mbed NXP LPC11U24 Hello World!

Agenda

- Introduction to mbed
- mbed registration and Hello World
- Lab 1: Basic IO
- Lab 2: Interrupts and timing
- Lab 3: Local File system and file handling
- Lab 4: Saving Energy with Sleep and Deepsleep
- Lab 5: USB Device

Resources and electronic copies of these slides available from:

http://mbed.org/users/chris/notebook/mbed-nxp-lpc11u24-hello-world/
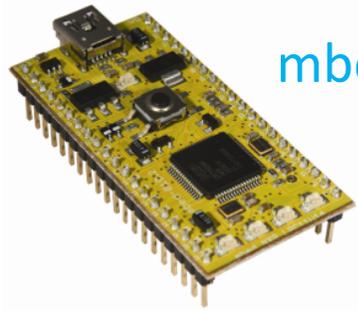
mbed NXP LPC11U24
Hello World!

Introduction to mbed

What's happening in Microcontrollers?

- Microcontrollers are getting cheap
  - 32-bit ARM Cortex-M3 Microcontrollers @ $1

- Microcontrollers are getting powerful
  - Lots of processing, memory, I/O in one package

- Microcontrollers are getting interactive
  - Internet connectivity, new sensors and actuators

- Creates new opportunities for microcontrollers

Rapid Prototyping

- Rapid Prototyping helps industries create new products
  - Control, communication and interaction increasingly define products
  - Development cycles for microelectronics have not kept pace

3D Moulding   **3D Printing**   **2D/3D Design**   **Web Frameworks**

http://mbed.org | Rapid Prototyping for Microcontrollers

---

mbed Rapid Prototyping Platform

- Complete Hardware, Software and Web 2.0 Solution
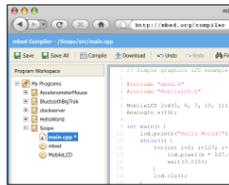
Dedicated Developer Website   **Lightweight Online Compiler**
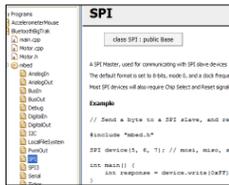
High-level Peripheral APIs   **Prototyping Form-Factor**

http://mbed.org | Rapid Prototyping for Microcontrollers

mbed NXP LPC11U24 Features

- Based on the NXP LPC11U24 MCU
  - 50 MHz 32-bit ARM Cortex-M0, 8KB SRAM, 32 KB Flash
  - Full Speed USB 2.0 device controller
  - SPI, I2C, UART, ADC, GPIO
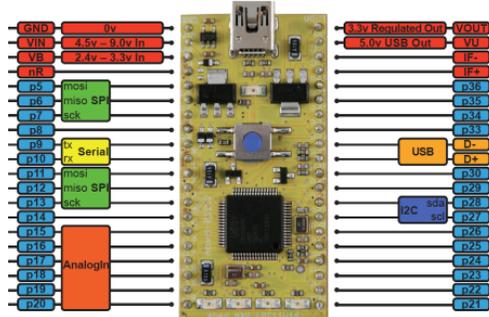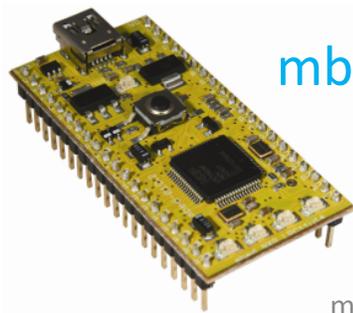
mbed NXP LPC11U24 Hello World!

mbed registration and hello world!

Registration

- mbed microcontroller enumerates as a USB disk

- Double-click the mbed.htm file on the mbed USB disk

- Log in or sign up for a new account
- The mbed microcontroller contains your license to the compiler
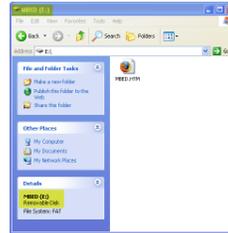
Enable beta mode!

