SmeshLink 美信凌科

Mbed 快速入门

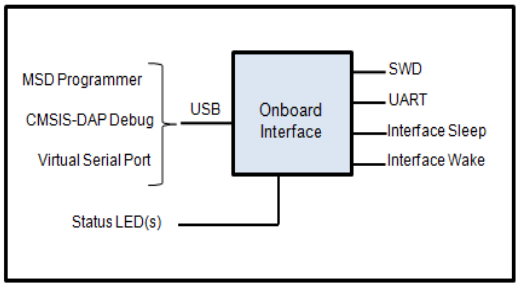
fuqian@smeshlink.com
010-59456917

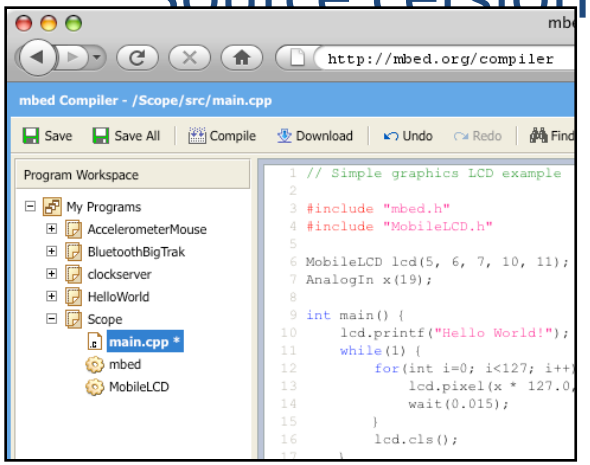
主要内容

- Mbed是什么
- HelloWorld
- 为什么要用Mbed
- eclipse IDE for Mbed(HelloWorld V2)
- Mbed SDK体系结构
- 数字管脚输入输出，中断
- TIMER
- 模拟管脚输入输出，PWM
- UART，SPI，I2C串行通讯，TF卡
- 以太网，802.15.4通讯
- USB设备，USB主机
- RTOS

Mbed是什么

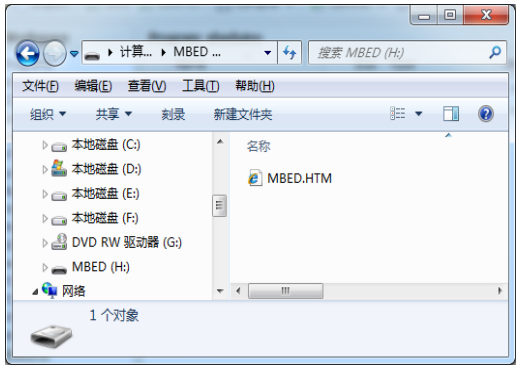
- 完备的电子产品原型快速开发平台
  - SDK(Open Source Lib)
  - HDK(Debug dnd Upload)
  - WEB(Online compile and Source cersion

