



ARM mbed

# Agenda

- About ARM
- mbed Developer Ecosystem
- mbed Developer Tools
- mbed Software
- mbed Hardware
- mbed Workshop

# About ARM....

- The World's leading processor design company

**ARM<sup>®</sup> TRUSTZONE<sup>®</sup>**

System Security

**ARM<sup>®</sup> ARTISAN<sup>®</sup>**

Physical IP

**ARM<sup>®</sup> MALI<sup>™</sup>**

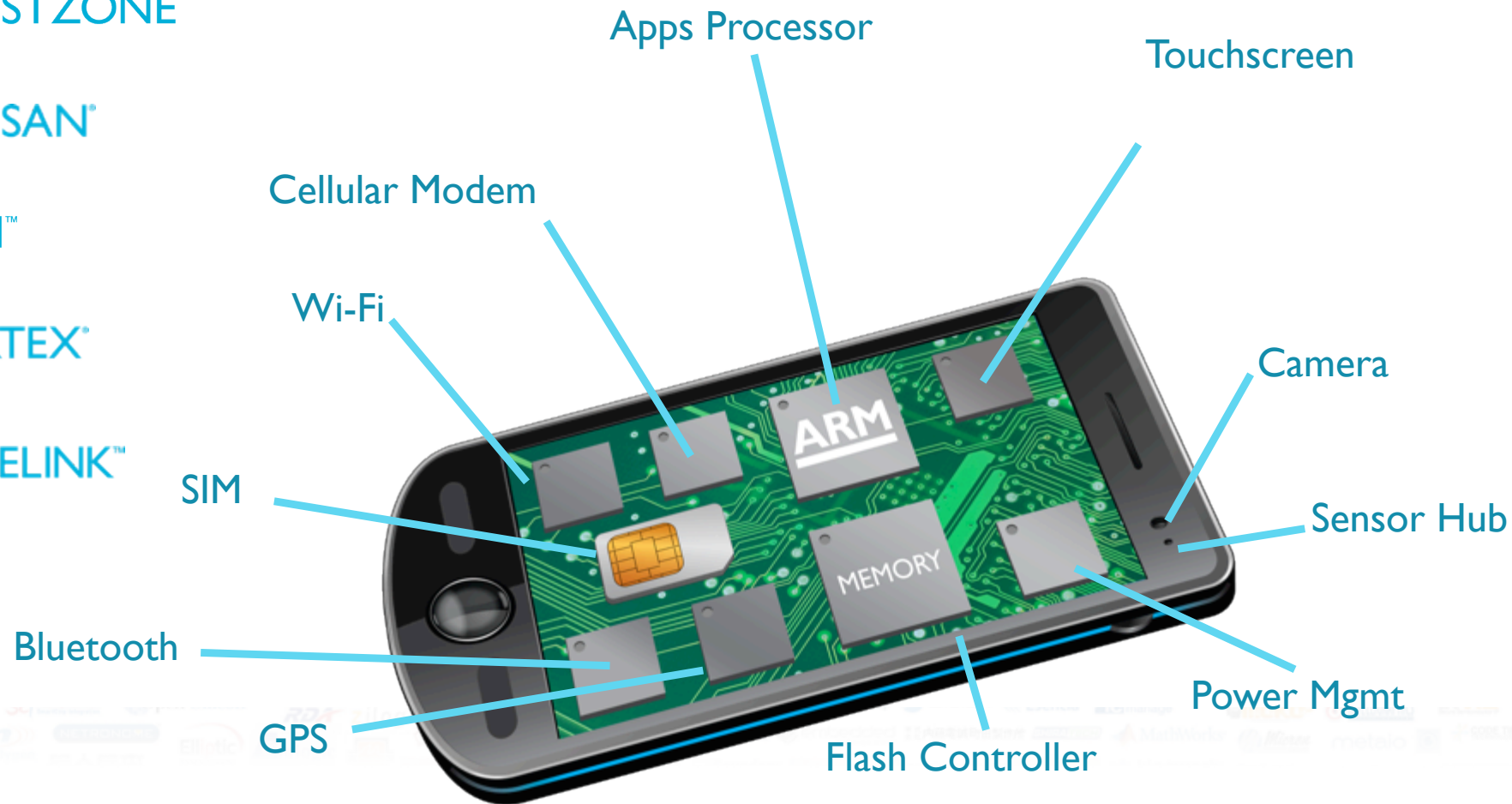
Visual Technology

**ARM<sup>®</sup> CORTEX<sup>®</sup>**

Processor Technology

**ARM<sup>®</sup> CORELINK<sup>™</sup>**

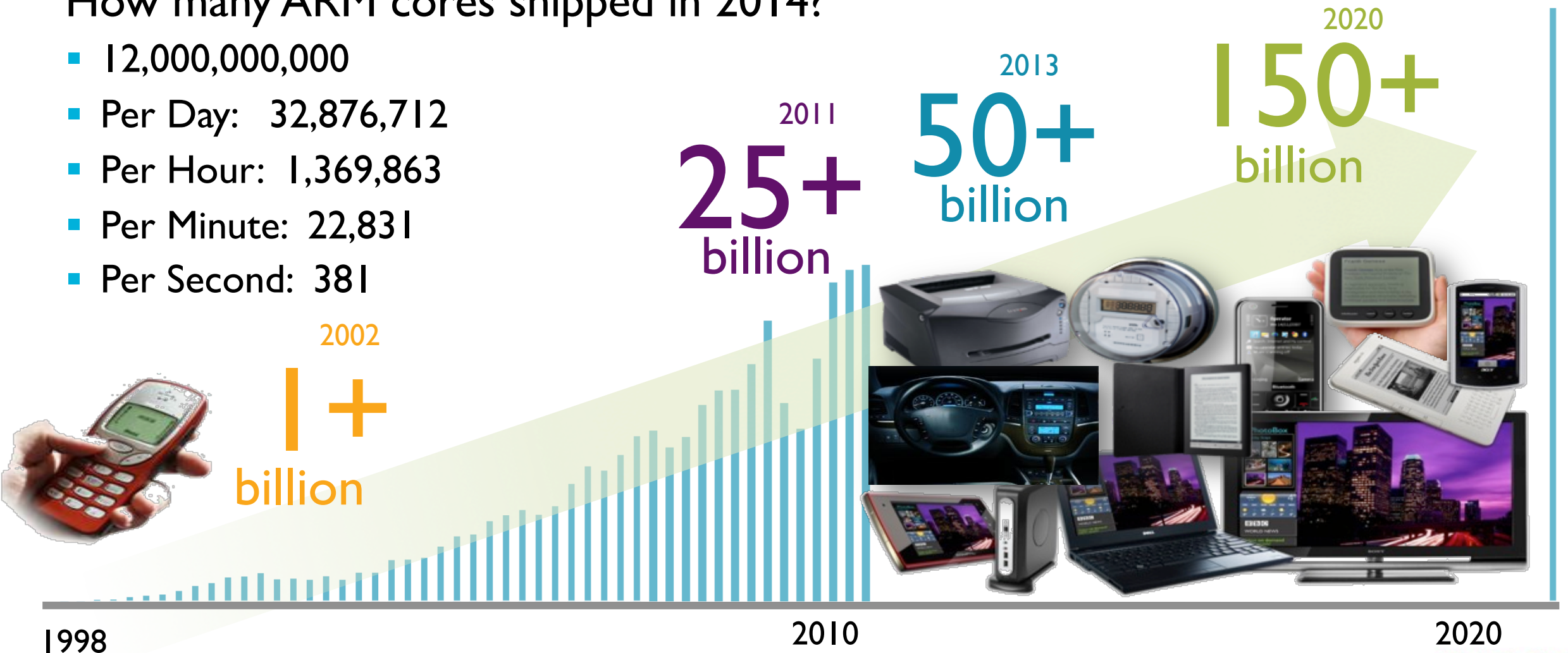
Processor System IP



# Which results in:

How many ARM cores shipped in 2014?

- 12,000,000,000
- Per Day: 32,876,712
- Per Hour: 1,369,863
- Per Minute: 22,831
- Per Second: 381





# ARM mbed Developer Ecosystem

# Next Era of Embedded Development



**Assembler**

**1990s**



**C**

**2000s**

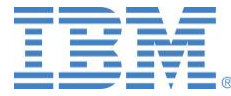


**Platform**

**2010s**



Cloud



Telefonica



MegaChips

130,000+ Developers

mbed OS Services



60+ Boards



Ecosystem



13k+ Published Programs



mbed OS



ESPOTEL



Atmel



CAPTIVA

CSR



Silicon



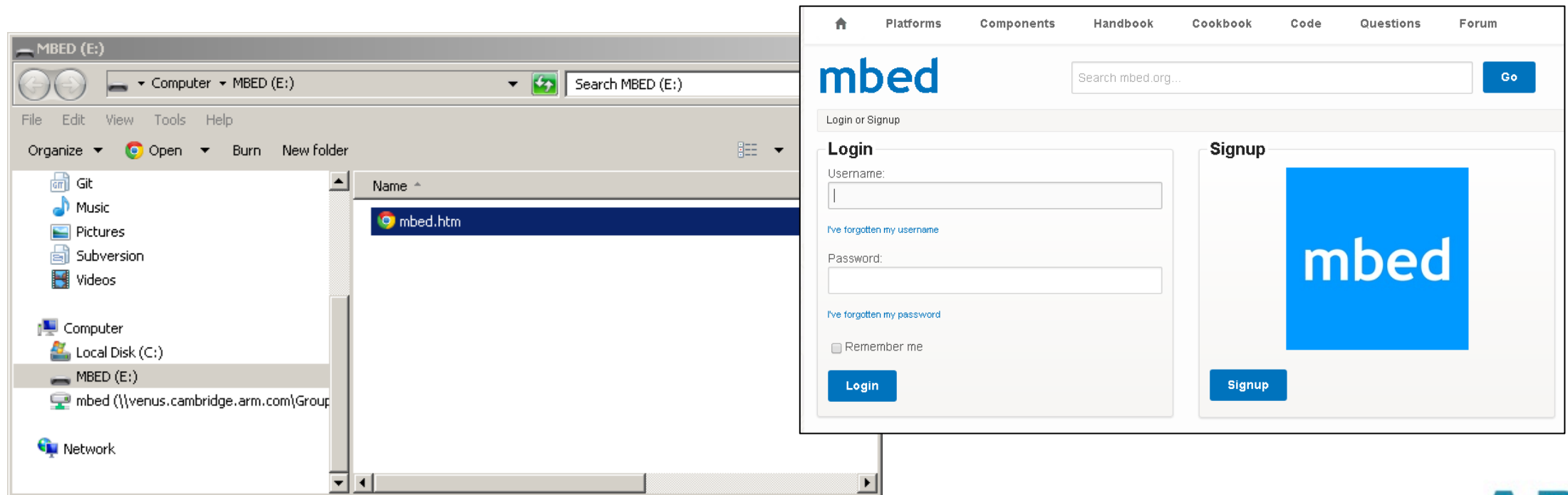
ARM

# ARM mbed Developer Tools

# Create an Account

## ■ Registration

1. Connect a mbed platform to a Windows / Mac / Linux computer
2. mbed platforms is identified as a mass storage device (USB disk)
3. Double-click the mbed.htm file on the mbed USB disk
4. Log in or sign up for a new account



# Know your Hardware

- Connection diagram and example programs on the platform page

Example Program

Connection Diagram

