



**MICROCHIP**

**Extending the Arduino  
Runtime Environment  
for 32-bit MCUs**

**June 2016**

# Agenda

---

- **Introduction**
  - From the Outside Looking In
- **Extending the Arduino System**
  - 32-bit Development Boards
  - Industrial Applications
  - Runtime Environment
- **What's Coming Soon**
- **Conclusions**

**chipKIT™**



# **chipKIT™ Introduction**



# What is the chipKIT™ Platform?

---

- **A high-performance, Arduino-compatible computing environment designed for ease-of-use and rapid prototyping**
- **Using hardware abstraction and PIC32 MCUs, the platform is intended for beginners as well as experienced engineers**
- **The system also provides a migration path to professional engineering tools**

## What's in a Name?

---

- In 2010 we resolved to offer our 32-bit technology to this community
  - So we visited with the Arduino leaders
    - ... but that didn't go very well*
  - In response, we created our own brand
-



## What's in a Name?

---

- **chipKIT™ name is a trademarked brand of Microchip Technology Inc.**
- **Free license is available to encourage community participation**
- **We make it easy to leverage the chipKIT brand and ecosystem**

# What's in a Name?

---

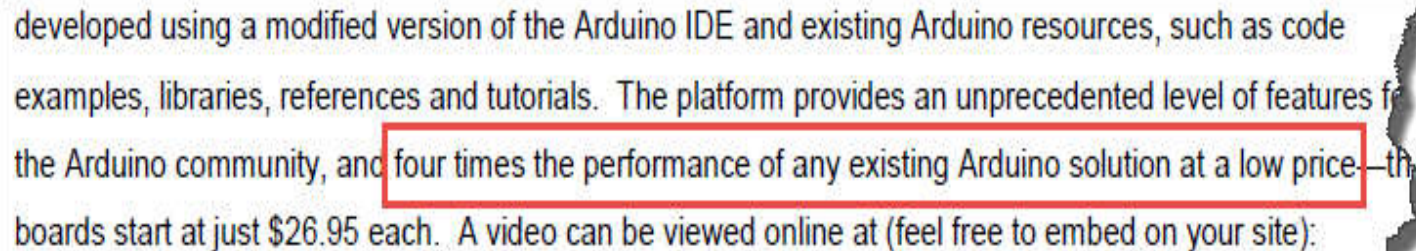
- After rejection by Arduino.cc, we named our board “chipKIT Uno32”, the first 32-bit Arduino-compatible development board
- The Uno32 name was seen as provocative  
*... Adafruit refused to carry our boards, because the names were too similar*
- Lesson learned: Use a less confrontational name



# chipKIT™ Introduction

---

- **At Maker Faire Bay Area in 2011, we issued a press release with a direct comparison of our 32-bit solution vs. Arduino 8-bit boards**

A graphic of a pair of scissors cutting through a piece of paper, revealing text underneath. The text is a press release snippet.

developed using a modified version of the Arduino IDE and existing Arduino resources, such as code examples, libraries, references and tutorials. The platform provides an unprecedented level of features for the Arduino community, and **four times the performance of any existing Arduino solution at a low price**—the boards start at just \$26.95 each. A video can be viewed online at (feel free to embed on your site):

- **Later, we received an unfriendly letter**
- **Lesson learned: Use a more respectful tone**





# Engineers Know How to Cooperate



- **We created a multi-platform version of Arduino IDE**
  - Retained AVR functionality
  - Added PIC32 support
- **Later, chipKIT engineers and Arduino engineers worked together to incorporate this functionality into Arduino IDE**
- **Lesson learned:**  
**Cooperation is more productive than confrontation**

```
/*
 * Blink
 * Turns on an LED on for one second, then off for one second, repeatedly.
 *
 * This example code is in the public domain.
 */

void setup() {
  // initialize the digital pin as an output.
  // Pin PIN_LED1 has an LED connected on most Arduino and compatible boards
  pinMode(PIN_LED1, OUTPUT);
}

void loop() {
  digitalWrite(PIN_LED1, HIGH); // set the LED on
  delay(1000);                 // wait for a second
  digitalWrite(PIN_LED1, LOW);  // set the LED off
  delay(1000);                 // wait for a second
}
```





# Let's Keep Working Together

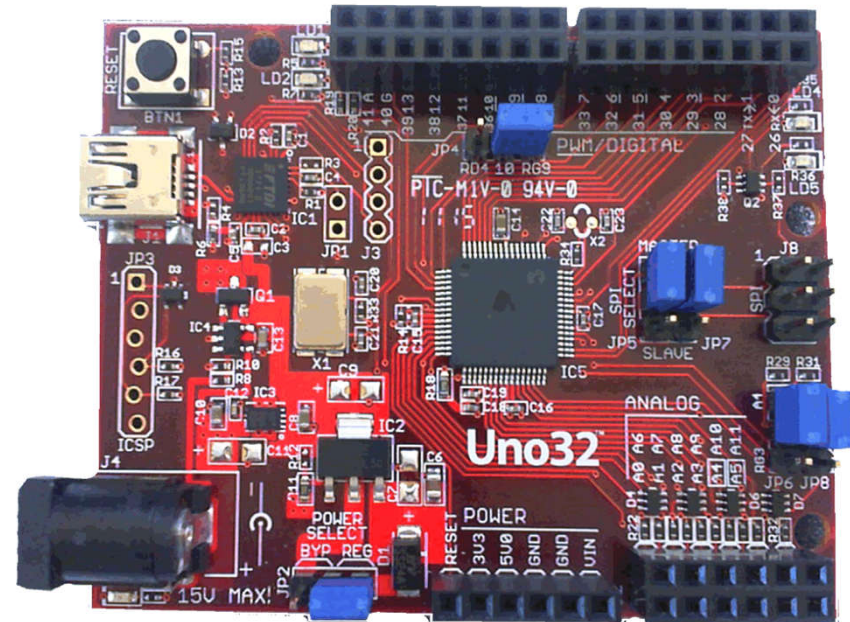
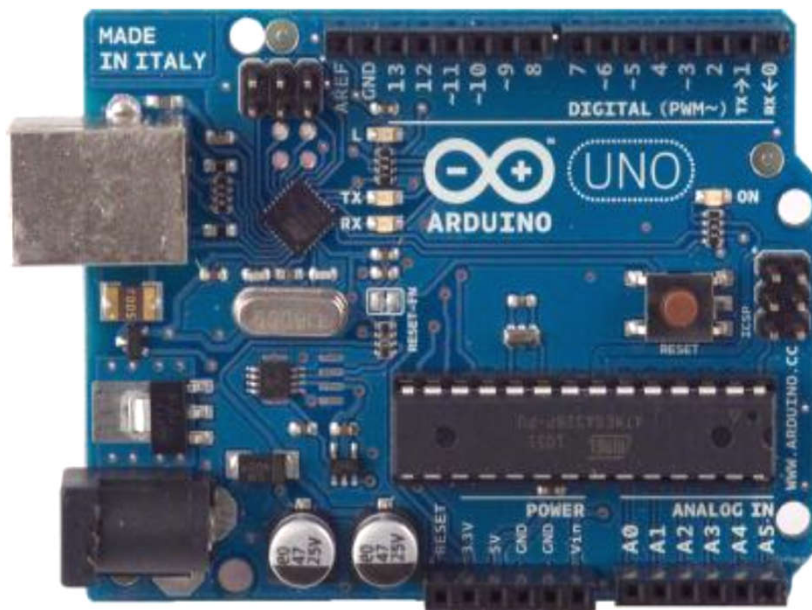
---

