A Leading Provider of Microcontroller, Security, Mixed-Signal, Analog & Flash-IP Solutions

Presented by:
Eugene Choi, Sr. Embedded Solutions Engineer
December 6, 2018

Easy to install and Easy-to-use Arduino® Compatible chipKIT® Platform for Beginners

Agenda

Presentation will introduce software constructs and concepts:

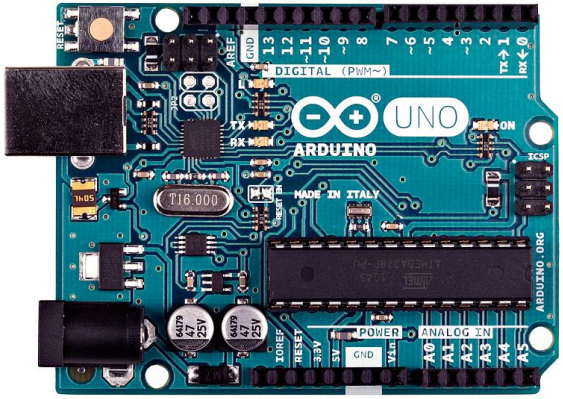
- **Arduino®/chipKIT®: What are they?**
- **Arduino IDE quick start**
- **Demo #1: Get blinky lights working**
- **Demo #2: Built in Serial I/O**
- **Using libraries**
- **How to debug a sketch with MPLAB® X Integrated Development Environment (IDE)**

Arduino®/chipKIT®: What are they?

- Both are rapid prototyping platforms
- Compatible with each other at source code and physical interface level
- Fills need for beginners to quickly construct simple microcontroller-based designs (and many more)

Arduino Beginnings

"Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments."



What is Arduino Now?

- **An environment for learning embedded systems**
- **A huge ecosystem**
  - Boards and extension shields
  - Thousands of libraries
  - Projects, web pages, videos
  - IDEs (PC and web-based)
- **Learning tool for most beginners**
  - THE de facto embedded teaching tool

Arduino Vocabulary

- **Sketch**
  - Source code/program in C/C++
- **Shield**
  - An add-on board to extend functionality
  - Standard pinout – function and layout
- **Hardware Abstraction Layer**
  - Eliminates need to know hardware details
  - Refers to pins in a standard way

chipKIT Platform Project Goals

- Start with Arduino compatibility
- Use powerful 32-bit PIC32 processors
- Remain open and inviting to all developers who want to contribute – hardware or software
- Continue to improve core Arduino libraries to be higher quality, more feature rich
- Create variety of board offerings - some with physical Arduino compatibility - all easy to use
- Continue to add high-performance libraries for PIC32 peripherals

chipKIT Platform Core Software

- **Includes:**
  - Compiler tool chain
  - Standard Arduino libraries
  - Contributed libraries (e.g., SoftPWMServo, DSPI)
- **A modular distribution:**
  - For use in many different IDEs
  - Provides hardware abstraction benefits

Demo #1
Blink that LED

Demo #1 Objective/Procedure

Demonstrate how simple it is to use the Arduino IDE by writing, compiling and uploading LED blink code into the chipKIT development board.

- Open Arduino IDE
- Write code
- Select proper development board target
- Put chipKIT board into bootloader mode
- Compile, upload, and run

Arduino IDE

- **Open Arduino IDE from desktop**

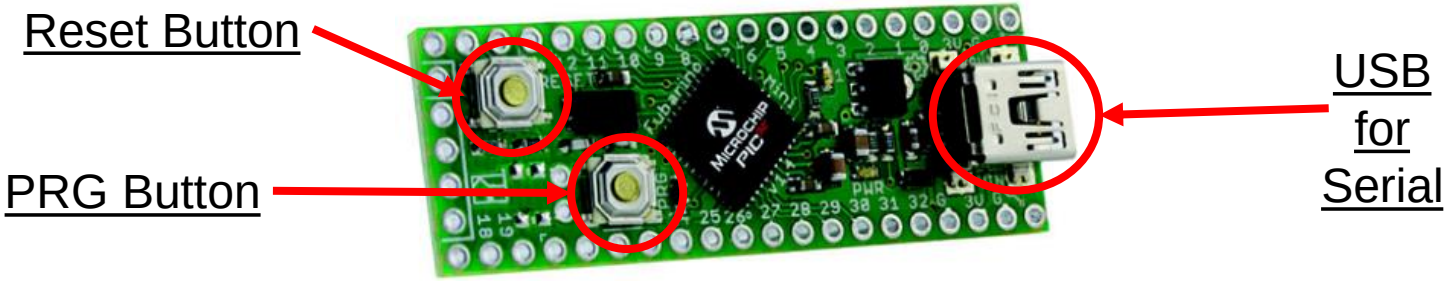
Add some code

- **Select File -> New**
- **Make your source code look like this:**

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
}
```

Prepare for the Upload

- **Select Tools -> Board -> Fubarino® Mini**
- **Activate the bootloader**
  - Press and hold PRG (program) button while pressing and releasing RESET

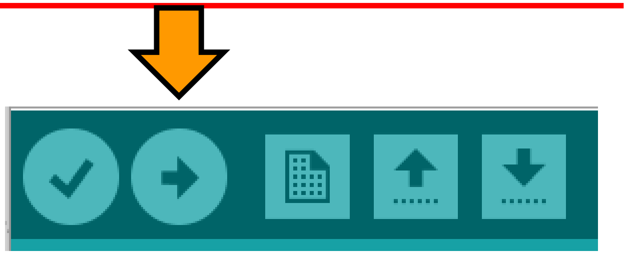


chipKIT™ Fubarino Mini

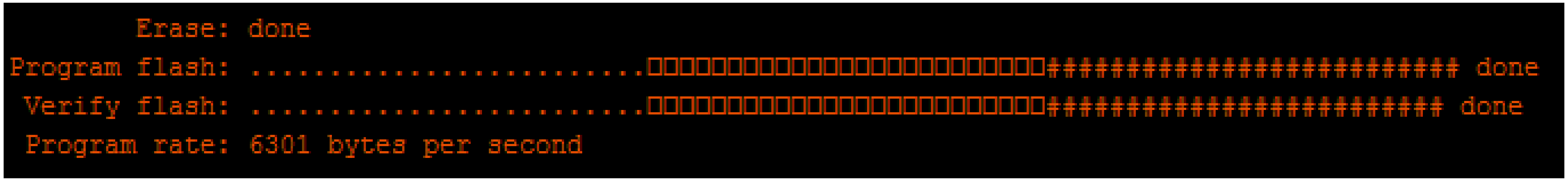
  - Should see fast blinking green LED = bootloader mode active

Compile, Program and Run

- Click the Upload button



- Arduino IDE will compile your sketch
- Arduino IDE will upload the sketch



