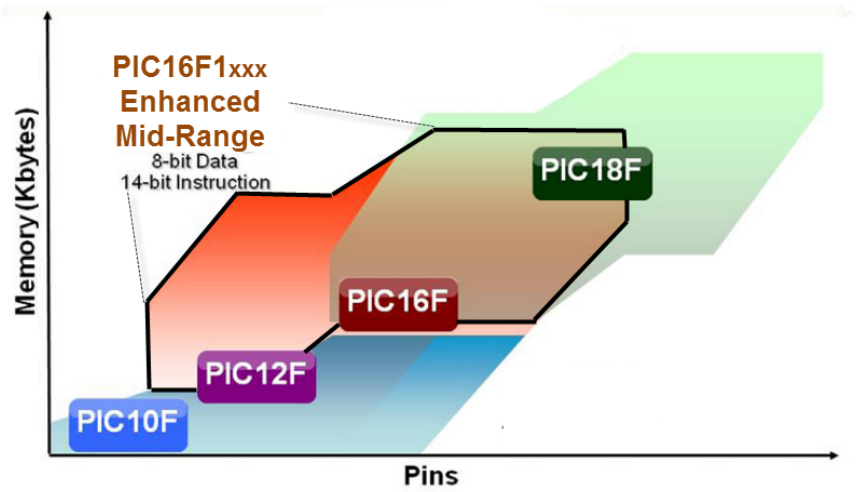
Descripción general de la arquitectura MCU PIC de gama media mejorada

Resumen

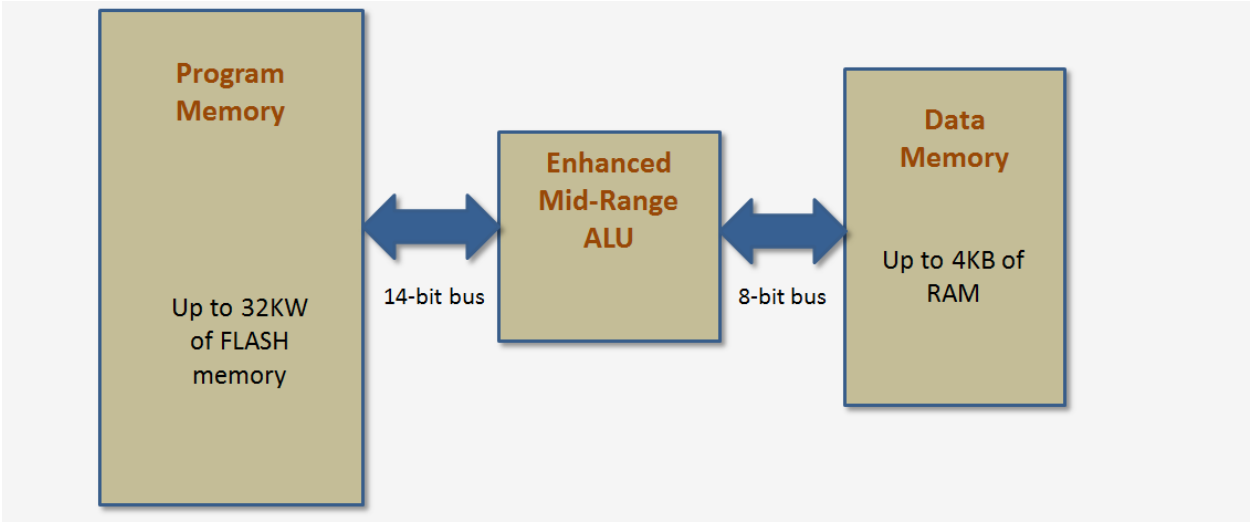
Familias de MCU de 8 bits de Microchip



La familia de MCU PIC® de 8 bits de rango medio mejorado PIC16F1xxx abarca una amplia variedad de tamaños de memoria y pines de E/S.
Esta página presenta las características arquitectónicas clave de la familia de MCU PIC16F1xxx. En esta página se proporcionan enlaces a los detalles técnicos necesarios para implementar aplicaciones en la familia de microcontroladores PIC de gama media mejorada.

Arquitectura de Harvard

Los microcontroladores PIC ® de gama media mejorados utilizan una arquitectura Harvard de doble bus.



autobús de instrucción

Las instrucciones del programa se introducen en la ALU desde la memoria del programa FLASH a través del bus de instrucciones de 14 bits. En cada ciclo de reloj de instrucción, se lee una palabra de programa de 14 bits en la ALU.