- **Continue to push the technology forward**
- **Continue to support Open Source**
- **Share resources when appropriate**
- **Respect the Arduino community**

# **Extending the Arduino System: 32-bit Development Boards**

# The First chipKIT™ Board

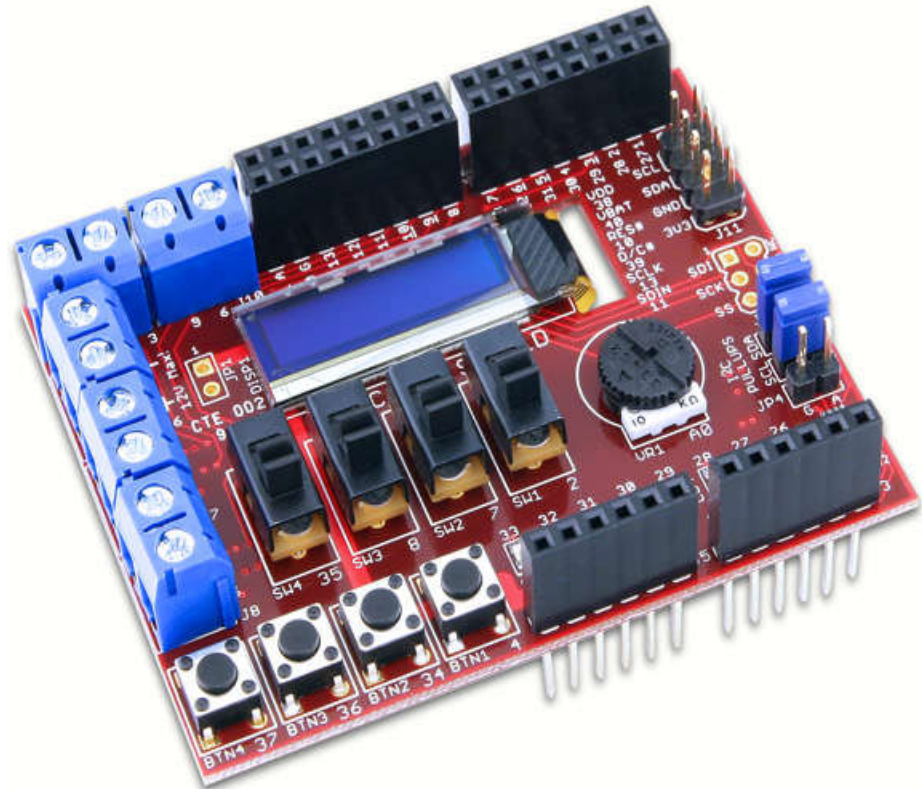
- Pin-out compatible with Arduino shields
  - But much faster, more memory and I/O



**Lesson learned: Stackable headers are expensive**

# Basic I/O Shield

- 128x32 OLED display
- Four buttons
- Four slide switches
- Eight LEDs
- Four open drain FETs
- I<sup>2</sup>C EEPROM
- I<sup>2</sup>C Temp sensor
- Potentiometer

