

WT32L064 Application Note
EVB 與簡易程式說明

(中文版)

Rev. 1.0

September 2020

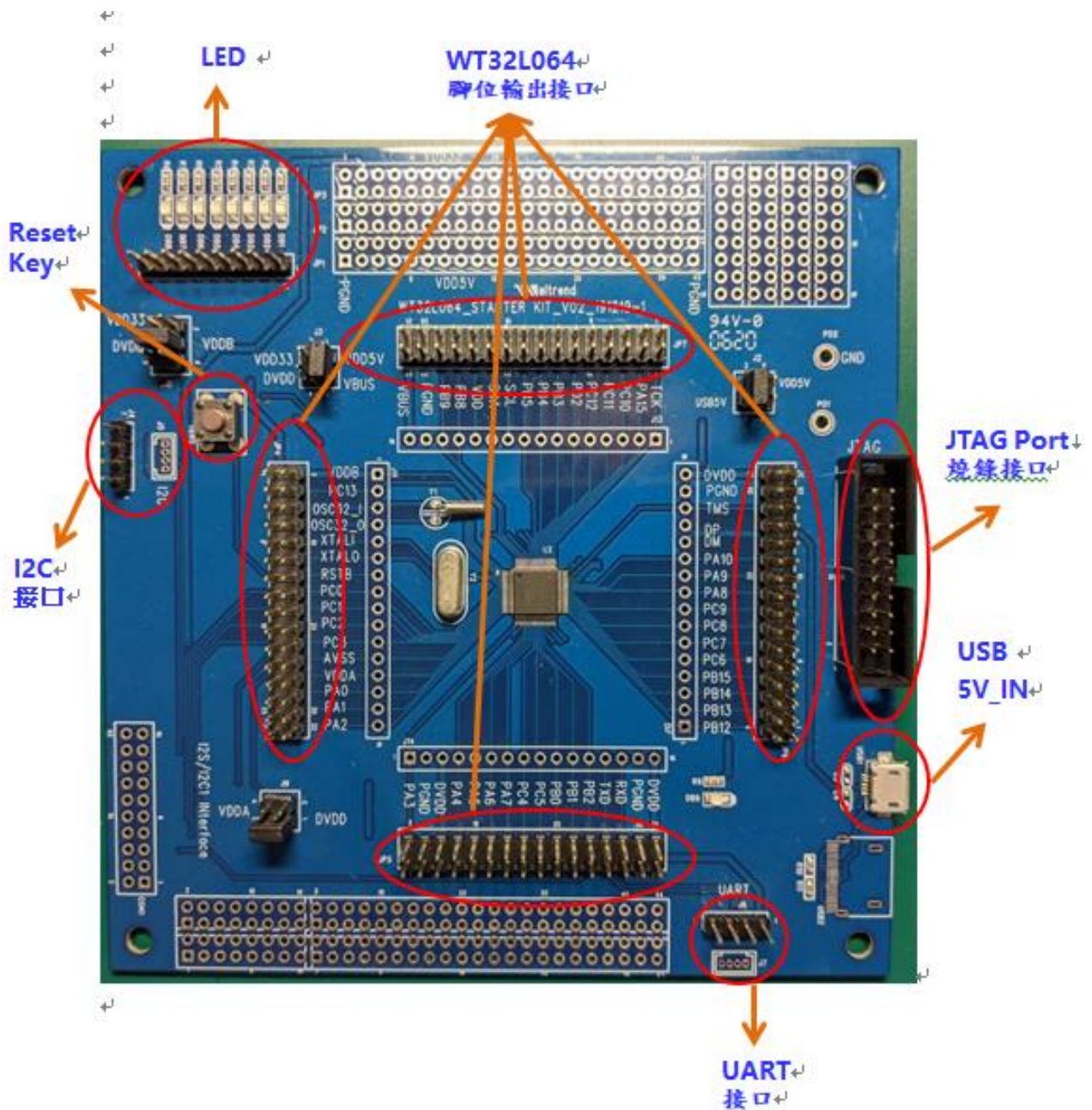
目 錄

1. EVB 功能說明	3
1.1 外觀說明 :	3
1.2 EVB 接口說明 :	4
1.2.1 Micro USB Port (USB1)	4
1.2.2 I2C 介面接口 (J4/J5)	4
1.2.3 UART 介面接口 (J6/J7)	4
1.2.4 JTAG 介面接口 (J6/J7)	5
1.2.5 WT32L064 腳位輸出接口 (JP4/JP5/FP6/JP7)	5
1.3 EVB 線路圖 :	7
1.3.1 Power	7
1.3.2 WT32L064	8
2. ARM-MDK 安裝與環境設定	9
3. CMSIS 中間層驅動說明	12
3.1 定義:	12
3.2 CMSIS 內容說明:	12
4. PACK 範例程式架構說明	13
4.1 EXAMPLES 資料夾內功能說明	13
5. 實例程式操作說明	15
5.1 範例流程圖	17
5.2 主程式流程	17
5.3 周邊功能的初始化	20
5.3.1 工作頻率選擇	20
5.3.2 周邊函式功能說明	21
6. 版本更改紀錄:	24

1. EVB 功能說明

WT32L064 是一款超低功耗 ARM 32 位 Cortex M0 處理器，具有 64KB 嵌入式閃存和 8KB SRAM，而此 Starter Kit Board 則是使用 64-pin LQFP 包裝作為設計並將其功能演示。

1.1 外觀說明：



1.2 EVB 接口說明：

1.2.1 Micro USB Port (USB1)

此為 EVB 直流電壓(5V)輸入接口

腳位編號	說明
1	VBUS
2	D-
3	D+
4	GND
5	GND

1.2.2 I2C 介面接口 (J4/J5)

此為 SLAVE I2C 介面接口

腳位編號	說明
1	DVDD (3.3V)
2	SCL
3	SDA
4	GND

1.2.3 UART 介面接口 (J6/J7)

此為 UART 串列傳輸介面接口

腳位編號	說明
1	DVDD (3.3V)
2	RXD
3	TXD
4	GND

1.2.4 JTAG 介面接口 (J6/J7)

此為 JTAG 傳輸介面接口
做為燒錄程式及偵錯檢查使用

腳位編號	說明	腳位編號	說明
1	VDD33	2	VDD33
3	NC	4	GND
5	NC	6	GND
7	TMS	8	GND
9	TCK	10	GND
11	NC	12	GND
13	NC	14	GND
15	RSTB	16	GND
17	NC	18	GND
19	NC	20	GND

1.2.5 WT32L064 腳位輸出接口 (JP4/JP5/FP6/JP7)

此為 WT32L064 腳位輸出接口，提供外接測試使用

JP4			
腳位編號	說明	腳位編號	說明
1-2	Vddb	3-4	GPIOC13
5-6	GPIOC14/LXTALI	7-8	GPIOC15/LXTALO
9-10	GPIOD0/HXTALI	11-12	GPIOD1/HXTALO
13-14	NRST	15-16	GPIOC0
17-18	GPIOC1	19-20	GPIOC2
21-22	GPIOC3	23-24	AVSS
25-26	VDDA	27-28	GPIOA0
29-30	GPIOA1	31-32	GPIOA2