The screenshot shows the mbed LPC11U24 platform page. The page title is "mbed LPC11U24" with a subtitle "Rapid Prototyping for general microcontroller applications, USB and 32-bit ARM® Cortex™-M0 based designs". There are three buttons: "Edit platform", "Edit Features", and "Admin this Platform". A photo of the microcontroller board is shown. Below it is an "Overview" section with a "Table of Contents" box containing links for "Overview", "Features", and "See also". A "Remove from your mbed Compiler" button is visible. At the bottom right, there is an "Example programs" section featuring the "mbed\_blinky" program, updated on 09 May 2014. Two red arrows point from external text labels to specific parts of the page: one from "Example Program" to the "mbed\_blinky" program, and another from "Connection Diagram" to a detailed pinout diagram of the board.

# About Programs

**mbed** / **mbed\_blinky**

The example program for mbed pin-compatible platforms

**Dependencies:** mbed

[Home](#) [History](#) [Graph](#) [API Documentation](#) [Wiki](#) [Pull Requests](#) [Admin settings](#)

You can edit this area! Download repository: [zip](#) [gz](#)

[Edit repository homepage](#)

## Files at revision 6:e8cd76f38fa9

[/ default](#) [tip](#)

Name	Size	Actions
<a href="#">[up]</a>		
<input type="checkbox"/> <a href="#">main.cpp</a>	168	<a href="#">Revisions</a> <a href="#">Annotate</a>
<input type="checkbox"/> <a href="#">mbed.bld</a>	53	<a href="#">Revisions</a> <a href="#">Annotate</a>

[Ask a question](#) [Start a discussion](#)

### Discussion topics

Nothing here yet!

### Questions

Nothing here yet!