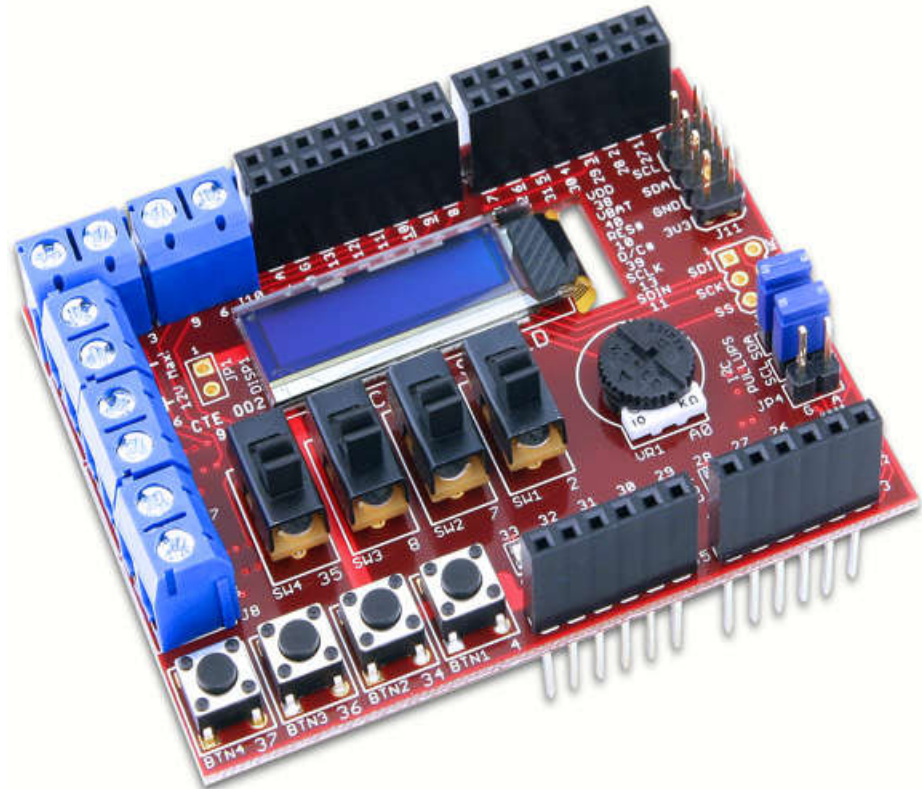


*Why did we call this “Basic”?*



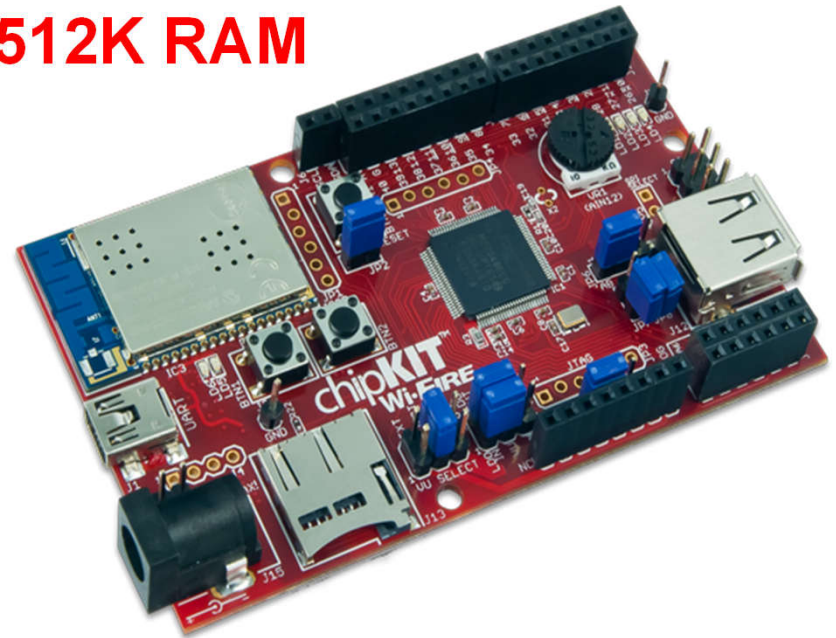
# *Complesso* ~~Basic~~ I/O Shield

- 128x32 OLED display
- Four buttons
- Four slide switches
- Eight LEDs
- Four open drain FETs
- I<sup>2</sup>C EEPROM
- I<sup>2</sup>C Temp sensor
- Potentiometer



# chipKIT™ Wi-FIRE

- PIC32MZ MCU w/ **2MB Flash, 512K RAM**
- 200 MHz 32-bit MIPS core
  - Four 64-bit accumulators
  - Floating Point Unit
- MRF24WG0MA WiFi module
- Micro SD card slot
- 50 MHz SPI ports
- USB 2.0 Full-Speed / Hi-Speed controller
- 43 available I/O pins with on-board user interfaces:
  - 4 LEDs, 2 Buttons, 1 Potentiometer



