WT32L064 Application Note
EVB 与简易程式说明

(简体版)

Rev. 1.0

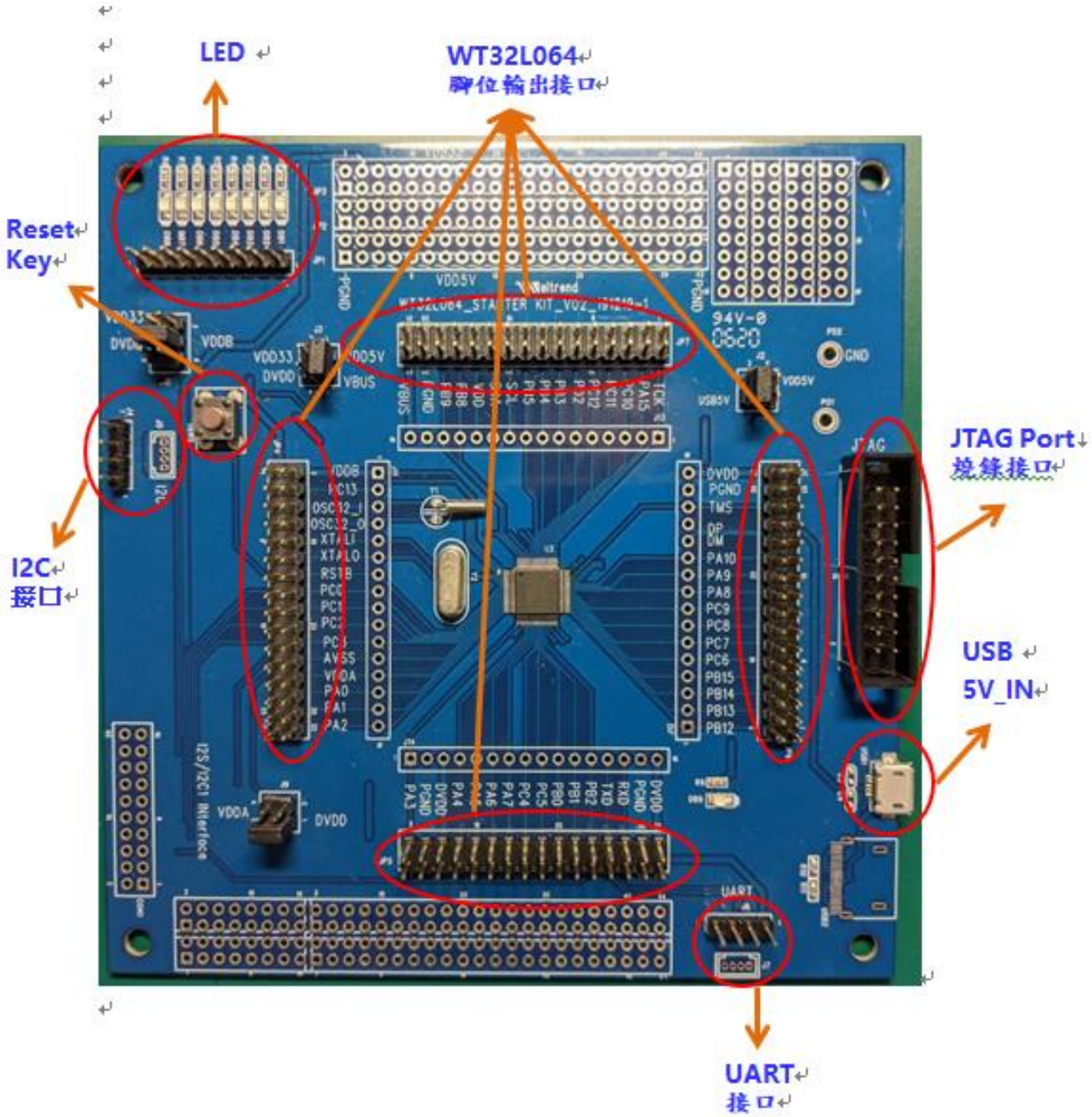
September 2020

目 录

1. EVB 功能说明

WT32L064 是一款超低功耗 ARM 32 位 Cortex M0 处理器，具有 64KB 嵌入式闪存和 8KB SRAM，而此 Starter Kit Board 则是使用 64-pin LQFP 包装作为设计并将其功能演示。

1.1 外观说明 ：



1.2 EVB 接口说明 ：

1.2.1 Micro USB Port (USB1)

此为 EVB 直流电压(5V)输入接口

脚位编号	说　明
1	VBUS
2	D-
3	D+
4	GND
5	GND

1.2.2 I2C 介面接口 (J4/J5)

此为 SLAVE I2C 介面接口

脚位编号	说　明
1	DVDD (3.3V)
2	SCL
3	SDA
4	GND

1.2.3 UART 介面接口 (J6/J7)

此为 UART 串列传输介面接口

脚位编号	说　明
1	DVDD (3.3V)
2	RXD
3	TXD
4	GND

1.2.4 JTAG 介面接口 (J6/J7)

此为 JTAG 传输介面接口
做为烧录程式及侦错检查使用

脚位编号	说　明	脚位编号	说　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 脚位输出接口 (JP4/JP5/FP6/JP7)