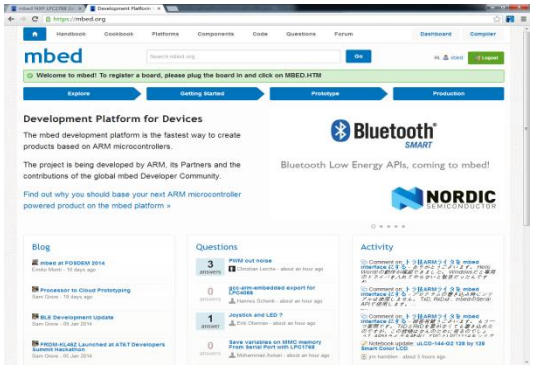




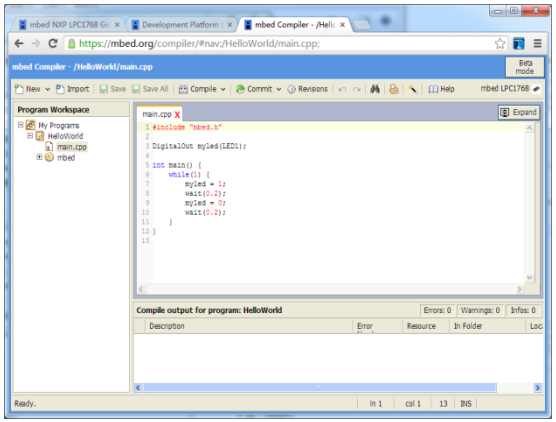
HelloWorld

- 连接mBed LPC1768到电脑
- 注册并登陆到mBed
- 编写HelloWorld
- 编译上载HelloWorld






```
#include "mbed.h"
 DigitalOut myled(LED1);
 int main() {
    while(1) {
        myled = 1;
        wait(0.2);
        myled = 0;
        wait(0.2);
    }
}
```



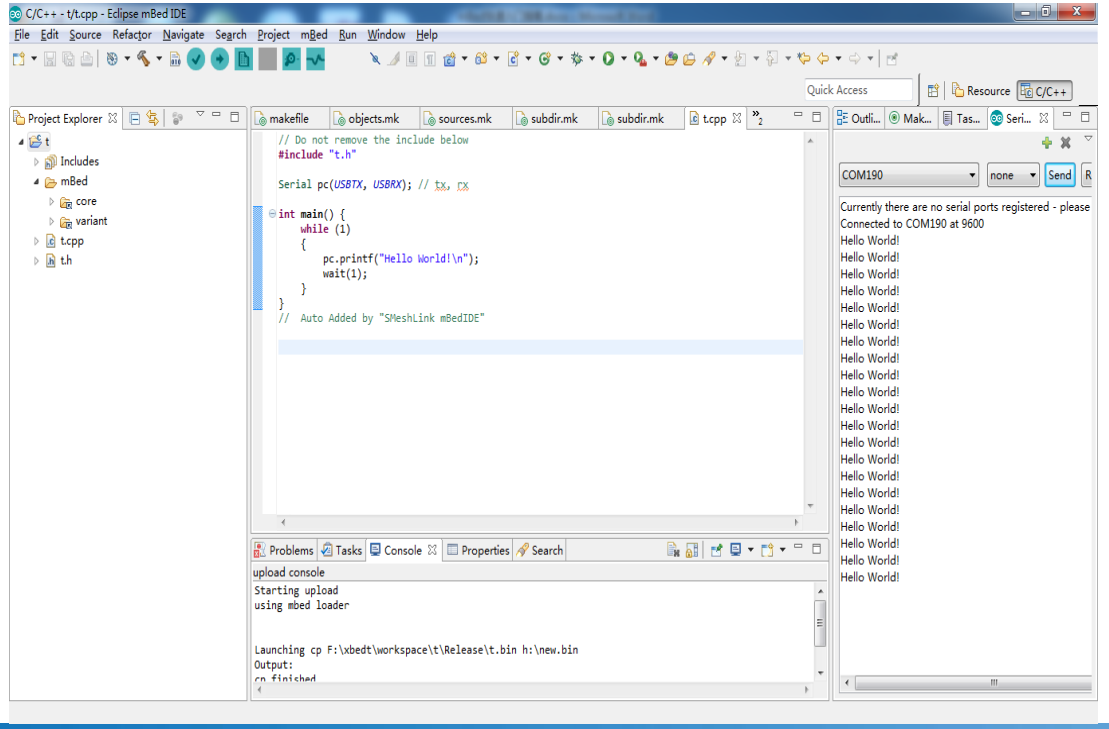
为什么要用Mbed

- 软件开源，Apache License，商业应用无限制
- 硬件成本低
  - M0+ ，<0.3$
  - M3 ，<1$
- 应用广泛
  - NXP,ST,Freescale,Nordic
- 可靠性高，正式版由ARM公司测试
- 使用简单
  - C++
  - 具有硬件抽象层，屏蔽不同MCU之间的差异
- 功能强大
  - 支持丰富外设
  - 自带操作系统，支持分时操作