```
        Erase: done
Program flash: .........................□□□□□□□□□□□□□□□□□□□□□□□□□########################## done
 Verify flash: .........................□□□□□□□□□□□□□□□□□□□□□□□□□######################### done
 Program rate: 6301 bytes per second
```

- Confirm that your green LED is blinking 1000 ms on, 1000 ms off

Basic Sketch

- **An Arduino program is called a sketch**
- **Language used is C++**
- **Embedded program event loop is built into the system**
- **Two required functions**
  - `setup()` – one time initialization
  - `loop()` – called repeatedly by system

Writing the Sketch

- **Every sketch must include:**

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Compiling

- **The sketch needs to be converted into a form that can be executed on the board**
- **After writing the sketch:**
  - The user presses the upload button
  - The sketch is compiled and a hex file is created
  - The hex file is uploaded to dev board using command-line program pic32prog
  - Any compile or upload errors are reported in the IDE output window

Bootloader

- **All chipKIT boards come pre-programmed with a bootloader**
- **Arduino IDE uses this bootloader to upload (program) the hex file to the board**
  - Replaces the need for a hardware programmer
- **There are two basic kinds of chipKIT boards:**
  - Automatic reset through on-board FTDI chip
  - Manual reset via PROGRAM button

Code – Blink LED

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

Summary

- **Experience ArduinoIDE workflow**
  - Enter new sketch
  - Compile and upload to chipKIT® board
- **Basic code template**
- **Basic Arduino Hardware Abstraction Library (HAL) concepts**
  - Set pin as output
  - Set output pin state
  - Create time delay

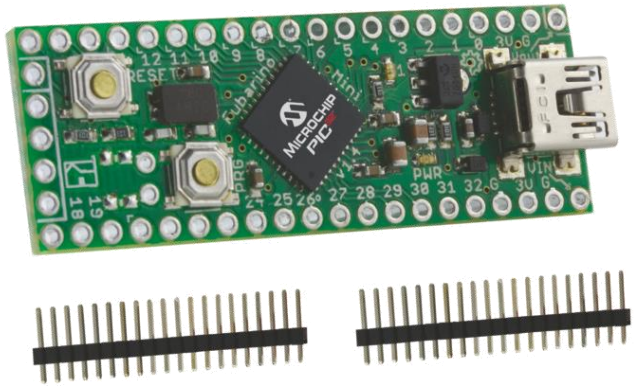
Why Use chipKIT Platform?

- **A PIC32-based, open-source software and hardware platform**
  - Compatible with Arduino code and add-on boards
- **Software tools to build, load and run Arduino sketches**
- **A diverse community of contributors, developers, and users**
  - More than just Digilent or Microchip

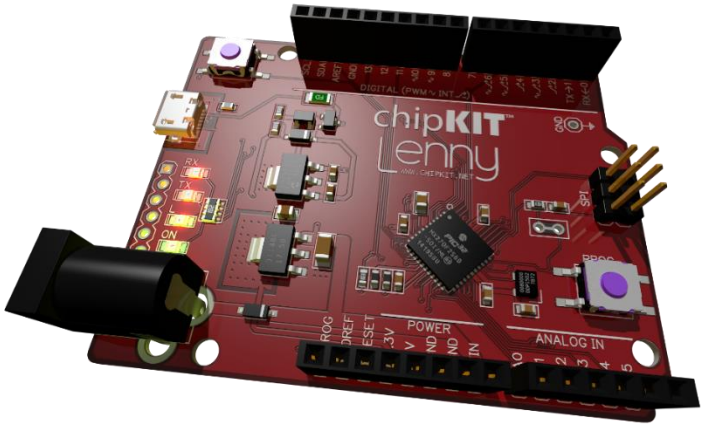
Some chipKIT Boards

uC32

Wi-FIRE

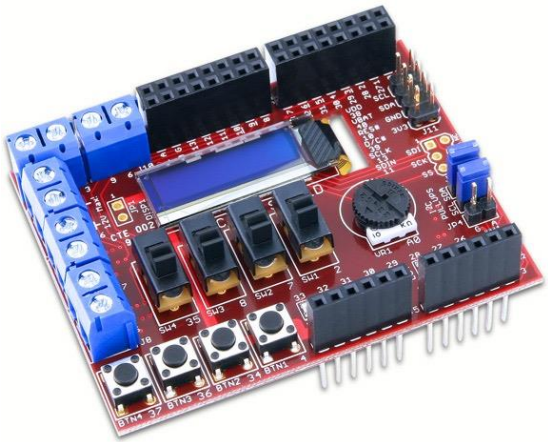


Fubarino® Mini



Lenny

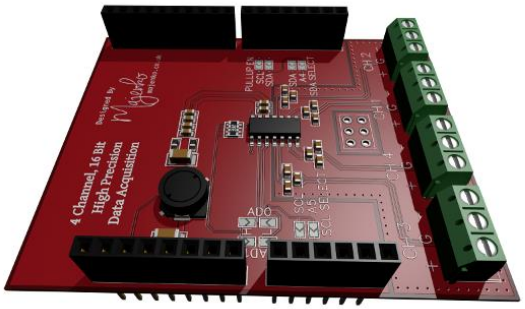
chipKIT Shields



Basic I/O Shield

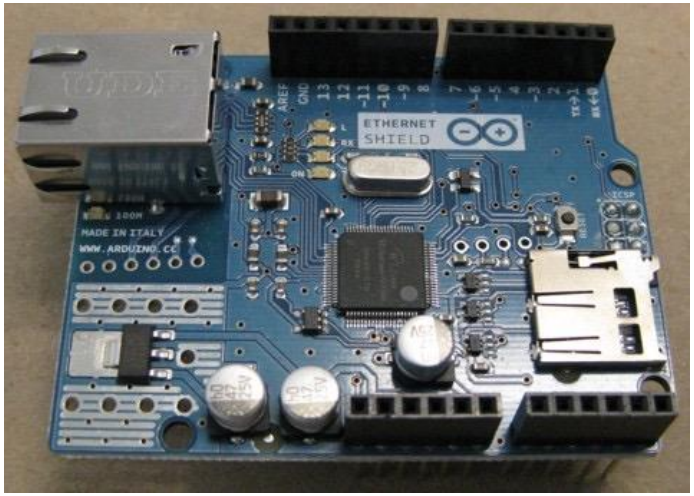
Wi-Fi® Shield

Quick IO Shield

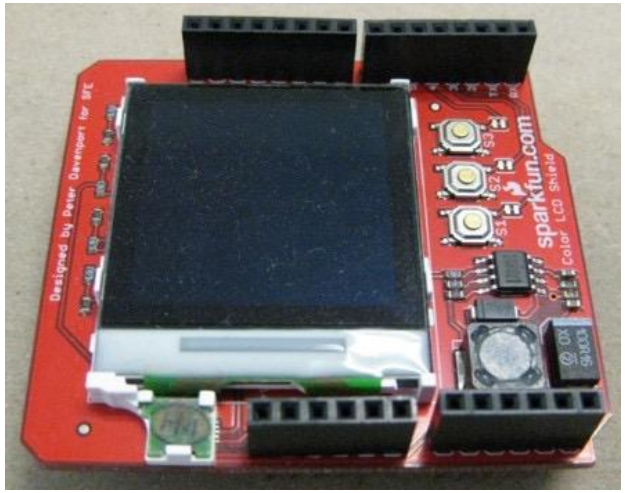


High Precision DAQ Shield

Some Arduino® Shields



Ethernet Shield



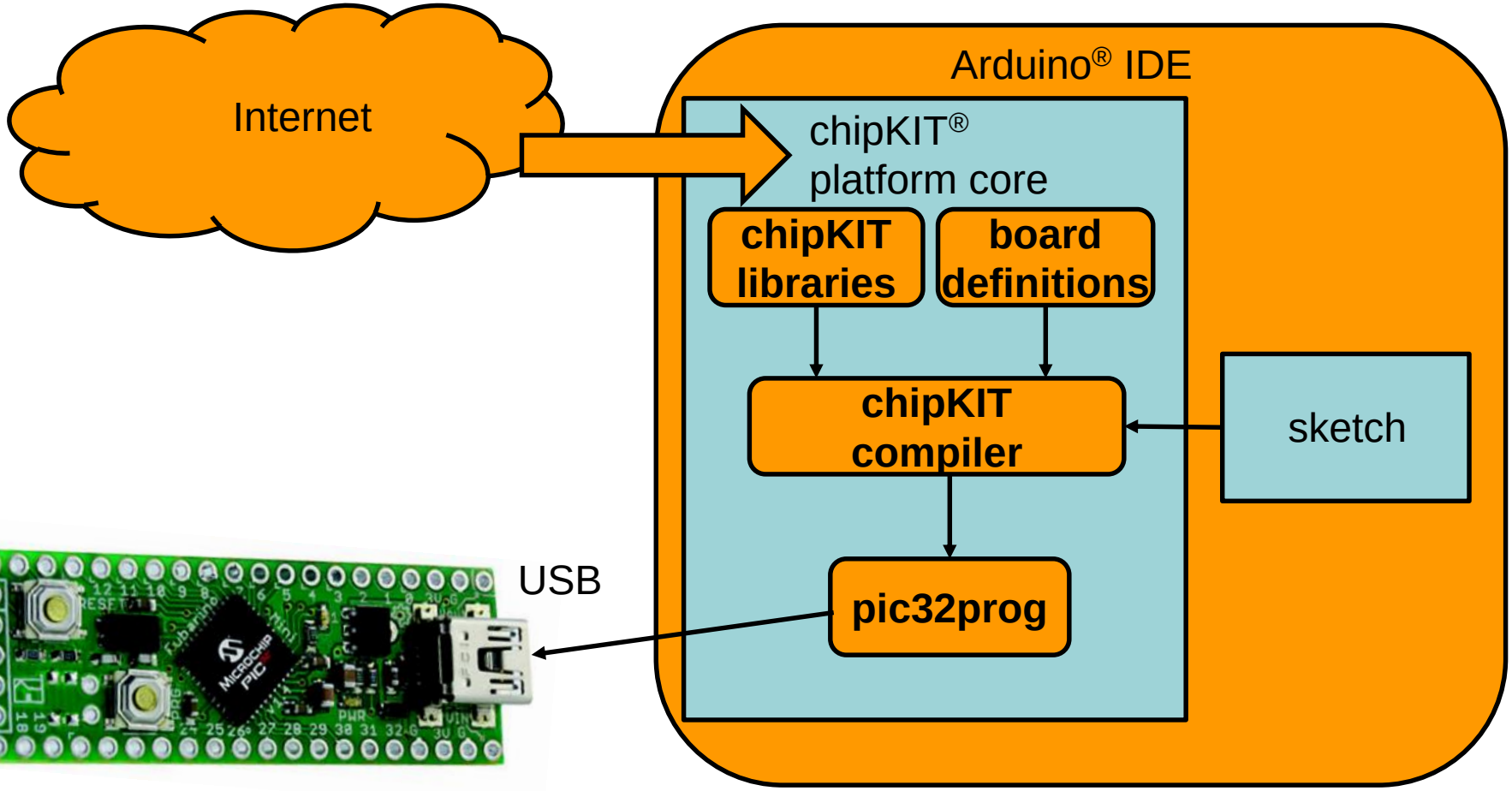
Graphic Display Shield

An enormous number of Arduino shields are suitable for use with chipKIT® boards, but you must mind your voltage levels.

Motor Drive Shield

Arduino® and ChipKIT® Software Environment

Software Architecture



chipKIT® board

chipKIT® Platform Core Software

- **Major components**
  - Compiler tool chain
  - Standard Arduino® libraries
  - Contributed libraries (e.g. SoftPWMServo, DSPI)
- **A modular distribution**
  - For use in many different IDEs
  - Provides hardware abstraction benefits

HAL Pins chipKIT® Hardware -> PIC32

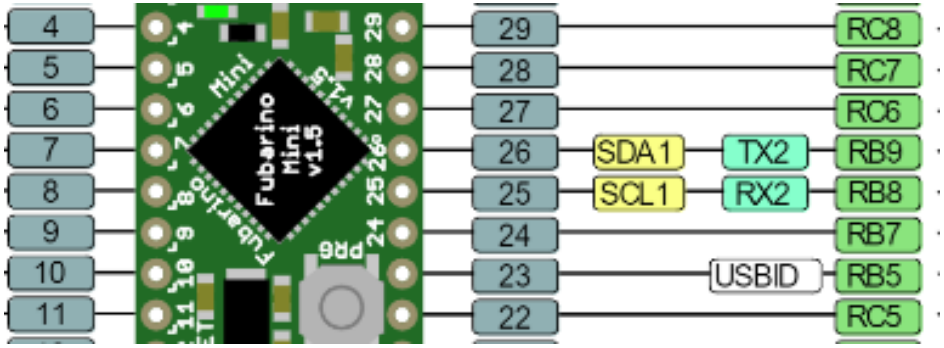
- **Every pin is given a ‘chipKIT’ number**
- **This is the only number needed in the code**

chipKIT Pin	PIC32 Pin	Port/Pin	Function
26	1	RB9	RPB9/SDA1/CTED4/PMD3/RB9

- **No need to worry about port or PIC32 pin**