此为 WT32L064 脚位输出接口，提供外接测试使用

JP4			
脚位编号	说　明	脚位编号	说　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

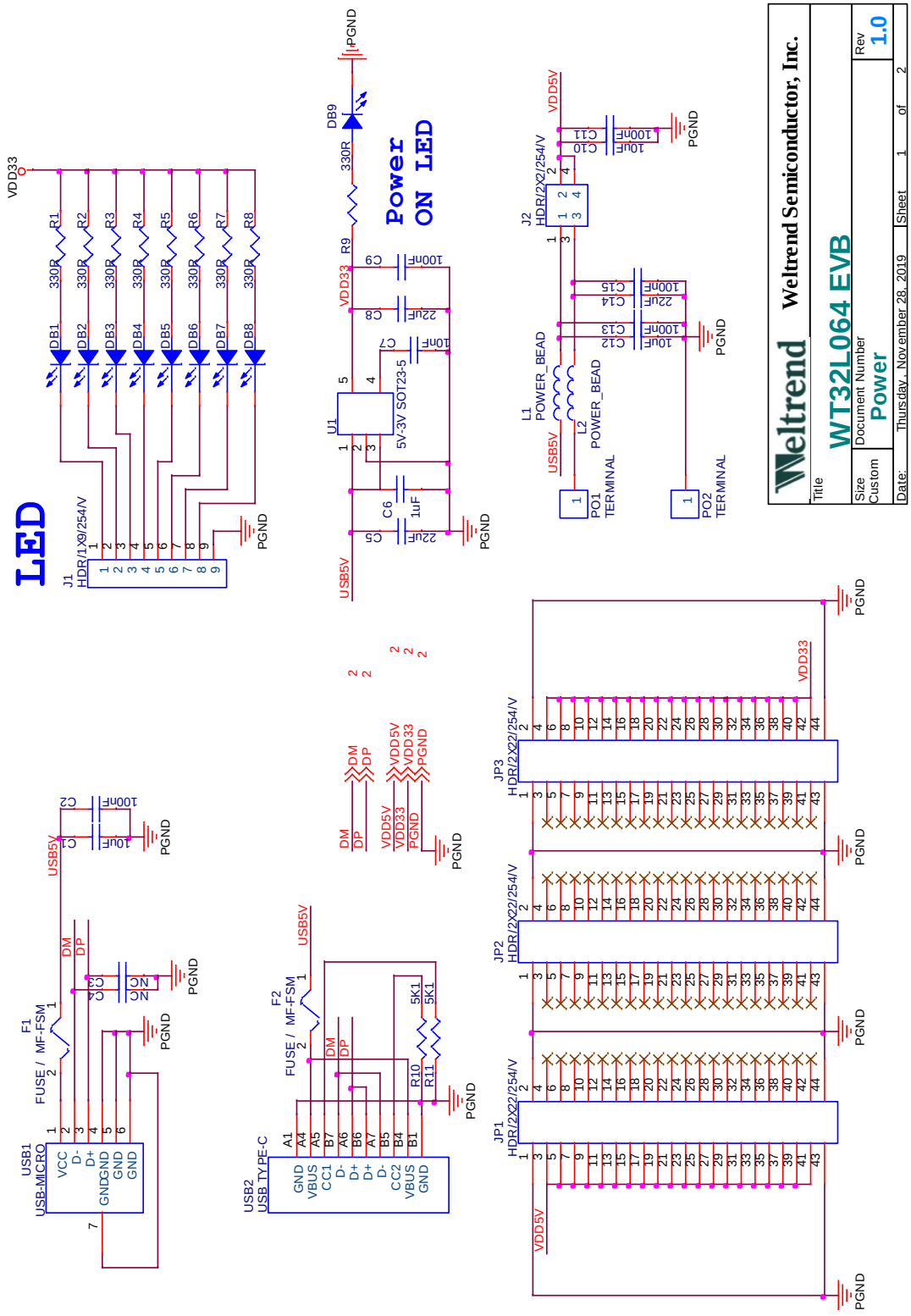
JP5			
脚位编号	说　明	脚位编号	说　明
1-2	GPIOA2	3-4	VSS
5-6	DVDD	7-8	GPIOA4
9-10	GPIOA5	11-12	GPIOA6
13-14	GPIOA7	15-16	GPIOC4
17-18	GPIOC5	19-20	GPIOB0
21-22	GPIOB1	23-24	GPIOB2
25-26	GPIOB10/TXD	27-28	GPIOB11/RXD
29-30	VSS	31-32	DVDD

JP6			
脚位编号	说　明	脚位编号	说　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

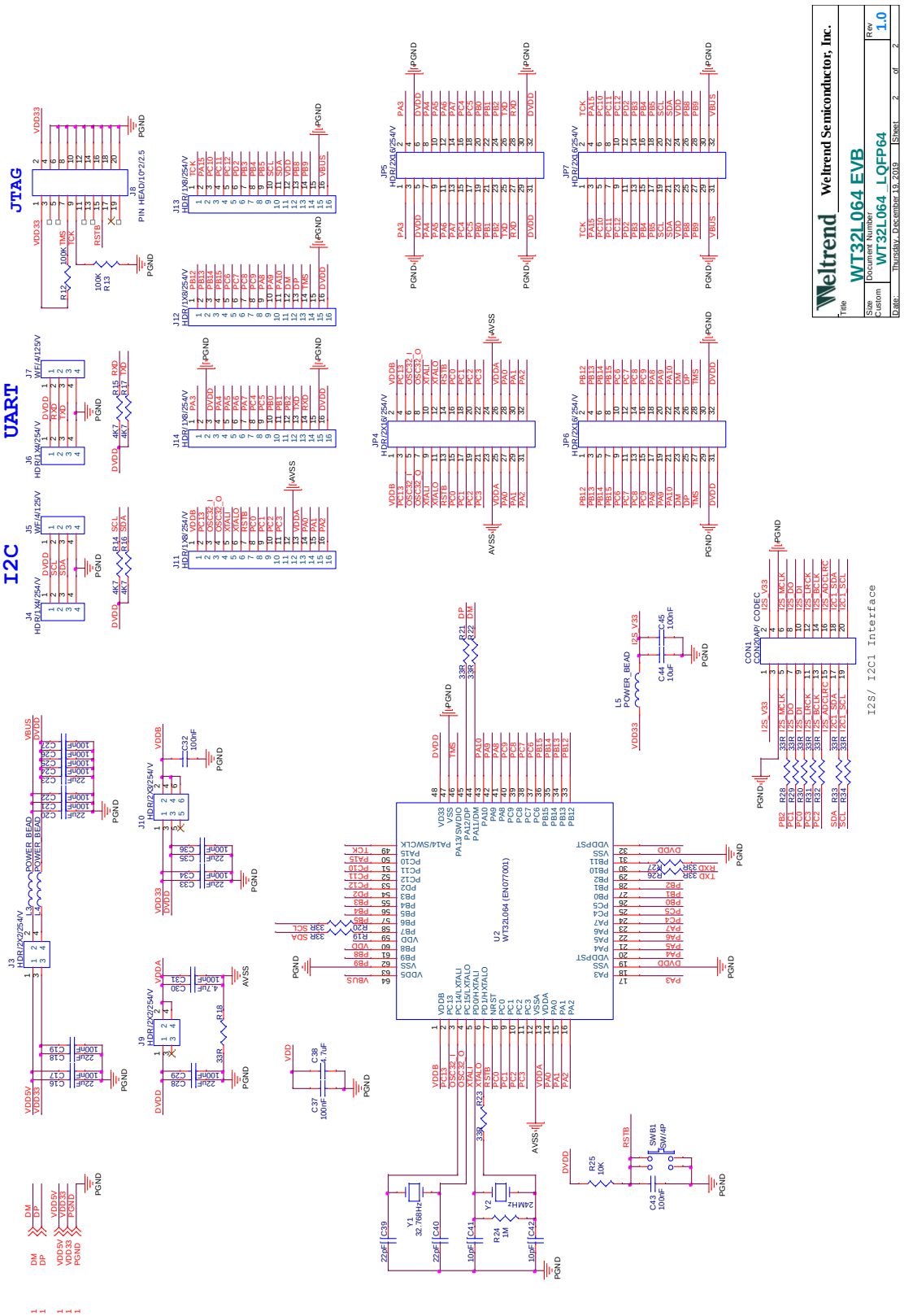
JP7			
脚位编号	说　明	脚位编号	说　明
1-2	GPIOA14/SWCLK	3-4	GPIOA15
5-6	GPIOC10	7-8	GPIOC11
9-10	GPIOC12	11-12	GPIOD2
13-14	GPIOB3	15-16	GPIOB4
17-18	GPIOB5	19-20	GPIOB6
21-22	GPIOB7	23-24	VDD
25-26	GPIOB8	27-28	GPIOB9
29-30	VSS	31-32	VDD5

