MICROCHIP

Regional Training Centers

PIC32 & MPLAB C32
周邊基礎應用課程

課程代號：MCU4101

MICROCHIP
Regional Training Centers

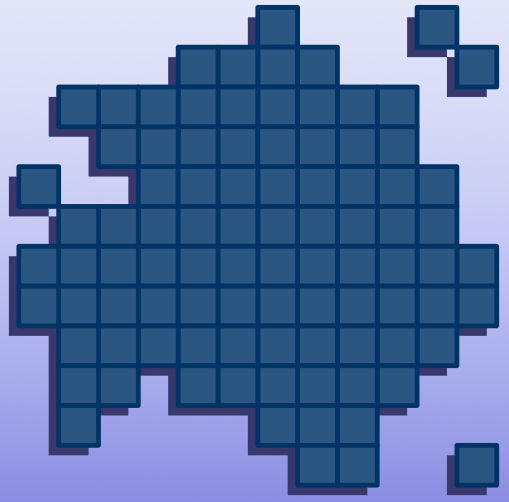
MCU4101 課程內容

- PIC32 32-bits 架構描述
- MPLAB C32 簡介
- PIC32 中斷及控制方式
- 基本週邊的設定與使用
  - PIC32 Configuration 與執行效能的設定
  - 使用 PIC32 的 functions & macros 完成 I/O 的操作
  - PIC32 的中斷操作與 MPLAB C32 對中斷的 support
  - Core Timer 的介紹與使用
  - General Purpose Timer 的介紹與操作
  - PMP Module – 驅動 LCD 模組
  - ADC 的操作練習
  - UART (Optional)
  - I2C (Optional)

MICROCHIP
Regional Training Centers

MCU4101 課程包含的練習

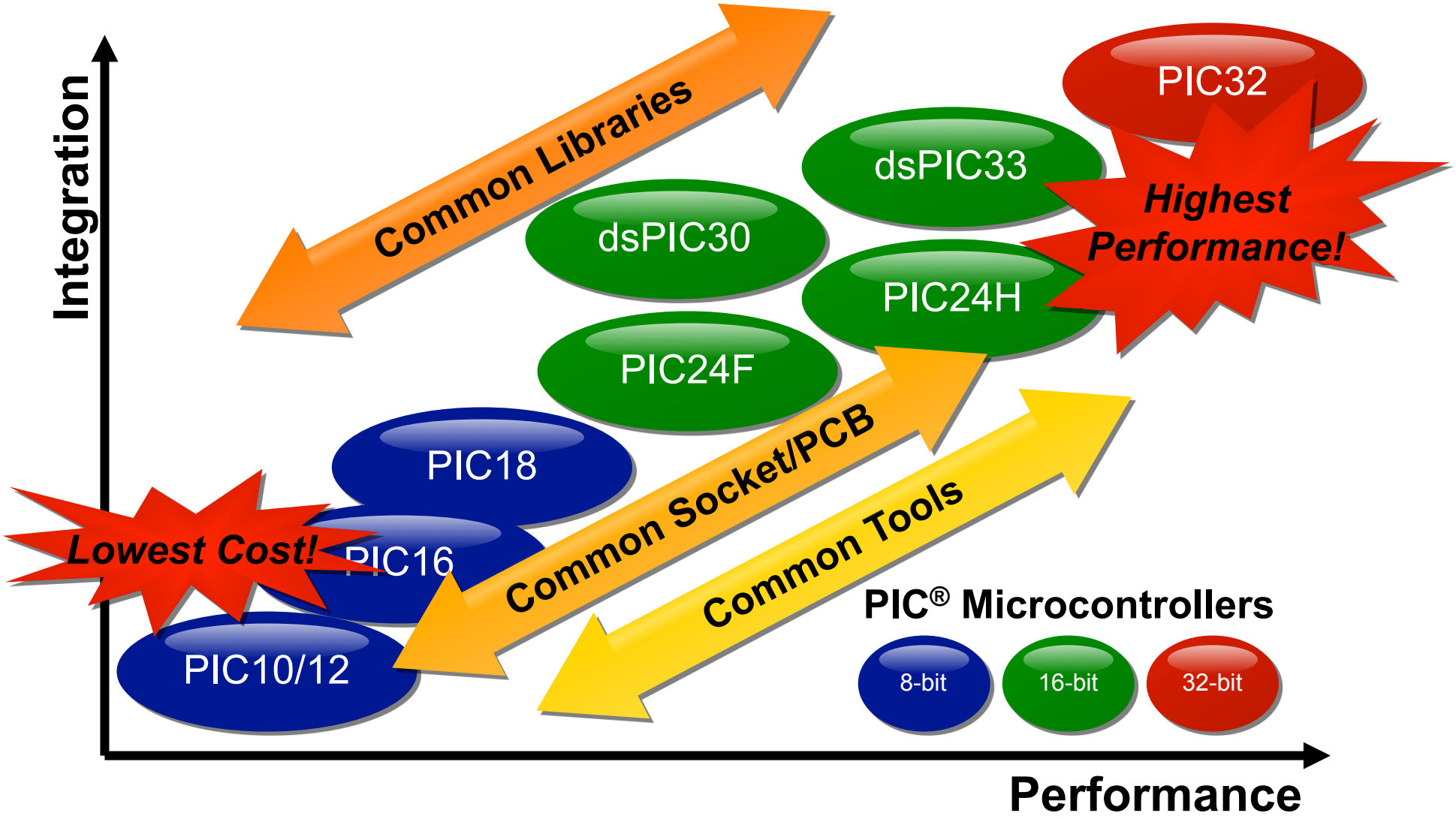
- 練習前的準備: PIC32 的配置設定
- 練習一 : LED 跑馬燈 – 使用軟體 delay … 及如何調整 PIC32 執行的效能
- 練習二 : Using Core Timer Interrupt
- 練習三 : Using Timer1 Interrupt
- 練習四 – Demo : 使用 PMP 設定與操作 LCD 模組
- 練習五 – Demo : ADC 的讀取與顯示 – 單一 Channel
- 練習六 – Demo : ADC 的讀取與顯示 – 多 Channel 掃描
- 練習七 – Demo : ADC 的讀取與顯示 – 使用 Timer 3 觸發轉換



PIC32 Introduction

MICROCHIP
Regional Training Centers

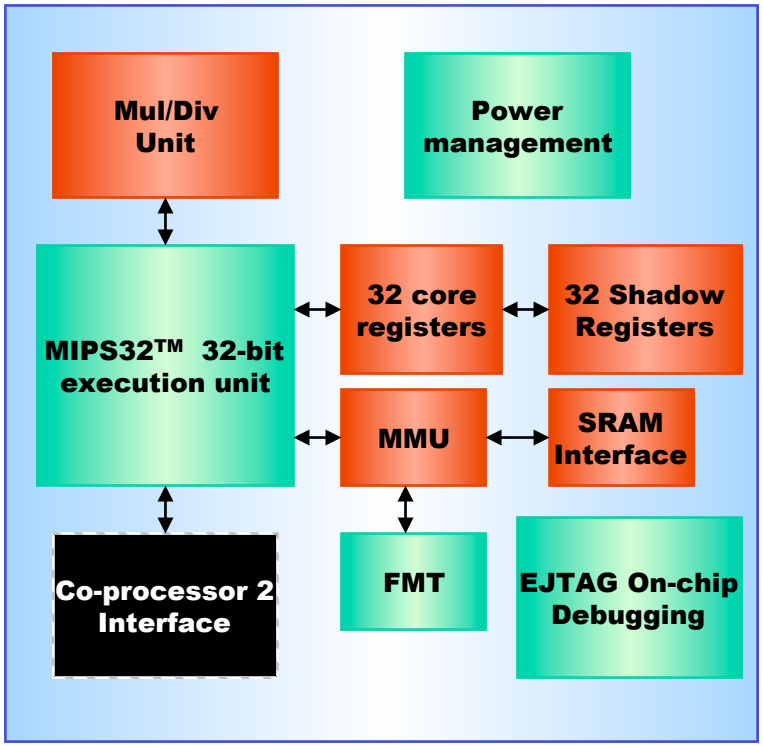
The PIC® Microcontroller Development Platform (Cont'd)



MICROCHIP
Regional Training Centers

MIPS TECHNOLOGIES

MIPS32® M4K® Core



MIPS32 M4K 的特色

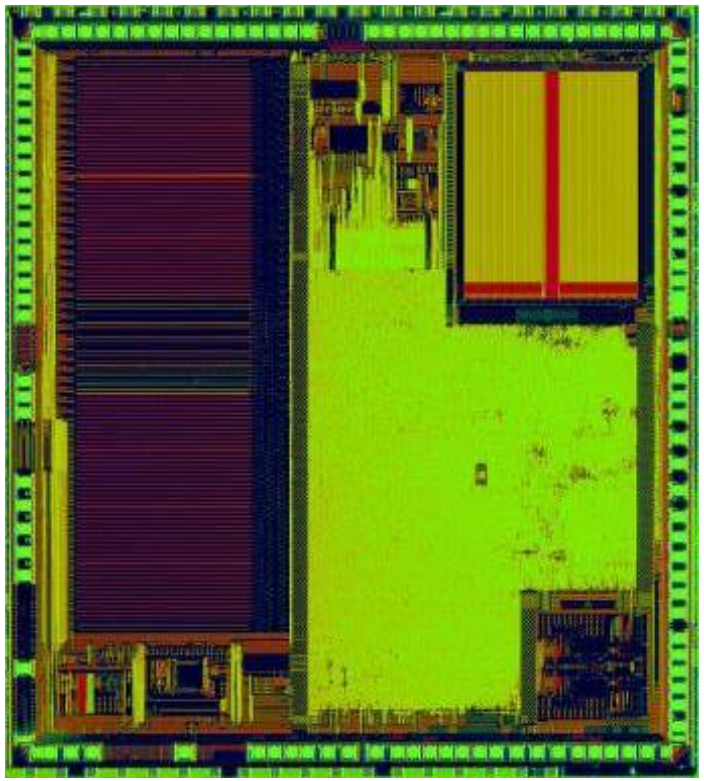
- High Performance
- Low Power
- Small Core Size
- Broad Software Ecosystem
- Widely Accepted
- Expandability

MICROCHIP
Regional Training Centers

MIPS® M4K® Core

With Microchip Enhancements

- The MIPS engine
  - **Up to 80 MHz**
  - **1.56 DMIPS/MHz *measured***
  - **32 32-bit core registers with shadow set**
  - **32-bit ALU, single cycle MAC**
- Microchip system enhancements for embedded performance
  - **256 byte prefetch cache**
  - **Low-latency vectored interrupt controller**
  - **Bus Matrix for parallelism**
- Debugging
  - **Hardware Trace with debug support**
  - **JTAG and Boundary Scan**



32-bit instr. mode for best speed
16-bit instr. mode for small code size
Easy to mix modes in single source file

MICROCHIP
Regional Training Centers

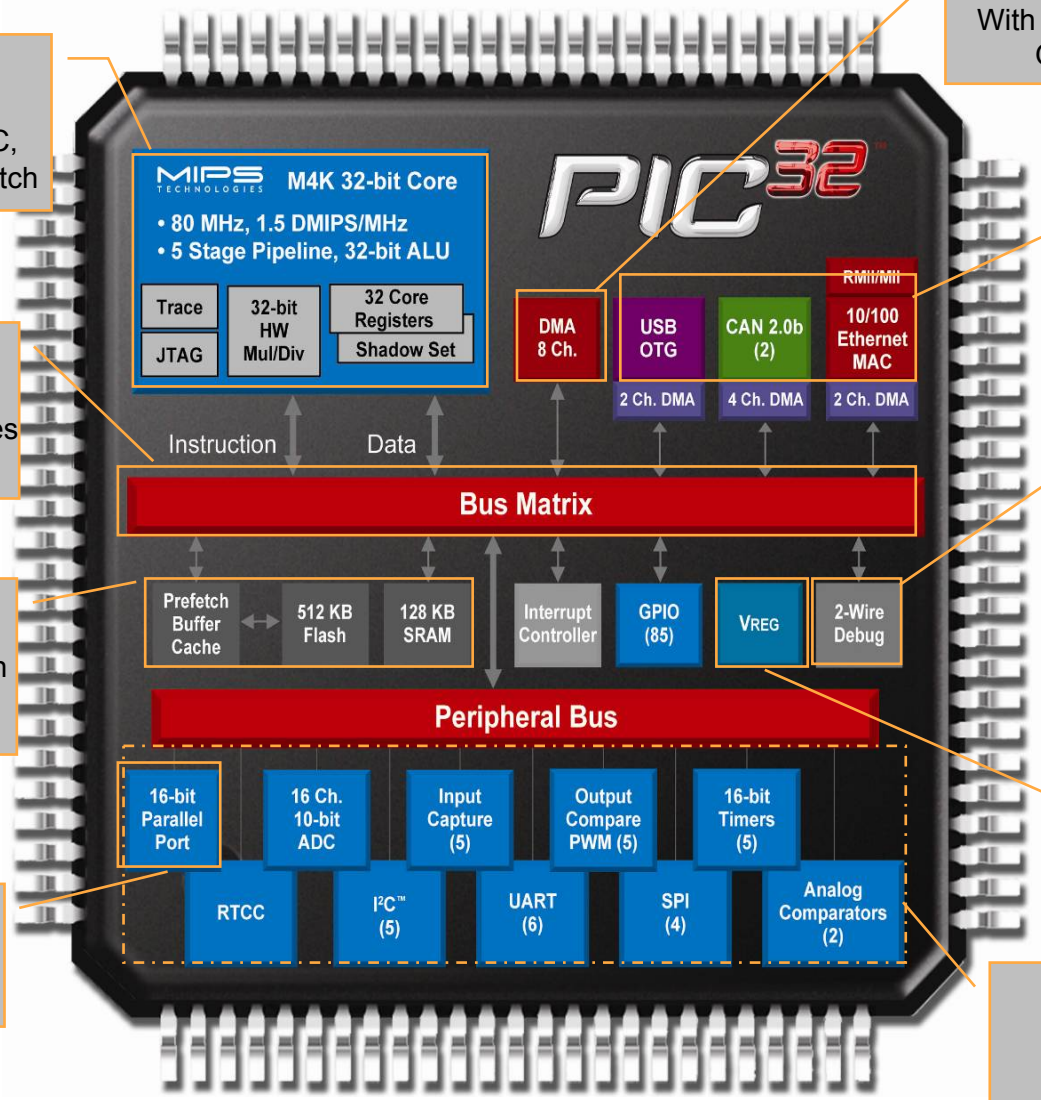
PIC32 Microcontroller Key Features

32-bit MIPS M4K® Core, Harvard Architecture, Single Cycle Hardware MAC, Fast Interrupts & Context Switch

High Throughput Bus Matrix, which Supports High Speed Concurrent Access to Memories and Peripherals

512KB 128-bit wide Self-programmable Flash, Predictive Instruction Pre-fetch 256 Byte Lockable Cache 128K RAM

16-bit Parallel Master Port, Connect SRAM, Flash, QVGA LCDs or other Peripherals



Direct Memory Access Controller, With Integrated CRC Module… Operates in Idle Mode

Integrated Connectivity Peripherals for fast cost effective operation: 10/100 Ethernet, 2x CAN, USB OTG

Compatible with Microchip Development Tools MPLAB® ICD 3, MPLAB REAL ICE™, PICkit™ 3, PM3

Single 2.3 to 3.6V Supply Power On Reset, Brown Out Reset, Low Voltage Detection

Rich Integrated Analog and Digital peripheral set, Compatible with 16-bit PIC® Microcontrollers

Silicon Highlights

- Compatible with PIC24/dsPIC33
  - Pin, Peripheral, and Library Compatible
- Higher Performance
  - MIPS32® Core at 80 MHz
  - Up to 4 Channels of DMA (PIC32MX3/4 Family)
  - Up to 16 Channels of DMA (5/6/7 Family)
- More Memory
  - 512 + 12 Kbytes Flash, Up to 128 KB RAM
  - Pre-fetch buffer w/cache
- Fast Interrupts
  - Hardware Vectored Interrupt Controller
  - Full 32 Register Shadow Set

MICROCHIP
Regional Training Centers

Compatibility and Performance 的例子 – 以常用的 UART 為例

PIC18F

SPEN	RX9		CREN	ADDEN	FERR	OERR	RX9D
	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
ABDOVF	RCMT	DTRXP	CKTXP	BRG16		WUE	ABDEN

- Basic control (RX/TX)
  - 9-bit mode
  - Address detect
- 16-bit baud rate generator
  - LIN support
  - Auto-baud, wake, sync
  - Polarity
- Error checking

PIC24F

SPEN	UFRZ	USIDL	IREN	RSTMD	ALTIO	UEN1	UEN0
WAKE	LPBACK	ABAUD	RXINV	BGRH	PDSEL1	PDSEL0	STSEL
TXISEL1	TXINV	TXISEL0		TXBRK	TXEN	TXBF	TRMT
RCISEL1	RCISEL0	ADDEN	RIDLE	PERR	FERR	OERR	RCDA

- Basic control (RX/TX)
  - 9-bit mode
  - Address detect
- 16-bit baud rate generator
  - LIN support
  - Auto-baud, wake, sync
  - Polarity
- Error checking

- Parity
- Flow control
- IRDA

PIC32MX

ON	FRZ	SIDL	IREN	RSTMD	ALTIO	UEN1	UEN0
WAKE	LPBACK	ABAUD	RXINV	BGRH	PDSEL1	PDSEL0	STSEL
TXISEL1	TXINV	TXISEL0		TXBRK	TXEN	TXBF	TRMT
RCISEL1	RCISEL0	ADDEN	RIDLE	PERR	FERR	OERR	RCDA

- Basic control (RX/TX)
  - 9-bit mode
  - Address detect
- 16-bit baud rate generator
  - LIN support
  - Auto-baud, wake, sync
  - Polarity
- Error checking

- Parity
- Flow control
- IRDA

- Address matching
- DMA

Peripheral Library APIs Make Code Transition Seamless

可使用於 PIC32 的硬體除錯工具

Regional Training Centers

Features/Speed/Trace

PICkit™ 3

Full Speed USB HID,
Run, Halt, SS,
Simple Breakpoints,
Program, Read
All of Microchip's Flash PIC® MCU and dsPIC DSCs

MPLAB® ICD 3

High Speed USB 2.0,
Run, Halt, SS
Software Breakpoints,
Complex Breakpoints,
Program, Read,
All of Microchip's Flash PIC® MCUs and dsPIC® DSCs

MPLAB REAL ICE™ Emulator

High Speed USB2.0
Run, Halt, SS

Software Breakpoints,
Complex Breakpoints/Triggers,
Stopwatch,
Program, Read,
Real-Time Watch,
Log, Trace,
Logic Probes,
Performance Pak
All of Microchip's Flash PIC MCUs and dsPIC DSCs

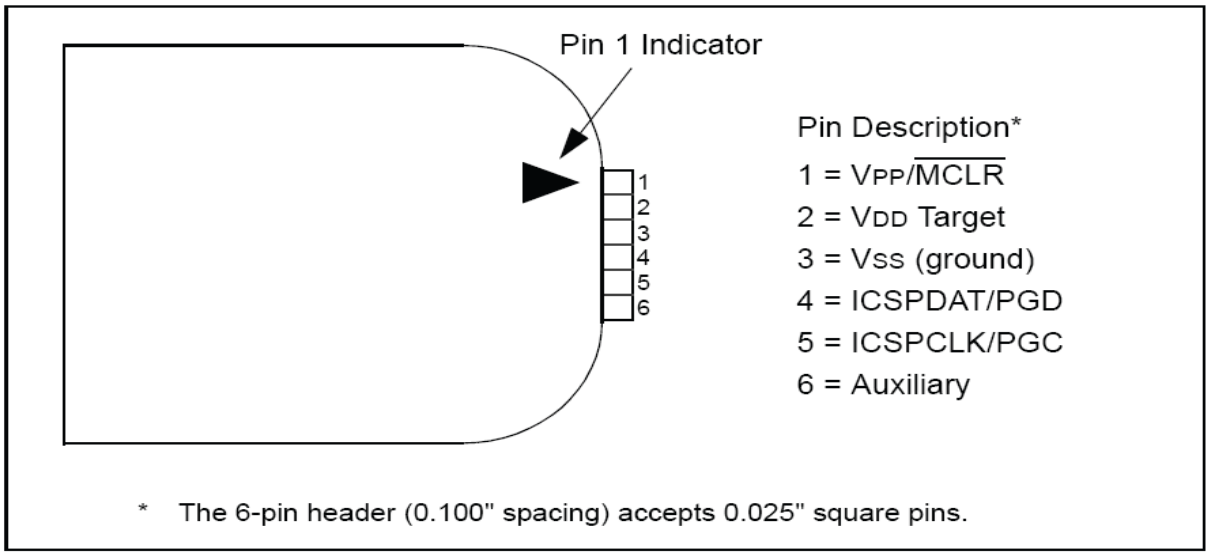
USD $70 $220 $500

PICkit 3

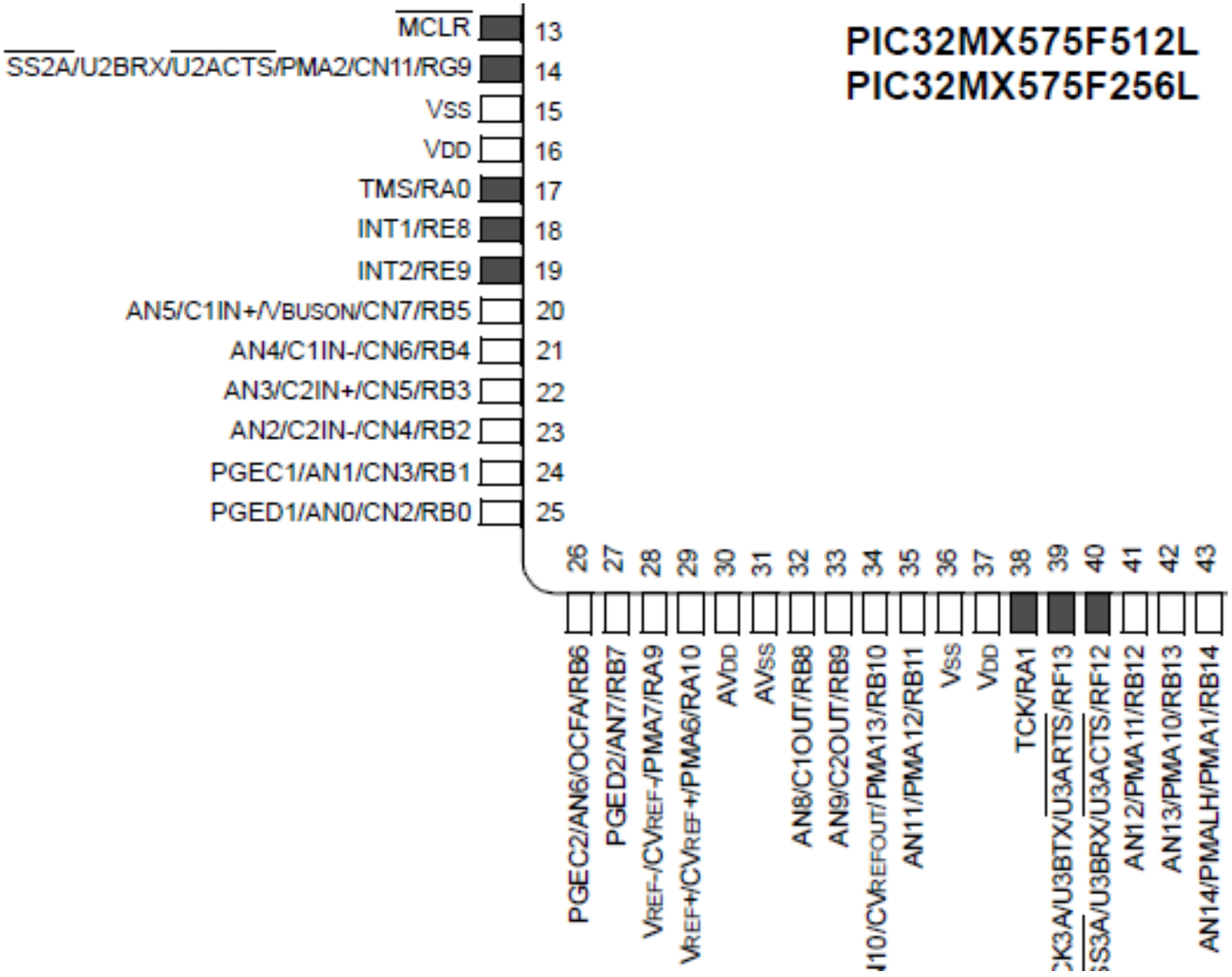
Debugger/Programmer 的界面

- Microchip MCU 使用 ICD/ICSP 界面與 Debugger/Programmer 連接
- APP1632 使用的 PIC32MX795F512L 總共有 2 組信號可供選擇
  - PGED1/PGEC1 , PGED2/PGEC2
  - 每一組都可用來燒錄程式但 Debug 必須在 Configuration Bits 被選到的接腳
  - 在 MPLAB IDE 中可看到 User 對 Configuration bits 的設定與選用
  - PICkit 3 的介面定義如下

FIGURE 1-2: PICkit™ 2 PROGRAMMER CONNECTOR PINOUT



PIC32 ICD/ICSP 信號的位置圖

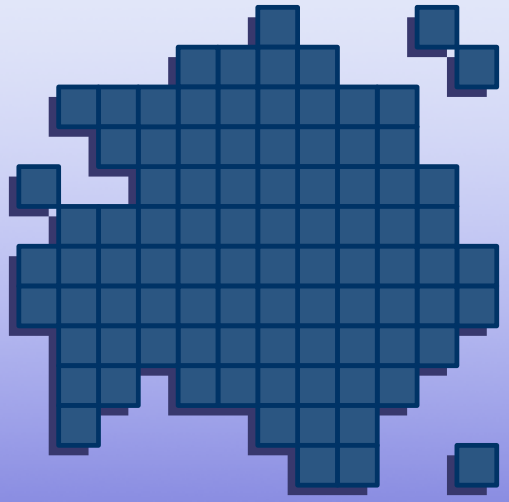


MICROCHIP
Regional Training Centers

周邊及開發工具的高度相容性 帶來快速且低風險的升級方式

Features	PIC24F	PIC24H	dsPIC33F	PIC32
Core	16-bit core with single cycle instructions (24-bit instructions) Power of 3 operand instructions			80MHz 32-bit ALU
Working Registers	Sixteen, 16-bit			Thirty two, 32-bit
Hardware DMA	NO	YES	YES	YES
Performance	16 MIPS	40 MIPS	40 MIPS	1.52 DMIPS/MHz
Voltage	3.3 V	3.3 V	3.3 V	3.3 V
Package	28 - 100 Pins	18 - 100 Pins	18 - 100 Pins	64 - 128 Pins
DSP	NO	NO	YES	NO
Operating Temperature	-40 to 85 Deg C -40 t0 125 Deg C	-40 to 85 Deg C -40 t0 125 Deg C -40 t0 140 Deg C	-40 to 85 Deg C -40 t0 125 Deg C -40 t0 140 Deg C	-40 to 85 Deg C
Flash	16K to 256K	12K to 256K	12K to 256K	32K to 512K
RAM	4K to 16K	1K to 30K	1K to 30K	8K to 128K
	Low Power	High Performance	High Performance	High Performance

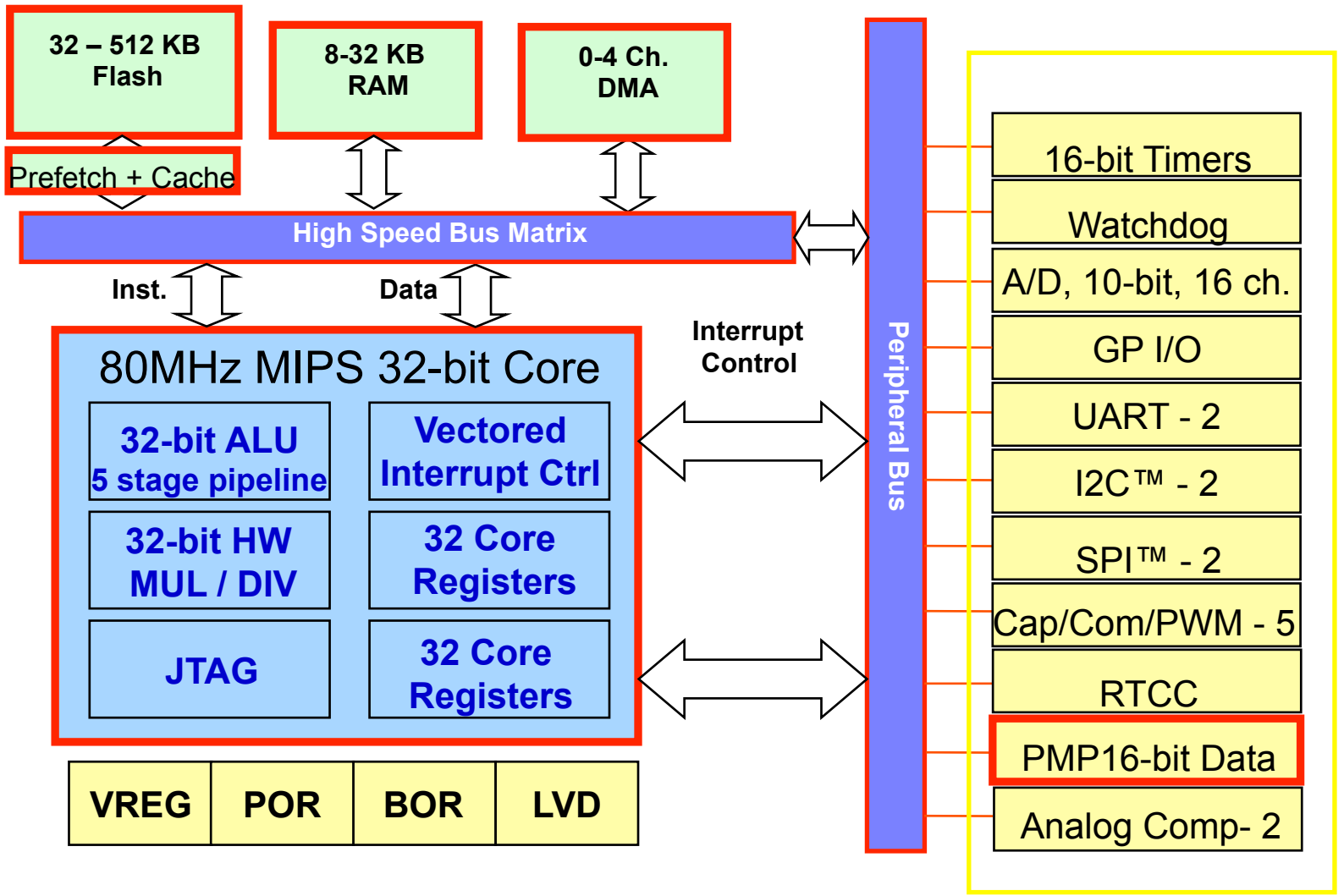
Scale Performance in MIPS & Add Computational Power
Across 4 Code-Compatible Families Of Products



PIC32系列
元件規格比較

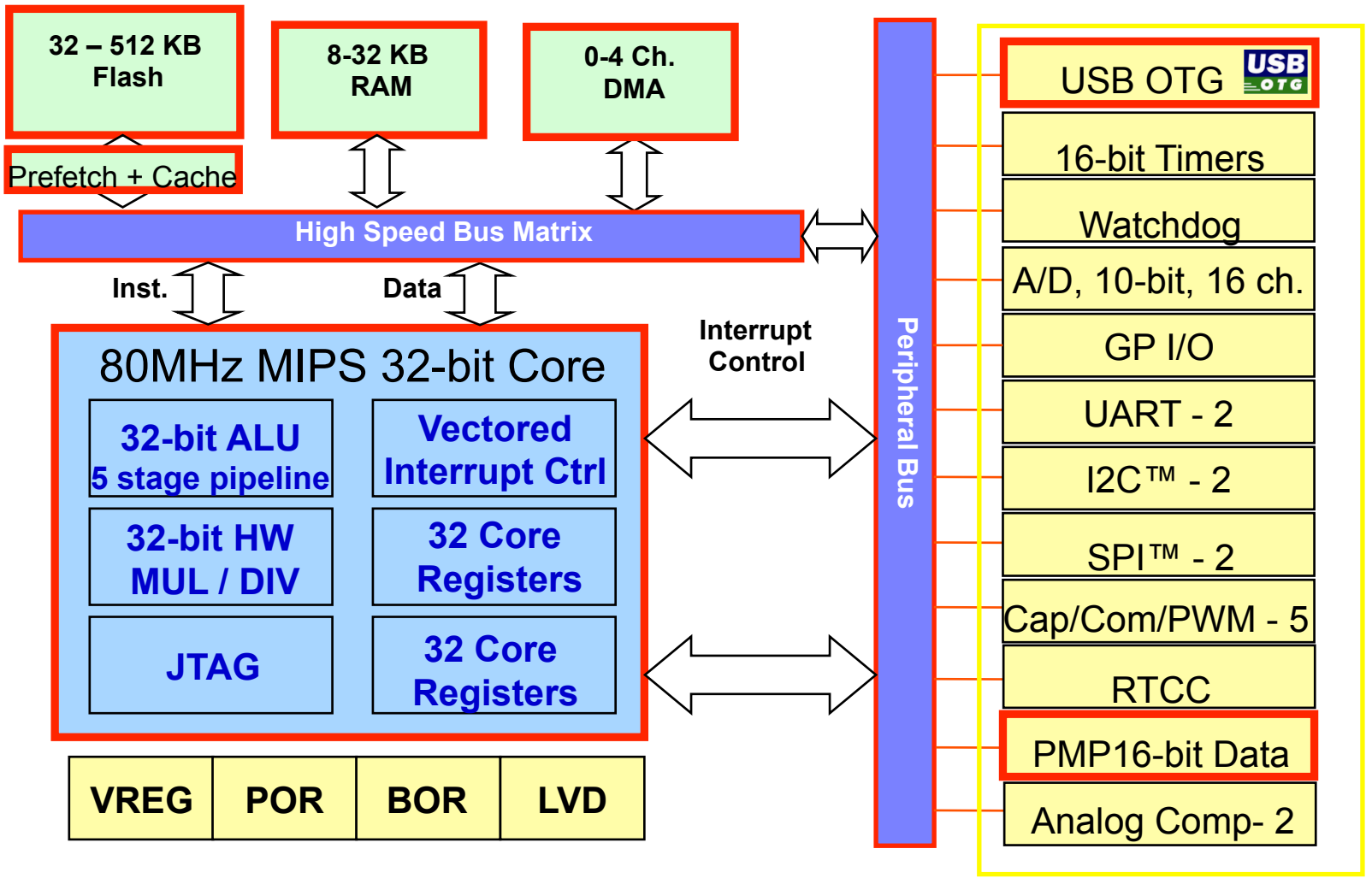
MICROCHIP
Regional Training Centers

PIC32MX3 (第一代 PIC32)
General Purpose Family



MICROCHIP
Regional Training Centers

PIC32MX4 (第二代)
USB 2.0 Family



PIC32MX5/6/7 (第三代)
New Peripheral Integration

6 UARTs
4 SPI
5 I2C

10/100 Megabit
Ethernet MAC

64k to 128k
RAM

1 or 2 CAN
Controllers

12 to 16 DMA
Channels

CERTIFIED USB

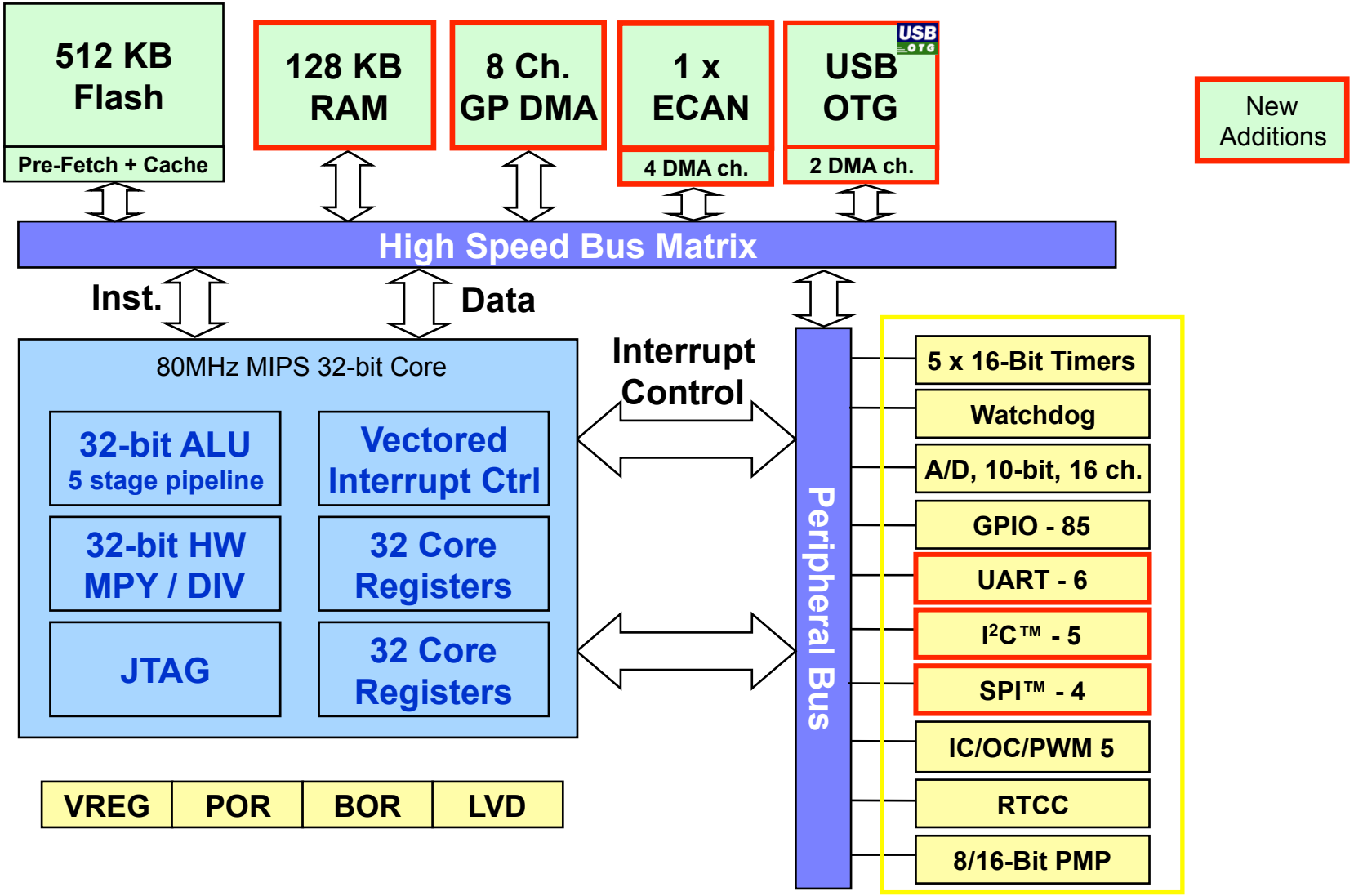
CAN

Carries Forward:

- Existing USB On-the-Go peripheral
- Same pin out as PIC32MX4 (USB) family
- Same high performance core

MICROCHIP
Regional Training Centers

PIC32MX5
USB OTG + 1 ECAN Family



MICROCHIP
Regional Training Centers

PIC32MX5 USB OTG + 1ECAN Family

- High-Performance 32-bit RISC CPU:
  - MIPS32® M4K™ 32-bit Core with 5-Stage Pipeline
  - 80 MHz Maximum Frequency
  - 1.56 DMIPS/MHz (Dhrystone 2.1) Performance at 0 Wait State Flash Access
  - Single-Cycle Multiply and High-Performance Divide Unit
  - MIPS16e™ Mode for Up to 40% Smaller Code Size
  - Two Sets of 32 Core Register Files (32-bit) to Reduce Interrupt Latency
  - Prefetch Cache Module to Speed Execution from Flash

- Microcontroller Features:
  - Operating Voltage Range of 2.3V to 3.6V
  - 256K to 512K Flash Memory (plus an additional 12KB of Boot Flash)
  - 64K SRAM Memory
  - Multiple Power Management Modes
  - Multiple Interrupt Vectors with Individually Programmable Priority
  - Fail-Safe Clock Monitor Mode
  - Configurable Watchdog Timer with On-Chip
  - Low-Power RC Oscillator for Reliable Operation

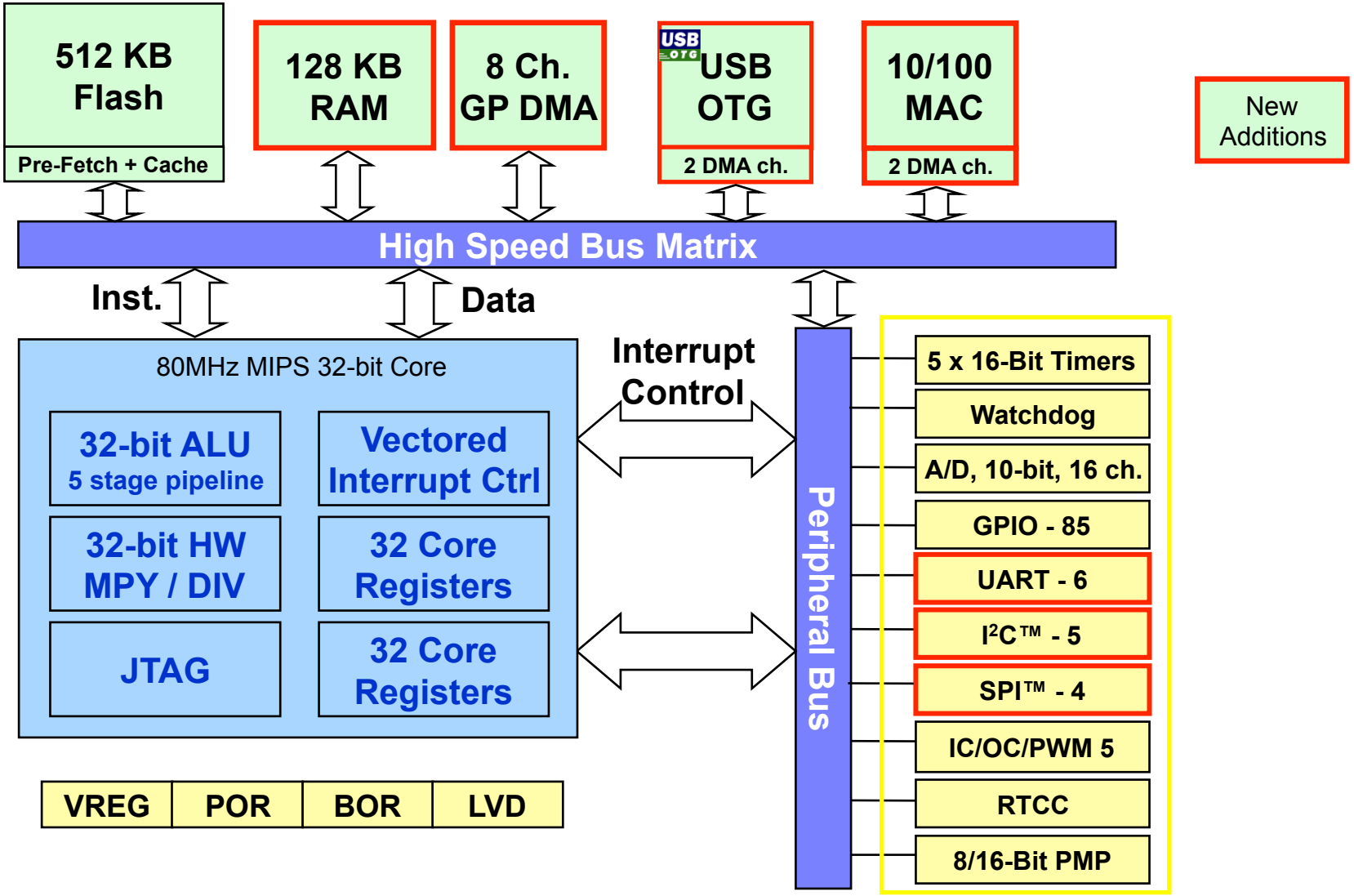
Packaging: 64/100 Pin TQFP and QFN
Temperature: -40 to +85 Deg C

- Peripheral Features:
  - Up to 4-Channel Hardware DMA with Automatic
  - Data Size Detection
  - USB 2.0 Compliant Full Speed Device and
  - On-The-Go (OTG) Controller
  - USB has a Dedicated DMA Channel
  - CAN module:
    - 2.0B Active with DeviceNet™ addressing support
    - Dedicated DMA channel
  - 10 MHz to 40 MHz Crystal Oscillator
  - Internal 8 MHz and 32 kHz Oscillators
  - Up to 5 I2C™ Modules
  - Up to 4 SPI Modules
  - Six UART Modules with:
    - RS-232, RS-485 and LIN 1.2 support
    - IrDA® with On-Chip Hardware Encoder and Decoder
  - Parallel Master and Slave Port (PMP/PSP)
  - Hardware Real-Time Clock/Calendar (RTCC)
  - Five 16-bit Timers/Counters
  - Five Capture Inputs,Five Compare/PWM Outputs
  - High-Speed I/O Pins Capable of Toggling Up to 80 MHz
  - High-Current Sink/Source (18 mA/18 mA) on All I/O Pins

- Analog Features:
  - Up to 16-Channel 10-bit Analog-to-Digital Converter:
    - 1000 ksps Conversion Rate
    - Conversion Available During Sleep, Idle
  - Two Analog Comparators
  - 5V Tolerant Input Pins (digital pins only)

MICROCHIP
Regional Training Centers

PIC32MX6
USB OTG + Ethernet Family



MICROCHIP
Regional Training Centers

PIC32MX6
USB OTG + Ethernet Family

- High-Performance 32-bit RISC CPU:
  - MIPS32® M4K™ 32-bit Core with 5-Stage Pipeline
  - 80 MHz Maximum Frequency
  - 1.56 DMIPS/MHz (Dhrystone 2.1) Performance at 0 Wait State Flash Access
  - Single-Cycle Multiply and High-Performance Divide Unit
  - MIPS16e™ Mode for Up to 40% Smaller Code Size
  - Two Sets of 32 Core Register Files (32-bit) to Reduce Interrupt Latency
  - Prefetch Cache Module to Speed Execution from Flash

- Microcontroller Features:
  - Operating Voltage Range of 2.3V to 3.6V
  - 256K to 512K Flash Memory (plus an additional 12KB of Boot Flash)
  - 64K to 128K SRAM Memory
  - Multiple Power Management Modes
  - Multiple Interrupt Vectors with Individually Programmable Priority
  - Fail-Safe Clock Monitor Mode
  - Configurable Watchdog Timer with On-Chip
  - Low-Power RC Oscillator for Reliable Operation

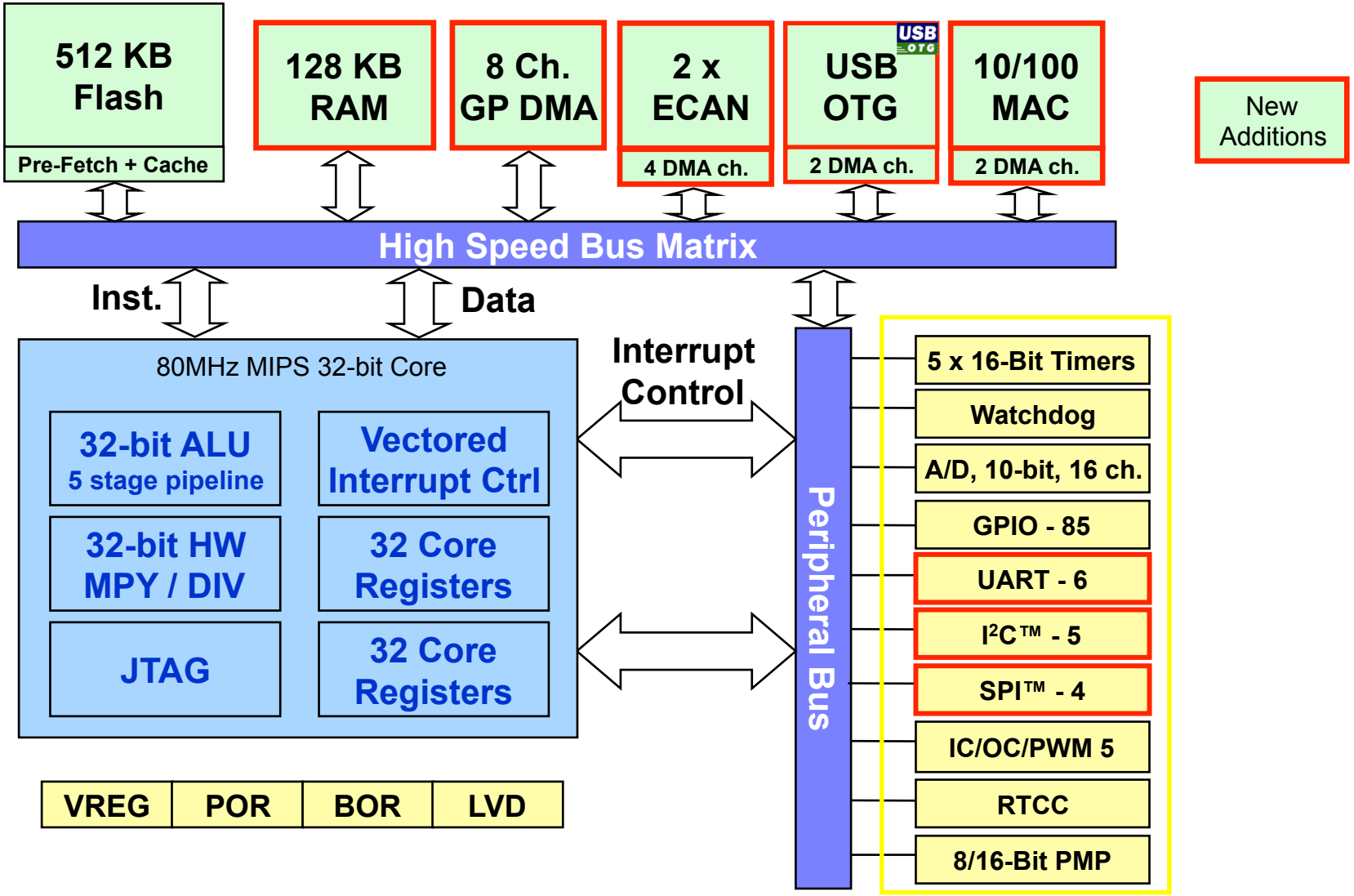
- Peripheral Features:
  - Up to 4-Channel Hardware DMA with Automatic
  - Data Size Detection
  - USB 2.0 Compliant Full Speed Device and
  - On-The-Go (OTG) Controller
  - USB has a Dedicated DMA Channel
  - 10/100 Mbps Ethernet MAC with MII and RMII interface:
    - Dedicated DMA channel
  - 10 MHz to 40 MHz Crystal Oscillator
  - Internal 8 MHz and 32 kHz Oscillators
  - Up to 5 I2C™ Modules
  - Up to 4 SPI Modules
  - Six UART Modules with:
    - RS-232, RS-485 and LIN 1.2 support
    - IrDA® with On-Chip Hardware Encoder and Decoder
  - Parallel Master and Slave Port (PMP/PSP)
  - Hardware Real-Time Clock/Calendar (RTCC)
  - Five 16-bit Timers/Counters
  - Five Capture Inputs,Five Compare/PWM Outputs
  - High-Speed I/O Pins Capable of Toggling Up to 80 MHz
  - High-Current Sink/Source (18 mA/18 mA) on All I/O Pins