```
pinMode(26, OUTPUT);
digitalWrite(26, HIGH);

TRISBbits.TRISB9 = 0;
LATBbits.LATB9 = 1;
```



Supported Toolchains

- **Arduino® IDE**
  - Supports many third-party plug-ins, including chipKIT® technology core
- **UECIDE**
  - Similar to Arduino IDE, but substantially rewritten and improved, supports chipKIT technology core
- **MPIDE**
  - An early fork of Arduino IDE, w/ chipKIT technology core built-in
  - Archived in August 2015
- **Other Sketch-Enabled Environments**
  - EmbedXcode, Visual Micro, PlatformIO, etc.
- **MPLAB® X IDE**
  - NetBeans-based professional IDE (Win, Mac, Linux)
  - Supports chipKIT hardware; sketch support (see advanced class)

MPLAB® Harmony vs. chipKIT® Platform

- **All chipKIT boards can be used directly with MPLAB X IDE and MPLAB Harmony Framework**
  - MPLAB X IDE and XC32 C/C++
  - MPLAB programmer and debugger tools
- **Note: MPLAB Harmony and chipKIT development environments are completely different**
  - Source code is not directly compatible

Programming Concepts

Programming Model

Remember: The sketch implements two required functions

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Programming Model

Core library adds this code behind the scenes

```
int main(void) {
  init();      // system initialization

  setup();    // sketch initialization

  while (1) {
    loop();  // sketch main loop
  }
  return 0;
}
```

Core Runtime Functions

`pinMode(pin, dir)`

- Sets pin direction and drive type

`digitalRead(pin)`

- Reads the state of a digital pin

`digitalWrite(pin, val)`

- Sets a digital pin to specified state

`delay(ms)`

- Delay for specified number of milliseconds

Example: How to Dim an LED

```
#include "SoftPWMServo.h"

// LED on pin 1 (built-in green LED on FBMini)
const int pinLed = LED_BUILTIN;
//assume pot on analog 2 (pin 4 on FBMini)
const int pinPot = A2;

void setup() {   //nothing needed
}

void loop() {
  int val;
  val = analogRead(pinPot);
  SoftPWMServoPWMWrite(pinLed, val/(1024/256));
  delay(100);
}
```

Hardware Serial

`Serial.begin(baud)`

- Initialize the UART and set the baud rate

`Serial.print(val)`

- Print the specified value to UART

`Serial.read()`

- Read characters from UART

Demo #2:
Serial Monitor

Activity

Send serial data output from chipKIT® board back to PC over USB, then view using the **Arduino®** IDE Serial Monitor

- **helloserial sketch**

Code – Hello World