1.3 EVB 线路图 ：

1.3.1 Power

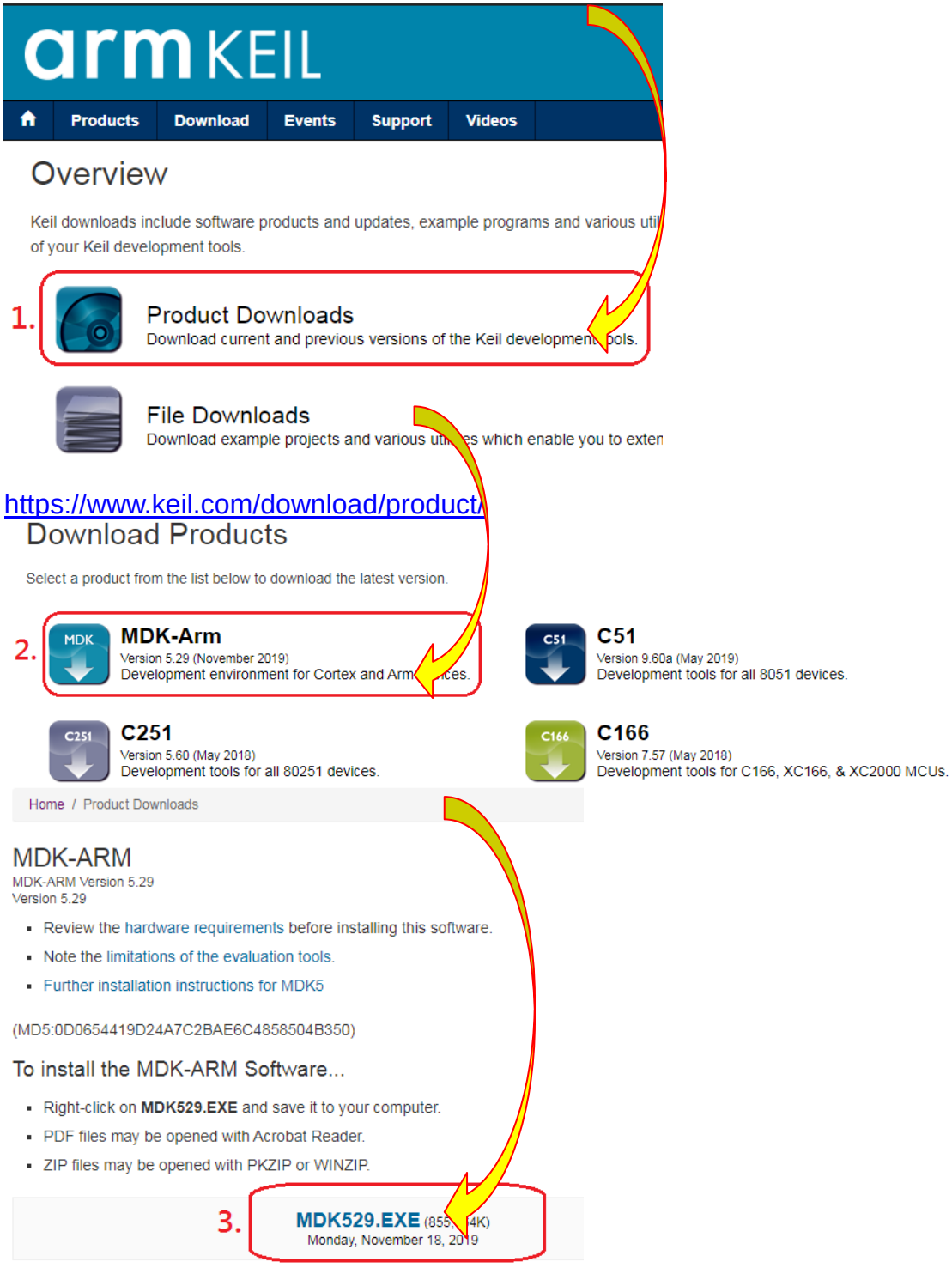


1.3.2 WT32L064

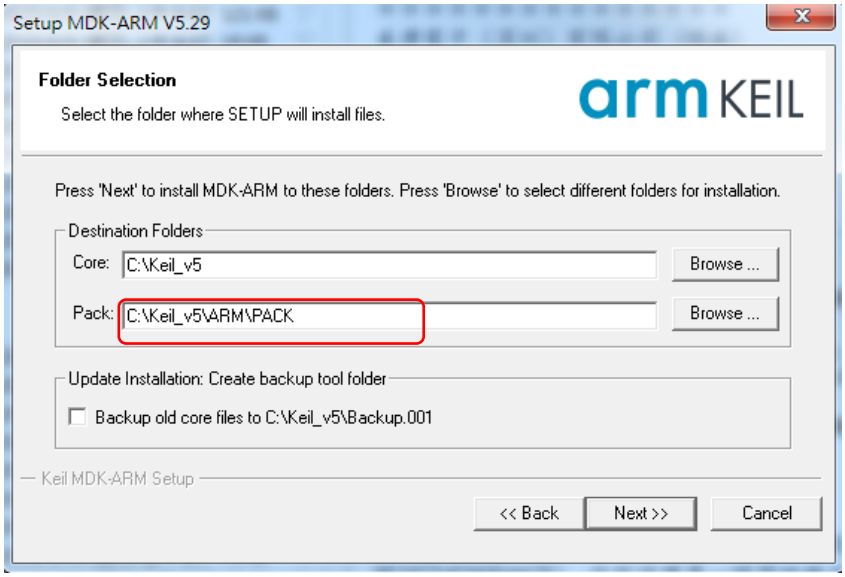


2. ARM-MDK 安装与环境设定

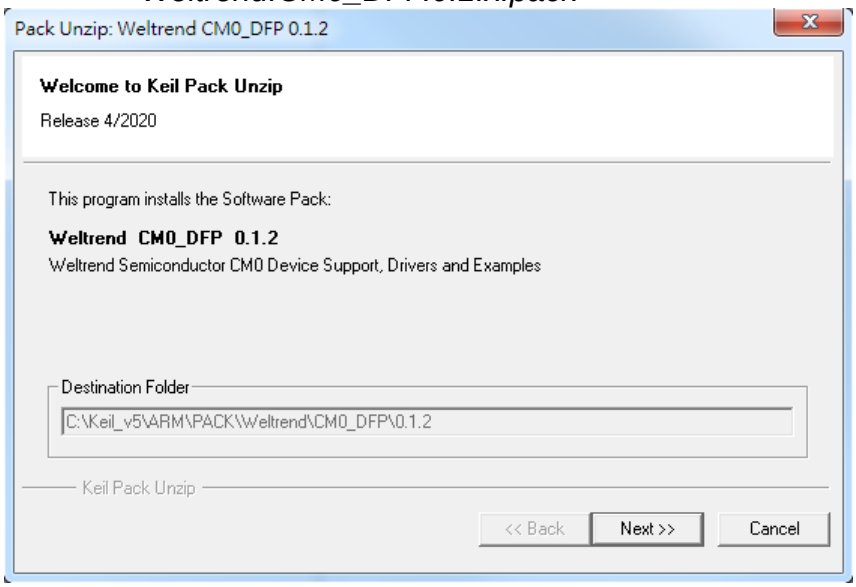
(Step 1) 请先上网下载 **ARM-MDK**，https://www.keil.com/download/



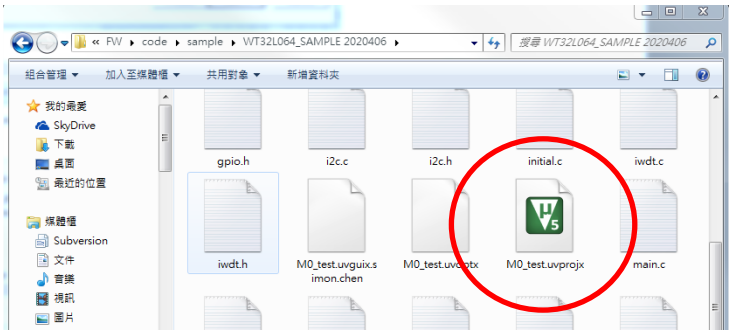
安装过程中会询问预设 PACK 路径，请指定 **C:\Keil_v5\ARM\PACK** 如下，避免后续 PACK 安装问题



(Step 2) 下载并安装 MDK 后，请于 PC 端再安装伟诠 PACK 档案 *Weltrend.CM0_DFP.0.1.x.pack*