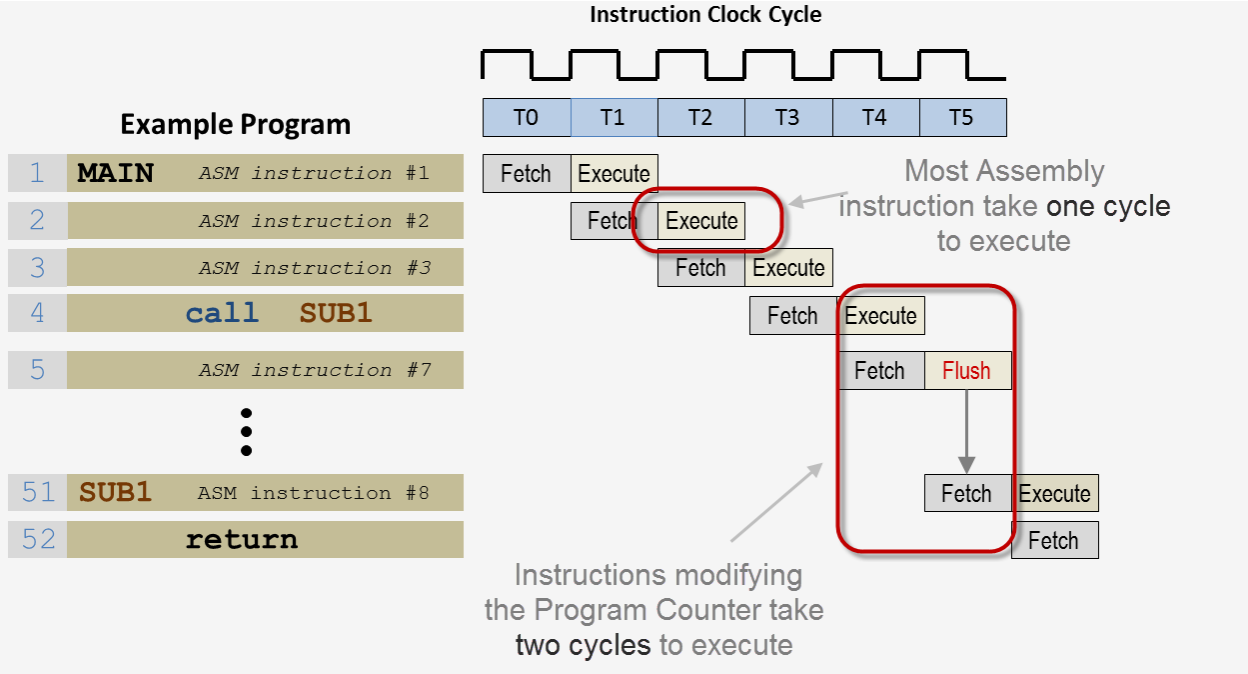
Bus de datos

Un bus de datos de 8 bits conecta la ALU al espacio de memoria de datos. Durante cada instrucción, la ALU puede leer datos desde la ubicación de la memoria de datos, modificar los datos y luego volver a escribir los datos en la memoria.