- Analog Features:
  - Up to 16-Channel 10-bit Analog-to-Digital Converter:
    - 1000 ksps Conversion Rate
    - Conversion Available During Sleep, Idle
  - Two Analog Comparators
  - 5V Tolerant Input Pins (digital pins only)

Packaging: 64/100 Pin TQFP and QFN
Temperature: -40 to +85 Deg C

MICROCHIP
Regional Training Centers

PIC32MX7 USB OTG+ 2 ECAN + Ethernet Family



MICROCHIP
Regional Training Centers

PIC32MX7
USB OTG+ 2 ECAN + Ethernet

- High-Performance 32-bit RISC CPU:
  - MIPS32® M4K™ 32-bit Core with 5-Stage Pipeline
  - 80 MHz Maximum Frequency
  - 1.56 DMIPS/MHz (Dhrystone 2.1) Performance at 0 Wait State Flash Access
  - Single-Cycle Multiply and High-Performance Divide Unit
  - MIPS16e™ Mode for Up to 40% Smaller Code Size
  - Two Sets of 32 Core Register Files (32-bit) to Reduce Interrupt Latency
  - Prefetch Cache Module to Speed Execution from Flash
- Microcontroller Features:
  - Operating Voltage Range of 2.3V to 3.6V
  - 256K to 512K Flash Memory (plus an additional 12KB of Boot Flash)
  - 64K to 128K SRAM Memory
  - Multiple Power Management Modes
  - Multiple Interrupt Vectors with Individually Programmable Priority
  - Fail-Safe Clock Monitor Mode
  - Configurable Watchdog Timer with On-Chip
  - Low-Power RC Oscillator for Reliable Operation

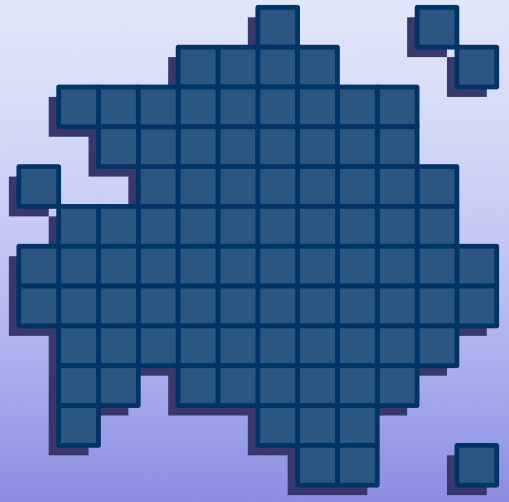
Packaging: 64/100 Pin TQFP and QFN
121-pin BGA
Temperature: -40 to +85 Deg C

- Peripheral Features:
  - Up to 8-Channel Hardware DMA with Automatic Data Size Detection
  - USB 2.0 Compliant Full Speed Device and
  - On-The-Go (OTG) Controller
  - USB has a Dedicated DMA Channel
  - 10/100 Mbps Ethernet MAC with MII and RMII interface:
    - Dedicated DMA channel
  - CAN module:
    - 2, CAN2.0B Active with DeviceNet™ addressing support
    - Dedicated DMA channel
  - 10 MHz to 40 MHz Crystal Oscillator
  - Internal 8 MHz and 32 kHz Oscillators
  - Up to 5 I2C™ Modules
  - Up to 4 SPI Modules
  - Six UART Modules with:
    - RS-232, RS-485 and LIN 1.2 support
    - IrDA® with On-Chip Hardware Encoder and Decoder
  - Parallel Master and Slave Port (PMP/PSP)
  - Hardware Real-Time Clock/Calendar (RTCC)
  - Five 16-bit Timers/Counters
  - Five Capture Inputs,Five Compare/PWM Outputs
- Analog Features:
  - Up to 16-Channel 10-bit Analog-to-Digital Converter:
    - 1000 ksps Conversion Rate
    - Conversion Available During Sleep, Idle
  - Two Analog Comparators

MICROCHIP
Regional Training Centers

MCU4101 使用實驗板

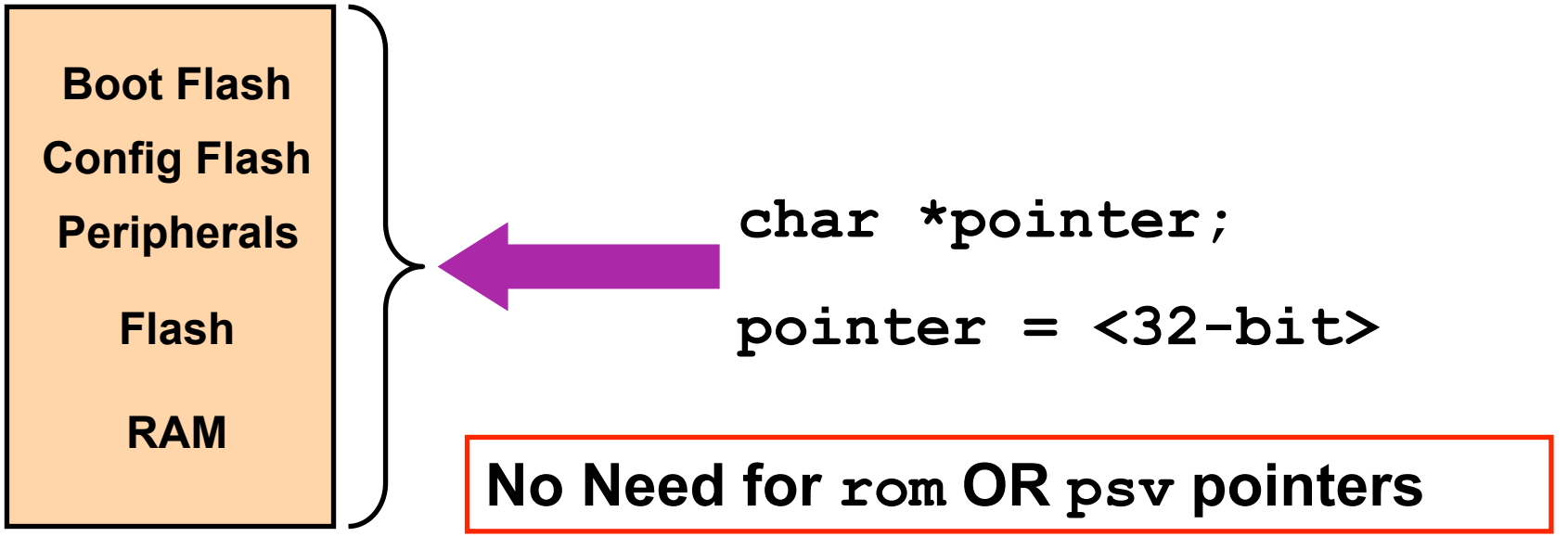
- PIC32 RTC (MCU4101) 所使用的實驗板為：
  APP1632 or Explorer-16
- 實驗板使用 CPU 模組 (APP1632-2)所搭載的
  CPU: PIC32MX795F512L-80I/PT
  - F512 : 512K Bytes Flash Memory
  - L : 100-pin, H: 64-pin
  - 80：最高執行速度 80MHz
  - I ：表示工業溫度規格(-40 ºC ~ +85 ºC)
  - PT : TQFP 薄型包裝
- APP1632-1 & APP1632-3 CPU 模組可支援
  PIC24FJ GA or PIC24FJ GB Family



PIC32 Memory Organization

MICROCHIP
Regional Training Centers

Unified Memory Map



MICROCHIP
Regional Training Centers

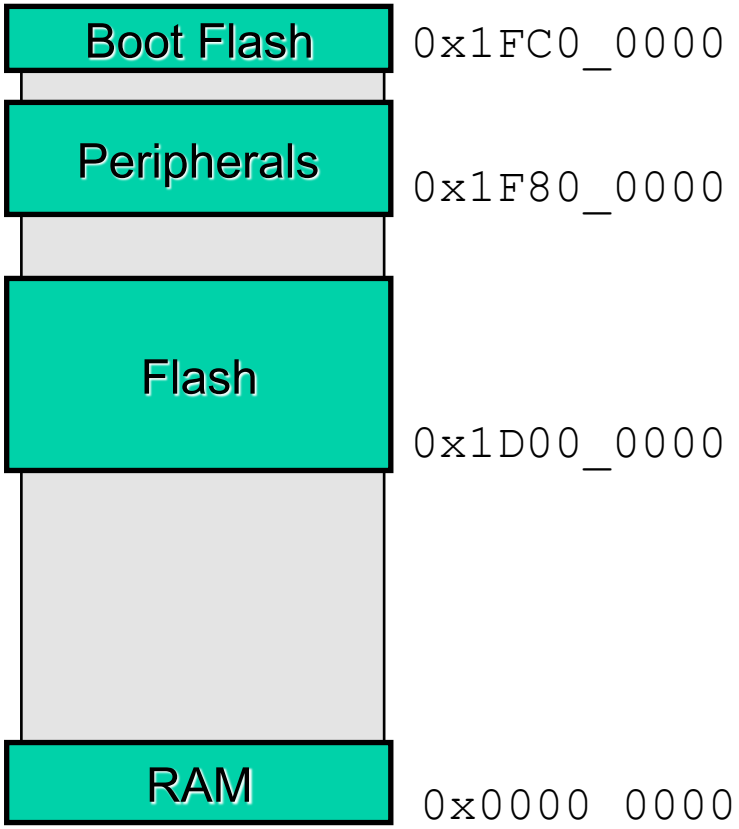
虛擬與實體記憶體
Virtual & Physical Memory

- **Physical Memory**
  - 所有的硬體都是使用實體(Physical) 記憶體，包含
    - **Program Memory**
    - **Data Memory**
    - **Peripherals**
      - **DMA** 及 **Flash Controllers** 使用實體記憶體
- **Virtual Memory**
  - **CPU** 所使用
  - 使用於指令的提取與執行

MICROCHIP
Regional Training Centers

實體記憶體

- Boot Flash is 12KB for all parts
- Peripherals are memory -mapped
- User Flash is 32K to 512K depending on device
- RAM 8K to 128K



實體記憶體最大定址空間為4GB，實際使用到512MB

Kernel & User Mode 定義
C32 使用 Kernel Mode 執行程式

Kernel Mode

In order to access many of the hardware resources, the processor must be operating in Kernel mode. Kernel mode gives software access to the entire address space of the processor as well as access to privileged instructions.

The processor operates in Kernel mode when the DM bit in the DEBUG register is '0' and the STATUS register contains one, or more, of the following values:

UM = 0 ERL = 1 EXL = 1

User Mode

When executing in User mode, software is restricted to use a subset of the processor's resources. In many cases it is desirable to keep application-level code running in User mode where if an error occurs it can be contained and not be allowed to affect the Kernel mode code.

Applications can access Kernel mode functions through controlled interfaces such as the SYSCALL mechanism.

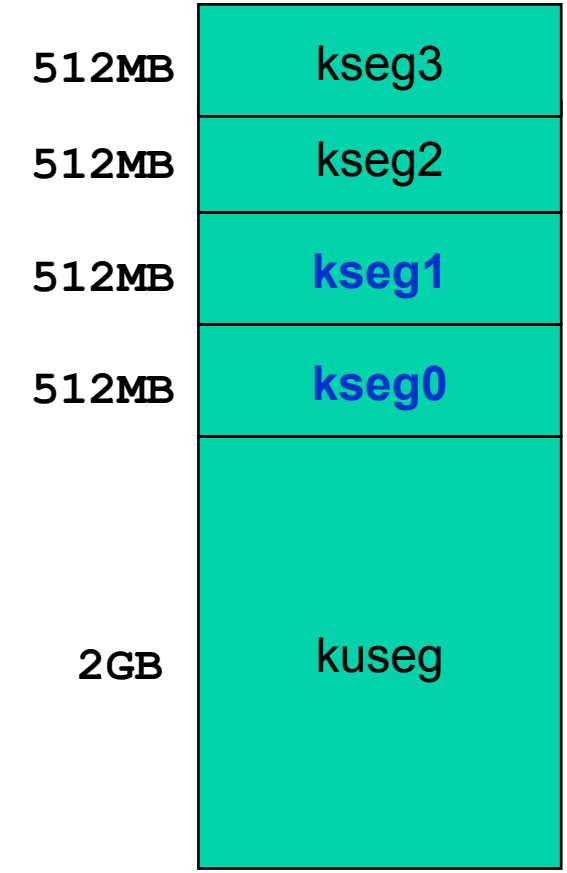
As seen in Figure 2-12, User mode software has access to the USEG memory area.

To operate in User mode, the STATUS register must contain each the following bit values:

UM = 1 EXL = 0 ERL = 0

MICROCHIP
Regional Training Centers

虛擬記憶體的配置



For EJTAG probe only

Not used in PIC32

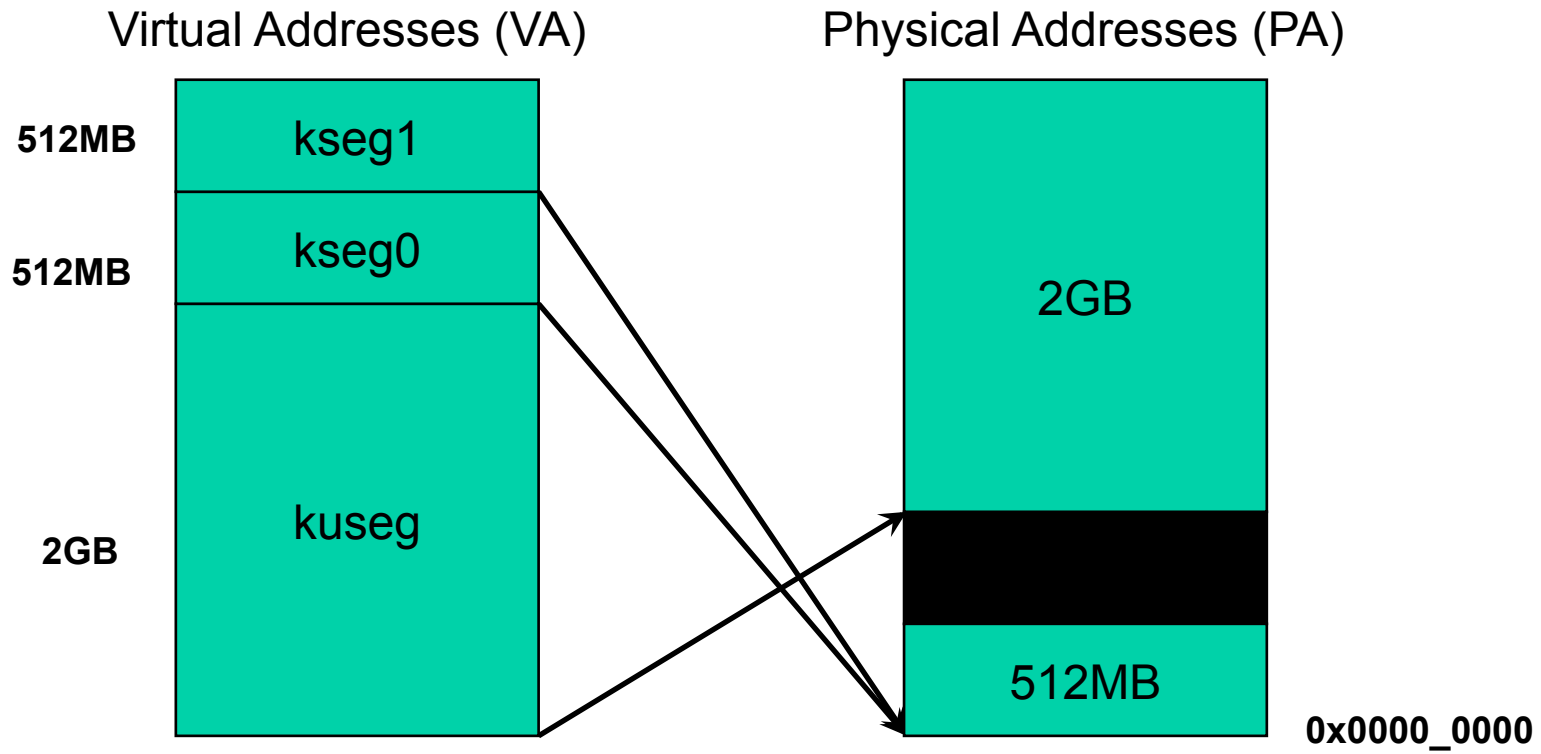
Non-cacheable Kernel

Cacheable Kernel

Cacheable Kernel & User

MICROCHIP
Regional Training Centers

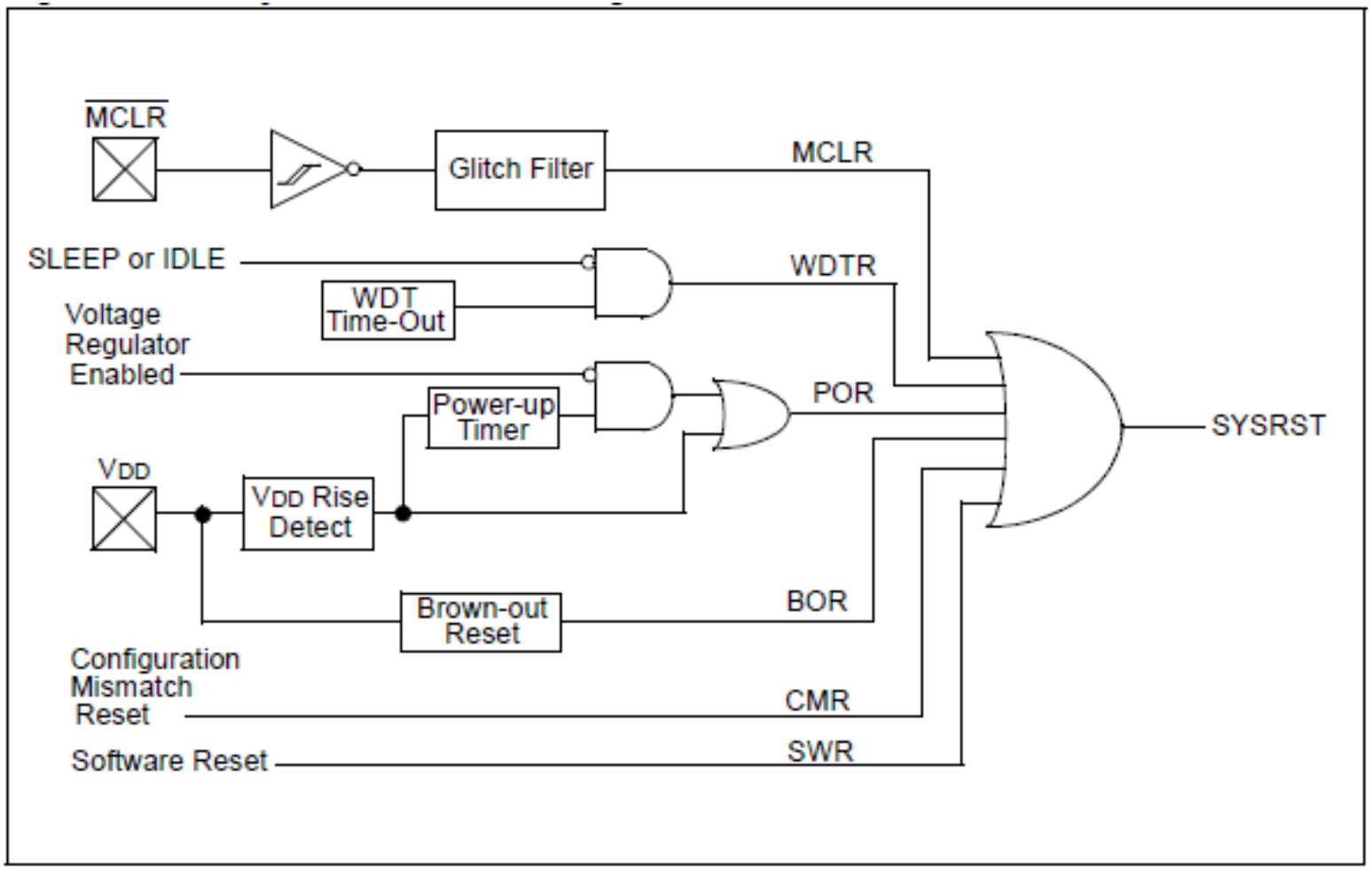
Fixed Mapping Translation



- CPU uses VA, Peripherals (DMA , Flash Controllers) use PA
- kseg0 & kseg1 對應到同一塊實體記憶體 (512MB)
- kseg0 支援指令快取功能
- Kseg1 不支援指令快取功能

PIC32 RESET 來源

- 底下為 **PIC32 RESET** 的來源



MICROCHIP
Regional Training Centers

PIC32 RESET

- 不同 **RESET** 對 **RCON** 暫存器中各相關旗標的影響
  - **RESET** 位址為 **KSEG1** 上的 **Boot Flash** 起點
    - **BFC0_0000h**
  - **MPLAB C32** 的起動模組會接續必要的工作

Condition	Program Counter	EXTR	SWR	WDTO	SLEEP	IDLE	CMR	BOR	POR
Power-on Reset	BFC0_0000h	0	0	0	0	0	0	1	1
Brown-out Reset	BFC0_0000h	0	0	0	0	0	0	1	u
$\overline{\text{MCLR}}$ Reset during Run Mode	BFC0_0000h	1	u	u	u	u	u	u	u
$\overline{\text{MCLR}}$ Reset during IDLE Mode	BFC0_0000h	1	u	u	u	1(1)	u	u	u
$\overline{\text{MCLR}}$ Reset during SLEEP Mode	BFC0_0000h	1	u	u	1(1)	u	u	u	u
Software Reset Command	BFC0_0000h	u	1	u	u	u	u	u	u
Configuration Word Mismatch Reset	BFC0_0000h	u	u	u	u	u	1	u	u
WDT Time-out Reset during Run Mode	BFC0_0000h	u	u	1	u	u	u	u	u
WDT Time-out Reset during IDLE Mode	BFC0_0000h	u	u	1	u	1(1)	u	u	u
WDT Time-out Reset during SLEEP Mode	BFC0_0000h	u	u	1	1(1)	u	u	u	u
Interrupt Exit from IDLE Mode	Vector	u	u	u	u	1(1))	u	u	u
Interrupt Exit from SLEEP Mode	Vector	u	u	u	1(1)	u	u	u	u

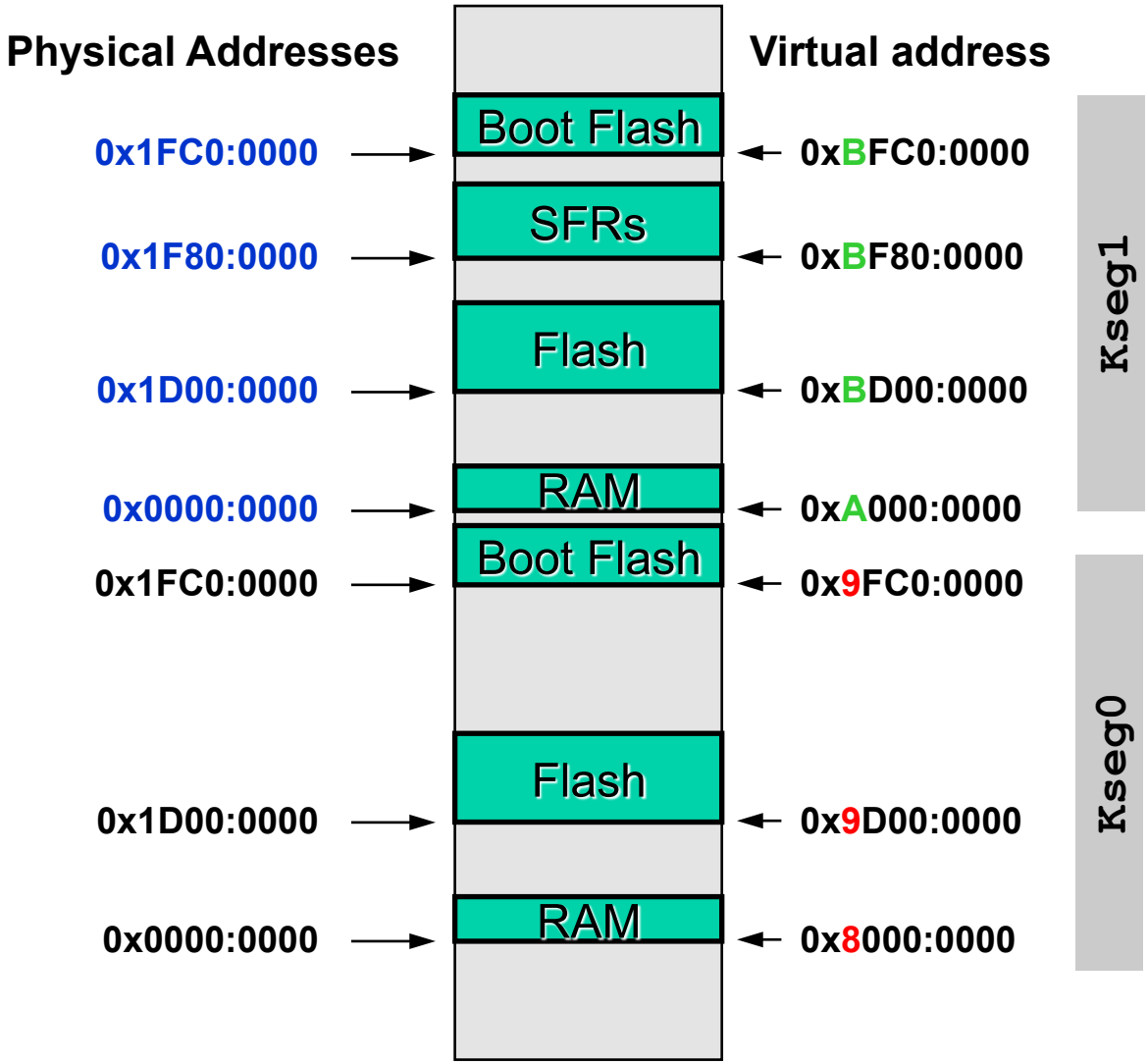
MICROCHIP
Regional Training Centers

Program Memory Partitioning

- When RESET
  - The user partition does not exist
    - BMXPUBPBA is initialized to “0”
  - RESET Address @ Boot Flash in KSEG1
    - VA @ BFC0_0000h, PA @ 1FC0_0000
  - Program Flash Memory mapped to Kernel mode program space
    - Virtual address : KSEG1 (0xBD00_0000)
      - PA @ 0x1D00_0000
    - Virtual address : KSEG0 (0x9D00_0000)
      - PA @ 0x1D00_0000 (有快取功能)

MICROCHIP
Regional Training Centers

PIC32 Memory Mapping

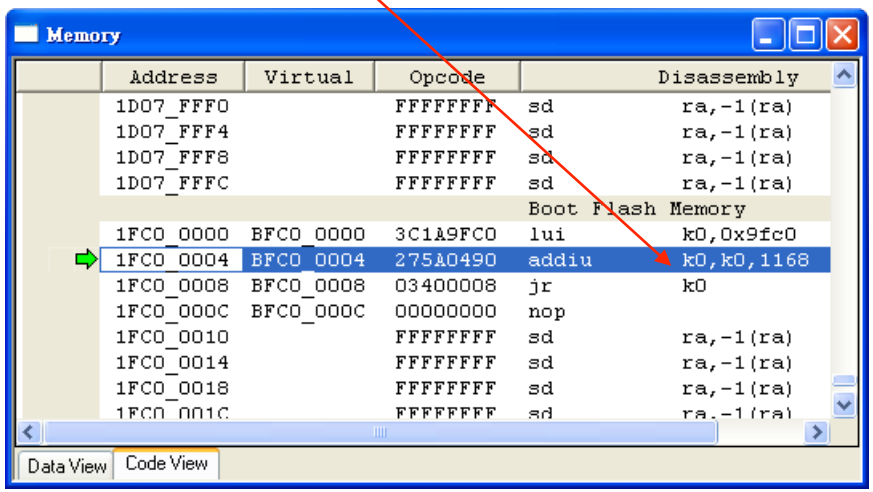


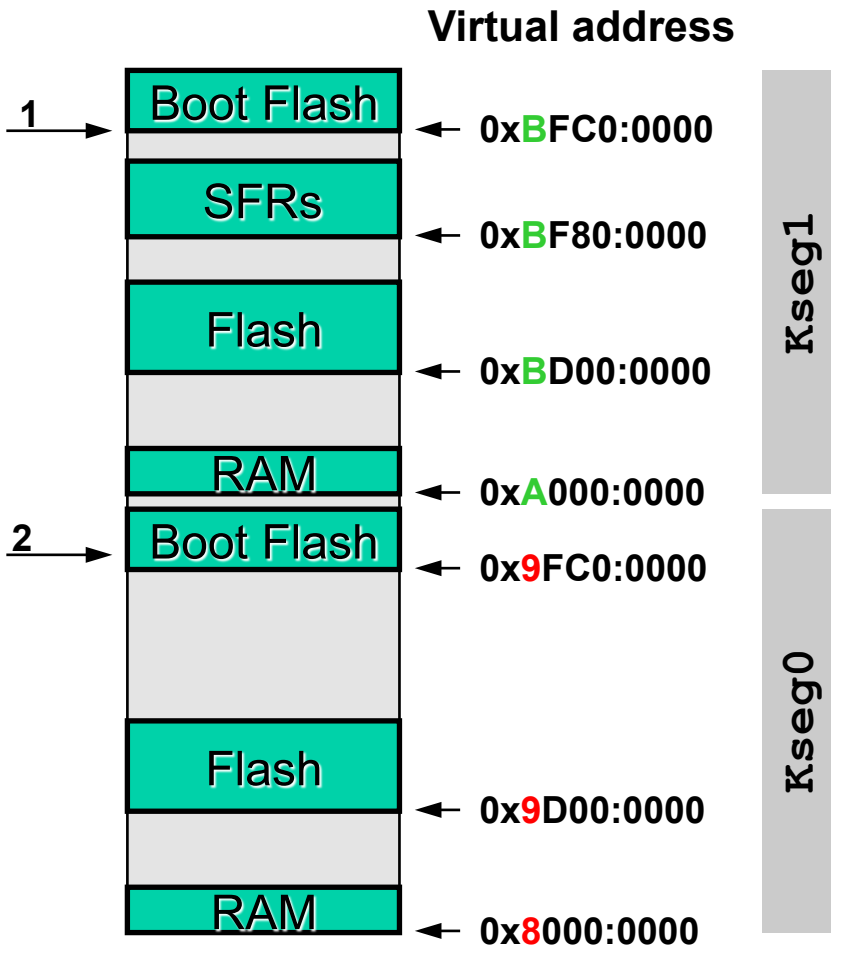
C32 Boot Flow-*1*

1. **Reset 後 PC 跳到 0xBFC0_0000 (KSEG1 Boot Flash)執行**

2. **KSEG1 切換到 KSEG0 的 Boot Flash @ 0x9FC0_0490 執行。**

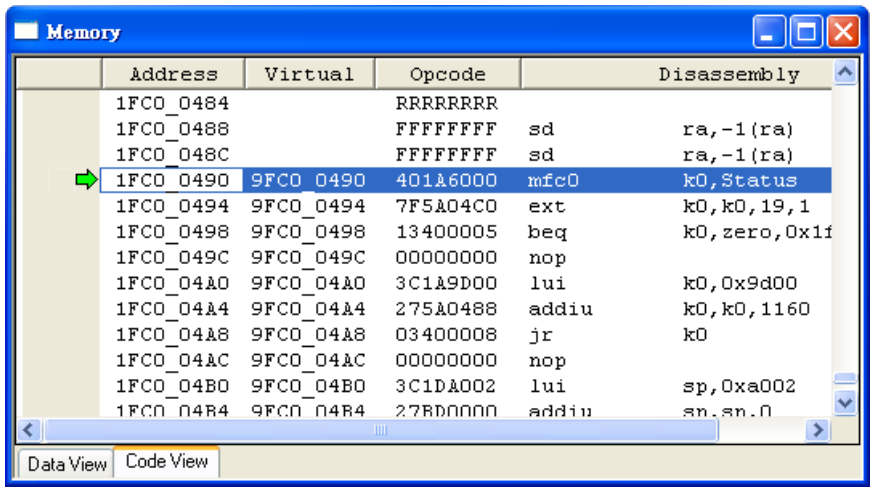
PC = 0x9FC0:0000 + *offset*
(1168 = 0x490) *Kseg0

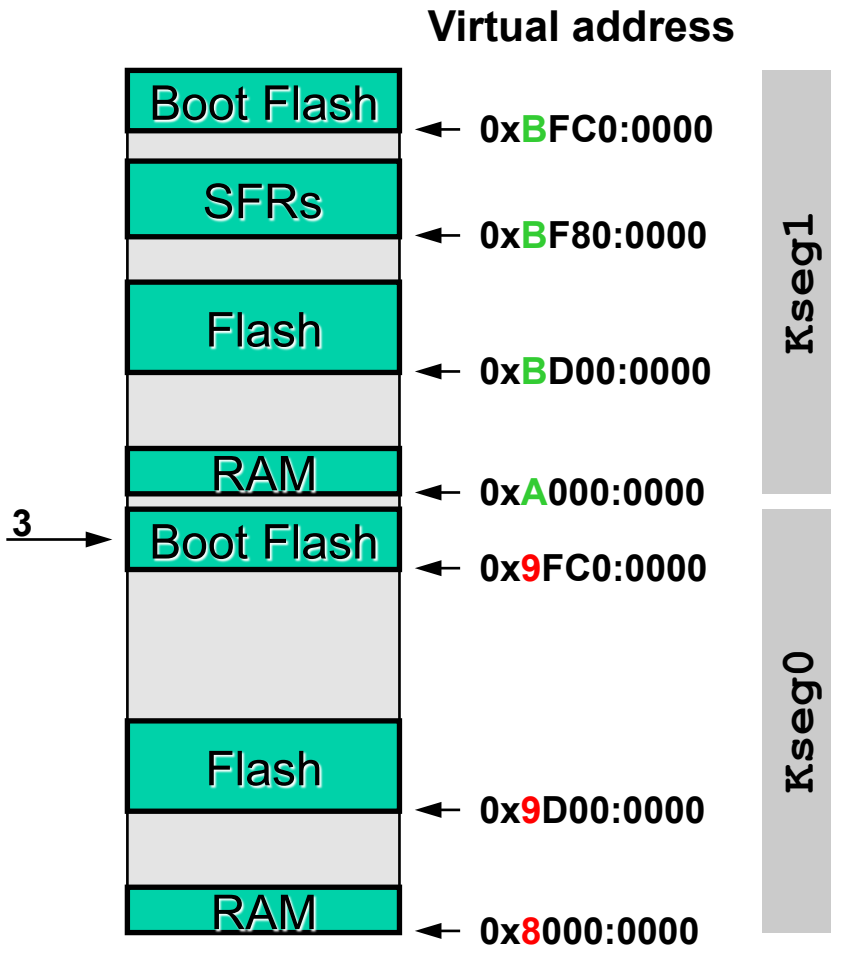




C32 Boot Flow-2

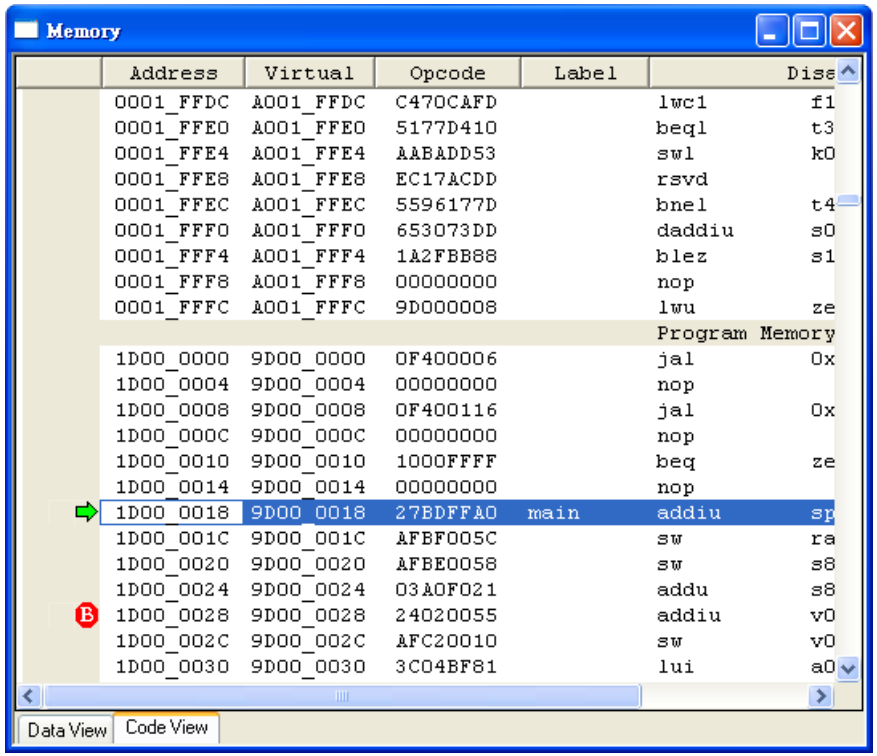
3. 程式在 **KSEG0** 的 **Boot Flash** 區塊開始執行 **C32** 的初始設定。

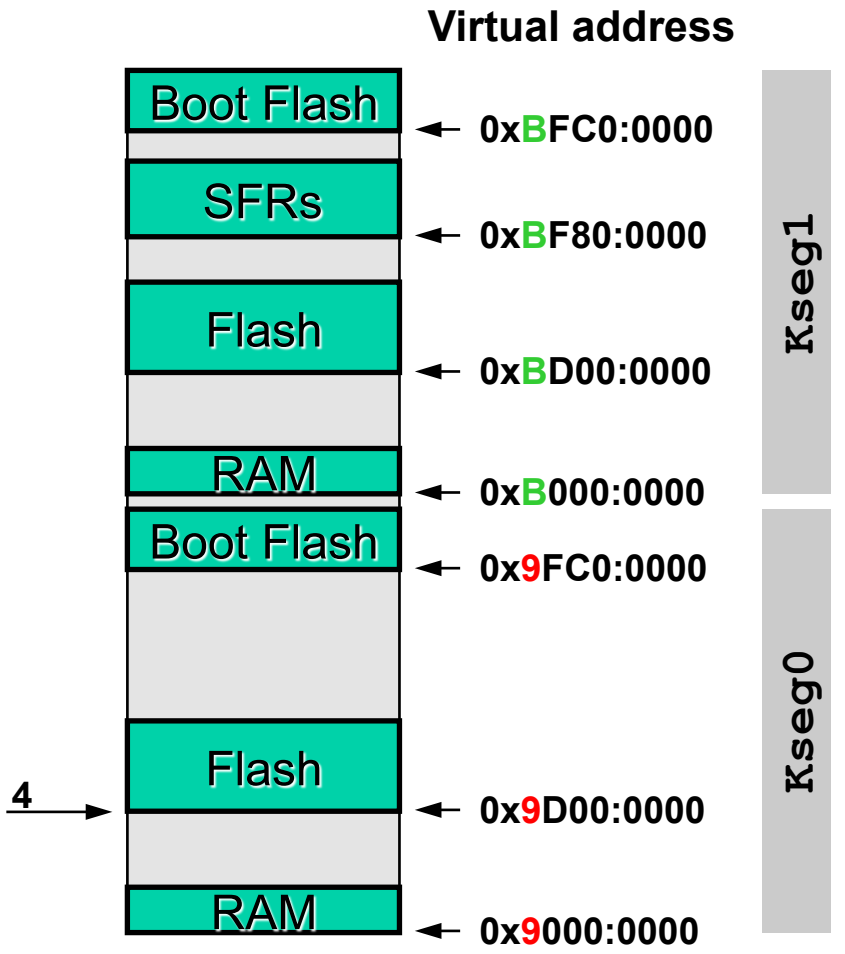




C32 Boot Flow-3

4. 程式再跳到 **KSEG0 (Program Flash)**
   0x9D00_0000 處執行然後進入 **main( )** 。





MICROCHIP
Regional Training Centers

誰控制著這些執行的位址的分配

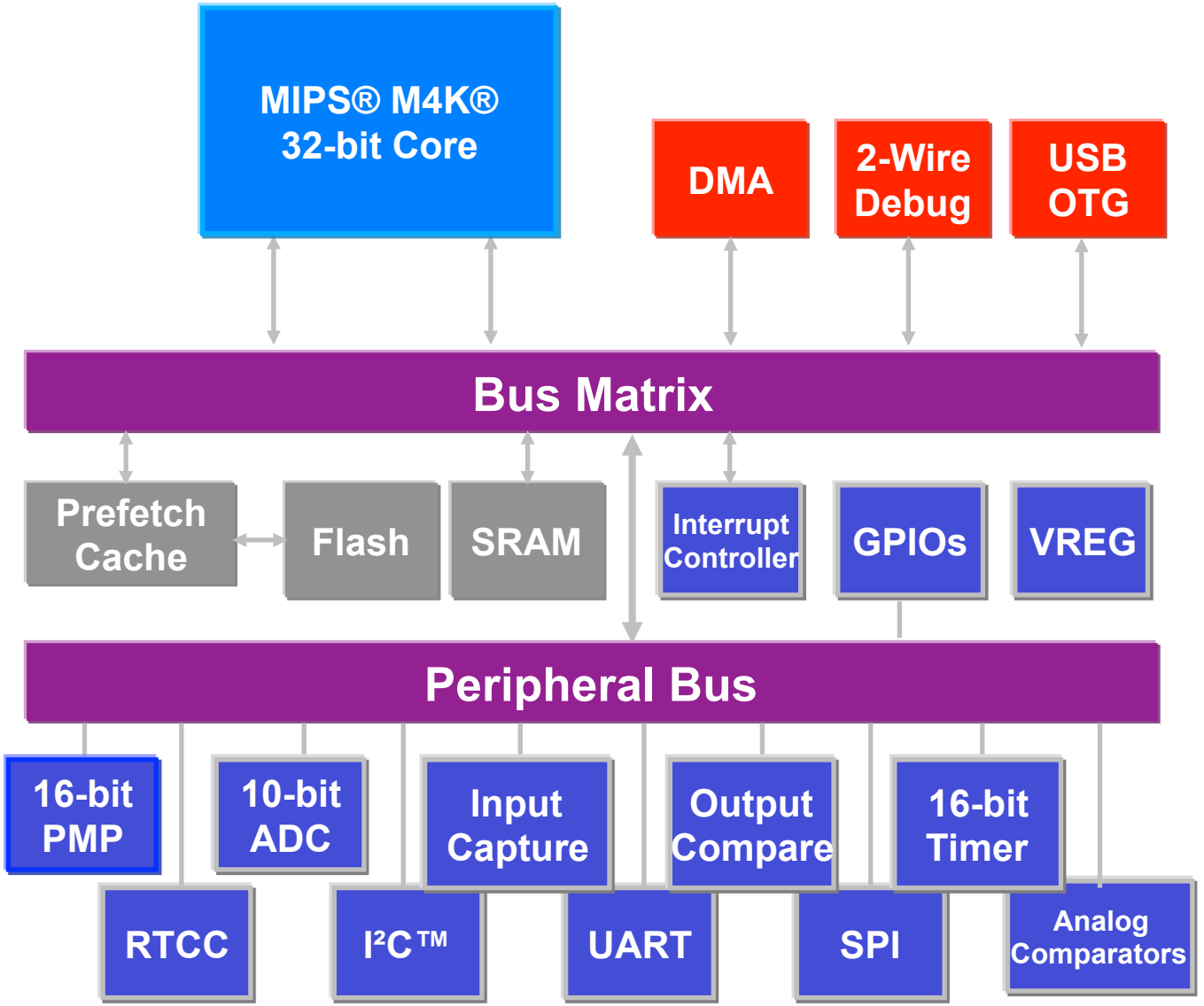
- **MPLAB LINK32 -** 需要每個元件的記憶體配置資料
- **MPLAB C32** 語言工具已經為個別元件定義了預設的 **Linker Script !**
  - **procdefs.ld**
  - 每一個不同的 **PIC32** 都有一個 對應的**procdefs.ld** 檔
  - **PIC32MX795F512L** 這個 **MCU** 的 **.ld** 檔的位置 :
    - **C:\Program Files\Microchip\MPLAB C32\ pic32mx\lib\proc\32MX795F512L**
- 所以只要正確修改 **procdefs.ld** 就可以更動程式擺放的起點，進而做到可以被 **Bootloader** 載入的 **C** 應用程式。

摘錄 PIC32MX795F512L procdefs.ld

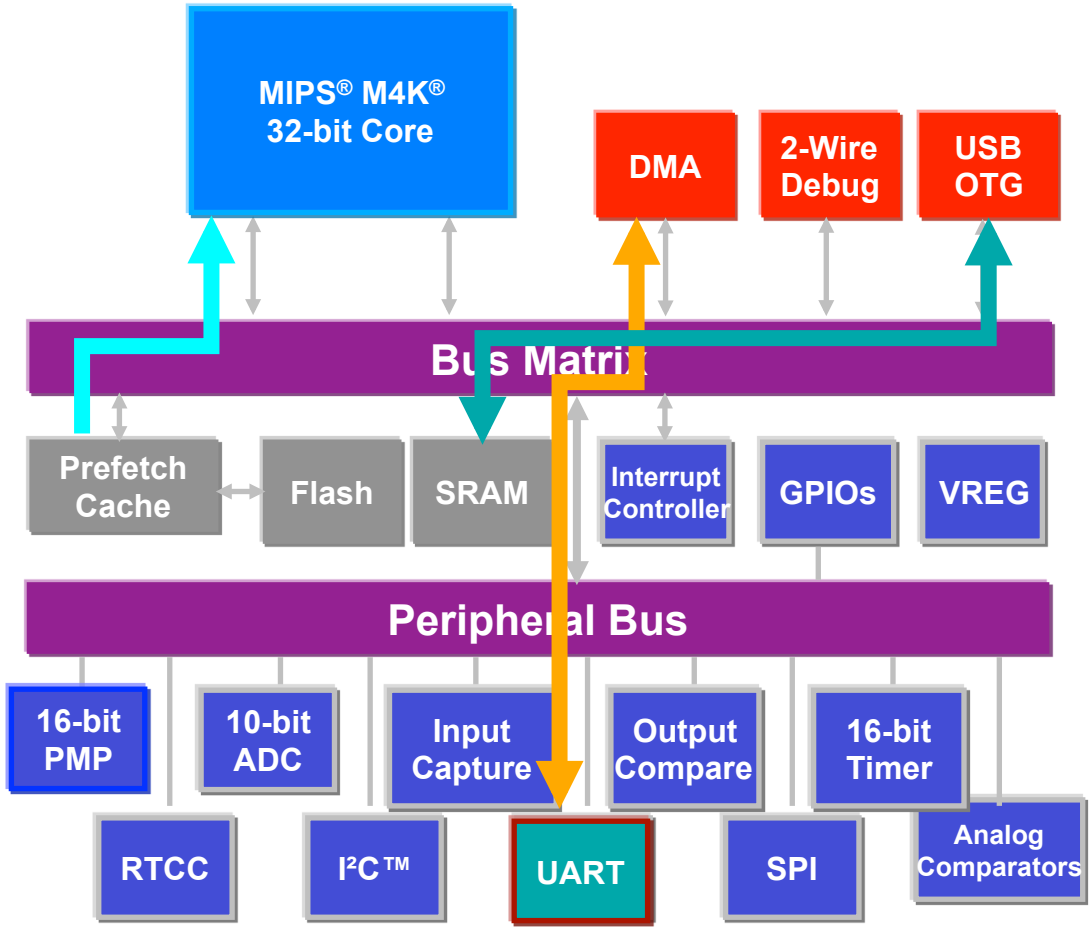
```
/*************************************************************************
 * Memory Address Equates
 *************************************************************************/
_RESET_ADDR                 = 0xBFC00000;    第一步
_BEV_EXCPT_ADDR             = 0xBFC00380;
_DBG_EXCPT_ADDR             = 0xBFC00480;
_DBG_CODE_ADDR              = 0xBFC02000;
_GEN_EXCPT_ADDR             = _ebase_address + 0x180;

/*************************************************************************
 * Memory Regions
 *
 * Memory regions without attributes cannot be used for orphaned sections.
 * Only sections specifically assigned to these regions can be allocated
 * into these regions.
 *************************************************************************/
MEMORY
{
  kseg0_program_mem    (rx)          : ORIGIN = 0x9D000000, LENGTH = 0x80000   第三步
  kseg0_boot_mem                     : ORIGIN = 0x9FC00490, LENGTH = 0x970     第二步
  exception_mem                      : ORIGIN = 0x9FC01000, LENGTH = 0x1000
  kseg1_boot_mem                     : ORIGIN = 0xBFC00000, LENGTH = 0x490
  debug_exec_mem                     : ORIGIN = 0xBFC02000, LENGTH = 0xFF0
```