### Repository toolbox

- [Import this program](#)
- [Export to desktop IDE](#)
- [Build repository](#)
- [Send Pull Request from here](#)
- [Make featured](#)
- [Following](#)
- Embed url:  
`<<program /teams/mbled/cod`
- Clone repository to desktop:  
`hg clone https://sam_grove@`

### Repository details

Type:	Program
Created:	11 Oct 2013
Imports:	99080
Forks:	24
Commits:	7
Dependents:	0
Dependencies:	1
Followers:	134



# Compiling your Program

The screenshot shows the mbed IDE interface. The top menu bar includes 'New', 'Import', 'Save', 'Save As', 'Compile', 'Commit', 'Revisions', and 'Help'. The 'Compile' button is highlighted with a red box. The main workspace displays the code for 'main.cpp' in a text editor. A 'Save As' dialog box is open, showing the file name 'mbed\_blinky\_LPC11U24.bin' and the save location 'MBED (E:)'. The 'Program Details' panel on the right shows memory usage for Flash (9.9 kB, 31%) and RAM (0.2 kB, 2%). The 'Compile output' panel at the bottom shows a 'Success!' message with a 'Build Details' link highlighted in red.

```
1 #include "mbed.h"
2
3 DigitalOut myled(LED1);
4
5 int main() {
6     while(1) {
7         myled = 1;
8         wait(0.2);
9         myled = 0;
10        wait(0.2);
11    }
12 }
13
```

Usage	Flash	RAM
	9.9 kB (31%)	0.2 kB (2%)

Description	Error Number	Message	In Folder	Location
Success!				

# Online IDE

- Platform Selection

The screenshot shows the mbed online IDE interface. The browser address bar displays `https://developer.mbed.org/compiler/#nav:/test_uart_echo/mbed.bld;`. The IDE title bar shows `mbed` and `/test_uart_echo/mbed.bld`. The main interface includes a menu bar with `New`, `Import`, `Save`, `Save All`, `Compile`, `Commit`, `Revisions`, and `Help`. On the left, the **Program Workspace** shows a tree view with `My Programs`, `test_uart_echo`, `main.cpp`, and `mbed` (containing `Classes`, `Files`, `Structs`, and `Groups`). The center pane displays a table of build components:

Name	Size	Type	Modified
Classes		Classes Documentation	28 Oct 2014
Files		Files Documentation	28 Oct 2014
Groups		Grouped Documentati	28 Oct 2014
Structs		Structs Documentation	28 Oct 2014

Below the table is a search filter with options for `Match Case` and `Whole Word`. The bottom section shows the **Compile output for program: test\_uart\_echo** with a table for errors and warnings. On the right, the **Library Build Details** panel for `mbed` is visible, showing summary information and a warning: **A newer version is available**. The top right corner shows the selected platform `FRDM-K64F`.

- Programs Workspace

- Integrated Version Control

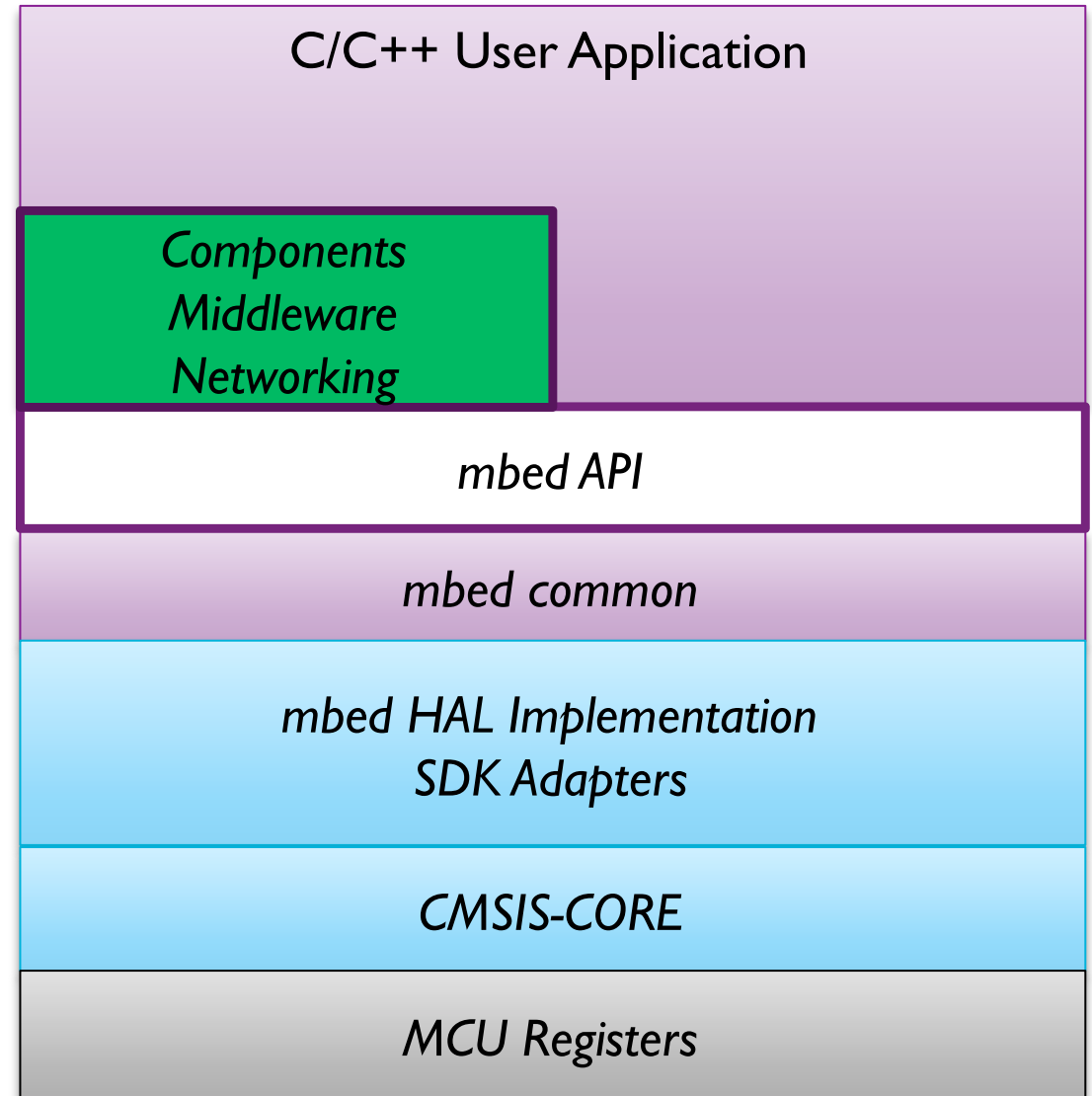
- Program/Library Details

# ARM mbed Software



# mbed SDK Software Stack

- Networking and USB stacks
- CMSIS-RTOS implementation
- Easy-to-use C++ APIs
- stdlib setup, board support, systems configuration
- Hardware Abstraction Layer (HAL) for MCU peripherals
- CMSIS-CORE: hardware register access and Cortex-M startup code



# mbed Program Example

```
1 #include "mbed.h"
2
3 DigitalOut myled(LED1);
4
5 int main() {
6     while(1) {
7         myled = 1;
8         wait(0.2);
9         myled = 0;
10        wait(0.2);
11    }
12 }
```

- Hello World example
  - All startup code is taken care of by the mbed SDK – user code starts at main()
  - Definition of a digital output variable, myled
  - Use of overloaded “=” operator
  - Alter digital output (and LED) by simply assigning a value to the variable.