(Step 3) 安装 ARM-MDK 后有基础 32KB 可免费使用，或可自行采购软体，安装后请在电脑上开启相关 WT32L064 专案进行编译工作。



3. CMSIS 中间层驱动说明

3.1 定义:

ARM® Cortex™ 微控制器软体介面标准 (CMSIS)是一组韧体库可驱动 ARM 处理器，该韧体介面提供一标准函式直接面向周边且名称一致使用简单，可利软体的重复呼叫使用，缩短微控制器开发人员开发时间。于此架构上厂商再提供一组范例程式或周边程式库之基本周边应用，直接面向应用端可加快程式操作与编写。

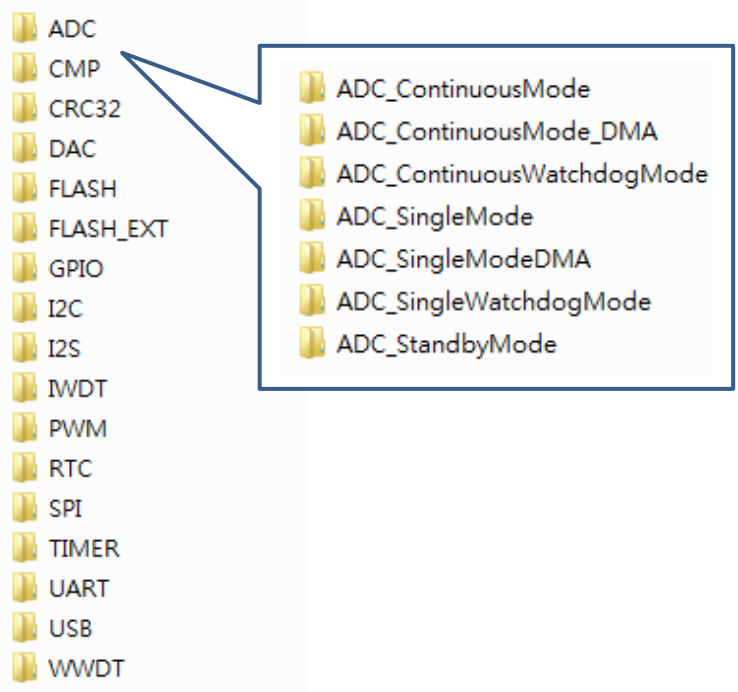
3.2 CMSIS 内容说明:

安装完 WT32L064 PACK 后，预设 CMSIS 的路径为 C:\Keil_v5\ARM\Packs\Weltrend\CM0_DFP\0.1.x \WT32L064\StdPeriph_Driver，标头档放置 Include 资料夹，原始档放置 Source，其内容有对 WT32L064 所有的周边做基础设定，档案清单如下。

