MICROCHIP

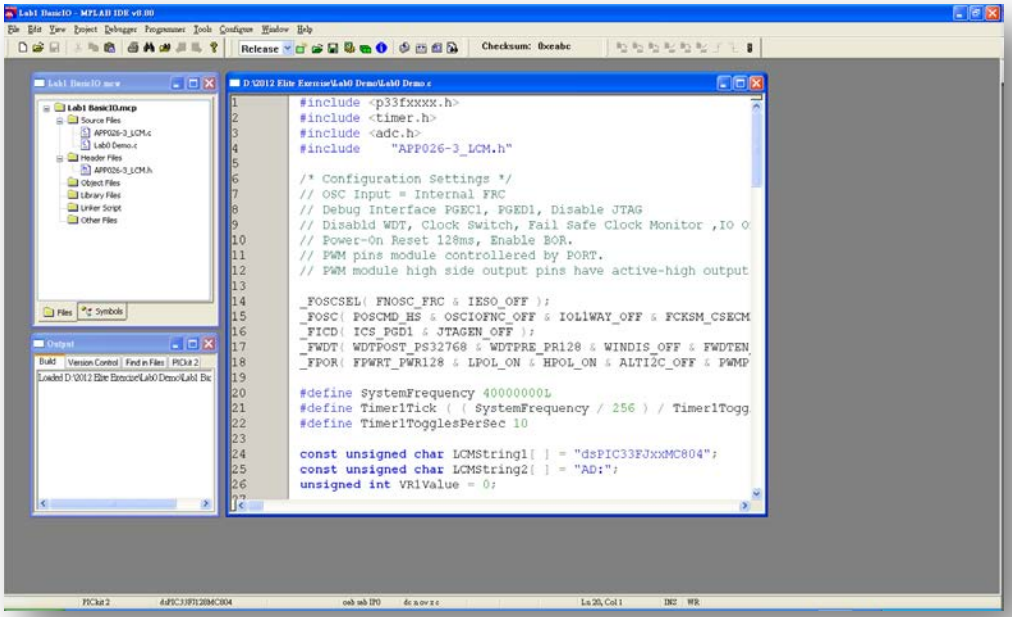
Regional Training Centers

Section 2
IDE, MPLAB XC32 and Development Tools Introduction

MICROCHIP
Regional Training Centers

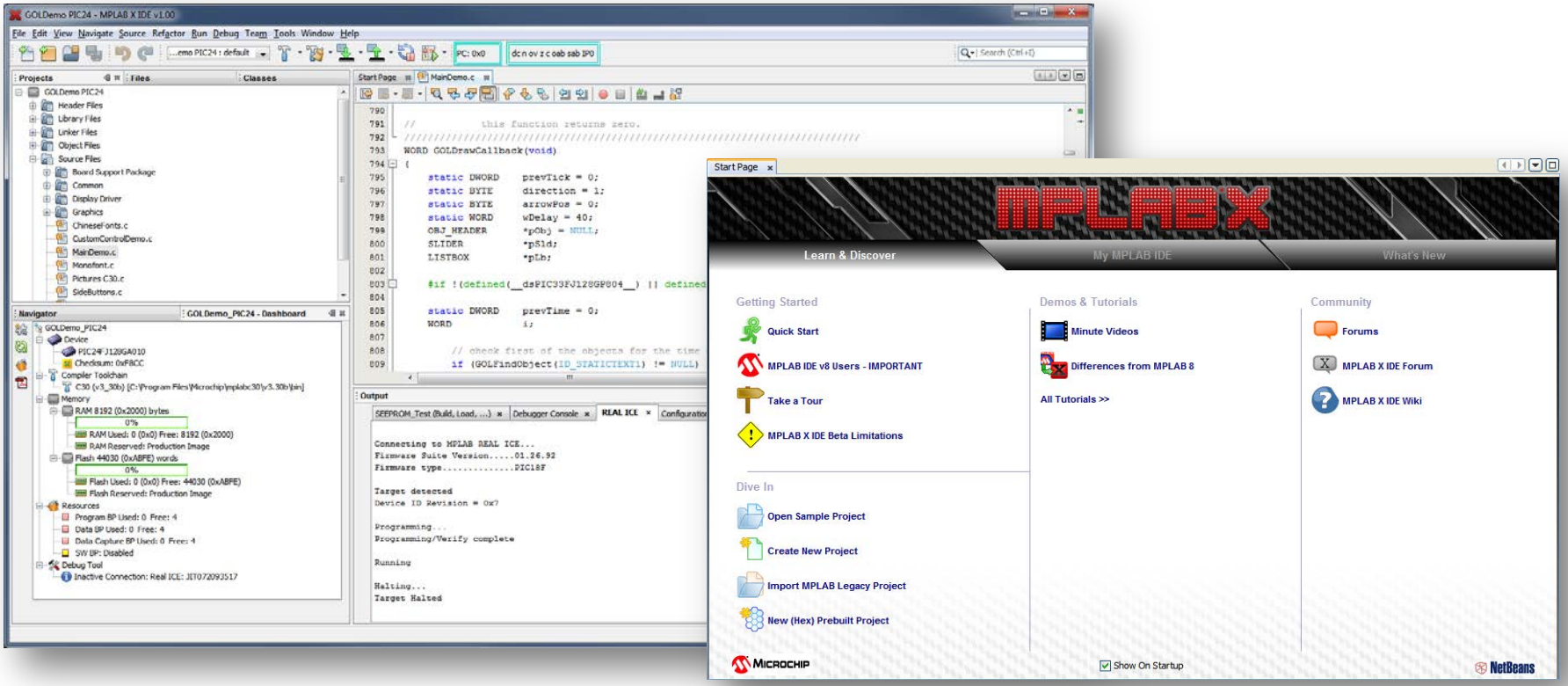
MPLAB IDE

- Microchip提供的整合式開發環境, 支援全系列8-Bits, 16-Bits及32-Bits的MCU。所有的MCU都可透過相同的環境開發。目前最新的版本是v8.92。
- 可整合各式的組譯器/編譯器(MPLAB C, Hi-TECH C, etc.),開發工具(PICkit3, ICD3, Real ICE, etc..) 。



MPLAB X IDE

- 新一代的整合式開發環境, 可支援全系列MCU, 擁有許便利的功能與特徵。Java Base, 可以跨平台, 目前最新的版本是v1.20。
- 不支援ICD2, ICE2000/4000, ProMATE II, PICStart Plus.



MICROCHIP
Regional Training Centers

Debug/Program Tools

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

USD $69.0

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

USD $200.0

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 $500.0

MICROCHIP
Regional Training Centers

MPLAB XC32

- 新一代的C Compiler, 支援Microchip 32-Bits MCU。架構於GCC Compiler, 採用GNU General Public License (GPL)。
  * GPL:使用者可以自由執行/複製/修改/改進再公開
- 支援PIC32全系列MCU, MPLAB XC32可整合於 MPLAB IDE / MPLAB X IDE中。
- 提供標準C Functions(printf, scanf, etc..),週邊函式庫,數學運算函數等。
- 可至Microchip網站, 取得免費版本。
  http://www.microchip.com/xc32