Canalización de instrucciones

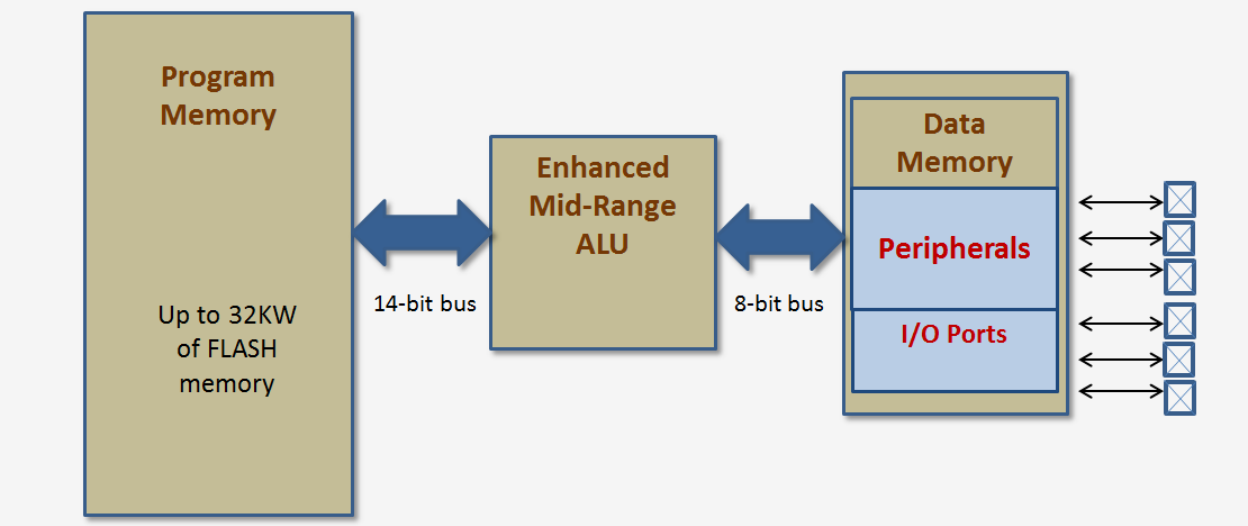
La arquitectura de doble bus del PIC de rango medio mejorado proporciona una línea de instrucción de dos etapas. Una cada ciclo de reloj ejecuta dos fases de instrucción:

1. La siguiente instrucción se **"obtiene"** de la memoria del programa
2. La instrucción actual se **"ejecuta"** y lee/modifica/escribe la memoria de datos (si es necesario)



Periféricos asignados a la memoria

Una mirada más cercana a la sección de memoria de datos de la MCU PIC® de gama media mejorada muestra que se accede a los registros que controlan los periféricos y a los puertos de E/S leyendo o escribiendo en direcciones de memoria de datos específicas. Esta asignación de periféricos a la dirección de la memoria simplifica enormemente el aprendizaje de cómo programar el PIC mejorado de rango medio.



La **página de memoria (/mcu1102:data-memory)** de datos del tutorial mejorado de gama media ofrece una descripción completa junto con ejemplos de programas que acceden a los periféricos asignados a la memoria.

Conjunto de instrucciones ortogonales

Cada MCU PIC ® de gama media mejorada tiene 49 instrucciones. Las instrucciones que acceden directamente a las direcciones de la memoria de datos se ejecutan en un ciclo de instrucción. Las instrucciones que provocan un cambio en el contador del programa BRA, GOTO, RETURN, CALL, ..etc) tardan dos ciclos de instrucción en ejecutarse.

Al mapear los registros de E/S y periféricos a direcciones de memoria, las MCU PIC no necesitan instrucciones especiales para las operaciones de E/S o para establecer registros periféricos. Escribir en un puerto de E/S o configurar un periférico es una simple escritura en una ubicación de memoria. Leer el valor de un pin de entrada, registro de resultado ADC o temporizador es una simple lectura de una ubicación de memoria. Mediante el uso de una pequeña cantidad de instrucciones ortogonales, los MCU PIC de rango medio mejorado son fáciles de programar, usan menos silicio para construir y consumen menos energía.

Ejemplos de implementación de instrucciones

Task	C Language	Assembly
WRITE a value to a variable	var1 = 7 ;	**movlw** 7 **movf** var1
WRITE a value to a Register	T1CON = 0x72;	**movlw** 0x72 **movf** T1CON
WRITE to an Output Latch	LATD = 0xFF ;	**movlw** 0xFF **movf** LATD
READ an input PORT	var1 = PORTB;	**movf** PORTB **movwf** var1
READ a variable's value	var1 = var2;	**movf** var2 **movwf** var1
READ an SFR into another SFR	CCPR1L = ADRESH;	**movf** ADRESH **movwf** CCPR1L