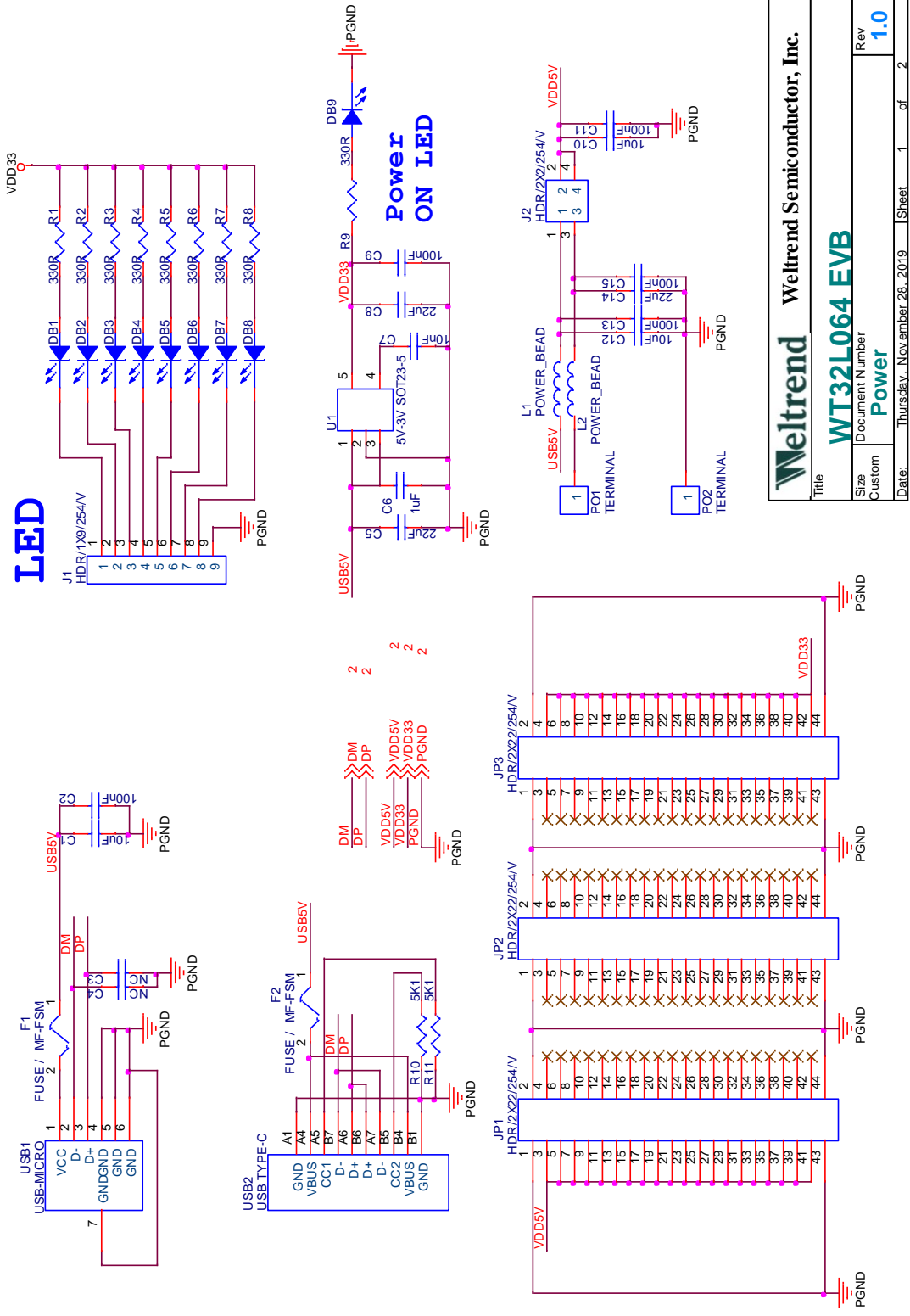
JP5			
腳位編號	說明	腳位編號	說明
1-2	GPIOA2	3-4	VSS
5-6	DVDD	7-8	GPIOA4
9-10	GPIOA5	11-12	GPIOA6
13-14	GPIOA7	15-16	GPIOC4
17-18	GPIOC5	19-20	GPIOB0
21-22	GPIOB1	23-24	GPIOB2
25-26	GPIOB10/TXD	27-28	GPIOB11/RXD
29-30	VSS	31-32	DVDD

JP6			
腳位編號	說明	腳位編號	說明
1-2	GPIOB12	3-4	GPIOB13
5-6	GPIOB14	7-8	GPIOB15
9-10	GPIOC6	11-12	GPIOC7
13-14	GPIOC8	15-16	GPIOC9
17-18	GPIOA8	19-20	GPIOA9
21-22	GPIOA10	23-24	GPIOA11/DM
25-26	GPIOA12/DP	27-28	GPIOA13/TMS
29-30	VSS	31-32	DVDD

JP7			
腳位編號	說明	腳位編號	說明
1-2	GPIOA14/SWCLK	3-4	GPIOA15
5-6	GPIOC10	7-8	GPIOC11
9-10	GPIOC12	11-12	GPIOD2
13-14	GPIOB3	15-16	GPIOB4
17-18	GPIOB5	19-20	GPIOB6
21-22	GPIOB7	23-24	VDD
25-26	GPIOB8	27-28	GPIOB9
29-30	VSS	31-32	VDD5