档案名称	功能说明
wt32l064_adc	类比侦测 ADC 相关函式
wt32l064_crc32	CRC32 计算关函式
wt32l064_crs	校正 IC 内部频率相关函式
wt32l064_dac	类比输出 DAC 相关函式
wt32l064_dma	直接记忆体存取 DMA 相关函式
wt32l064_flash	仿真式 EEPROM 烧录 FLASH 相关函式
wt32l064_gpio	GPIO 相关函式
wt32l064_i2c	I2C 相关函式
wt32l064_i2s	I2S 相关函式
wt32l064_iwdt	IWDT 独立看门狗相关函式
wt32l064_pmu	PMU 电源控制单元相关函式
wt32l064_pwm	PWM 相关函式
wt32l064_rcc	RCC 频率控制单元相关函式
wt32l064_rtc	RTC 计时器相关函式
wt32l064_spi	SPI 相关函式
wt32l064_timer	TIMER 相关函式
wt32l064_usart	UART 相关函式
wt32l064_usbd	USB 相关函式
wt32l064_wwdt	WWDT 视窗型看门狗相关函式

4. PACK 范例程式架构说明

针对各种应用单元有基本范例程式，当 PACK 安装后参考下列路径 C:\...\Arm\Packs\Weltrend\ CM0_DFP\0.1.x \WT32L064\Examples，资料夹内有子单元内含原始档与专案，下列为 ADC 范例程式，依其功能有单一次转换与连续转换与其分类，如下图示所示。



资料夹内含有相对应功能的范常式式，例如 ADC->ADC_SingleMode 为单次类比转换功能，该资料夹内有 readme.txt 可参阅简易说明

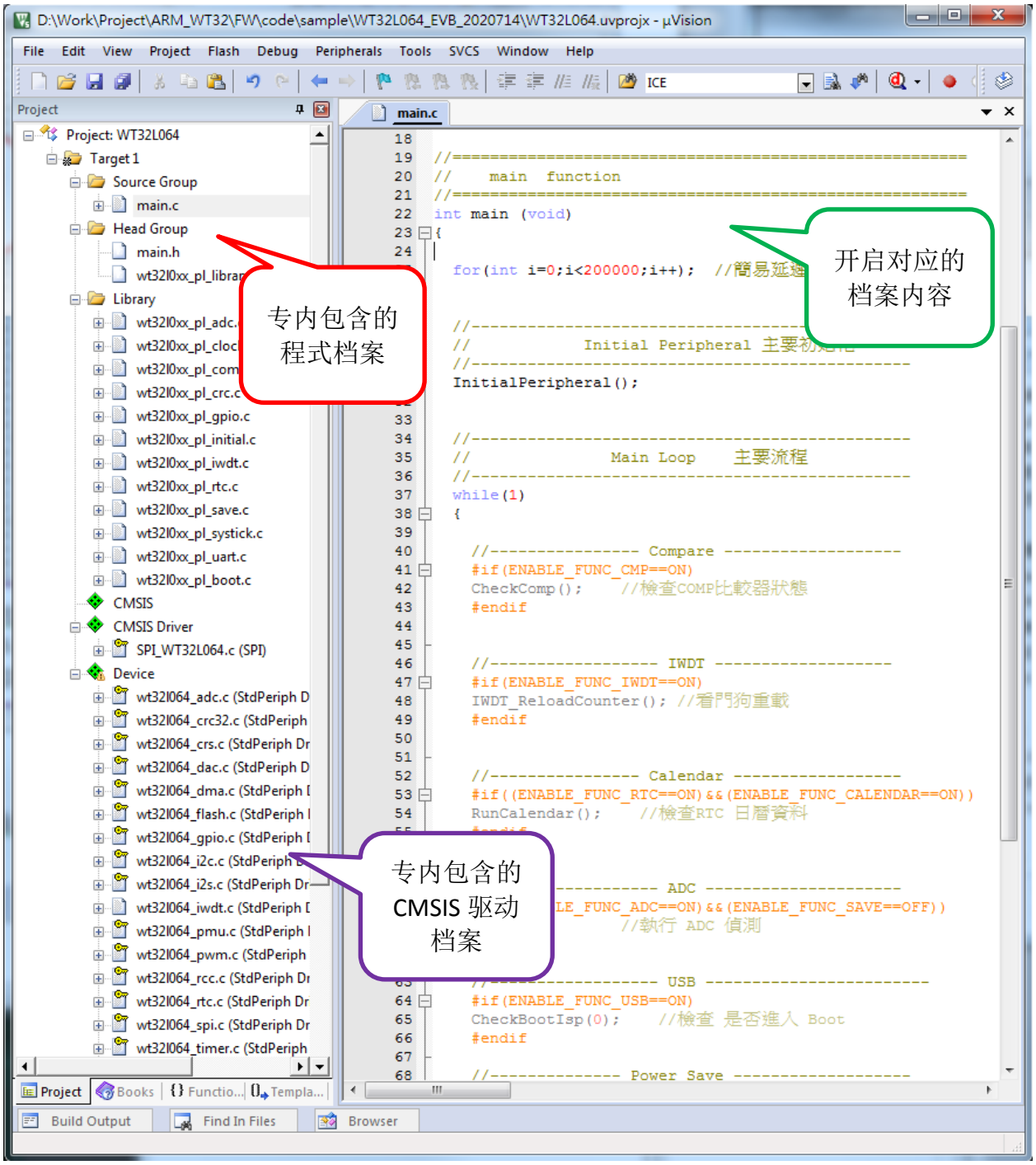
4.1 Examples 资料夹内功能说明

根资料夹	资料夹名称	功能说明
ADC	ADC_ContinuousMode	连续 ADC 侦测
	ADC_ContinuousMode_DMA	使用 DAM 作连续 ADC 侦测
	ADC_ContinuousWatchdogMode	使用连续 ADC 做边界侦测
	ADC_SingleMode	单一 ADC 侦测
	ADC_SingleModeDMA	使用 DAM 作单一 ADC 侦测
	ADC_SingleWatchdogMode	使用单一 ADC 做边界侦测
	ADC_StandbyMode	使用低耗电 ADC 模式
CMP	CMP	比较器范例
CR32	CRC32	CRC32 计算范例
DAC	DAC	DAC 输出范例
	DAC_HighCurrent	DAC 高推力输出范例
FLASH	FLASH_PROGRAM	烧录程式区(EEPROM)范例
	FLASH_PROGRAM_INT	烧录程式区(EEPROM)中断范例
FLASH_EXT	FLASH_OB_EEPROM	烧录资料区(EEPROM)范例
	FLASH_OB_LEVEL	于资料区(OB)作加密等级

