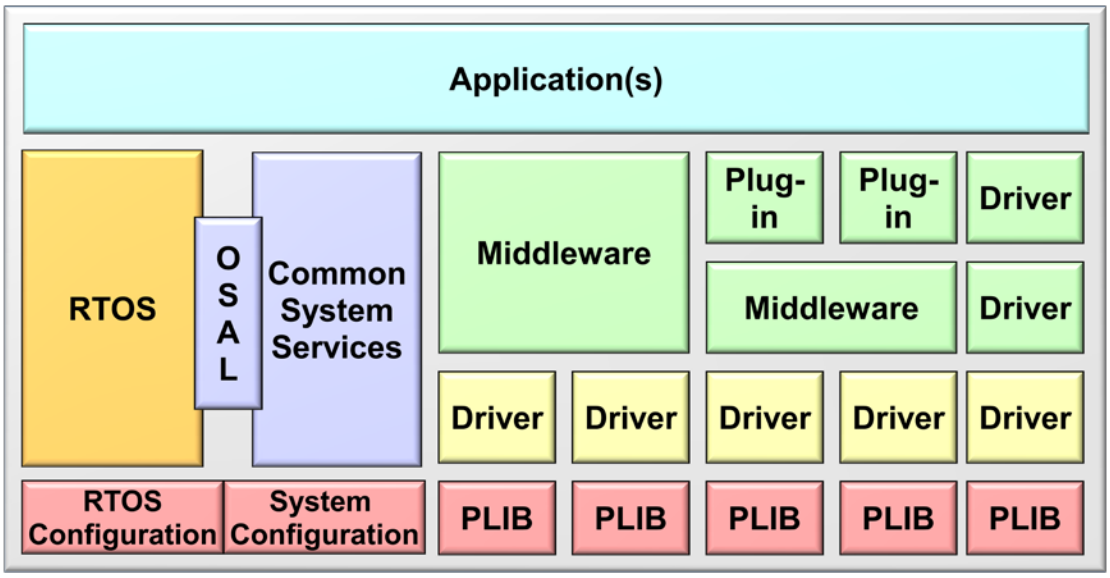
MPLAB® Harmony 之学习篇（八）

-- 利用 Harmony 已有的应用示例集成自己的应用

Microchip Technology Inc.
MCU32产品部
资深应用工程师
王翀

Harmony 框架如下图所示，接下来我们介绍一下 Harmony 框架下的应用示例以及如何利用已有的示例开发自己的应用程序。



Harmony 框架下提供的应用示例：

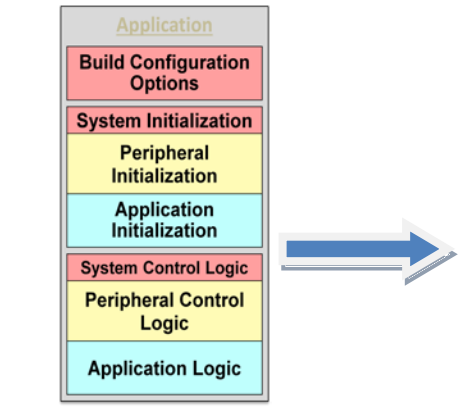
Harmony 下提供了很多应用程序示例。每个应用示例包含一个 main 函数，一个或多个独立的软件模块（设备驱动，中间件，以及系统服务）。示例展示了 Harmony 软件架构的各个模块是如何被上层调用的。

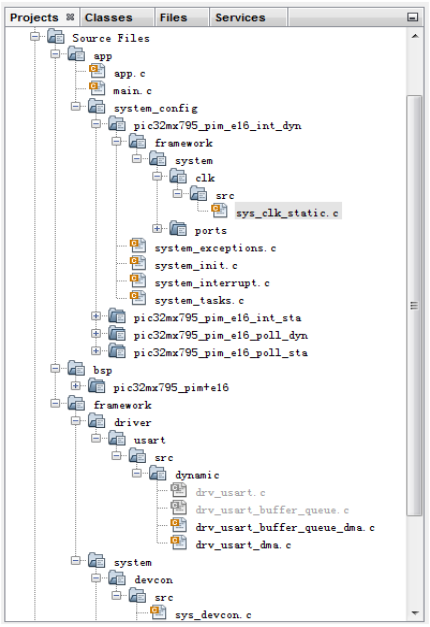
打开 Harmony 安装目录的 apps 文件夹（\harmony\v1_0x\apps），就可以看到如下目录结构：