```
void setup() {
  Serial.begin(9600);  // initialize UART
  delay(5000);
}

void loop() {
  Serial.println();
  Serial.println("Hello World!");
  for (int i = 1; i <= 10; i++) {
    Serial.print(" i = ");
    Serial.println(i, DEC);
  }
  delay(5000);          //wait five seconds
}
```

Summary

- **Display text output to PC over USB using Arduino® IDE Serial Monitor**

Using Libraries

Adding Libraries

- **Use built-in Library Manager to search for and install extra libraries**

- **Some libraries are not available in Library Manager; instead download zip file from GitHub repo or website**

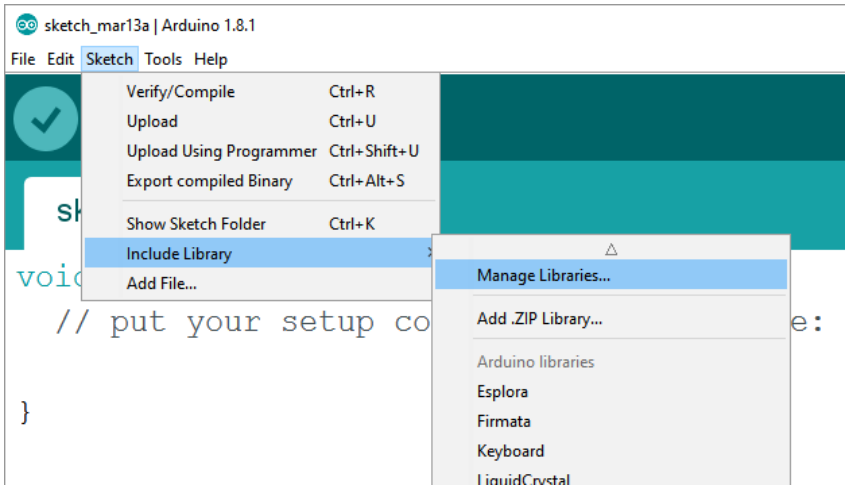
- **Adafruit DotStar Library: We will use this one to control our LED strip**

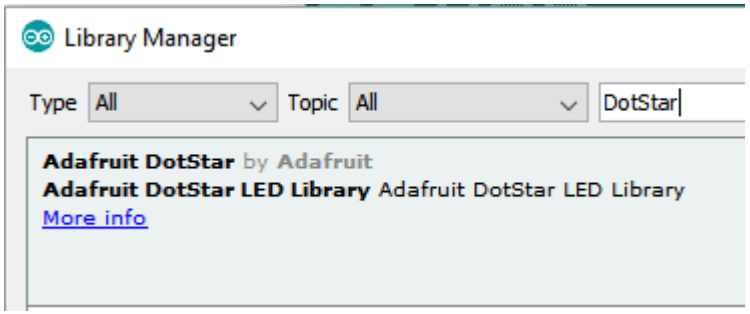
Using Libraries

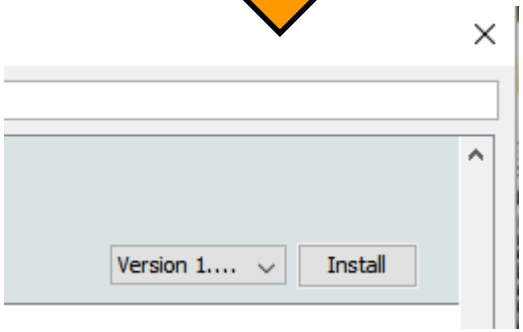
- **Pre-written libraries provide additional functionality**
- **If you add hardware to your design, there are often libraries available to help make use of the hardware**
- **Use** `#include <libraryname>` **to add a library to your sketch**
- **chipKIT® technology core includes highly optimized libraries for PIC32**

Example: Library Manager