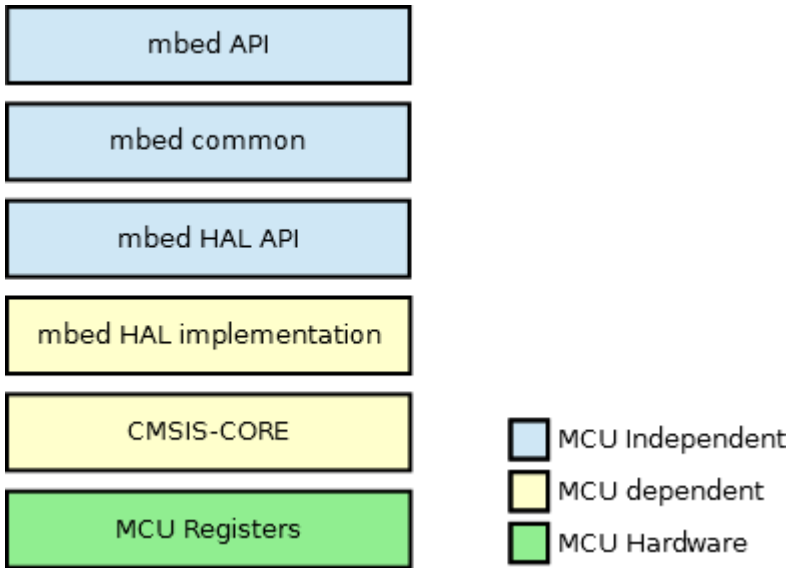
eclipse IDE for

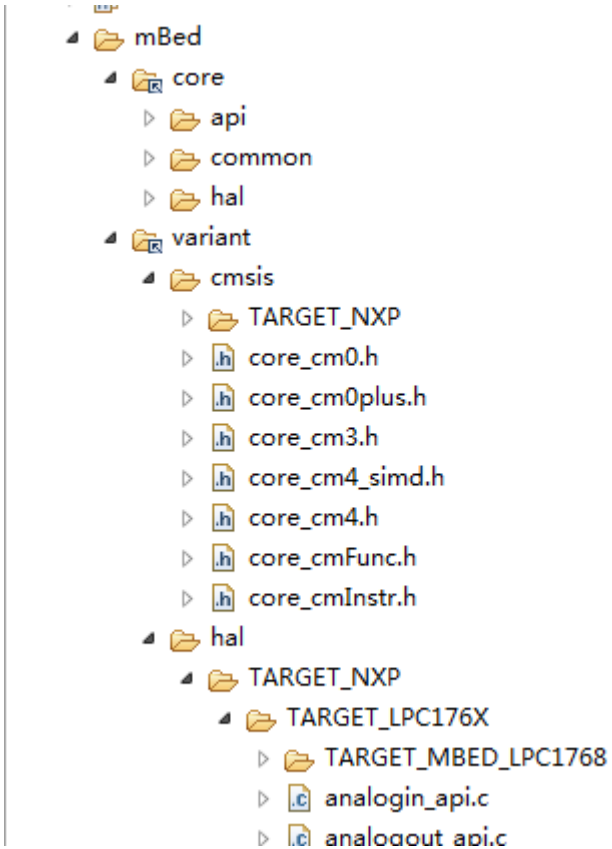
- 基于Arduino for Eclipse Plugin开发
- 完全离线操作，使用方便

```
Serial pc(USBTX, USBRX);
int main() {
while (1)
{
pc.printf("Hello World!\n");
wait(1);
}
}
```