根资料夹	资料夹名称	功能说明
	FLASH_OB_READ_PROTECTION	于资料区(OB)作防读加密
	FLASH_OB_WRITE_PROTECTION	于资料区(OB)作防写加密
GPIO	GPIO	GPIO 基本范例
	GPIO_Bit_Set_Reset	设定 GPIO 位元的范例
	GPIO_Input	设定 GPIO 输入的范例
	GPIO_Interrupt	设定 GPIO 中断的范例
	GPIO_Output	设定 GPIO 输出的范例
	GPIO_Toggle	设定 GPIO 输出反向的范例
I2C	I2C_Master_Slave_DMA_FLAG	I2C 从端模式与 DMA 搬移
	I2C_Master_Slave_DMA_INT	I2C 从端模式与 DMA 中断
	I2C_Master_Slave_FLAG	I2C 从端模式
	I2C_Master_Slave_FLAG_EEPROM	I2C 从端模式与 EEPROM 烧录
	I2C_Master_Slave_INT	I2C 主端与从端模式各一组互传
I2S	I2S_DMA	I2S 从端模式与 DMA 搬移
	I2S_INT	I2S 从端模式与 DMA 中断
	I2S_POLLING	I2S 从端模式
IWDT	IWDT	看门狗设定范例
PWM	PWM	PWM 脉波调变范例
RTC	RTC_1sec	RTC 计时器设定 1 秒范例
	RTC_Alarm	RTC 计时器设定闹钟范例
SPI	MSPI_DMA_FLAG	SPI 使用 DMA 传输范例
	MSPI_DMA_INT	SPI 使用 DMA 传输与中断范例
	MSPI_FLAG	SPI 传输范例
	MSPI_FLAG_FLASH_MX25L4006	SPI 搭配 MX25L4006 传输范例
	MSPI_INT	SPI 传输与中断范例
TIMER	TMR_Capture_Mode	Timer 捕捉模式范例
	TMR_Compare_Mode	Timer 比较模式范例
	TMR_Counter_Mode	Timer 计数模式范例
	TMR_DMA_Mode	Timer 搭配 DMA 使用范例
	TMR_PWM_MODE	Timer 输出 PWM 使用范例
	TMR_Timer_Mode	Timer 普通计时范例
UART	UART_DMA	串列传输搭配 DMA 使用范例
	UART_HalfDuplexMode	串列传输使用半双工范例
	UART_InterruptAndFlagManage	串列传输使用中断范例
	UART_IrDA_Mode	串列传输使用 IRDA 范例
	UART_TxRx	串列传输同时发射与接收范例
USB	USB_HID	HID KEYBOARD 简易范例
	USB_HID_AUDIO_WM8731	HID 搭配 I2S 拨放 WM8731 音乐
WWDT	WWDT	视窗型看门狗

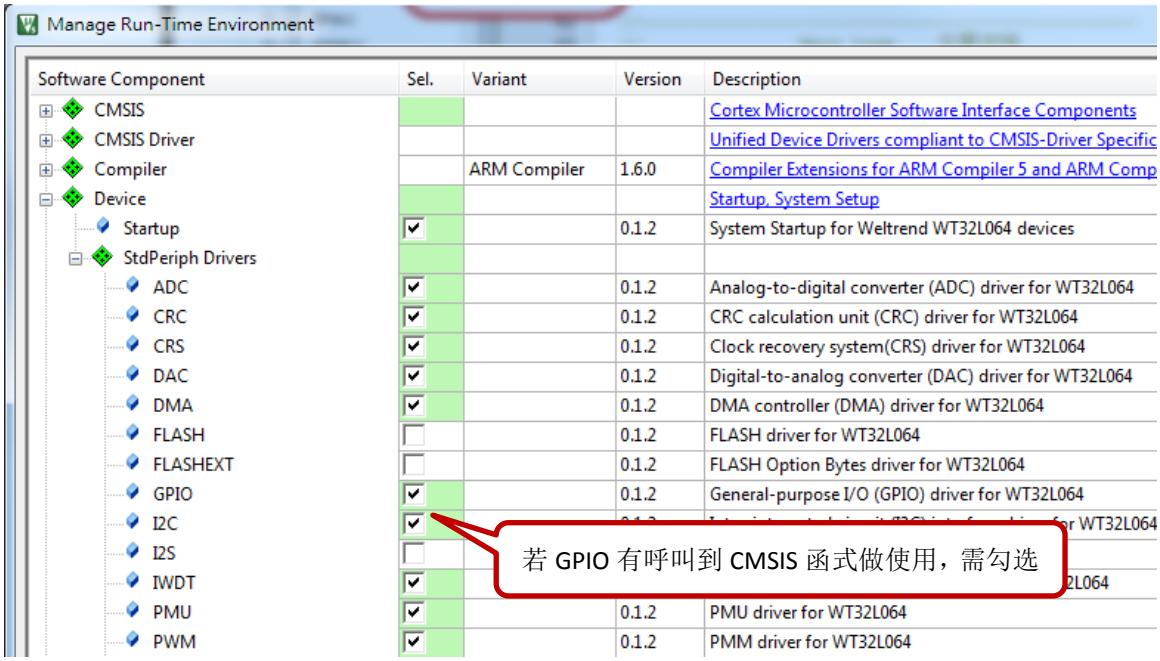
5. 实例程式操作说明

下列为如何使用参考范例的说明，专案名称周边程式库参考前章节范例程式，依周边功能放置个别档案，开启专案后的画面如下，主要分三部分: 专案包含档案、CMSIS 驱动、各源始档内容



