 bits, una implementación sencilla de un byte (el equivalente al PCL descrito anteriormente es suficiente). En este caso el PCLATH del fichero de registros será un registro más.
  - Registro de instrucción (INST): en este caso sí debe ser de 14 bits.
  - Registro de Estado (STATUS): debe reflejar la funcionalidad básica de interacción con la ALU (bits Z, DC y C) y la selección de banco de registros que sigue el PIC16F83. Como se verá luego, las interrupciones y el temporizador no se deben realizar).
  - Registro de Trabajo (W), cuyo papel en las instrucciones aritméticas y lógicas es fundamental.
- Fichero de registros (RF), con un número de registros suficiente para simular el funcionamiento de un PIC16F83.
- Unidad Aritmético-Lógica (ALU). Debe ser *combinacional*.
- Registros de la pila: si bien no es necesario implementar los 8 registros que tiene el PIC16F83, un número que sirva de nuevo para emular el funcionamiento del microcontrolador.
- El decodificador de instrucciones, que llevará a cabo el control del microcontrolador. Este módulo es el encargado de decodificar la instrucción, y dependiendo de qué se trata, generar las señales necesarias para la correcta ejecución del programa.
- Puertos de Entrada/Salida (PORTA y PORTB) más sus registros de configuración. Funcionamiento básico sin complicar nada.
- Del conjunto de instrucciones sólo aquellas que sean realmente necesarias. Por ejemplo, en la tabla 1 aparecen con un asterisco aquellas que hemos juzgado no necesarias.

Es importante destacar que **NO ES NECESARIO** implementar la siguiente funcionalidad:

- Los temporizadores.
- El funcionamiento del direccionamiento indirecto.
- Los registros que no se han enumerado en la lista anterior (aunque aparezcan en la figura 4).
- Las interrupciones.
- Un PC con 13 bits y siguiendo el funcionamiento descrito de PCLATH y PCL.

En la figura 7 se presenta una arquitectura simplificada del PIC que se puede utilizar como ejemplo de lo que se pretende hacer en este proyecto. Como se puede ver, se han realizado las simplificaciones anteriores y se han cambiado algunos anchos de bits. Por ejemplo, los dos

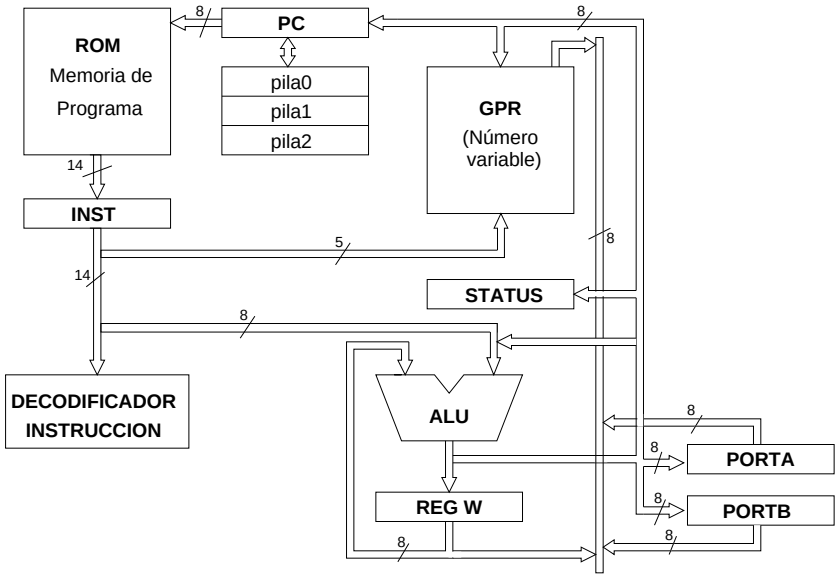

Figura 7: MINI-PIC: arquitectura simplificada propuesta.

puertos de entrada/salida son iguales y con una anchura estándar de 8 bits. Otra simplificación realizada es el desdoblamiento del bus de datos en dos buses, cada uno recogiendo el flujo de información en diferente sentido (hacia el bus/desde el bus), con lo que la implementación final se facilita considerablemente.