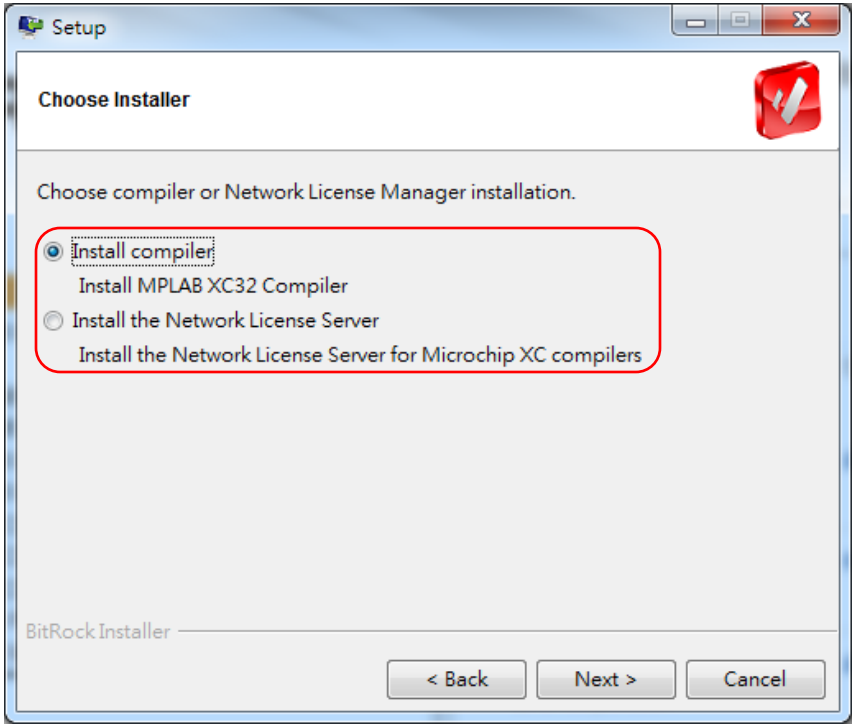
MICROCHIP
Regional Training Centers

About MPLAB XC32 Version

- MPLAB XC32提供四種不同版本的C Compiler:
  - 標準版(Standard) (SW006023-1 Standard Compiler Workstation License, 495 USD)
    付費版本,擁有完整編譯功能及最佳化(Optimizations)的功能,可跟精簡版相比,最高可以減少20%~25%的記憶體空間。
  - 專業版(Pro) (SW006023-2 Pro Compiler Workstation License, 995 USD)
    付費版本,擁有完整編譯功能及最佳化(Optimizations)的功能,可跟精簡版相比,最高可以減少50%的記憶體空間。
  - 精簡版(Lite)
    免費版本,擁有完整編譯功能, 與正式版唯一的差異,就是沒有完整的最佳化功能,只擁有Level 1最佳化功能。
  - 評估版(Evaluation)
    免費版本, 安裝後60天內,具有標準版的所有功能,60天後自動變成精簡版。
- 需要付費的標準版(Standard) 跟專業版(Pro), 其授權區分成Workstation License跟Network Server License兩種。

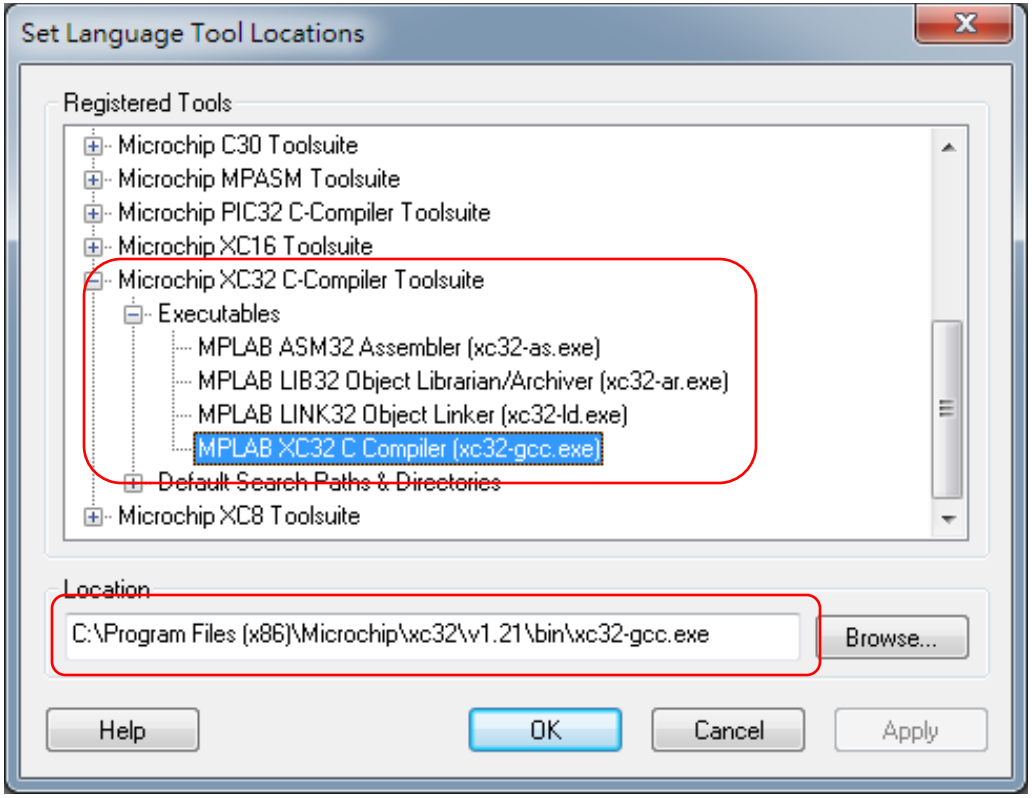
MPLAB XC32 Install

- MPLAB IDE安裝時, 並不會安裝MPLAB XC32。必須自行安裝。
- MPLAB XC32安裝時, 預設都是精簡版。如果要使用其他版本, 必須透過後續的線上註冊來改變(專業版, 標準版必須付費, 評估版必須線上申請)。
- 本次課程會使用v1.30的精簡版, 預設安裝路徑為
  C:\Program Files\Microchip\XC32\v1.30\