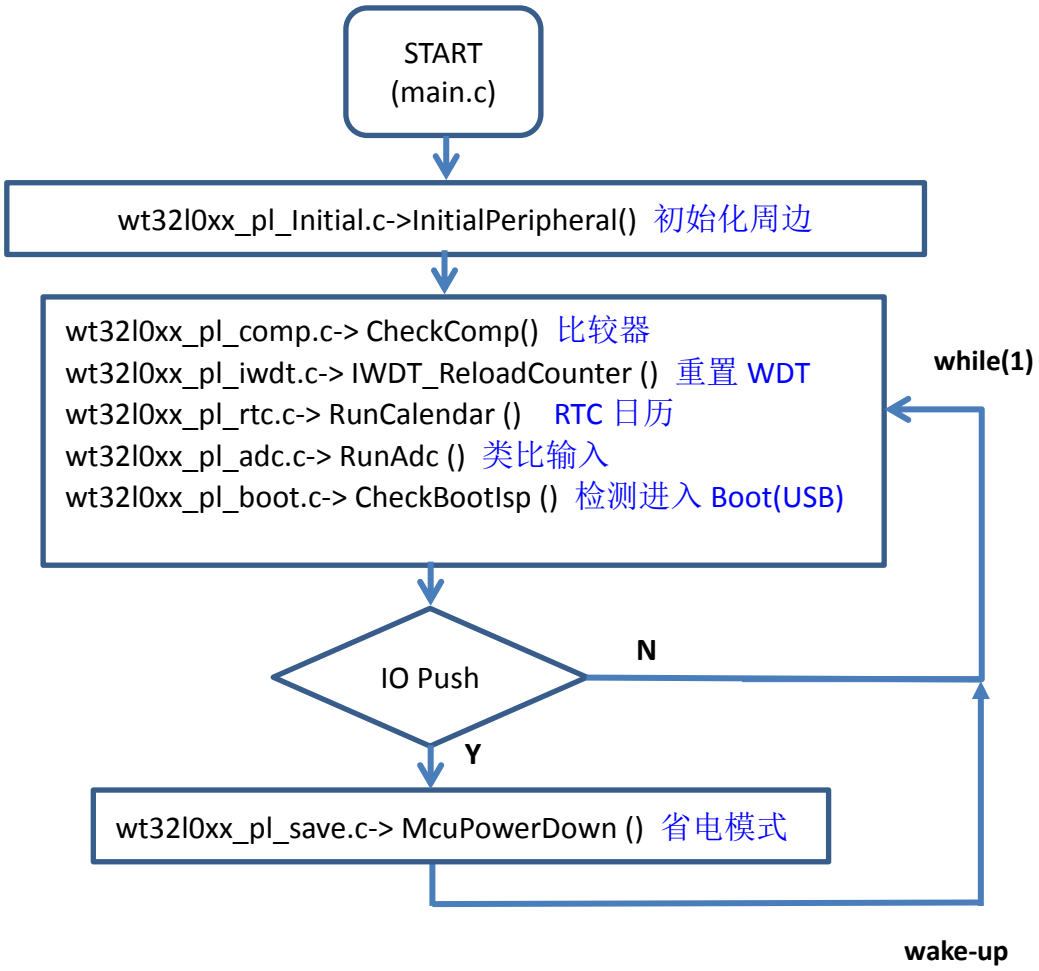
针对周边功能新增 CMSIS 驱动层函式，于 ARM-MDK 上方选单上点选 Manage Run-Time Environment 如下图示。

依序点选 Device->StdPeriph Drivers 如下图所示，可依应用需求加入所需功能，例如: ADC、DAC、FLASH、GPIO、I2C 等等，一般范例程式都已加入参考到的 CMSIS，若有缺少部分或反黄项可再补加选。



5.1 范例流程图

下列说明范例程式的流程图、主要档案内容与功能如下



依专案内档案名称与函式进行说明如下:

5.2 主程式流程

main.c 使用的函式如下:

1.) InitialPeripheral() ----------参考到 wt32l0xx_pl_initial.c，对周边的初始化
2.) CheckComp () ----------参考到 wt32l0xx_pl_comp.c，比较器输出结果
3.) IWDT_ReloadCounter() ------参考到 wt32l0xx_pl_iwdt.c，重置看门狗计数器
4.) RunCalendar() ----------参考到 wt32l0xx_pl_rtc.c，检测日历数值
5.) RunAdc() ----------参考到 wt32l0xx_pl_adc.c，执行 ADC 侦测
6.) CheckBootIsp() ----------参考到 wt32l0xx_pl_boot.c，检测 boot 状况
7.) McuPowerDown() ----------参考到 wt32l0xx_pl_save.c，执行省电功能

程式主回圈内容如下:

```
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();          //检查COMP比较器状态
        #endif

        //------------------ IWDT --------------------
        #if(ENABLE_FUNC_IWDT==ON)
        IWDT_ReloadCounter();//看门狗重载
        #endif

        //---------------- Calendar ------------------
        #if((ENABLE_FUNC_RTC==ON)&&(ENABLE_FUNC_CALENDAR==ON))
        RunCalendar();          //检查RTC  日历资料
        #endif

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

        //------------------ BOOT ----------------------
        #if(ENABLE_FUNC_BOOT==ON)
        CheckBootIsp();         //检查  是否进入  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();             //进入省电模式
              #endif
              }
        }
     };//while(1);
}
```

- wt32l0xx_pl_library.h 周边功能的开关，请依需求依序开启或关闭个别功能，程式内容如下。

```
//------------ Enable Function for Project ---------------------------
// 请依序开启下列功能， 使用ON为致能， OFF则为关闭

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


//------------ IO LED --------------------------
#define       ENABLE_FUNC_GPIO                  ON           // 设定 GPIO 功能是否启动
#if(ENABLE_FUNC_GPIO==ON)
#define ENABLE_GPIO_INT                 OFF          // 设定 GPIO Interrupt 是否启动
#define       ENABLE_LED_BLINK          ON           // 设定 GPIO Port-C LED 是否启动
#define       ENABLE_LED_RESET          OFF          // 设定 GPIO 测试 Reset 是否启动
#endif
#define       ENABLE_FUNC_SYSTICK       ON           // 设定 Systick 是否启动

//------------ Digtal ---------------
#define       ENABLE_FUNC_UART          ON           // 设定 UART 功能是否启动
#if(ENABLE_FUNC_UART==ON)
#define ENABLE_FUNC_UART0               ON           // 设定 UART0 是否启动
#define ENABLE_FUNC_UART1               OFF          // 设定 UART1 是否启动
#define ENABLE_HW_IRDA            OFF          // 设定 IRDA是否启动 使用 UART0+1
#endif

#define       ENABLE_FUNC_IWDT                  OFF // 设定 IWDT 是否启动


//----------- Analog -----------------
#define       ENABLE_FUNC_CMP           ON           // 设定 COMPARE 是否启动
#define       ENABLE_HW_CMP_SPEED_HI    OFF          //HI:4.5uA  LO:5.5uA

#define       ENABLE_FUNC_ADC           OFF          // 设定 ADC 是否启动
#if(ENABLE_FUNC_ADC==ON)
#define ENABLE_HW_ADC_AWD               OFF
#define ENABLE_HW_ADC_ALL               OFF
#endif


//-------- RTC -------
#define       ENABLE_FUNC_RTC                   OFF // 设定 RTC 是否启动
#if(ENABLE_FUNC_RTC==ON)
#define       ENABLE_FUNC_ALARM                 OFF //RTC Enable first (59 sec)
#define       ENABLE_FUNC_CALENDAR      OFF                 //RTC Enable first (not for sleep)
#define       ENABLE_RESET_RTC                  OFF //ON: Test RTC keep RAM data
#endif
```