# http://developer.mbed.org/ handbook

- [mbed Memory Model](#) - The memory model used by the mbed Library

## Analog I/O

- [AnalogIn](#) - Read the voltage applied to an analog input pin
- [AnalogOut](#) - Set the voltage of an analog output pin

## Digital I/O

- [DigitalIn](#) - Configure and control a digital input pin.
- [DigitalOut](#) - Configure and control a digital output pin.
- [DigitalInOut](#) - Bi-directional digital pins
  
- [BusIn](#) - Flexible way to read multiple DigitalIn pins as one value
- [BusOut](#) - Flexible way to write multiple DigitalOut pins as one value
- [BusInOut](#) - Flexible way to read/write multiple DigitalInOut pins as one value
  
- [PortIn](#) - Fast way to read multiple DigitalIn pins as one value
- [PortOut](#) - Fast way to write multiple DigitalOut pins as one value
- [PortInOut](#) - Fast way to read/write multiple DigitalInOut pins as one value
  
- [PwmOut](#) - Pulse-width modulated output
  
- [InterruptIn](#) - Trigger an event when a digital input pin changes.

## Timers

- [Timer](#) - Create, start, stop and read a timer
- [Timeout](#) - Call a function after a specified delay
- [Ticker](#) - Repeatedly call a function
  
- [wait](#) - Wait for a specified time
- [time](#) - Get and set the realtime clock

## Digital Interfaces

- [Serial](#) - Serial/UART bus
  
- [SPI](#) - SPI bus master
- [SPISlave](#) - SPI bus slave
  
- [I2C](#) - I<sup>2</sup>C bus master
- [I2CSlave](#) - I<sup>2</sup>C bus slave
  
- [CAN](#) - Controller-area network bus

## Real-time Operating System

- [mbed RTOS](#)

## File System

- [LocalFileSystem](#) - Using the mbed disk as storage from within a program
- [SDFFileSystem](#) - Using the mbed disk as storage from within a program

## USB

- [USBDevice](#) - Using mbed as a 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 c
  - [USBHID](#) - Communicate over a raw USBHID interface, great for driverless commur
  - [USBMIDI](#) - Send and recieve MIDI messages to control and be controlled by PC m
  - [USBSerial](#) - Create a virtual serial port over the USB port. Great to easily communi
  - [USBAudio](#) - Create a USBAudio device able to receive audio stream from a compu
  - [USBMSD](#) - Generic class which implements the Mass Storage Device protocol in o
  
- [USBHost](#) - Using mbed to act as USBHost
  - [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

# Digital Inputs and Outputs

## mbed - DigitalInOut Class Reference

### Public Member Functions

	<b>DigitalInOut</b> (PinName pin) Create a <b>DigitalInOut</b> connected to the specified pin.
	<b>DigitalInOut</b> (PinName pin, PinDirection direction, PinMode mode, Create a <b>DigitalInOut</b> connected to the specified pin.
void	<b>write</b> (int value) Set the output, specified as 0 or 1 (int)
int	<b>read</b> () Return the output setting, represented as 0 or 1 (int)
void	<b>output</b> () Set as an output.
void	<b>input</b> () Set as an input.
void	<b>mode</b> (PinMode pull) Set the input pin mode.
int	<b>is_connected</b> () Return the output setting, represented as 0 or 1 (int)
<b>DigitalInOut</b> &	<b>operator=</b> (int value) A shorthand for <b>write()</b>
	<b>operator int</b> () A shorthand for <b>read()</b>

### Boolean logic - NOT, AND, OR, XOR

```
1 #include "mbed.h"
2
3 DigitalIn a(p5);
4 DigitalIn b(p6);
5 DigitalOut z_not(LED1);
6 DigitalOut z_and(LED2);
7 DigitalOut z_or(LED3);
8 DigitalOut z_xor(LED4);
9
10 int main() {
11     while(1) {
12         z_not = !a;
13         z_and = a && b;
14         z_or = a || b;
15         z_xor = a ^ b;
16     }
17 }
```



# Ticker

## mbed - Ticker Class Reference

### Public Member Functions

void **attach** (void(\*fptr)(void), float t)  
Attach a function to be called by the **Ticker** , specifying the interval.

template<typename T >  
void **attach** (T \*tptr, void(T::\*mptr)(void), float t)  
Attach a member function to be called by the **Ticker** , specifying the interval.

void **attach\_us** (void(\*fptr)(void), timestamp\_t t)  
Attach a function to be called by the **Ticker** , specifying the interval in microseconds.

template<typename T >  
void **attach\_us** (T \*tptr, void(T::\*mptr)(void), timestamp\_t t)  
Attach a member function to be called by the **Ticker** , specifying the interval in microseconds.

void **detach** ()  
Detach the function.

### Static Public Member Functions

static void **irq** (uint32\_t id)  
The handler registered with the underlying timer interrupt.









































### Protected Attributes

timestamp\_t **\_delay**  
Time delay (in microseconds) for re-setting the multi-shot callback.