MICROCHIP
Regional Training Centers

High Performance Bus Matrix



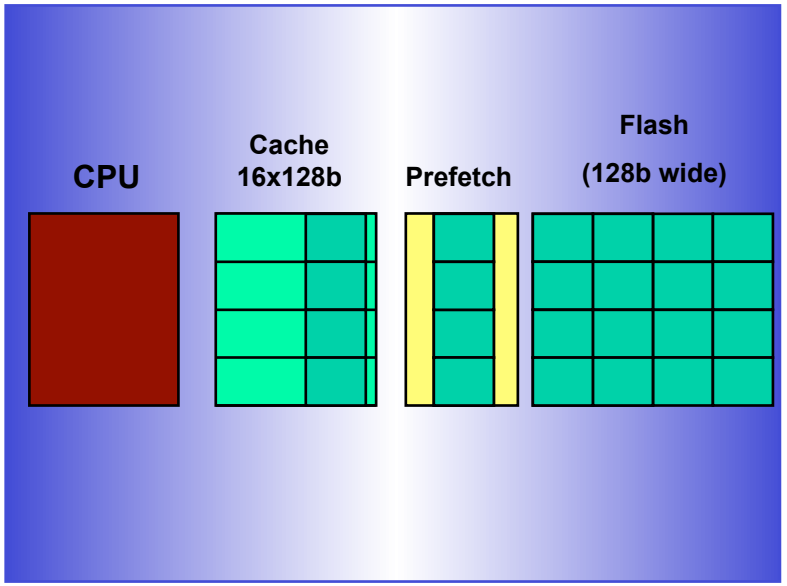
High Performance Bus Matrix



- **High Performance Bus Masters**
  - CPU Core
  - DMA
  - 2-Wire Debug
  - USB OTG
- **Multiple bus transactions occur simultaneously between bus masters and targets**

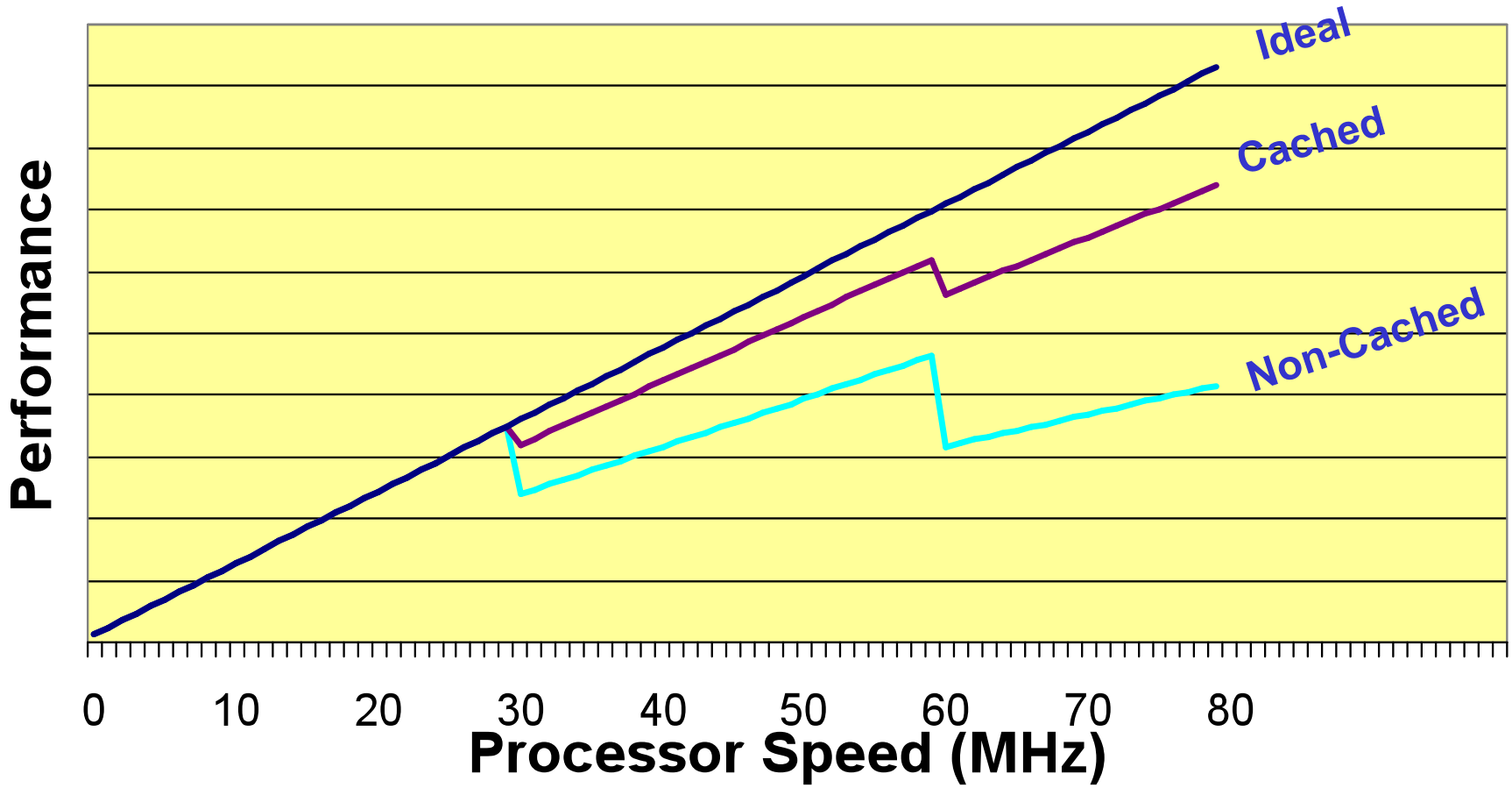
Instruction Prefetch and Cache

- Fetches 128-bit data
  - 4x32-bit instructions
  - 8x16-bit instructions
- Up to 80 MIPS of linear execution
- 256 Bytes Cache
  - 16 total lines
  - Up to 4 data lines
  - Lockable Lines



MICROCHIP
Regional Training Centers

Effect of Prefetch Cache



MICROCHIP
Regional Training Centers

New Concept
Co-processor 0

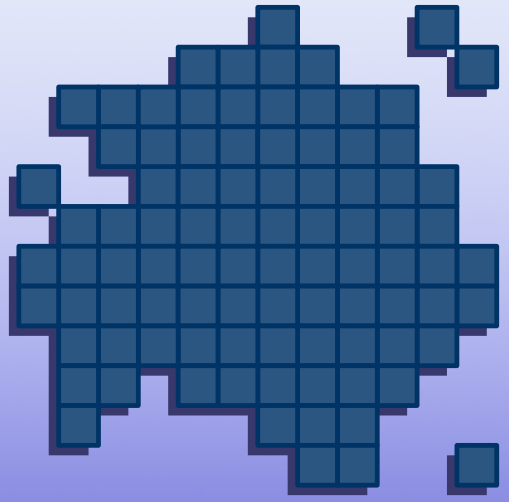
Coprocessor 0, called the System Control Processor, provides support for memory mapping and exception handling functions.

- Every MIPS processor implements CP0
- CP0's registers are split into two categories:
  - General - can be loaded from one of the processor's registers, or directly from memory
  - Control - must be loaded directly from a processor register
- All of its registers are treated like control registers
  i.e. they cannot be loaded or stored directly from system memory

MICROCHIP
Regional Training Centers

CPU Features visible through Coprocessor 0

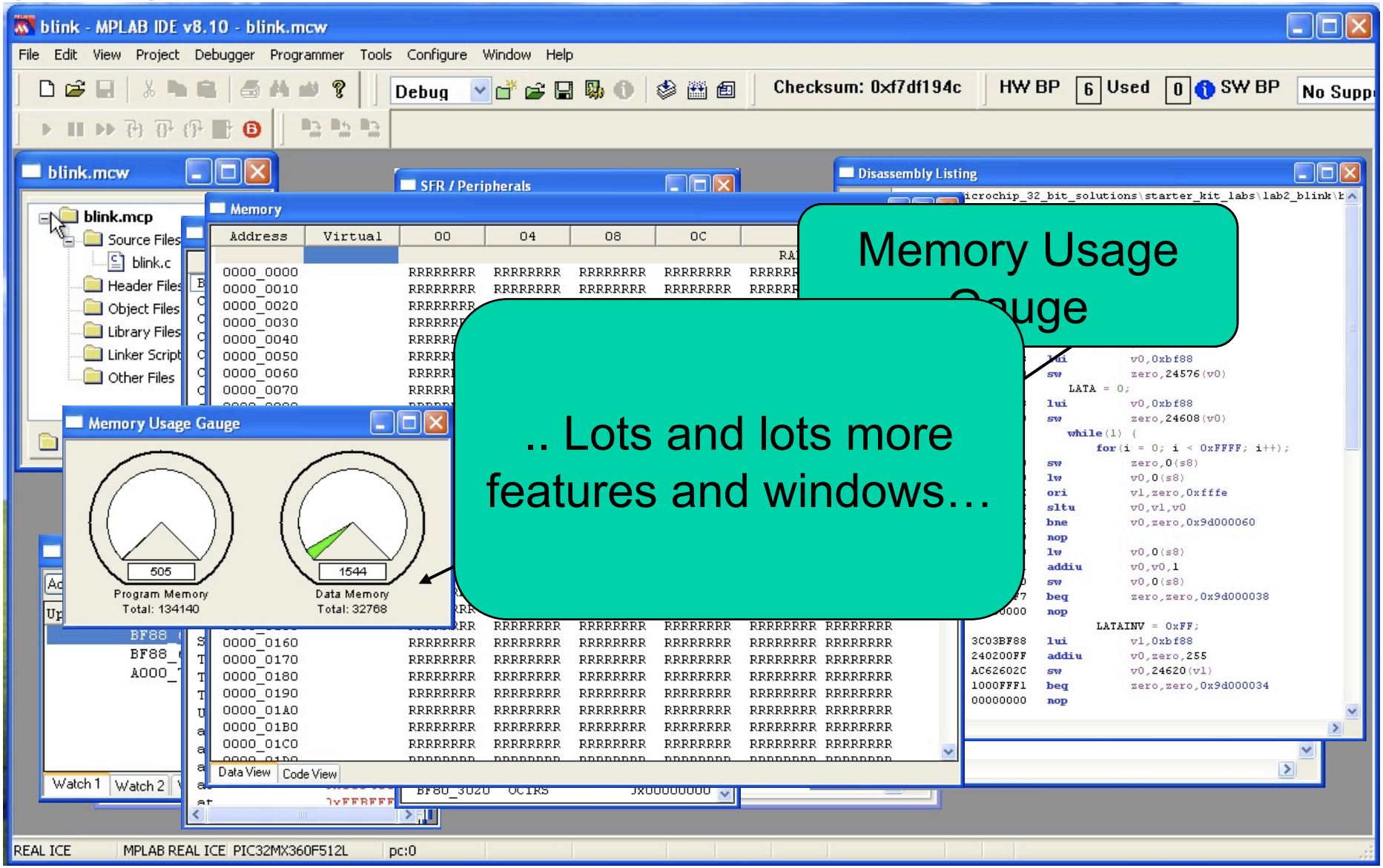
- CPU Configuration and Status
- Cache Control
- Core Timer
- Virtual memory configuration
- Shadow register set control
- Debugger control
- Exception and Interrupt Control



MPLAB IDE & MPLAB C32

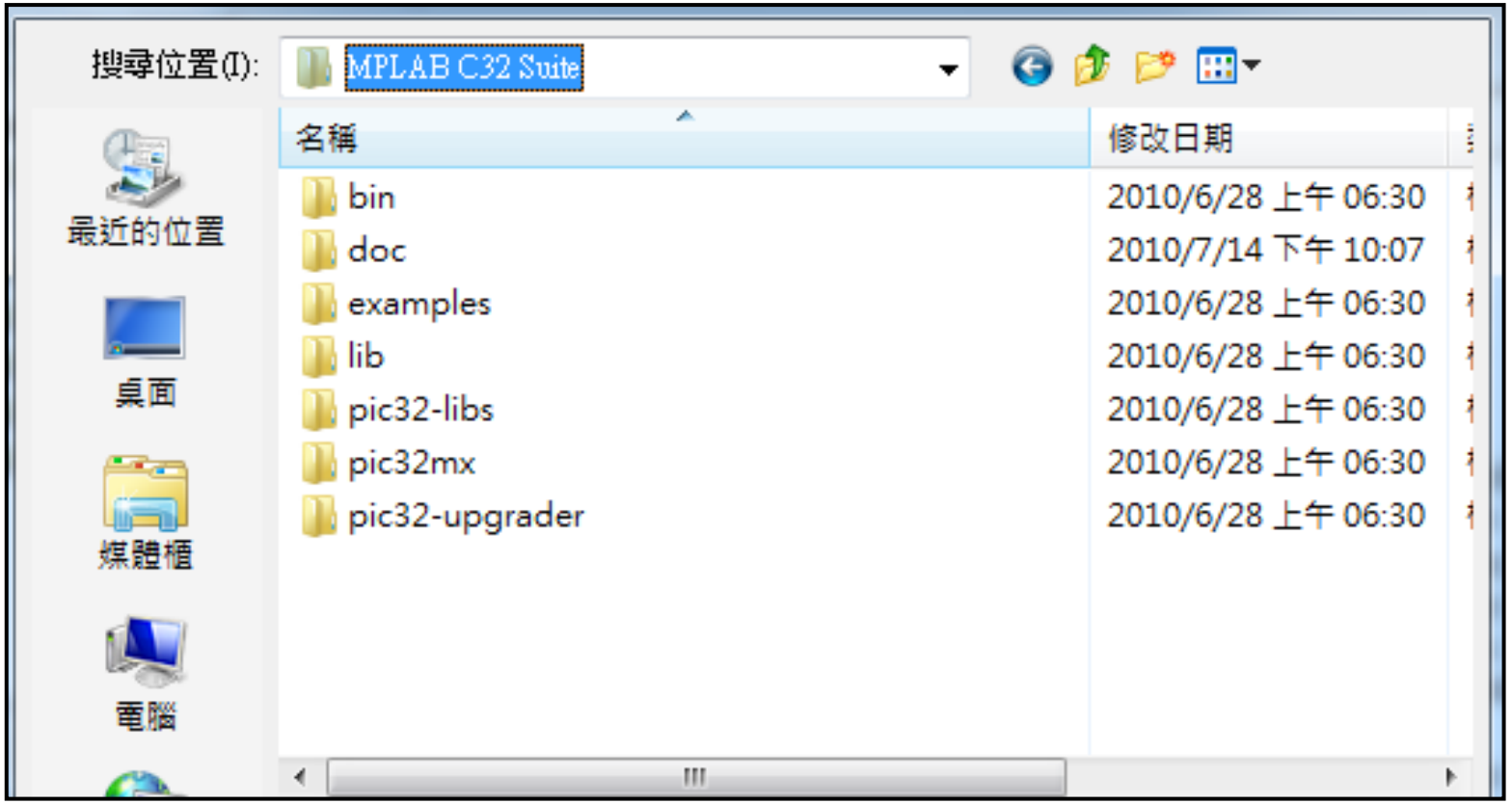
MICROCHIP
Regional Training

MPLAB IDE



MPLAB C32 的安裝

- 安裝新版的 **MPLAB IDE** 即一併地會將 **MPLAB C32** 安裝於 **C:\Program Files\Microchip\MPLAB C32 Suite**



MICROCHIP
Regional Training Centers

MPLAB C32 支援的資料型別(1)

- 整數資料型別

Type	Bits	Min	Max
char, signed char	8	-128	127
unsigned char	8	0	255
short, signed short	16	-32768	32767
unsigned short	16	0	65535
int, signed int, long, signed long	32	-2^{31}	$2^{31}-1$
unsigned int, unsigned long	32	0	$2^{32}-1$
long long, signed long long	64	-2^{63}	$2^{63}-1$
unsigned long long	64	0	$2^{64}-1$

MPLAB C32 支援的資料型別(2)

- 浮點數資料型別

The compiler uses the IEEE-754 floating-point format. Detail regarding the implementation limits is available to a translation unit in `float.h`.

Type	Bits
`float`	32
`double`	64
`long double`	64

MPLAB C32 對 PIC32 的 support

- 與 **MPLAB C30** 一樣，所有的 **PIC32** 使用相同的 **.h** 檔, **Compiler time** 會自行判斷並載入所需的 **MCU header file**
  - **#include <p32xxxx.h>**

- 要使用 **MPLAB C32** 所提供的 **Peripheral Library**，只要 **include** 一個 **.h** 檔
  - **#include <plib.h>**

MICROCHIP
Regional Training Centers

p32xxxx.h 檔案

- **MCU header file ：p32xxxx.h**
- **MPLAB IDE 會依據 User 所選用的 PIC32 的型號傳遞給 C32 編譯器，P32xxxx.h 會依照所傳遞的資訊找出該型號的 Header file 並加以引用 。**
- **Header File 檔案路徑：**
  - **C:\Program Files\Microchip\MPLAB C32\pic32mx\ include\P32xxxx.h**
- 判斷與引用檔案的範例：

```
#elif defined(__32MX795F512L__)
#include <proc/p32mx795f512l.h>
```

MICROCHIP
Regional Training Centers

plib.h 檔案

- **MPLAB C32 Peripheral Library** 提供許多個別的 **header file** 來為各周邊函式定義相關 引數、參數名稱。
- **plib.h** 會將 **PIC32** 所有周邊函式的 **header files** 一併加入。
- **plib.h** 檔案路徑：
  - **C:\Program Files\Microchip\MPLAB C32\pic32mx\include\plib.h**
- **plib.h** 實際的內容如下:

```
#include <peripheral/adc10.h>
#include <peripheral/bmx.h>
#include <peripheral/cmp.h>
#include <peripheral/cvref.h>
#include <peripheral/dma.h>
#include <peripheral/i2c.h>
:
```

MICROCHIP
Regional Training Centers

MPLAB C32 對 Configuration 的支援

- 利用 **#pragma config** 在程式裡來設定
- **config** 的有效定義字需參考:

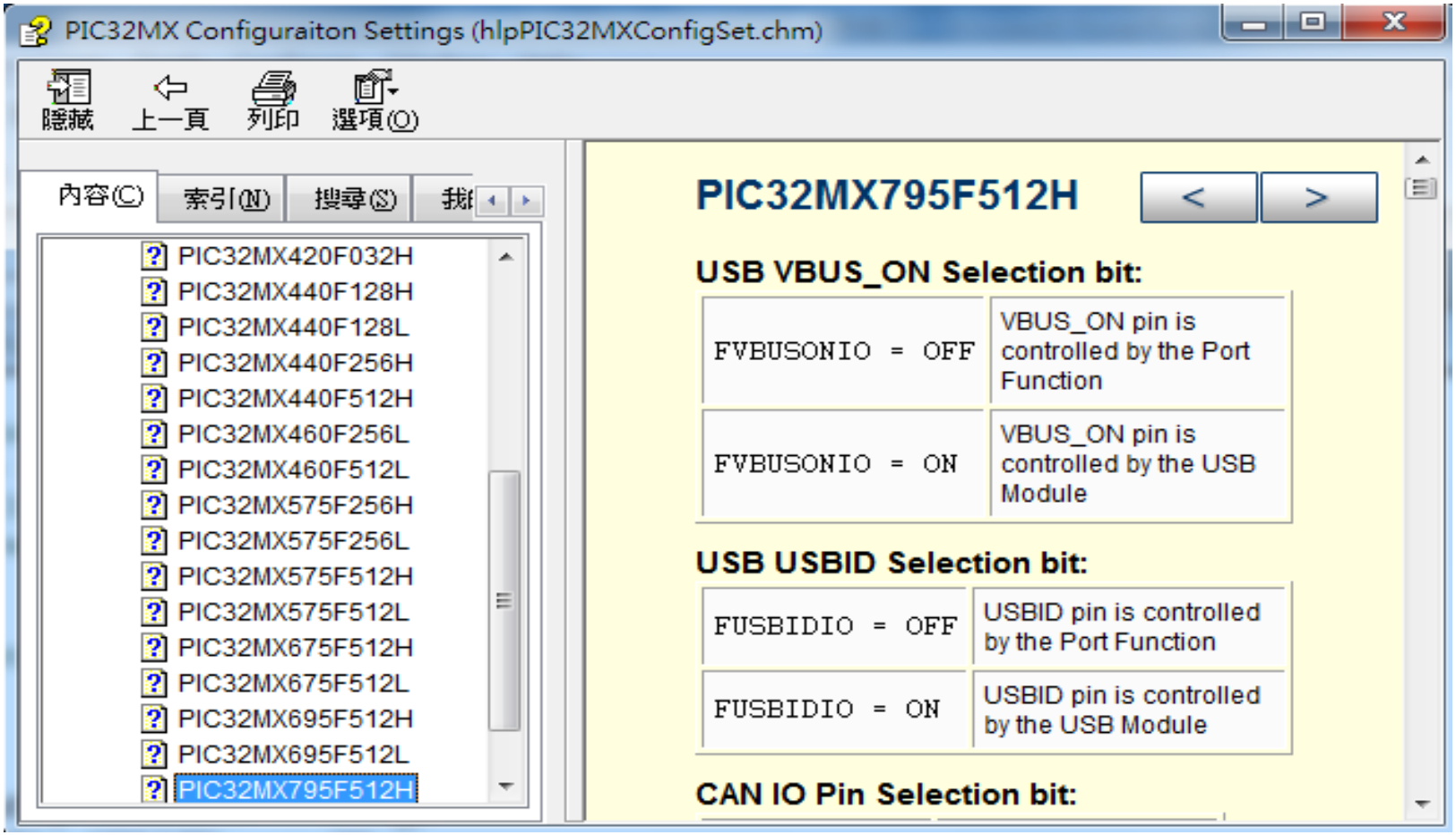
  C:\Program Files\Microchip\MPLAB C32\doc\hlpPIC32MXConfigSet.chm

  - **#pragma config ICESEL = ICS_PGx2**
  - **#pragma config FPLLMUL = MUL_15, FPLLIDIV = DIV_2, FPLLODIV = DIV_1**

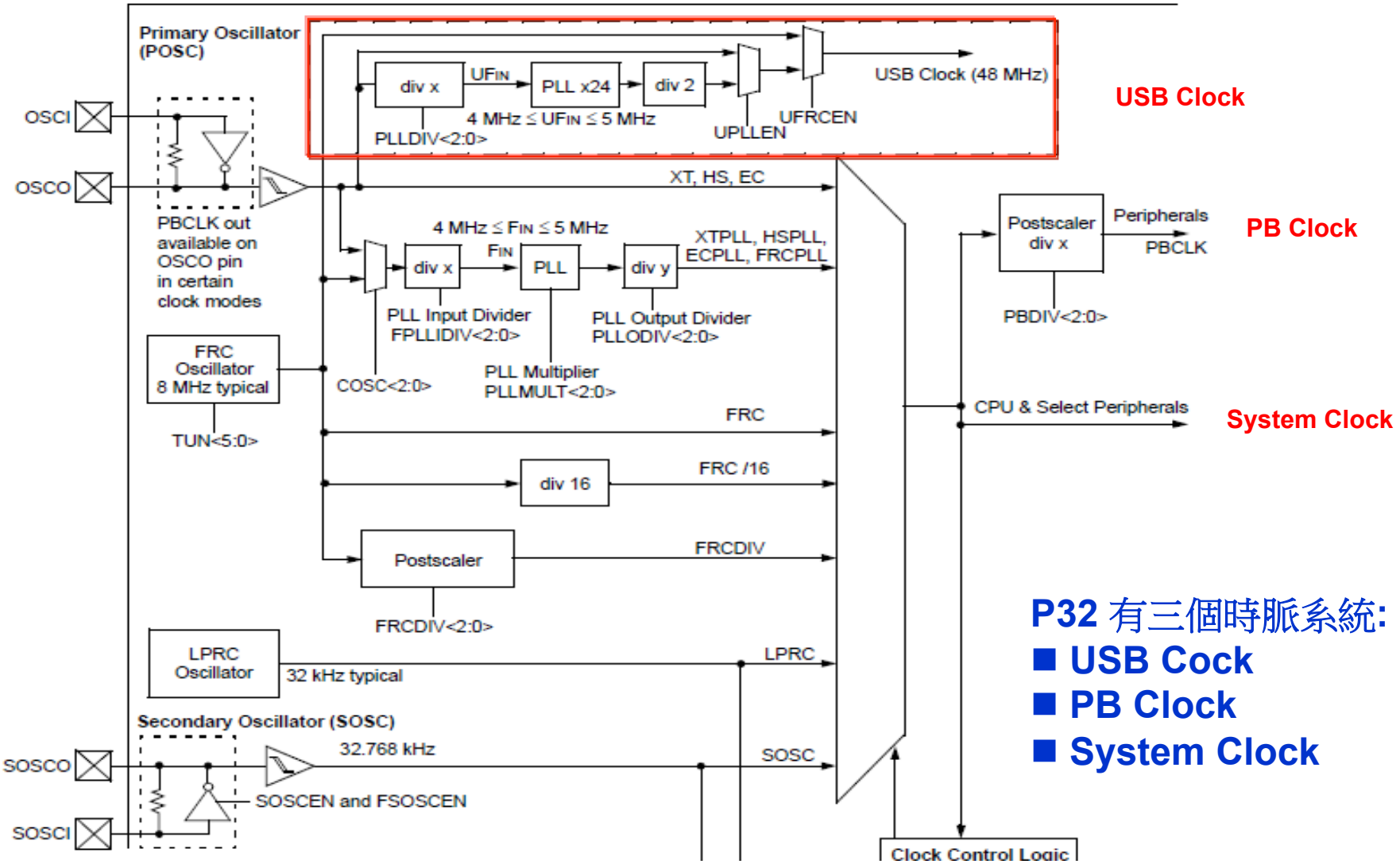
- 也可以在 **MPLAB IDE** 下直接修改所需的硬體配置

MPLAB IDE 中對 PIC32MX Configuration Setting 的說明

- 檔案 : **C:\Program Files\Microchip\MPLAB C32\doc\hlpPIC32MXConfigSet.chm**



PIC32 的 Oscillator Block
(Refer to Figure 4-1 of PIC32 Datasheet)



P32 有三個時脈系統:

- USB Cock
- PB Clock
- System Clock

MICROCHIP
Regional Training Centers

震盪模式的頻率範圍

- **Crystal 頻率與 Mode 的參考值**

Oscillator Mode	Description
HS	10 MHz-40 MHz crystal
XT	3.5 MHz-10 MHz resonator
EC	External clock input (0-80 MHz)
HSPLL	10 MHz-40 MHz crystal, PLL enabled
XTPLL	4 MHz-10 MHz resonator, PLL enabled
ECPLL	External clock input (5-80 MHz), PLL enabled

MICROCHIP
Regional Training Centers

PIC32 Oscillator 的 PLL 區塊

- 內部的 8Mhz FRC 以及外加的 Primary Osc. 都可以經由 PLL 倍頻
- PLL 有 3 個主要的部份
  - PLL 的 輸入除頻區塊 (4Mhz < 結果< 5Mhz)
  - PLL 的倍率調整區塊 (可將 FIN 乘上 15 ~ 24 倍)
  - PLL 的輸出除頻區塊 (可除以 1 ~ 256 後再輸出)

APP1632 使用 8MHz 石英晶體，實驗時設定使用 60MHz 的執行速度，所以所做的設定為:

(8MHz / 2) x 15 倍頻再除以 1 = 60MHz

MICROCHIP
Regional Training Centers

PIC32 的 PBCLK 與 ICD Pins

- PIC32 的 PBCLK 可以選取幾種不同的速度
  - SYSCLK/1
  - SYSCLK/2
  - SYSCLK/4
  - SYSCLK/8
  - 可以由 MCU 的 Configuration Bits 選項設定或程式中動態地更改 OSCCON<20:19> (PBDIV<1:0>) 來改變比例

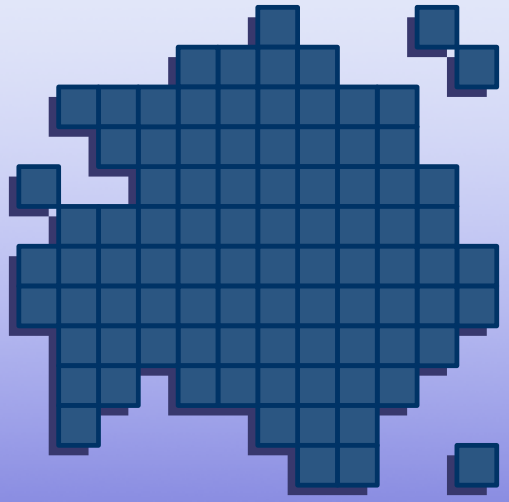
- PIC32 的 Device 有兩組 ICD/ICSP 的接腳
  - PGEC1/PGED1
  - PGEC2/PGED2 (APP1632使用這組來燒錄與除錯)
  - 這兩組接腳都可以用來燒錄 PIC32，且自動偵測。
  - 除錯接腳的連接必須與 Configuration Setting 中的設定一致 。

MICROCHIP
Regional Training Centers

MPLAB C32 的基本程式

- 語法正確，但未加入任何功能的程式範本 …
  - 以 **APP1632** 外加 **8Mhz Crystal** 為例

```
#include              <p32xxxx.h>
#include  <plib.h>
#pragma config UPLLIDIV = DIV_2   // 設定USB PLL 的輸入為 4MHz 輸入頻率 ( 8Mhz/2)
#pragma config UPLLEN   = ON      // 開啟 USB PLL 得到 48MHz 輸出
#pragma config FPLLMUL = MUL_15, FPLLIDIV = DIV_2, FPLLODIV = DIV_1
// 輸入為 8MHz /4 後再做 15 倍頻後輸出 60MHz 的 System Clock
#pragma config POSCMOD = HS, FSOSCEN = OFF , FNOSC = PRIPLL , FPBDIV = DIV_1
#pragma config FCKSM = CSECMD,IESO = ON , FWDTEN = OFF
#pragma config ICESEL = ICS_PGx2   // 選用第二組來除錯

int main (void)
{
    while(1) ;
}
```



學習 PIC32 可用的實驗板
(參考資料)

MICROCHIP
Regional Training Centers

最簡易使用的 Starter Kit
內建 Starter Kit 專用的 Debugger/Programmer

PIC32 Ethernet Starter Kit

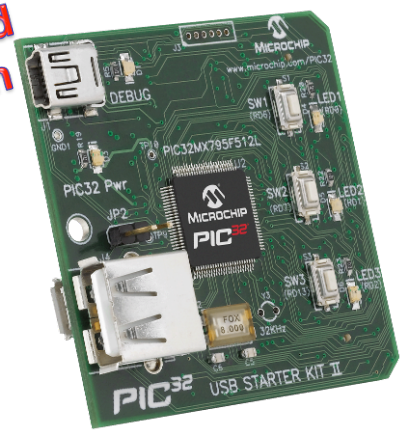
NEW!

(DM320004, $72)

Same expansion Connector for Application Specific Development

PIC32 USB Starter Kit II

Updated Design



(DM320003-2, $55)

Ethernet Starter Kit

- Ready for an Ethernet Network
- National DP83848 PHY & RJ45
- USB Host/Device/OTG
- Works with I/O Expansion Board (DM320002)

Both Starter Kits

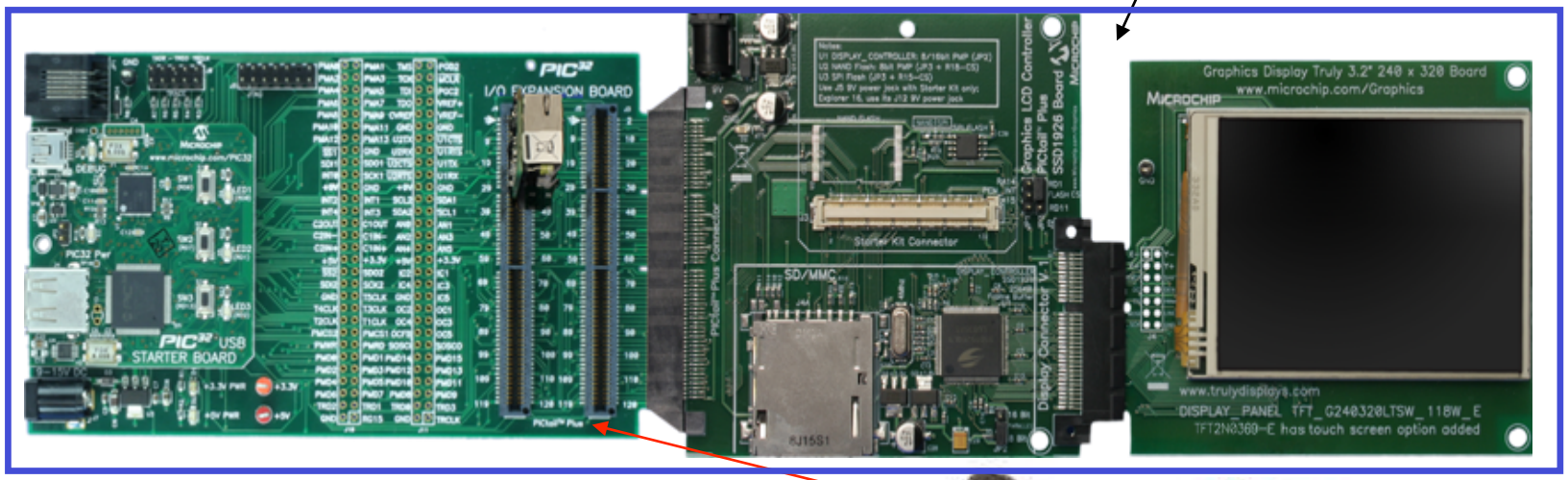
- PIC32MX795F512L (512k/128k)
- Upgraded debug chip to PIC32 QFN
- Upgraded USB host power supply
- Everything else the same

PIC32 I/O Expansion Board

PIC32 I/O Expansion Board

DM320002	$72

Add Optional PICtail™ Plus Daughter Boards



- Customize PIC32 Starterkit platform
- Easy access to every signal
- Add feature specific boards
  - Connect Microchip Tools
  - Connect JTAG & Trace

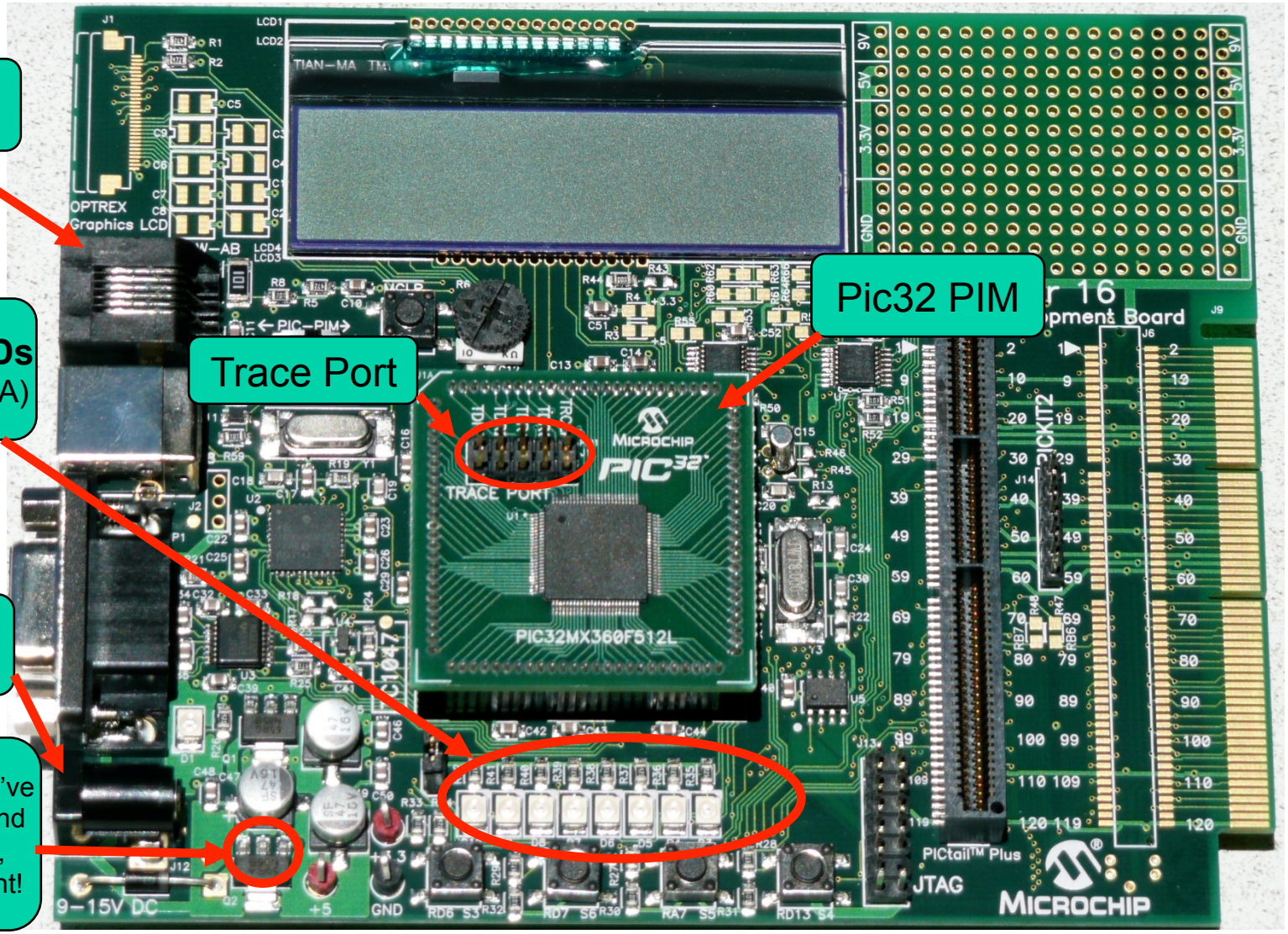
(shown with PIC32 USB Starter Board)

Popular PICtail Plus Daughter Boards

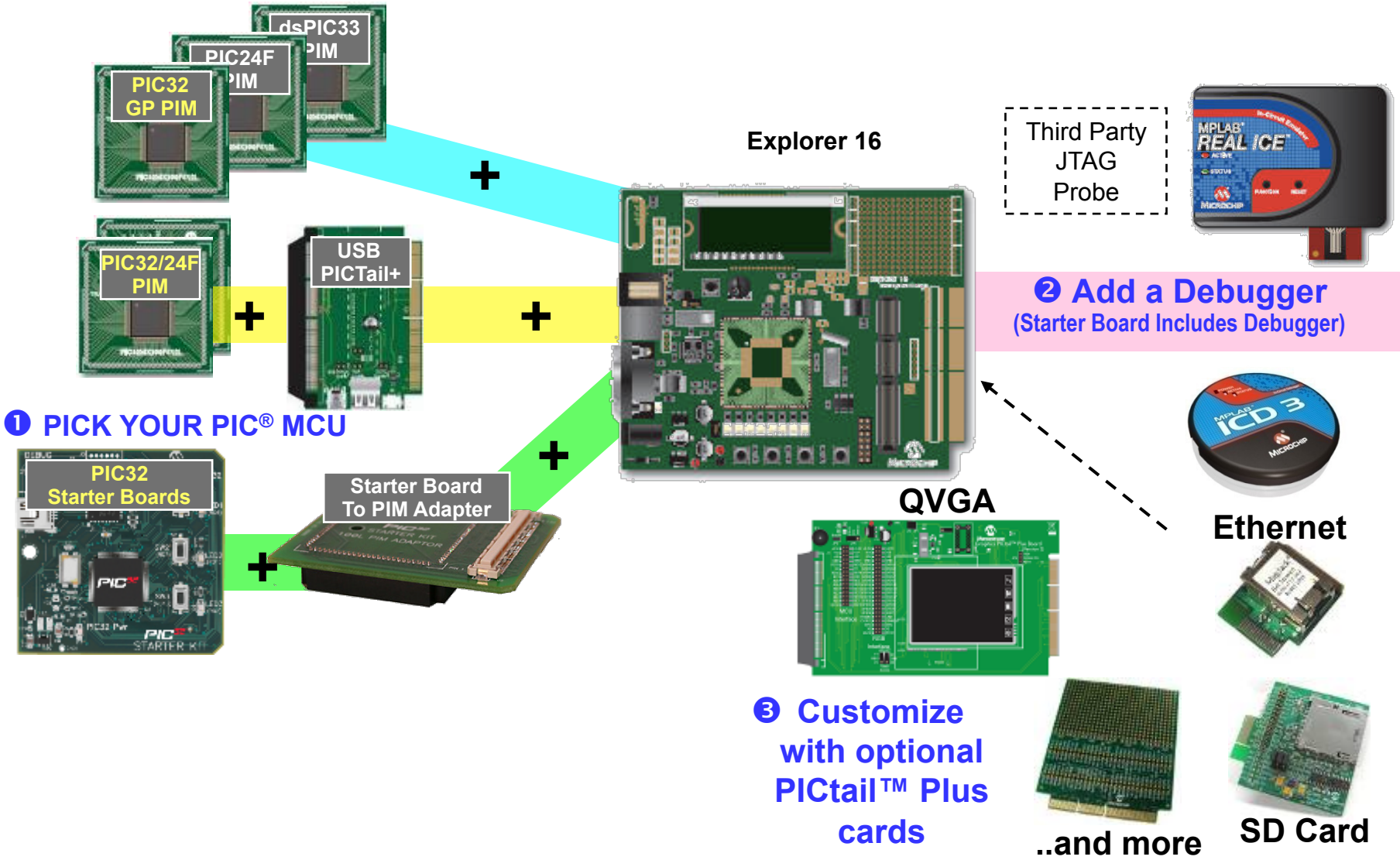
Graphics	SD Card
Ethernet 802.15.4	CAN 802.11

MICROCHIP
Regional Training Centers

Explorer16 Development Board



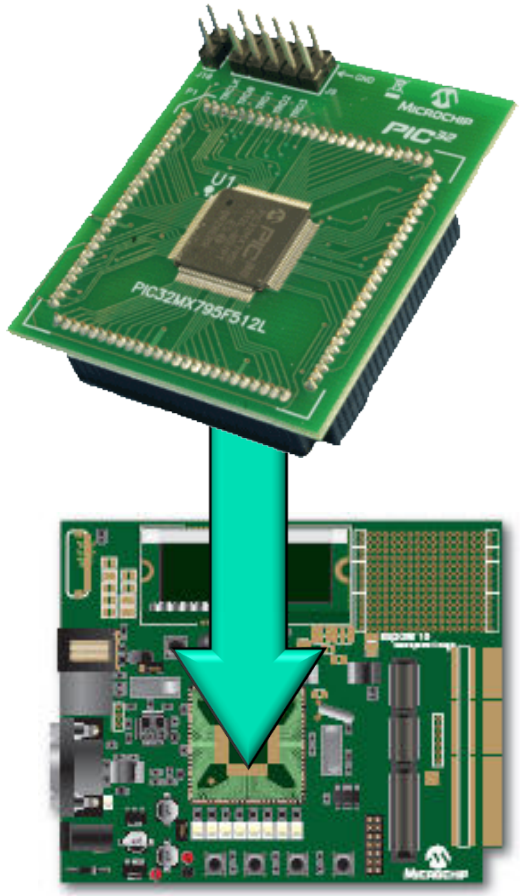
Explorer 16 Compatibility Overview



MICROCHIP
Regional Training Centers

PIC32 PIM Modules and Other Accessories to Explorer16 Board

Plug-in Modules	Part No
PIC32MX3 100P QFP TO 100P PLUG IN MODULE	MA320001
PIC32MX4 USB PIM	MA320002
PIC32MX7 CAN PIM	MA320002
ECAN/LIN PICTail+ Daughter Card	AC164130
USB PICTAIL+ Daughter Card for Explorer 16	AC164131
PIC32 Starter Board to Explorer 16 PIM Adapter	AC320002

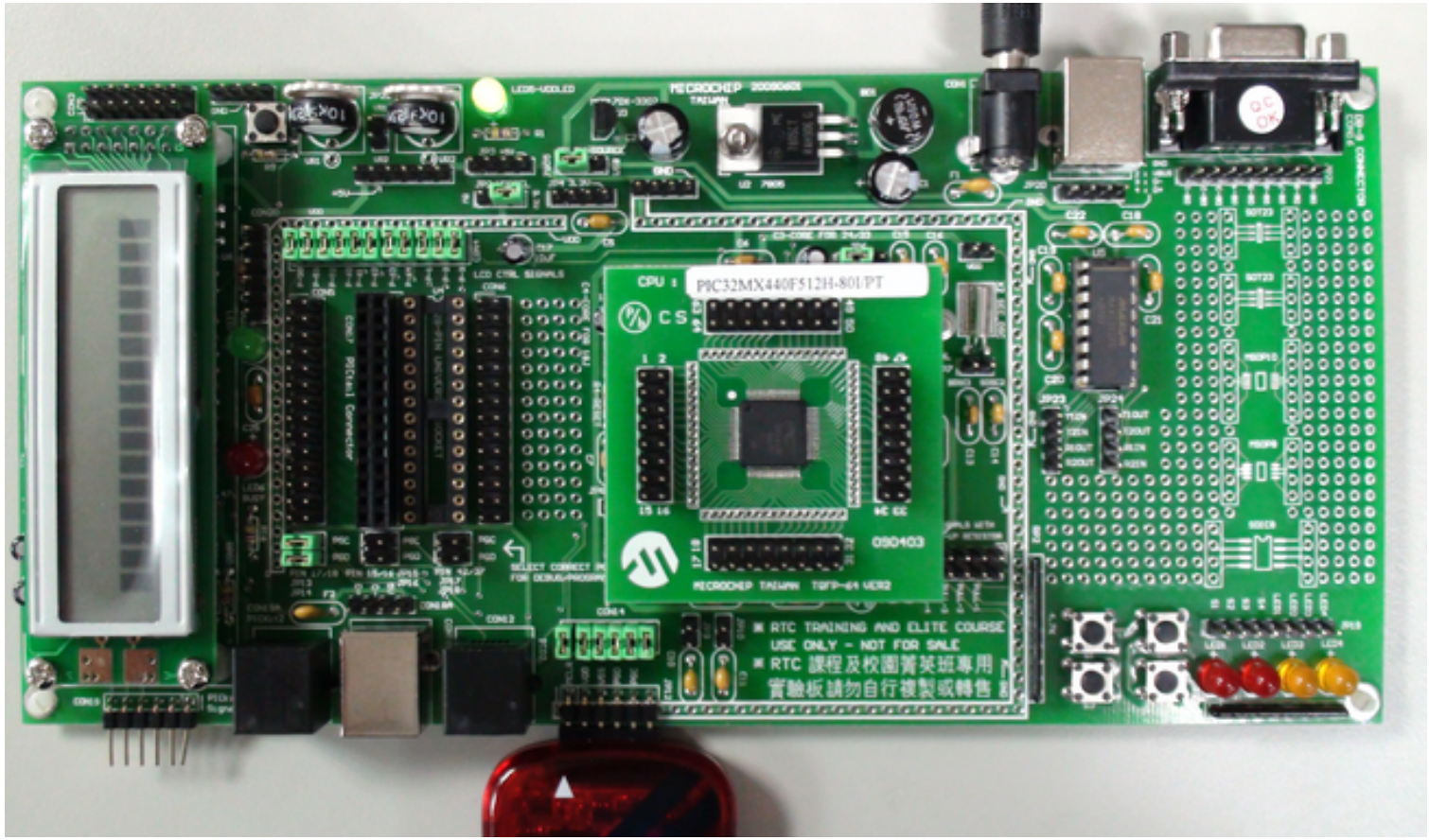


Adding PIM for Explorer 16

MICROCHIP
Regional Training Centers

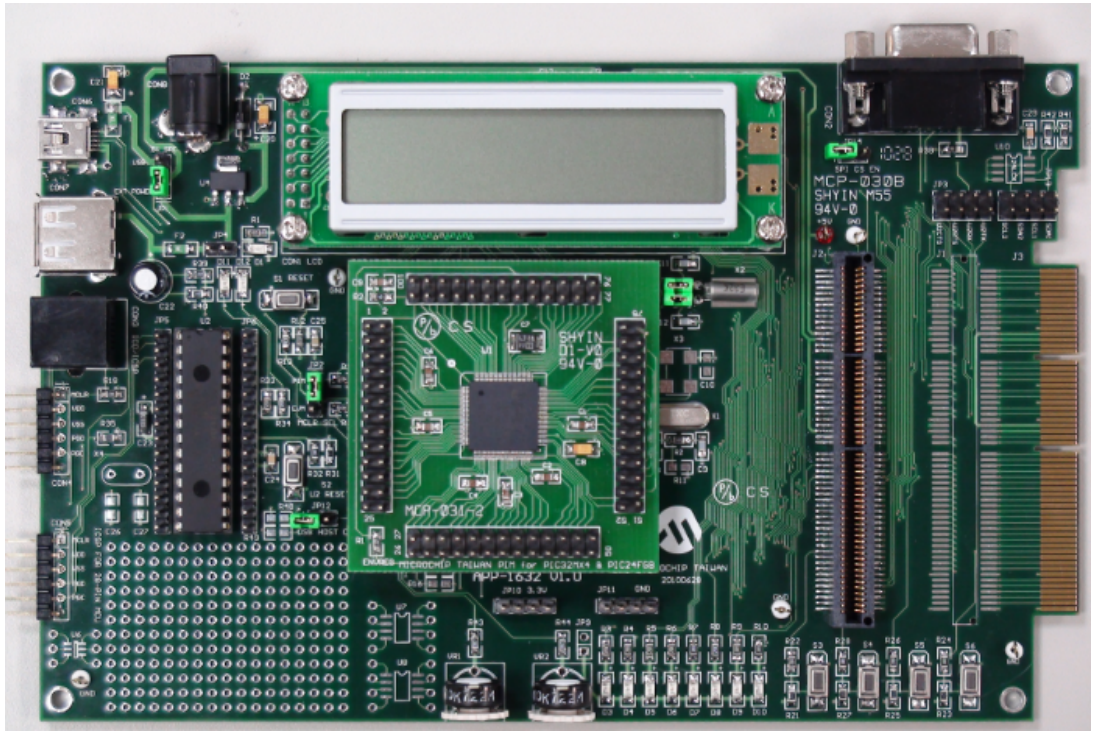
使用 APP026-3X + PIC32 PIM module

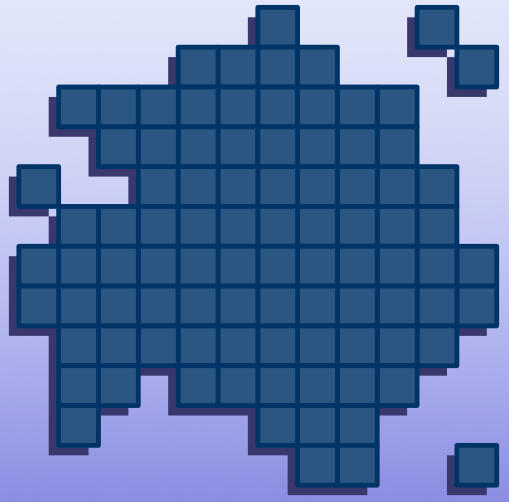
- **APP026-3X** 為歷年菁英班使用的 **64-pin** 泛用實驗板
- 可以自行用隨附空板焊上 **64-pin PIC32 MCU** 來使用