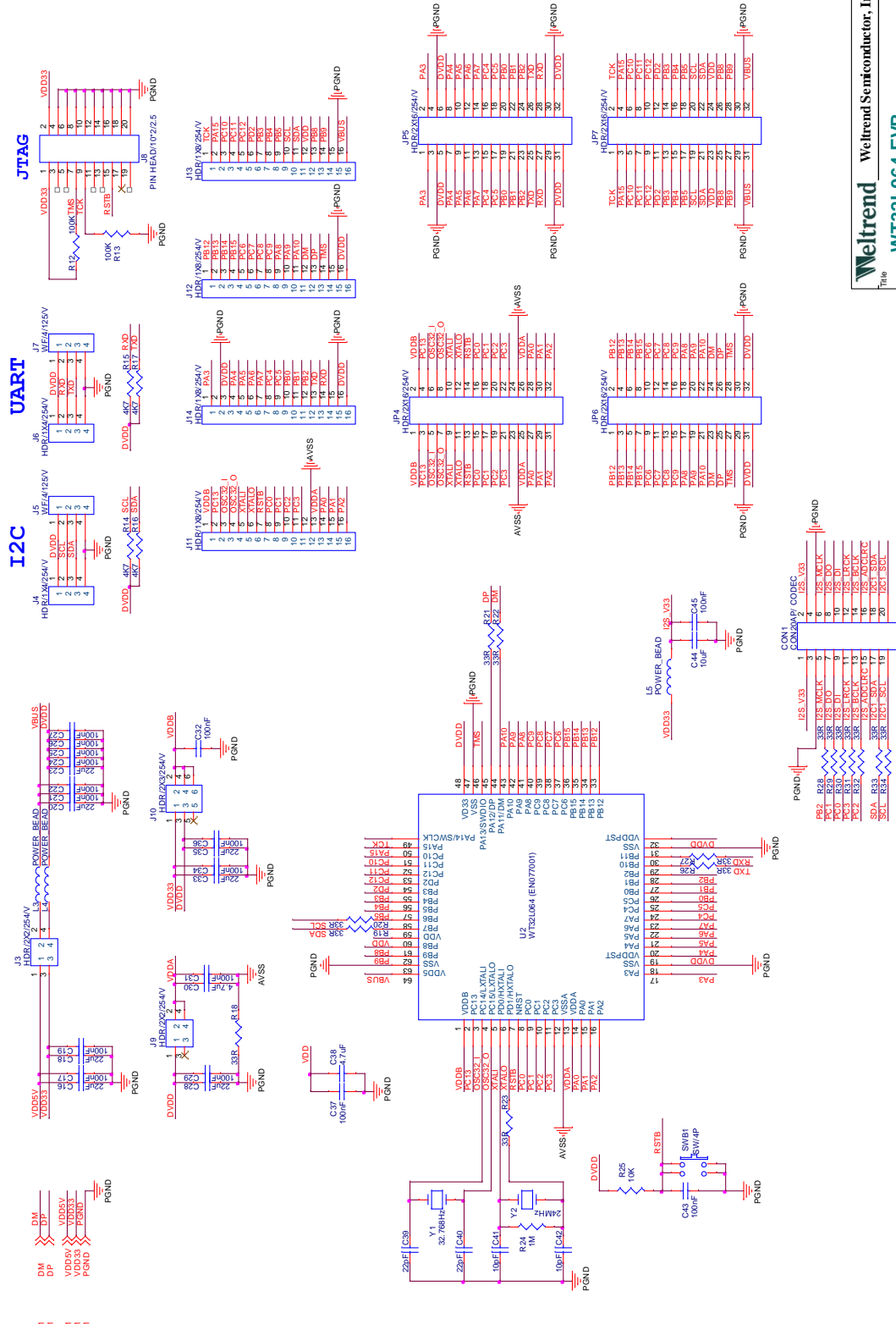
1.3 EVB 線路圖：

1.3.1 Power



Title		Weitrend Semiconductor, Inc.	
Size	Document Number	Rev	
Custom	WT32L064 EVB	Power	1.0
Date:	Thursday, November 28, 2019	Sheet	1 of 2

1.3.2 WT32L064



The **Weltrend** Wellrend Semiconductor, Inc.
WT32L064 EVB
Docu: WT32L064_LQFP64
Rev: 1.0
Customer: WT32L064_LQFP64
Date: Thursday, December 13, 2019 13:01:22 of 2

2. ARM-MDK 安裝與環境設定

(Step 1) 請先上網下載 **ARM-MDK** , <https://www.keil.com/download/>

The screenshot shows the Keil website's 'Product Downloads' page. A red box highlights the 'Product Downloads' section, with a yellow arrow pointing to the 'MDK-Arm' product. Another red box highlights the 'MDK-Arm' product, with a yellow arrow pointing to the 'MDK529.EXE' download button. A third red box highlights the 'MDK529.EXE' download button, with a yellow arrow pointing to the download link.

1. Product Downloads
Download current and previous versions of the Keil development tools.

File Downloads
Download example projects and various utilities which enable you to extend your development environment.

<https://www.keil.com/download/product/>

Download Products
Select a product from the list below to download the latest version.

2. MDK-Arm
Version 5.29 (November 2019)
Development environment for Cortex and Arm devices.

C51
Version 9.60a (May 2019)
Development tools for all 8051 devices.

C251
Version 5.60 (May 2018)
Development tools for all 80251 devices.

C166
Version 7.57 (May 2018)
Development tools for C166, XC166, & XC2000 MCUs.

Home / Product Downloads

MDK-ARM
MDK-ARM Version 5.29
Version 5.29

- Review the [hardware requirements](#) before installing this software.
- Note the [limitations of the evaluation tools](#).
- [Further installation instructions for MDK5](#)

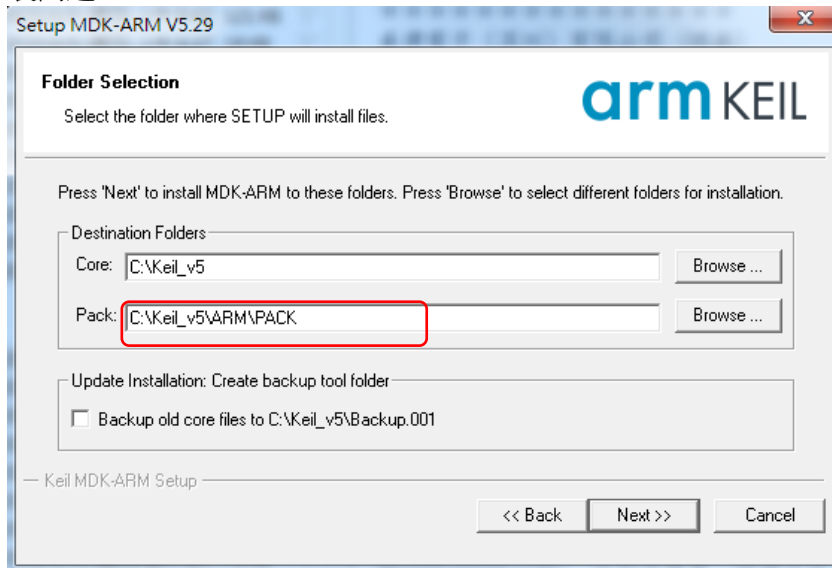
(MD5:0D0654419D24A7C2BAE6C4858504B350)

To install the MDK-ARM Software...

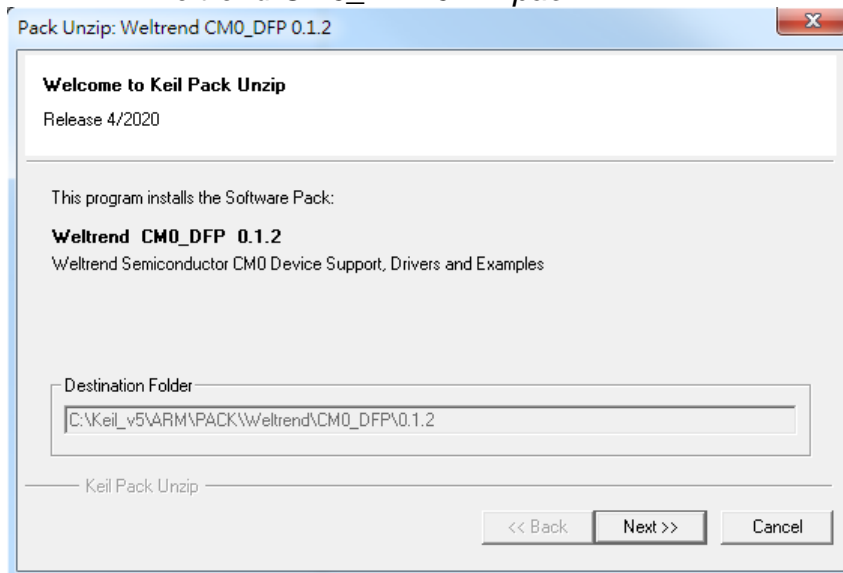
- Right-click on **MDK529.EXE** and save it to your computer.
- PDF files may be opened with Acrobat Reader.
- ZIP files may be opened with PKZIP or WINZIP.