4. Plan de trabajo

4.1. Revisión del diseño. Anteproyecto

La primera etapa es una breve memoria que se utilizará para revisar su diseño. Debe convencer con este anteproyecto de la viabilidad del plan previsto para el proyecto. Esta memoria debe contener:

1. Una idea general de su diseño, incluyendo un plano de base general y un plano general de la temporización del sistema (mostrado a dos colores si se utiliza reloj con dos fases).
2. Una breve descripción de cada bloque funcional con esquemáticos que faciliten la comprensión del diseño (puertas o transistores según se considere).
3. Finalmente un plano de conexionado, no necesariamente a escala, pero que incluya cómo se haría el encaminamiento de las principales señales de datos y control, así como el reloj, la alimentación y la masa.

Con excepción de los diagramas, esta memoria **no debe tener más de 4 páginas**.

4.2. Memoria final

La memoria final debe describir el diseño completamente. Es decir, el camino seguido desde las especificaciones hasta el plano de base detallado. Sugerimos el siguiente índice:

1. Introducción (formulación del problema, idea general de la solución).
2. Descripción detallada (algoritmos, diagrama de bloques, esquemáticos, temporización, caracterización de consumo, retardo y área para cada célula).
3. Plano de base (planos de color y de conexionado, dimensiones de los trazados, caracterización de consumo, retardo y área para el sistema completo).
4. Interfaces de entrada/salida: descripción del diseño del anillo de pads, circuito de protección de entrada, buffers de entrada y de salida.
5. Test (plan de prueba, características de *comprobabilidad* en su diseño).
6. Conclusiones.
7. Apéndice A: Resultados (tabla que comprenda la caracterización de cada célula y la caracterización global)
8. Apéndice B: Trazados (trazados de cada célula y trazado global)

Además de este índice, la memoria debe ser **sólo** lo suficientemente larga como para describir lo que Ud. ha hecho. Sea conciso e intente hacer una memoria atractiva de leer.

A continuación se incluye más información sobre lo que se espera de su memoria.

4.2.1. Introducción

Describa el diseño en términos generales. Dado que se le han entregado las especificaciones básicas, no necesita discutirlas con ningún detalle. Sin embargo, sí debe discutir cualquier separación de ellas, añadiduras, o aspectos de ellas que jueguen un papel importante en su solución. Utilice figuras para representar la idea global. Piense que el objetivo es centrar a un lector sobre el objeto y contenido del trabajo que sigue. Recuerde: "una imagen vale más que mil palabras".

4.2.2. Descripción detallada

Después de la idea general, continúe con una descripción más detallada de los algoritmos involucrados y de los diagramas que muestren cómo fluye la información. Discuta la arquitectura que haya elegido, dando un diagrama de bloques físicos para ella, y explique las decisiones tomadas durante el diseño para llegar a dicha arquitectura. Explique la temporización del sistema.

Para cada una de las células que componen el sistema explique cómo ha realizado el diseño, si lo ha hecho partiendo de cero, si ha utilizado el sintetizador de Microwind o si ha reutilizado algún diseño existente. Si el diseño lo ha realizado a mano, incluya un esquemático o un diagrama lógico que explique el funcionamiento. Caracterice el área, el consumo y el retardo. Para ello explique cómo ha calculado el camino crítico de la célula y justifique el procedimiento para la medida del consumo.

Explique cuidadosamente cualquier circuito con alguna particularidad o truco en la temporización que haya decidido utilizar.

4.2.3. Plano de base

Comience con un plano de base aproximado que puede realizar con cualquier programa de dibujo (como el Powerpoint, el Gimp o incluso el Paint), mostrando la comunicación entre los bloques, el propósito de las áreas de cableado, y el encaminamiento de las líneas de alimentación, masa y el reloj. En el apéndice B se incluirá el plano de base completo desarrollado con Microwind, este plano incluirá los pads. Los canales de cableado, si por cualquier eventualidad no están terminados, deben estar dimensionados adecuadamente, considerando tanto las líneas de alimentación como las de datos y control. Indique claramente cómo se empalman las células para formar los bloques incluidos en el plano de base y cómo se conectarían, si es que no ha podido incluir todas las interconexiones.

Encuentre dónde está el camino crítico del sistema y justifique la frecuencia máxima de funcionamiento del mismo. Calcule el consumo de potencia y corriente de su diseño y muestre cómo estos requisitos se traducen en la anchura de las líneas de alimentación. Se valorará positivamente el análisis de variaciones en parámetros principales del circuito, como puede ser la caída en la tensión de alimentación al distribuirla en el chip.