APP1632 實驗板
32-Bit MCU 專用實驗板

- APP1632 為 Microchip Taiwan 推出專為 Microchip 16/32 bit MCU 所設計之實驗板
- 基本電路與 Explorer-16 相容，可以直接載入原本為 Explorer-16 所設計的範例程式
- PICtail Plus !
- USB Connectors
- 2nd MCU
- 2.54mm CPU PIM



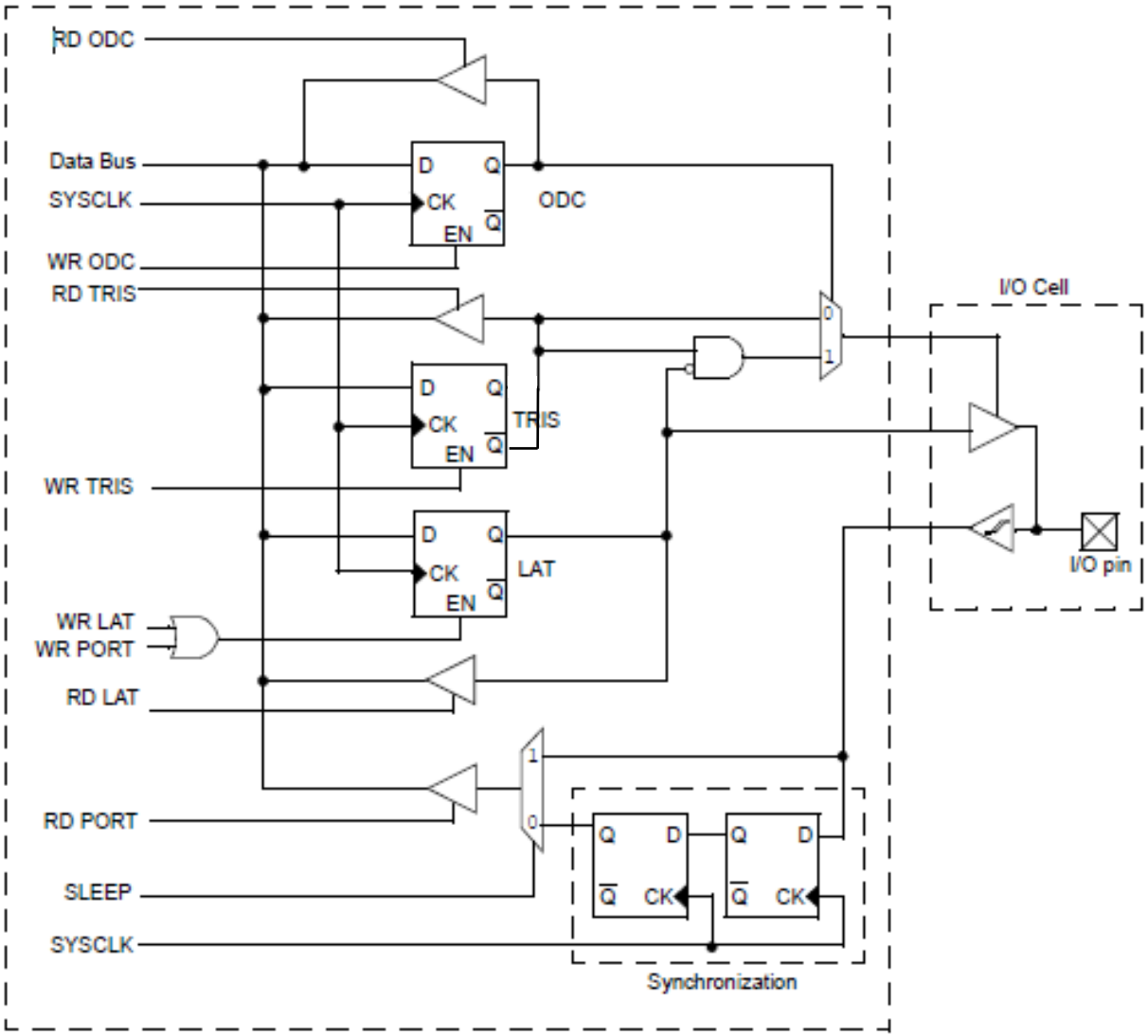


I/O Port

請參考 PIC32 Family Reference Manual – 第 12 章

MICROCHIP
Regional Training Centers

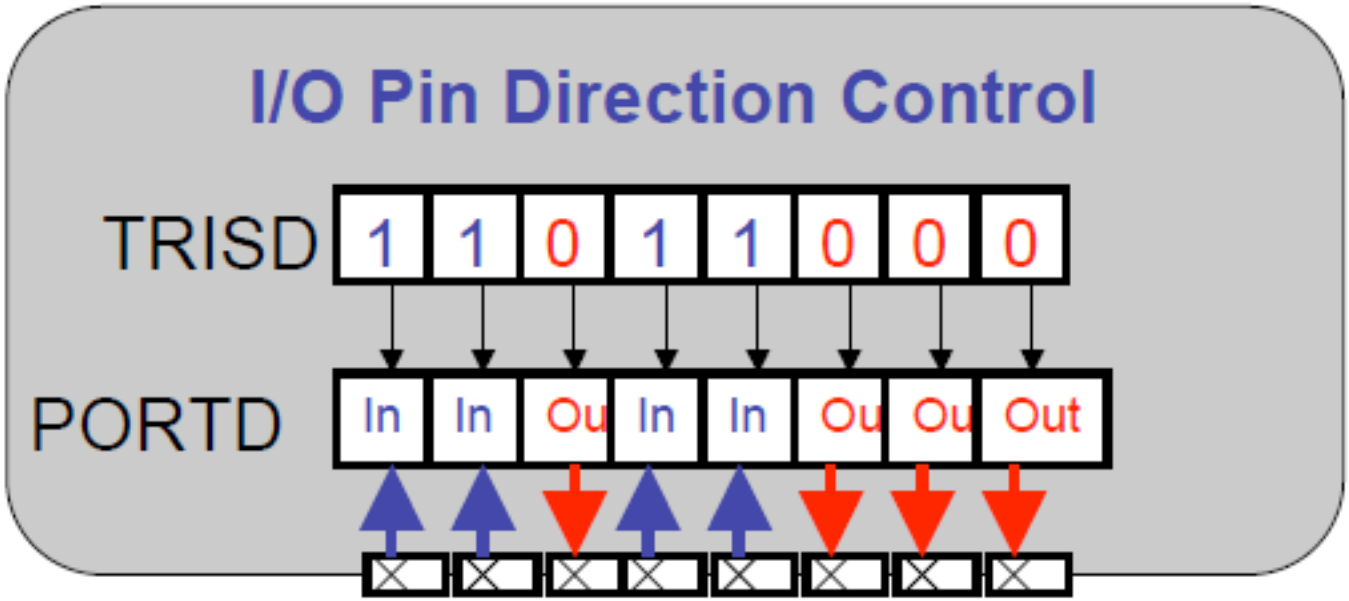
I/O Port 的基本架構



MICROCHIP
Regional Training Centers

I/O Port 輸出入的設定

- PIC32 的 I/O 腳位的基本操作與 PIC24，PIC18 方式相容
- TRISx 暫存器為輸出或輸入控制



MICROCHIP
Regional Training Centers

基本的 I/O Port 控制

- **TRISx：輸入、輸出設定**
- **LATx：PORT的輸出 Latch 暫存器**
- **PORTx：**
  - 當輸出時，與 **LATx** 一樣的功能
  - 當輸入時，讀取外部接腳的實際輸入

- **I/O Port 控制暫存器 (以PORTD 為例)：**
  - TRSID = 0x00FF; //RD0~RD7 輸入，RD8~RD15 為輸出
  - LATD = 0x5500; // 輸出 0x55 到 RD8~RD15
  - Var1 = 0x00ff & PORTD ;
- 對單支 **I/O** 腳的控制 **(使用 MPLAB C32 定義的位元結構):**
  - TRISDbits.TRISD0 = 0 ; // RD0 為輸出腳
  - LATDbits.LATD0 = 1 ; // RD0 輸出 Hi
  - While (PORTDbits.RD1); // 檢查 RD1 狀態

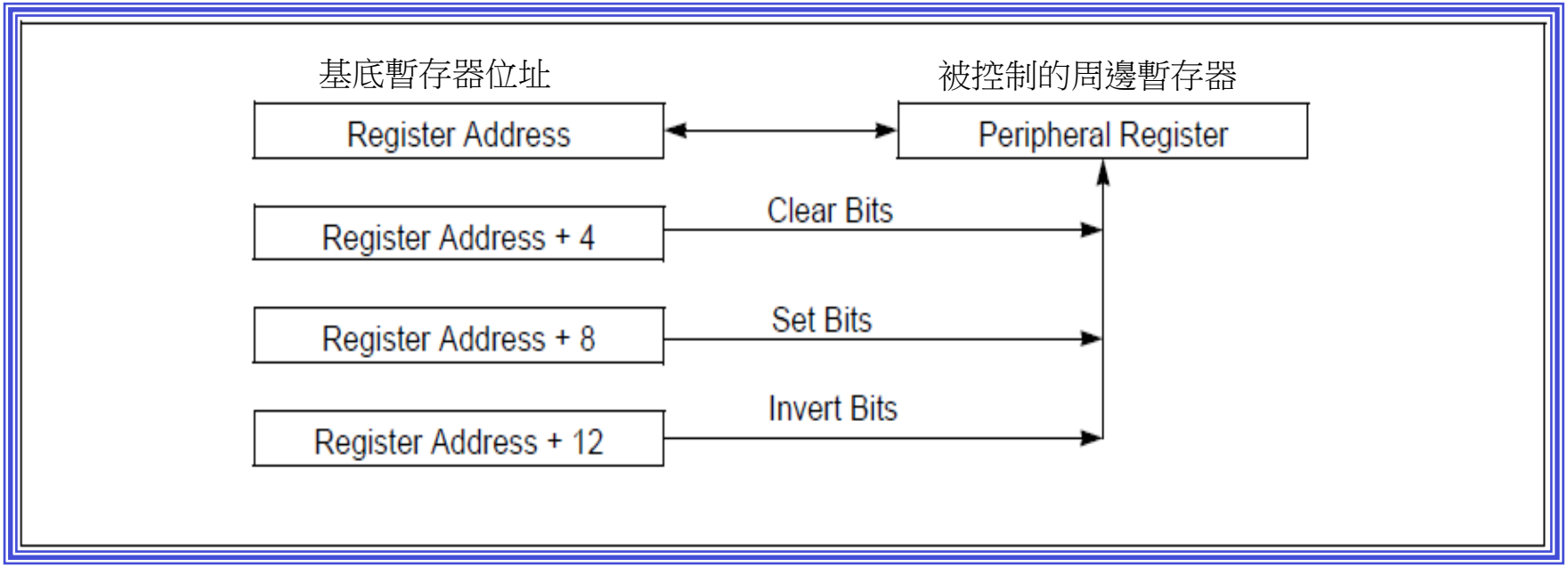
MICROCHIP
Regional Training Centers

PIC32 新加入的位元操作功能
Atomic bit manipulation

- 使用特殊的控制暫存器做大量的位元控制
- 每個**SFR** 皆搭配有 **3** 個使用於位元操作的暫存器以提供三種不同的操作模式
- **3** 種操作方式為：
  - Set Bits
  - Clear Bits
  - Invert Bits
- 針對所要操作的位元採用 **MASK** 的控制方式
- 一般的 **RAM** 不具有此項操作功能

SET/CLEAR/INVERT 的操作

- 基底的暫存器位址提供一般的 **R/W** 操作
- 所搭配的多位元操作暫存器只提供特殊的寫入功能
- **SET/CLEAR/INVERT** 暫存器的多 位元操作只需一個指令週期！



MICROCHIP
Regional Training Centers

CLR / SET / INV 暫存器位址說明

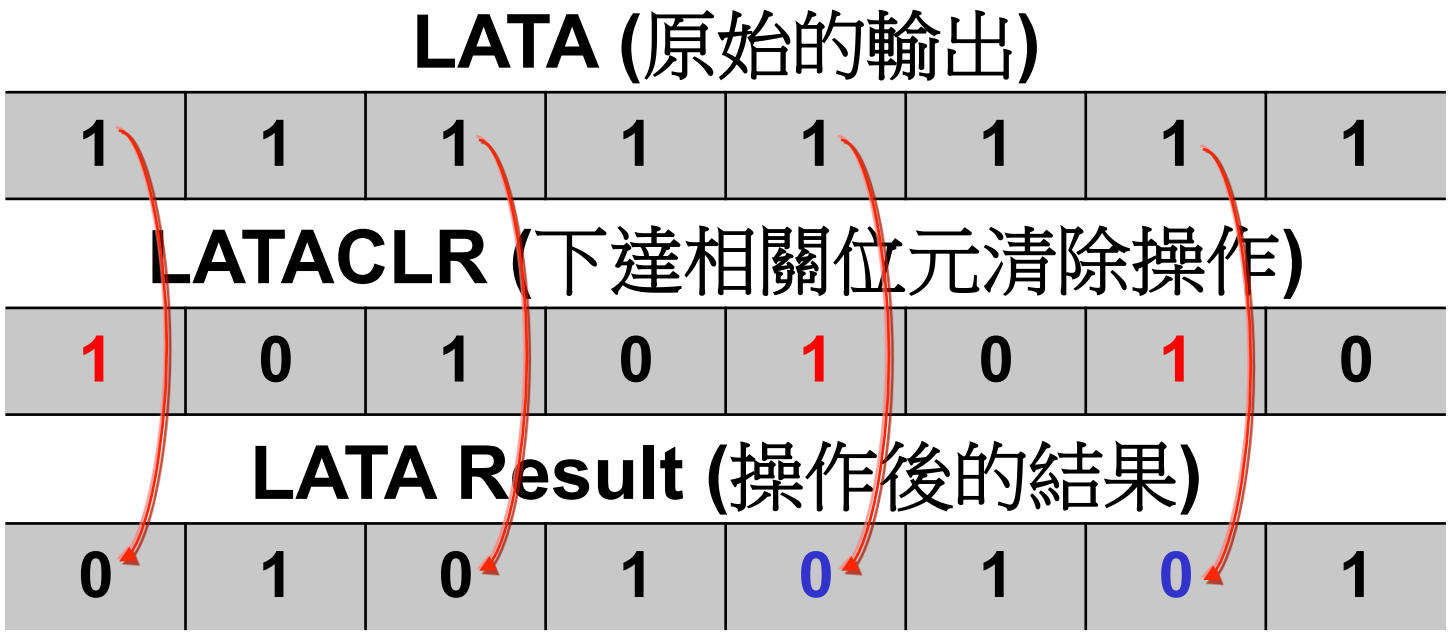
- **Example of TRISD**

Virtual Address	Name		Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
BF88_60C0	TRISD	31:24	—	—	—	—	—	—	—	—
		23:16	—	—	—	—	—	—	—	—
		15:8	TRISD15[1]	TRISD14[1]	TRISD13[1]	TRISD12[1]	TRISD[11:8]			
		7:0	TRISD[7:0]							
BF88_60C4	TRISDCLR	31:0	Write clears selected bits in TRISD, Read yields undefined							
BF88_60C8	TRISDSET	31:0	Write sets selected bits in TRISD, Read yields undefined							
BF88_60CC	TRISDINV	31:0	Write inverts selected bits in TRISD, Read yields undefined							

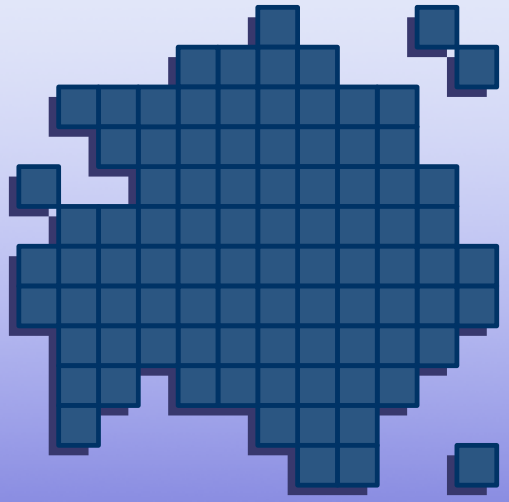
- Most peripheral registers can be accessed at 4 addresses
  - Base address — Reads OK. Writes all 0's and 1's to dest
  - Base+4 — No Reads. Clears bits in dest that are written with 1
  - Base+8 — No Reads. Sets bits in dest that are written with 1
  - Base+12 — No Reads. Inverts bits in dest that are written with 1
- *Bits that are written as 0 at Clear/Set/Invert addresses DO NOT MODIFY the destination register*

MICROCHIP
Regional Training Centers

以 PORTA 為例

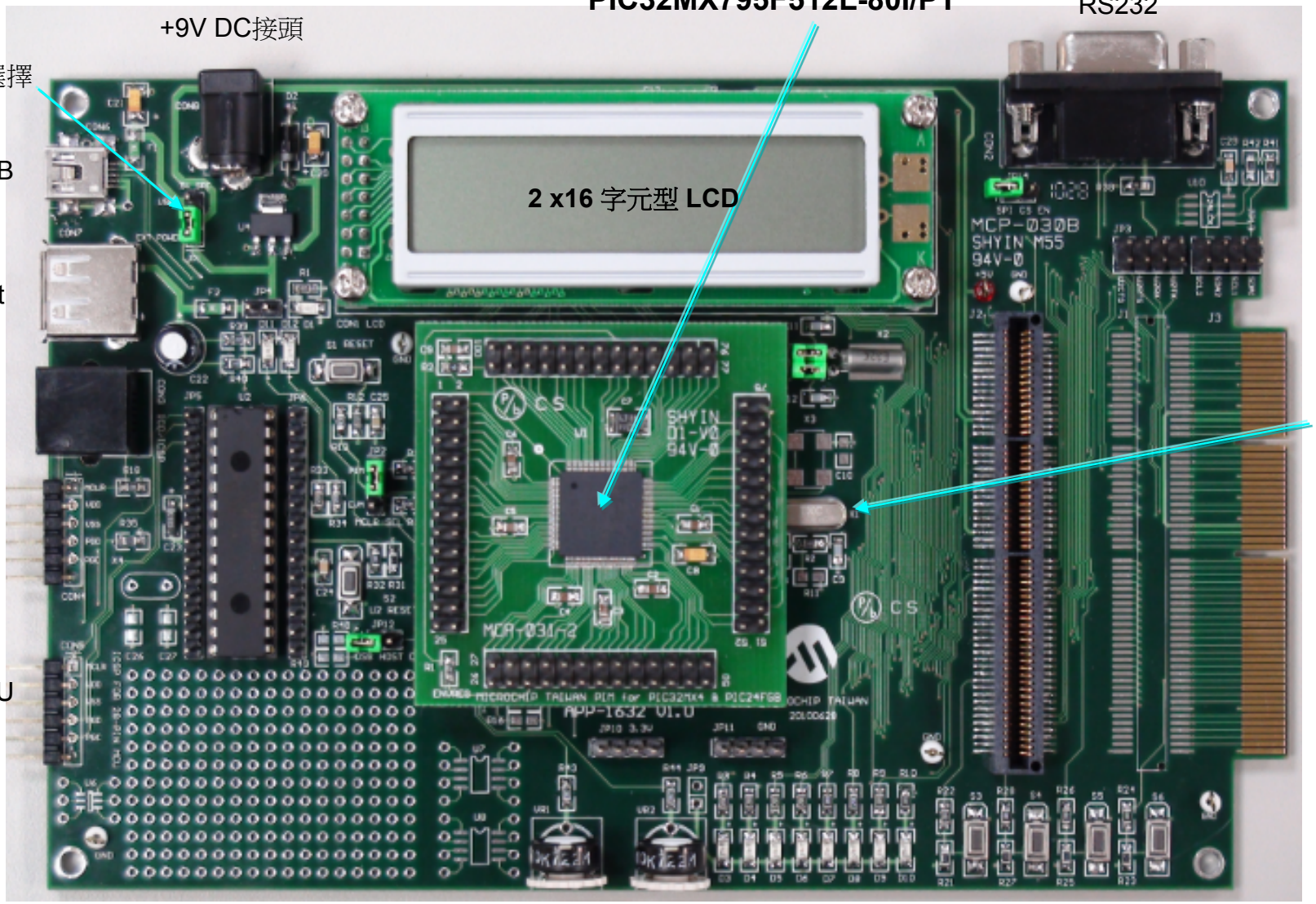


對於 LATACLR 暫存器的操作將會把被該暫存器 Mask 的位元清除為”0”



練習前準備 ……

APP1632 實驗版



MICROCHIP
Regional Training Centers

C32 重要參考資料 - 1 (.pdf 檔案)

- APP1632 使用手冊：
  - APP1632 使用說明書 (TW-UG-APP1632-C).pdf
- C32 周邊函數庫使用手冊:
  - 32-bit-Peripheral-Library-Guide.pdf
- C32 編譯器使用手冊：
  - MPLAB C32 User Guide.pdf
- C32 標準函數庫使用手冊
  - MPLAB C32 Libraries.pdf

以上參考資料皆可在 **32-bit RTC Workshop** 資料光碟找到

MICROCHIP
Regional Training Centers

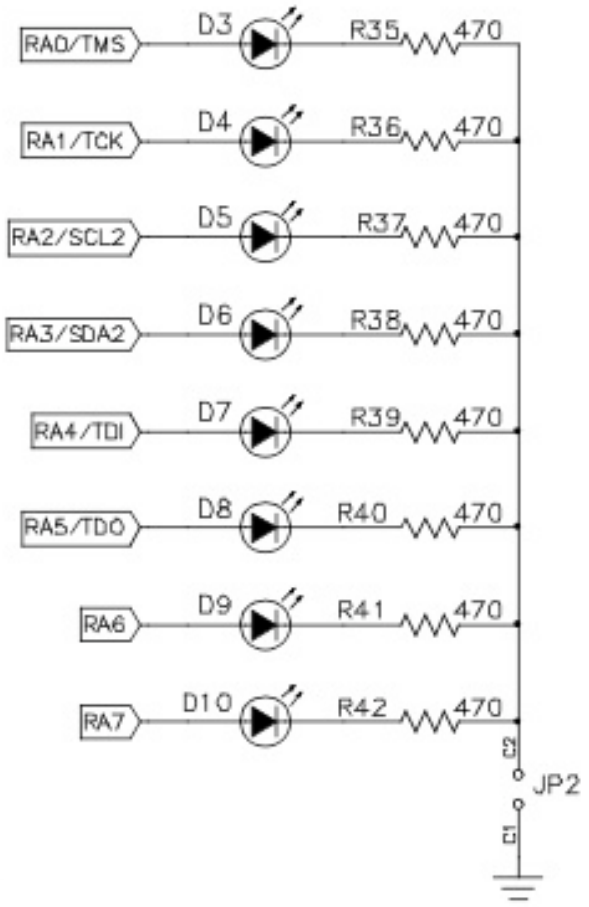
C32 重要參考資料 - 2 (已編譯的HTML說明檔)

- C32 在安裝完成後也有提供 html 格式的使用手冊：
  - C:\Program Files\Microchip\MPLAB C32 Suite\doc
- 這些 html 檔案也可以在 MPLAB IDE 下的 Help 選項下的 Topics → C32 Compiler 裡找到。

使用 PORTA 控制 LEDs

若使用 APP1632 搭配 PIC32 的 I/O做 LED 的控制 , 使用時需記住如右圖所示的 LED 連接 .

APP1632 的 LEDs 被連接至 PIC32 的 PORTA<RA7:RA0> !



PORTA 的許多接腳與 JTAG 介面是共用的 , 所以要將 JTAG Disable後才可以控制 LEDs !

DDPCONbits.JTAGEN = 0;

** main() 函式的第一行即執行此功能

MICROCHIP
Regional Training Centers

C32 對 I/O Port 的支援 (一)

- **PORT 的函式支援**
  - 開啟 **32-bit-Peripheral-Library-Guide.pdf** 檔案，查詢 **10.0 I/O PORT LIBRARY** 裡的函式用法

PORTSetPinsDigitalIn( )
PORTSetPinsDigitalOut( )
PORTSetPinsAnalogIn( )
PORTSetPinsAnalogOut( )
PORTRead( )
PORTReadBits( )
PORTWrite( )
PORTSetBits( )
PORTClearBits( )
PORTToggleBits( )

MICROCHIP
Regional Training Centers

C32 對 I/O Port 的支援 (二)

- **PORT 的巨集支援**
  - 一樣開啟 **32-bit-Peripheral-Library-Guide.pdf** 檔案，查詢 **10.0 I/O PORT LIBRARY** 裡的巨集用法

mPORTASetPinsDigitalIn( ) ... mPORTGSetPinsDigitalIn( )
mPORTASetPinsDigitalOut( ) ... mPORTGSetPinsDigitalOut( )
mPORTBSetPinsAnalogIn( )
mPORTBSetPinsAnalogOut( )
mPORTADirection( ) ... mPORTGDirection( )
mPORTARead( ) ... mPORTGRead( )
mPORTAReadBits( ) ... mPORTGReadBits( )
mPORTAWrite( ) ... mPORTGWrite( )
mPORTASetBits( ) ... mPORTGSetBits( )

MICROCHIP
Regional Training Centers

使用 PORT 函式或巨集

- 客隨主便，只要會動就好；看起來使用巨集會比較好用。
  - 也可以直接使用 **Atomic Bit Manipulation**
- 使用範例：

```
mPORTAClearBits ( BIT_0 | BIT_1 | BIT_2 | BIT_3 | BIT_4 | BIT_5 | BIT_6 | BIT_7);
                                                    // Turn Off all LEDs
mPORTASetPinsDigitalOut ( BIT_0 | BIT_1 | BIT_2 | BIT_3 | BIT_4 | BIT_5 | \
                                  BIT_6 | BIT_7);          // RA0 ~ RA7 設為輸出推 LEDs

mPORTDSetPinsDigitalIn(BIT_6 | BIT_7);                    // RD6 & RD7 設定為按鍵輸入

PORTACLR = 0xff ;

PORTA = 0x00 ;
```

MICROCHIP
Regional Training Centers

開始做練習一 (Lab1)

- 32-bit MCU 課程使用裝設 PIC32MX795F512L MCU 的 APP1632
- APP1632 使用 8MHz 石英晶體作震盪源
- 電源可以用 9V Adapter 或USB 供電，PIC32 工作在 3.3V，LCD 模組工作在 5V
- APP1632 上的 8 個 LED 接於 PORTA <RA7:RA0>
  - 要注意 JTAG Port 的影響
  - PORTA 的腳位有和按鍵腳位重疊的接腳
  - JP9 需短路以接通 LEDs 對地的連接

MICROCHIP
Regional Training Centers

練習 一 (Lab1 BasicIO.mcp)

- C:\RTC\MCU4101T\Lab1 BasicIO\
  - Lab1 BasicIO.mcp 專案管理檔案(Project)
  - Lab1 BasicIP.mcw 工作環境設定檔
- 直接開啟 Lab1 BasicIO.mcp，使用 Debug 模式並確定編譯成功 (BUILD SUCCEEDED)
- 使用 APP1632 實驗板與 PICkit 3 or ICD3 連線並進行 Debug ..
- 在 RUN 的模式下，LED 是否可以計數顯示？

MICROCHIP
Regional Training Centers

有關 PIC32 最佳效能的設定

- 如練習一的程式，將 **SYSTEMConfig( )** 註解掉後重新編譯、燒錄再執行看看有何變化?
- 原本是 **100mS** 變化一次的 **LED** 現在變成 **1 Sec** 變化一次，速度好像差了 **10** 倍。
- 因為
  - **pre-fetch buffer** 及 **cache** 被關掉了
  - **Flash** 的 **Wait Cycle** 設為最長時間 **(7 Waits)**
  - **RAM** 的 **Wait Cycle** 設為最長時間 **(1 Waits)**

MICROCHIP
Regional Training Centers

使用 SYSTEMConfig( ) 設定效能

- 函式原型：

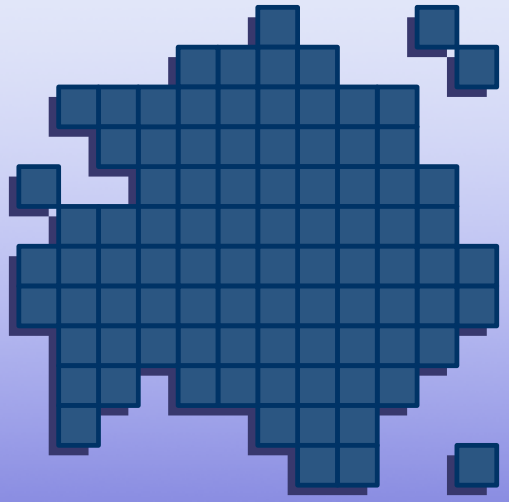
  unsigned int SYSTEMConfig(unsigned int sys_clock, unsigned int flags) ;

  - **sys_clock：系統所使用的工作頻率 (System Clock)**
  - **Flags (設定項) :**
    - **SYS_CFG_WAIT_STATES : 依據系統頻率設定 flash 的 wait states**
    - **SYS_CFG_PB_BUS : 設定 PB Clock = System Clock**
    - **SYS_CFG_PCACHE : 啟用 pre-fetch buffer 及 cache**
    - **SYS_CFG_ALL : 啟用以上三項設定**
- 相關的函式：
  - SYSTEMConfigPerformance( )
  - SYSTEMConfigWaitStatesAndPB( )
  - SYSTEMConfigPB( )
- 參考 32-bit-Peripheral-Library-Guide.pdf 裡的 2.0 SYSTEM FUNCTIONS 或 Microchip-PIC32MX-Peripheral-Library.chm

MICROCHIP
Regional Training Centers

使用 MPLAB SIM 注意事項

- 有關 **SYSTEMconfig** 之類的設定函式在使用 **MPLAB SIM** 時無法模擬出實際的效能，因為 **MPLAB SIM** 都是以系統理想狀態並假設 **PB Clock = System Clock**下去模擬的；關於這點在使用軟體模擬時要注意此差異。



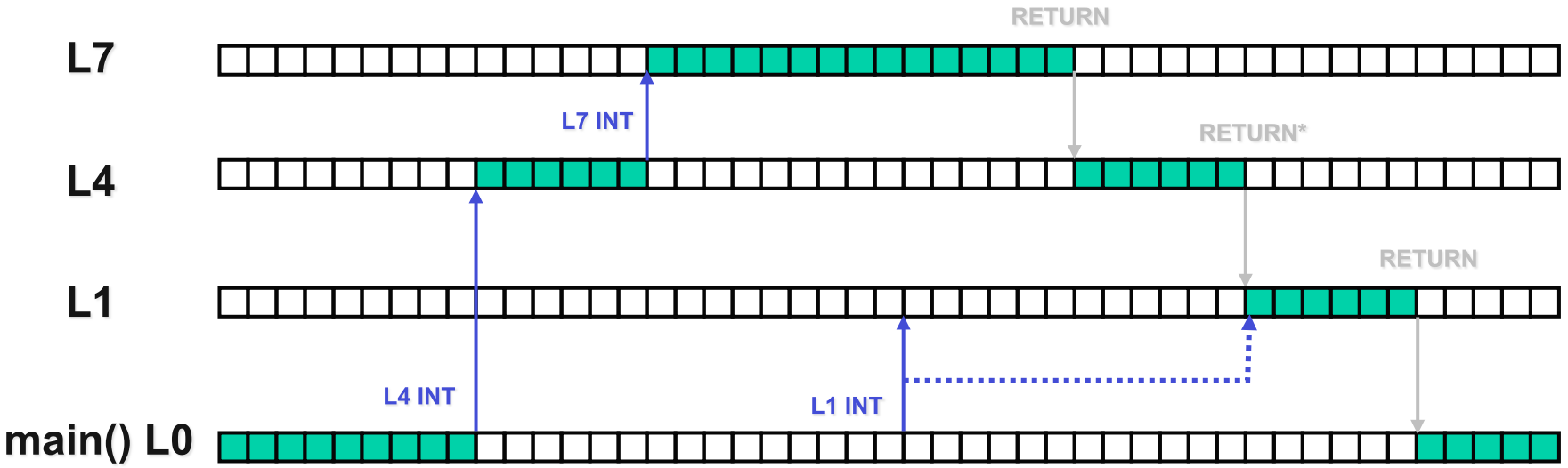
PIC32 Interrupts

MICROCHIP
Regional Training Centers

高效能的中斷支援

- PIC32 has a strong architectural focus on interrupt performance
- 內建 Vectored Interrupt Controller
  - Single vector mode for RTOSs – 預設值
  - 也支援 Multi Vector 的模式
    - Up to 96 Interrupt Sources
    - Up to 64 Interrupt Vectors
  - Multiple Priorities (7) with sub-priorities (4)
  - Shadow Register Set (for Level 7 only)

Preemptive Interrupt



CPU EXECUTION TRACE

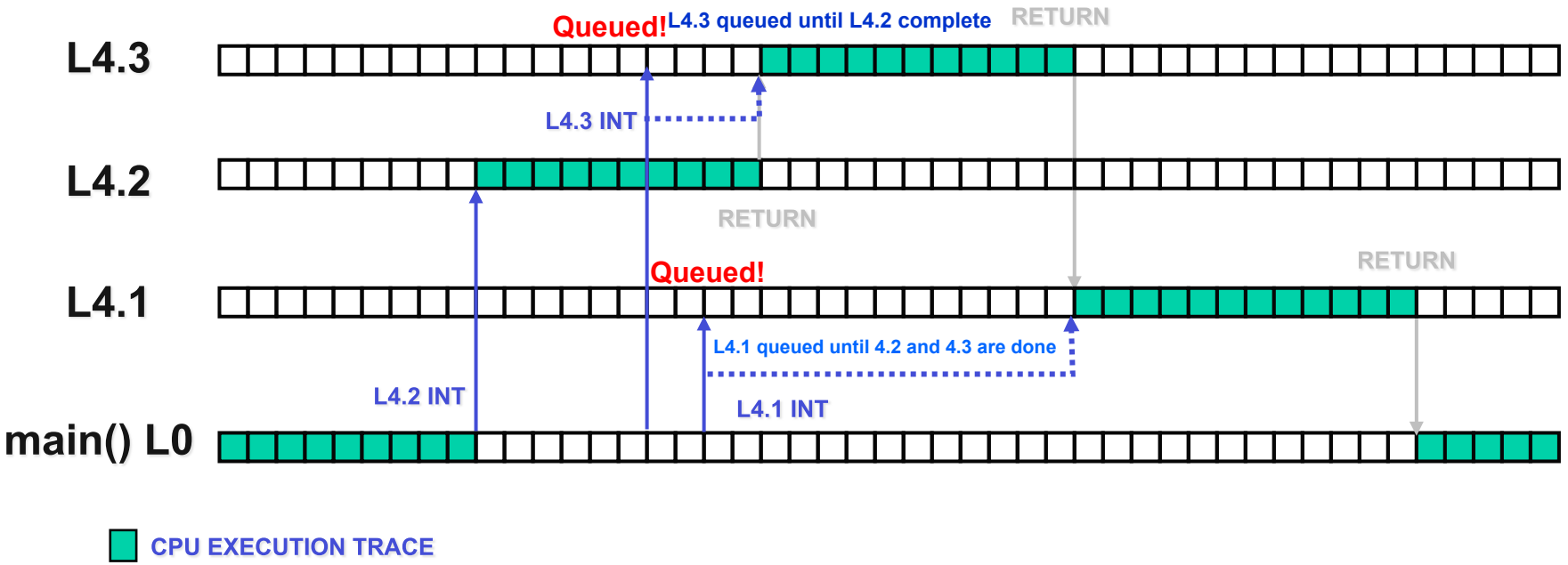
Status Register

15	14	13	12	11	10		
IPL 15	IPL 14	IPL 13	IPL 12	IPL 11	IPL 10		

IPL<15:10>

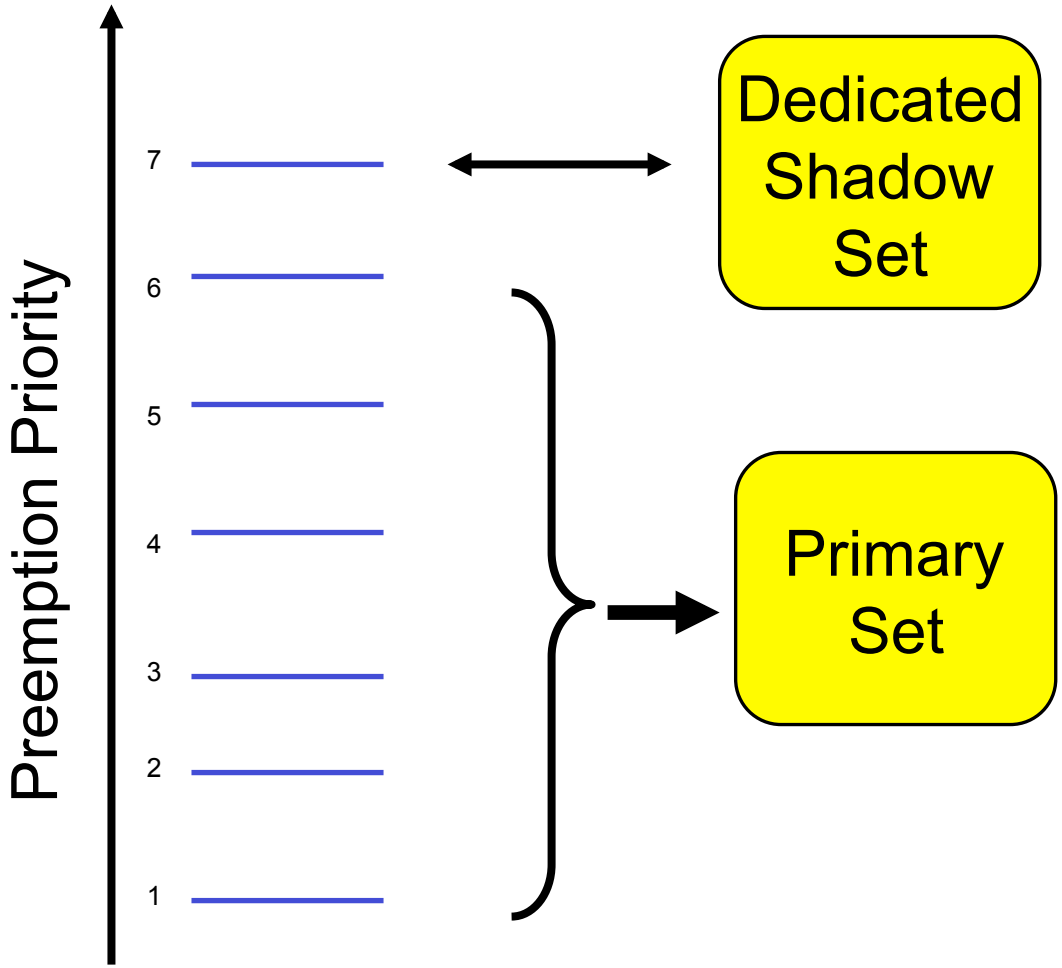
15	14	13	12	11	10
0	0	0	0	0	0

Interrupt Queueing
(sub-priority)



MICROCHIP
Regional Training Centers

Shadow Set



只有在最高優先權中斷 (Level 7)才有此功能。

CPU 自動切換為使用 Shadow 暫存器，無需軟體協助

加快中斷處理響應時間.

MICROCHIP
Regional Training Centers

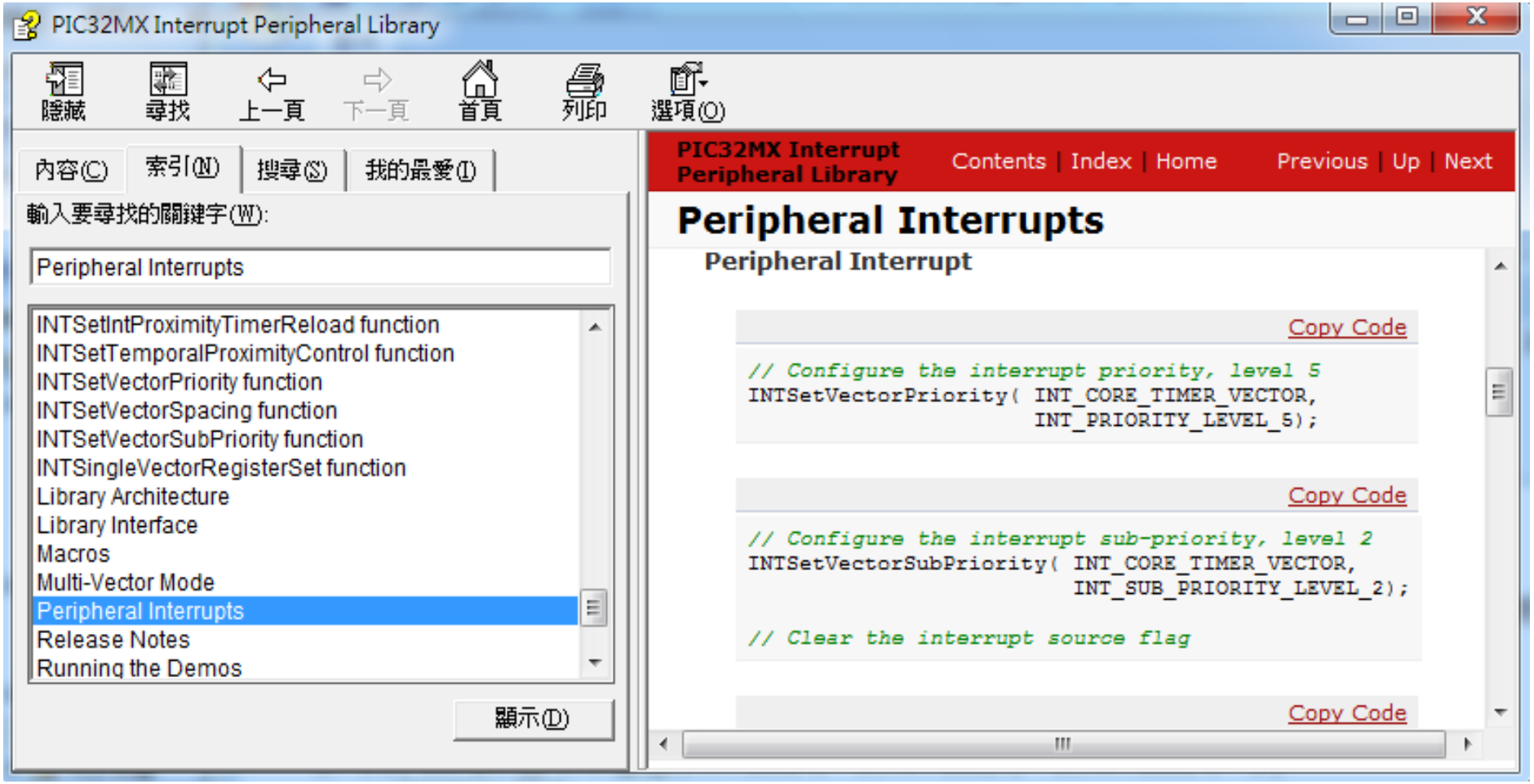
設定周邊中斷 1. 2. 3...