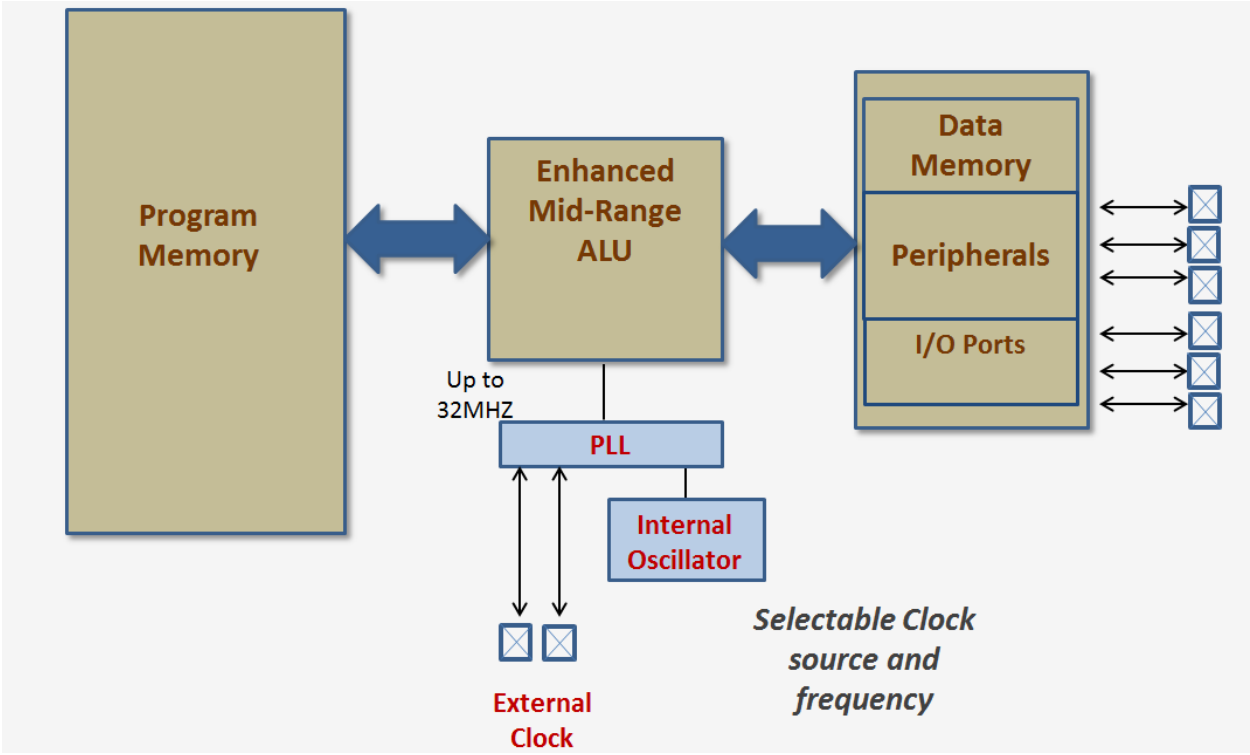
Para obtener una lista detallada del conjunto de instrucciones y una discusión completa del tiempo de instrucción, consulte la página **Conjunto (/mcu1102:instruction-set)** de instrucciones del tutorial de rango medio mejorado.

Opciones de reloj flexibles (hasta 32 MHz)

Seleccionado por los **bits de configuración (/mcu1102:configuration-bits)** del PIC ® MCU, el reloj del sistema tiene las siguientes propiedades: **(/mcu1102:configuration-bits)**

- Fuente opcional (**oscilador interno** o **circuito externo**)

- Opciones de velocidad flexibles **hasta 32MHz**
- **Arranque de dos velocidades** : permite que el sistema ejecute el software de inicialización mientras el oscilador externo se estabiliza
- **Cambio** de reloj: la fuente de reloj del sistema se puede cambiar entre fuentes de reloj externas e internas a través del software.
- **Monitor de reloj** a prueba de fallas: cambia al oscilador interno en caso de falla del reloj externo



Para obtener una descripción detallada de las opciones de configuración del oscilador, consulte la **página del oscilador de 8 bits (/8bit:osc)** en el Tutorial mejorado de rango medio.