Mbed SDK体系结构





Mbed SDK体系结构

- Device.h 设备功能定义
- PinNames.h 管脚定义文件

```
letine PORT_SHIFT  5

rpedef enum {
  // LPC Pin Names
  P0_0 = LPC_GPIO0_BASE,
       P0_1, P0_2, P0_3, P0_4, P0_5, P0_6, P0_7, P0_8, P0_9, P0_10, P0_11, P0_12, P0
  P1_0, P1_1, P1_2, P1_3, P1_4, P1_5, P1_6, P1_7, P1_8, P1_9, P1_10, P1_11, P1_12, P1
  P2_0, P2_1, P2_2, P2_3, P2_4, P2_5, P2_6, P2_7, P2_8, P2_9, P2_10, P2_11, P2_12, P2
  P3_0, P3_1, P3_2, P3_3, P3_4, P3_5, P3_6, P3_7, P3_8, P3_9, P3_10, P3_11, P3_12, P3
  P4_0, P4_1, P4_2, P4_3, P4_4, P4_5, P4_6, P4_7, P4_8, P4_9, P4_10, P4_11, P4_12, P4

  // mbed DIP Pin Names
  p5 = P0_9,
  p6 = P0_8,
  p7 = P0_7,
  p8 = P0_6,
  p9 = P0_0,
  p10 = P0_1,
  p11 = P0_18,
  p12 = P0_17,
  p13 = P0_15,
  p14 = P0_16,
  p15 = P0_23,
  p16 = P0_24,
  p17 = P0_25,
```

```
/* mbed Microcontroller Library
#ifndef MBED_DEVICE_H
#define MBED_DEVICE_H

#define DEVICE_PORTIN           1
#define DEVICE_PORTOUT          1
#define DEVICE_PORTINOUT        1

#define DEVICE_INTERRUPTIN      1

#define DEVICE_ANALOGIN         1
#define DEVICE_ANALOGOUT        1

#define DEVICE_SERIAL           1
#define DEVICE_SERIAL_FC        1

#define DEVICE_I2C              1
#define DEVICE_I2CSLAVE         1

#define DEVICE_SPI              1
#define DEVICE_SPISLAVE         1

#define DEVICE_CAN              1

#define DEVICE_RTC              1

#define DEVICE_ETHERNET         1

#define DEVICE_PWMOUT           1

#define DEVICE_SEMIHOST         1
#define DEVICE_LOCALFILESYSTEM  1
#define DEVICE_ID_LENGTH       32
#define DEVICE_MAC_OFFSET      20

#define DEVICE_SLEEP            1

#define DEVICE_DEBUG_AWARENESS  1

#define DEVICE_STDIO_MESSAGES   1
```

数字管脚输入输出，中断

- 输入输出

```
#include "mbed.h “

DigitalIn enable(p5);
DigitalOut led(LED1);
DigitalInOut pin(p6);

int main() {
    while(1) {
        if(enable) {
            led = !led;
        }
         pin.output();
         pin = 0;
        wait(0.25);
        pin.input();
          wait(0.25);
    }
}
```

- 中断

```
#include "mbed.h"

InterruptIn button(p5);
DigitalOut led(LED1);
DigitalOut flash(LED4);

void flip() {
    led = !led;
}

int main() {
    button.rise(&flip);
    while(1) {
        flash = !flash;
        wait(0.25);
    }
}
```

TIMER

```
#include "mbed.h"

Timer t; //计时

int main() {
    t.start();
    printf("Hello
     World!\n");
    t.stop();
    printf("The time taken
     was %f seconds\n",
     t.read());
}
```

```
include "mbed.h  “
Timeout flipper; //单次触发
DigitalOut led1(LED1);
DigitalOut led2(LED2);
void flip() {
    led2 = !led2;
}
int main() {
    led2 = 1;
    flipper.attach(&flip, 2.0
    while(1) {
        led1 = !led1;
        wait(0.2);
    }
}
```