- 設定周邊中斷基本上有以下步驟 …
  1. **INTEnableSystemMultiVectoredInt();**
  2. 使用 **_ _ ISR (Vector, ipl)** 定義中斷函數
  3. 設定該週邊的中斷是否要致能及中斷的優先權等級 – 使用巨集 **ConfigIntxxxx( )** 或 **C32** 的基本 函式
  4. 設定該週邊的工作模式、功能 – 使用 **Openxxxx()**

- 致能及設定周邊中斷優先等級的 **MPLAB C32** 基本函式
  - **INTEnable( )**
  - **INTSetVectorPriority( )**
  - **INTSetVectorSubPriority()**

MPLAB C32 設定優先權的基本函式

- 在 **MPLAB C32** 的 **help** 檔可以找到對 **Interrupt** 設定的重要資訊 **..\MPLAB C32 Suite\doc\pic-lib-help\INT-PLIB_HELP**



MICROCHIP
Regional Training Centers

Software Support

- 需開啟多中斷向量功能
  - **INTEnableSystemMultiVectoredInt( )**
  - **Power-On reset 後是 Single Vector 的 mode !!**
- **C32 使用底下兩種方式宣告為 中斷函數**
  - **#pragma interrupt**
  - 或使用 **_ _ISR( )** 巨集

MICROCHIP
Regional Training Centers

定義中斷服務程式

- 使用 **void _ _ISR(vector,ipl) ISR_Function_Name (void)**

  來定義服務特定中斷向量的 “中斷服務程式”
  - **vector** 為中斷向量的編號 (從 **0** 開始)，相對的定義名稱在 **h** 檔裡
  - **Ipl** 為該中斷函數的優先權等級從 **ipl1 ~ ipl7 (ipl0** 為關閉此中斷)
  - **ISR** 所設定的 **ipl** 需與該周邊所設定的 **Interrupt Priority** 相同
  - **ISR_Function_Name** 為中斷函數名稱

  定義 **Core Timer** 中斷優先權等級為 **5** 的中斷函數

```
void __ISR(_CORE_TIMER_VECTOR, ipl5) CT_Int _Routine (void)
{
    :
    :
    mCTClearIntFlag( );   // 清除 Core-Timer 中斷旗號
}
```

MICROCHIP
Regional Training Centers

查詢 _ _ISR 向量名稱

- _ _ISR(vector,ipl) 的使用可以在 PIC32 C Compiler Guide 中的 3.4.4 章節中找到
- __ISR 使用的第一個參數為中斷向量編號 (vector) , 其定義在各 MCU 的 header file
  - PIC32MX795F512L.h
    - /* Vector Numbers */
    - #define _CORE_TIMER_VECTOR /*定義的名稱*/ 0 // (向量編號)
    - #define _CORE_SOFTWARE_0_VECTOR 1
    - #define _CORE_SOFTWARE_1_VECTOR 2
    - #define _EXTERNAL_0_VECTOR 3
    - #define _TIMER_1_VECTOR 4
    - #define _INPUT_CAPTURE_1_VECTOR 5
    - #define _OUTPUT_COMPARE_1_VECTOR 6
    - #define _EXTERNAL_1_VECTOR 7

MICROCHIP
Regional Training Centers

完整的 Vector Number 名稱

/* Vector offset Symbols */

_CORE_TIMER_VECTOR
_CORE_SOFTWARE_0_VECTOR
_CORE_SOFTWARE_1_VECTOR
_EXTERNAL_0_VECTOR
_TIMER_1_VECTOR
_INPUT_CAPTURE_1_VECTOR
_OUTPUT_COMPARE_1_VECTOR
_EXTERNAL_1_VECTOR
_TIMER_2_VECTOR
_INPUT_CAPTURE_2_VECTOR
_OUTPUT_COMPARE_2_VECTOR
_EXTERNAL_2_VECTOR
_TIMER_3_VECTOR
_INPUT_CAPTURE_3_VECTOR
_OUTPUT_COMPARE_3_VECTOR
_EXTERNAL_3_VECTOR
_TIMER_4_VECTOR
_INPUT_CAPTURE_4_VECTOR
_OUTPUT_COMPARE_4_VECTOR
_EXTERNAL_4_VECTOR
_TIMER_5_VECTOR
_INPUT_CAPTURE_5_VECTOR
_OUTPUT_COMPARE_5_VECTOR
_CHANGE_NOTICE_VECTOR
_SPI1_VECTOR
_UART1_VECTOR
_I2C1_VECTOR
_ADC_VECTOR
_PMP_VECTOR
_COMPARATOR_1_VECTOR
_COMPARATOR_2_VECTOR
_SPI2_VECTOR
_UART2_VECTOR
_I2C2_VECTOR
_FAIL_SAFE_MONITOR_VECTOR
_RTCC_VECTOR
_DMA0_VECTOR
_DMA1_VECTOR
_DMA2_VECTOR
_DMA3_VECTOR

MICROCHIP
Regional Training Centers

ConfigInt[xxxx]
設定周邊中斷及優先權等級

- **ConfigInt[xxxx] (INT ON/OFF | INT_Priority | INT_Sub_Priority)**
- 參數的傳遞 (config)基本有三項，使用 OR (|) 的運算加入
  - 參數一：開啟 / 關閉 該週邊的中斷
  - 參數二：主要的中斷優先權等級
  - 參數三：次要的中斷優先權等級
- 有關此函數的使用請參考:
  - **32-bit-Peripheral-Library-Guide.pdf**
  - **Microchip-PIC32MX-Peripheral-Library.chm**

MICROCHIP
Regional Training Centers

Open[xxxx]
設定該週邊的工作模式、功能

- **Open[xxxx] (config1, config2, …)**
- 基本上**Open** 這個函數是設定該週邊的工作模式及功能，相對的也就複雜多了。建議先研讀相關的周邊後再來設定輸入的參數項
  - 周邊詳細資料 : **PIC32_Family Reference Manuals**
- 參數的多寡不一定，看該週邊有多少的控制暫存器要設定
- 一樣的，有關此函數的使用請參考:
  - **32-bit-Peripheral-Library-Guide.pdf**
  - **Microchip-PIC32MX-Peripheral-Library.chm**

MICROCHIP
Regional Training Centers

清除周邊的中斷旗號

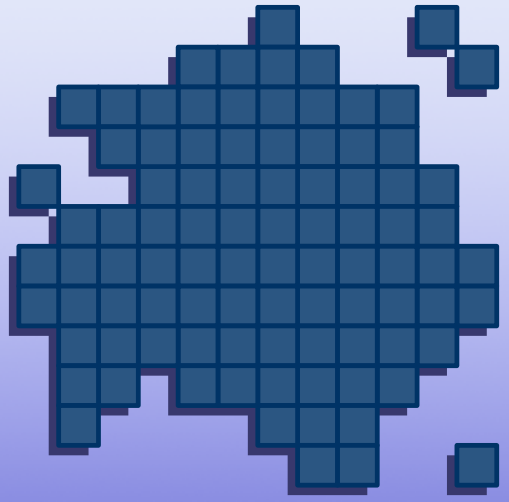
- 這裡有三種方式可以清除中斷旗號
  1. 傳統使用位元結構：**IFS0bits.T1IF = 0;**
  2. 使用 **C32** 的巨集：**m[xx]ClearIntFlag( );**
  3. 使用 **C32** 的清除函數：**INTClearFlag(xx);**
     - **[xx]** 為該週邊的 **INT_SOURCE** 列舉名稱
     - 可以在 **32-bit-Peripheral-Library-Guide.pdf** 查到
       - **TABLE 8-1: INTERRUPT ENUMERATIONS**
     - 或 **.. \MPLAB C32\doc\pic32-lib-helpINT-PLIB-Help.chm**
       - 搜尋 **“ INT_SOURCE Enumeration “**

列舉名稱	周邊名稱
INT_INT0	External Interrupt 0
INT_T1	Timer 1 Interrupt
INT_IC1	Input Capture 1 Interrupt
INT_OC1	Output Compare 1 Interrupt
:	:

所以使用中斷是很簡單的

- 使用 **Timer1** 中斷設定範例：

```
INTEnableSystemMultiVectoredInt();
void Init_Timer1(void)
{
    ConfigIntTimer1(T1_INT_ON | T1_INT_PRIOR_4) ;       // 將Timer1的中斷優先權設定為 4
                                                        // 將Timer1中斷致能打
    開
    OpenTimer1( T1_ON | T1_IDLE_STOP | T1_GATE_OFF | T1_PS_1_256 | T1_SOURCE_INT, \
                ( (PBCLK / (1000*256) ) * T1_TICK_TIME_MS - 1 ) ) ;
    // Timer1 On, Gate Time Off, Pre-Scaler 1:256, Internal_Source, Delay 200mS
}
//*****************************************************************//
/*              TIMER1中斷副程式                                   */
//*****************************************************************//
void __ISR(_TIMER_1_VECTOR, ipl4) Timer1Handler(void)
{
        mT1ClearIntFlag() ;                     // 清除 Timer1 interrupt Flag ..  // C32 所提供的巨集
        // IFS0bits.T1IF = 0;                   // 傳統清除旗號的語法
        :
        :  // 中斷處理程序
        :
}
```



PIC32 的 Core Timer

MICROCHIP
Regional Training Centers

Core Timer 介紹

- **Core Timer 是 PIC32 中 CP0 可以直接控制的 32-bit Timer**
- **Core Timer 用System Clock，一般 Timer 用是 PB_Clock**
- **PIC32 Core Timer Interrupt is controlled by:**
  - *CP0 COUNT and COMPARE Registers*
  - *CP0 COUNT 的計數是兩個 System Clock 就加 1*

- **Interrupt Service Routine**
  - *MPLAB C32 provides __ISR( ) macro to declare ISRs*
  - *Every ISR is associated with Interrupt Vector and User -assigned Interrupt Priority (ipl1 ~ ipl7)*

MICROCHIP
Regional Training Centers

Core Timer 的函式

- **CPU Core Timer Operations**
  - OpenCoreTimer( ) 初始化並清除CT再載入新的 period 值
  - UpdateCoreTimer( ) compare = 目前的計數值 + period
  - mConfigIntCoreTimer ( )
  - mEnableIntCoreTimer( )
  - mDisableIntCoreTimer( )
  - mSetPriorityIntCoreTimer( )
  - ReadCoreTimer( )
  - WriteCoreTimer( )

- 查看 **32-bit-Peripheral-Library-Guide.pdf**
  - 第 **11.0 TIMER FUNCTIONS** 有詳細的說明

MPLAB C32 Peripheral libraries 對 Core Timer 的 support

- 在主程式中可用的 **Functions & macro**

```
/* setup system wide interrupts */
INTEnableSystemMultiVectoredInt();

mConfigIntCoreTimer (CT_INT_ON | CT_INT_PRIOR_3);

OpenCoreTimer(TIMER_PERIOD);
```

- **Interrupt Handler 的可用 macros 及寫法**

```
void __ISR(_CORE_TIMER_VECTOR, ipl3)  CoreTimerIntHandler(void)
{
   mCTClearIntFlag( );                          /* clear interrupt */
   UpdateCoreTimer(TIMER_PERIOD);          /* setup next timer interrupt */

   /* do work here */
}
```

CT 為 Core Timer 的縮寫名稱

MICROCHIP
Regional Training Centers

練習 二：PIC32 Core Timer 搭配 Interrupts

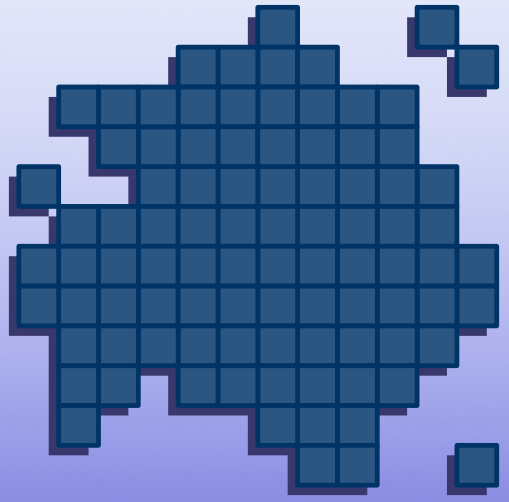
- 開啟C:\RTC\MCU4101T\Lab2 CoreTimer.mcp
  - 完成 Core Timer 的設定及中斷優先權的設定
  - 熟悉 Interrupt Handler (Interrupt Service Routine) 的宣告方式
  - 將 Interrupt Handler 與正確的 Vector 對應
  - 學會 PIC32MX Core Timer & GPIO 的操作
- 程式執行結果 8 個 LED 在中斷裡每 200mS 會交互閃爍。

練習 二的 200mS 是怎樣算出來的

```
#define SystemFrequency      60000000L
#define CoreTick             ( SystemFrequency / 2 / TogglesPerSec )
#define TogglesPerSec        5
```

System Clock 為60MHz，Core Timer 的計數頻率為 30MHz
(SystemFrequency / 2) : 為 1 Sec 的 Core Timer 所要的計數值
(1Sec / TogglesPerSec) 就是 Core Timer 每 200mS 的計數值

- **UpdateCoreTimer(CoreTick);**
  - 更新下一次 200mS 的Core Timer 新的計數值；此函數會將目前的計數值再 + 200mS 的計數增量值 = 新的 Core Timer 設定值
  - 記住! Core Timer 是 32-bit 的 Timer



PIC32 的 General Purpose Timer

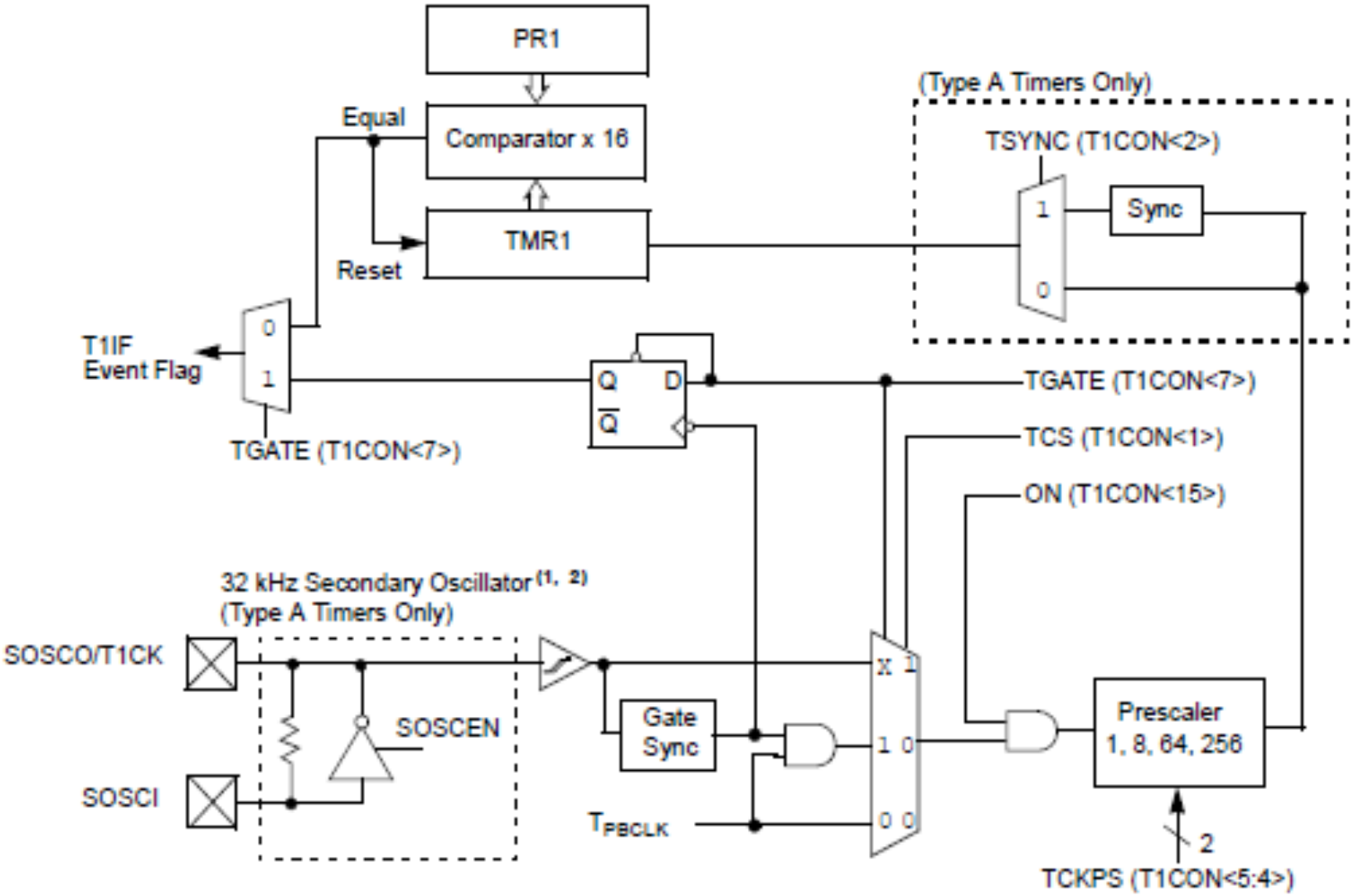
請參考 PIC32 Family Reference Manual – 第 14 章

MICROCHIP
Regional Training Centers

PIC32 Timers

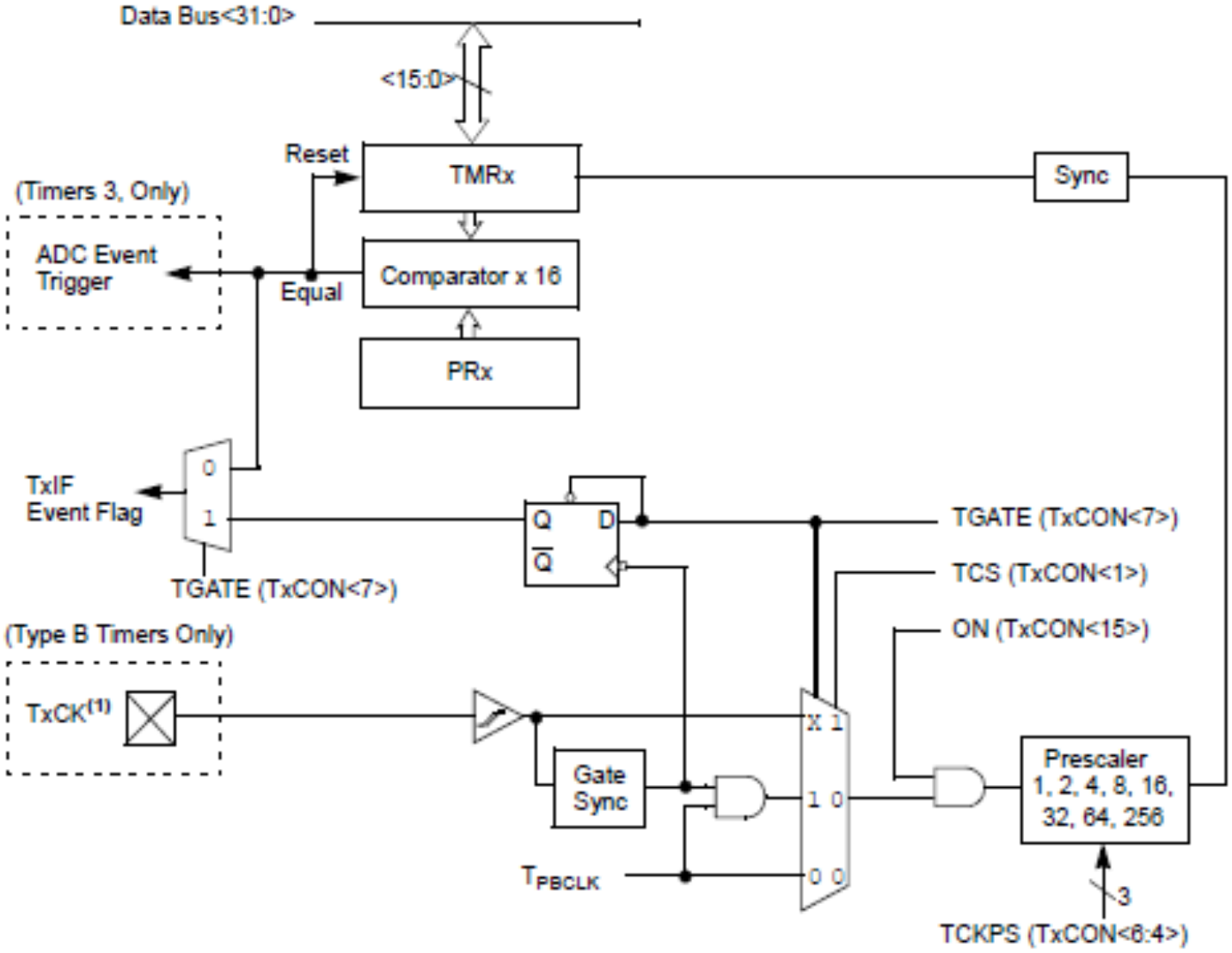
- PIC32 與 PIC24F 有相同的 Timer 結構
- 為保持相容性，TXCON 暫存器
  - 為 16-bit 的暫存器
- 只有Timer1 具有非同步計數器功能
  - Prescaler 可支援：1,8, 64,256
- Timer 2 , 3, 4 , 5 一定是同步計數器功能
  - Prescaler 可支援：1,2,4,8,16,32,64,256
- 使用內部時脈時，clock 來源為 PBCLK
- 都可以支援外部閘控(Gate Time) 計時功能

Timer 1 方塊圖



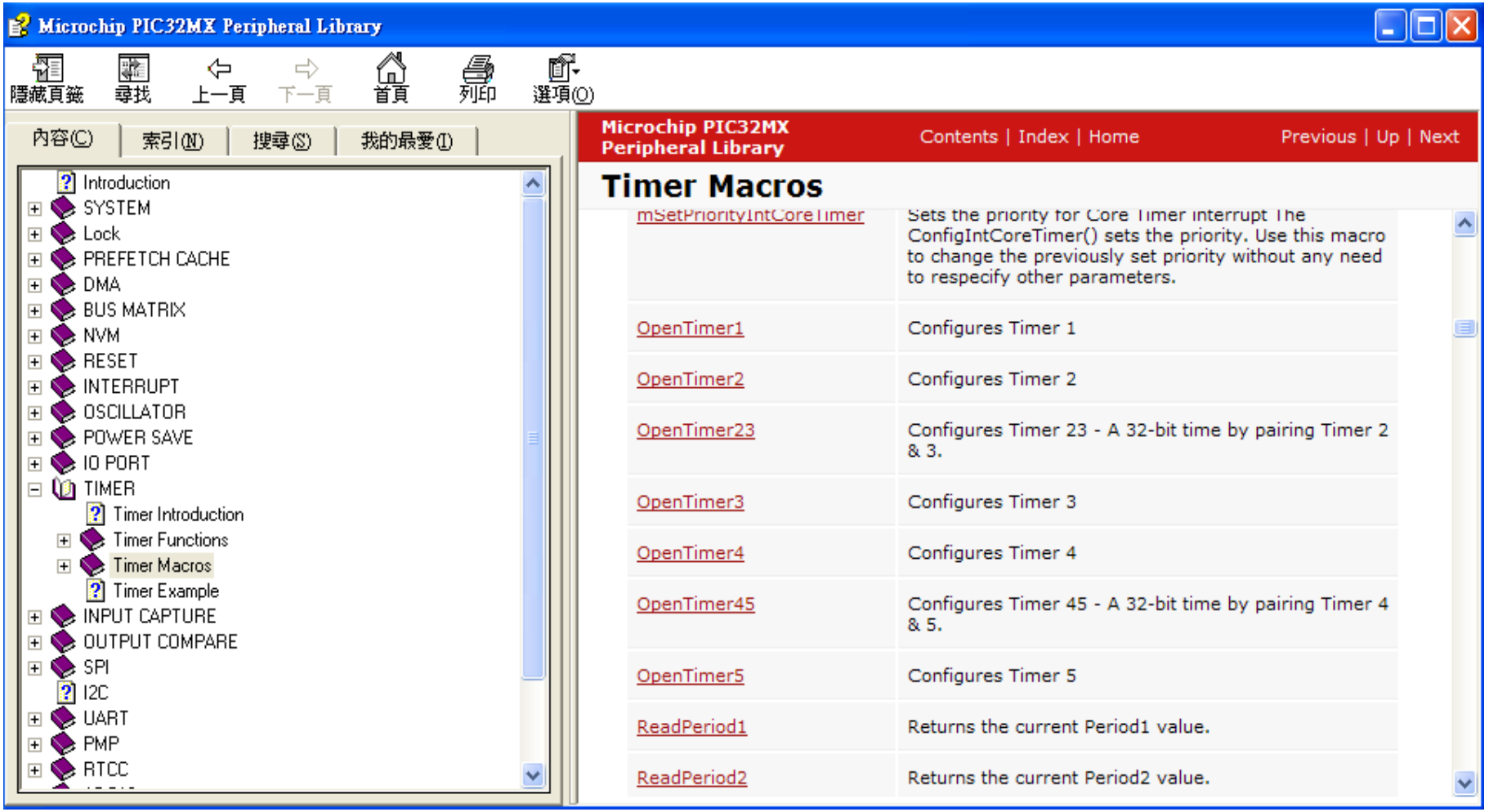
MICROCHIP
Regional Training Centers

其它 Timer 的方塊圖



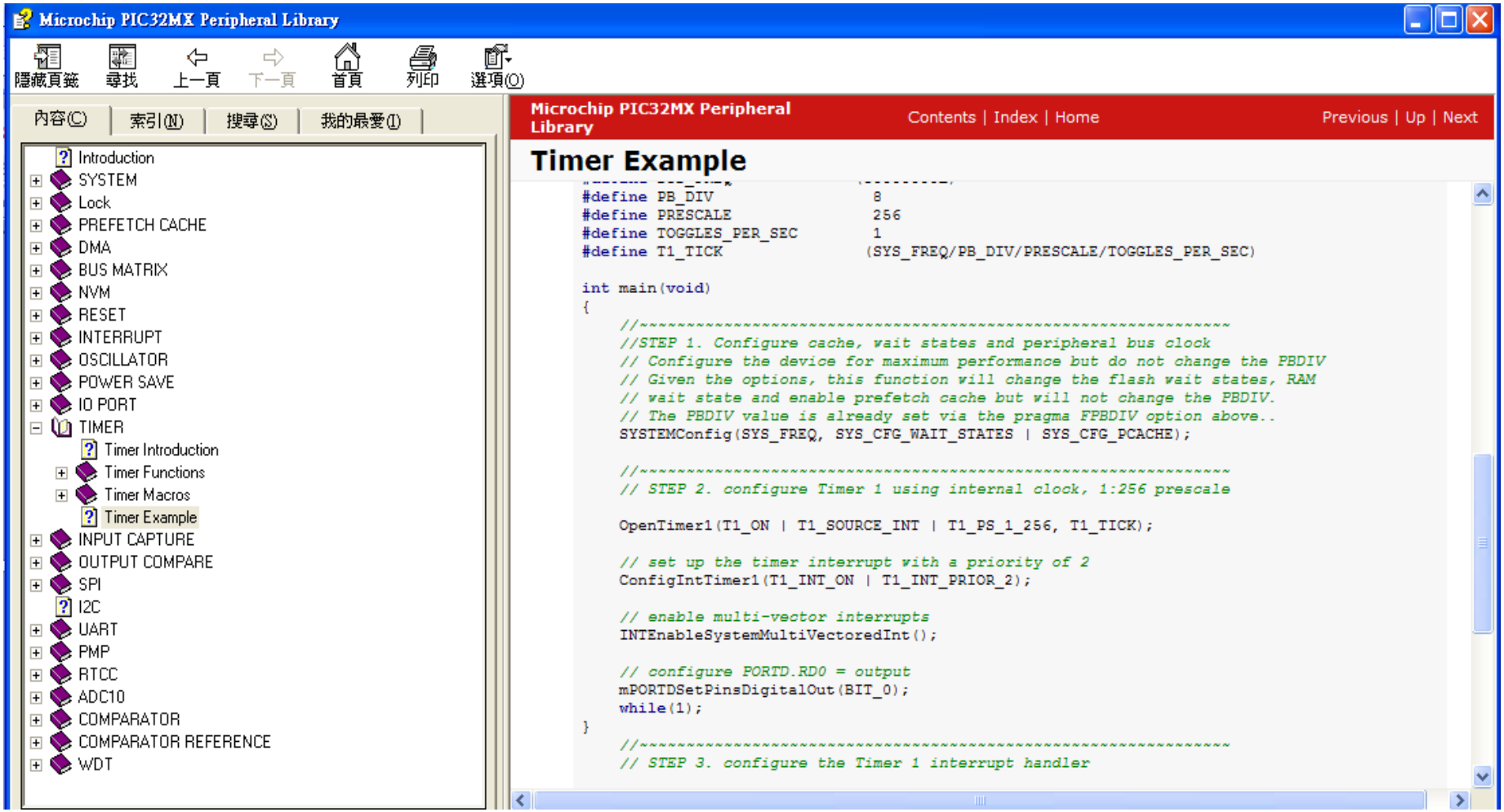
PIC32 周邊 Library 對 Timer 的 Support

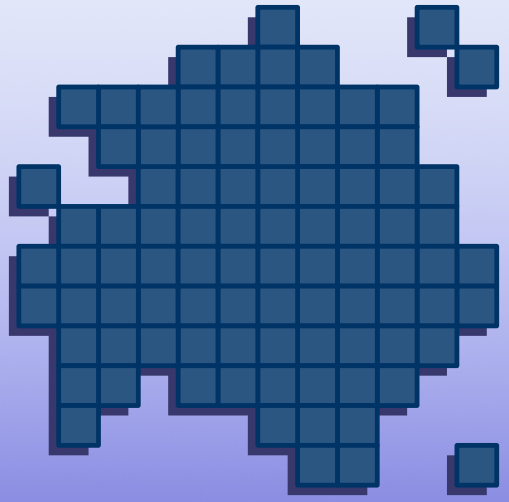
- 參考：Microchip-PIC32MX-Peripheral-Library.chm
- 詳細的參數定義請參考 MPLAB C32 的 timer.h



Timer function & macro 的使用範例

- 參考：Microchip-PIC32MX-Peripheral-Library.chm





練習 三：使用一般的 Timer 來計時

MICROCHIP
Regional Training Centers

練習三：Timer1 Functions

- 對 Timer1 的設定
  - ConfigIntTimer1 ( ) ;
  - OpenTimer1( ) ;
- 對 Timer1 的中斷 Flag 清除
  - mT1ClearIntFlag()
  - INTClearFlag(INT_T1) ;
- Timer1 的服務程式與 Vector Number 的連結

```
void __ ISR (_TIMER_1_VECTOR, ipl4) ISR_Timer1 (void)
    {
      ……
     }
```

MICROCHIP
Regional Training Centers

練習 三：Timer1

- 開啟C:\RTC\MCU4101T\Lab3 Timer1.mcp
- 使用 Peripheral Library 來設定 Timer1
  - OpenTimer1(… , …) ;
  - 讓 Timer1 每 100 ms 能 Toggle LED 一次
- 練習 __ISR( ) 的設定與 Vector 的安排
  - void __ ISR (_TIMER_1_VECTOR, ipl4) ISR_Timer1 (void)
- 致能與設定 Timer1 Interrupt Priority & Level
  - ConfigIntTimer1 (…) ;
- 使用 GPIO PORTA/LATA 來 toggle LEDs
  - mPORTAToggleBits(BIT_X)
  - 可以用 | 符號來一次 操作多個 Bits
- 參考 C32 Peripheral Library Guide

Microchip PIC32 MCU

內建周邊模組

Parallel Master Port (PMP)

請參考 PIC32 Family Reference Manual

第 13 章

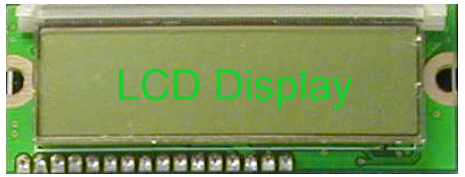
What's Parallel Master Port – PMP

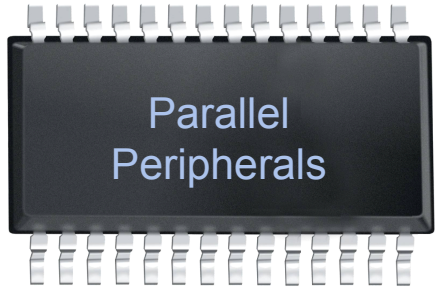
Read, Write, Enable

Up to 2 Chip Select

Up to 16-bit Address

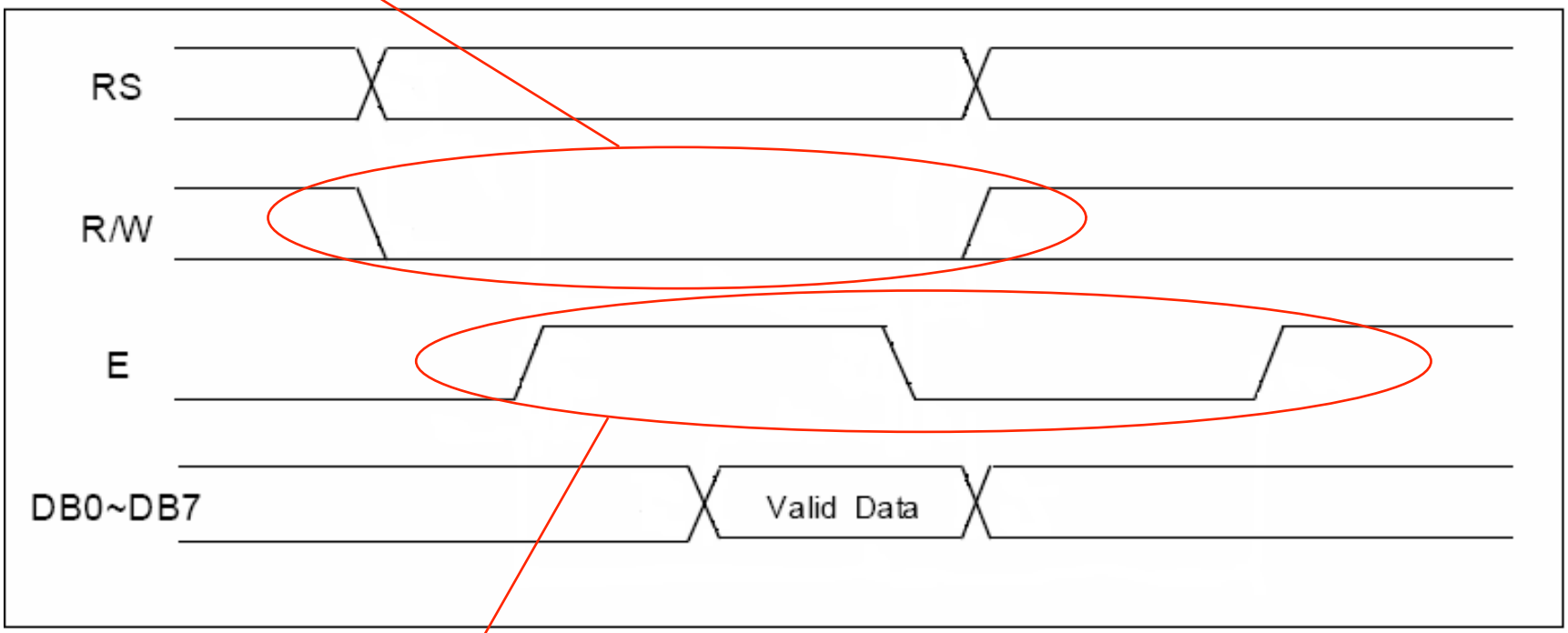
8- or 16-bit DATA





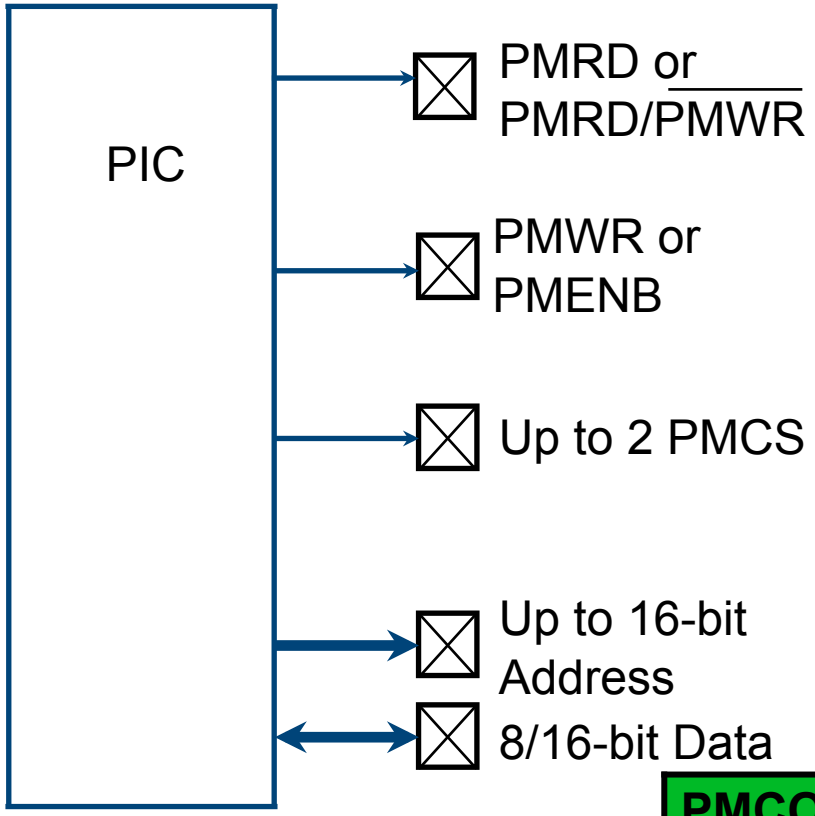
Why PMP
Example : LCD Write Timing

Write is active low.
But remember! PMP signal we are using is PMRD/PMWR,
so the pin needs to be configured as active high.



Enable is active High.

PMP Configuration & Control Signals



The Signal can be individually Enabled or Disabled

PMCON: PTWREN,PTRDEN PTRDEN

PMAEN: CS2, CS1

The Signal polarity can be individually selected

PMCON: ALP, CS2P, CS1P, ,WRSP, RDSP

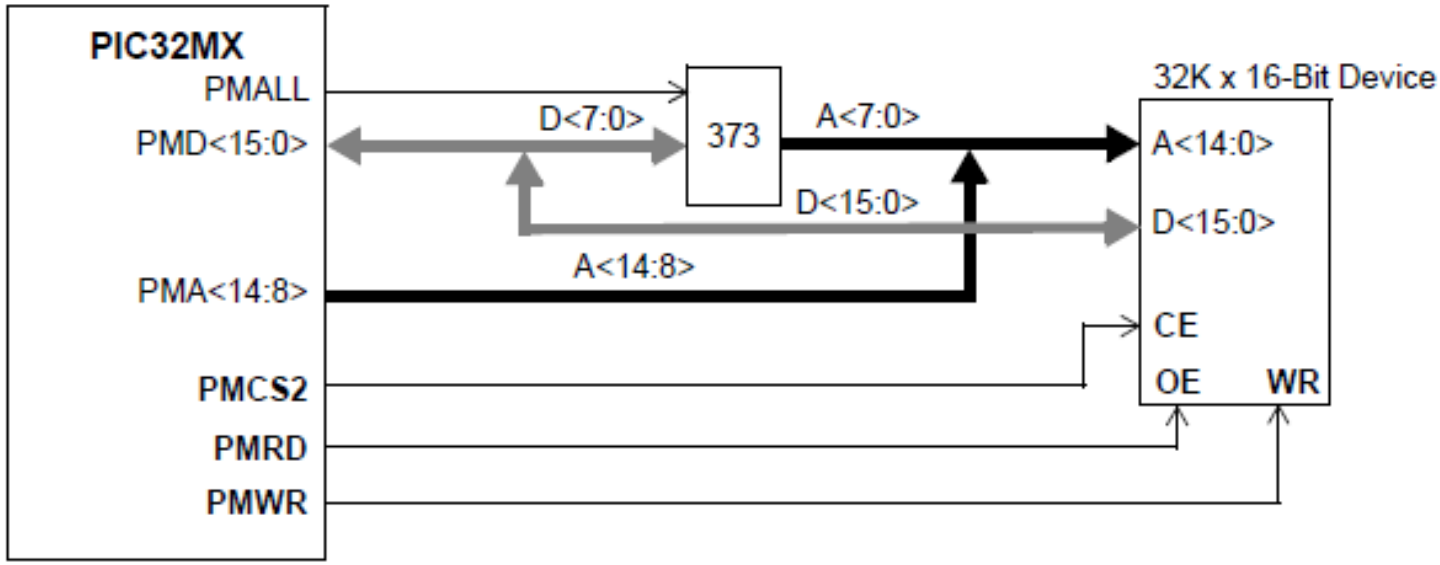
The Address pins can be individually Enabled or Disabled

PMAEN: Reg

PMCON<CSF1:CSF0>	Chip Select Function
00	CS1, CS2, A15, A14
01	CS1, CS2, A15, A14
10	CS1, CS2, A15, A14

使用 D0..D7 達成 A0~A7 的位址多工

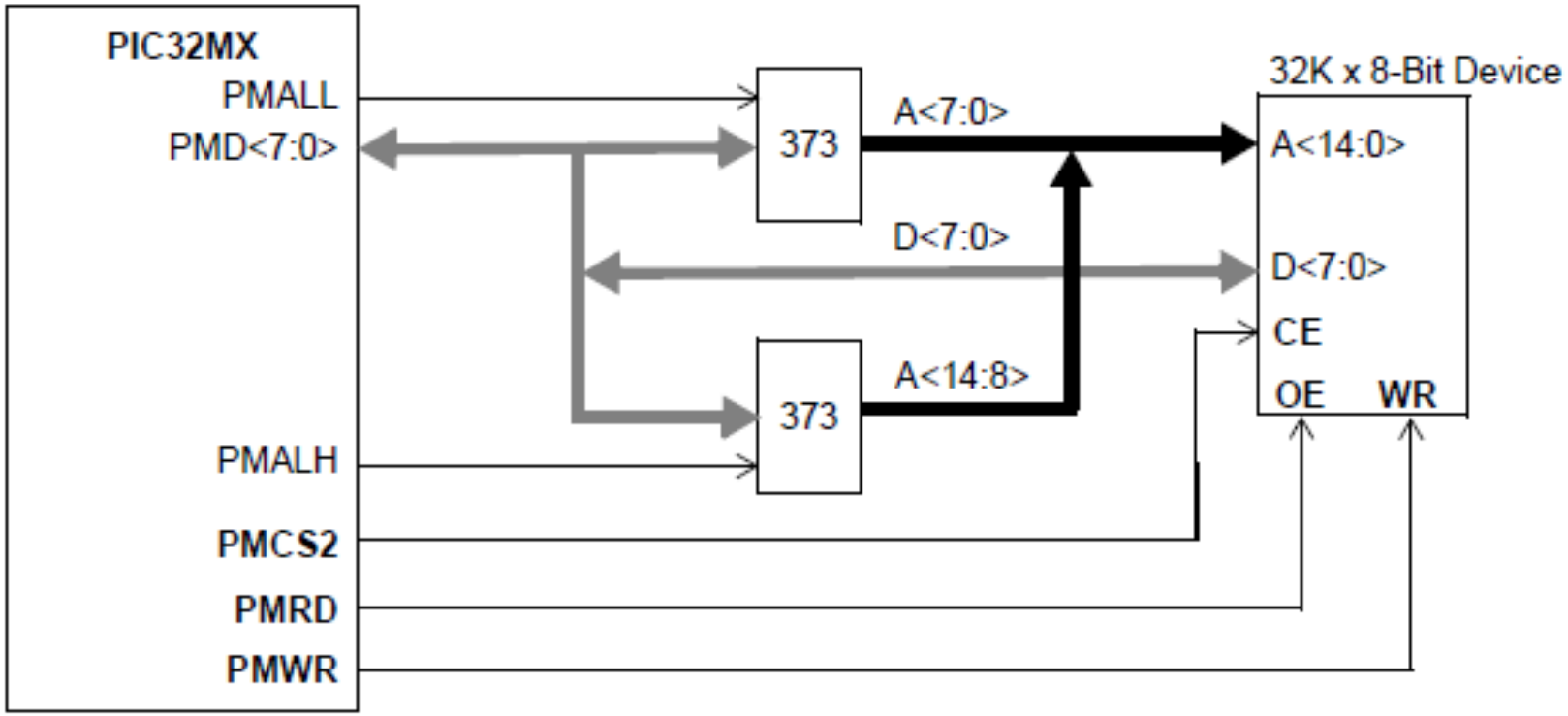
** 64-pin device 的 PMD 只有 0 .. 7



Note: (Master mode 2) MODE<1:0> = 10
(16-bit data width) MODE16 (PMMODE<10>) = 1
(Partial Multiplexed mode) ADRMUX (PMCON<12:11>) = 01
The 373 shown in the diagram represents a generic 74XX family 373 latch.

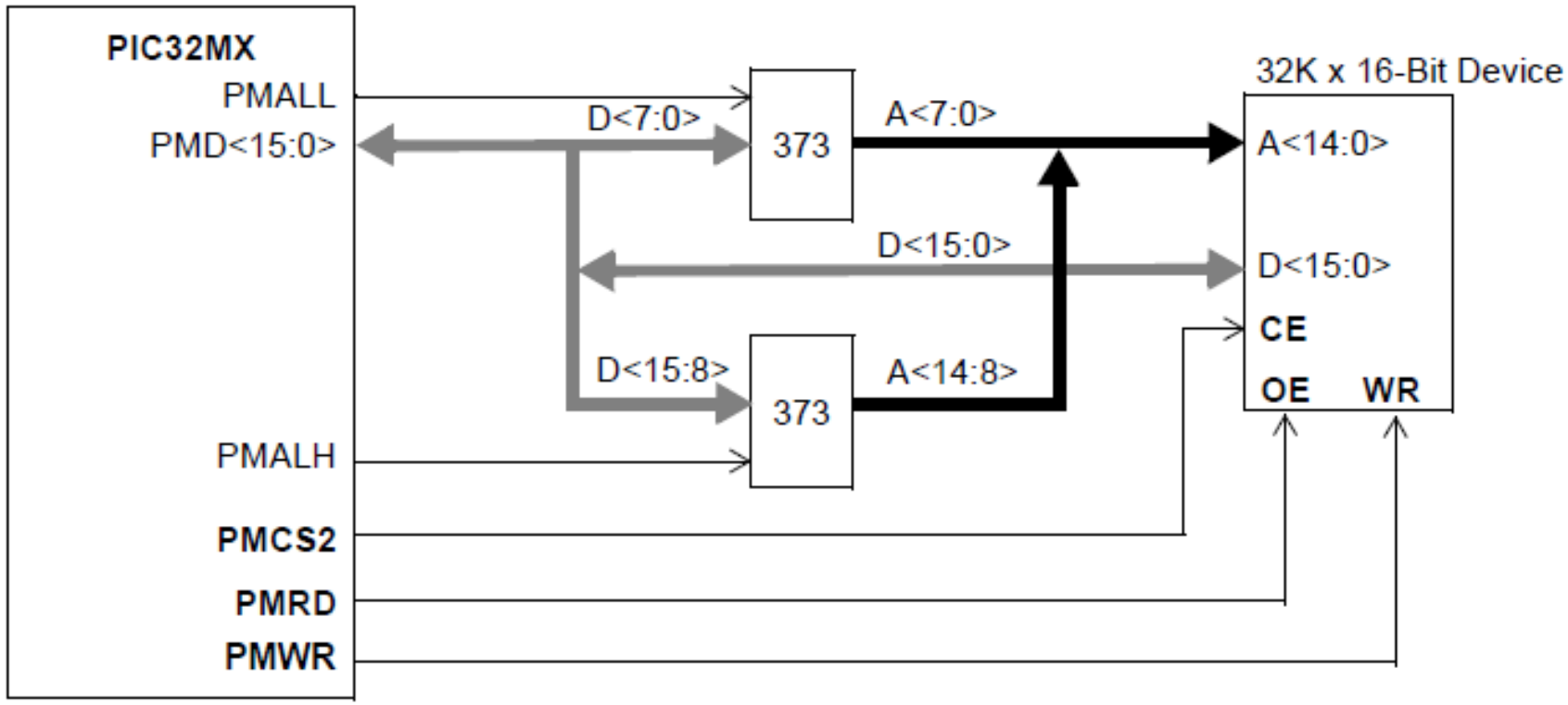
Address Bus
Data Bus
Control Lines

使用 D0..D7 達成 A0~A15 的位址多工



使用 D0..D15 達成 A0~A15 的位址多工 (只有 100-pin 的 PIC32)

- **PMCON<ADRMUX1:ADRMUX0> = 11**



MICROCHIP
Regional Training Centers

PMP 對低速周邊的時序調整

PMMODE Register

bit 7-6 **WAITB1:WAITB0:** Data Setup to Read/Write Wait State Configuration bits(1)

11 = Data wait of 4 **TPB** ; multiplexed address phase of 4 **TPB**
10 = Data wait of 3 **TPB** ; multiplexed address phase of 3 **TPB**
01 = Data wait of 2 **TPB** ; multiplexed address phase of 2 **TPB**
00 = Data wait of 1 **TPB** ; multiplexed address phase of 1 **TPB**

bit 5-2 **WAITM3:WAITM0:** Read to Byte Enable Strobe Wait State Configuration bits

1111 = Wait of additional 15 **TPB**
...
0001 = Wait of additional 1 **TPB**
0000 = No additional Wait cycles (operation forced into one **TPB**)

bit 1-0 **WAITE1:WAITE0:** Data Hold After Strobe Wait State Configuration bits(1)

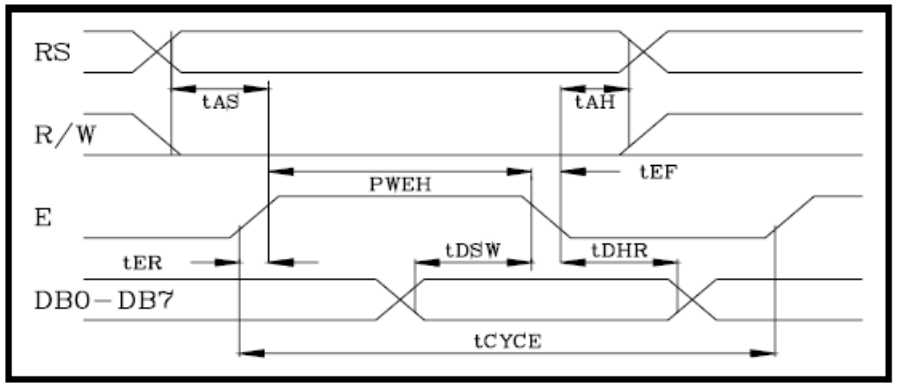
11 = Wait of 4 **TPB**
10 = Wait of 3 **TPB**
01 = Wait of 2 **TPB**
00 = Wait of 1 **TPB**

Note 1: WAITBx and WAITEx bits are ignored whenever WAITM3:WAITM0 = 0000.