- While the mbed NXP LPC11U24 is in beta trials, you will need to enable "betamode"

- Follow this link :
    http://mbed.org/betamode

- Click here
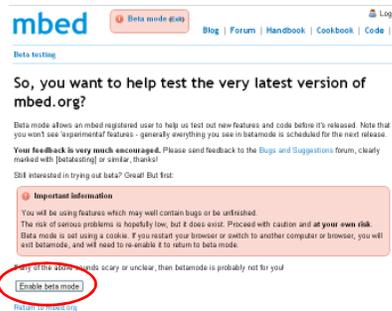
Getting Started

- Useful resources linked from the first page, including very clear links to "Hello World" and the Getting Started guide

- Compiler linked from front page
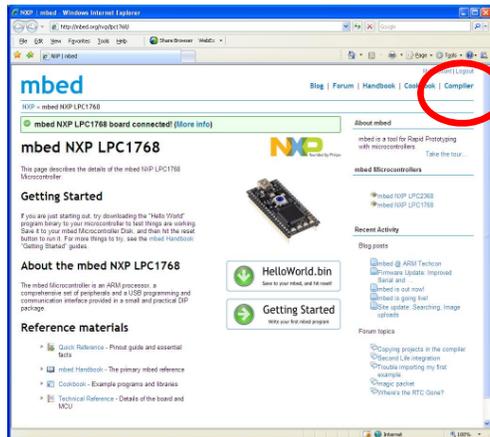
Getting Started

- While mbed NXP LPC11U24 is in beta, do not use the "new" button, instead import the default project :
  - http://mbed.org/users/chris/programs/m0-helloworld/latest
- Develop code in the text editor
- Save and compile
- Compiler outputs
  - Errors and warnings
  - -or-
  - A downloadable binary
- Save to the USB flash disk
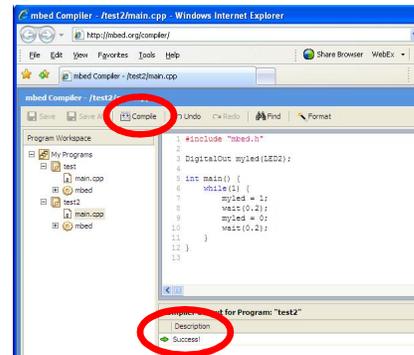
Getting Started

- Once the file has saved to the flash disk, it needs to be programmed into the microcontroller
- Press the button on the mbed module
- Your code will start running!
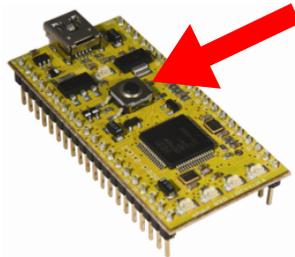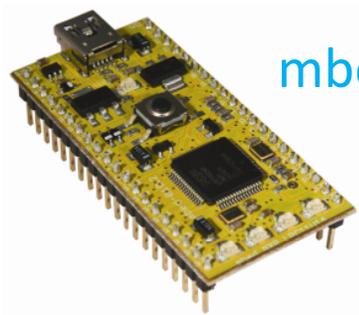
mbed NXP LPC11U24
Hello World!

Lab 1
Basic IO

DigitalOut and Analog Input

- In the hello world session, we simply compiled the Hello World program, but we didn't take too much notice of the code
- It was simple, it set up a digital output (DigitalOut) called "myled" and run a loop forever turning it on and off.
- Lets see if we can expand on this, and use other interfaces.

What IO is there?

- Take another look at your compiler window. In your default project there the mbed library with a "+" box. Try expanding this, and exploring the libraries.
- Note that these are libraries that relate to the microcontroller on chip hardware.
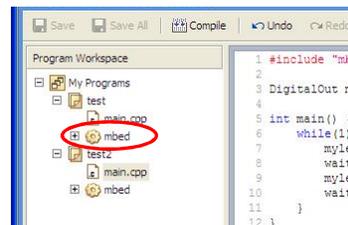
- We'll be using the AnalogIn and BusOut objects, so take time to have a look at their APIs

DigitalOut and Analog Input

- The AnalogIn object returns a normalised float between 0.0 (0.0v) and 1.0 (3.3v)
- The BusOut object groups multiple DigitalOuts, and enables them to be updated simultaneously

---
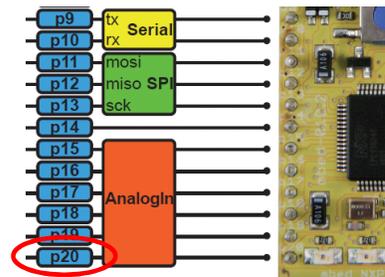
Challenge: BusOut and AnalogIn

- Write a program to display the voltage on an AnalogIn on the four LEDs
  - http://mbed.org/users/chris/programs/m0-AnalogIn/latest