# **Extending the Arduino System: Industrial Applications**



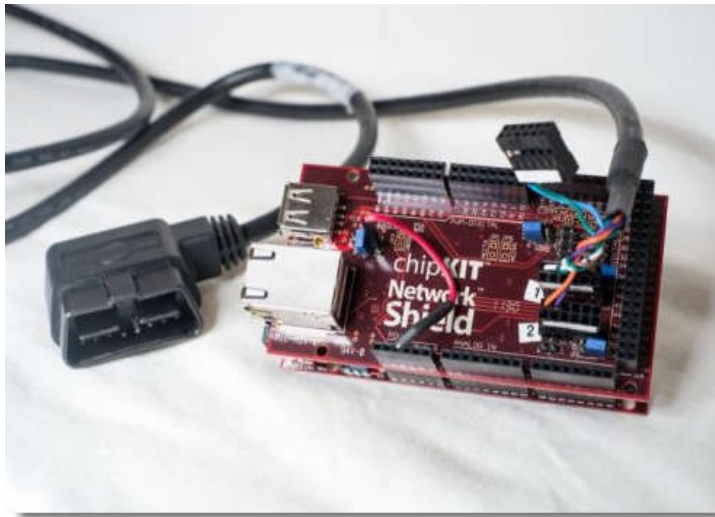


# OpenXC Platform

Open-source CAN bus  
interface created by Ford

Based on chipKIT™  
Max32 and Network  
Shield

OpenXC

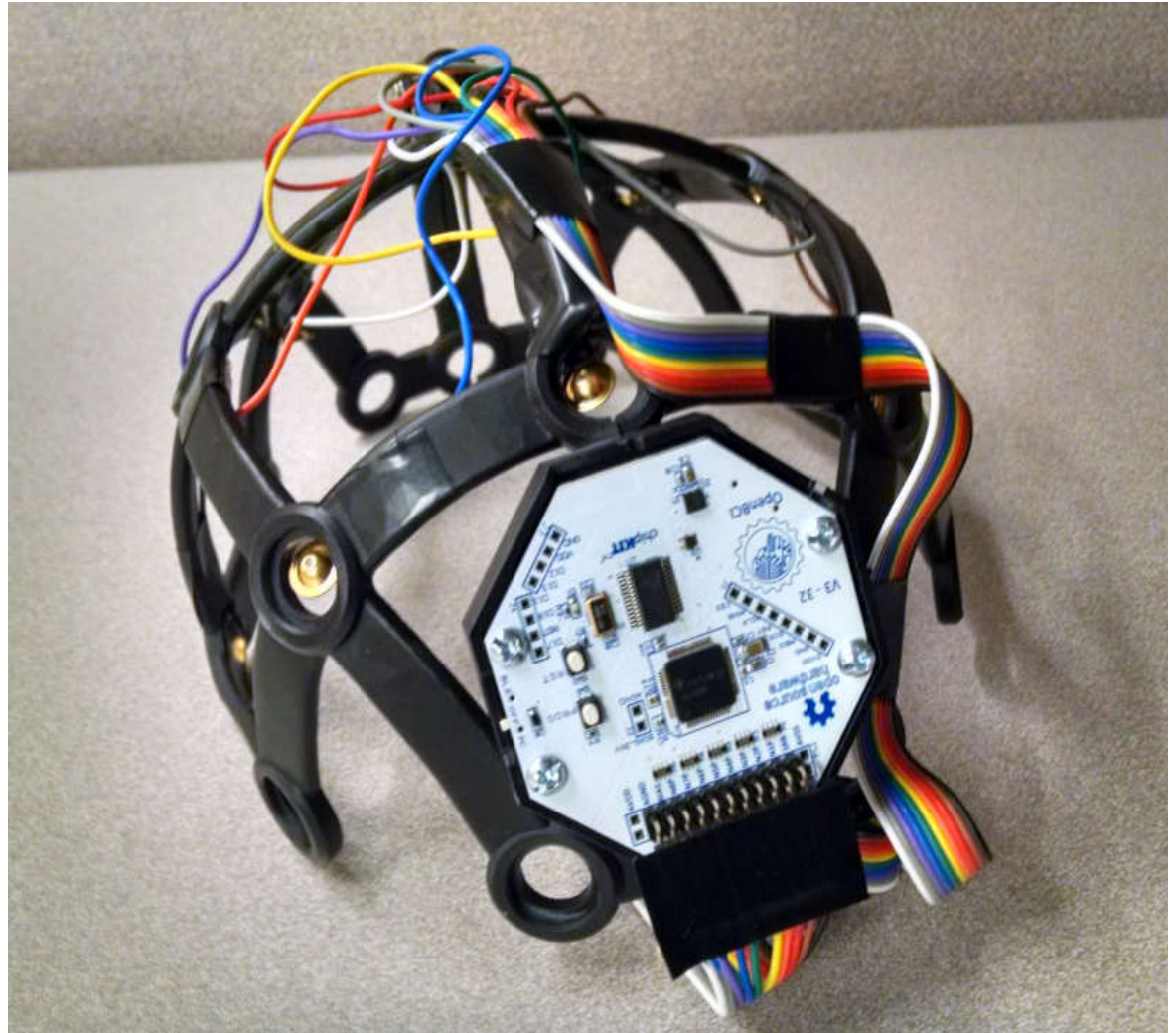




# OpenBCI Brain Computer Interface

**Open-source brain  
research tool,  
funded by  
Kickstarter**

**Used at many  
leading institutions**

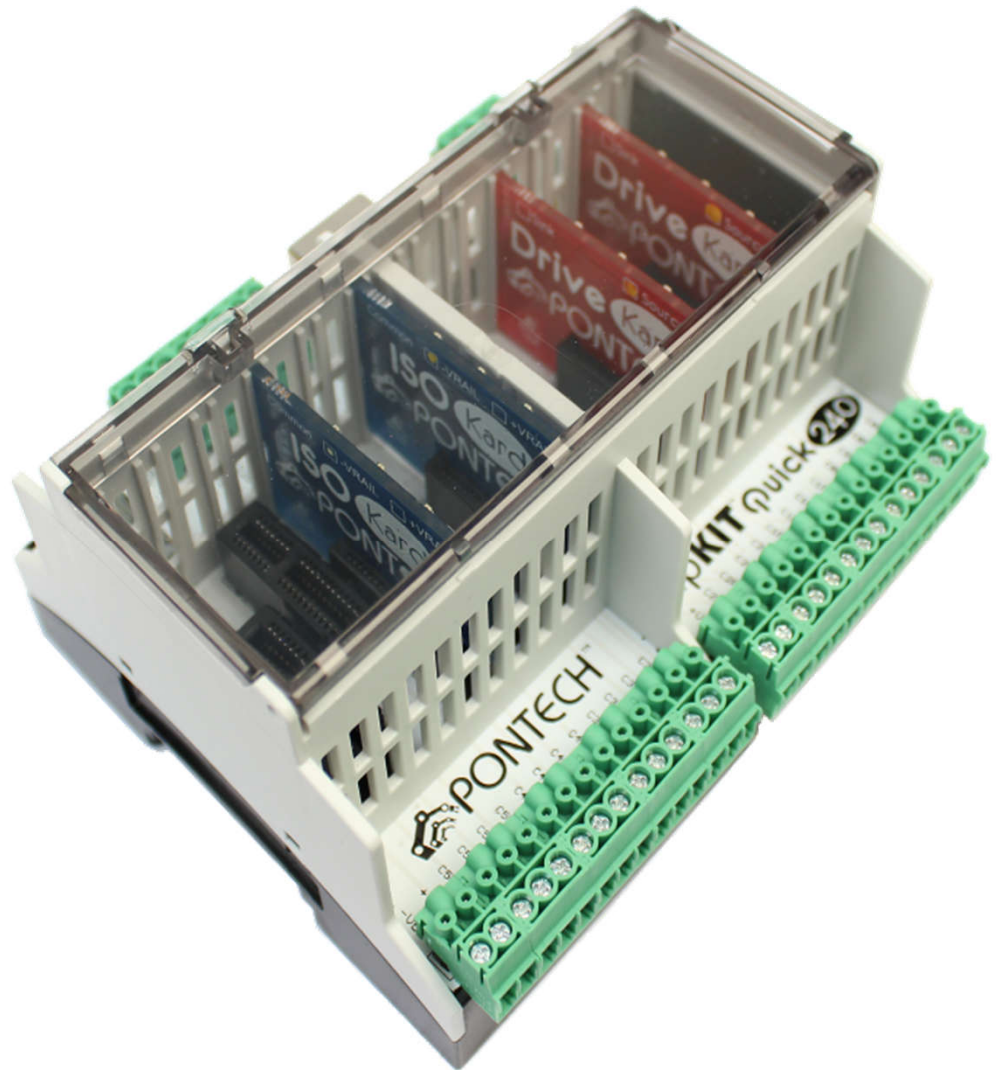




# PONTECH Quick 240 Industrial Controller

**Open-source factory  
automation controller,  
based on chipKIT Max32**

**Used at biomedical and  
electronics factories in  
California**



# **Extending the Arduino System: Runtime Environment**



# Open-Source Libraries

---

**Here are some of the C++ libraries we have contributed to the community:**

**Core Timer  
SoftPWMServo  
Task Manager  
SoftSPI  
DSPI  
DTWI**

**TCP/IP Stack  
HTTP Server  
DFATFS  
DSDVOL  
RAMVOL  
DisplayCore**

*Where there is overlap with a standard Arduino library, we've enhanced it with additional error codes and/or callback functions.*

# Core Timer Library

---

- **Simple time-based callback mechanism for scheduling system functions**
- **Registered functions are called from ISR context**
- **25ns resolution, up to 90s in the future**
- **Only a few Core Timer functions are permitted, to minimize latency**
- **Provides for accurate timing of system functions**





# SoftPWMServo Library

---

- **Utilizes Core Timer functions to schedule RC servo or AnalogWrite() type PWM output**
- **Can run 82 servos simultaneously using only 10% of CPU time with very low jitter**
- **All pins can be servo or PWM output, simplifying project development**



# SoftPWMServo Example

---

```
#include <SoftPWMServo.h>

char dim = 0;

void setup() {}

void loop() {
    SoftPWMServoPWMWrite(0, dim++); // fade pin 0
    delay(20);
    if (dim == 70) dim = 0;
}
```



# Task Manager

---

- **User can register functions to be called at periodic intervals, or at a specific time**
- **Task scheduling is hidden within system functions `loop()` and `delay()`**
- **Scheduling is non-preemptive and round-robin**
- **Complexity is hidden from the user**