3. MDK529.EXE (855,44K)
Monday, November 18, 2019

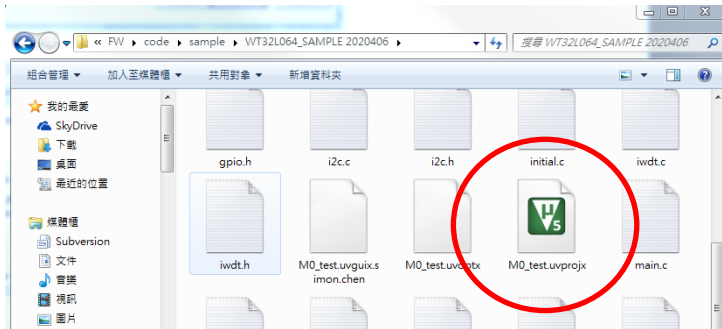
安裝過程中會詢問預設 PACK 路徑，請指定 **C:\Keil_v5\ARMPACK** 如下，避免後續 PACK 安裝問題



(Step 2) 下載並安裝 MDK 後，請於 PC 端再安裝偉詮 PACK 檔案 *Weltrend.CM0_DFP.0.1.x.pack*



(Step 3) 安裝 ARM-MDK 後有基礎 32KB 可免費使用，或可自行採購軟體，安裝後請在電腦上開啟相關 WT32L064 專案進行編譯工作。



3. CMSIS 中間層驅動說明

3.1 定義:

ARM® Cortex™ 微控制器軟體介面標準 (CMSIS) 是一組韌體庫可驅動 ARM 處理器，該韌體介面提供一標準函式直接面向周邊且名稱一致使用簡單，可利軟體的重複呼叫使用，縮短微控制器開發人員開發時間。於此架構上廠商再提供一組範例程式或周邊程式庫之基本周邊應用，直接面向應用端可加快程式操作與編寫。

3.2 CMSIS 內容說明:

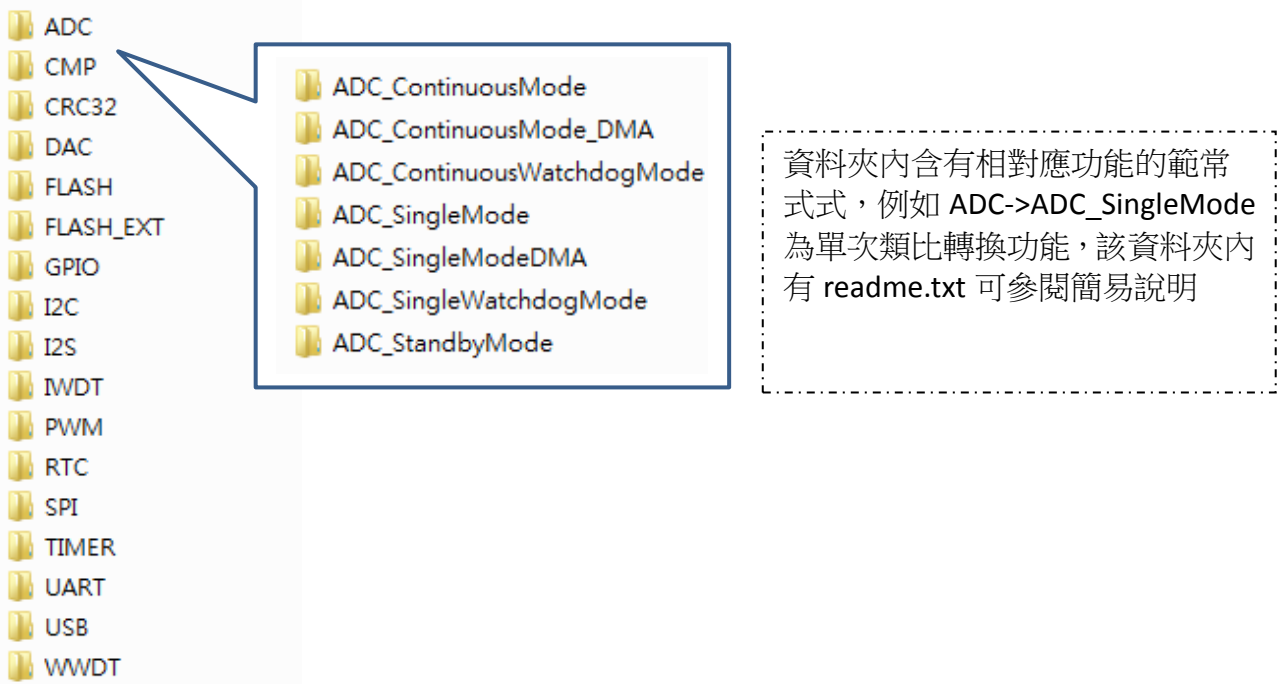
安裝完 WT32L064 PACK 後，預設 CMSIS 的路徑為 C:\Keil_v5\ARM\ Packs\Weltrend\CM0_DFP\0.1.x\WT32L064\StdPeriph_Driver，標頭檔放置 Include 資料夾，原始檔放置 Source，其內容有對 WT32L064 所有的周邊做基礎設定，檔案清單如下。

檔案名稱	功能說明
wt32l064_adc	類比偵測 ADC 相關函式
wt32l064_crc32	CRC32 計算關函式
wt32l064_crs	校正 IC 內部頻率相關函式
wt32l064_dac	類比輸出 DAC 相關函式
wt32l064_dma	直接記憶體存取 DMA 相關函式
wt32l064_flash	仿真式 EEPROM 燒錄 FLASH 相關函式
wt32l064_gpio	GPIO 相關函式
wt32l064_i2c	I2C 相關函式
wt32l064_i2s	I2S 相關函式
wt32l064_iwdt	IWDT 獨立看門狗相關函式
wt32l064_pmu	PMU 電源控制單元相關函式
wt32l064_pwm	PWM 相關函式
wt32l064_rcc	RCC 頻率控制單元相關函式
wt32l064_rtc	RTC 計時器相關函式
wt32l064_spi	SPI 相關函式
wt32l064_timer	TIMER 相關函式
wt32l064_usart	UART 相關函式
wt32l064_usbd	USB 相關函式
wt32l064_wwdt	WWDT 視窗型看門狗相關函式

4. PACK 範例程式架構說明

針對各種應用單元有基本範例程式，當 PACK 安裝後參考下列路徑

C:\...\Arm\Packs\Weltrend\CM0_DFP\0.1.x\WT32L064\Examples，資料夾內有子單元內含原始檔與專案，下列為 ADC 範例程式，依其功能有單一次轉換與連續轉換與其分類，如下圖示所示。