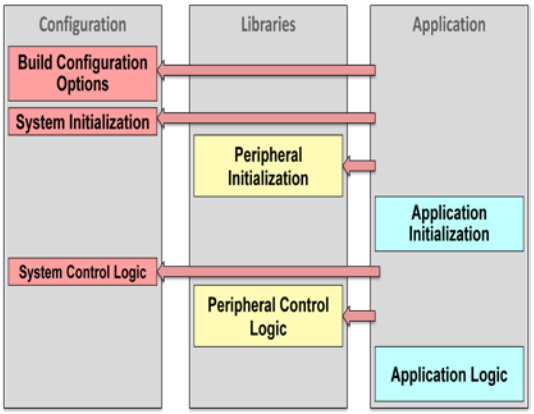
- audio 目录：包含所有音频处理相关的应用程序示例，包括麦克风，音频解码，USB 耳机等。
- bluetooth 目录：包含蓝牙相关的应用程序示例，包括基础蓝牙协议栈，蓝颜数据和蓝颜音频的应用示例。
- bootloader 目录：bootloader 程序是驻留在芯片中的一段代码，它可以帮用户在不用调试/编程工具的情况下，烧写、更新芯片的应用程序。该目录下的应用示例展示了通过 USB，串口，以太网来烧写应用程序的 bootloader，以及 MZ 系列芯片的 LiveUpdate 特性。
- crypto 目录：加密程序实例。
- driver 目录：动态应用实例，包括 I^2C，USART，SPI 和 NVM 驱动示例，其中有静态驱动也有动态驱动。
- example 目录：包括一些简单的示例，如所有硬件外设和部分系统服务等等。
- fs 目录：FAT 文件系统相关示例。支持的存储介质包括芯片内部 Flash，SD 卡和 SQI Flash。
- gfx 目录：图形显示相关示例，包括图形原语和控件的显示示例。
- meb_ii 目录：MEB2 是 Microchip PIC32 处理器配套开发板，具有蓝牙，WIFI，LCD 显示等多种功能。
- programmer 目录：包含外部 Flash 读写示例，可以通过它烧写外部 SPI 或 SQI Flash。
- rtos 目录：操作系统示例，包括 FreeRTOS，ucOS，OpenRTOS 等等。
- tcpip 目录：网络相关的示例，如 TCP，UDP client/server 示例等。
- test 目录：该目录下的示例用来测试操作系统或无操作系统下，各个软件库是否能正常工作。
- usb 目录：包含大量的 USB 主从设备应用实例。