- A normalised float is returned by the read function, and is used to scale the full scale value of the 4 bit bus.

```
#include "mbed.h"

BusOut leds(LED1,LED2,LED3,LED4);
AnalogIn ain(p20);

int main () {
    while (1) {
        leds = 0xF * ain.read();
    }
}
```

9
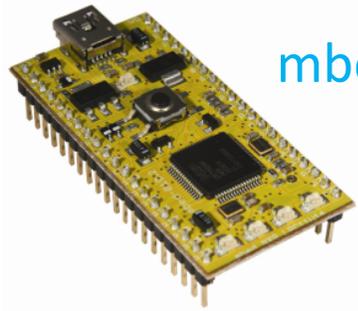
mbed NXP LPC11U24
Hello World!

Lab 2
Interrupts and Timing

Interrupts and timing

- As a microcontroller application grows in complexity, the use of interrupts often becomes inevitable

- This is most commonly for scheduling software tasks, or for external hardware to request attention

- These example shows how timing and hardware interrupt might be used concurrently in a program.

- Open the mbed library tree in the online compiler and familiarise yourself with "InterruptIn" and "Ticker"

Using wait() and the Ticker object

- Write a program to toggle LED2 from a ticker
  - http://mbed.org/users/chris/programs/m0-Ticker/latest

- As the ticker handler executes very fast, there is no impact on the loop that toggles led1

```
#include "mbed.h"

DigitalOut led1(LED1);
DigitalOut led2(LED2);
Ticker tick;

void tick_handler () {
    led2 = !led2;
}

int main () {

    tick.attach(&tick_handler,0.3);

    while (1) {
        led1 = !led1;
        wait(0.2);
    }
}
```

Using InterruptIn

- Write a program to toggle LED3 on each rising edge of pin 14
  - http://mbed.org/users/chris/programs/m0-InterruptIn/latest

- Notice the need for switch debouncing may mean it doesn't quite work as reliably as expected

```
#include "mbed.h"

DigitalOut led3(LED3);
InterruptIn din(p14);

void int_handler () {
    led3 = !led3;
}

int main () {

    din.rise(&int_handler);

    while (1) {
        // do nothing
    }

}
```

Challenge: Interrupt, Ticker and more

- As a final challenge, combine the previous two programs, so that LED1, LED2 and LED3 are all toggling, from different sources

- For interest :
  - http://mbed.org/users/chris/programs/m0-TickerInterruptIn/latest
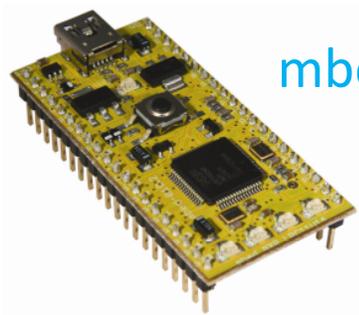
mbed NXP LPC11U24
Hello World!

Lab 3
Local File system and file handling

Example : Data Logging

- Applications often include data logging capabilities, and access to the data often involves bespoke software and interface cables.

- This example shows how standard methods and interfaces can be used to display, save and retrieve data from an application

- For the purposes of the experiment, we will be displaying and logging noise from an unconnected ADC. Touching the pin will influence the noise, it is a demonstration, imagine it is real data!

Example : Data Logging

- The mbed Flash disk is accessible using the LocalFileSystem object
- Standard C file handling techniques apply
- fscanf for reading files
- fprintf for writing files
- This example logs 100 samples to a CSV file

```
#include "mbed.h"

DigitalOut led1(LED1);
DigitalOut led2(LED2);

AnalogIn ain(p20);

LocalFileSystem fs("fs");

int main () {

    FILE *fp = fopen("/fs/data.csv","w");

    for (int i=0; i < 100; i++) {
        fprintf(fp,"%.2f\n",ain.read());
        wait(0.05);
        led2 = !led2;
    }

    fclose(fp);
    led1=1;
}
```

http://mbed.org/users/chris/programs/m0-Filesystem/latest

Data quickly visible to a PC
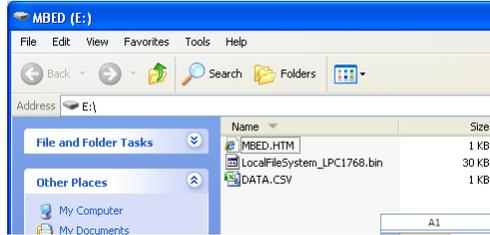
While the program executes the flash drive disappears from the PC, and returns when the file is closed