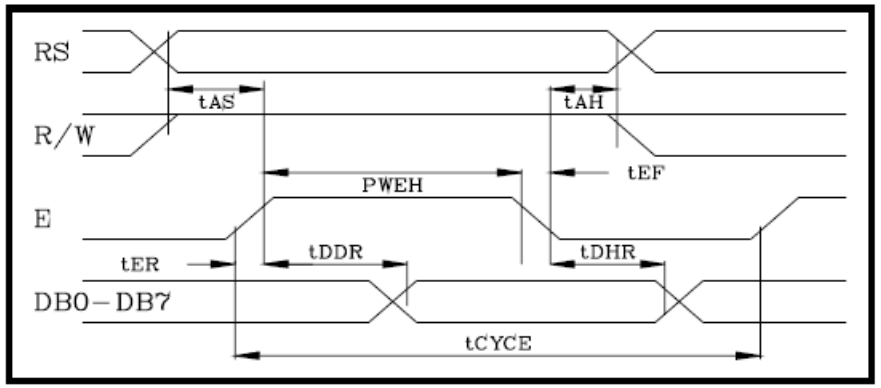
MICROCHIP
Regional Training Centers

Wait State 使用範例
符合LCD W/R Timing

Write Operation

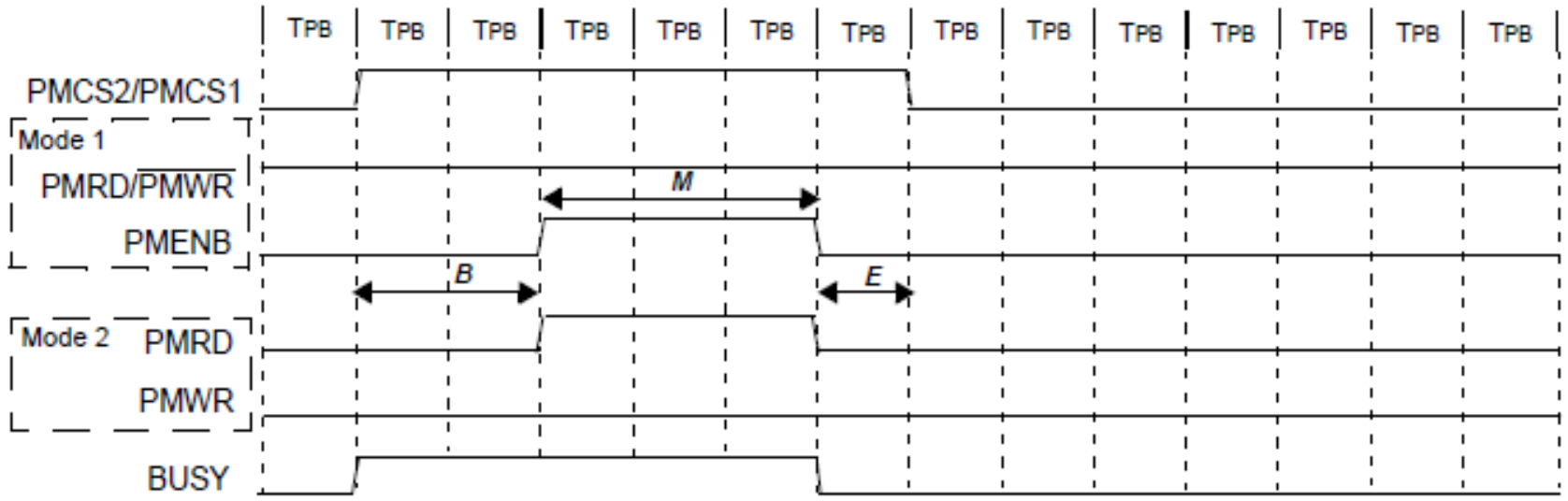


Read Operation



Item	Symbol	Limit (Min.)	Limit (Max.)	Unit
Enable Cycle Time	tCYCE	1000	--	ns
Enable Pules Width (High level)	PWEH	450	--	ns
Enable Rise/Fall Time	tER,tEF	--	25	ns
Address Set-Up Time (RS,R/W,E)	tAS	100	--	ns
Address Hole Time	tAH	10	--	ns
Data Set-Up Time	tDSW	100	--	ns
Data Delay Time	tDDR	--	190	ns
Data Hold Time	tDHR	20	--	ns

PMP Wait State 時序範例

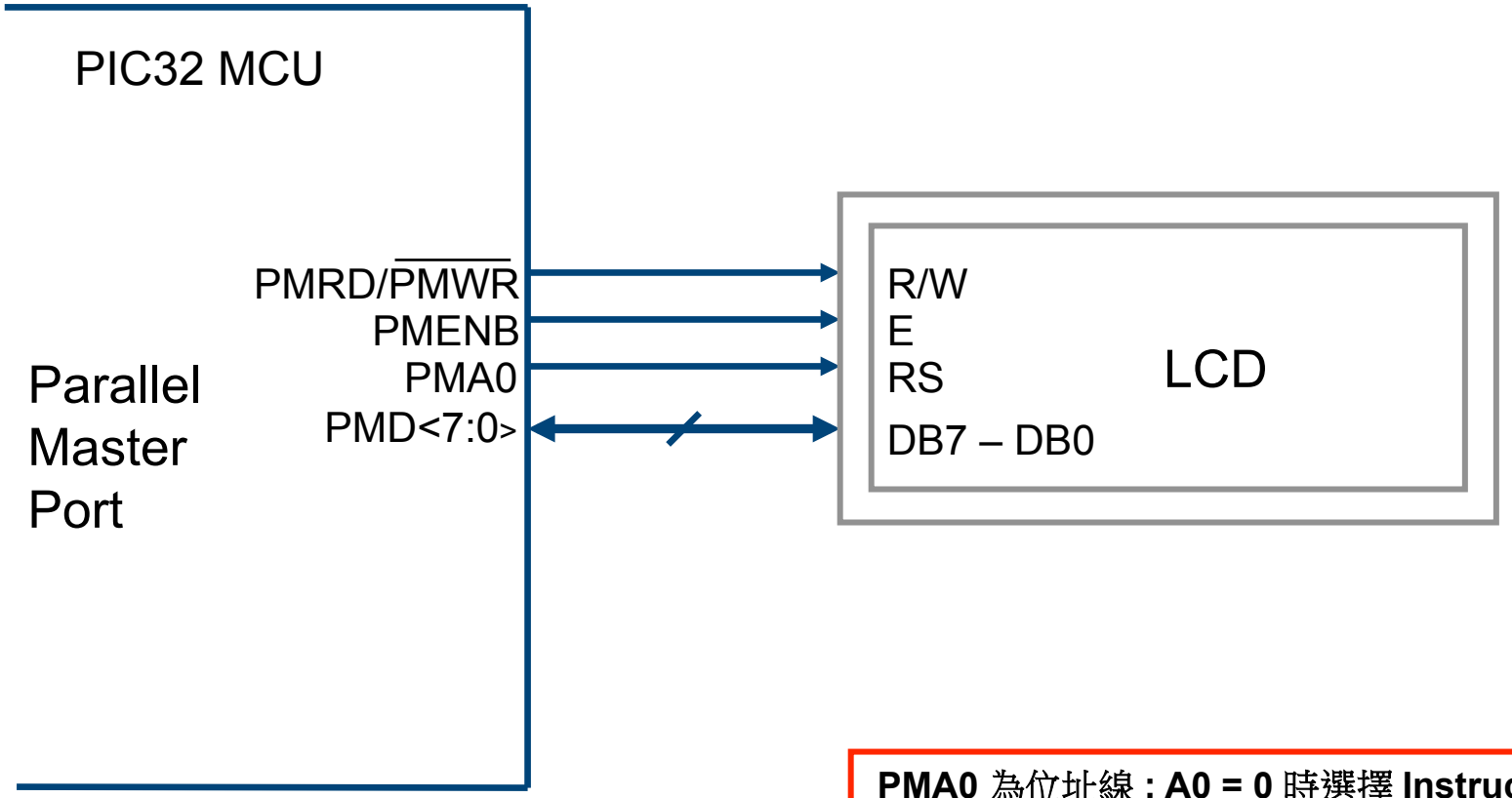


Legend:

B = WAITB<1:0> = 01 (2 Wait states)
M = WAITM<3:0> = 0010 (3 Wait states)
E = WAITE<1:0> = 01 (1 Wait state)

Note: If WAITM<3:0> = 0000, M is forced to 1 T_{PBCLK}, WAITB is ignored (B forced to 1 T_{PBCLK}), and WAITE is ignored (E forced to 0 T_{PBCLK}).

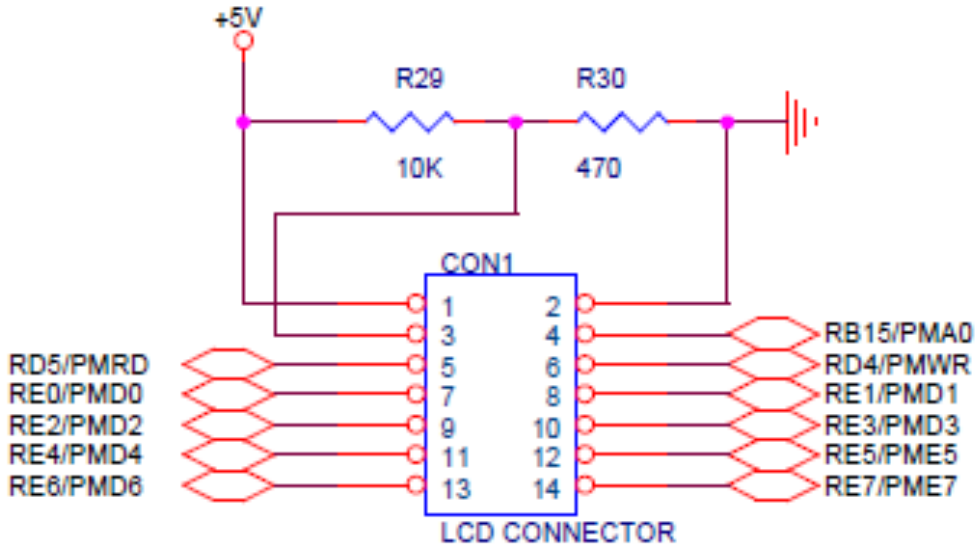
APP1632 中LCD 與 PMP 的硬體連接方式

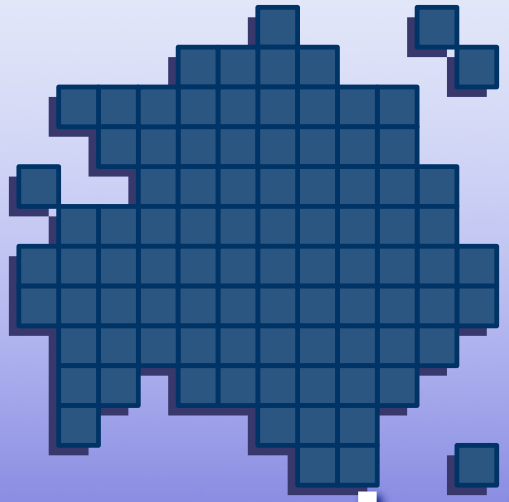


PMA0 為位址線；A0 = 0 時選擇 Instruction
A0= 1 時選擇 Data

APP1632 LCD 的線路圖

- 已參考 **PIC32MX795F512L PMP** 的接腳，將 **LCD** 信號接至正確腳位
- **PIC32MX795F512L** 的 **PMP** 腳位安排相容於 **24FJ128GA010**
- **APP1632** 可以使用原先為 **Explorer-16** 所寫的 **LCD** 範例程式





Let’s go Hands on
&
Case Study first !!

MICROCHIP
Regional Training Centers

PMP_Exercise
使用 PMP 來控制 LCD

- 使用 PIC32MX795F512L 的 PMP
  - 以 PMP 來完成 LCD 的初始化
  - 完成存在於 LCM.c 的各個副程式
    - 把 MCU 型號輸出至 LCD 第一行
    - 在第二行將程式中 Timer 計時完成的次數 update 在 LCD 上
- 只有 100-pin 的 PIC32 Device 有 PMD8 .. PMD15

LCD Module 的接腳
LMC-SSC2A16

NO	SYMBOL	LEVEL	FUNCTION
1	VDD	--	DC +5V
2	VSS	--	GND (0V)
3	VO	H/L	Contrast Adjust
4	RS	H/L	Register select
5	R/W	H/L	Read/Write
6	E	H,H→L	Enable signal
7	DB0	H/L	Data Bit 0
8	DB1	H/L	Data Bit 1
9	DB2	H/L	Data Bit 2
10	DB3	H/L	Data Bit 3
11	DB4	H/L	Data Bit 4
12	DB5	H/L	Data Bit 5
13	DB6	H/L	Data Bit 6
14	DB7	H/L	Data Bit 7

MICROCHIP
Regional Training Centers

LCD Commands [1]

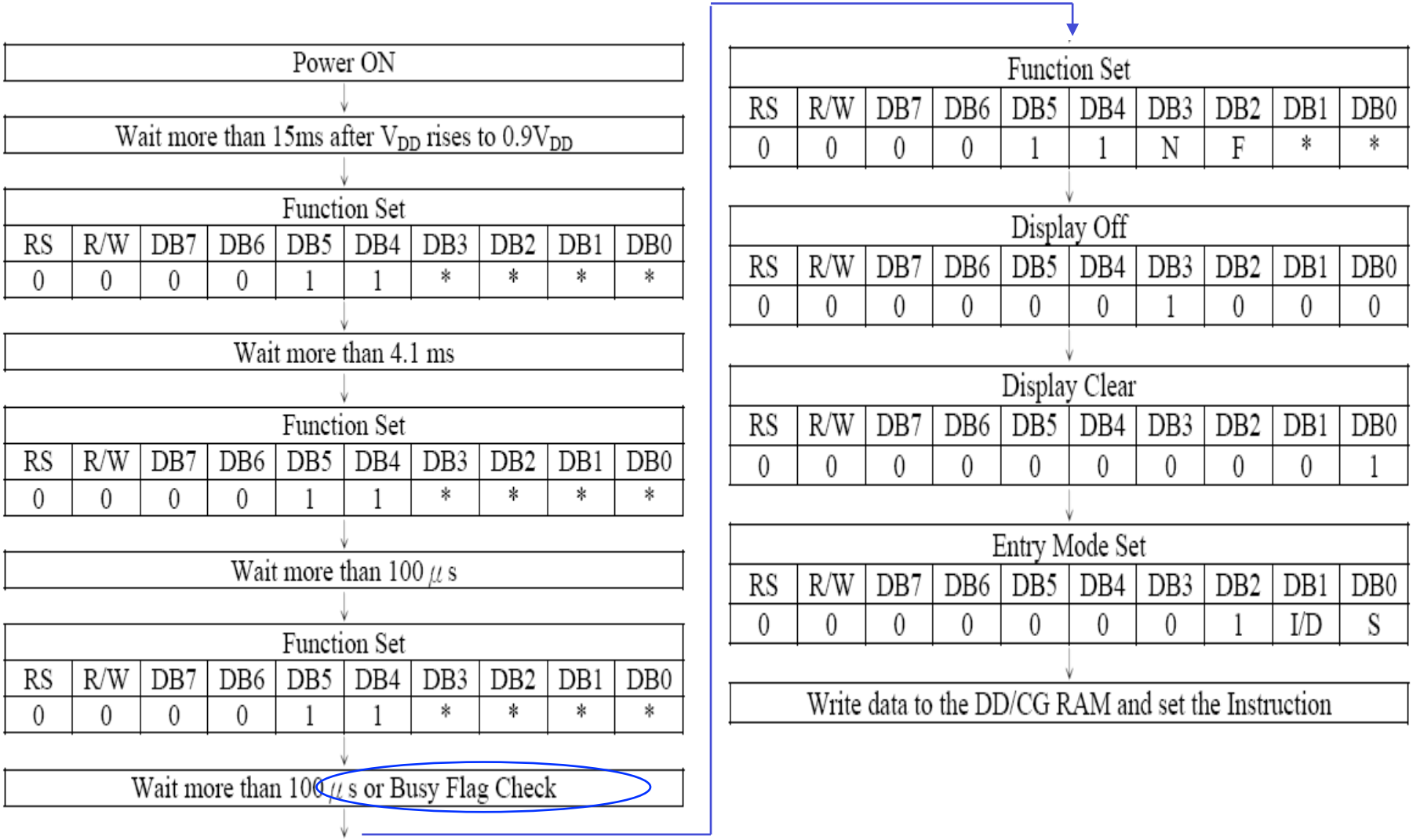
Instruction	Instruction Code										Description	Execution time (fosc= 270 kHz)
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM and set DDRAM address to "00H" from AC	1.53 ms
Return Home	0	0	0	0	0	0	0	0	1	-	Set DDRAM address to "00H" from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed.	1.53 ms
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	SH	Assign cursor moving direction and enable the shift of entire display.	39 µs
Display ON/ OFF Control	0	0	0	0	0	0	1	D	C	B	Set display(D), cursor(C), and blinking of cursor(B) on/off control bit.	39 µs
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	-	-	Set cursor moving and display shift control bit, and the direction, without changing of DDRAM data.	39 µs

MICROCHIP
Regional Training Centers

LCD Commands [2]

Function Set	0	0	0	0	1	DL	N	F	-	-	Set interface data length (DL: 8-bit/4-bit), numbers of display line (N: 2-line/1-line) and, display font type (F:5×11dots/5×8 dots)	39 μs
Set CGRAM Address	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Set CGRAM address in address counter.	39 μs
Set DDRAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set DDRAM address in address counter.	39 μs
Read Busy Flag and Address	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Whether during internal operation or not can be known by reading BF. The contents of address counter can also be read.	0 μs
Write Data to RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Write data into internal RAM (DDRAM/CGRAM).	43 μs
Read Data from RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Read data from internal RAM (DDRAM/CGRAM).	43 μs

LCD 操作於 8-bit 模式的初始流程



MICROCHIP
Regional Training Centers

PMP Data Write & Read

- 決定要寫入的位址
  - PMPADDR = ????
- 將資料寫入 PMPIN – 將資料輸出
  - PMPIN = ????
- 由 PMP data bus 讀取資料(注意Dummy R/W)
  - Var1 = PMPIN
- Wait State ? → 如果 CPU 的速度過快
  - PIC32 的執行速度與 LCD 的 Timing是否須 Wait State 配合 ?

練習 4：PMP_Exercise 的內容

- PMP 的初始程式已經完成
  - C:\RTC\MCU4101T\Lab4 PMP\Lab4 PMP.mcp
- LCD 的相關副程式也已經完成
  - 檔案名稱：LCM.c
  - Head Files：LCM.h
- LCD Operation Mode：
  - 2 lins , 8-bit bus , 5X7 character

MICROCHIP
Regional Training Centers

C32 裡常數的宣告

- 宣告一個有初值的陣列需放在 **Flash Memory** 做查表之用

Examples：

- **const unsigned char MyName[] = " I am PIC32MX795 ";**
- **const unsigned char * pname;**
  - 宣告一指向 **ROM** 的指標

已完成的 LCD Functions (LCM.c)

- void LCM_Init( )
  - 將 LCD 規劃於指定模式
- int LCM_IsBusy( )
  - 讀取 LCD 的 BUSY Flag 並傳回狀態
    - 1 = Busy 0 = Not Busy
- void LCM_PutASCII(unsigned char)
  - 將傳入的指定字元輸出至 LCD
- Void LCM_SetCursor(char X , char Y)
  - 將 LCD 的游標重新設定於 (X,Y)

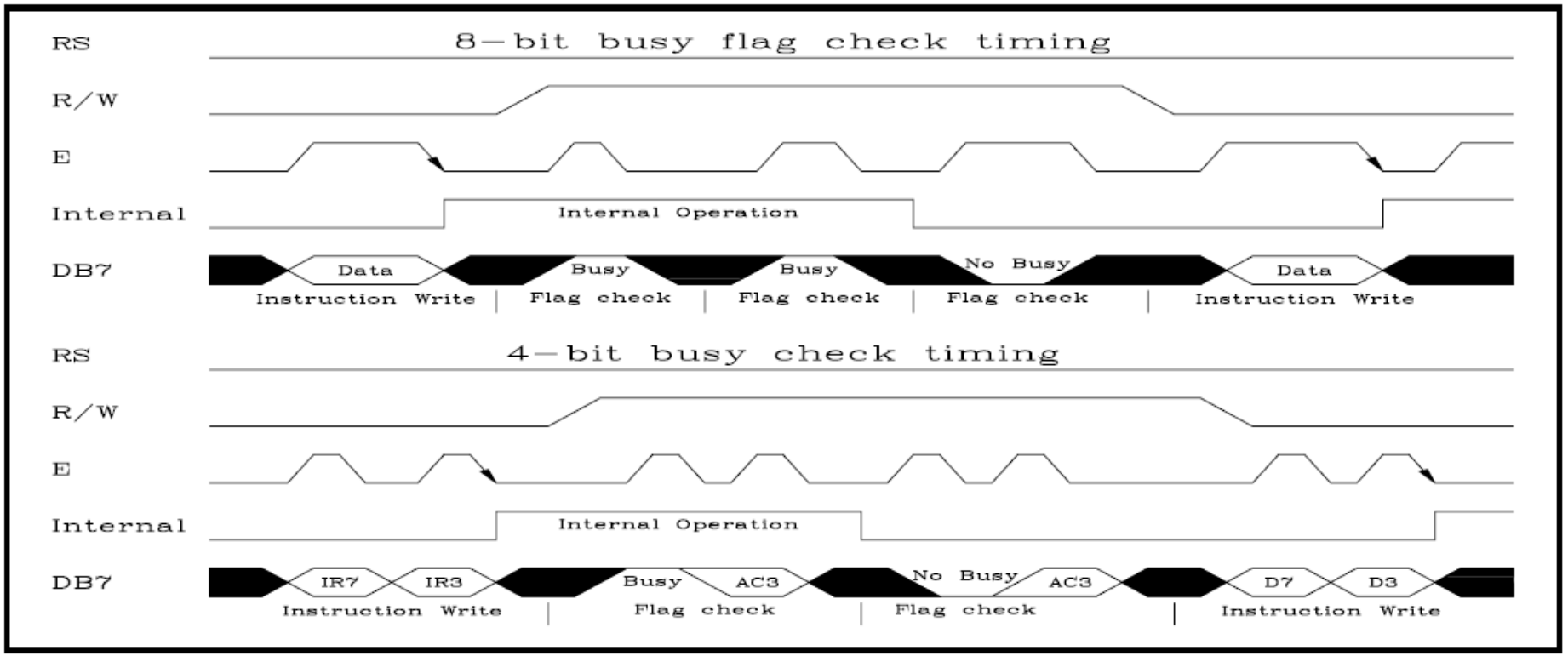
已完成的 LCD Functions (LCM.c)

- LCM_PutROMString(const unsigned char *String)
  - 將 const data memory 的字串由 LCD 印出
- LCM_PutRAMString(unsigned char *String)
  - 將 data memory 的字串由 LCD 印出
- LCM_PutHex(unsigned char Hex)
  - 將變數轉為十六位元的方式由 LCD 印出
- 你可能須要其它的 functions 來輔助以上功能，這些 function 可自行命名

MICROCHIP
Regional Training Centers

LCD 命令的確認

- 雖然 LCD 的每一 command 都有確定的執行時間，但檢查 Busy Flag 是較有效率的方法？Why？



MICROCHIP
Regional Training Centers

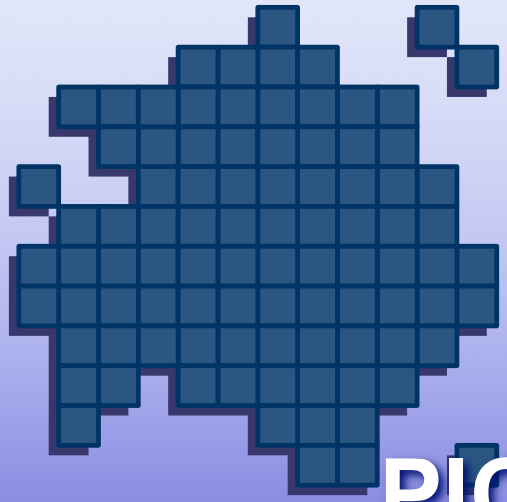
LCM_IsBusy( ) 的參考寫法

```
unsigned char LCM_IsBusy( void )
{
    unsigned char Temporary;
    // Clear the PMPIF
    PMPSetAddress( LCM_INSTRUCTION );
    Temporary = mPMPMasterReadByte( );   // Dummy Read
     for trigger PMP active  ( 讀走的是上一次狀態 )
    // Check The PMPIF
    Temporary = mPMPMasterReadByte( );
    // 第二次讀取，才是現在的狀態反應

    return ( ( Temporary >> 7 ) & 0x01 );
}
```

MICROCHIP
Regional Training Centers

Lab4_PMP.c 的重點

- 宣告儲存姓名的字串
  - **MyName[ ]="I am PIC32MX795" ;**
- 宣告指向與所操作字串型別相同的指標！
- 在 **main( ) 建立PMP Module**的初始化
- 在 **main( ) PMP Module**的初始化後呼叫 **LCM_Init( )** 來對 **LCD** 做初始化
- 在 **main( )** 中呼叫 **LCM.c** 中的副程式來顯示字串
  - **LCM_PutASCII()**
  - **LCM_IsBusy()**
  - **LCM_SetCursor()**
  - **LCM_PutROMString()**
  - **LCM_PutRAMString()**



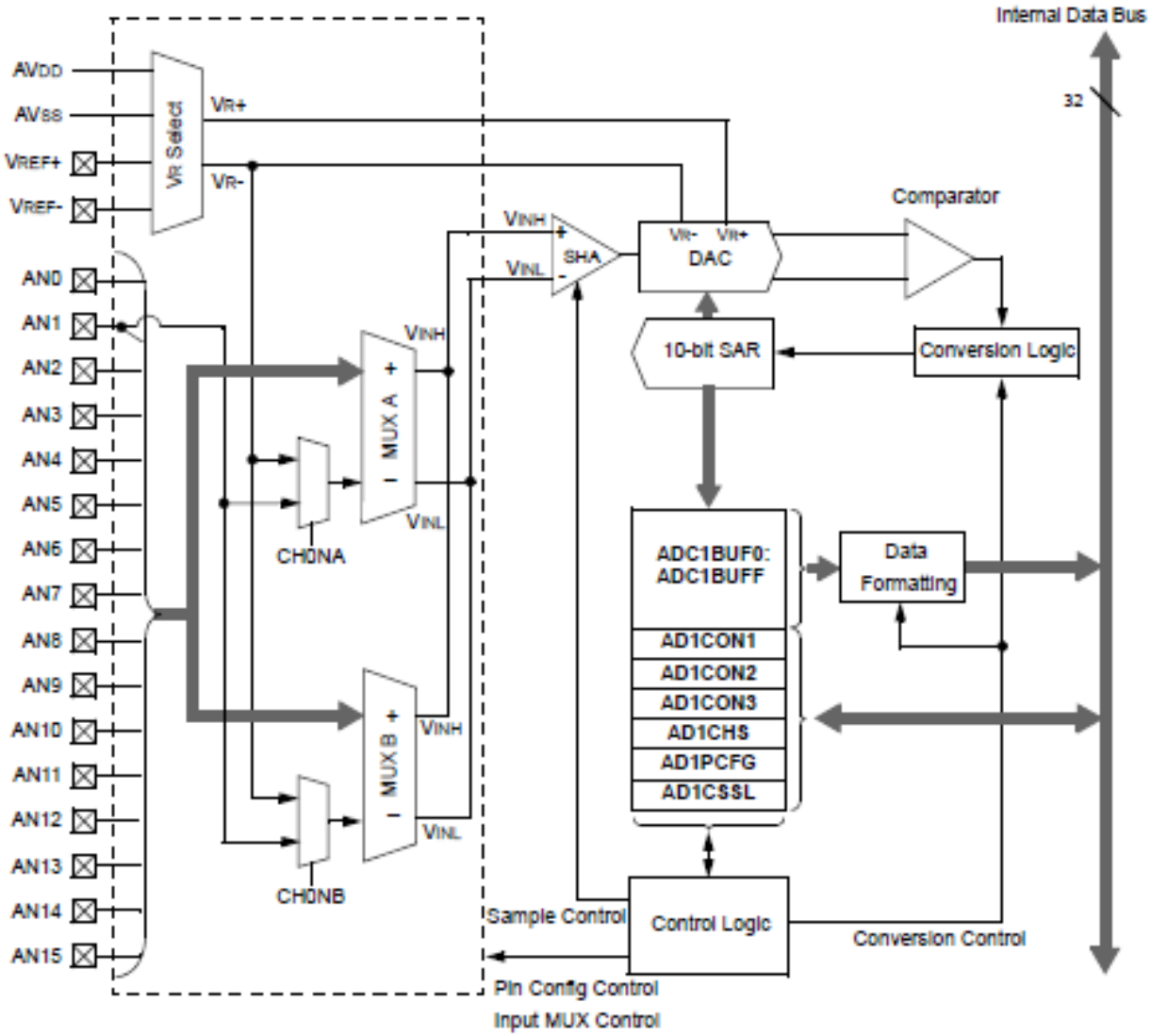
PIC32 的 ADC Module

請參考 PIC32 Family Reference Manual

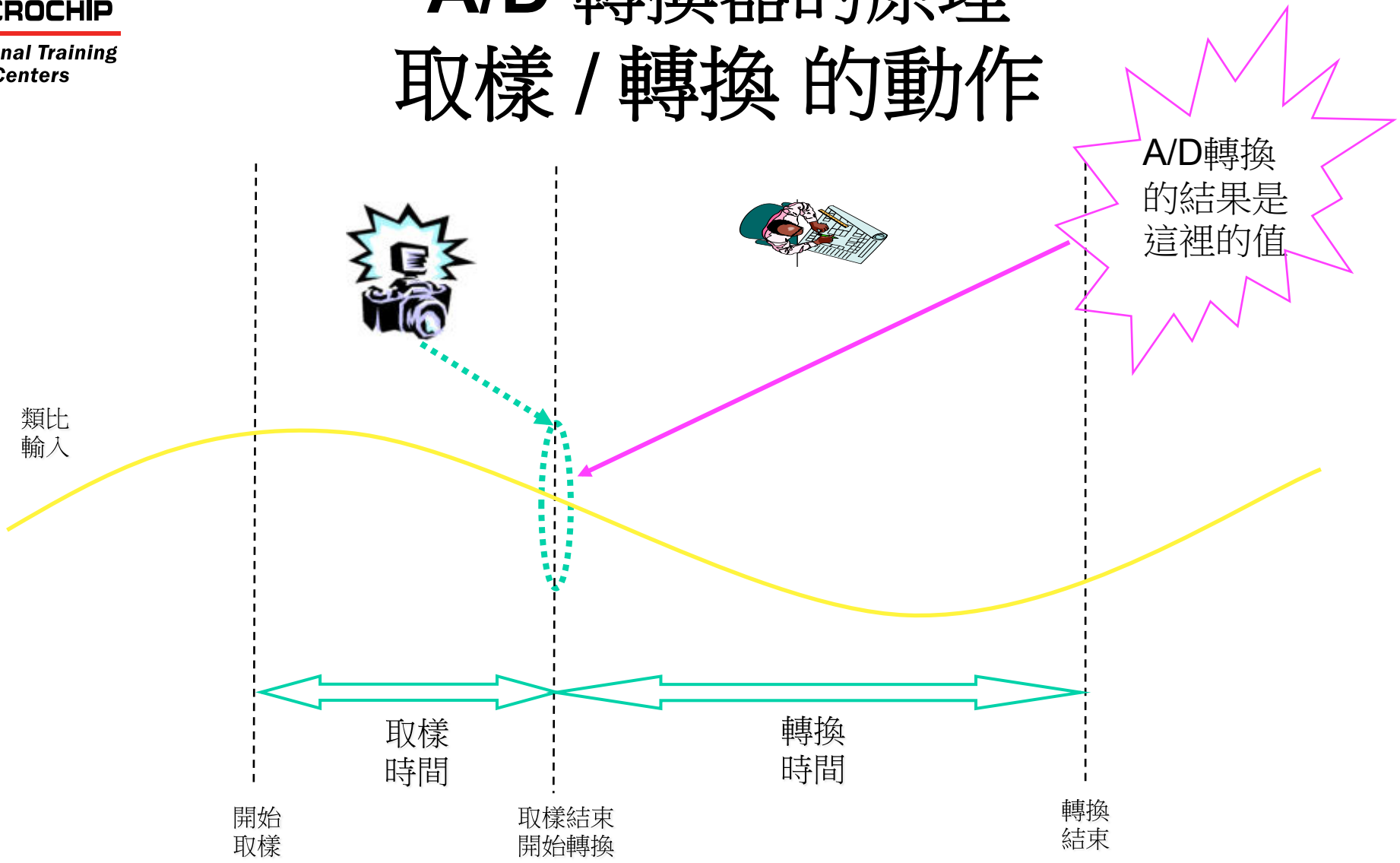
第 17 章

MICROCHIP
Regional Training Centers

PIC32 ADC 的基本構造



A/D 轉換器的原理
取樣 / 轉換 的動作



MICROCHIP
Regional Training Centers

PIC32 ADC 的重點整理 (1)

- 輸入的類比信號以 **Vref+ & Vref-** 為最高級最低的參考點，將之分割為 **2n** 的刻度
  - 以 **10-bit** 解析度而言，可分割為 **1024** 個刻度
  - **Vref+ & Vref-** 可以外加或使用內部的 **VDD & VSS**
- **PIC32** 只有一組 **ADC**，但因為有類比多工器的切換，所以可以支援多組的 **ADC** 轉換
- **PIC32** 的 **ADC** 使用 **SAR (連續逼近法)** 的轉換方式，所以轉換的時脈 **TAD** 為設定的重點
- **PIC32** 的 **ADC** 有兩組多工器，可彈性選用
- **PIC32** 的 **ADC** 若選用 **A** 多工器，可達到自動掃描的功能

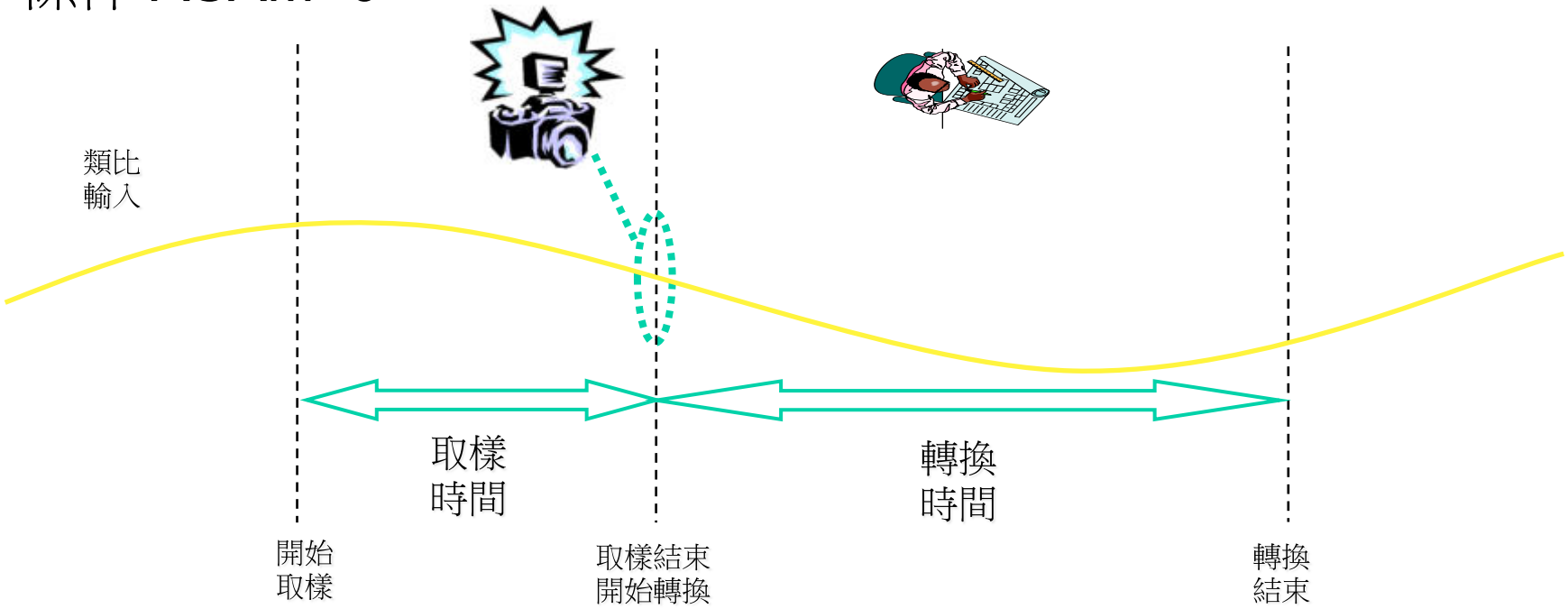
MICROCHIP
Regional Training Centers

PIC32 ADC 的重點整理 (2)

- PIC32 ADC 的輸出格式比 PIC24F 多，共有 8 種不同格式
- PIC32 ADC 內建有 16 個 32-bit 的 Buffer
  - ADC1BUF0 ~ ADC1BUFF
- PIC32 的 ADC 可選擇中斷 CPU 的轉換次數
- 常用的 ADC macros
  - OpenADC10( )
  - EnableADC10( )
  - SetChanADC10( )
  - ConvertADC10( )
  - BusyADC10( )
  - ReadADC10( )

A/D 轉換器最基礎的轉換方式
手動取樣 / 轉換步驟

條件 ASAM=0



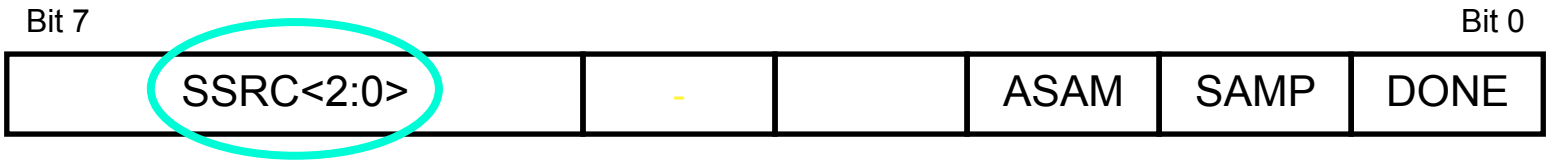
設定
SAMP=1

清除
SAMP=0

檢查
DONE=1 ?

ADC 的轉換觸發方式

AD1CON1



SSRC<2:0> - 啟動AD轉換的觸發信號來源選擇

111 = 使用內部時序設定取樣時間及轉換時間(自動轉換)

(需參考到 AD1CON3 暫存器的設定)

010 = Timer 3 計時比較完成後，結束取樣ADC開始轉換

001 = INT0 腳位電位轉態時，結束取樣ADC開始轉換

(需參考到 INTCON2 <INT0EP> 位元的設定)

000 = 清除 SAMP 位元後轉換 (手動轉換)

MICROCHIP
Regional Training Centers

PIC32 ADC 使用前注意事項

- 使用 **AD1PCGF** 設定類比輸入腳位

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PCFG15	PCFG14	PCFG13	PCFG12	PCFG11	PCFG10	PCFG9	PCFG8
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
PCFG7	PCFG6	PCFG5	PCFG4	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	P = Programmable bit	r = Reserved bit
U = Unimplemented bit	-n = Bit Value at POR: ('0', '1', x = Unknown)		

bit 31-16 **Reserved:** Write '0'; ignore read

bit 15-0 PCFG<15:0>: Analog Input Pin Configuration Control bits

1 = Analog input pin in Digital mode, port read input enabled, ADC input multiplexer input for this analog input connected to AVss

0 = Analog input pin in Analog mode, digital port read will return as a '1' without regard to the voltage on the pin, ADC samples pin voltage

MICROCHIP
Regional Training Centers

練習五 – ADC 轉換結果的讀取與顯示

- 練習五須完成的工作
  - C:\RTC\MCU4101T\Lab5 ADC Single\Lab5 ADC Single.mcp
    - 使用練習 4 的 PMP 範例程式為起點
    - 將 ADC 設定為自動取樣與手動轉換
    - 自行定義時間然後完成 VR1 的讀取，並將其以 0 ~ 1023 的值顯示於 LCD 的第二行
    - 顯示的結果為 VR1 = xxxx
  - 注意事項：在 APP1632 的 VR1 接到的是 PICtail Plus 中 AN5/RB5 的腳位，但 APP1632-2 的 PIC32 PIM Module 已經依照美國實驗板的跳接將 CPU 的 AN2/RB2 跳接至 PICtail Plus 介面的 AN5/RB5
    - 所以 VR1 的正確接線為接至 PIC32MX795 的 AN2/RB2

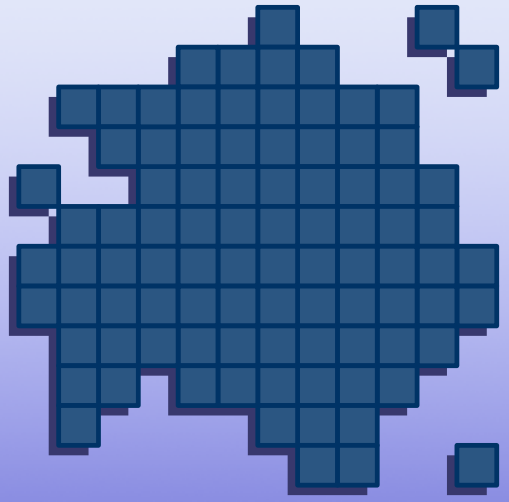
練習六 – ADC 轉換結果的讀取與顯示

- 練習六須完成的工作
  - C:\RTC\MCU4101T\Lab6 ADC Scan\ Lab6 ADC Scan.mcp
    - 修改練習5或使用練習4的PMP範例程式為起點
    - 將 ADC 設定為自動取樣與手動轉換，並自動掃描VR1, VR21。並將VR1, VR2 讀值以 0 ~ 1023 的值顯示於 LCD 的第二行
    - 顯示的結果為 VR1 = xxxx, VR2 = xxxx

MICROCHIP
Regional Training Centers

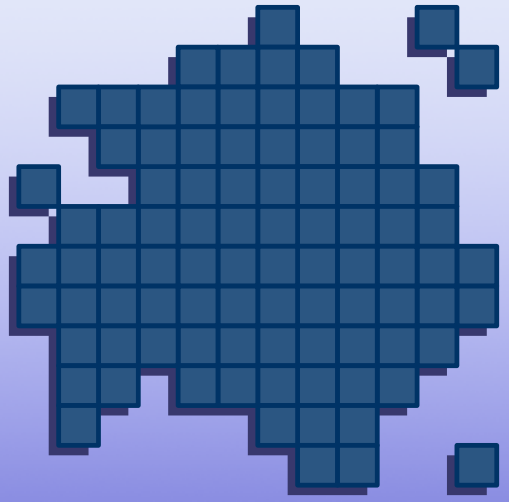
練習7 ADC Timer觸發轉換與顯示

- 練習7須完成的工作(請以練習5為範本)
  - 將ADC設定為自動取樣與TIMER觸發轉換。
  - 開啟ADC中斷，並設定優先權。
  - 設定正確的AD通道及IO狀態。
  - 設定並開啟TIMER3 。
  - 建立AD中斷服務常式，並於AD中斷服務常式中讀取AD轉換數值，接著透過自訂旗標指明AD數值讀取狀態。
  - 於主程式中輪詢該旗標，並將VR1讀值以 0 ~ 1023 的方式顯示於 LCD 的第二行。



課程結束！

以下為參考資料請自行研讀



PIC32 的 UART Module

請參考 PIC32 Family Reference Manual 第 17 章

(Options)

MICROCHIP
Regional Training Centers

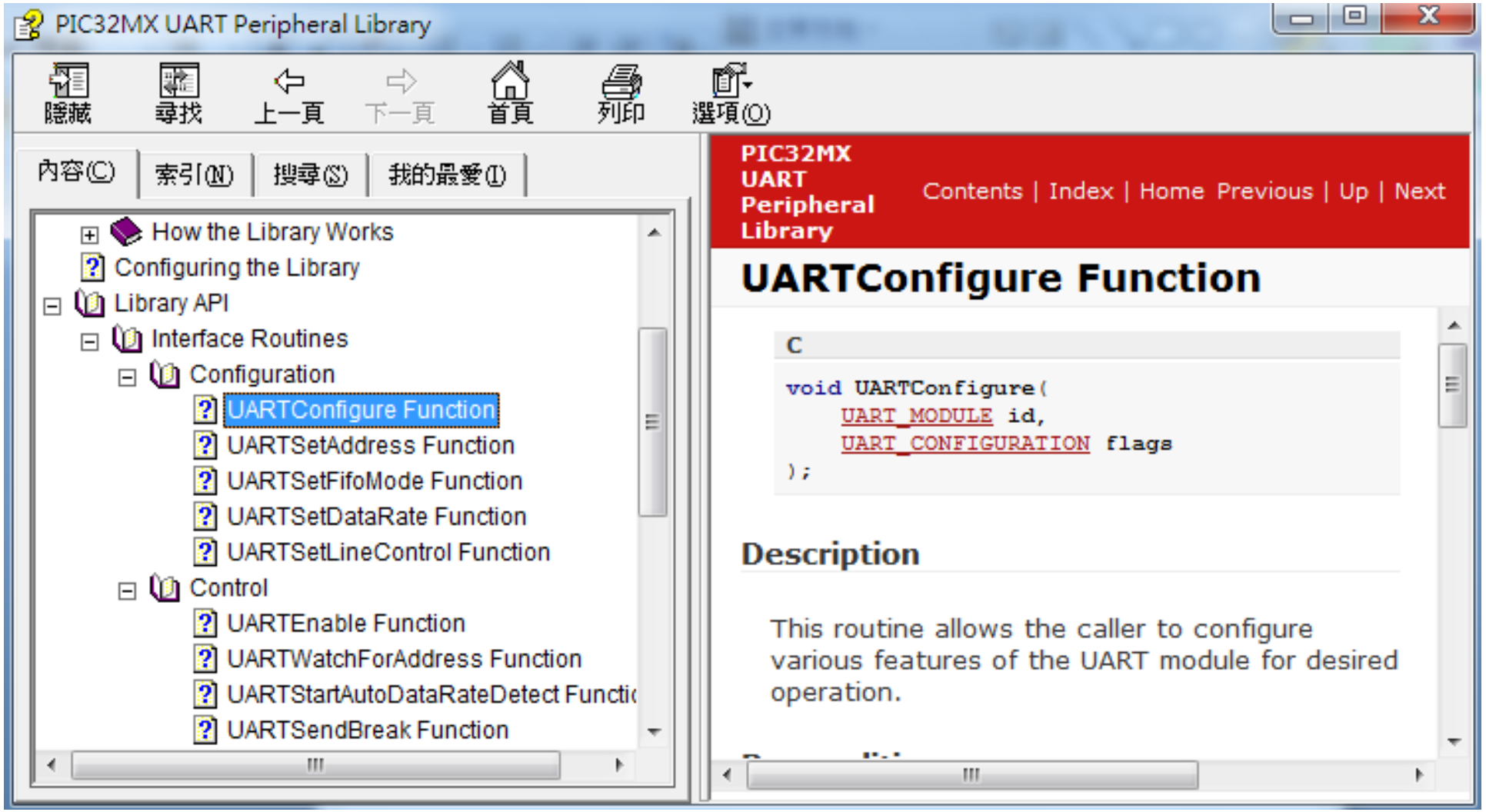
PIC32 的 UART module

- PIC32 的 UART module 與 PIC24F 的 module 幾近相同
- MPLAB C32 的周邊 Library 對 UART 有完整的 support
- UART 的 Library 在新版的 MPLAB C32 做了大幅的修改，可參考下列目錄中的檔案
  - C:\Program Files\Microchip\MPLAB C32 Suite\doc\pic32-lib-help

MICROCHIP
Regional Training Centers

MPLAB C32 新的 UART Library

- 參考 UART Peripheral Library 的說明



MICROCHIP
Regional Training Centers

通信 Baud Rate 的設定(1)

- **BRGH = 0 的計算公式**

EQUATION 19-1: UART BAUD RATE WITH BRGH = 0(1)

$$Baud\ Rate = \frac{FPB}{16 \bullet (UxBRG + 1)}$$

$$UxBRG = \frac{FPB}{16 \bullet Baud\ Rate} - 1$$

Note 1: FPB denotes the peripheral bus clock frequency.

MICROCHIP
Regional Training Centers

通信 Baud Rate 的設定(2)

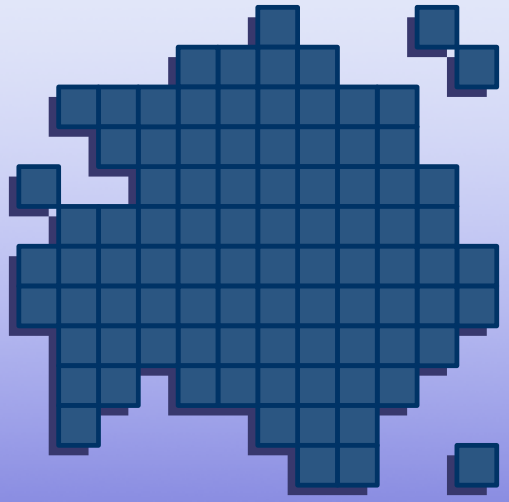
- **BRGH = 1 的計算公式**

EQUATION 19-2: UART BAUD RATE WITH BRGH = 1(1)

$$Baud\ Rate = \frac{FPB}{4 \bullet (UxBRG + 1)}$$

$$UxBRG = \frac{FPB}{4 \bullet Baud\ Rate} - 1$$

Note 1: FPB denotes the instruction cycle clock frequency.