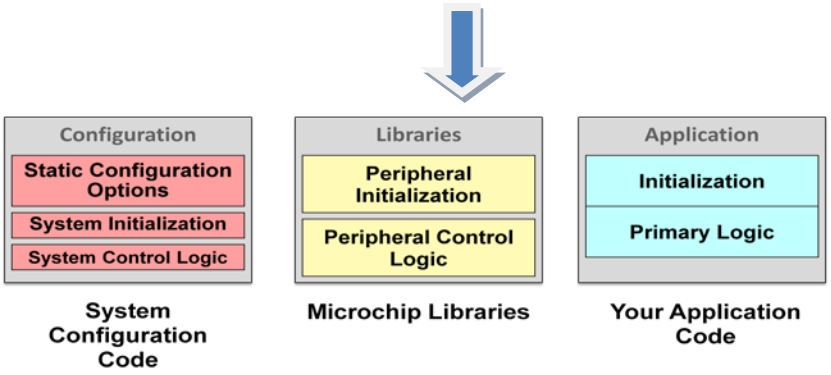
理解 Harmony 下应用示例的架构：

接下来讲一下 Harmony 框架下应用的结构。我们说，Harmony 下的应用程序是经过重构的。重构过程如下：









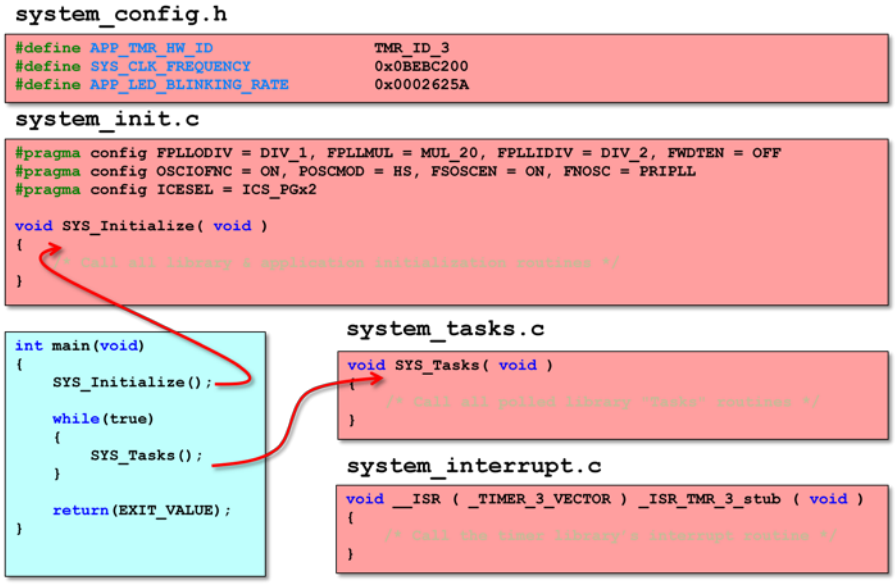
之所以 Harmony 下的应用要满足以上结构划分，主要是因为重构后的应用程序，软件都是模块化，易于软件功能的添加和删除。系统代码和软件库是 Microchip 来实现的，用户只需要编写应用代码即 app.c。

一个应用软件主要有这 4 部分组成：

1. App.c：客户业务相关代码
2. Main.c：Harmony 下应用程序结构
3. System_config 目录下代码：芯片系统相关代码
   a) System_exception.c：异常处理相关
   b) System_init.c：系统配置的外设、系统服务和其它软件建模块在这里进行初始化
   c) System_interrupt.c：系统中所有使能的中断处理函数
   d) System_task.c：各个软件模块所维护的任务

4. Framework 目录下的代码：主要是驱动和系统服务，也包括其他软件库如 tcpip 栈、RTOS 等

重构过程如上图所示，其中系统代码和软件库通过 MHC 来添加、删除和配置。Main 函数结构如下所示：



用户在 app.c 中需要实现两个主要的函数 app_initialize 和 app_task。这两个函数分别被 SYS_Initialize 和 SYS_Tasks 所调用。

总结一下，在 harmony 框架下用户创建自己的应用只需要两步：

1. 在 MHC 中选择系统软件模块，驱动、系统服务、操作系统、TCPIP 协议栈等；
2. 编写应用程序，app_initialize 和 app_task，通过 API 调用系统软件实现功能。

如何利用 Harmony 下已有的应用示例来集成用户自己的应用程序：

了解了 Harmony 下应用软件示例的结构，就可以利用已有的应用示例轻松集成自己的应用程序。

第一步，将多个已有例的底层软件（系统代码和软件库）通过 MHC 集成到一起；

第二步，将多个 app.c 和 app.h，添加到同一个工程里，并修改相应的函数和文件名为

app1.c, app2.c… appn.c

app1.h, app2.h… appn.h

app1_initialize, app1_task… appn_initialize, appn_task

其中，第一步有一个技巧，用户可以安装两个版本的 MPLAB X IDE，一个用来打开已有示例 MHC，查看配置选项；另一个用来编辑新的工程项目。这样用户只要在自己的工程里把示例的配置选项打上勾即可。

下面我们来实现一个例子，展示代码合并的过程。例子是将 usart_basic

(c:\microchip\harmony\v1_06_02\apps\examples\peripheral\usart\uart_basic) 和 msd_basic (c:\microchip\harmony\v1_06_02\apps\usb\host\msd_basic) 两个工程集成起来。使新的应用同时拥有 USART 通信和 USB 存储功能。