E/S digitales

Casi todos los pines de la MCU PIC Enhance de rango medio se pueden usar como pines de entrada o salida digital. Los pines digitales comparten estos atributos:

- Supervisión de entradas digitales
- Controlar dispositivos digitales
- Dominadas débiles internas
- Multiplexado con Periféricos
- Alta capacidad de accionamiento (hasta 25 mA sumidero/fuente en muchos pines de E/S)
- Manipulación directa de bit de ciclo único
- Diodos de protección ESD de 4kV

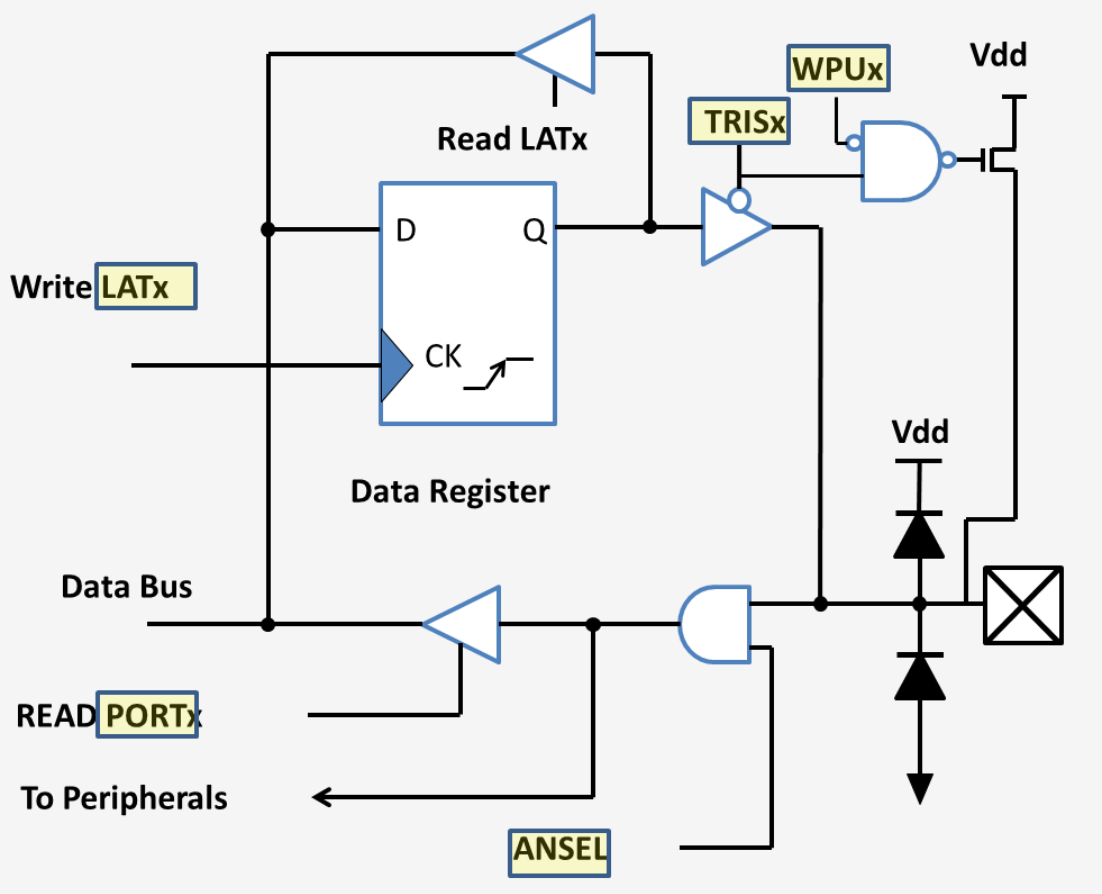
Al reiniciar:

- Los pines digitales vuelven a la entrada (Hi-Z)
- Los pines con capacidad analógica vuelven a ser analógicos

Estructura típica de pines digitales

Cinco registros controlan el funcionamiento de un pin digital. Estos registros de 8 bits controlan 8 pines de un PUERTO. Usando los registros `TRISX, PORTx, LATx, WPUx` y `ANSEL` , el programa puede:

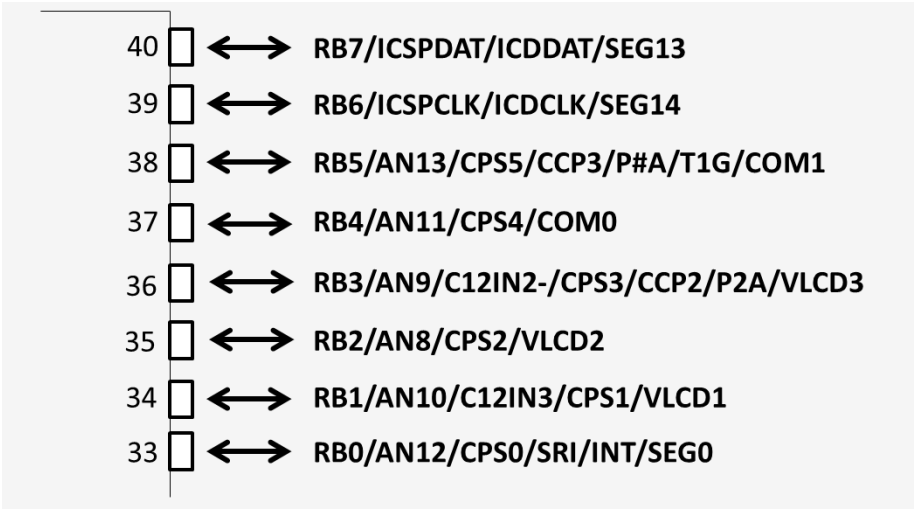
- Configure el pin a como entrada o salida `TRISx`
- Leer un pin de entrada (o los 8 pines PORT) `PORTx`
- Envía un 1 o 0 a un pin `LATx`
- Habilitar o deshabilitar la resistencia pull up interna `WPUx`
- Determine si los pines con capacidad analógica funcionan en modo analógico o digital `ANSEL`



Para una discusión completa de la E/S digital PIC mejorada de rango medio, incluidos los detalles sobre la programación de las operaciones de entrada y salida digital, consulte la sección de **E (/mcu1102:digital-io)** /S digital de este tutorial mejorado de rango medio.

Pines multiplexados

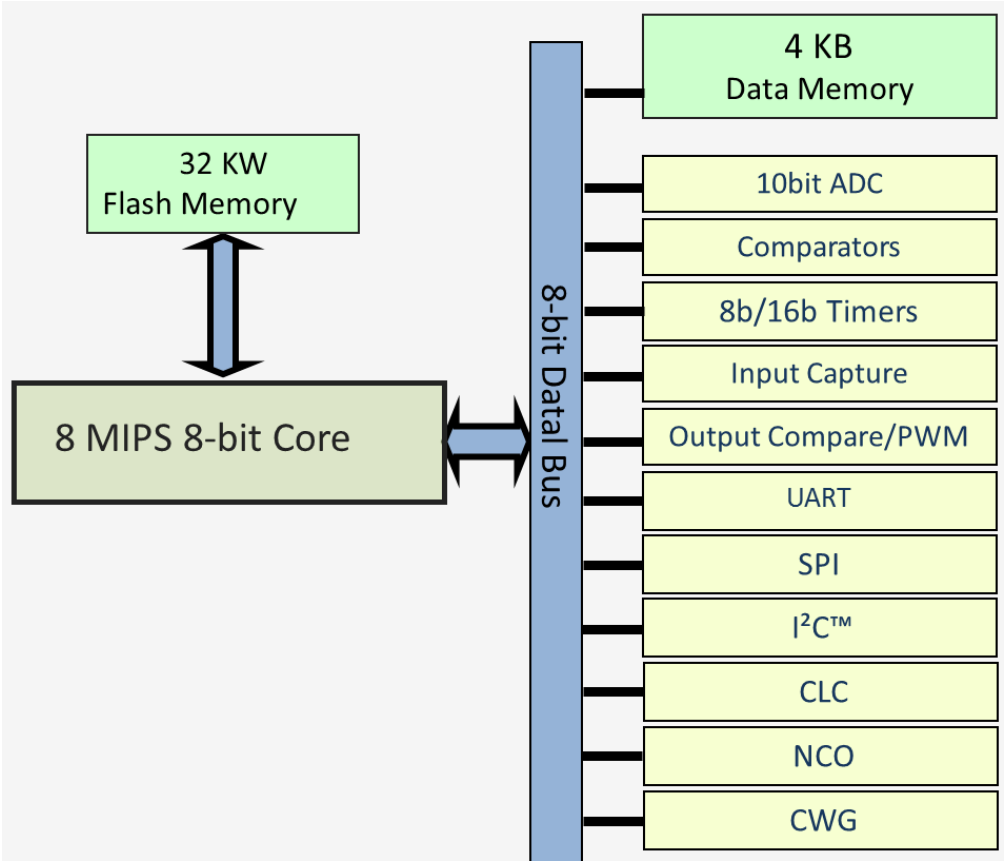
Además de configurarse como E/S digitales, los pines de los MCU PIC mejorados de rango medio pueden tener varias funciones posibles. El diagrama de pines en la hoja de datos muestra las opciones para cada pin. Al inicio, el programa tiene la opción de configurar los pines.