TIMER-2

```
void wait(float s);

void wait_ms(int ms);

void wait_us(int us);
```

```
#include "mbed.h"

Ticker flipper; //多次触发
DigitalOut led1(LED1);
DigitalOut led2(LED2);
void flip() {
    led2 = !led2;
}
int main() {
    led2 = 1;
    flipper.attach(&flip, 2.0);
    while(1) {
        led1 = !led1;
        wait(0.2);
    }
}
```

模拟管脚输入输出，PWM

```
#include "mbed.h  “
AnalogIn ain(p19);
AnalogOut aout(p18);
DigitalOut led(LED1);
int main() {
    while (1){
            if(ain > 0.3) {
                    led = 1;
            } else {
                    led = 0;
            }
            for(float i=0.0; i<1.0; i+=0.1) {
                    aout = i;
                    wait(0.0001);
            }
    }
}
```

```
#include "mbed.h"

PwmOut led(LED1);

int main() {
    while(1) {
        for(float p = 0.0f; p
< 1.0f; p += 0.1f) {
            led = p;
            wait(0.1);
        }
    }
}
```

UART串行通讯

- **了解PinMap结构**

```
typedef struct {
    PinName pin; //管脚定义
    int peripheral; //外设类型
    int function; //管脚的第几个功能
} PinMap;
```

- **外设功能都应对应定义**

```
static const PinMap PinMap_UART_TX[] =
    {
  {P0_0,  UART_3, 2},
  {P0_2,  UART_0, 1},
  {P0_10, UART_2, 1},
  {P0_15, UART_1, 1},
  {P0_25, UART_3, 3},
  {P2_0 , UART_1, 2},
  {P2_8 , UART_2, 2},
  {P4_28, UART_3, 3},
  {NC   , NC    , 0}
};
```

```
static const PinMap PinMap_UART_RX[] =
    {
  {P0_1 , UART_3, 2},
  {P0_3 , UART_0, 1},
  {P0_11, UART_2, 1},
  {P0_16, UART_1, 1},
  {P0_26, UART_3, 3},
  {P2_1 , UART_1, 2},
  {P2_9 , UART_2, 2},
  {P4_29, UART_3, 3},
  {NC   , NC    , 0}
};
#define UART_NUM    4
```

- **代码参见helloworld v2**

SPI串行通讯

- SPI Master

```
#include "mbed.h"

SPI spi(p5, p6, p7); // mosi, miso, sclk
DigitalOut cs(p8);

int main() {
    cs = 1;
    spi.format(8,3);
    spi.frequency(1000000);
    cs = 0;
    // Send 0x8f, WHOAMI register
   spi.write(0x8F);
    // receive the contents of the WHOAMI
   int whoami = spi.write(0x00);
   printf("WHOAMI register = 0x%X\n", whoami);
    // Deselect the device
   cs = 1;
}
```

- SPI Slave

```
#include "mbed.h"

SPISlave device(p5, p6, p7,
  p8); // mosi, miso, sclk, ssel

int main() {
   device.reply(0x00);
  // Prime SPI with first reply

   while(1) {
      if(device.receive()) {
         int v = device.read();
         v = (v + 1) % 0x100;
         device.reply(v);
      }
   }
}
```

I2C串行通讯-Master

```
#include "mbed.h"
I2C i2c(p28, p27); // Read temperature from LM75BD

const int addr = 0x90;

int main() {
    char cmd[2];
    while (1) {
        cmd[0] = 0x01;
        cmd[1] = 0x00;
        i2c.write(addr, cmd, 2);

        wait(0.5);

        cmd[0] = 0x00;
        i2c.write(addr, cmd, 1);
        i2c.read(addr, cmd, 2);

        float tmp = (float((cmd[0]<<8)|cmd[1]) / 256.0);
        printf("Temp = %.2f\n", tmp);
    }
}
```