环境准备：

软件　　Harmony v1.06.02

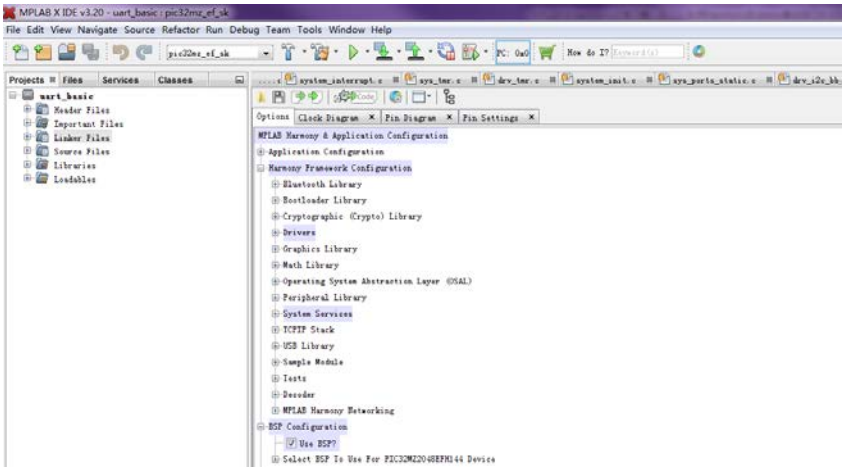
MPLAB X v3.20 和 MPLAB v3.26

硬件

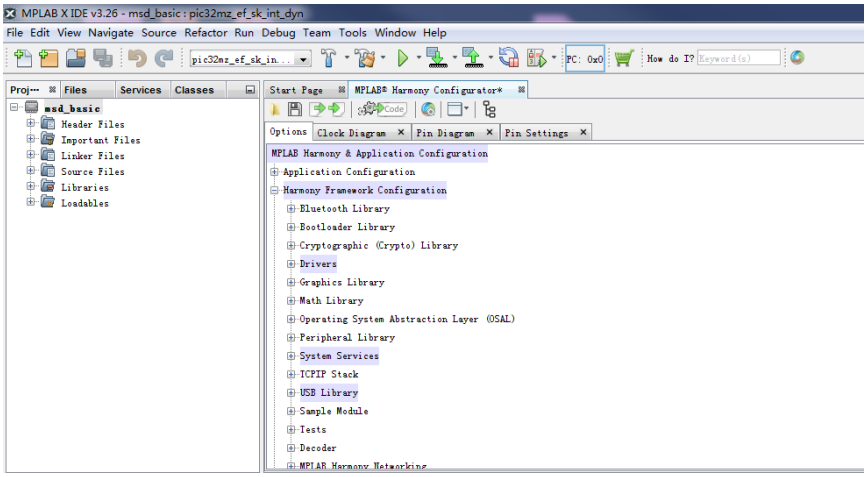
PIC32MZ EF starter kit

集成步骤：

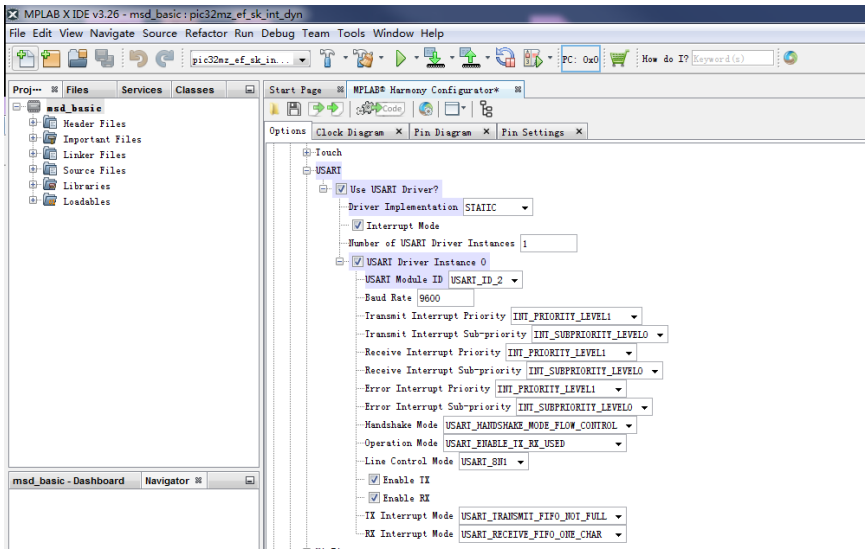
1. 用 MPLAB X v3.20 打开 usart_basic 工程

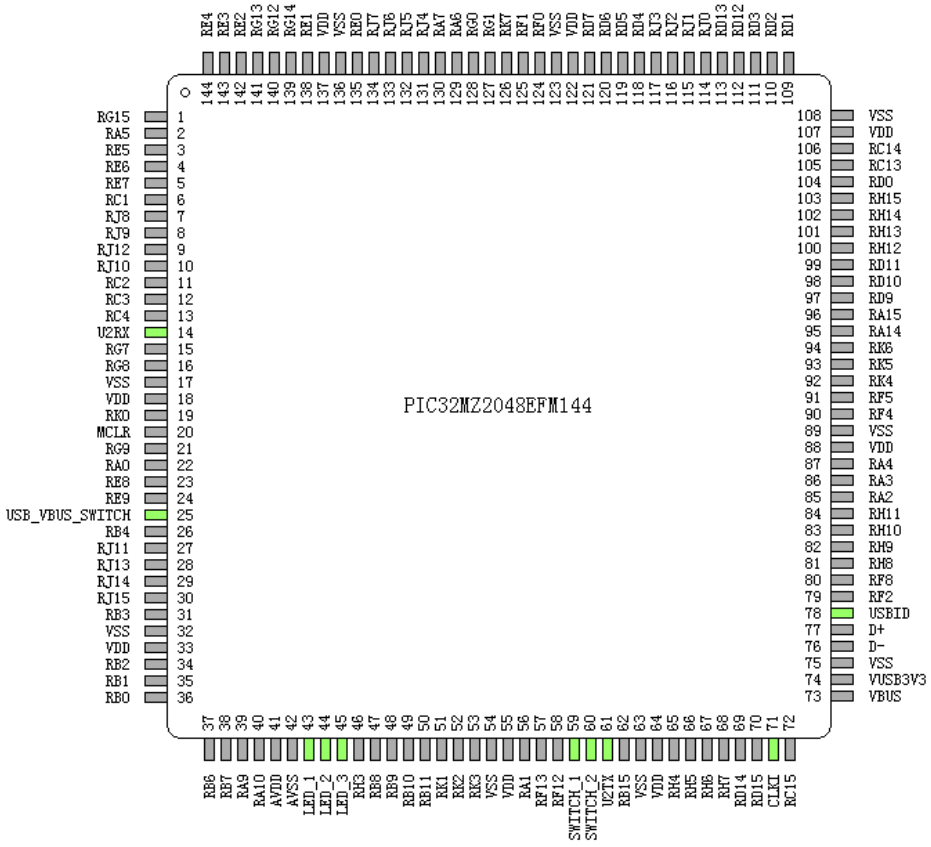


2. 用 MPLAB X v3.26 打开 msd_basic 工程，作为基线工程（即最终的工程）

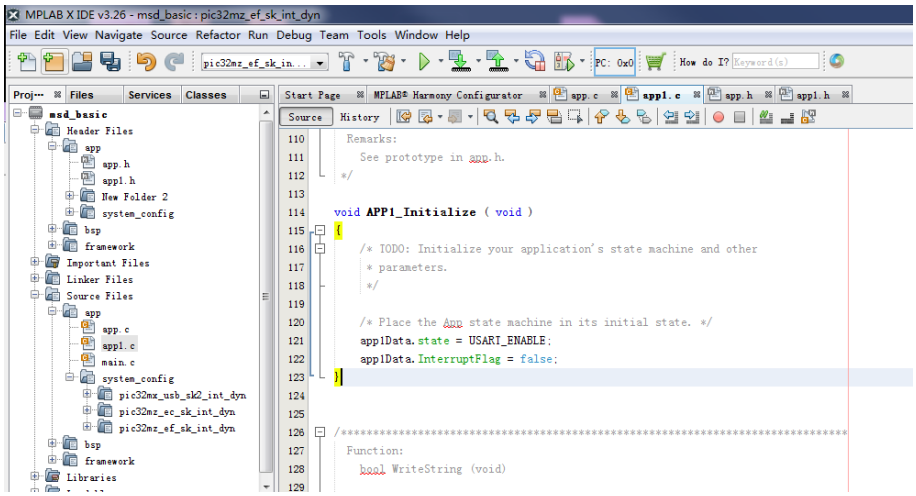