**FunctionPointer** **\_function**  
Callback.

```
1 #include "mbed.h"
2
3 // A class for flip()-ing a DigitalOut
4 class Flipper {
5 public:
6     Flipper(PinName pin) : _pin(pin) {
7         _pin = 0;
8     }
9     void flip() {
10         _pin = !_pin;
11     }
12 private:
13     DigitalOut _pin;
14 };
15
16 DigitalOut led1(LED1);
17 Flipper f(LED2);
18 Ticker t;
19
20 int main() {
21     t.attach(&f, &Flipper::flip, 2.0); //
22
23     // spin in a main loop. flipper will
24     while(1) {
25         led1 = !led1;
26         wait(0.2);
27     }
28 }
```

# http://developer.mbed.org/ components

Component	Short Description	Notes	Image	Component	Short Description	Notes	Image	Component	Short Description	Notes	Image
1.3" SHARP Memory LCD	✓	✗		DisplayModule 2.2" TFT with 8-bit interface	✓	✓		Grove - Air Quality Sensor	✓	✓	
128x32 LCD	✓	✓		DisplayModule 2.4" Touch TFT with 8-bit Interface	✓	✓		Grove - Alcohol Sensor	✓	✓	
2.2 QVGA display with SD card socket	✓	✓		DisplayModule 2.8" Touch TFT with 8-bit Interface	✓	✓		Grove - Barometer Sensor	✓	✓	
2.7 inch E-paper display	✓	✗		DisplayModule 2.8" Touch TFT with SPI and 4MB Flash	✓	✓		Grove - Button	✓	✓	
24LCxx Serial EEPROM library	✓	✓		DisplayModule 2.8" Touch TFT with SPI and 4MB Flash	✓	✓		Grove - Buzzer	✓	✓	
25LCxxx SPI	✓	✓		DisplayModule 3.5" Touch TFT with SPI and 4MB Flash	✓	✓		Grove - Collision Sensor	✓	✓	
4D SGC TFT Screen	✓	✓		DMU02 Dynamics Measurement Unit	✓	✓		Grove - Colour Sensor	✓	✗	
4D Systems 128 by 128 Smart Color LCD uLCD-144-G2	✓	✓		DORJI Data Radio Modem (433Mhz)	✓	✓		Grove - Digital Light Sensor	✓	✓	
ACM1602NI-FLW-FBW-M01	✓	✓		DS1302 Timekeeping Chip	✓	✓		Grove - Ear-clip Heart Rate Sensor	✓	✓	
AD8556	✓	✓		DS1307 RTC	✓	✓		Grove - Electricity Sensor	✓	✓	
Adafruit / SSD1306 OLED 128x32 or 128x64	✓	✗		DS1721	✓	✗		Grove - HCHO Sensor	✓	✓	
Adafruit NeoPixels (WS2812)	✓	✓		DS1820	✓	✓		Grove - I2C Touch Sensor	✓	✓	
Adafruit Ultimate GPS Breakout v3	✓	✓		EA DOGS102-6 Graphic LCD	✓	✓		Grove - Moisture Sensor	✓	✓	
								Grove - PIR Motion Sensor	✓	✓	

# mbed Component Database

The screenshot shows the mbed Component Database website. At the top, there are navigation tabs: Home, Platforms, Components (highlighted), Handbook, Cookbook, Code, Questions, and Forum. On the right, there are links for Dashboard and Compiler, and a user profile for sam\_grove with a Logout button. A search bar is located below the navigation tabs, with a 'Go' button. The main content area is titled 'Components' and includes an 'Add a component' button. A sidebar on the left lists various component categories with their respective counts: Actuators (6), Communication (24), Display (39), Expansion boards (16), Internet of Things (7), Online Services (2), and Robotics (7). The main content area features a grid of component categories, each with an image and a label: Actuators, Communication, Display, Expansion boards, HTML WebSockets Internet of Things, Online Services, Robotics, Sensors, Storage, and Other.

Components

**Actuators** (6)  
Motor (3)  
Servomotor (3)  
Solenoid (0)

**Communication** (24)  
Bluetooth (2)  
CAN (1)  
Cellular (4)  
Ethernet (3)  
Infrared (1)  
NFC (1)  
RFID (1)  
Wifi (4)

**Display** (39)  
LCD (16)  
LED Controller (11)  
Touchscreen (5)

**Expansion boards** (16)

**Internet of Things** (7)

**Online Services** (2)

**Robotics** (7)

**Components**

The Component Database hosts reusable libraries for different hardware, middleware and IoT services that you can use with ARM Microcontrollers. These components can be used as building blocks for quickly developing prototypes and products.

Components and the associated libraries, examples and documentation are created and added to the database by mbed developers, component manufacturers and service providers. The goal is to create a canonical database of rock-solid code and resources for every useful component that can be used with ARM microcontrollers.

Actuators

Communication

Display

Expansion boards

HTML WebSockets Internet of Things

Online Services

Robotics

Sensors

Storage

Other

Components are portable across all platforms and tools




# Component Entry

Components » Sensors » FXOS8700Q Accelerometer / Magnetometer

## FXOS8700Q Accelerometer / Magnetometer

This is a 6 axis combination Accelerometer / Magnetometer


**Hello World**

 **Hello\_FXOS8700Q**

Example program for FXOS8700Q sensor

Last commit 4 days ago by [Freescale](#)

**Library**

 **FXOS8700Q**

Driver library for the Freescale FXOS8700Q sensor

Last commit 3 days ago by [Freescale](#)

**Actions:** Delete, Edit this component

**Import program** (highlighted with a red box)

**Import library** (highlighted with a red box)

**Follow this component**

**Tested platforms**

- FRDM-K64F
- Ethernet IoT Starter Kit

**Example program to evaluate the component**

**Directly import into Your current program**

# Public Member Functions

# FXOS8700Q

virtual void	<b>enable</b> (void) const =0 Enable the sensor for operation.
virtual void	<b>disable</b> (void) const =0 disable the sensors operation
virtual uint32_t	<b>sampleRate</b> (uint32_t frequency) con Set the sensor sample rate.
virtual uint32_t	<b>dataReady</b> (void) const =0 Tells of new data is ready.
virtual int16_t	<b>getX</b> (int16_t &x) const =0 Get the x data in counts.
virtual int16_t	<b>getY</b> (int16_t &y) const =0 Get the y data in counts.
virtual int16_t	<b>getZ</b> (int16_t &z) const =0 Get the z data in counts.
virtual float	<b>getX</b> (float &x) const =0 Get the x data in units.
virtual float	<b>getY</b> (float &y) const =0 Get the y data in units.
virtual float	<b>getZ</b> (float &z) const =0 Get the z data in units.
virtual void	<b>getAxis</b> ( <b>motion_data_counts_t</b> &xyz) Get the x,y,z data in counts.
virtual void	<b>getAxis</b> ( <b>motion_data_units_t</b> &xyz) const =0 Get the x,y,z data in units.

```
#include "mbed.h"
#include "FXOS8700Q.h"
I2C i2c(PTE25, PTE24);
FXOS8700QAccelerometer acc(i2c, FXOS8700CQ_SLAVE_ADDR1);
FXOS8700QMagnetometer mag(i2c, FXOS8700CQ_SLAVE_ADDR1);
int main(void)
{
    motion_data_units_t acc_data, mag_data;
    motion_data_counts_t acc_raw, mag_raw;
    float faX, faY, faZ, fmX, fmY, fmZ, tmp_float;
    int16_t raX, raY, raZ, rmX, rmY, rmZ, tmp_int;
    acc.enable();
    mag.enable();
    while (true) {
        // counts based results
        acc.getAxis(acc_raw);
        mag.getAxis(mag_raw);
        acc.getX(raX);
        acc.getY(raY);
        acc.getZ(raZ);
        mag.getX(rmX);
        mag.getY(rmY);
        mag.getZ(rmZ);
        // unit based results
        acc.getAxis(acc_data);
        mag.getAxis(mag_data);
        acc.getX(faX);
        acc.getY(faY);
        acc.getZ(faZ);
        mag.getX(fmX);
        mag.getY(fmY);
        mag.getZ(fmZ);
        wait(0.1f);
    }
}
```