Para una discusión completa de la E/S digital PIC mejorada de rango medio, incluidos los detalles sobre la configuración de los pines, consulte la **sección Periféricos (/mcu1102:peripherals)** del tutorial de rango medio mejorado.

Periféricos avanzados

Además de las E/S digitales mejoradas, los miembros de la familia de MCU PIC de rango medio tienen una variedad de periféricos avanzados. Estos periféricos incluyen periféricos para conversión de datos, comunicación y acondicionamiento de señales.



Para obtener una lista completa de los periféricos disponibles, consulte la **sección Periféricos (/mcu1102:peripherals)** del tutorial .

Interrupciones

Los microcontroladores PIC de rango medio mejorados utilizan una estructura de interrupción preventiva de un solo vector.

Cada periférico en el PIC es capaz de generar una solicitud de interrupción. Cuando ocurre una solicitud de interrupción Y las interrupciones para el dispositivo solicitante están habilitadas, se producirá una interrupción.

El PIC de rango medio mejorado utiliza una pila de hardware de 16 niveles para almacenar el contenido actual de la PC cuando ocurre una interrupción. El contexto del programa se guarda en registros sombra y el control se pasa a la dirección de memoria del programa 0x04.

Rutina de servicio de interrupción (ISR)

El usuario es responsable de escribir el código para dar servicio a la interrupción y colocar el código en la dirección 0x04. Esta Rutina de Servicio de Interrupción (ISR) determina la fuente de la interrupción, luego realiza la tarea necesaria para dar servicio al periférico interrumpido. La instrucción final de un ISR es la instrucción Return From Interrupt (RETFIE).

Guardado automático de contexto

Los siguientes registros se guardan en un registro de sombra de un solo nivel establecido en caso de una interrupción

- registro W
- BSR
- ESTADO
- FSR
- PCLATH

Cuando el ISR ejecuta la instrucción RETFIE, estos registros se restauran al valor anterior a la interrupción.

Prioridad de interrupción de un solo nivel

Cuando ocurre una interrupción, el bit de habilitación de interrupción global (GIE) en el registro de estado está deshabilitado. Esto evitará que una interrupción sea reemplazada por otra interrupción.

Upon executing a RETFIE the state of the GIE control bit is restored to its pre-interrupt value.

For a complete description of the interrupt process and programming examples please refer to the **Interrupt Section (/mcu1102:interrupts)** of the Enhanced Mid-range tutorial.

What happens at System Start-up (RESET)

There are several sources of a RESET on the enhanced mid-range PIC MCU. The RESET sources common to almost all applications are the Power On Reset (POR) and Brown Out Reset (BOR) due to a sagging power supply voltage (i.e. brown-out). There are several other methods for resetting the MCU including Watchdog timeout and directly accessing the MCLR pin.

Program Counter is set to 0x00.

After a RESET the instruction located at address 0 is the first instruction executed. The application developer is responsible for placing code into this address to 'boot up' the PIC. Microchip's MPLAB® XC8 compiler will insert the appropriate instructions to start-up the PIC and transfer control to `main`. Assembly level programmers will have to write the code to initialize the PIC and jump past the Interrupt Vector located at address 0x04.

All Special Function Registers are set to a Default Value

The datasheet for each enhanced mid-range PIC MCU shows the values which the registers will contain at RESET.

Sample Register

R/W-0/u	R/W-0/u	R/W-0/u	R/W-0/u	R/W/HC-0/u	R-x/x	R/W-0/u	R/W-0/u
TMR1GE	T1GPOL	T1GTM	T1GSPM	T1GGO/$\overline{\text{DONE}}$	T1GVAL	T1GSS<1:0>	
bit 7							bit 0

Legend:		
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'
u = Bit is unchanged	x = Bit is unknown	-n/n = Value at POR and BOR/Value at all other Resets
'1' = Bit is set	'0' = Bit is cleared	HC = Bit is cleared by hardware

The **Programming Examples (/mcu1102:programming-examples)** section of the enhanced mid-range tutorial provides programming examples of how to 'boot-up' the MCU.