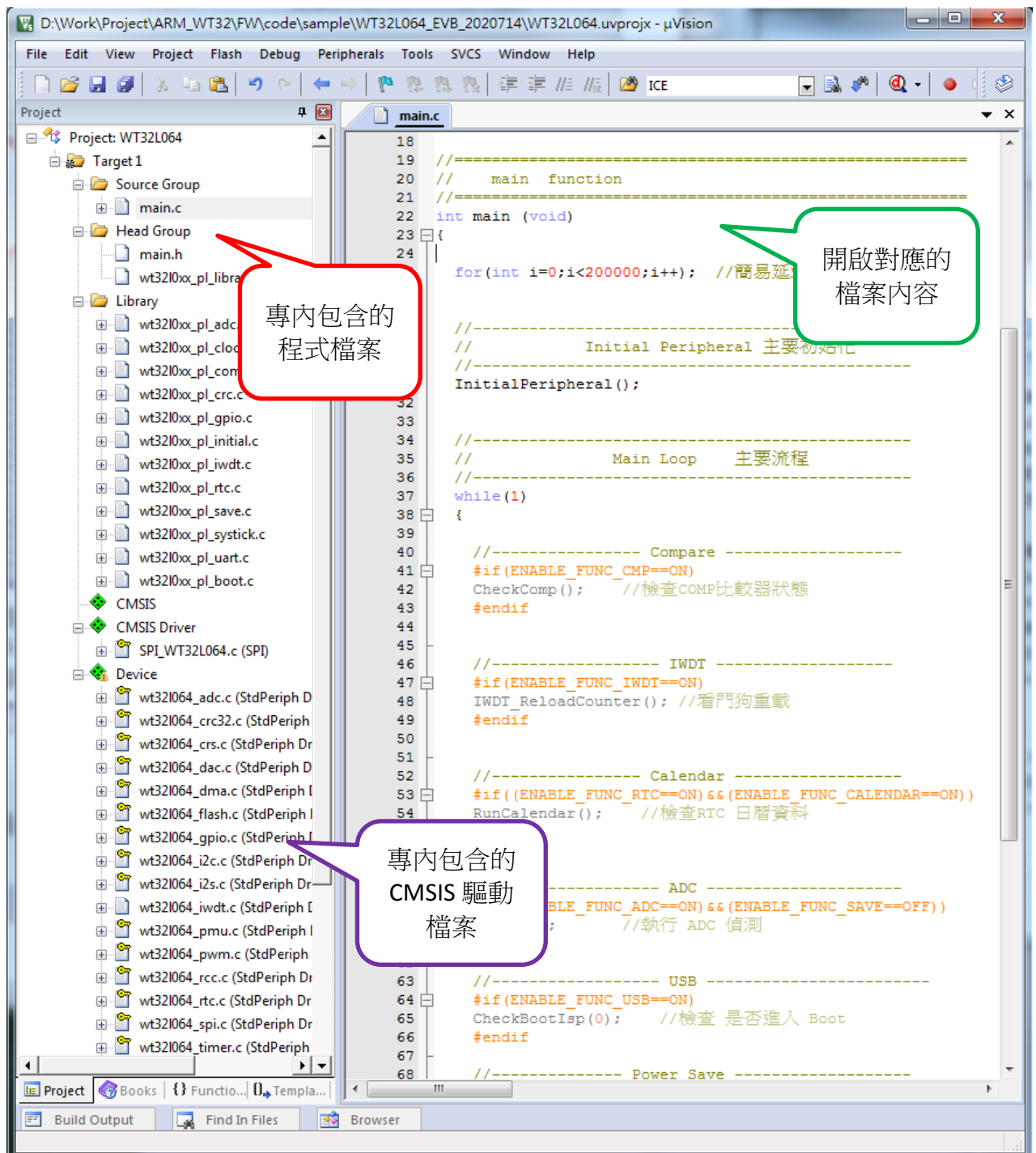
4.1 Examples 資料夾內功能說明

根資料夾	資料夾名稱	功能說明
ADC	ADC_ContinuousMode	連續 ADC 偵測
	ADC_ContinuousMode_DMA	使用 DAM 作連續 ADC 偵測
	ADC_ContinuousWatchdogMode	使用連續 ADC 做邊界偵測
	ADC_SingleMode	單一 ADC 偵測
	ADC_SingleModeDMA	使用 DAM 作單一 ADC 偵測
	ADC_SingleWatchdogMode	使用單一 ADC 做邊界偵測
	ADC_StandbyMode	使用低耗電 ADC 模式
CMP	CMP	比較器範例
CR32	CRC32	CRC32 計算範例
DAC	DAC	DAC 輸出範例
	DAC_HighCurrent	DAC 高推力輸出範例
FLASH	FLASH_PROGRAM	燒錄程式區(EEPROM)範例
	FLASH_PROGRAM_INT	燒錄程式區(EEPROM)中斷範例
FLASH_EXT	FLASH_OB_EEPROM	燒錄資料區(EEPROM)範例
	FLASH_OB_LEVEL	於資料區(OB)作加密等級

根資料夾	資料夾名稱	功能說明
	FLASH_OB_READ_PROTECTION	於資料區(OB)作防讀加密
	FLASH_OB_WRITE_PROTECTION	於資料區(OB)作防寫加密
GPIO	GPIO	GPIO 基本範例
	GPIO_Bit_Set_Reset	設定 GPIO 位元的範例
	GPIO_Input	設定 GPIO 輸入的範例
	GPIO_Interrupt	設定 GPIO 中斷的範例
	GPIO_Output	設定 GPIO 輸出的範例
	GPIO_Toggle	設定 GPIO 輸出反向的範例
I2C	I2C_Master_Slave_DMA_FLAG	I2C 從端模式與 DMA 搬移
	I2C_Master_Slave_DMA_INT	I2C 從端模式與 DMA 中斷
	I2C_Master_Slave_FLAG	I2C 從端模式
	I2C_Master_Slave_FLAG_EEPROM	I2C 從端模式與 EEPROM 燒錄
	I2C_Master_Slave_INT	I2C 主端與從端模式各一組互傳
I2S	I2S_DMA	I2S 從端模式與 DMA 搬移
	I2S_INT	I2S 從端模式與 DMA 中斷
	I2S_POLLING	I2S 從端模式
IWDT	IWDT	看門狗設定範例
PWM	PWM	PWM 脈波調變範例
RTC	RTC_1sec	RTC 計時器設定 1 秒範例
	RTC_Alarm	RTC 計時器設定鬧鐘範例
SPI	MSPI_DMA_FLAG	SPI 使用 DMA 傳輸範例
	MSPI_DMA_INT	SPI 使用 DMA 傳輸與中斷範例
	MSPI_FLAG	SPI 傳輸範例
	MSPI_FLAG_FLASH_MX25L4006	SPI 搭配 MX25L4006 傳輸範例
	MSPI_INT	SPI 傳輸與中斷範例
TIMER	TMR_Capture_Mode	Timer 捕捉模式範例
	TMR_Compare_Mode	Timer 比較模式範例
	TMR_Counter_Mode	Timer 計數模式範例
	TMR_DMA_Mode	Timer 搭配 DMA 使用範例
	TMR_PWM_MODE	Timer 輸出 PWM 使用範例
	TMR_Timer_Mode	Timer 普通計時範例
UART	UART_DMA	串列傳輸搭配 DMA 使用範例
	UART_HalfDuplexMode	串列傳輸使用半雙工範例
	UART_InterruptAndFlagManage	串列傳輸使用中斷範例
	UART_IrDA_Mode	串列傳輸使用 IRDA 範例
	UART_TxRx	串列傳輸同時發射與接收範例
USB	USB_HID	HID KEYBOARD 簡易範例
	USB_HID_AUDIO_WM8731	HID 搭配 I2S 撥放 WM8731 音樂
WWDT	WWDT	視窗型看門狗