Logging to a CSV file means Excel can open the file and interpret, manipulate or plot the data.
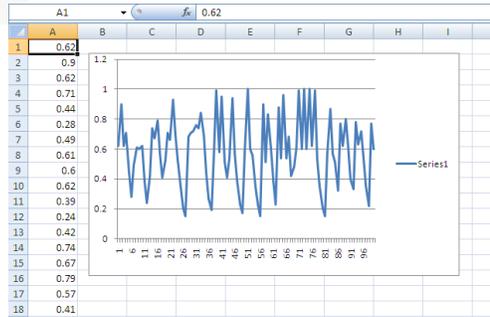
Extend it to store lots of data (Theory)

- Perhaps a final system might want to store lots of data
  - SD cards are ideal, ubiquitous and recognisable by everyone
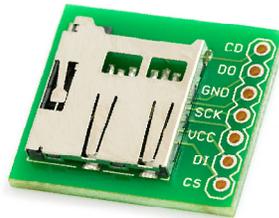

```
GND
MISO – p6
SCL  – p7
Vcc
MOSI – p5
nCS  – p8
```


- Hardware for an SD Card is minimal
  - SPI Port connection using simple breakout
- As before, mbed keeps it simple
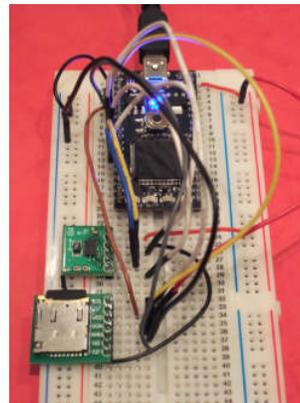
Extend it to store lots of data (Theory)

- An SDFileSystem Library has been published in the community

- Reusing the library is simple
  - Import the SDFile system library
  - Include the SDFileSystem header
  - Swap LocalFileSystem for SDFileSystem
  - Everything else remains the same

- Using standard C file handling techniques keeps it simple

```
1  #include "mbed.h"
2  #include "SDFileSystem.h"
3
4  AnalogIn ain(p20);
5  DigitalOut led(LED1);
6
7  SDFileSystem local(p5, p6, p7, p8, "fs");
8
9  int main() {
10
11     FILE *fp = fopen("/fs/data.csv","w");
12
13     for (int i = 0; i < 100; i++) {
14         fprintf(fp,"%.2f\n", ain.read());
15         wait(0.05);
16     }
17
18     fclose(fp);
19
20     led=1;
21 }
```

---
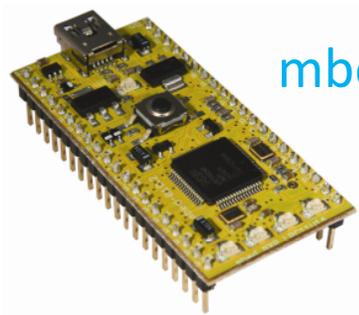
mbed NXP LPC11U24
Hello World!

Lab 4
Saving Energy with Sleep and Deepsleep

LPC11U24 Low Power modes

- The LPC11U24 has four power saving modes; sleep, deep sleep, power down and deep power down
- The sleep modes retain internal and IO state, and so can simply wake up and resume
- The power down modes offer greater savings, but the loss of state makes them a little more complex to use
- For prototyping, sleep modes are a good compromise

Using sleep()

- This program uses sleep()
- When powering from VB, the sleep current is 10mA
- This compares with 50mA for the whole mbed
- Peripheral interrupts can be used to exit sleep mode

- This example uses a ticker to regularly wake up

```
#include "mbed.h"

BusOut leds(LED1,LED2,LED3,LED4);

Ticker wakeup;

void dostuff() {
    for (int i=0; i<5; i++) {
        leds = 1 << i;
        wait(0.25);
    }
}

int main () {

    wakeup.attach(NULL, 3.0);

    while (1) {
        dostuff();
        sleep();
    }

}
```

http://mbed.org/users/chris/programs/m0-sleep/latest

Using deepsleep()

- This program uses deepsleep()
- When powering from VB, the deep sleep current is 1mA
- This compares with 50mA for the whole mbed
- An external interrupt is required to exit deep sleep