Adafruit DotStar Library







How to Debug a Sketch with MPLAB® X

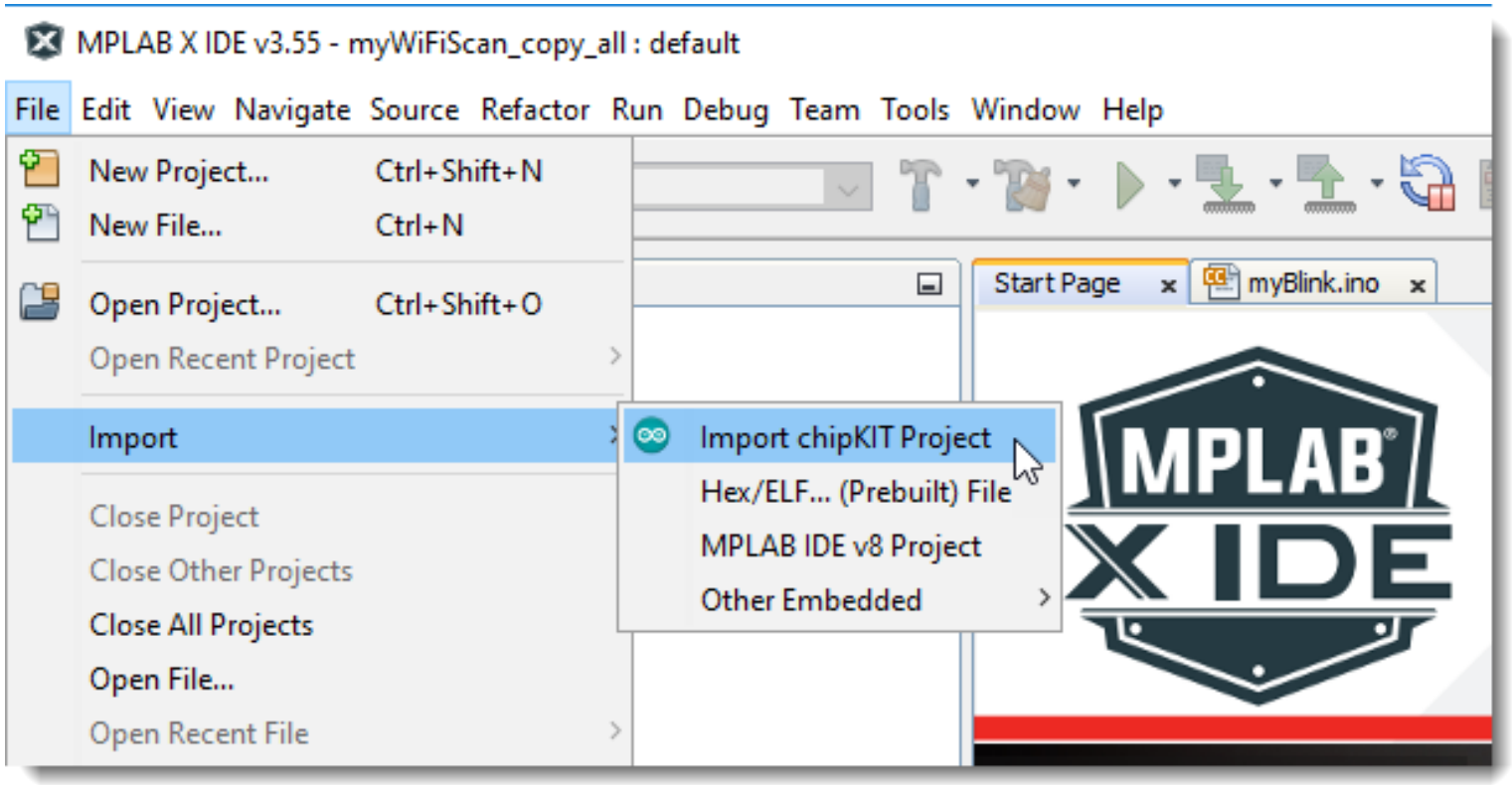
- Introduction to chipKIT® sketch importer for MPLAB X IDE
- Importing a sketch into MPLAB X IDE
- Uploading
- Debugging

chipKIT® Sketch Importer

- Takes existing sketches from Arduino® IDE
- Creates MPLAB® X IDE project, includes all code
- ‘default’ configuration – serial bootloading
- ‘debug’ configuration – hardware debugging

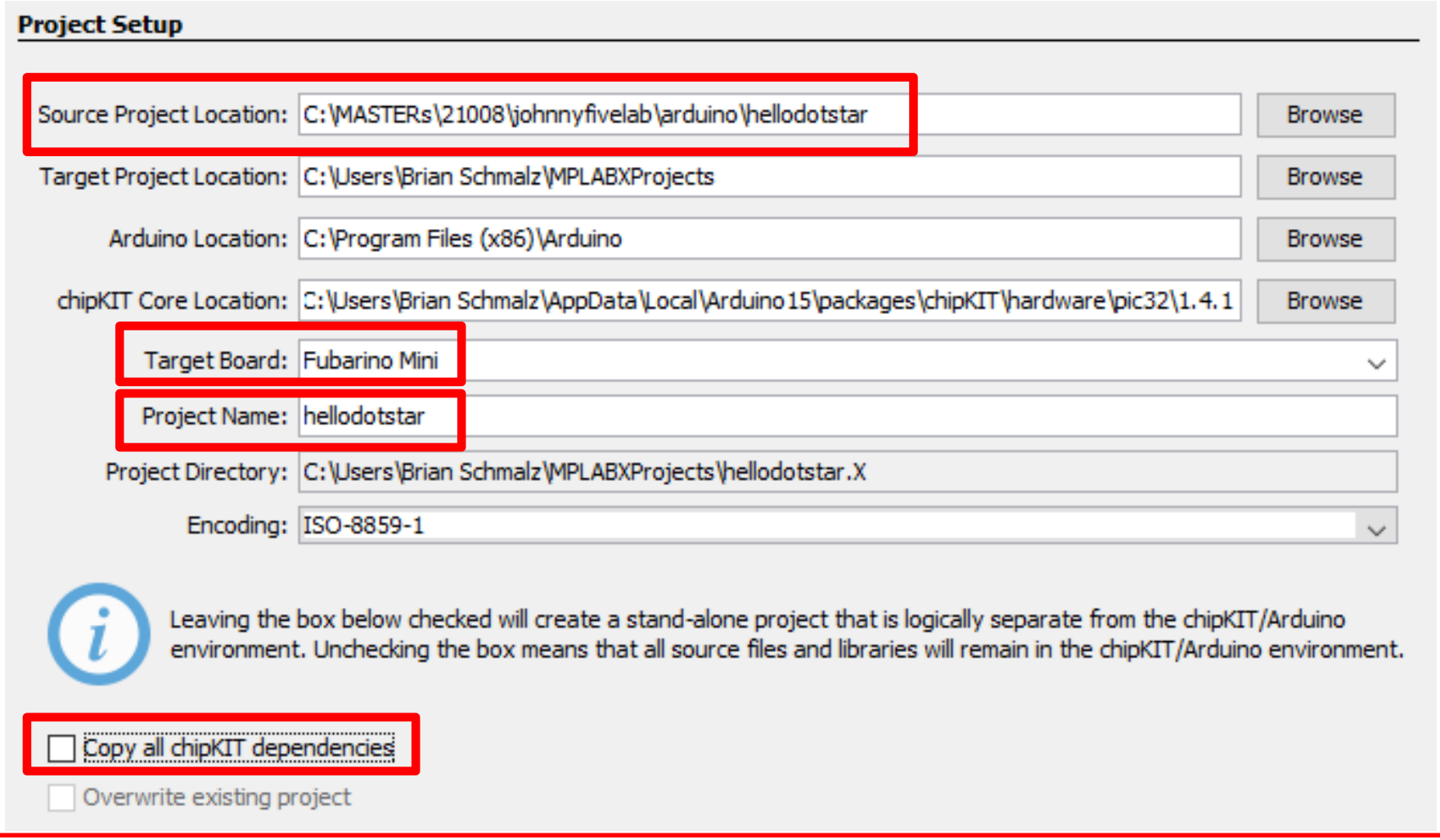
Importing chipKIT® Sketches

- **The plugin installs an Import Wizard under the File menu**

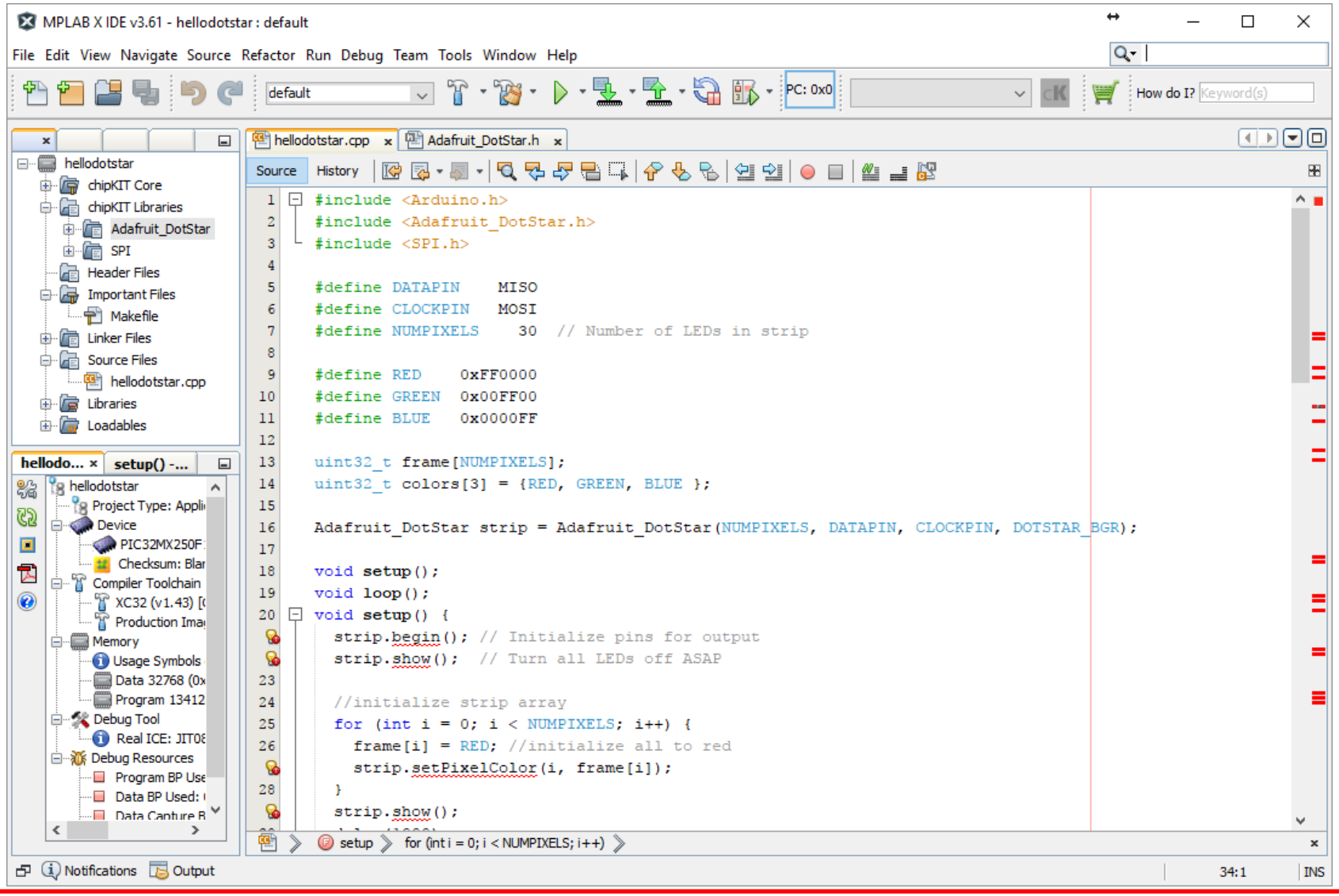


Importing chipKIT® Sketches

- **Specify sketch, Arduino® directory, and target board**



Build/Debug in MPLAB® X IDE



Debugging/Breakpoints

MPLAB X IDE v3.61 - hellodotstar : debug

File Edit View Navigate Source Refactor Run Debug Team Tools Window Help

debug | PC: 0x9D004784 | How do I? | 362.8/1020MB

hellodotstar
- chipKIT Core
- chipKIT Libraries
  - Adafruit_DotStar
  - SPI
- Header Files
- Important Files
  - Makefile
- Linker Files
- Source Files
  - hellodotstar.cpp
- Libraries

hellodotstar.cpp

Source | History