I2C串行通讯-Slave

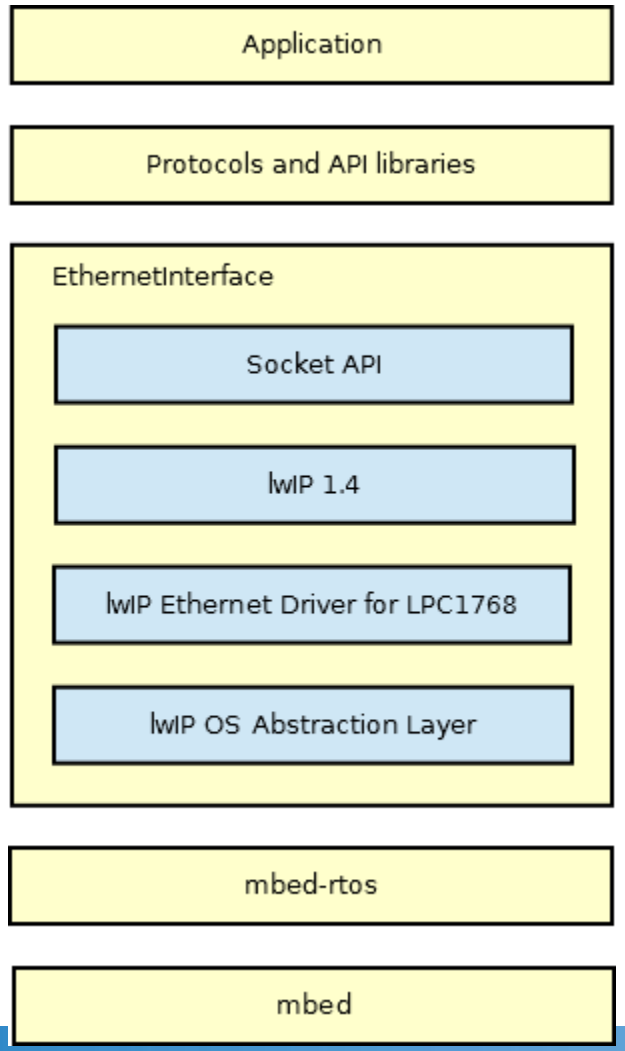
```
#include <mbed.h>
 I2CSlave slave(p9, p10);
 int main() {
    char buf[10];
    char msg[] = "Slave!";
     slave.address(0xA0);
    while (1) {
        int i = slave.receive();
        switch (i) {
            case I2CSlave::ReadAddressed:
                slave.write(msg, strlen(msg) + 1); // Includes null char
                break;
            case I2CSlave::WriteGeneral:
                slave.read(buf, 10);
                    break;
            case I2CSlave::WriteAddressed:
                slave.read(buf, 10);
                break;
        }
        for(int i = 0; i < 10; i++) buf[i] = 0;    // Clear buffer
    }
}
```

TF卡

```
#include "mbed.h"
#include "SDFileSystem.h"

SDFileSystem sd(p5, p6, p7, p8, "sd"); // the pinout on the mbed Cool Components

int main() {
    printf("Hello World!\n");

    mkdir("/sd/mydir", 0777);

    FILE *fp = fopen("/sd/mydir/sdtest.txt", "w");
    if(fp == NULL) {
        error("Could not open file for write\n");
    }
    fprintf(fp, "Hello fun SD Card World!");
    fclose(fp);
}
```

以太网

Protocol	Library	Example Application
HTTP Client	HTTPClient	HTTPClient_HelloWorld
HTTP Client	TinyHTTP_b	denki-yohou_b
NTP Client	NTPClient	NTPClient_HelloWorld
Websocket Client	WebSocketClient	WebSocket_HelloWorld
HTTP Server	RPC over HTTP Server	RPC over HTTP Server
HTTP Server	HTTPServer_echoback	HTTPServer_echoback
SMTP Client	SimpleSMTPClient	SimpleSMTPClient_HelloWorld
Twitter API	HTTPClient_Twitter	HTTPClient_Twitter
IEEE1888(FIAP)	FiapV2	temp_FIAP
Evrythng Api	EvrythngApi	EvrythngApiExample