4.2.4. Interfaces de Entrada/Salida

Describa brevemente cuál es la interfaz de entrada/salida de su diseño. Caracterice el anillo de pads (área, capacidad y retardo) y detalle tanto las protecciones necesarias para las señales

de entrada como las estructuras necesarias para la transferencia de datos entre el interior y el exterior del chip.

4.2.5. Test

Discuta cómo comprobaría su circuito. No dé una descripción superdetallada del procedimiento de prueba, sino muestre cómo, bajo suposiciones razonables -por ejemplo, parte de la lógica de control es incorrecta-, proporciona un acceso adecuado al interior del chip. Para la comprobación (test) debe ser posible tanto el control como la observación de su estado; un contador de cuenta libre sin reset NO es aceptable. Describa cualquier hardware especial para test que haya incluido, tal como caminos serie para exploración, circuitos de inicialización, ...

Procure presentar su esquema de test de forma esquemática, de forma que resulte fácil comprender su idea.

4.2.6. Apéndices

El Apéndice A será una hoja resumen con los resultados numéricos más importantes del proyecto. Aquí estarán incluidas todas las caracterizaciones de las células en términos de área, retardo y consumo (y cualquier otra métrica que Ud. considere interesante) junto con la caracterización del sistema completo.

En el Apéndice B incluya el trazado de todas las células individuales y el trazado del sistema global. Si quiere, puede ampliar alguna zona que le haya resultado especialmente compleja y explicar las soluciones que ha adoptado. No es preciso entregar este apéndice impreso.

4.3. Entrega del Proyecto

La memoria se entregará por correo electrónico a los profesores de la asignatura como un único archivo comprimido ".zip" o ".rar" en el que se incluirá la propia memoria en formato ".pdf" junto con todos los trazados, pruebas, esquemas, imágenes y datos que considere relevantes para una mejor comprensión del proyecto.

Con respecto al formato de nombre de los distintos ficheros, se establece que comiencen con `grupo<número de grupo>_` y sigan con la explicación más clara posible del contenido del archivo. Por ejemplo, si el grupo 3 quisiera incluir el trazado de un sumador completo, lo nombraría `grupo3_FullAdder`. El archivo comprimido de la entrega se llamará `grupo<número>.rar` o `grupo<número>.zip` (e.g. `grupo3.zip` o `grupo3.rar`).

4.3.1. Edición de archivos .MSK de Microwind

La edición de los trazados de las entregas individuales, que deberá hacer durante el curso, han de realizarse en el siguiente directorio de su ordenador:

> `C:\microelectronica\<su número de dni>\` + el árbol de directorios que considere necesario
>
> Por ejemplo:

`C:\microelectronica\67859483\entrega2\67859483_ejer2b.MSK`

Cuando se hayan asignado los grupos del proyecto, el directorio de trabajo que emplearán todos los miembros del equipo, será:

`C:\microelectronica\grupo<número>\` + el árbol de directorios que considere necesario

Por ejemplo:

`C:\microelectronica\grupo3\combinacional\sumador\grupo3_FullAdder.MSK`

Esto se debe a que el programa crea en el archivo .MSK una referencia al directorio de trabajo que si no coincide con el lugar donde se encuentra el archivo, se produce un error y el archivo no se abre. Cabe la posibilidad de cambiar esta referencia "a mano" editando el archivo .MSK. Esta opción puede ser muy tediosa cuando se trabaja con decenas de archivos procedentes de las máquinas de cada uno de los componentes de un grupo de trabajo. Por lo tanto todos vamos a trabajar en el mismo directorio, facilitando así la puesta en común del trabajo colectivo y la corrección de los ejercicios por parte del profesorado.

4.4. Críticas

Se le anima a hacer por escrito una crítica del curso, anónima si quiere al margen de la memoria del proyecto. El interés fundamental es conocer la adecuación, tanto a un nivel conceptual como concreto, de la clase (si considera que su opinión no quedó suficientemente reflejada en la encuesta final), así como sugerencias para mejorar esta asignatura.

Si tiene problemas con su compañero/a debe hacerlo saber. No todos los compañeros trabajan. No debe confundirse compañerismo con encubrimiento de los que "pasan" a costa de los demás. Si no existe información en contra, debe suponerse que todo marcha bien y, por tanto, el trabajo será evaluado como el resultado de un equipo de dos personas.