# Task Manager Example

---

```
#include <SoftPWMServo.h>

char dim = 0;
void UpdateLED(int id, void * tptr) {
    SoftPWMServoPWMWrite(0, dim++);  // fade pin 0
    if (dim == 70) dim = 0;
}

void setup() {
    createTask(UpdateLED, 20, TASK_ENABLE, NULL);
}

void loop() {}
```



# SoftSPI Library

- **SoftSPI supports basic SPI comms on any set of 4 pins**
- **Multiple slaves on one SPI bus can be problematic; SoftSPI allows for arbitrary separate SPI busses**
- **PIC32 speed and efficiency is leveraged to hide bit-banging complexity**



# SoftSPI Example

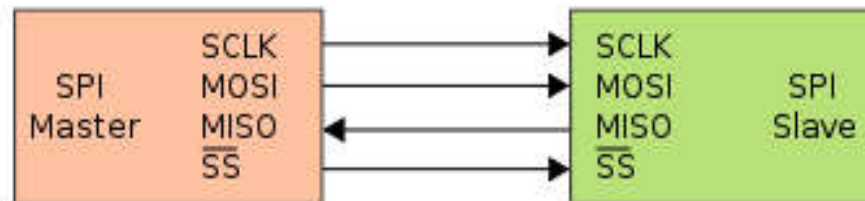
---

```
#include <SoftSPI.h>

SoftSPI spi;
/* ... */
    spi.begin(pinSS, pinMOSI, pinMISO, pinSCK);
    spi.setSpeed(250000);                // 250 kHz
    spi.setMode(SSPI_MODE0);
    spi.setDirection(SSPI_SHIFT_LEFT);
    spi.setDelay(6);                     // 6 uSec
/* ... */
    spi.setSelect(LOW);
    spi.transfer(5, Send_Bytes, Receive_Bytes);
    spi.setSelect(HIGH);
```

# DSPI Library

- **DSPI supports advanced SPI comms**
- **Uses all hardware SPI ports on chipKIT boards (2, 4, or 6) up to 50 MHz**
- **Supports 32-bit transfers and block transfers under Interrupt control**





# TCP/IP Library

- **Digilent's Embedded IP Stack**
  - Mostly, RFC 1122 / 793 compliant
  - Supports multiple concurrent network interfaces
- **Written in C, with C++ wrappers**
- **Processor Specific Hardware Abstraction Layer**
  - Big/Little Endian, Timers, Checksum, Processor speed
- **MAC/PHY Abstraction Layer (Network Adaptors)**
- **Memory Abstraction Layer**
  - Network Packets and Socket Buffers
- **Designed for a cooperative, non-preemptive embedded environment**

# HTTP Server Library

---

- **Implements basic HTTP Server framework**
  - Manages Network / WiFi connections, TCP sockets, cooperative task scheduling
  - Manages reading / writing data from / to the TCP socket, URL identification, line parsing, and calling Compose functions
- **Provides helper functions to create basic HTTP headers**
- **Enables multiple concurrent connections and page processing**

# Robust Network Software

**Torture testing  
with 30 clients  
banging away on  
a server for days  
at a time**

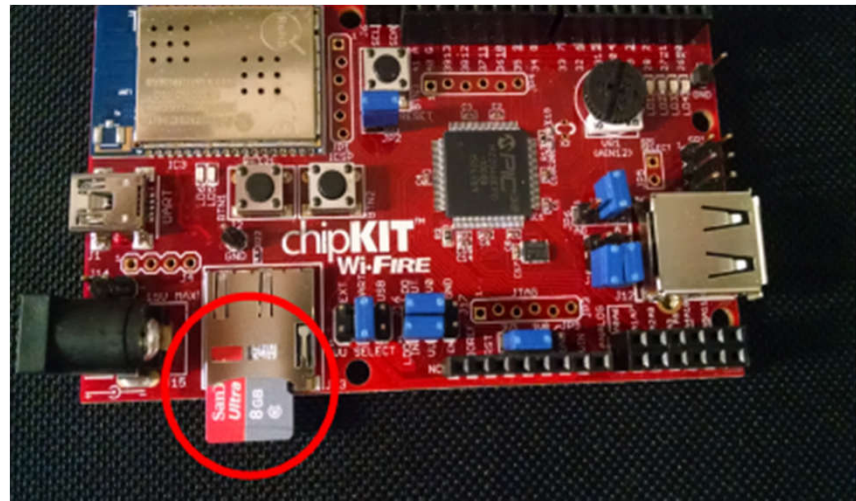
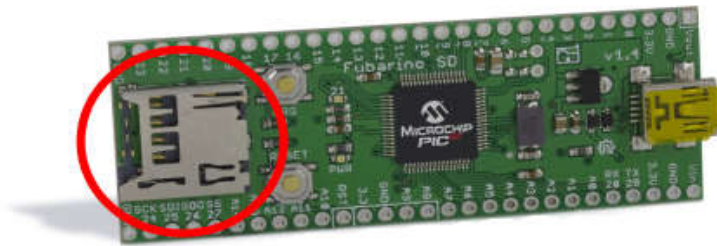
**Every router is  
different;  
creating a truly  
robust system is  
difficult**





# DFATFS Library

- Several chipKIT boards have microSD card slots



- Users can mount Windows-compatible file systems, create and access files stored on memory cards
- Supports virtual disk volumes in internal or SPI RAM
- Up to 5 volumes can be mounted and used at once



# DSDVOL Example

---

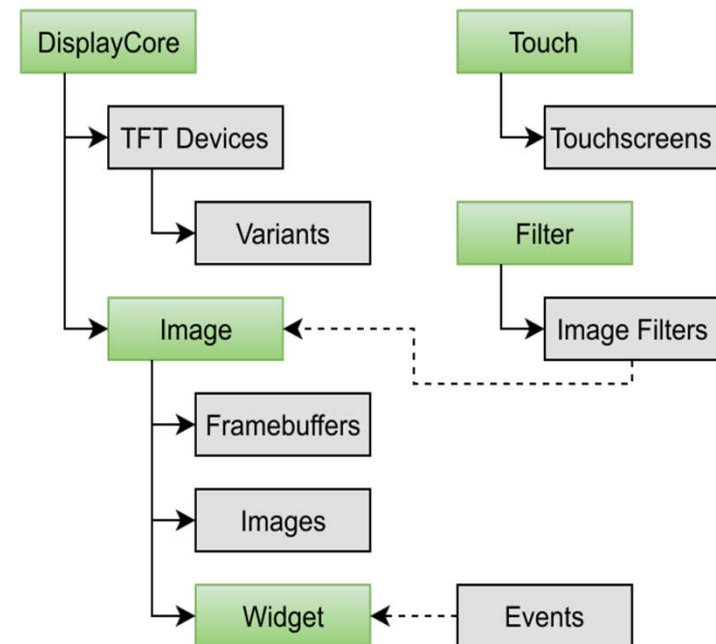
```
#include <DSDVOL.h>

// create the sd volume and a file instance to use
DefineSDSPI(dSDSpi);           // Create an SPI object
DSDVOL  dSDVol(dSDSpi);        // Create an SD Vol
DFILE   dFile;                 // Create a File handle

/* ... */
// Mount the SD Vol to drive "0"
// Use the helper Volume strings provided by szFatFsVols
if((fr = DFATFS::fsmount(dSDVol, DFATFS::szFatFsVols[0], 1)) == FR_OK)
{
    Serial.print("Drive ");
    Serial.print(DFATFS::szFatFsVols[0]);
    Serial.println(" mounted!");
}
```