5. 實例程式操作說明

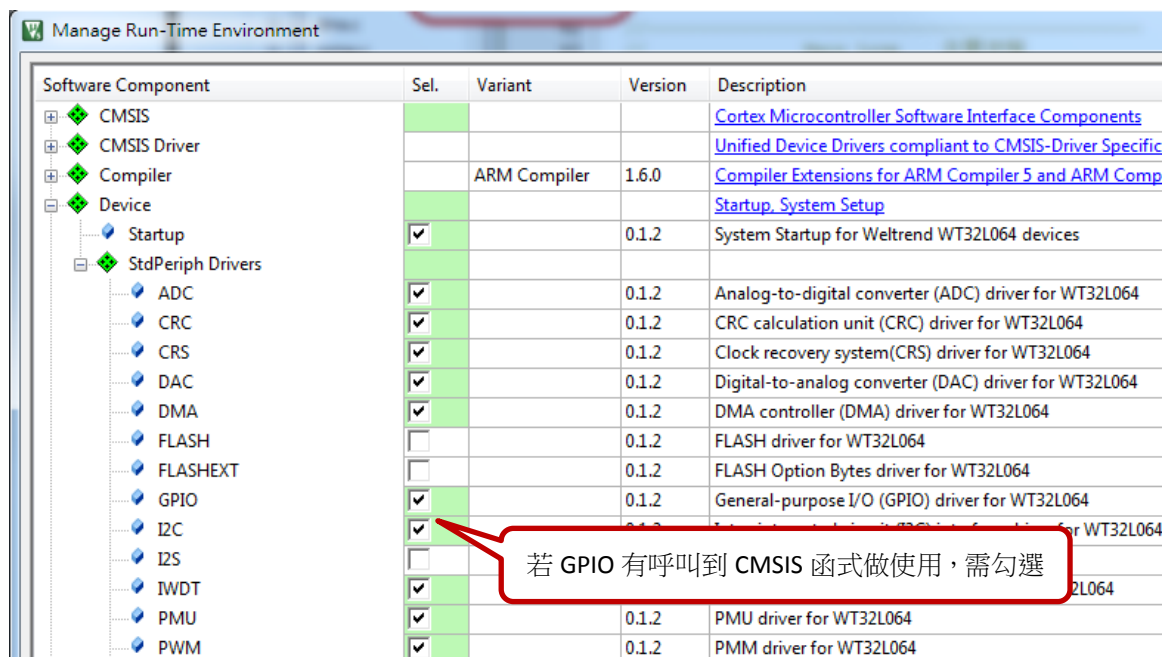
下列為如何使用參考範例的說明，專案名稱周邊程式庫參考前章節範例程式，依周邊功能放置個別檔案，開啟專案後的畫面如下，主要分三部分：專案包含檔案、CMSIS 驅動、各源始檔內容



針對周邊功能新增 CMSIS 驅動層函式，於 ARM-MDK 上方選單上點選 Manage Run-Time Environment 如下圖示。

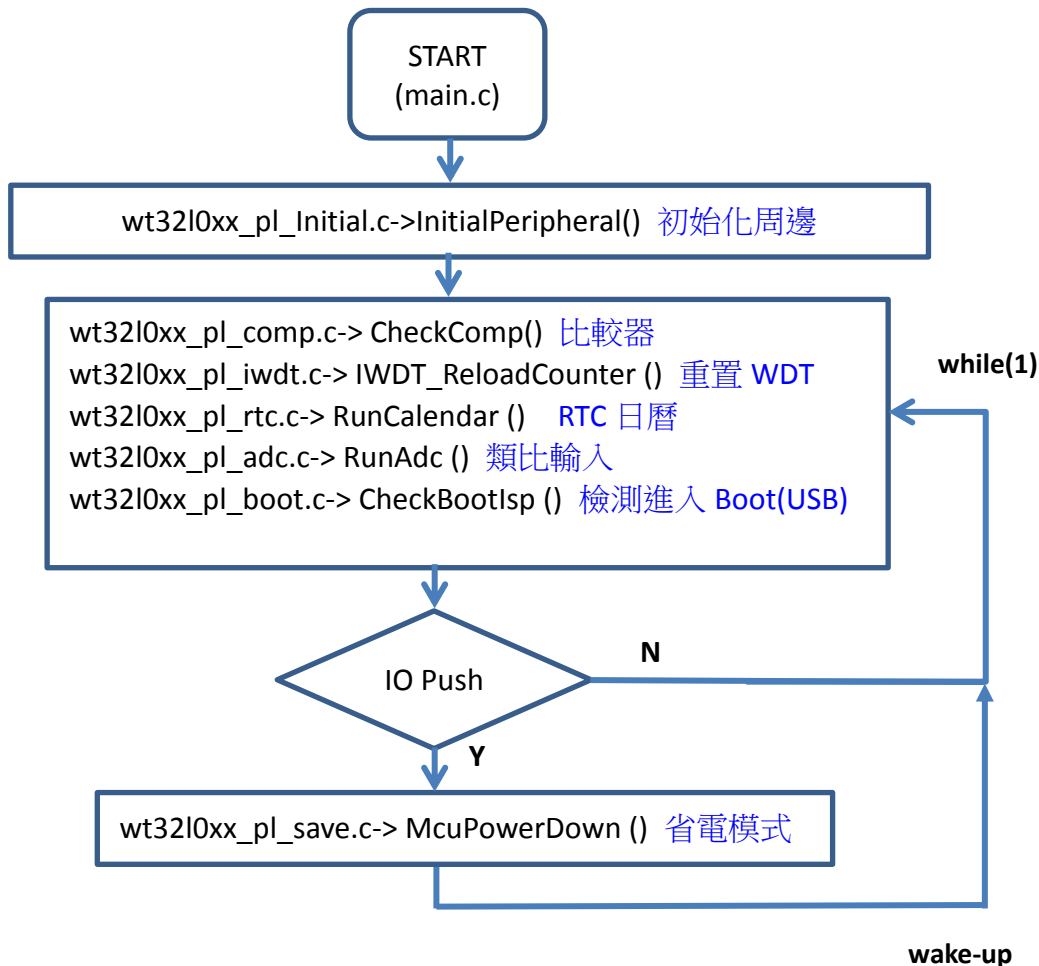


依序點選 Device->StdPeriph Drivers 如下圖所示，可依應用需求加入所需功能，例如：ADC、DAC、FLASH、GPIO、I2C 等等，一般範例程式都已加入參考到的 CMSIS，若有缺少部分或反黃項可再補加選。