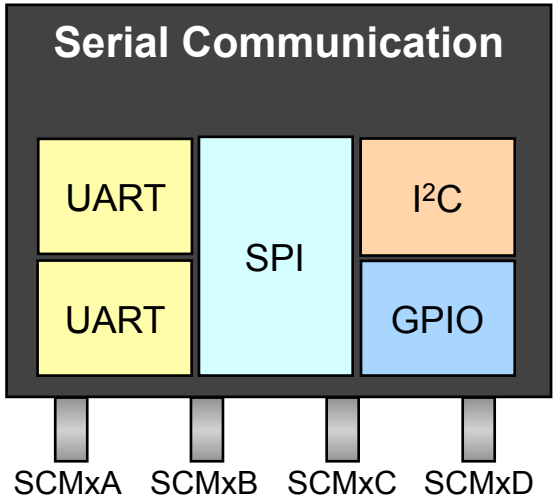


Serial Communication Modules in PIC32MX5/6/7

MICROCHIP
Regional Training Centers

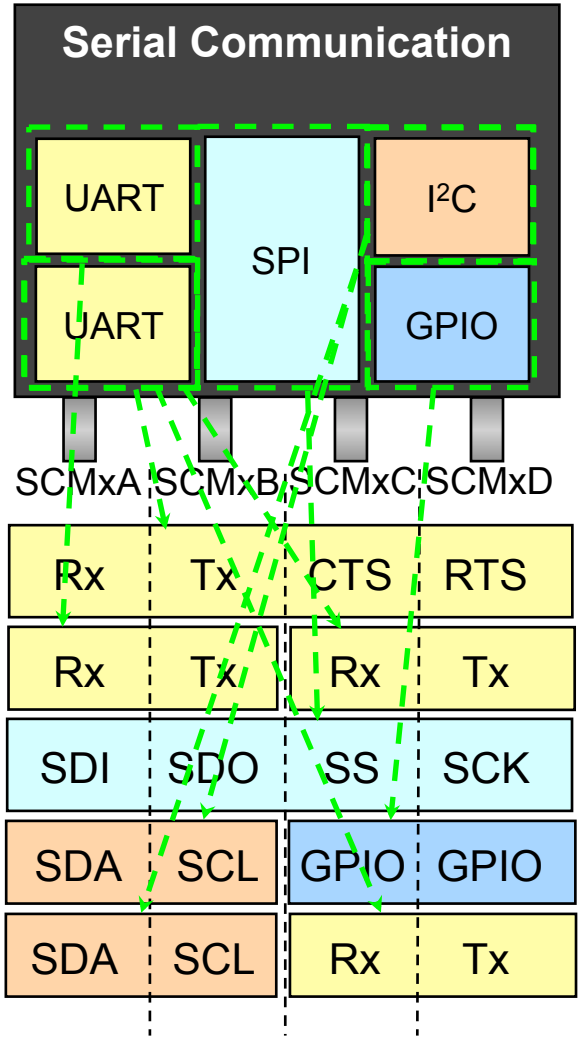
Serial Communications

- Combined UART, SPI and I²C
- User selectable serial communications
  - More serial communication peripherals available
  - Fewer muxing conflicts with other peripherals
  - Each module is configurable and has 4 i/o pins
- 3 Instantiations on each part



MICROCHIP
Regional Training Centers

Configuration Options…

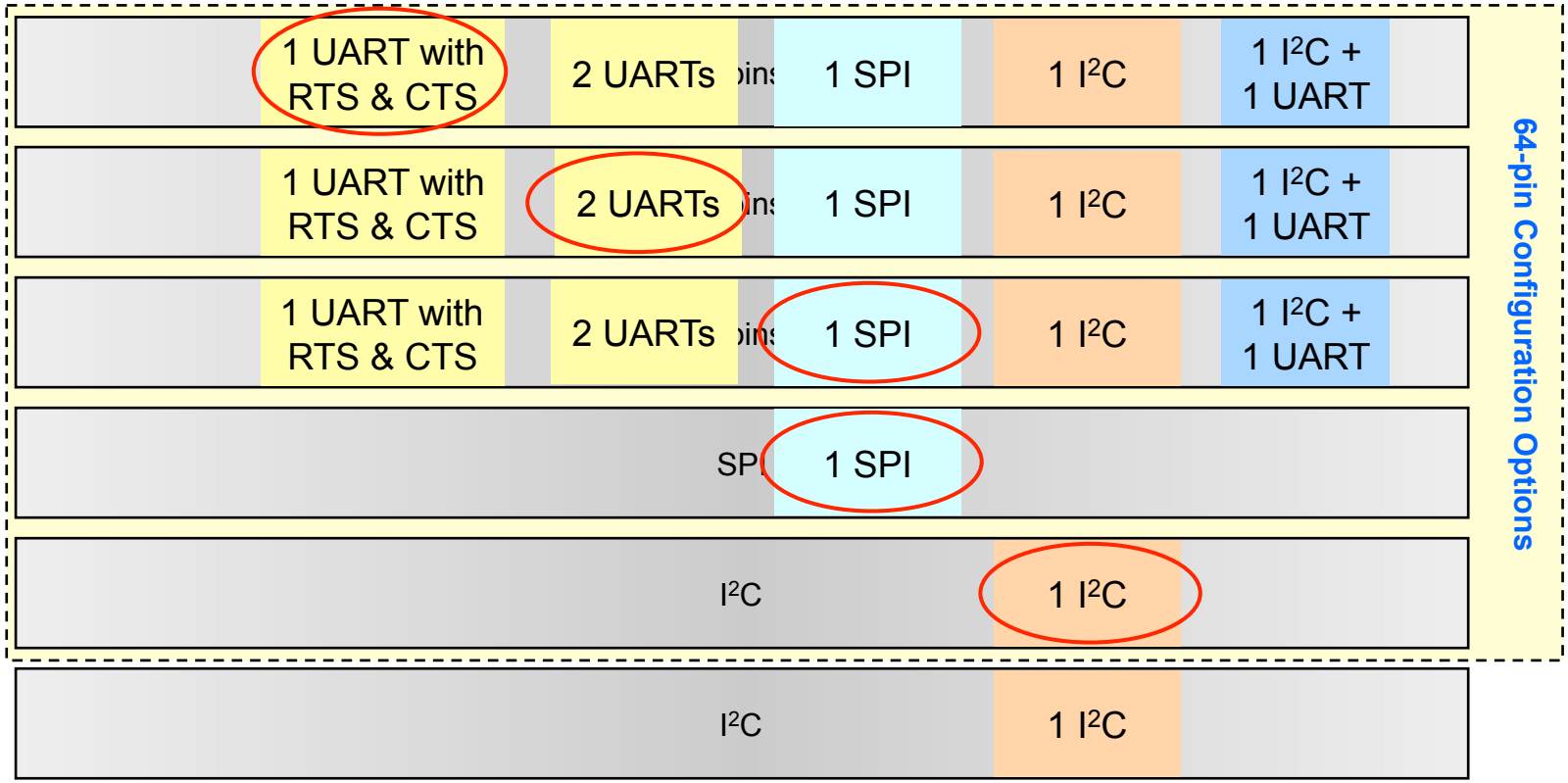


- 1 x UART with Flow Control
- 2 x UARTs no Flow Control
- 1 x SPI
- 1 x I²C with GPIO
- 1 x I²C + 1 UART no Flow Control

User Selectable Combinations of Peripherals

Example Configurations

(Select 1 function block from each row)

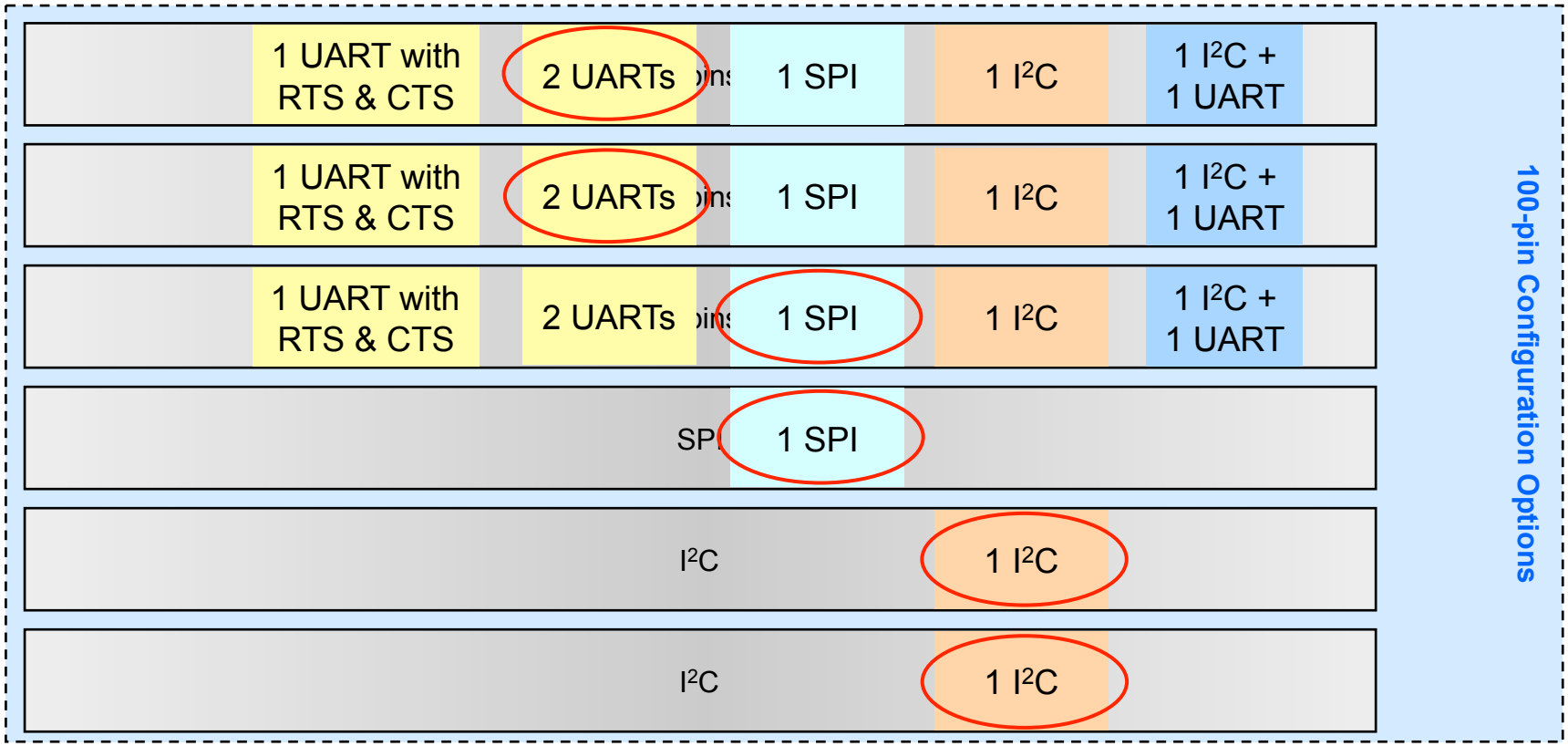


Example 1 = 1 UART (with flow control), 2 UARTs, 2 SPI, 1 I²C (64-pin)

User Selectable Combinations of Peripherals

Example Configurations

(Select 1 function block from each row)



Example 2 = 4 UARTs, 2 SPI, 2 I²C (100-pin)

MICROCHIP
Regional Training Centers

練習 八：使用 UART1 來傳送資料

- 使用 **Lab8_UART_TX** 目錄中的 **Project**
- **Init_UART1()** 已經將 **UART1** 設定為 **9600,N,8,1**
  - 使用 **MPLAB C32** 新的 **UART macros**
- 利用練習 **3** 的練習，製作一個能累積 分與秒的小時鐘，然後將時間以 **U3A (UART3A)**輸出至 **PC**
  - 使用 “ 超級終端機 “ 配合對的 **COM Port** 設定來接收資料
  - 格式為：**Time = MM . SS**

MICROCHIP
Regional Training Centers

練習八 的硬體接線

- APP1632 的 UART IC 以 JP3 接至本來 PIC24F & PIC32MX3/4 的 U2TX & U2RX
- 因為 PIC32MX567 系列使用新的 Communication module , 所以原先 PIC32MX3 的 U2TX/U2RX 變成
  - U3ATX & U3ARX
- APP1632 的硬體還是將 JP3 的第 1及第 2 個 Jumer 位置短路
- 練習八 的範例程式使用 U1ATA & U1ARX 來傳送 UART 資料到 PC
- 在練習八 使用 PIC32 的 UART Library 來嘗試完成 UART 傳送的工作

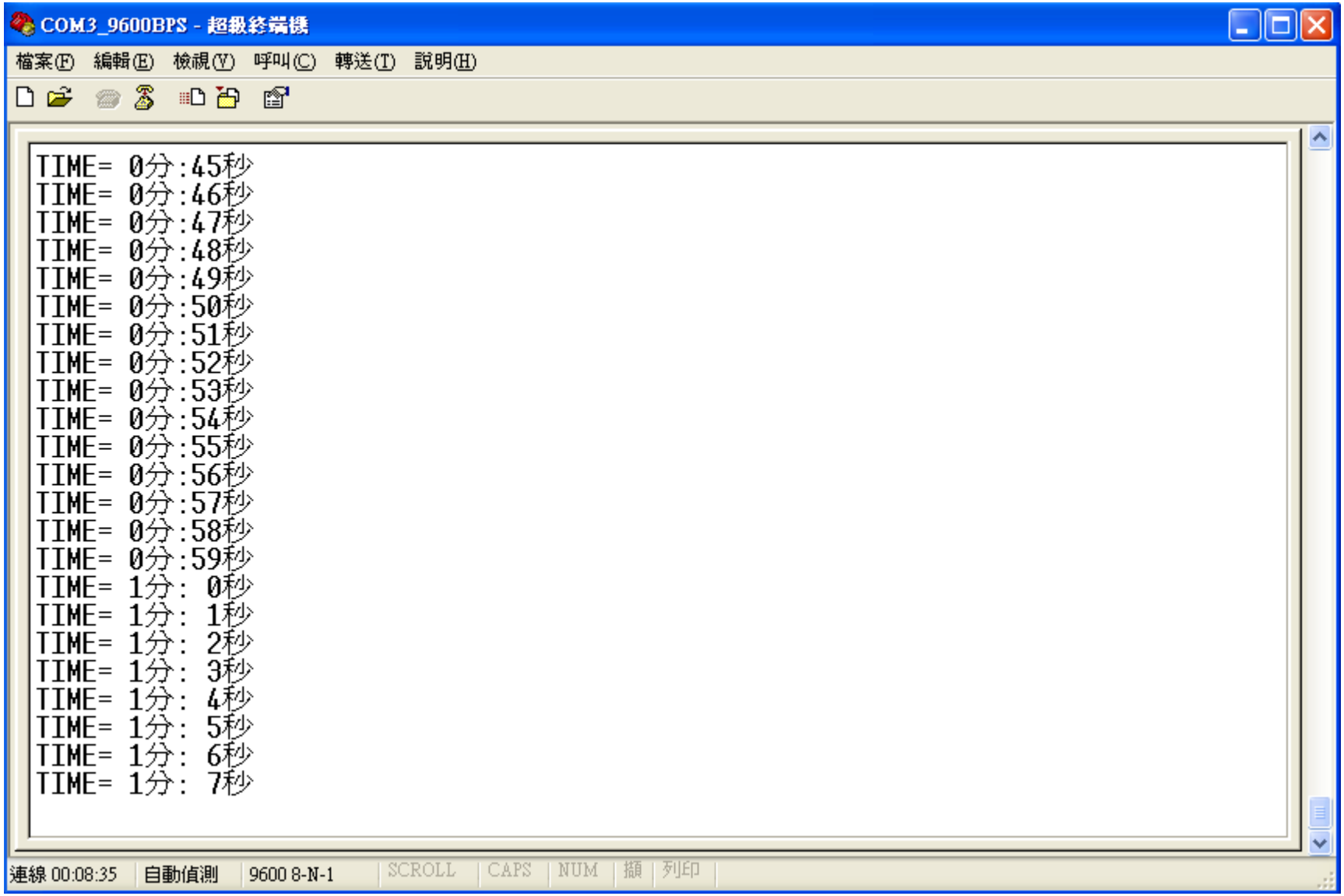
MICROCHIP
Regional Training Centers

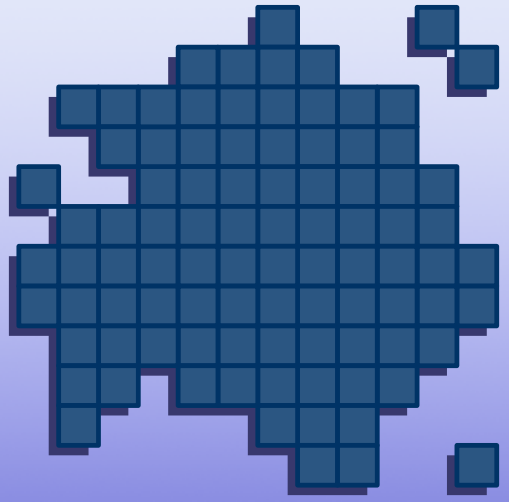
資料轉換及輸出可用的 functions

- sprintf(SprintfBuffer,“Time = %d 分: %d 秒\n\r", Min,Sec) ;
- putsUART1(SprintfBuffer) ;
  - 自行寫一副程式，將資料由 U3A 傳送出去
  - 使用 MPLAB C32 Peripheral Library 中的 UART 函數

MICROCHIP
Regional Training Centers

練習八的執行結果





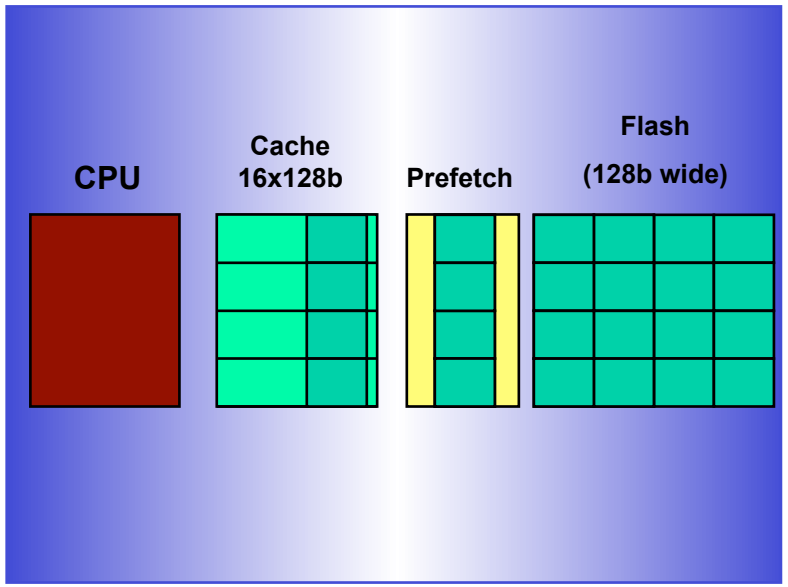
Optimizing Code

参考練習

MICROCHIP
Regional Training Centers

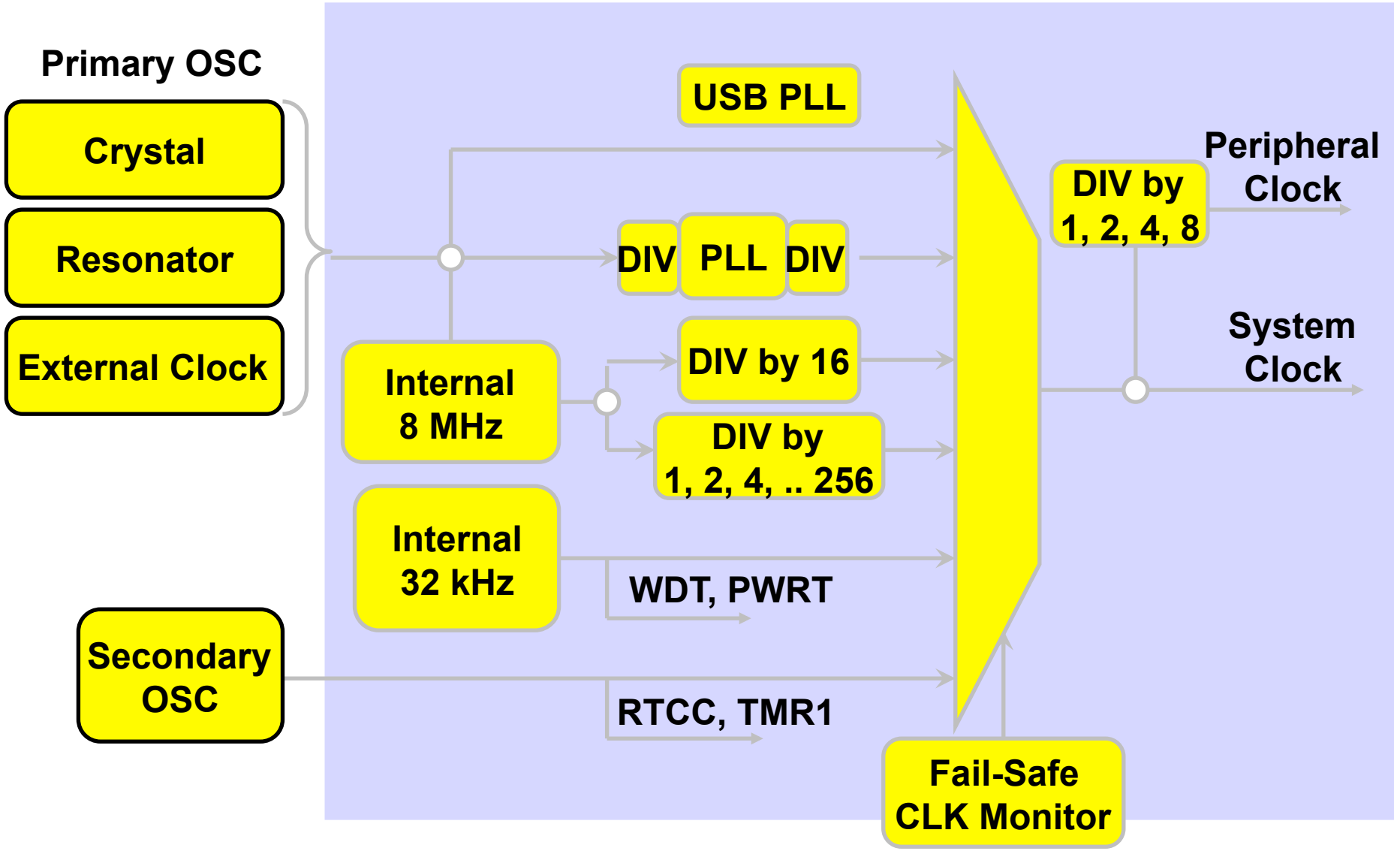
Caching

- Why Cache?
- Fetches 128-bit data
  - 4x32-bit instructions
  - 8x16-bit instructions
- Up to 80 MIPS execution
- 256 Bytes Cache
  - 16 total lines
- Must execute from a cacheable memory segment (KSEG0, KUSEG)
- Enable caching using peripheral lib



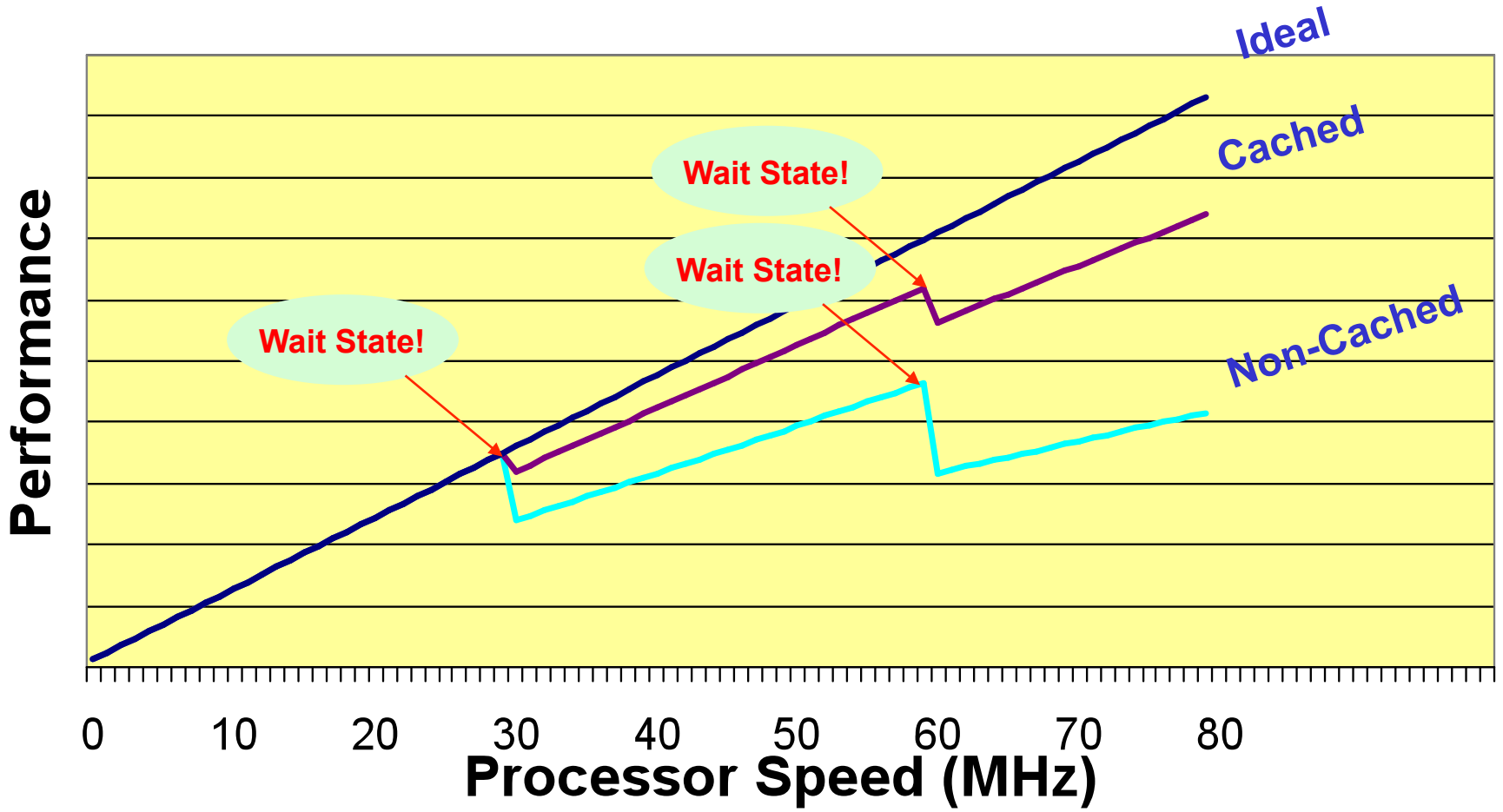
MICROCHIP
Regional Training Centers

Clocking



MICROCHIP
Regional Training Centers

Effect of Prefetch Cache



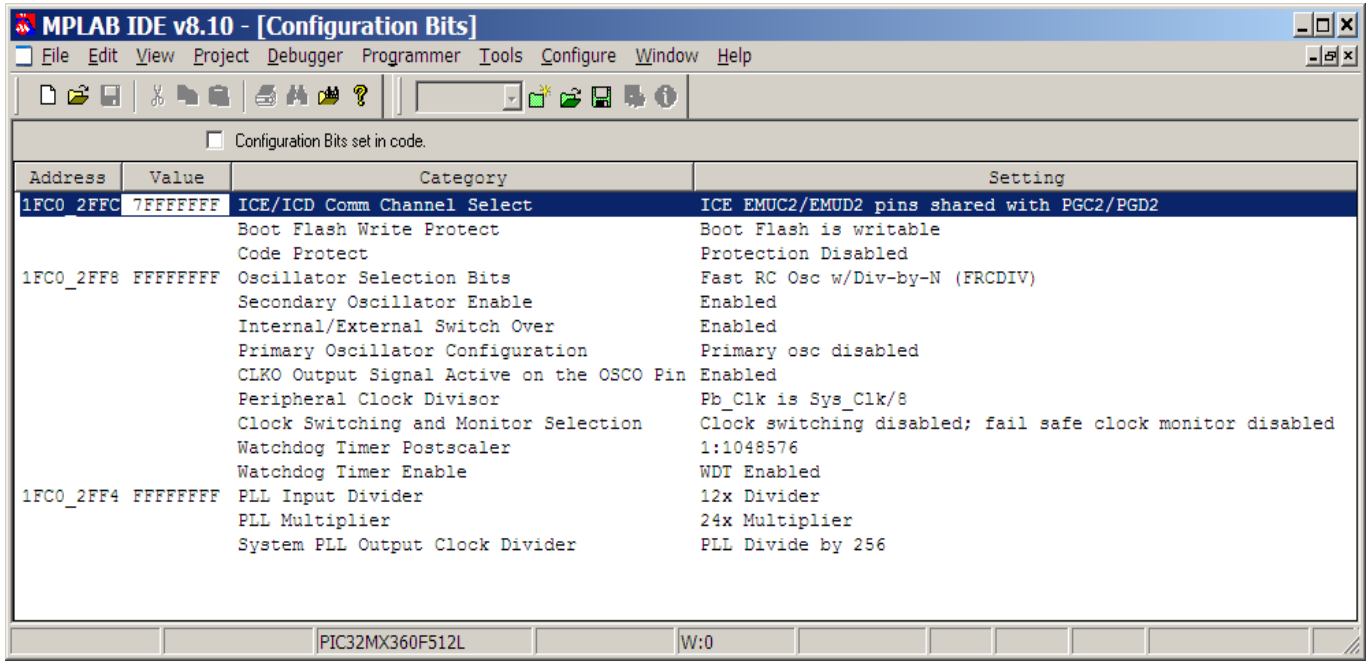
MICROCHIP
Regional Training Centers

Configuration

- **Clocks**
  - System Clock
  - PB Divider
- **Wait States**
  - Flash (0-7)
  - Data RAM (0-1)

PIC32 Configuration Bits

Method 1:



Method 2:

// Configuration Bit Settings

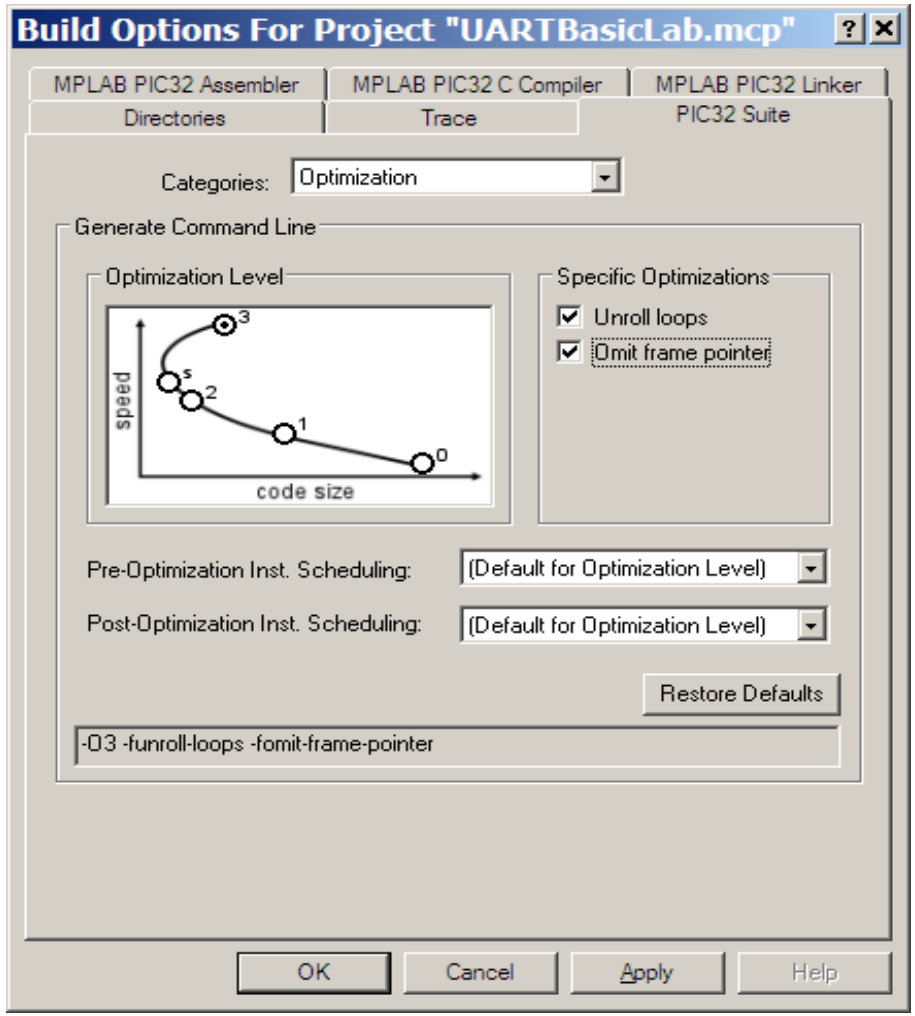
#pragma config FPLLMUL = MUL_20, FPLLIDIV = DIV_2, FPLLODIV = DIV_1
#pragma config POSCMOD = HS, FNOSC = PRIPLL, FPBDIV = DIV_1, OSCIOFNC = OFF, FWDTEN = OFF

MICROCHIP
Regional Training Centers

Compiler Settings

➢ Project Options

- -O3 usually provides best speed
- -Os provides smallest size
- Extra optimization
  - Loop unrolling
  - Omit frame pointer



MICROCHIP
Regional Training Centers

Optimizing Code

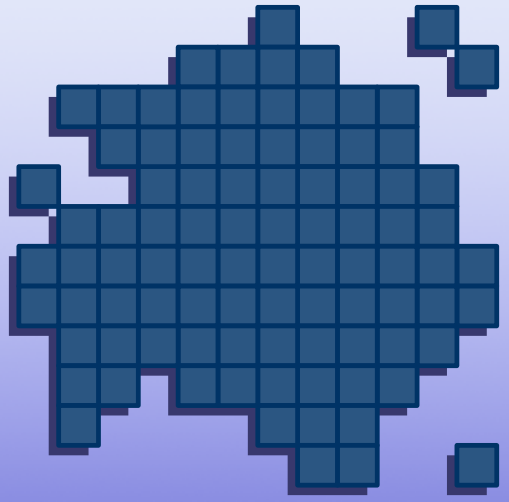
Wrap Up

- ***Default settings are not maximum performance***
- ***Remember the 4 C's:***
  - *Clocks, Cache, Config, and Compiler*
- ***Use Peripheral Library***
  - SYSTEMConfig ( )

MICROCHIP
Regional Training Centers

練習九 Code Optimized

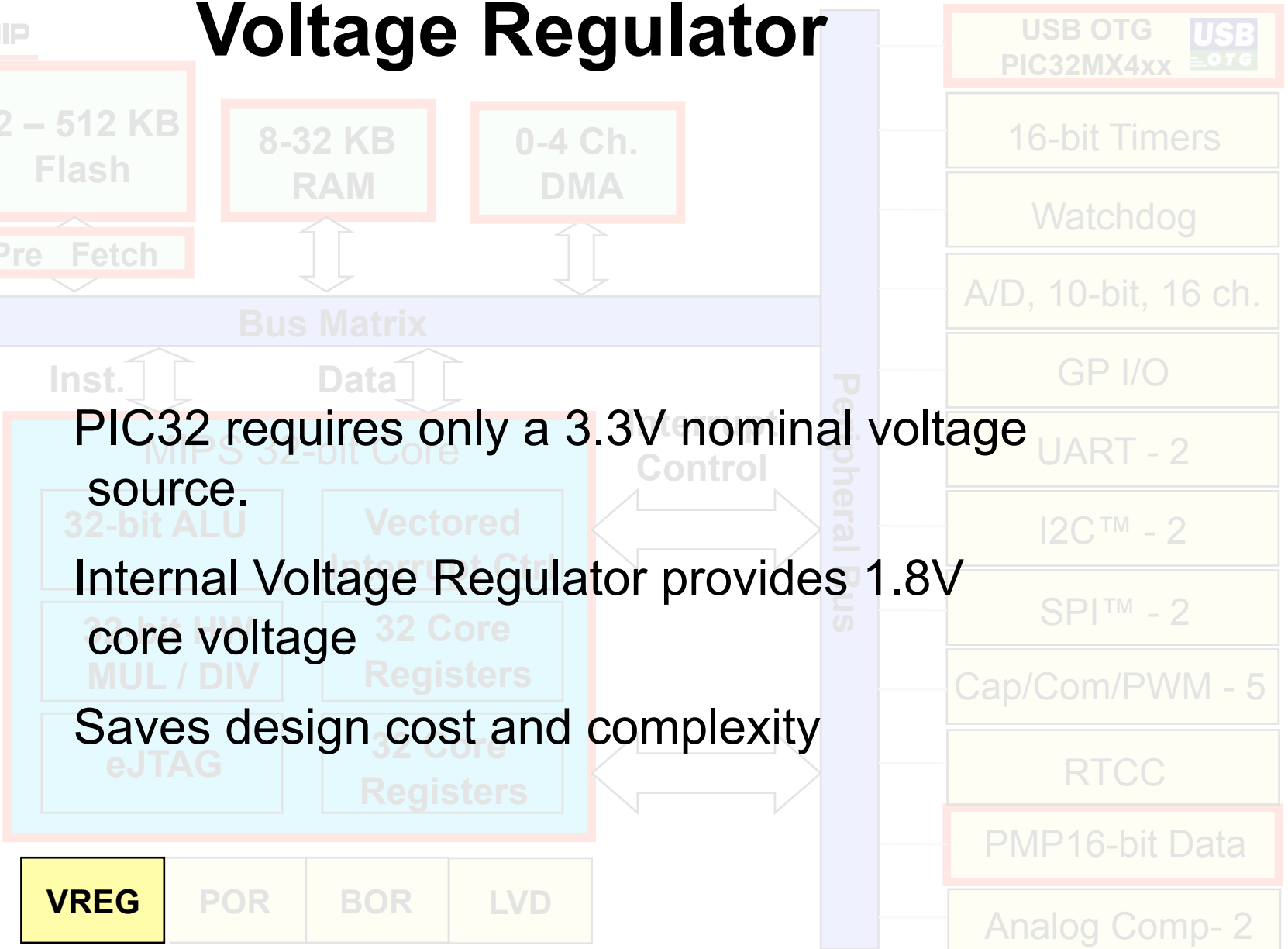
- 練習六提供一個 **SYSTEMConfig()** 的範例
  - 展現 **PIC32** 的 **Pre Cache** 的最佳化後對程式執行時間的影響
  - 使用讀取 **Core Timer** 的方式來得知程式執行所費的 **System Clock** 數量
  - ** **Core Timer** 每 **2** 個 **System Clock** 會加一
- 使用 **Watch Window** 來觀察 **time_in_ms** 變數
  - 使用 **SYSTEMConfig** 前後的數值會有明顯的差異



PIC32 Peripherals

參考資料

Voltage Regulator



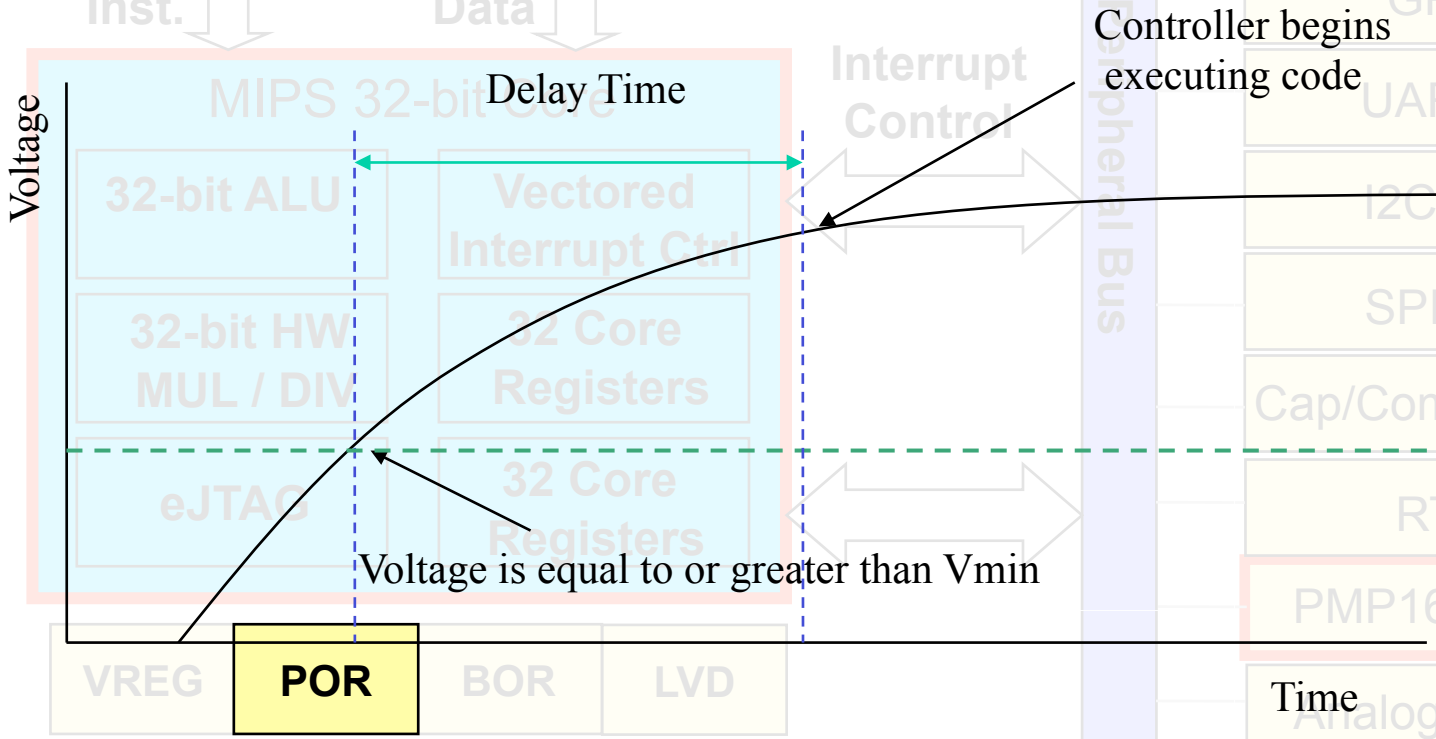
PIC32 requires only a 3.3V nominal voltage source.

Internal Voltage Regulator provides 1.8V core voltage

Saves design cost and complexity

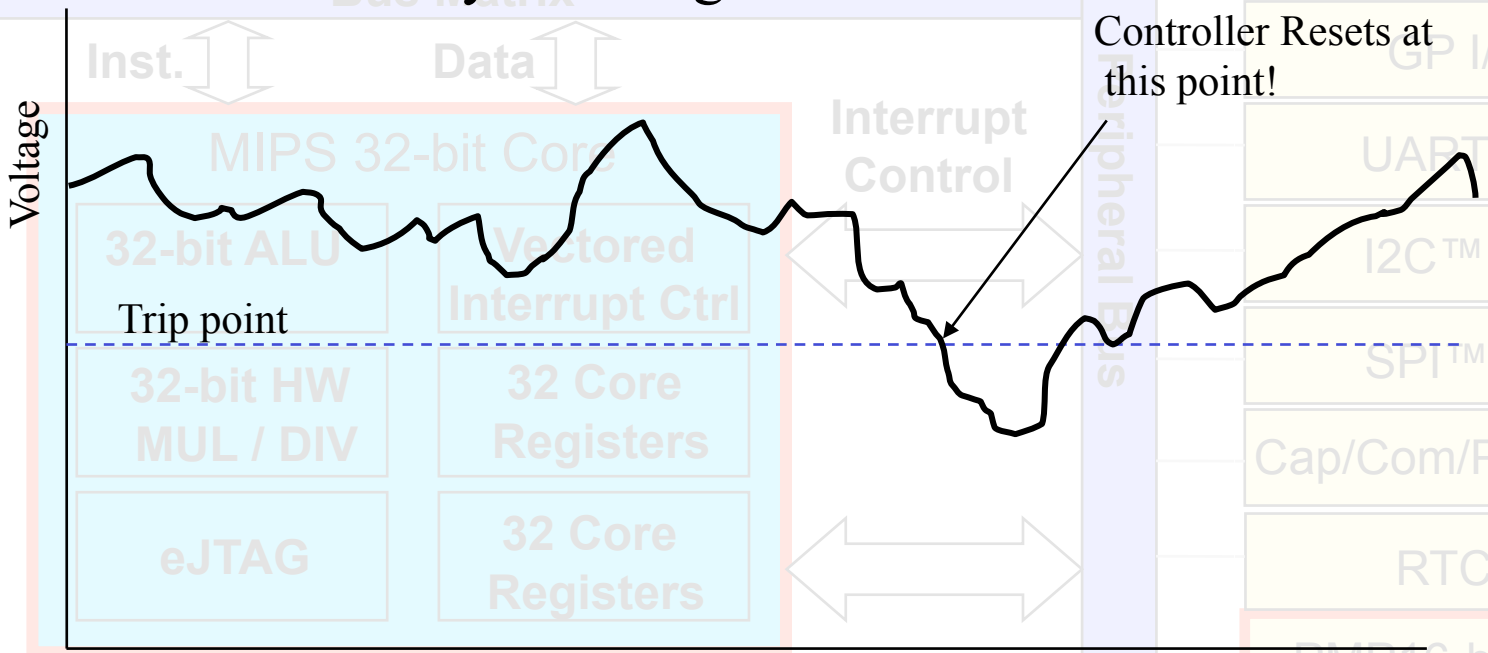
Power On Reset

- **When Supply voltage is first applied, this peripheral holds the device in Reset for a short time, allowing the supply to become stable before allowing the processor to start executing code.**



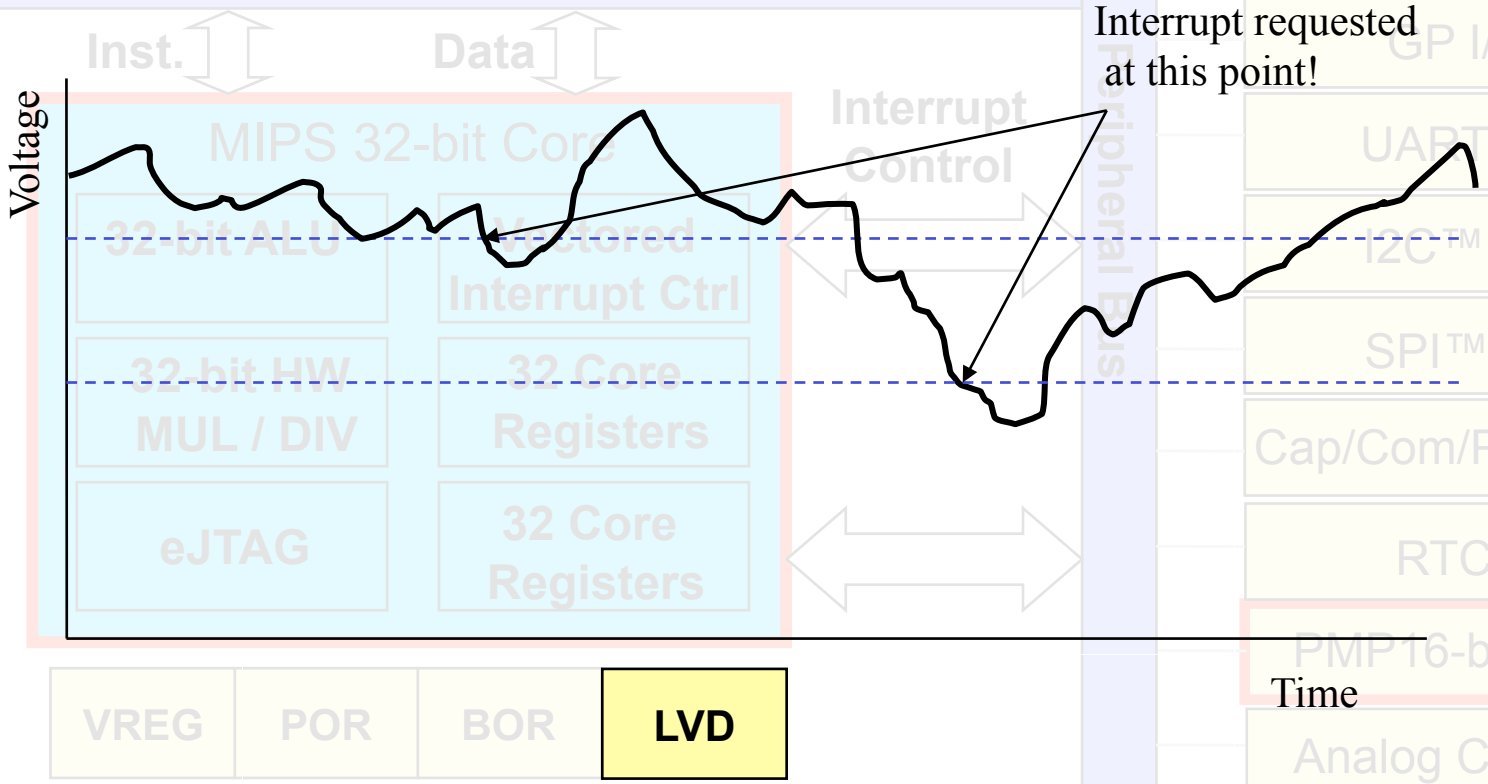
Brown Out Reset

If enabled, this peripheral will reset the microcontroller if the Source Voltage drops below a certain level, or Trip point. We want this to happen, because the program could behave erratically if voltage is bad.



Low Voltage Detect

- Low Voltage Detect and High/Low Voltage Detect
- Monitor Supply Voltage and trigger an Interrupt if voltage crosses the detect value



USB OTG

The PIC32MX USB module includes the following features:

- USB Full-Speed Support for Host and Device
- Low-Speed Host Support
- USB On-The-Go (OTG) Support
- Integrated Signaling Resistors
- Integrated Analog Comparators for Vbus Monitoring
- Integrated USB Transceiver
- Transaction Handshaking Performed by Hardware
- Endpoint Buffering Anywhere in System RAM
- Integrated DMA Controller to Access System RAM and Flash

16 bit Timers

All PIC32 Timers are 16 bit. There are two types, A and B.