OFF: 关闭

ON: 开启

数位功能类 开关

类比功能类 开关

RTC 功能开

省电功能开关

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

#if(ENABLE_LPRUN_MODE==OFF)
#define     ENABLE_FUNC_SAVE        OFF
#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
```

唤醒功能开关

```
//----- 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
```

5.3 周边功能的初始化

使用 wt32l0xx_pl_initial.c，包括函式如下:

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

初始化顺序: **InitialSysClock() -> InitialGpio() -> InitiSysTick() -> InitialUart0() -> InitialIwdt() -> InitialAdc() -> InitialRtc()->InitialComp()**

5.3.1 工作频率选择

wt32l0xx_pl_clock.h 工作频率的选择，可选择 HIS、MSI、HSE、PLL 四种类型，程式如下

```
//--------- Use PLL for HSI 32MHz -----------------------------
#define CLOCK_PLL_HSI_X2_32MHZ    ON    //ON:开启使用PLL倍频HSI 16MHz至32Hz给系统使用

//--------- 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          //当选择MSI时，系统选择的工作频率

//--------- 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         //系统选择频率的类型
```

选择速度

选择类型

- wt32l0xx_pl_clock.c 工作频率设置函式，包括函式如下

1.) InitialSysClock () ----------执行系统频率选择，节录内容如下

```
#if(SYS_CLOCK_SEL==CLK_HSI)         //使用 HSI 作系统频率
    PMU_PowerClockCmd(PMU_PowerClock_HSI, ENABLE);
    PMU_SYSCLKConfig(PMU_SystemClk_HSI16);

#elif(SYS_CLOCK_SEL==CLK_MSI)       //使用 MSI 作系统频率
    PMU_MSIConfig(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 作系统频率
    //...省略
#elif(SYS_CLOCK_SEL==CLK_HSE)       //使用 HSE 作系统频率
```

2.) Delayms() ----------执行延迟功能

3.) DelayCount() ----------执行延迟功能

5.3.2 周边函式功能说明

- wt32l0xx_pl_gpio.c 外设 IO 类型设置，包括函式如下，可参考章节 4

1.) GPIO_Handler ()----------中断服务 GPIO 功能

2.) InitialGpio ()----------初始化 GPIO 功能

```
GPIO的4种类型: GPIO_Mode_IN => 基本输入
               GPIO_Mode_OUT =>基本输出
               GPIO_Mode_AF =>复合使用功能，EX:UART、SPI、I2C …
               GPIO_Mode_AN =>类比输入功能，EX:ADC、USB、COMP …
```

- wt32l0xx_pl_systick.c 内建 24bit 计时器设置，包括函式如下

1.) SysTick_Handler ()----------中断服务 systick 功能
2.) InitiSysTick ()----------初始化 systick 功能
3.) SysDelay ()----------使用 systick 延迟功能

- wt32l0xx_pl_uart.c 异步收发器传输设置，包括函式如下，可参考章节 5

1.) UART0_Handler ()----------中断服务 UART0 功能
2.) UART1_Handler()----------中断服务 UART1 功能
3.) InitialUart0 ()----------初始化 UART0 功能
4.) InitialUart1()----------初始化 UART1 功能
5.) fputc ()----------搭配 printf()使用发射串列资料功能
6.) fgetc()----------搭配 printf()使用接收串列资料功能
7.) DRV_IntToStr()----------数字转字串
8.) Str2Num()----------字串转数字
9.) uart_send_str()----------使用 UART0/1 发射串列资料
10.) uart_clear_str()----------清除串列内容

- wt32l0xx_pl_adc.c 类比侦测设置，包括函式如下

1.) ADC_Handler ()----------中断服务 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()----------执行 ADC 目标通道转换功能
5.) RunAllAdc ()----------执行 ADC 所有通道转换功能
6.) RunAdcConvert()----------执行 ADC 通道单次转换功能
7.) API_AverADCData ()----------执行 ADC 通道转换功能，计算平均
8.) ADC_StartOfConversion_1() -启动 ADC 模组转换
9.) ADC_StopOfConversion_1() -停止 ADC 模组转换
10.) HEX2BCD()----------16 进制转 10 进制

- wt32l0xx_pl_save.c 省电功能设置，包括函式如下

1.) McuPowerDown ()----------执行省电功能前置作业并呼叫 Save()
2.) Save()----------依设定 SLEEP、STOP、STANDBY 执行省电功能

- wt32l0xx_pl_rtc.c 实时计数器设置，包括函式如下

1.) InitialRtc () -------------------------执行 RTC 初始化
2.) RTC_AlarmCmd ()-------------------执行 DAC 中断功能
3.) RTC_Handler()----------------------执行 RTC 中断功能
4.) RunCalendar()----------------------执行 RTC 日历功能
5.) SetAlarm()--------------------------设定 RTC 闹钟功能

- wt32l0xx_pl_comp.c 比较器设置，包括函式如下

1.) CheckComp () ----------------------
2.) CMP0_VOUT_Handler ()-------------------执行 CPM0 中断功能
3.) CMP1_VOUT_Handler()-------------------执行 CMP1 中断功能
4.) InitialComp()---------------------------------初始化 COMP 比较器
5.) RumComp()-----------------------------------执行 COMP 比较器

- wt32l0xx_pl_iwdt.c 看门狗设置，包括函式如下

1.) InitialIwdt () ---------------------- 初始化看门狗

- wt32l0xx_pl_boot.c 通用序列汇流排进入 Boot 设置，包括函式如下，详细方式可参阅 ”WT32L064_032 应用文件_使用 USB 更新 ISP 说明_v1x.pdf”

1.) CheckBootIsp()----------------------检查是否须进入 Boot 更新程式
2.) enter_bootisp()---------------------设定开机执行 Boot 功能，执行 MCU 复位

6. 版本更改纪录:

版 本	纪 录	日 期
1.0	初始版本	2020/9/12