5.1 範例流程圖

下列說明範例程式的流程圖、主要檔案內容與功能如下



依專案內檔案名稱與函式進行說明如下：

5.2 主程式流程

main.c 使用的函式如下：

- 1.) InitialPeripheral() -----參考到 wt32l0xx_pl_initial.c，對周邊的初始化
- 2.) CheckComp () -----參考到 wt32l0xx_pl_comp.c，比較器輸出結果
- 3.) IWDT_ReloadCounter() -----參考到 wt32l0xx_pl_iwdt.c，重置看門狗計數器
- 4.) RunCalendar() -----參考到 wt32l0xx_pl_rtc.c，檢測日曆數值
- 5.) RunAdc() -----參考到 wt32l0xx_pl_adc.c，執行 ADC 偵測
- 6.) CheckBootlsp() -----參考到 wt32l0xx_pl_boot.c，檢測 boot 狀況
- 7.) McuPowerDown() -----參考到 wt32l0xx_pl_save.c，執行省電功能

程式主迴圈內容如下:

```
int main(void)
{
    for (int i = 0; i < 200000; i++);    //Delay

    //-----
    //      Initial Peripheral 主要初始化
    //-----
    InitialPeripheral();

    //-----
    //      Main Loop          主要流程
    //-----
    while (1) {
        //----- Compare -----
        #if(ENABLE_FUNC_CMP==ON)
        CheckComp();    //檢查COMP比較器狀態
        #endif

        //----- IWDG -----
        #if(ENABLE_FUNC_IWDG==ON)
        IWDG_ReloadCounter();//看門狗重載
        #endif

        //----- Calendar -----
        #if((ENABLE_FUNC_RTC==ON)&&(ENABLE_FUNC_CALENDAR==ON))
        RunCalendar();    //檢查RTC 日曆資料
        #endif

        //----- ADC -----
        #if((ENABLE_FUNC_ADC==ON)&&(ENABLE_FUNC_SAVE==OFF))
        RunAdc();    //執行 ADC 偵測
        #endif

        //----- BOOT -----
        #if(ENABLE_FUNC_BOOT==ON)
        CheckBootIsr();    //檢查 是否進入 Boot
        #endif

        //----- Power Save -----
        if (GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_2) == 0) {    SysDelay(100);
            if (GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_2) == 0) {    //debounce

                //----- Sleep / Stop / Standby -----
                #if(ENABLE_FUNC_SAVE==ON)
                McuPowerDown();    //進入省電模式
                #endif
            }
        }
    };//while(1);
}
```

- wt32l0xx_pl_library.h 周邊功能的開關，請依需求依序開啟或關閉個別功能，程式內容如下。

```

//----- Enable Function for Project -----
// 請依序開啟下列功能， 使用ON為致能， OFF則為關閉

//----- Core -----
#define SELECT_CORE_1p2V          OFF //設定 1.8V // ON:
VCORE=1.2V
#define ENABLE_FUNC_CLOCK         ON  //設定 IRC 16M~32KHz
#define ENABLE_FUNC_LSI          OFF  //設定 LSI 37KHz 是否啟動

//----- IO LED -----
#define ENABLE_FUNC_GPIO         ON  //設定 GPIO 功能是否啟動
#if(ENABLE_FUNC_GPIO==ON)
#define ENABLE_FUNC_GPIO_INT    OFF  //設定 GPIO Interrupt 是否啟動
#define ENABLE_LED_BLINK        ON   //設定 GPIO Port-C LED 是否啟動
#define ENABLE_LED_RESET        OFF  //設定 GPIO 測試 Reset 是否啟動
#endif
#define ENABLE_FUNC_SYSTICK      ON   //設定 Systick 是否啟動

//----- Digital -----
#define ENABLE_FUNC_UART         ON  //設定 UART 功能是否啟動
#if(ENABLE_FUNC_UART==ON)
#define ENABLE_FUNC_UART0       ON   //設定 UART0 是否啟動
#define ENABLE_FUNC_UART1       OFF  //設定 UART1 是否啟動
#define ENABLE_HW_IRDA          OFF  //設定 IRDA是否啟動 使用 UART0+1
#endif

#define ENABLE_FUNC_IWDT         OFF //設定 IWDT 是否啟動

//----- Analog -----
#define ENABLE_FUNC_CMP          ON   //設定 COMPARE 是否啟動
#define ENABLE_HW_CMP_SPEED_HI  OFF  //HI:4.5uA LO:5.5uA

#define ENABLE_FUNC_ADC          OFF  //設定 ADC 是否啟動
#if(ENABLE_FUNC_ADC==ON)
#define ENABLE_HW_ADC_AWD        OFF
#define ENABLE_HW_ADC_ALL        OFF
#endif

//----- RTC -----
#define ENABLE_FUNC_RTC          OFF //設定 RTC 是否啟動
#if(ENABLE_FUNC_RTC==ON)
#define ENABLE_FUNC_ALARM        OFF //RTC Enable first (59 sec)
#define ENABLE_FUNC_CALEDAR      OFF //RTC Enable first (not for sleep)
#define ENABLE_RESET_RTC         OFF //ON: Test RTC keep RAM data
#endif

```

OFF: 關閉

ON: 開啟

數位功能類 開關

類比功能類 開關

RTC 功能開

```
//----- Power Save -----
#define ENABLE_LPRUN_MODE          OFF          //GPIO canot change without BLDO

#if(ENABLE_LPRUN_MODE==OFF)
#define     ENABLE_FUNC_SAVE          OFF
#endif
#if(ENABLE_FUNC_SAVE==ON)
#define ENABLE_STANDBY_MODE          OFF
#define ENABLE_SLEEP_MODE            OFF          //ENABLE_FUNC_SYSTICK must OFF
#define ENABLE_STOP_MODE              ON
#endif
#endif
```

省電功能開關

```
//----- wake up -----
#if(ENABLE_FUNC_SAVE==ON)
#define     ENABLE_WAKE_GPIO          ON          //STADBY must OFF
#define     ENABLE_WAKEUP_CMP         OFF
#define     ENABLE_WAKEUP_ADC         OFF
#define     ENABLE_WAKEUP_DAC         OFF          //Only Output
#define     ENABLE_WAKEUP_RTC         OFF
#define     ENABLE_WAKEUP_IWDT        OFF
#endif
#endif
```

喚醒功能開關

5.3 周邊功能的初始化

使用 wt32l0xx_pl_initial.c，包括函式如下：

- 1.) InitialPeripheral()-----初始化周邊功能: Systemtick、GPIO、UART、WDT、ADC、RTC、Comparator

初始化順序: InitialSysClock() -> InitialGpio() -> InitiSysTick() -> InitialUart0() -> InitialIwdt() -> InitialAdc() -> InitialRtc()->InitialComp()