# DisplayCore Library

- **Supports several different display types**
  - TFT, OLED, LCD, Virtual, and Touch
- **Provides a large collection of fonts (70+), icons, drivers, and widget toolkits**
- **Uses a modular design to optimize memory use**





# DisplayCore Example

---

```
#include <Picadillo.h>
#include <Roboto.h>

Picadillo tft;

void setup() {
    tft.initializeDevice();
    tft.fillScreen(Color::Black);
    tft.setFont(Fonts::Roboto);
    tft.setTextColor(Color::Green);
    tft.setCursor(100, 100);
    tft.println("Hello World");
}

void loop() {}
```

## **What's Coming Soon**



# Long Range, Low Power Wireless

---

- **RN2483 module has a simple UART interface**
  - **Compatible with any MCU**
- **Long range coverage with low power**
  - **Up to 5km range in the city**
  - **Up to 15km in the country**
  - **>10 year battery life capability**
- **No interference from Wi-Fi, Bluetooth, GSM, LTE, etc**





# LoRa<sup>®</sup> Modem Features

---

- **Transmit output power:**
  - +18 dBm @ 915 MHz (FCC)
  - +14 dBm @ 868 MHz (ETSI)
  - +10 dBm @ 433 MHz
- **High sensitivity: down to -148 dBm**
- **Transmit current: 40 mA typical at +14 dBm**
- **Receive current: 14.2 mA typical**
- **Sleep mode/ low power down mode: 1 uA typical**
- **Excellent blocking immunity**



## More chipKIT Libraries

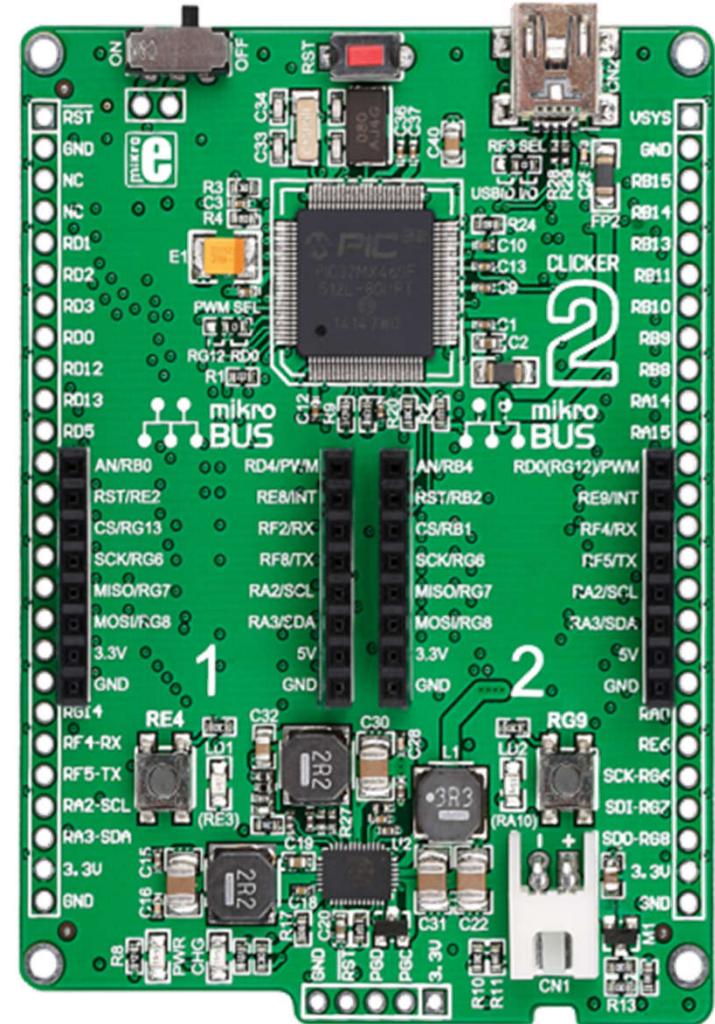
---

- **Standard C File I/O**
  - Provides `fopen()`, `fread()`, `fprintf()`, etc.
- **Harmony USB**
  - Provides Hi-Speed, Host, HID, MSD, etc.
- **Audio Special Effects**
  - For guitar and other musical instruments
  - Provides Filters, Chorus, Flanger, Phaser, etc.



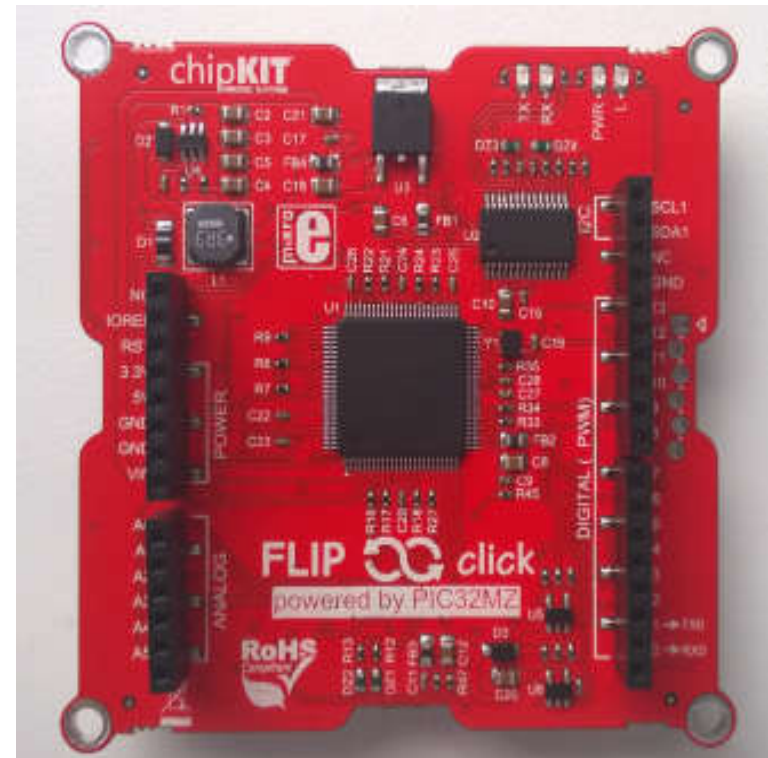
# MikroE Clicker 2

- PIC32MX460 MCU at 80 MHz
- 512K Flash, 32K RAM
- LiPo battery power circuit
- 2 sockets for Click boards