# ARM mbed-enabled Hardware





Platforms

Components

Handbook

Cookbook

Code

Questions

Forum

# ARM<sup>®</sup>mbed<sup>™</sup>

Go

Platforms

## Platforms

freescale



FRDM-KL25Z

- Cortex-M0+
- 128KB Flash, 16KB RAM
- USB OTG



freescale



FRDM-KL46Z

- Cortex-M0+, 48MHz
- 256KB Flash, 32KB RAM
- USB OTG



freescale



FRDM-K64F

- Cortex-M4, 120MHz
- 1MB Flash, 256KB RAM
- Ethernet, SD Filesystem



freescale



Ethernet IoT Starter Kit

- Freescale K64F Processor
- mbed application shield
- IBM IoT Client pre-loaded

freescale



FRDM-KL05Z

- Cortex-M0+, 48MHz
- 32KB Flash, 4KB RAM

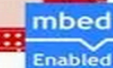


freescale



FRDM-K20D50M

Cortex-M4, 48MHz



freescale



FRDM-K22F

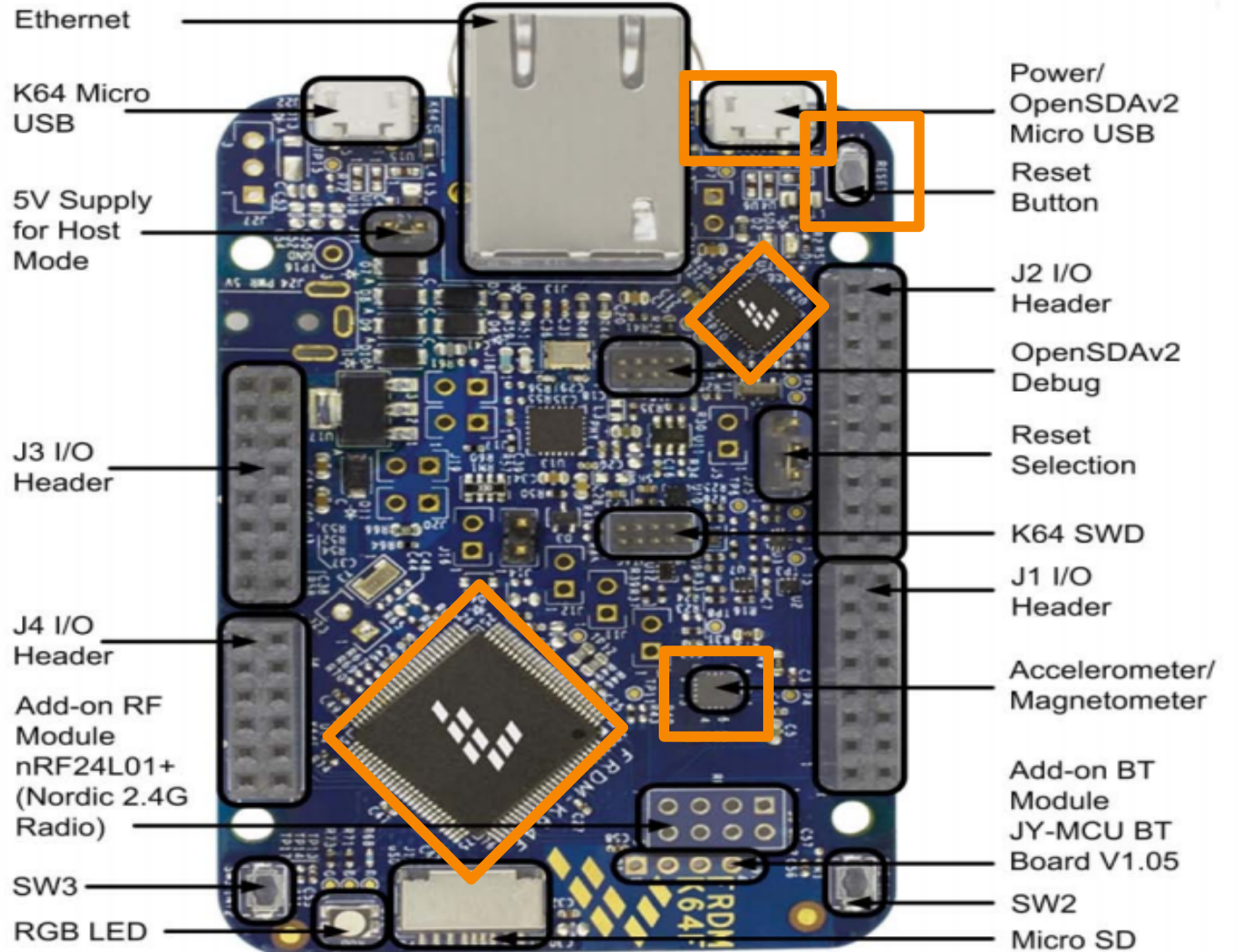
- Cortex-M4, 120MHz
- 512KB Flash, 128KB RAM