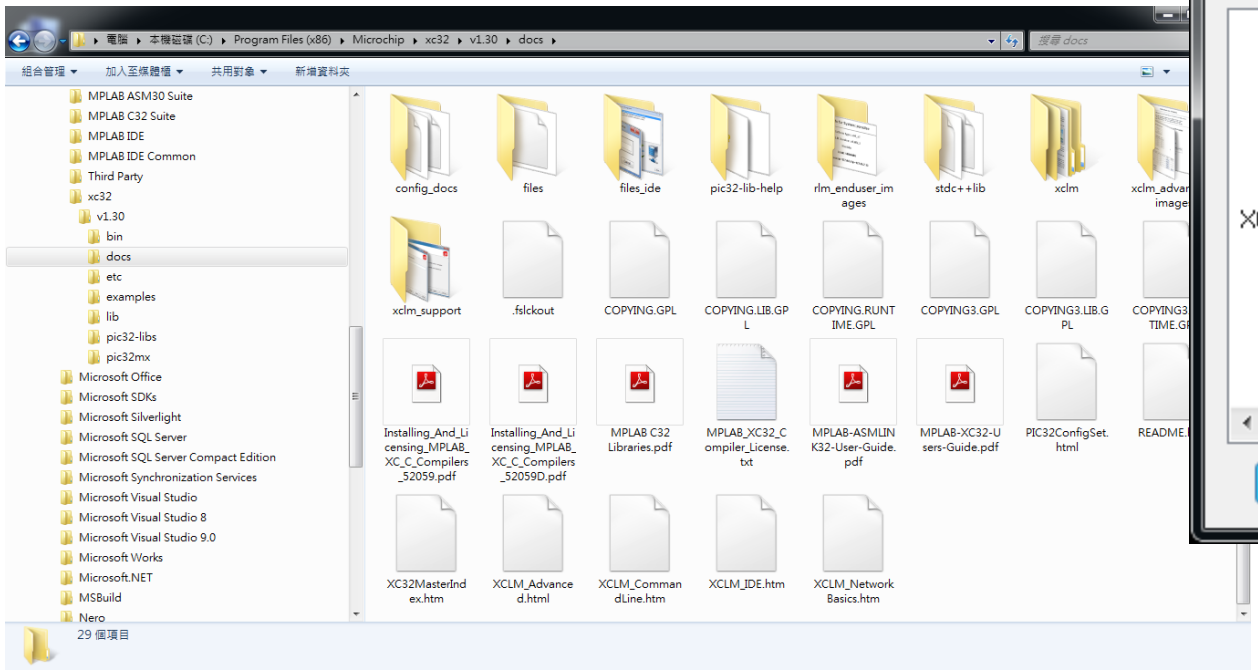
MPLAB XC32 Install

- 安裝成功後可從MPLAB IDE中的功能表選擇
  Project\Set Language Tool Location
  檢查Microchip XC32 C-Compiler Toolsuite的安裝狀態與路徑。



MPLAB XC32's Documents

- MPLAB XC32的相關文件,可以在MPLAB IDE中的功能表選擇Help\Topics,再由Topics的列表中尋找MPLAB XC32的相關文件。
- 也可以在C:\Program Files\Microchip\XC32\vx.xx\docs\裡找到許多相關的文件。



MICROCHIP
Regional Training Centers

MPLAB XC32's Data Type

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

MICROCHIP
Regional Training Centers

Access SFRs

- 每個特殊功能暫存器(Special Function Register)都有特定的名稱與位址。MPLAB XC32為了使用SFRs, 必需使用MCU的標頭檔(Header File)來定義其名稱與結構 ,並利用Processor.o(Object File)來定義其位址, 才可以正確存取SFRs。
- 要使用到SFRs時,必須先含入(include)對應MCU的標頭檔(Header File)。例如:
  #include <p32mx795f512l.h>
  #include <p32mx450f256h.h>
- 標頭檔(Header File)的檔案命名方式並非所有使用者都很熟悉, 因此建議可直接含入(include)通用標頭檔(Generic Header File), 由MPLAB XC32自己來尋找正確的標頭檔(Header File)。
  #include <p32xxxx.h>