- An rising edge InterruptIn is used in this example

```
#include "mbed.h"

BusOut leds(LED1,LED2,LED3,LED4);

InterruptIn wakeup(p14);

void dostuff() {
  for (int i=0; i<5; i++) {
    leds = 1 << i;
    wait(0.25);
  }
}

int main () {

  wakeup.rise(NULL);

  while(1) {
    dostuff();
    deepsleep();
  }

}
```

http://mbed.org/users/chris/programs/m0-deepsleep/latest

http://mbed.org | Rapid Prototyping for Microcontrollers

---
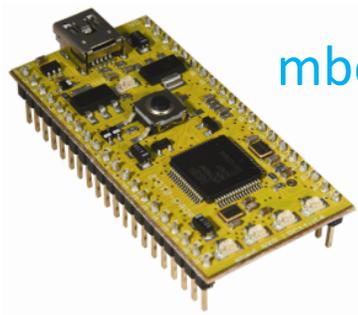
mbed NXP LPC11U24 Hello World!

Lab 5
USB Device

http://mbed.org | Rapid Prototyping for Microcontrollers

LPC11U24 USB Device

- The mbed NXP LPC11U24 has a USB device interface, enabling HID (Keyboard, mouse), Mass Storage, Audio,

- All required filters and circuits are present, only an external USB connect is needed

- Most operating systems have support for these devices built in, and so generic implementations can be very portable

USB Device – Keyboard

- This program implements a USB keyboard
- It is pre-programmed with Credit Card details, which it can automatically type
- The tab ("\t") key is used to move between fields
- The return key ("\n") is used to submit the form
- The program could be used to type a strong password

```
#include "mbed.h"
#include "USBKeyboard.h"

USBKeyboard key;

DigitalIn din(p14);
DigitalOut led1(LED1);

int main(void) {
  while (1) {
    if (din) {
      led1 = !led1;
      key.printf("Mr A N Other\t");
      key.printf("4929780506391234\t");
      key.printf("0611\t");
      key.printf("0513\t");
      key.printf("123\t");
      wait(1);
      key.printf("\n");
      wait(1);
      }
  }
}
```

http://mbed.org/users/chris/programs/m0-USBKeyboard/latest

USB Device – Mouse

- This program implements a USB relative mouse
- The pointer can be located to any x,y position
- Floating point calculations
- Trigonometric functions from the C math library

- Moves the mouse pointer in a circle

```
#include "mbed.h"
#include "USBMouse.h"
#include <math.h>

USBMouse mouse;

int main(void) {

  int16_t x = 0;
  int16_t y = 0;
  int32_t radius = 10;
  int32_t angle = 0;

  while (1) {
    x = cos((double)angle*3.14/180.0)*radius;
    y = sin((double)angle*3.14/180.0)*radius;
    mouse.move(x, y);
    angle += 3;
    wait(0.001);
    }
}
```

http://mbed.org/users/chris/programs/m0-USBMouse/latest

http://mbed.org | Rapid Prototyping for Microcontrollers

---
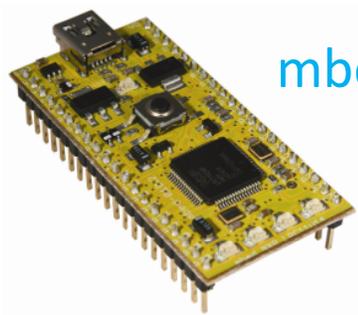
mbed NXP LPC11U24
Hello World!

Summary

http://mbed.org | Rapid Prototyping for Microcontrollers

Summary

- There is huge opportunity for microcontroller applications
  - A major barrier to adoption is simple experimentation

- mbed helps with getting started and rapid prototyping
  - Fast turnaround of experiments and prototyping new ideas
  - Try out new technology and new ideas

- Makes the technology very accessible
  - Demo showed a start to finish prototyping example
  - From getting a user started to enabling an application experiment

- Use at as a tool when you need to experiment!

http://mbed.org | Rapid Prototyping for Microcontrollers

Summary

- A solution focused on prototyping has a broad appeal

- Engineers new to embedded applications
  - Enables experimentation and testing product ideas for the first time
  - Create designs where electronics and MCUs are not the focus

- Experienced engineers
  - Provides a way to be more productive in the proof-of-concept stages
  - Introduce 32 bit microcontroller technology to existing designs

- Marketing, distributors and application engineers
  - Provides a consistent platform for demonstration, evaluation, support
  - Make promotion of MCUs more effective and efficient

http://mbed.org | Rapid Prototyping for Microcontrollers