3. 将 usart_basic 工程的配置项添加到 msd_basic 工程中去，包括 USART2 pin 的配置，生成底层驱动





4. 将 usart_basic 工程中的 app.c 和 app.h 改名为 app1.c 和 app1.h，然后添加到 msd_basic 工程中去。并将这两个文件中所有的 app 字段用 app1 代替。如 appData 改为 app1Data

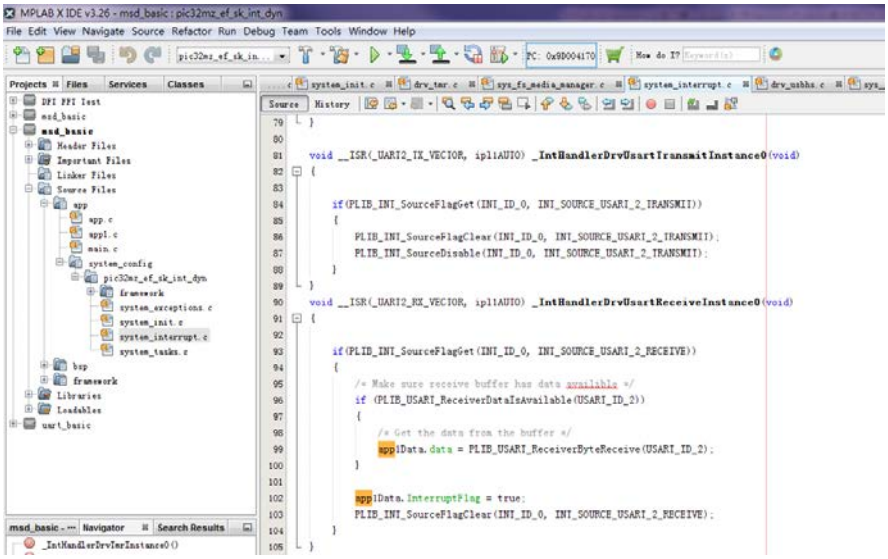


5. 添加 APP1_Task 和 APP1_Initialize 到 SYS_Task 和 SYS_Initialize 函数中去

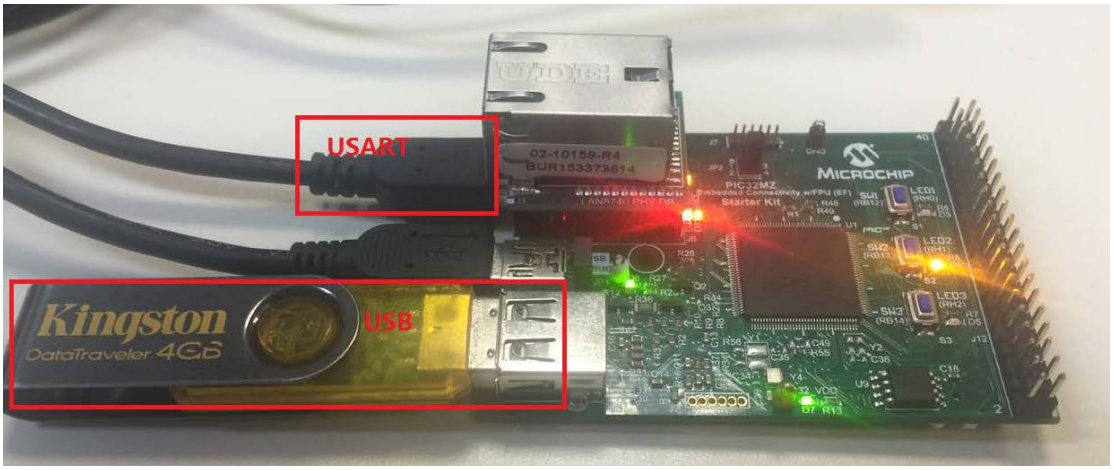
```
void SYS_Tasks ( void )
{
    /* Maintain system services */
    SYS_DEVCON_Tasks(sysObj.sysDevcon);
    SYS_FS_Tasks();
    SYS_TMR_Tasks(sysObj.sysTmr);

    /* Maintain Device Drivers */
    DRV_TMR_Tasks(sysObj.drvTmr0);

    /* Maintain Middleware & Other Libraries */

    /* USBHS Driver Task Routine */
    DRV_USBHS_Tasks(sysObj.drvUSBObject);

    /* USB Host layer task routine.*/
    USB_HOST_Tasks(sysObj.usbHostObject0);

    /* Maintain the application's state machine. */
    APP_Tasks();
    APP1_Tasks();
}
```