MICROCHIP
Regional Training Centers

Access SFRs

- 透過MCU標頭檔(Header File)與Processor.o(Object File)的定義, 可以正確的存取SFRs。例如:

  TRISA= 0xFFFE;

  LATA = 0x0001;

- MCU標頭檔(Header File)中, 也定義了各個SFRs的結構, 因此也可以透過結構, 存取SFRs內的特定成員(*SFRNAME*bits.*BITNAME*)。 如:

  TRISAbits.TRISA0 = 0;

  LATAbits.LATA0 = 1;

- TRISA及TRISAbits都參考到同一SFRs的位址(透過Processor.o的定義)。因此操作TRISA或TRISAbits結果是相同的。
- 每個SFRs的結構型態定義都不同,建議直接打開標頭檔(Header File)來觀察看看。

MCU's Header File

- p32mx450f256h.h MCU標頭檔(Head File)的內容片段

  C:\Program Files (x86)\Microchip\xc32\v1.30\pic32mx\include\proc\

```
extern volatile unsigned int      LATA __attribute__((section("sfrs")));
typedef union
{
   struct
   {
      unsigned LATA0:1;
      unsigned LATA1:1;
      …
      unsigned LATA7:1;
      unsigned :1;
      unsigned LATA9:1;
      unsigned LATA10:1;
      unsigned :3;
      unsigned LATA14:1;
      unsigned LATA15:1;
   };
   struct
   {
      unsigned w:32;
   };
} __LATAbits_t;
extern volatile __LATAbits_t LATAbits __asm__ ("LATA") __attribute__((section("sfrs")));
```

Virtual Address (BF88_#)	Register Name(1)	Bit Range	Bits															
			31/15	30/14	29/13	28/12	27/11	26/10	25/9	24/8	23/7	22/6	21/5	20/4	19/3	18/2	17/1	16/0
6010	TRISA	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
		15:0	TRISA15	TRISA14	—	—	—	TRISA10	TRISA9	—	TRISA7	TRISA6	TRISA5	TRISA4	TRISA3	TRISA2	TRISA1	TRISA0
6020	PORTA	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
		15:0	RA15	RA14	—	—	—	RA10	RA9	—	RA7	RA6	RA5	RA4	RA3	RA2	RA1	RA0
6030	LATA	31:16	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
		15:0	LATA15	LATA14	—	—	—	LATA10	LATA9	—	LATA7	LATA6	LATA5	LATA4	LATA3	LATA2	LATA1	LATA0

MICROCHIP
Regional Training Centers

MCU's Linker Script File

- 連結檔(Linker Script File, *.ld)定義定義了資料(SRAM)與程式記憶體(User Flash, Boot Flash)等記憶體區塊的起始位置與長度。
- 利用修改procdefs.ld, 可以自行規劃記憶體的使用範圍。
- procdefs.ld的內容片段(for PIC32MX450F256H)
  C:\Program Files (x86)\Microchip\xc32\v1.30\pic32mx\lib\proc\32MX450F256H\

```
MEMORY
{
  kseg0_program_mem    (rx)      : ORIGIN = 0x9D000000, LENGTH = 0x40000
  kseg0_boot_mem                 : ORIGIN = 0x9FC00490, LENGTH = 0x970
  exception_mem                  : ORIGIN = 0x9FC01000, LENGTH = 0x1000
  kseg1_boot_mem                 : ORIGIN = 0xBFC00000, LENGTH = 0x490
  debug_exec_mem                 : ORIGIN = 0xBFC02000, LENGTH = 0xFF0
  config3                        : ORIGIN = 0xBFC02FF0, LENGTH = 0x4
  config2                        : ORIGIN = 0xBFC02FF4, LENGTH = 0x4
  config1                        : ORIGIN = 0xBFC02FF8, LENGTH = 0x4
  config0                        : ORIGIN = 0xBFC02FFC, LENGTH = 0x4
  kseg1_data_mem       (w!x)     : ORIGIN = 0xA0000000, LENGTH = 0x10000
  sfrs                           : ORIGIN = 0xBF800000, LENGTH = 0x100000
  configsfrs                     : ORIGIN = 0xBFC02FF0, LENGTH = 0x10
}
```