```
55
56    for (int i = 0; i <= NUMPIXELS; i++) {
57      if (i == 0) {
58        frame[i] = 0xFFFFFF;
        strip.setPixelColor(i, frame[i]);
        strip.show();
61      delay(100);
62      }
63      else {
64        frame[i - 1] = 0x000000;
65        frame[i] = 0xFFFFFF;
        strip.setPixelColor(i - 1, frame[i - 1]);
        strip.setPixelColor(i, frame[i]);
        strip.show();
69      delay(100);
70      }
71
72      for (int i = NUMPIXELS; i < 0; i--) {
73        if (i == 30) {
74          frame[i] = 0xFFFFFF;
          strip.setPixelColor(i, frame[i]);
          strip.show();
77        delay(100);
78        }
79        else {
80          frame[i + 1] = 0x000000;
```

loop > for (int i = 0; i <= NUMPIXELS; i++) > if (i == 0) > else > strip.

Variables

Name	Value
frame[15]	0x00000000
frame[16]	0x00000000
frame[17]	0x00000000
frame[18]	0x00000000
frame[19]	0x00FFFFFF
frame[20]	0x00000000
frame[21]	0x00000000
frame[22]	0x00000000
frame[23]	0x00000000
frame[24]	0x00000000
frame[25]	0x00000000

Call Stack

Name
- loop () at c:/users/brian schmalz/mplabxprojects/hell
- main () at c:/users/brian schmalz/mplabxprojects/hellodotstar.x/
- Unknown function () at C:/Users/Brian Schmalz/MPLABXProjects/

Notifications | Output | Watches | hellodotstar (Build, Load, ...) | debugger halted | 66:39/1:5 | INS

Further Information

- **The chipKIT® platform advanced class contains complete information on how to install and use the chipKIT importer plugin, along with many other advanced topics**

Summary

- Provided overview of chipKIT® platform and how it relates to Arduino®
- Learned how to write, compile, load and execute sketches
- Explored basic I/O operations, including digital, analog, serial and LED animation
- Examined debugging sketch from within MPLAB® X IDE
- Experienced wonder and excitement at PIC32 power coupled with Arduino ease of use

Appendix

Resources

- **chipKIT® Platform web site**

  http://chipkit.net

  http://chipkit.net/wiki

  http://chipkit.net/forum

- **Fubarino® Site**

  http://fubarino.org

- **Arduino®**

  https://www.arduino.cc/

chipKIT® Cores and IDEs

- **chipKIT core: How to Install in Arduino® IDE**

  http://chipkit.net/wiki/index.php?title=ChipKIT_core

- **UECIDE**

  http://uecide.org

- **Others:**
  - PlatformIO – http://platformio.org
  - Visual Micro – http://www.visualmicro.com
  - EmbedXcode – http://embedxcode.weebly.com

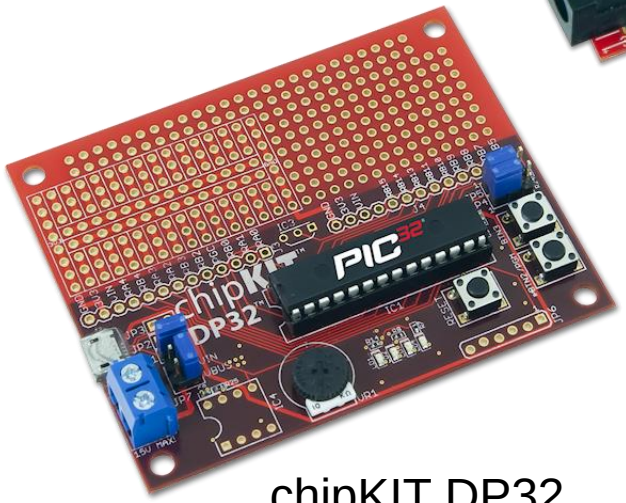
- **MPLAB X® IDE**

  http://www.microchip.com/mplabx

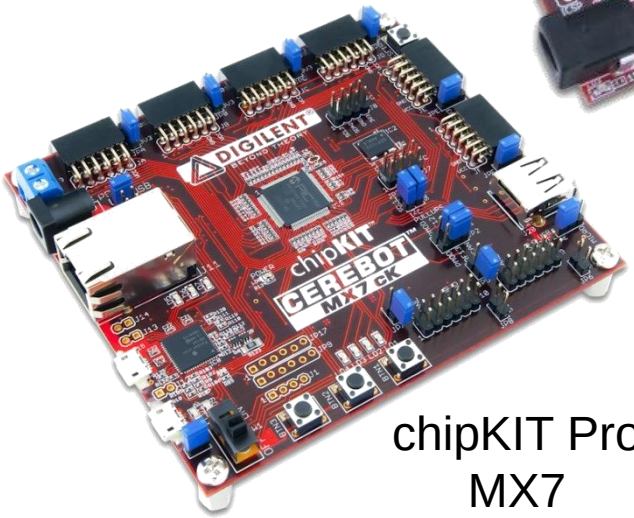
More chipKIT® Boards

chipKIT Wi•FIRE

chipKIT MAX32



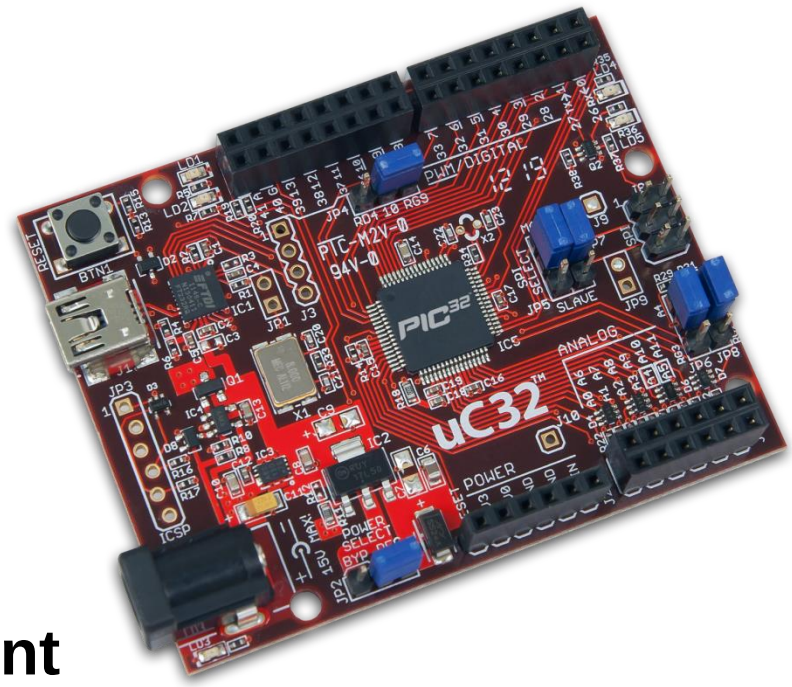
chipKIT DP32



chipKIT Pro MX7

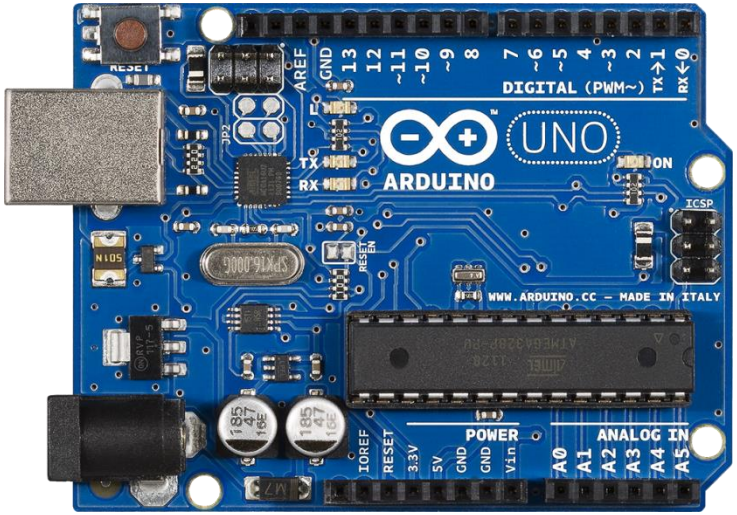
chipKIT® uC32 Board

- **Microchip PIC32MX340F512H**
  - 80 MHz 32-bit MIPS
  - 512K Flash
  - 32K SRAM
- **Arduino® Uno form factor**
- **42 available I/O pins**
- **Two user LEDs**
- **12 analog inputs**
- **75 mA typical operating current**
- **20V input voltage (maximum)**