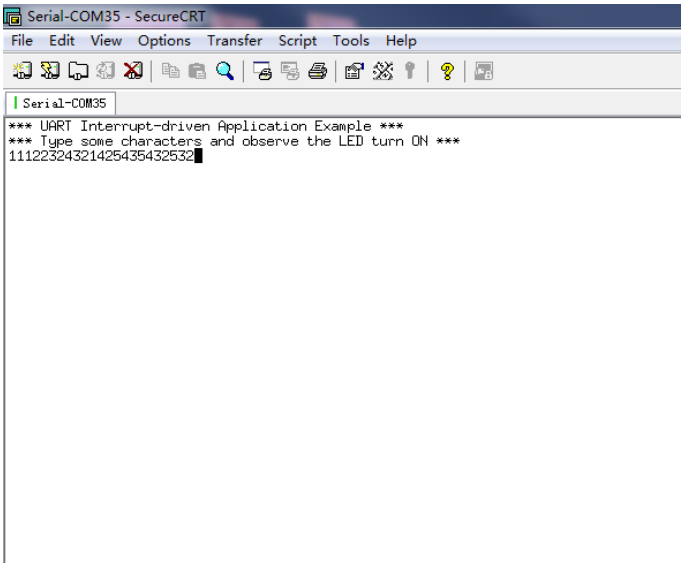
```
void SYS_Initialize ( void* data )
{
    /* Core Processor Initialization */
    SYS_CLK_Initialize( NULL );
    sysObj.sysDevcon = SYS_DEVCON_Initialize(SYS_DEVCON_INDEX_0, (S
    SYS_DEVCON_PerformanceConfig(SYS_CLK_SystemFrequencyGet());
    SYS_PORTS_Initialize();

    /* Board Support Package Initialization */
    BSP_Initialize();

    /* Initialize Drivers */

    sysObj.usbHostObject0 = USB_HOST_Initialize ((SYS_MODULE_INIT
    sysObj.drvUSBObject = DRV_USBHS_Initialize (DRV_USBHS_INDEX_0,

    SYS_INT_VectorPrioritySet(INT_VECTOR_USB1, INT_PRIORITY_LEVEL4)
    SYS_INT_VectorSubprioritySet(INT_VECTOR_USB1, INT_SUBPRIORITY_L

    /* Set the priority of the USB DMA Interrupt */
    SYS_INT_VectorPrioritySet(INT_VECTOR_USB1_DMA, INT_PRIORITY_LEV
    /* Set Sub-priority of the USB DMA Interrupt */
    SYS_INT_VectorSubprioritySet(INT_VECTOR_USB1_DMA, INT_SUBPRIORI

    /* Enable Global Interrupts */
    SYS_INT_Enable();

    /* Initialize the Application */
    APP_Initialize();
    APP1_Initialize();
}
```

6. 把 USART 中断处理函数拷贝到新的工程中，并把 appData 改为 app1Data



7. 编译、下载即可调试。此时新的工程就包含了 USB 存储和 USART 通信功能





注：Microchip 的名称和徽标组合及 MPLAB 均为 Microchip Technology Inc.在美国和其他国家或地区的注册商标。在此提及的所有其他商标均为各持有公司所有。