All timers have the following features:

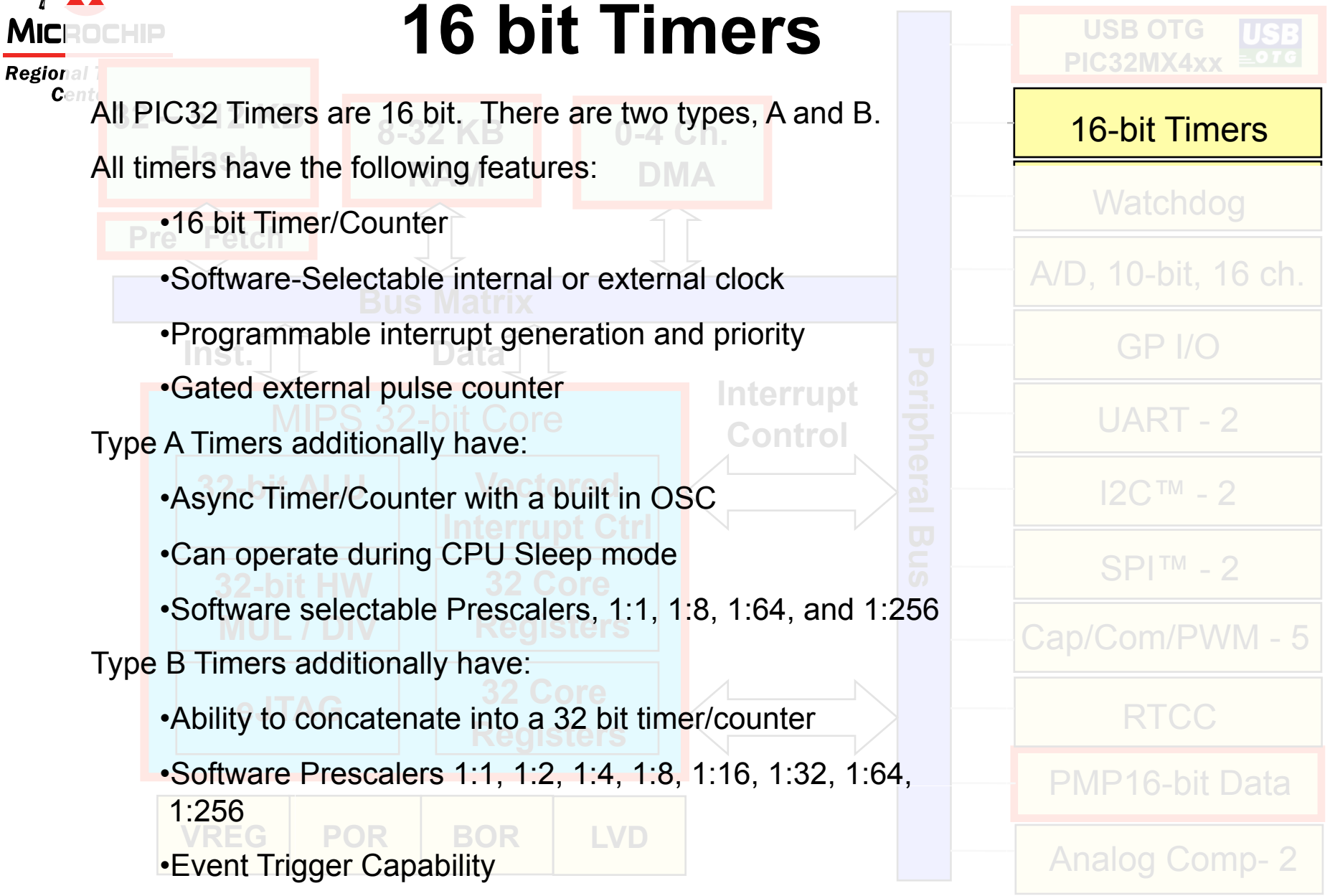
- 16 bit Timer/Counter
- Software-Selectable internal or external clock
- Programmable interrupt generation and priority
- Gated external pulse counter

Type A Timers additionally have:

- Async Timer/Counter with a built in OSC
- Can operate during CPU Sleep mode
- Software selectable Prescalers, 1:1, 1:8, 1:64, and 1:256

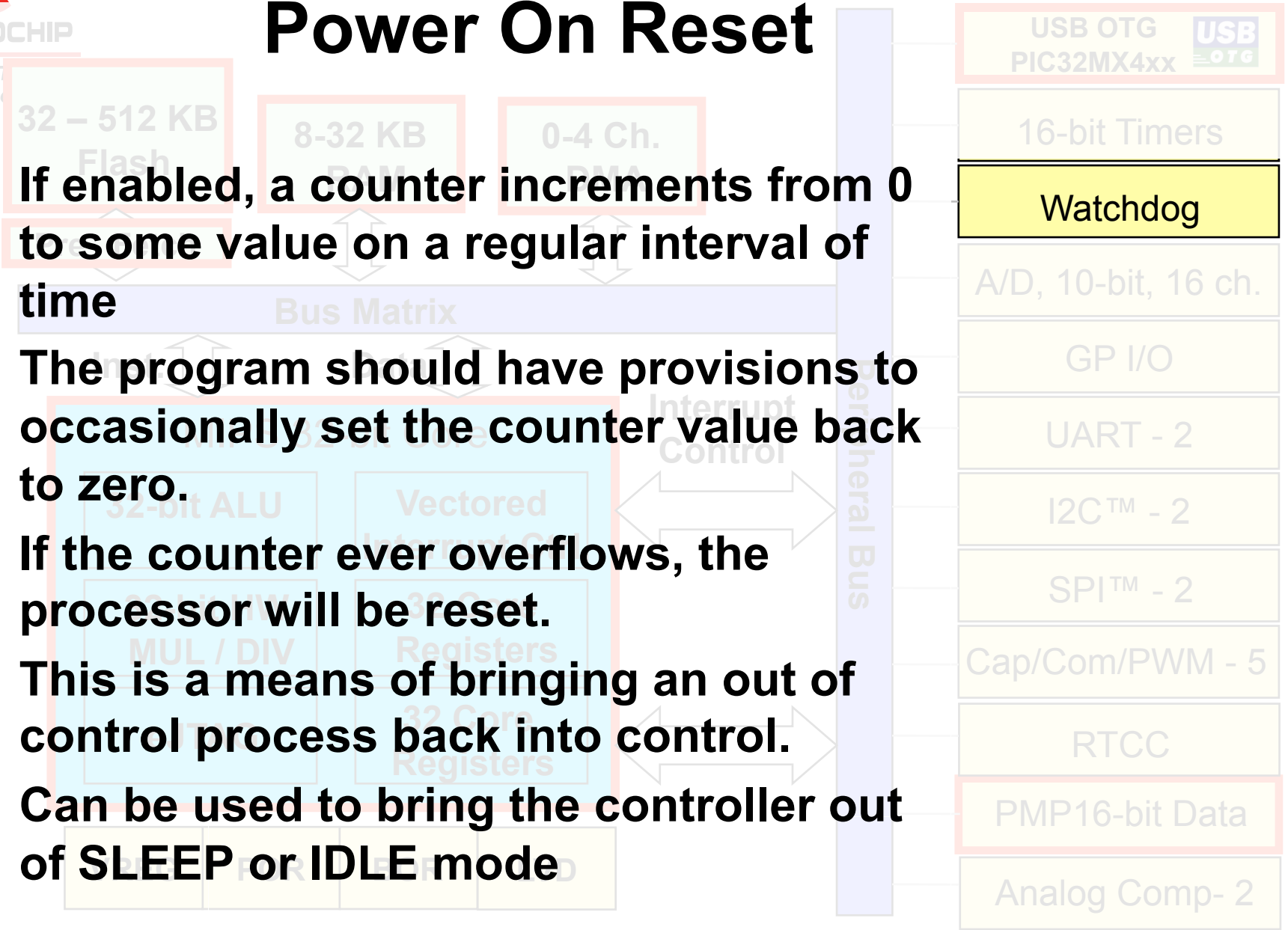
Type B Timers additionally have:

- Ability to concatenate into a 32 bit timer/counter
- Software Prescalers 1:1, 1:2, 1:4, 1:8, 1:16, 1:32, 1:64, 1:256
- Event Trigger Capability



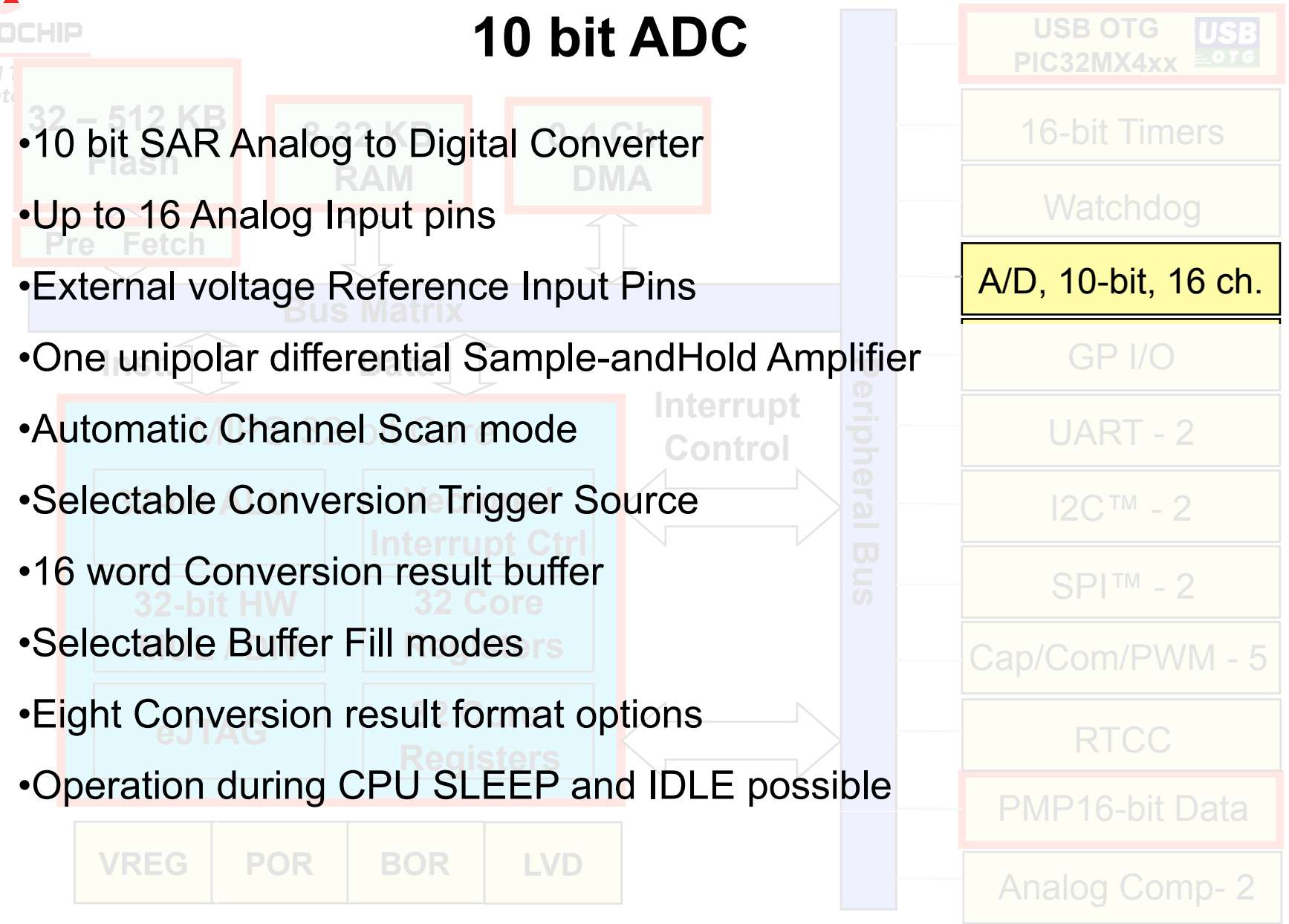
Power On Reset

- If enabled, a counter increments from 0 to some value on a regular interval of time
- The program should have provisions to occasionally set the counter value back to zero.
- If the counter ever overflows, the processor will be reset.
- This is a means of bringing an out of control process back into control.
- Can be used to bring the controller out of SLEEP or IDLE mode



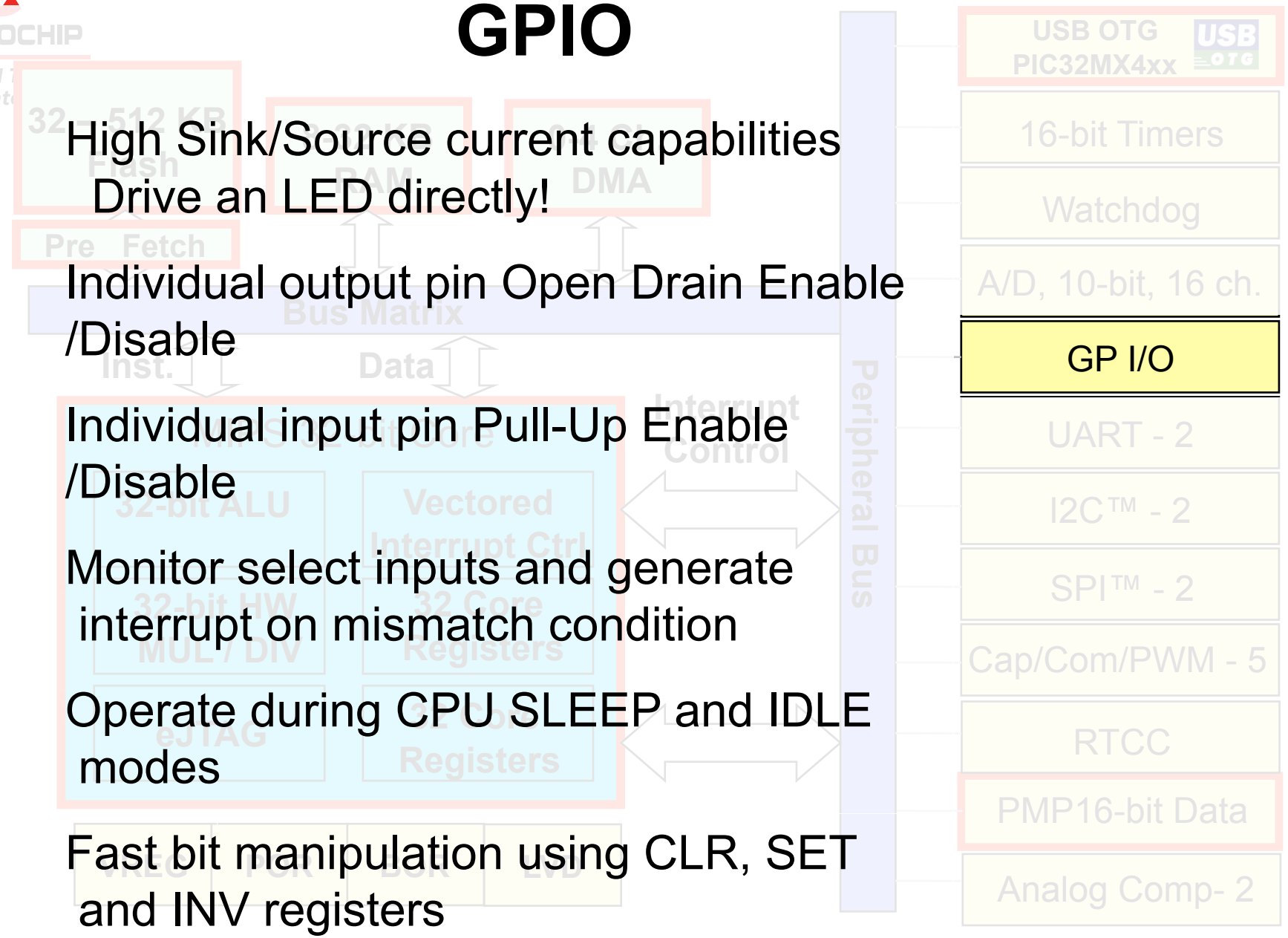
10 bit ADC

- 10 bit SAR Analog to Digital Converter
- Up to 16 Analog Input pins
- External voltage Reference Input Pins
- One unipolar differential Sample-andHold Amplifier
- Automatic Channel Scan mode
- Selectable Conversion Trigger Source
- 16 word Conversion result buffer
- Selectable Buffer Fill modes
- Eight Conversion result format options
- Operation during CPU SLEEP and IDLE possible



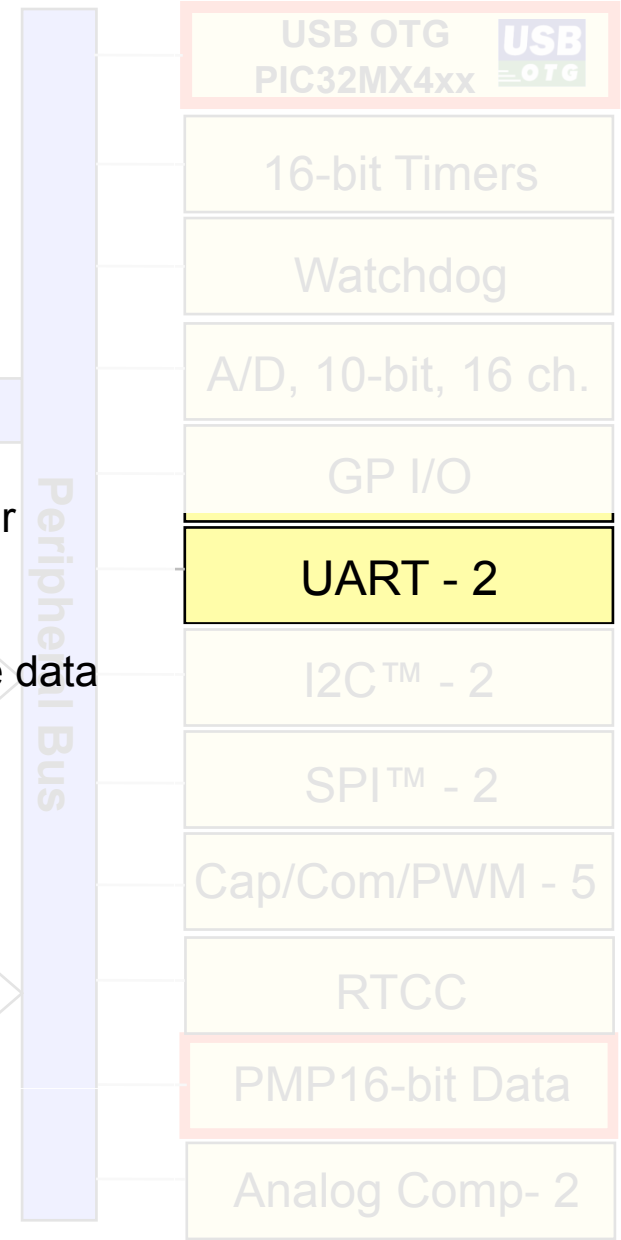
GPIO

- High Sink/Source current capabilities
  - Drive an LED directly!
- Individual output pin Open Drain Enable /Disable
- Individual input pin Pull-Up Enable /Disable
- Monitor select inputs and generate interrupt on mismatch condition
- Operate during CPU SLEEP and IDLE modes
- Fast bit manipulation using CLR, SET and INV registers



UARTS

- Full-duplex, 8-bit or 9-bit data transmission
- Even, odd or no parity options (for 8-bit data)
- One or two Stop bits
- Hardware auto-baud feature
- Hardware flow control option
- Fully integrated Baud Rate Generator with 16-bit prescaler
- Baud rates ranging from 76 bps to 20 Mbps at 80 MHz
- 4-level-deep First-In First-Out (FIFO) transmit and receive data buffer
- Parity, framing and buffer overrun error detection
- Support for interrupt only on address detect (9th bit = 1)
- Separate transmit and receive interrupts
- Loopback mode for diagnostic support
- LIN 1.2 protocol support
- IrDA encoder and decoder with 16x baud clock output for external IrDA encoder/decoder support

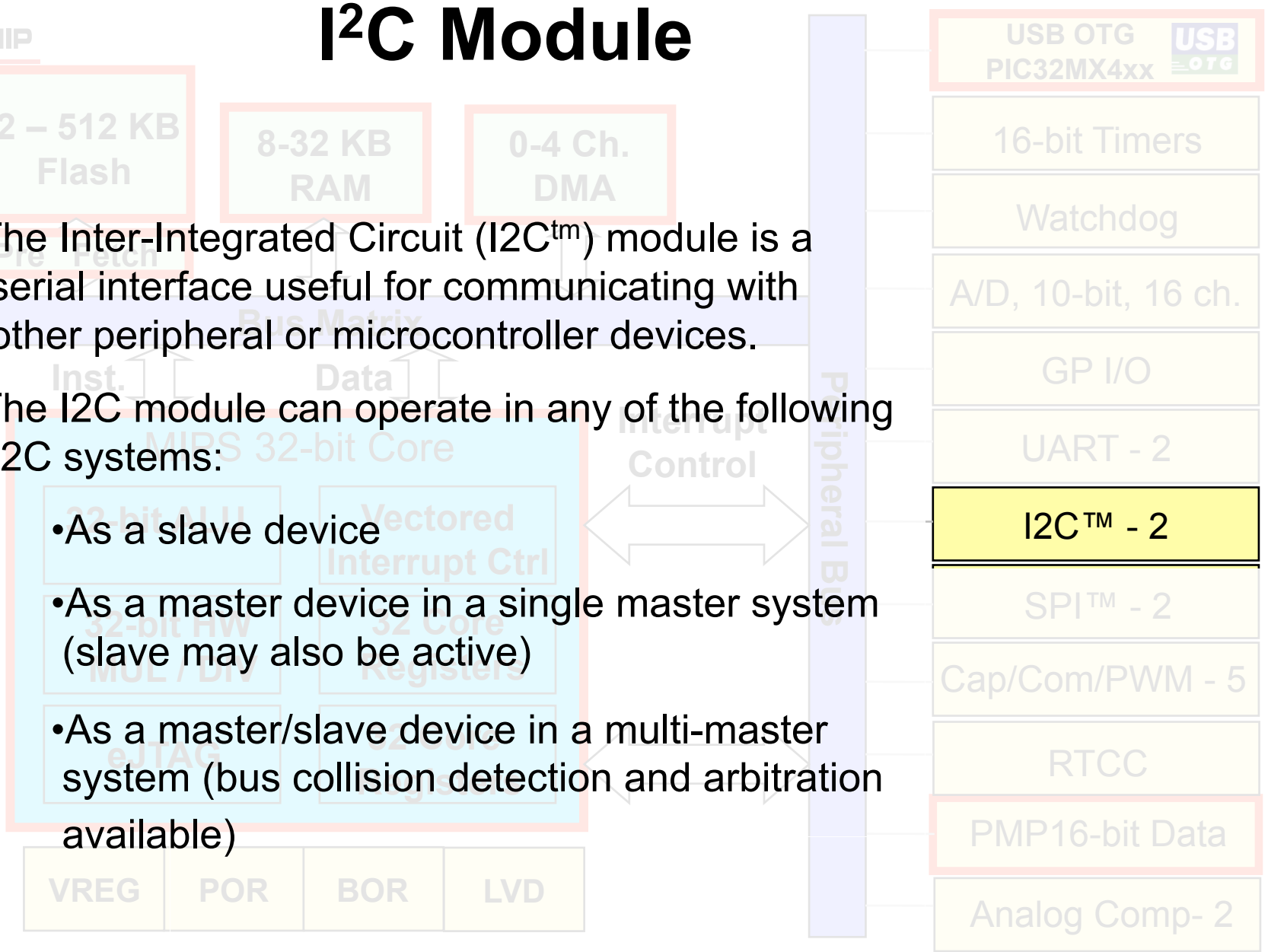


I²C Module

The Inter-Integrated Circuit (I2C™) module is a serial interface useful for communicating with other peripheral or microcontroller devices.

The I2C module can operate in any of the following I2C systems:

- As a slave device
- As a master device in a single master system (slave may also be active)
- As a master/slave device in a multi-master system (bus collision detection and arbitration available)

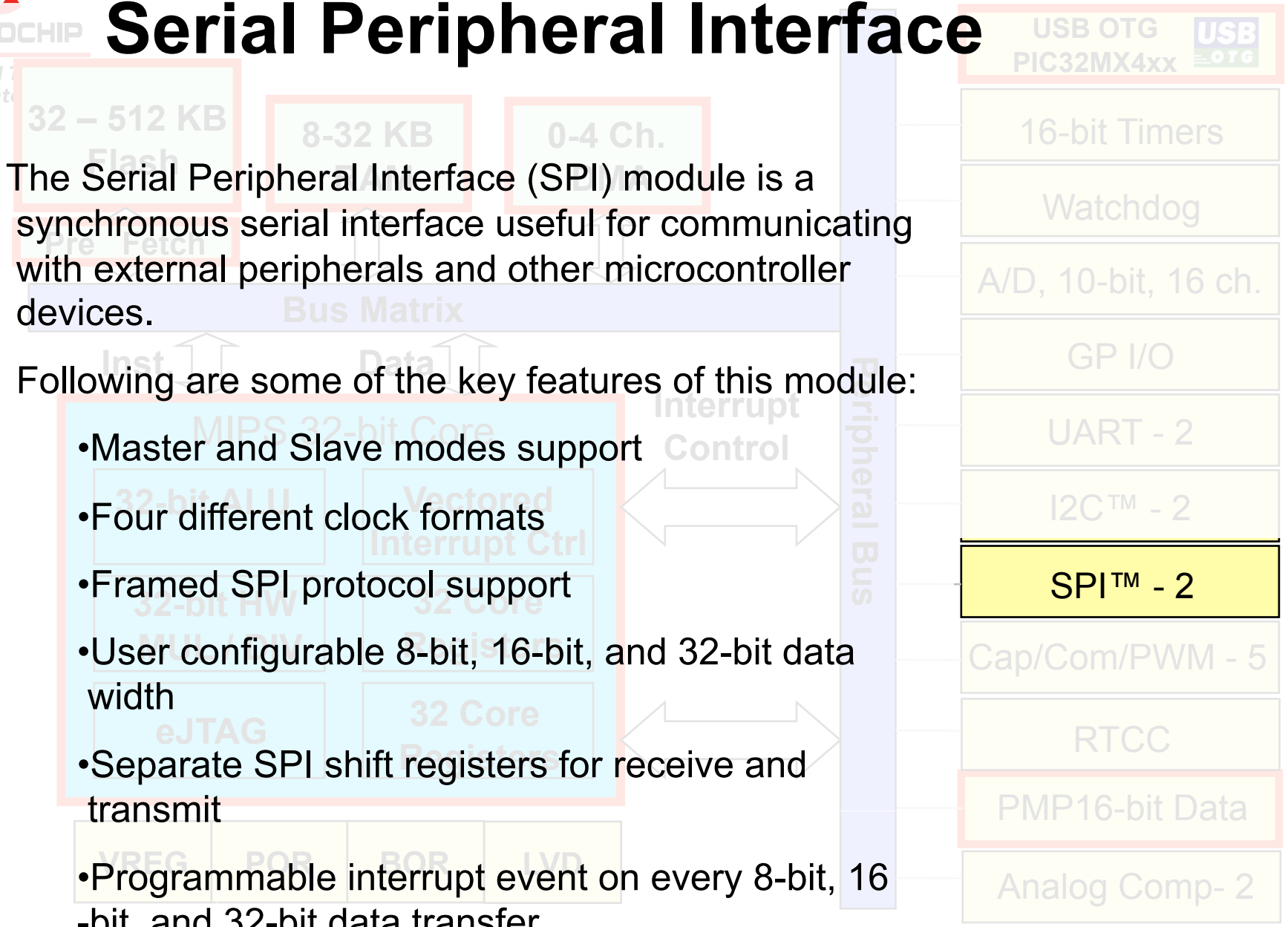


Serial Peripheral Interface

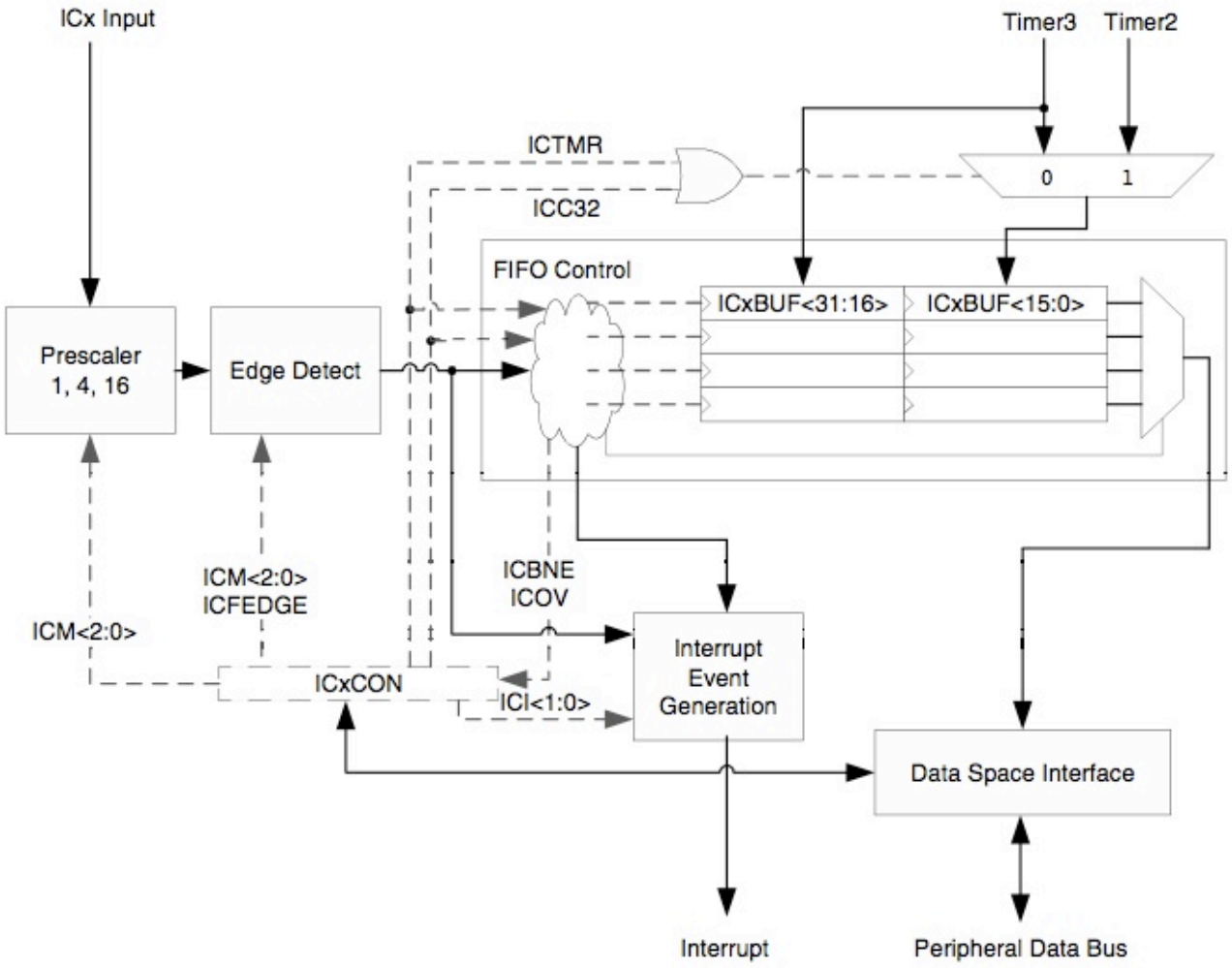
The Serial Peripheral Interface (SPI) module is a synchronous serial interface useful for communicating with external peripherals and other microcontroller devices.

Following are some of the key features of this module:

- Master and Slave modes support
- Four different clock formats
- Framed SPI protocol support
- User configurable 8-bit, 16-bit, and 32-bit data width
- Separate SPI shift registers for receive and transmit
- Programmable interrupt event on every 8-bit, 16 -bit, and 32-bit data transfer



Input Capture



Can capture every rising or falling event, and can count every 4th or 16th event

USB OTG PIC32MX4xx

16-bit Timers

Watchdog

A/D, 10-bit, 16 ch.

CR I/O

UART - 2

I2C™ - 2

SPI™ - 2

Cap/Com/PWM - 5

RTCC

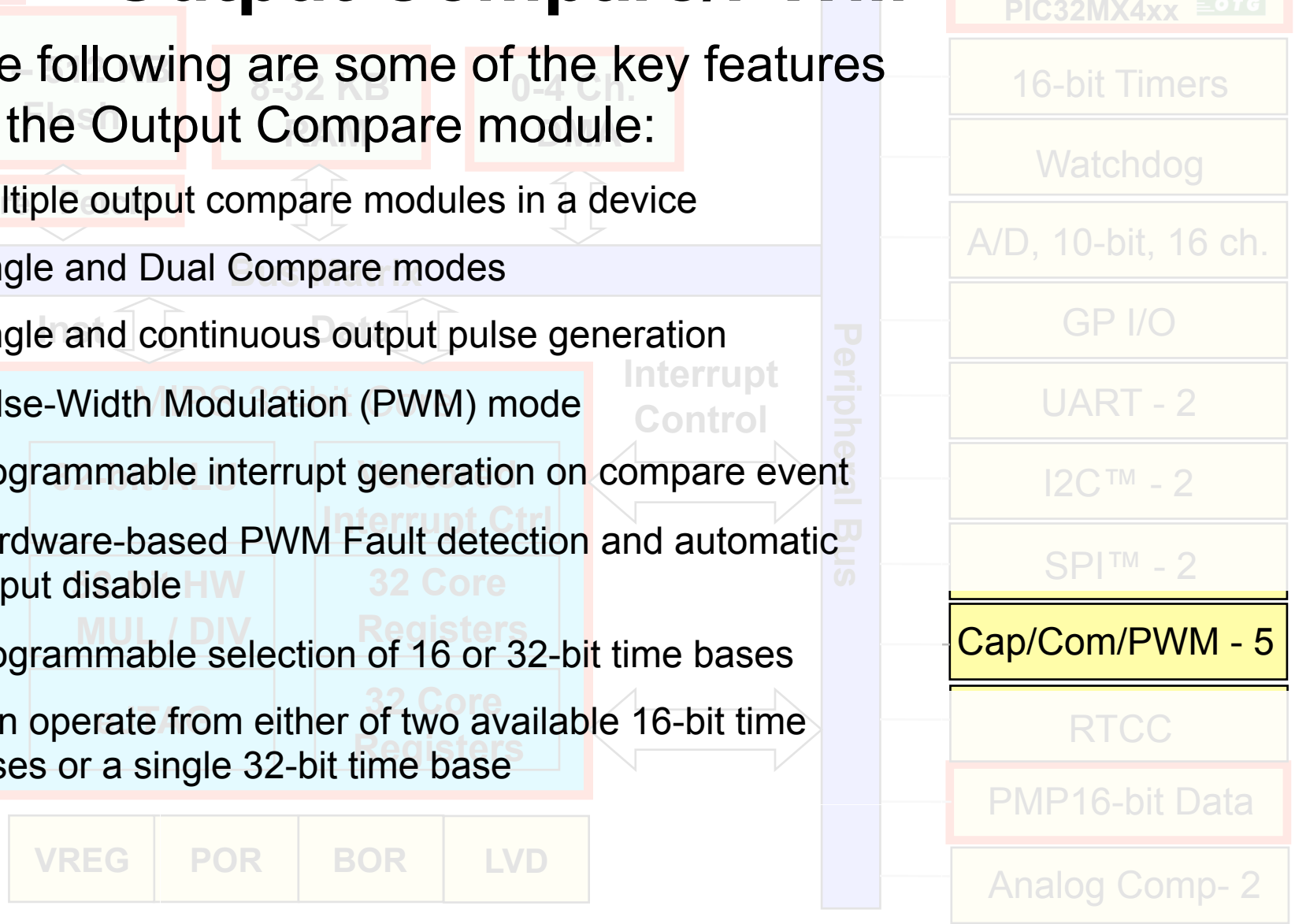
PMP16-bit Data

Analog Comp- 2

Output Compare/PWM

The following are some of the key features of the Output Compare module:

- Multiple output compare modules in a device
- Single and Dual Compare modes
- Single and continuous output pulse generation
- Pulse-Width Modulation (PWM) mode
- Programmable interrupt generation on compare event
- Hardware-based PWM Fault detection and automatic output disable
- Programmable selection of 16 or 32-bit time bases
- Can operate from either of two available 16-bit time bases or a single 32-bit time base



Real Time Clock/Calander

Intended for applications where time must be maintained with minimum intervention from the CPU.

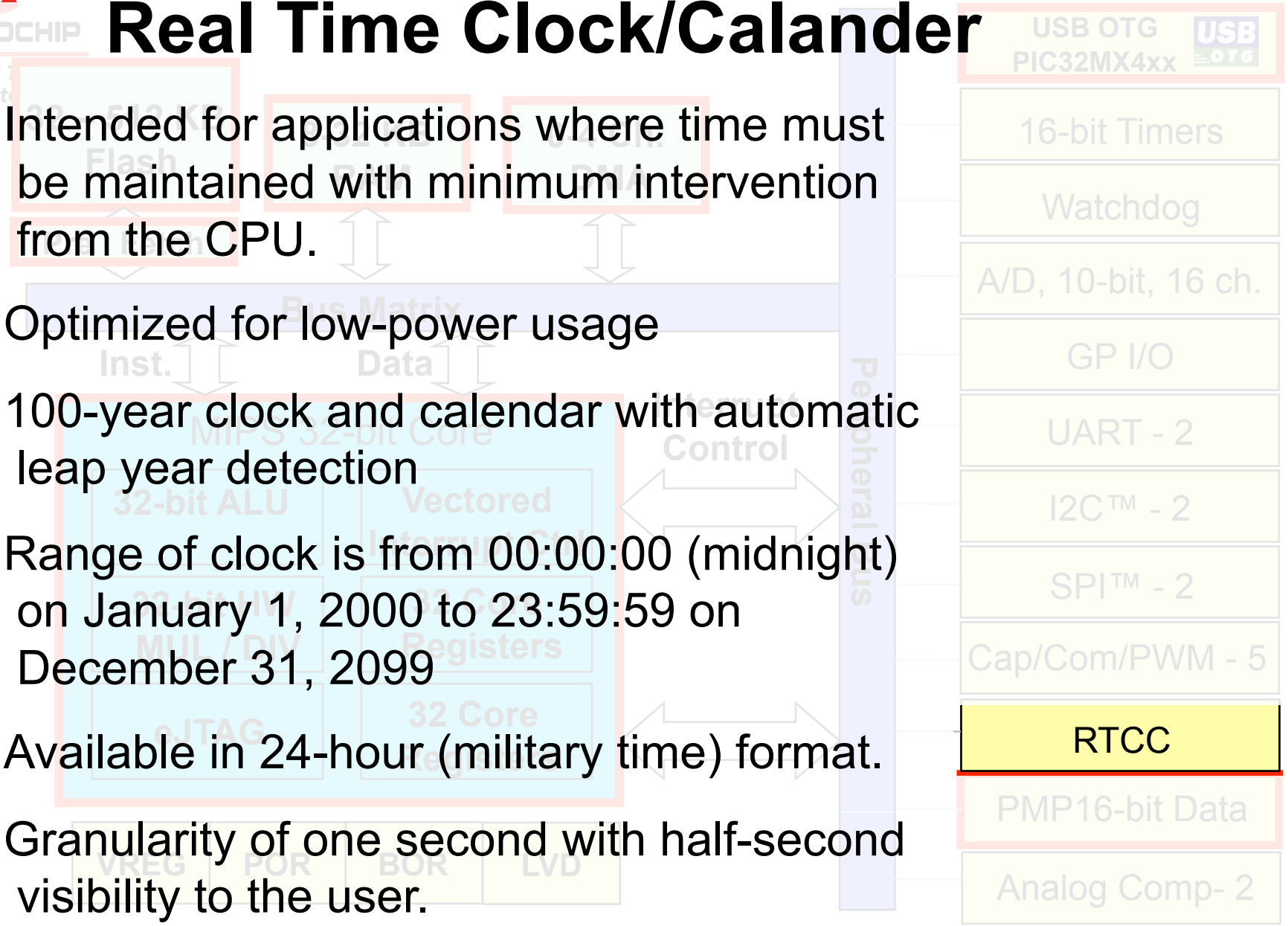
Optimized for low-power usage

100-year clock and calendar with automatic leap year detection

Range of clock is from 00:00:00 (midnight) on January 1, 2000 to 23:59:59 on December 31, 2099

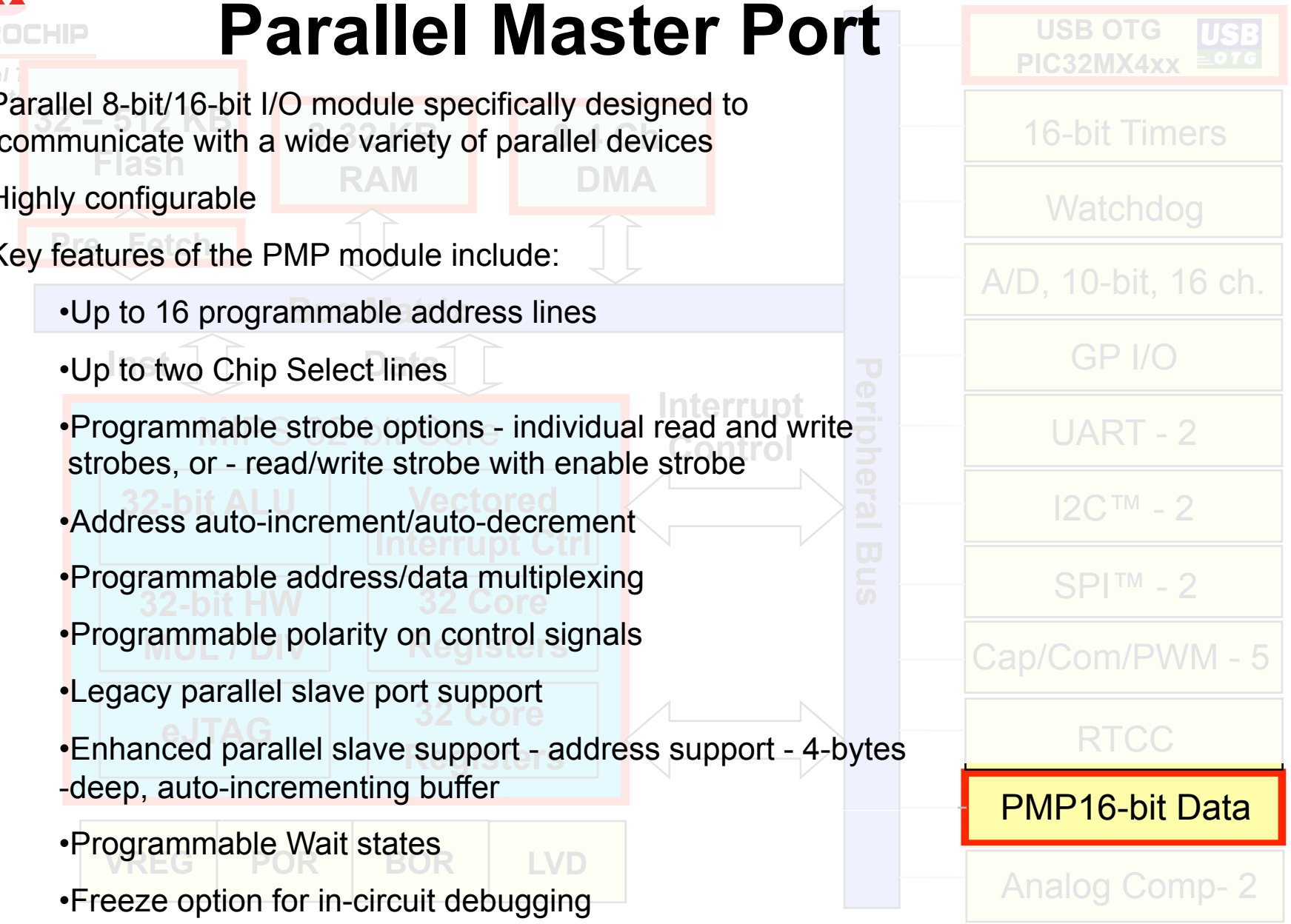
Available in 24-hour (military time) format.

Granularity of one second with half-second visibility to the user.



Parallel Master Port

- Parallel 8-bit/16-bit I/O module specifically designed to communicate with a wide variety of parallel devices
- Highly configurable
- Key features of the PMP module include:
  - •Up to 16 programmable address lines
  - •Up to two Chip Select lines
  - •Programmable strobe options - individual read and write strobes, or - read/write strobe with enable strobe
  - •Address auto-increment/auto-decrement
  - •Programmable address/data multiplexing
  - •Programmable polarity on control signals
  - •Legacy parallel slave port support
  - •Enhanced parallel slave support - address support - 4-bytes -deep, auto-incrementing buffer
  - •Programmable Wait states
  - •Freeze option for in-circuit debugging



Analog Comparator

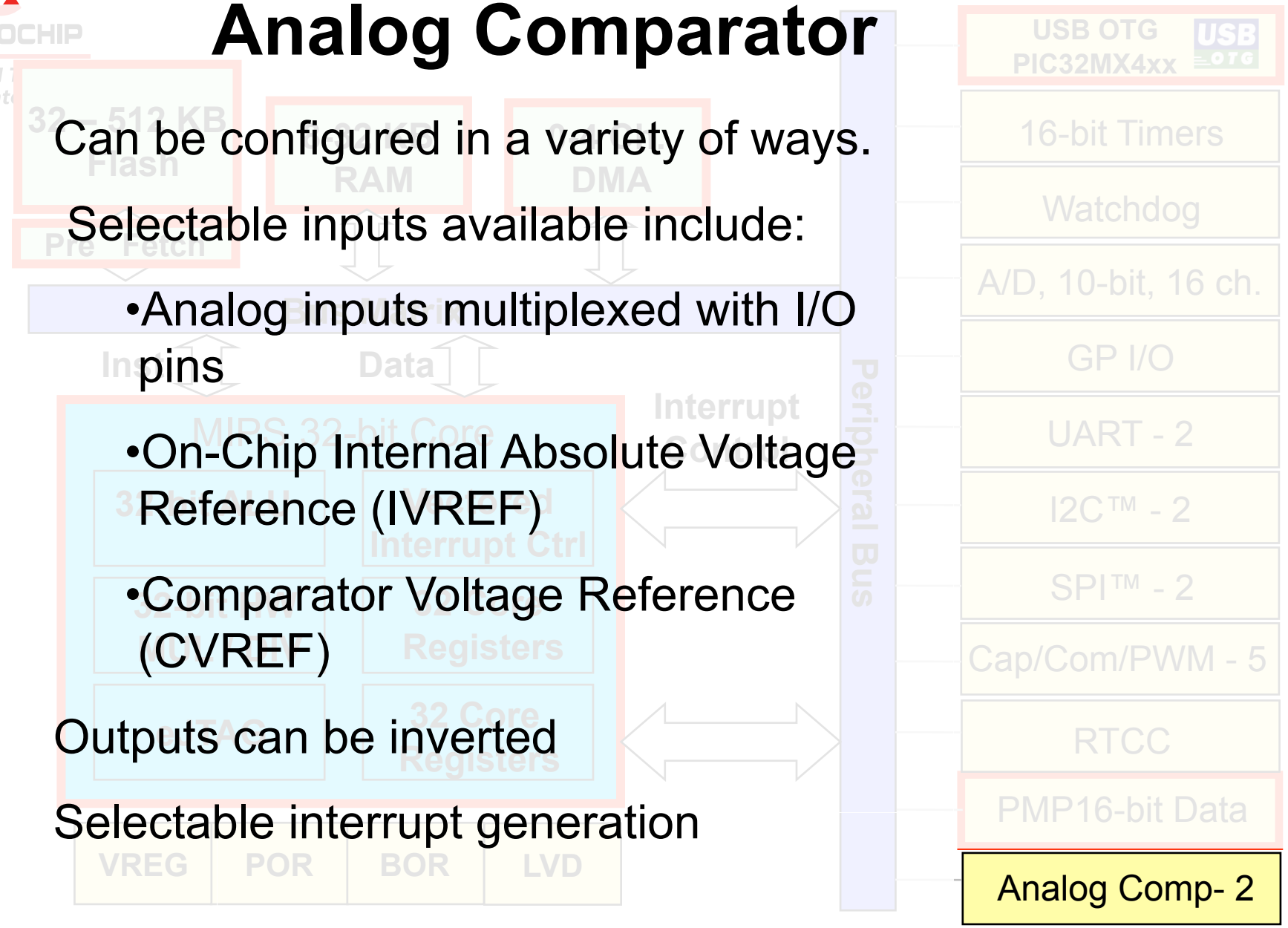
Can be configured in a variety of ways.

Selectable inputs available include:

- Analog inputs multiplexed with I/O pins
- On-Chip Internal Absolute Voltage Reference (IVREF)
- Comparator Voltage Reference (CVREF)

Outputs can be inverted

Selectable interrupt generation

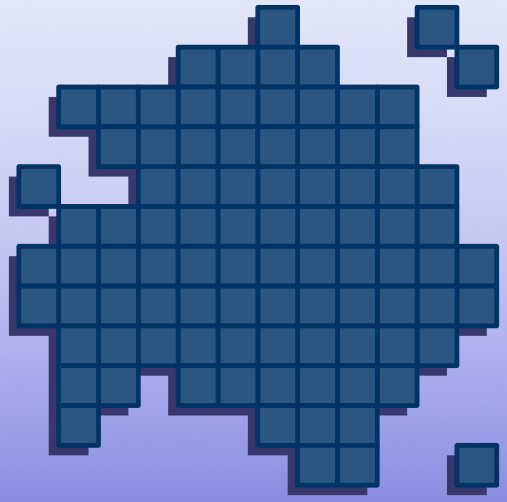


MICROCHIP
Regional Training Centers

Peripherals

wrap up

- Whirlwind tour through all the peripherals available in the PIC32 family
- Lots more information available in the datasheet
- Consider taking MCU 4201 for easy hands on experience with these peripherals



Thank you!