5.3.1 工作頻率選擇

wt32l0xx_pl_clock.h 工作頻率的選擇，可選擇 HIS、MSI、HSE、PLL 四種類型，程式如下

```
//----- Use PLL for HSI 32MHz -----
#define CLOCK_PLL_HSI_X2_32MHZ     ON          //ON:開啟使用PLL倍頻HSI 16MHz至32Hz給系統使用

//----- Use PLL for USB 48MHz -----
#define USB_PLL                      0          // 0:HSI48M, 1:PLL(From external crystal)

//----- Select Frequency for MSI -----
#define MSI_65K                       PMU_MSIClock_Range0
#define MSI_131K                      PMU_MSIClock_Range1
#define MSI_262K                      PMU_MSIClock_Range2
```

```
#define MSI_524K          PMU_MSIClock_Range3
#define MSI_1M           PMU_MSIClock_Range4
#define MSI_2M           PMU_MSIClock_Range5
#define MSI_4M           PMU_MSIClock_Range6           //4.2MHz

#define MSI_CLOCK        MSI_4M           //當選擇MSI時，系統選擇的工作頻率

//----- Select MCU Clock Type -----
#define CLK_HSI          0 //Internal OSC 16MHz
#define CLK_MSI          1 //Internal OSC 65K~4M
#define CLK_PLL          2 //Use Multiple X with HSI or HSE
#define CLK_HSE          3 //External OSC 1~25MHz

#define SYS_CLOCK_SEL    CLK_MSI           //系統選擇頻率的類型
```

選擇速度

選擇類型

- wt32l0xx_pl_clock.c 工作頻率設置函式，包括函式如下
 - 1.) InitialSysClock () -----執行系統頻率選擇，節錄內容如下

```
#if(SYS_CLOCK_SEL==CLK_HSI) //使用 HSI 作系統頻率
    PMU_PowerClockCmd(PMU_PowerClock_HSI, ENABLE);
    PMU_SYSCLKConfig(PMU_SystemClk_HSI16);

#elif(SYS_CLOCK_SEL==CLK_MSI) //使用 MSI 作系統頻率
    PMU_MSISConfig(MSI_CLOCK); //Speed Setting
    PMU_PowerClockCmd(PMU_PowerClock_MSI, ENABLE); //Power-On PLL
    PMU_SYSCLKConfig(PMU_SystemClk_MSI); //Select System clock

#elif(SYS_CLOCK_SEL==CLK_PLL) //使用 PLL 作系統頻率
    //...省略

#elif(SYS_CLOCK_SEL==CLK_HSE) //使用 HSE 作系統頻率
```

- 2.) Delaysms() -----執行延遲功能
- 3.) DelayCount() -----執行延遲功能

5.3.2 周邊函式功能說明

- wt32l0xx_pl_gpio.c 外設 IO 類型設置，包括函式如下，可參考章節 4

- 1.) GPIO_Handler ()-----中斷服務 GPIO 功能
- 2.) InitialGpio ()-----初始化 GPIO 功能

```
GPIO的4種類型: GPIO_Mode_IN => 基本輸入
                GPIO_Mode_OUT =>基本輸出
                GPIO_Mode_AF =>複合使用功能，EX:UART、SPI、I2C ...
                GPIO_Mode_AN =>類比輸入功能，EX:ADC、USB、COMP ...
```

- wt32l0xx_pl_systick.c 內建 24bit 計時器設置，包括函式如下
 - 1.) SysTick_Handler ()-----中斷服務 systick 功能
 - 2.) InitSysTick ()-----初始化 systick 功能
 - 3.) SysDelay ()-----使用 systick 延遲功能

- wt32l0xx_pl_uart.c 異步收發器傳輸設置，包括函式如下，可參考章節 5
 - 1.) UART0_Handler ()-----中斷服務 UART0 功能
 - 2.) UART1_Handler()-----中斷服務 UART1 功能
 - 3.) InitialUart0 ()-----初始化 UART0 功能
 - 4.) InitialUart1()-----初始化 UART1 功能
 - 5.) fputc ()-----搭配 printf()使用發射串列資料功能
 - 6.) fgetc()-----搭配 printf()使用接收串列資料功能
 - 7.) DRV_IntToStr()-----數字轉字串
 - 8.) Str2Num()-----字串轉數字
 - 9.) uart_send_str()-----使用 UART0/1 發射串列資料
 - 10.) uart_clear_str()-----清除串列內容

- wt32l0xx_pl_adc.c 類比偵測設置，包括函式如下
 - 1.) ADC_Handler ()-----中斷服務 ADC 功能
 - 2.) InitialAdc ()-----初始化 ADC 功能
 - 3.) InitialAllAdc ()-----初始化 ADC 所有通道功能
 - 4.) RunAdc()-----執行 ADC 目標通道轉換功能
 - 5.) RunAllAdc ()-----執行 ADC 所有通道轉換功能
 - 6.) RunAdcConvert()-----執行 ADC 通道單次轉換功能
 - 7.) API_AverADCData ()-----執行 ADC 通道轉換功能，計算平均
 - 8.) ADC_StartOfConversion_1() -啟動 ADC 模組轉換
 - 9.) ADC_StopOfConversion_1() -停止 ADC 模組轉換
 - 10.) HEX2BCD()-----16 進制轉 10 進制

- wt32l0xx_pl_save.c 省電功能設置，包括函式如下
 - 1.) McuPowerDown ()-----執行省電功能前置作業並呼叫 Save()
 - 2.) Save()-----依設定 SLEEP、STOP、STANDBY 執行省電功能

- wt32l0xx_pl_rtc.c 實時計數器設置，包括函式如下
 - 1.) InitialRtc () -----執行 RTC 初始化
 - 2.) RTC_AlarmCmd ()-----執行 DAC 中斷功能
 - 3.) RTC_Handler()-----執行 RTC 中斷功能
 - 4.) RunCalendar()-----執行 RTC 日曆功能
 - 5.) SetAlarm()-----設定 RTC 鬧鐘功能

- wt32l0xx_pl_comp.c 比較器設置，包括函式如下
 - 1.) CheckComp () -----
 - 2.) CMP0_VOUT_Handler ()-----執行 CPM0 中斷功能
 - 3.) CMP1_VOUT_Handler()-----執行 CMP1 中斷功能
 - 4.) InitialComp()-----初始化 COMP 比較器
 - 5.) RumComp()-----執行 COMP 比較器

- wt32l0xx_pl_iwdt.c 看門狗設置，包括函式如下
 - 1.) InitialIwdt () ----- 初始化看門狗

- wt32l0xx_pl_boot.c 通用序列匯流排進入 Boot 設置，包括函式如下，詳細方式可參閱 "WT32L064_032 應用文件_使用 USB 更新 ISP 說明_v1x.pdf"
 - 1.) CheckBootIsp()-----檢查是否須進入 Boot 更新程式
 - 2.) enter_bootIsp()-----設定開機執行 Boot 功能，執行 MCU 復位

6. 版本更改紀錄:

版本	紀錄	日期
1.0	初始版本	2020/9/12