# MikroE Flip & Click

- PIC32MZ MCU at 200 MHz
- 2048K Flash, 512K RAM, Floating Point Unit
- 4 sockets for Click™ expansion boards on the “Flip” side
- Uno-style I/O headers





# chipKIT™ Lenny

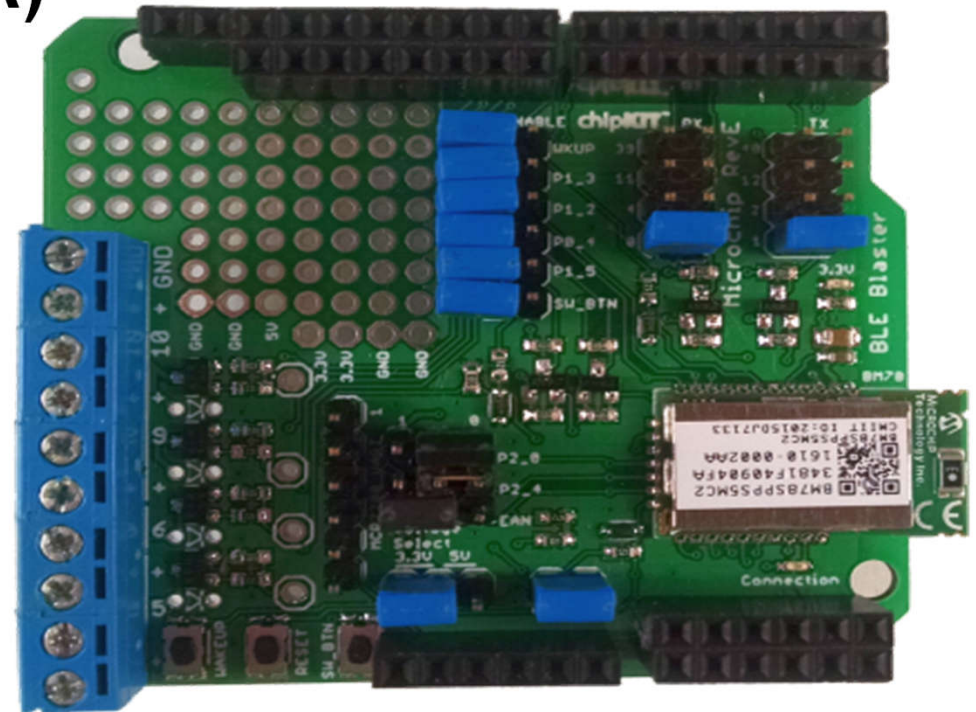
- PIC32MX270 MCU at 40 MHz
- 256K Flash, 64K RAM
- Peripheral Pin Select (PPS)
- Beefy power circuit





# chipKIT™ BLE Blaster

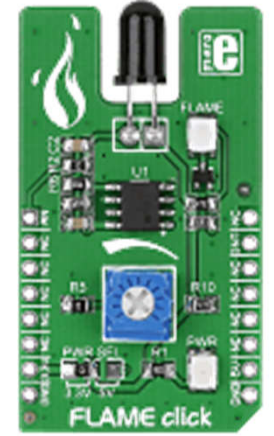
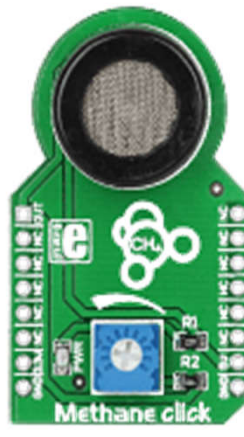
- Dual mode: Bluetooth Classic and BLE 4.2
- Designed for remote monitoring and control
- Compatible with 3.3V or 5V MCUs
- 4 Power FETS (20V, 4A)
- 4 RX,TX options





# chipKIT™ Data Station

- Four-channel remote datalogger w/ LoRa® and BLE 4.2
- Sends data to remote host, w/ SD card backup
- Android app for config and control
- Supports these Click™ sensors:

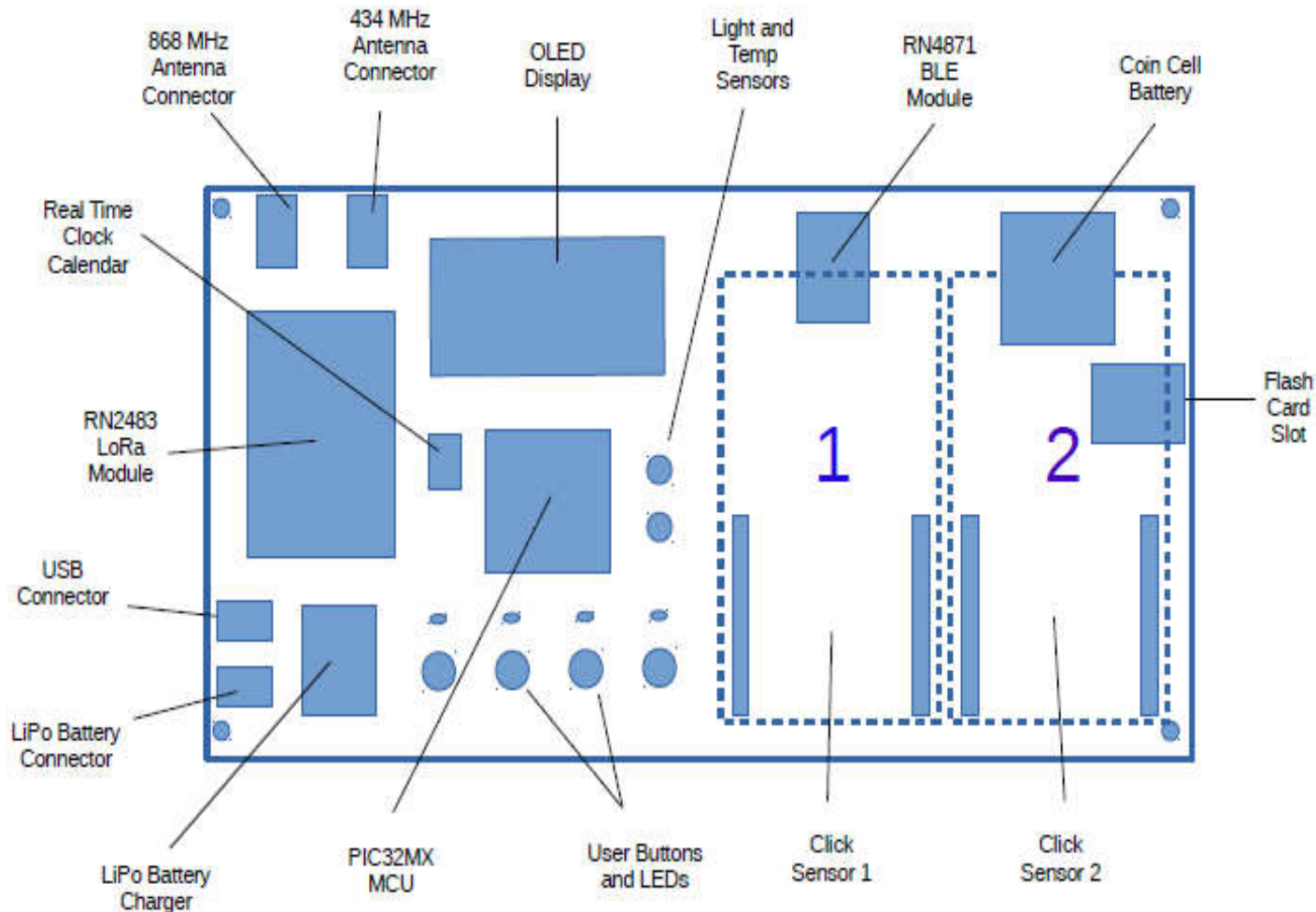




# chipKIT™ Data Station

**On-board  
Temp & Light  
sensors, plus  
2 Click™  
sockets**

**Operates on  
battery power  
for weeks or  
months at a  
time**

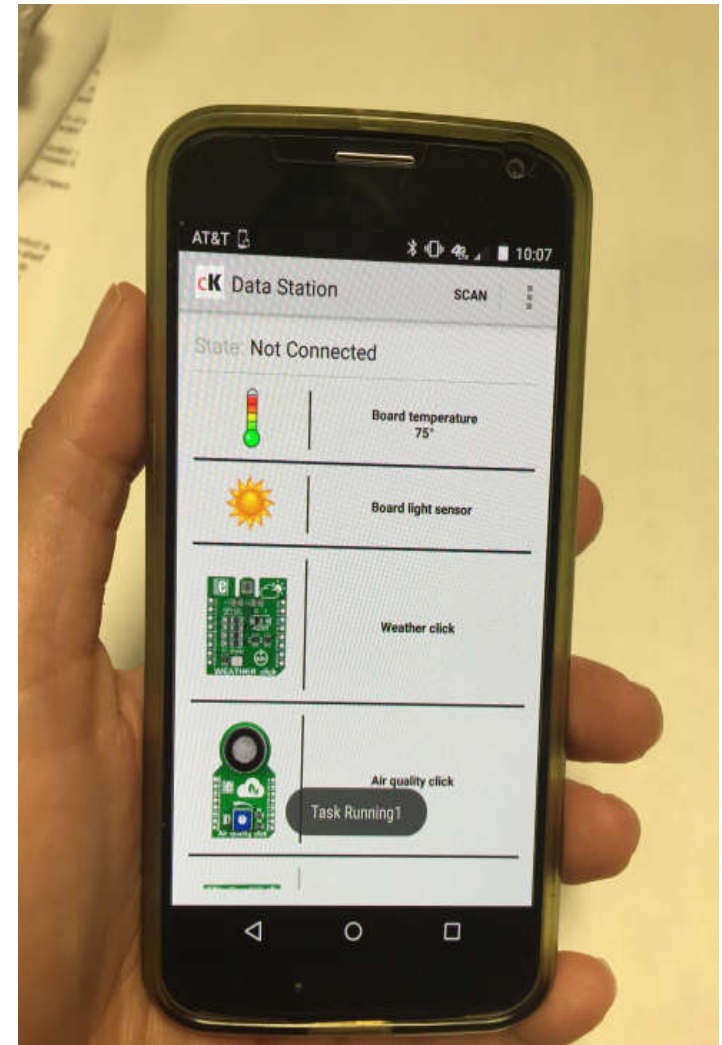
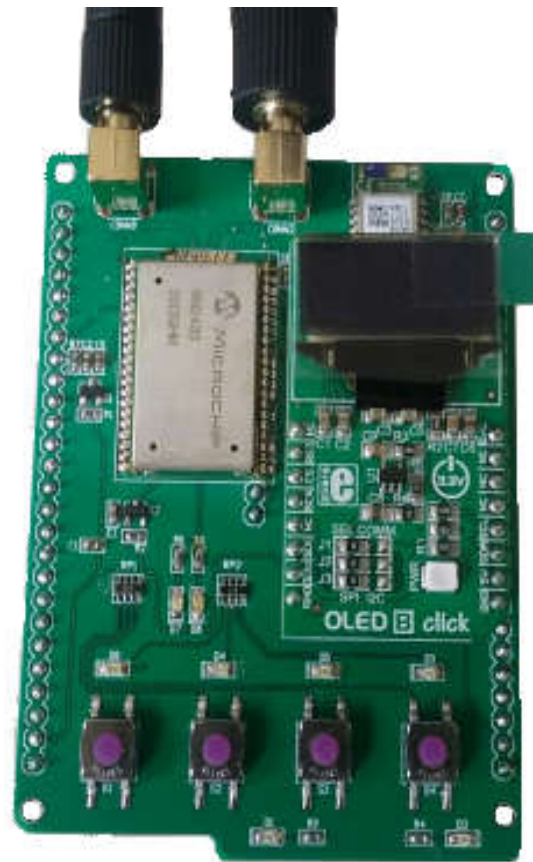




# chipKIT™ Data Station

**Prototype hardware and software available today**

**LoRa® and BLE modules controlled with simple ASCII commands**



# Conclusions



# Conclusions

---

- **Competition Can Be Fun**

*... but Cooperation is better*

- **Makers are Self-Directed Innovators**
- **Open Source is Important**
- **Do Good Things in the World**

*... no matter the Challenges*



**MICROCHIP**

**Thank you!**