以太网

```
#include "mbed.h"
#include "EthernetInterface.h"
 int main() {
    EthernetInterface eth;
    eth.init(); //Use DHCP
    eth.connect();
    TCPSocketConnection sock;
    sock.connect("mbed.org", 80);
    char http_cmd[] = "GET /media/uploads/mbed_official/hello.txt HTTP/1.0\n\n";
    sock.send_all(http_cmd, sizeof(http_cmd)-1);
    char buffer[300];
    int ret;
    while (true) {
        ret = sock.receive(buffer, sizeof(buffer)-1);
        if (ret <= 0)
            break;
        buffer[ret] = '\0';
    }
    sock.close();
    eth.disconnect();
}
```

802.15.4通讯

```
#include "mbed.h"
#include "MxRadio.h"

cMxRadio MxRadio(P0_18, P0_17, P0_15, P0_20, P2_11,P2_12, P2_13);
uint8_t i;

int main() {

    MxRadio.begin(0);
    while(1) {
        MxRadio.beginTransmission();
        MxRadio.write("Hello World!");
        MxRadio.write(i);
        i++;
        MxRadio.endTransmission();
        wait_ms(1000);
    }
}
```

USB Device

- **USBMouse** - Emulate a USB Mouse with absolute or relative positioning
- **USBKeyboard** - Emulate a USB Keyboard, sending normal and media control keys
- **USBMouseKeyboard** - Emulate a USB Keyboard and a USB mouse with absolute or relative positionning
- **USBHID** - Communicate over a raw USBHID interface, great for driverless communication with a custom PC program
- **USBMIDI** - Send and recieve MIDI messages to control and be controlled by PC music sequencers etc
- **USBSerial** - Create a virtual serial port over the USB port. Great to easily communicate with a computer.
- **USBAudio** - Create a USBAudio device able to receive audio stream from a computer over USB.
- **USBMSD** - Generic class which implements the Mass Storage Device protocol in order to access all kinds of block storage chips

USB Serial

```
#include "mbed.h"
#include "USBSerial.h"

//Virtual serial port over USB
USBSerial serial;

int main(void) {

    while(1)
    {
        serial.printf("I am a virtual serial port\r\n");
        wait(1);
    }
}
```

USB Host

- **USBHostMouse** - Receive events from a USB mouse
- **USBHostKeyboard** - Read keycode-modifier from a USB keyboard
- **USBHostMSD** - Read-write a USB flash disk
- **USBHostSerial** - Communicate with a virtual serial port
- **USBHostHub** - You can plug several USB devices to an mbed using a USB hub

USBHostMSD

```
USBHostMSD msd("usb");
int i = 0;
while(1) {
    while(!msd.connect()) {
        Thread::wait(500);
    }
    while(1) {
        FILE * fp = fopen("/usb/test1.txt", "a");
        if (fp != NULL) {
            fprintf(fp, "Hello fun SD Card World: %d!\r\n", i++);
            printf("Goodbye World!\r\n");
            fclose(fp);
        } else {
            printf("FILE == NULL\r\n");
        }
        Thread::wait(500);
        if (!msd.connected())
            break;
    }
}
```

RTOS

```
#include "mbed.h"
#include "rtos.h"
 DigitalOut led1(LED1);
DigitalOut led2(LED2);
 void led2_thread(void const *args) {
    while (true) {
        led2 = !led2;
        Thread::wait(1000);
    }
}
 int main() {
    Thread thread(led2_thread);
       while (true) {
        led1 = !led1;
        Thread::wait(500);
    }
}
```

- Thread
- Mutex
- Semaphore
- Signals
- Queue
- MemoryPool
- Mail
- RTOS Timer
- Interrupt Service Routines
- Default Timeouts
- Status and Error Codes
- osEvent
- Implementation

Thank You!