- **0V to 3.3V analog input voltage range**
- **+/-18 mA DC current per pin**



Arduino® Uno

- ATmega328
- 32K flash, 2K RAM, 1K EEPROM
- 22 I/O pins
- 1 UART
- 1 SPI
- 1 I^2C
- 6 10-bit A/D inputs
- 6 output compare/PWMs

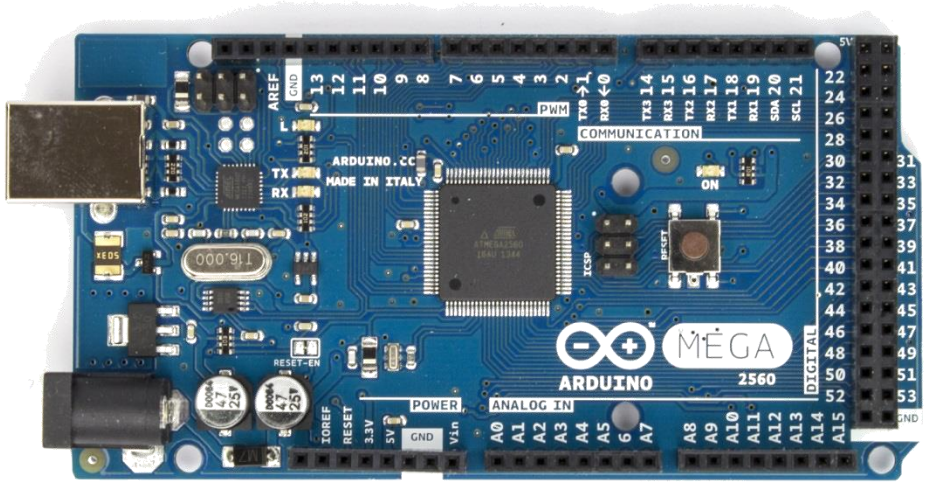


chipKIT® Max32

- PIC32MX795F512L
- 512K flash, 128K RAM
- 83 I/O pins
- 4 UARTs, 1 SPI, 2 I^2Cs
- 16 10-bit A/D inputs
- 5 output compare/PWMs
- USB 2.0 OTG controller
- 10/100 Ethernet MAC
- Dual CAN controllers

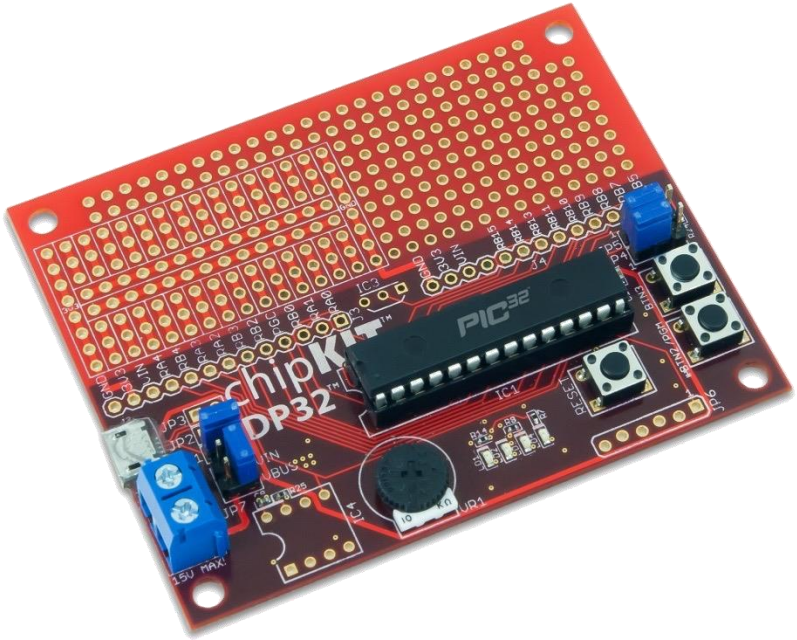
Arduino® Mega 2560

- ATmega2560
- 256K flash, 8K RAM, 4K EEPROM
- 70 I/O pins
- 4 UARTs
- 1 SPI
- 1 I^2C
- 16 10-bit A/D inputs
- 16 output compare/PWMs
- 16 pins left unconnected

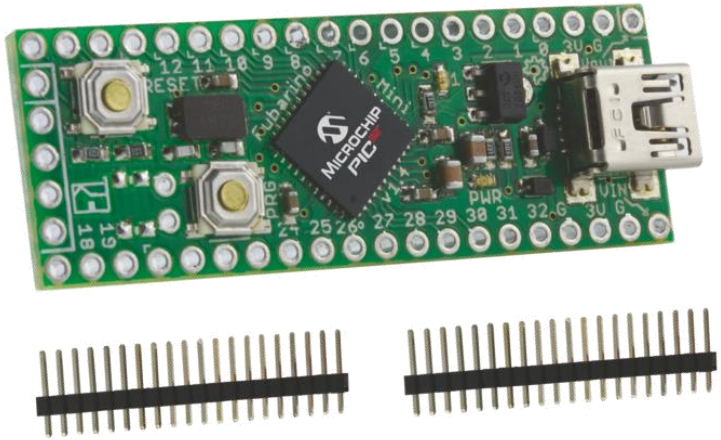


chipKIT® DP32 Board

- **Microchip PIC32MX250F128B**
  - 40/50 MHz 32-bit MIPS
  - 128K Flash
  - 32K SRAM
- **19 available I/O pins**
- **Four LEDs, two push-buttons**
- **9 analog inputs**
- **Potentiometer**
- **Wire-wrap prototype area**
- **Provision for an SPI EEPROM and an analog temperature sensor**
- **Mounting Hole compatible with Hammond 1591XXSSBK project box**



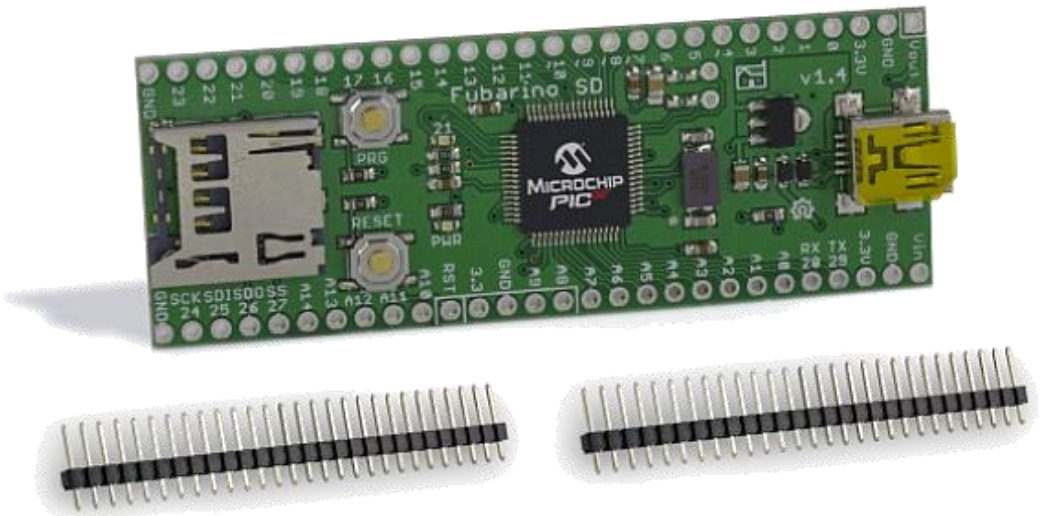
chipKIT® Fubarino® Mini



- **PIC32MX250F128D-50I/ML**
  - 48 MHz operation
  - 128K flash
  - 32K RAM
- **DIP form factor**
- **Max 33 I/O pins (normally 27)**
- **Pads for 32 kHz crystal**
- **Comes with headers (not installed) for easy mounting on a breadboard**
- **Two buttons: RESET for resetting the board, and PRG for getting into bootloader mode and user application use**

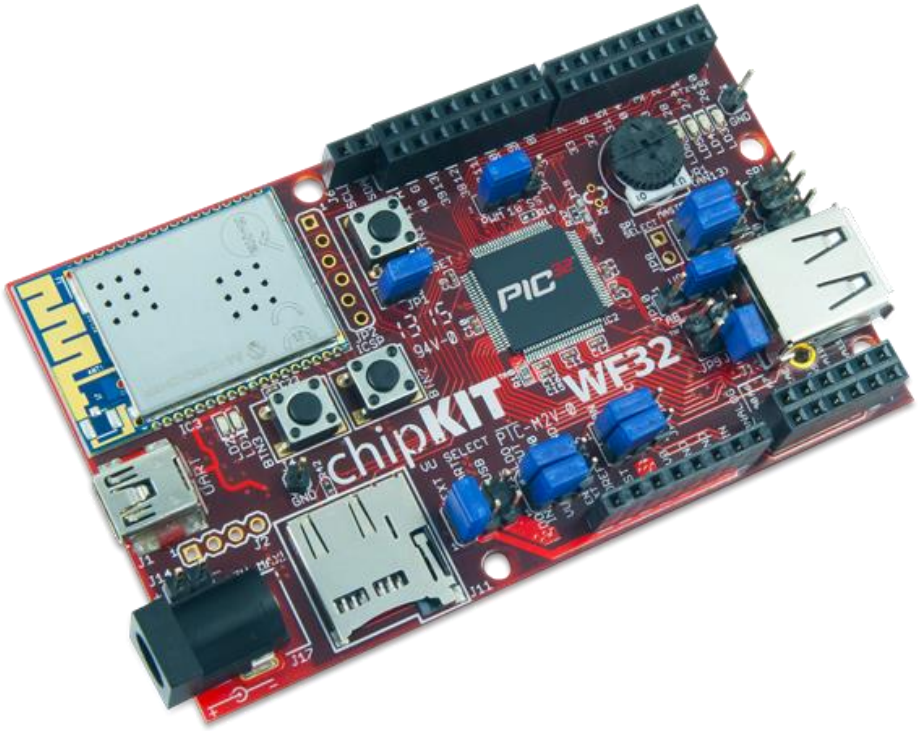
chipKIT® Fubarino® SD

- **PIC32MX795F512H**
  - 80 MHz operation
  - 512K Flash
  - 128K RAM
- **DIP form factor**
- **uSD card connector**
- **45 available I/Os**
- **15 analog inputs**
- **2 UARTs**
- **Comes with headers (not installed) for easy mounting on a breadboard**
- **Two buttons: RESET for resetting the board, and PRG for getting into bootloader mode and user application use**



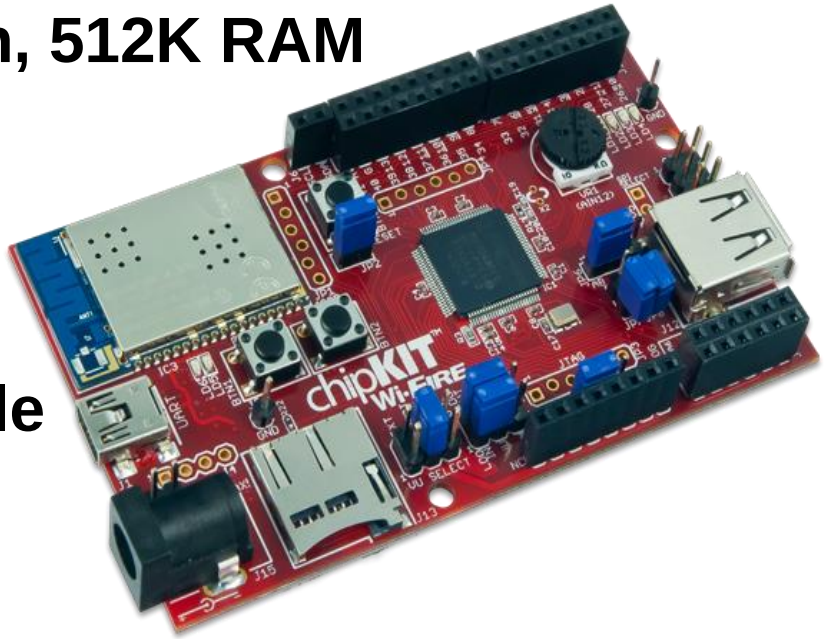
chipKIT® WF32 Board

- **Microchip PIC32MX695F512L**
  - 80 MHz 32-bit MIPS
  - 512K Flash
  - 128K SRAM
  - 802.11g Wi-Fi®, MRF24WG0MA
- **USB Host/Device**
- **uSD card connector**
- **Uno32 shield compatible**
- **42 available I/O pins**
- **Four LEDs, 2 Push buttons**
- **Potentiometer**
- **12 analog inputs**
- **Switching power supply**



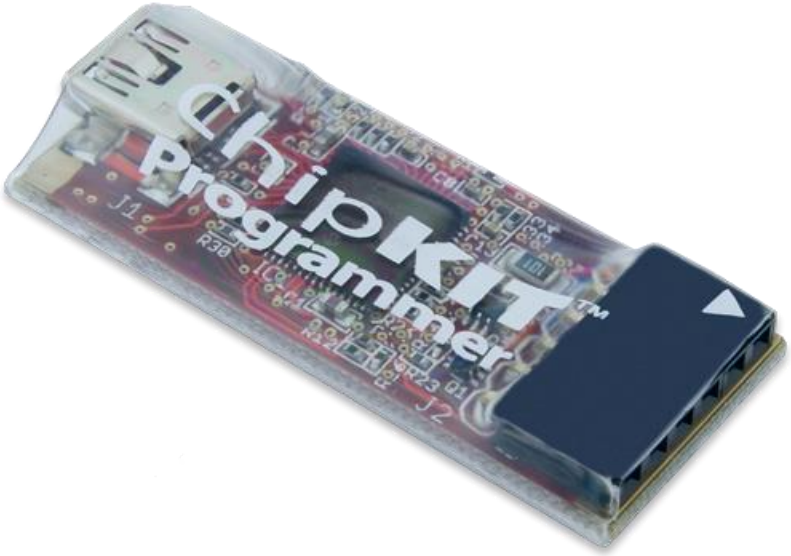
chipKIT® Wi-FIRE® Board

- **PIC32MZ MCU w/ 2 MB Flash, 512K RAM**
- **200 MHz 32-bit MIPS core**
  - Four 64-bit accumulators
  - Floating Point Unit
- **MRF24WG0MA Wi-Fi® 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



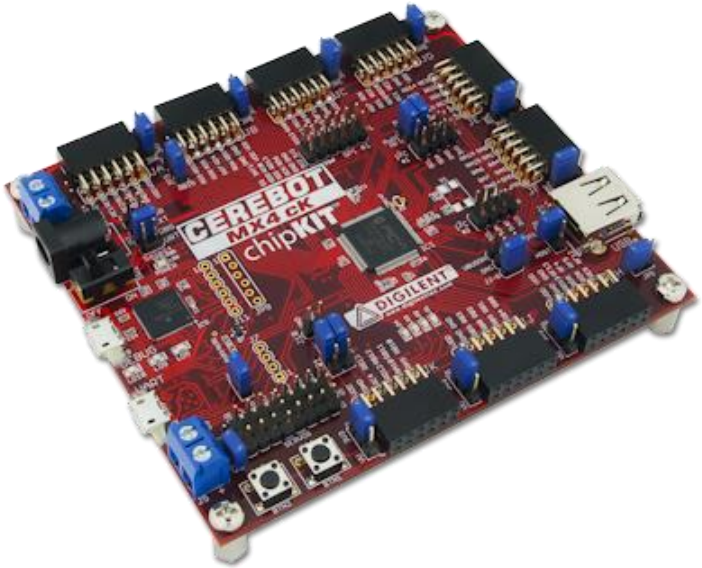
chipKIT® PGM Programmer

- **Provides in-system programming and debugging for Microchip PIC® MCU-based microcontroller boards**
- **Intended for use with chipKIT boards**
- **Works with the MPLAB® IDE and MPLAB IPE**

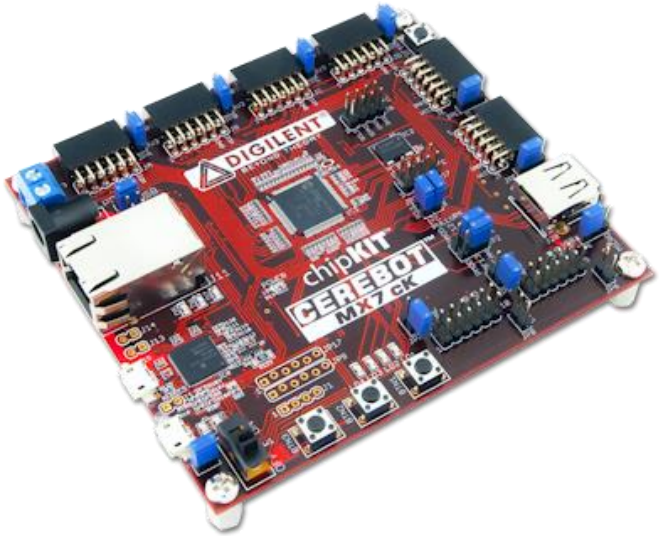


chipKIT® Pro MX4 Board