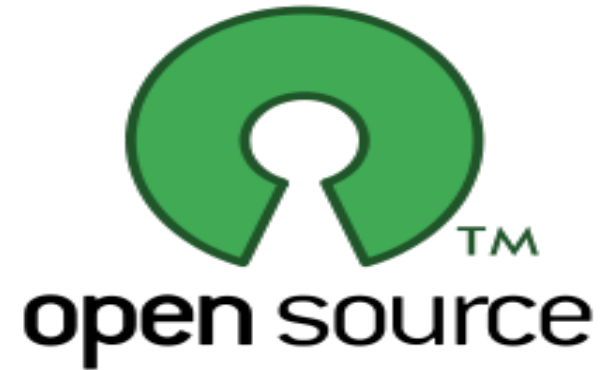
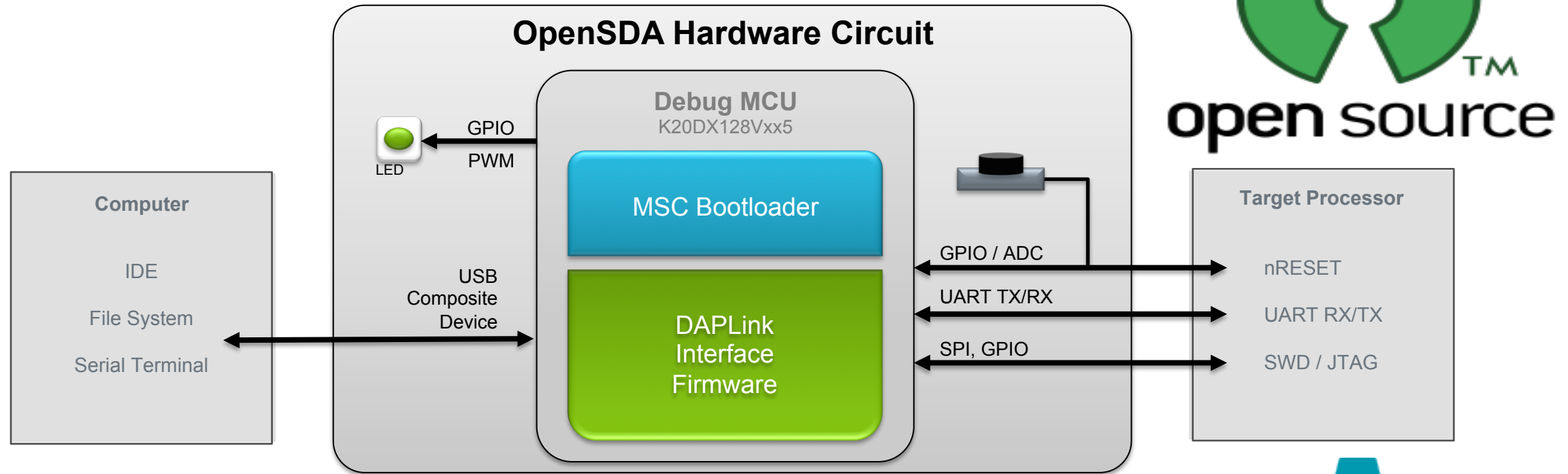


# FRDM-K64F Overview

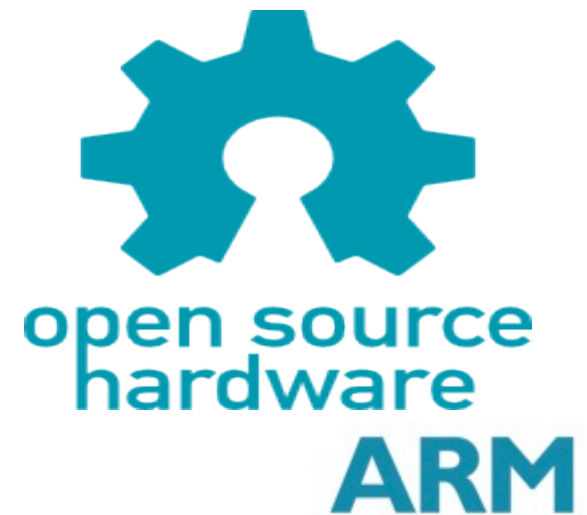
- **Quick, simple development experience with rich features**
  - Easy access to MCU I/O
  - 3-axis **accelerometer**/3-axis **magnetometer**
  - RGB LED
  - Add-on **Bluetooth** Module
  - Built-in Ethernet/Add-on **Wireless** Module
  - Micro SD
- **Arduino** shield compatible
- Flash programming functionality enabled by OpenSDA debug interface



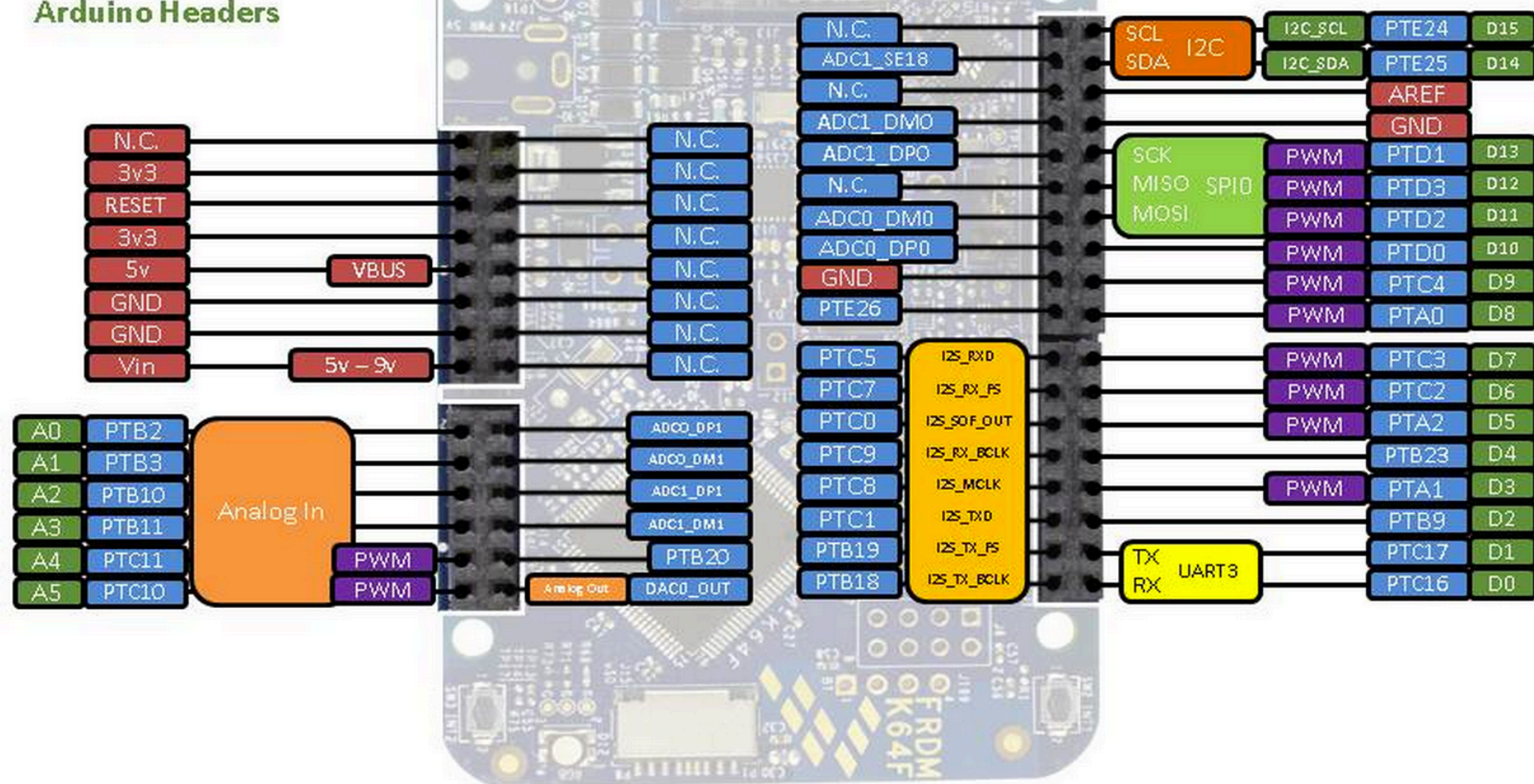
# OpenSDA & DAPLink Interface Firmware



- **DAPLink Interface Firmware includes:**
  - USB HID CMSIS-DAP Run-control debug interface
  - USB MSC disk for drag 'n' drop flash programming
  - USB CDC serial interface between the host and target



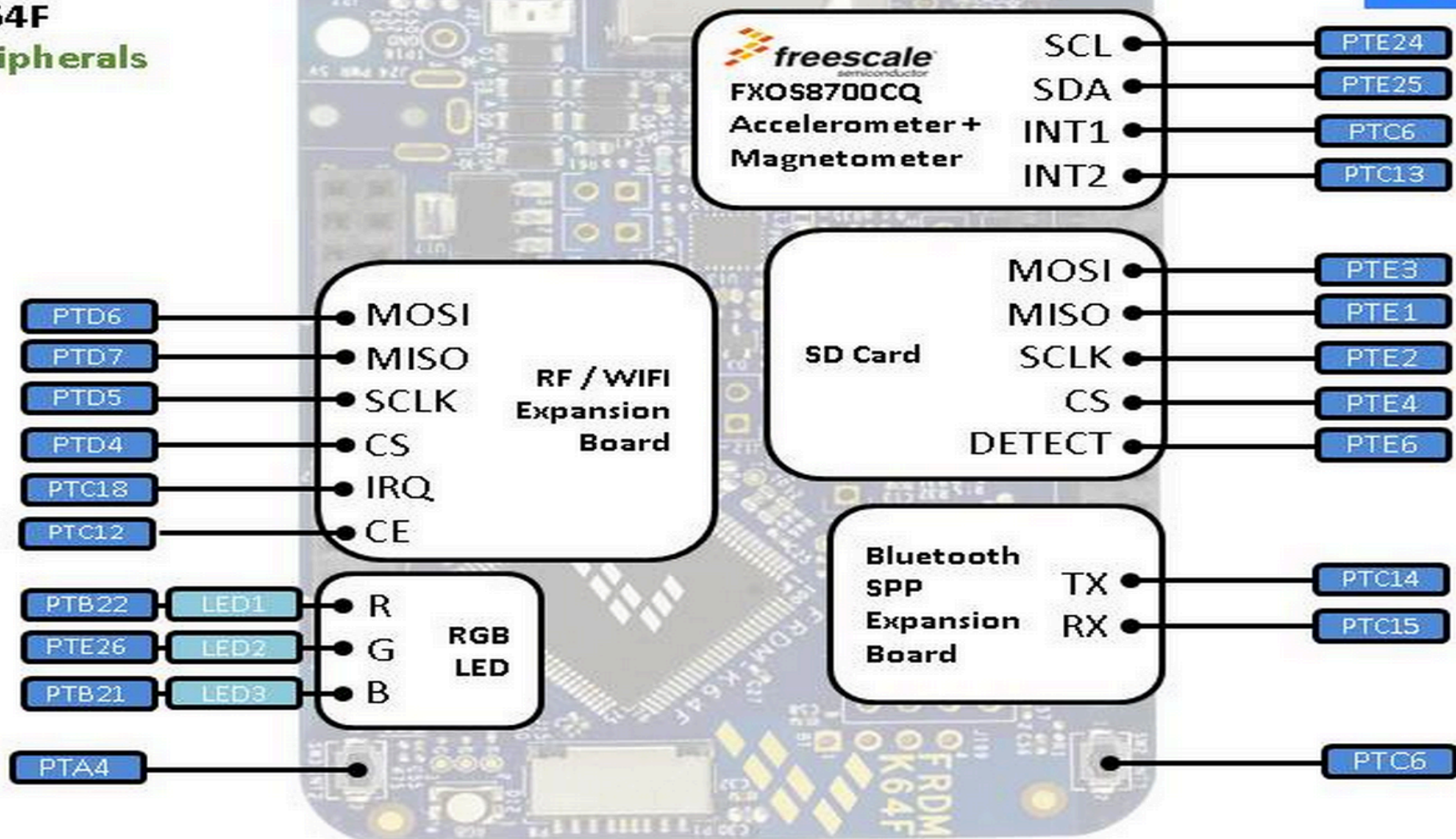






**freescale™**  
FRDM-K64F

Additional Peripherals





# Summary

- mbed IDE
  - C/C++ code editing and highlighting
  - Cloud-based build system
  - Documentation generation and integrated viewer
  - Distributed version control tools
  - Import from mbed.org or drag & drop zip import
  - Export to popular toolchains
  - Private personal workspace
- mbed Tools
  - Dependency tracking for code
  - Repository hosting & remote access
  - Wiki engine for tutorials and notes
  - Bug tracking
  - Admin control panel
- Components
  - Portability across platforms
  - Canonical reference
  - Categories - working groups
  - Find what you're looking for
- Community
  - Forum, Questions, Wikis,
  - How are questions integrated with the rest of the platform
  - Teams for group collaboration
  - Activity streams
  - Personal dashboard and ability to follow
  - United states: 17%, Japan: 14%, UK 8%, India 6%, long tail of others

# Workshop



# Hands-On Agenda

- Lab 1 – Hello World
  - Input / Output and serial module
  - Challenge – Change LED state based on button state
  - Challenge – Read serial characters and change RGB LED state
- Lab 2 – Interrupts and Timers
  - Interrupts and timers
  - Challenge – Drive RGB LED while sleeping between state change
  - Challenge – Change RGB LED state but sleep between state change
- Lab 3 – Using Sensors
  - I2C accelerometer / magnetometer
  - Challenge – Control an LED in a meaningful way based on the sensor readings
  - Challenge – Add sensor handling using the RTOS

<http://mbed.org/niworkshop>





***Thank You***

<http://mbed.org>