- **PIC32MX460F512L**
  - 80 MHz operation
  - 512K flash memory
  - 32K RAM memory
- **74 available I/O pins, 9 Pmods, 8 servo connectors, 2 push buttons, 4 LEDs**
- **2 UARTs, 2 SPIs, 2 I^2Cs**
- **5 output compare, PWMs**
- **14 10-bit A/D inputs**
- **MPLAB® IDE compatible licensed debugger**



chipKIT® Pro MX7 Board



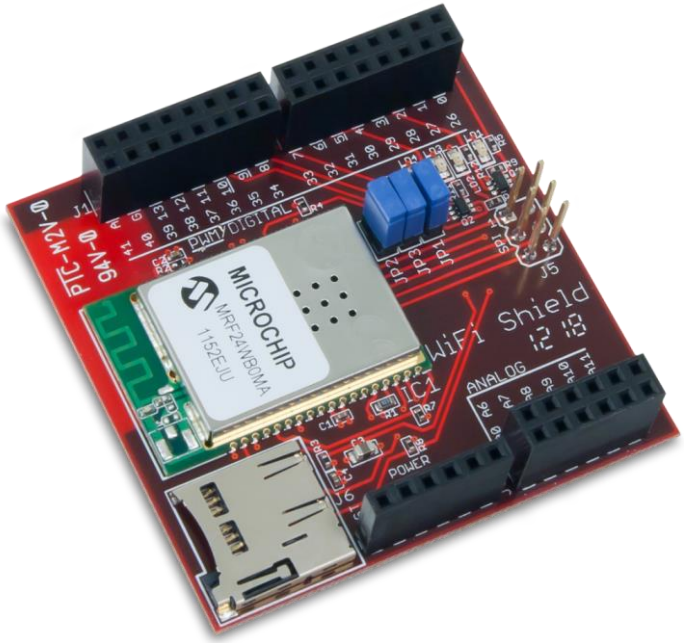
- **PIC32MX795F512L**
  - 80 MHz operation
  - 512K flash
  - 128K RAM
- **64 available I/O pins**
- **6 Pmod connectors**
- **10/100 Ethernet, 2 CANs, USB 2.0 OTG Host/Device**
- **2 UARTs, 3 SPIs, 2 I^2Cs**
- **5 output compare/PWMs**
- **12 10-bit analog inputs**
- **MPLAB® IDE compatible licensed debugger**

chipKIT® Basic I/O Shield

- Uno32 form factor
- 128x32 OLED display
- Four buttons
- Four slide switches
- Eight LEDs
- Four open drain FETs
- I^2C EEPROM
- I^2C temp sensor
- Potentiometer

chipKIT® Wi-Fi® Shield

- IEEE 802.11b-compliant RF transceiver
- Serialized unique MAC address
- 1 and 2 Mbps data rates
- IEEE 802.11b/g/n-compatible
- Integrated PCB antenna
- Range: up to 400m (1300 ft.)
- Radio regulation certification for the FCC, IC, ETSI, and ARIB
- Wi-Fi certified (WFA ID: WFA7150)
- Micro SD card connector
- Four LEDs



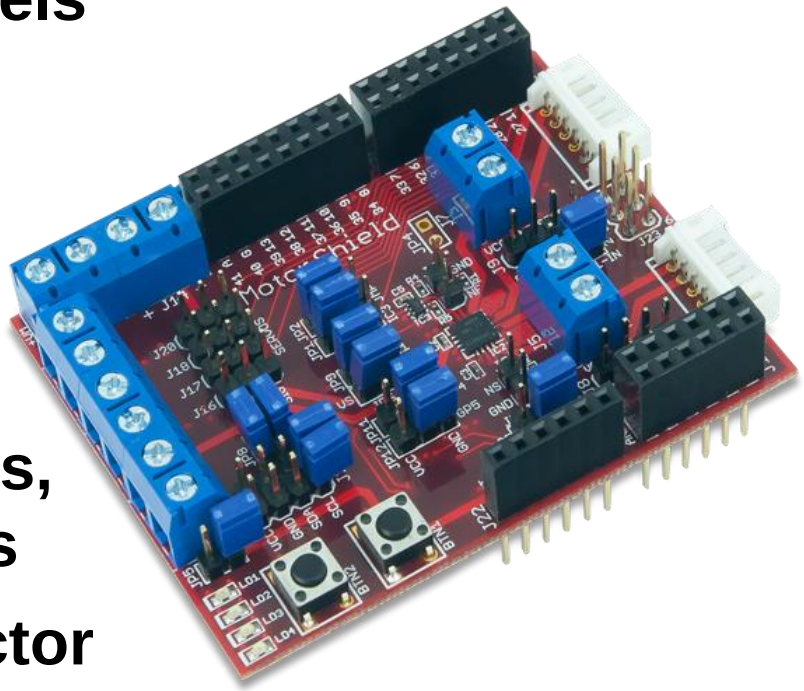
chipKIT® Network Shield

- **Expands I/O on Max32**
- **USB Host and Device connectors**
- **10/100 Ethernet PHY and connector**
- **Dual CAN transceivers and connectors**
- **Dual I^2C connectors**
- **I^2C EEPROM**
- **32 kHz oscillator**

The Network Shield in combination with the Max32 provides access to all of the features of the PIC32MX795F512L

chipKIT® Motor Shield

- Two H-bridge DC motor channels
- Quadrature encoder feedback inputs
- Four RC servo outputs
- Four low side N-FET outputs
- I^2C I/O expander with four LEDs, two push buttons, two jumpers
- Fits Uno32 style shield connector footprint



LEGAL NOTICE

SOFTWARE:

You may use Microchip software exclusively with Microchip products. Further, use of Microchip software is subject to the copyright notices, disclaimers, and any license terms accompanying such software, whether set forth at the install of each program or posted in a header or text file.

Notwithstanding the above, certain components of software offered by Microchip and 3rd parties may be covered by "open source" software licenses – which include licenses that require that the distributor make the software available in source code format. To the extent required by such open source software licenses, the terms of such license will govern.

NOTICE & DISCLAIMER:

These materials and accompanying information (including, for example, any software, and references to 3rd party companies and 3rd party websites) are for informational purposes only and provided "AS IS." Microchip assumes no responsibility for statements made by 3rd party companies, or materials or information that such 3rd parties may provide.

MICROCHIP DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, OR STATUTORY, INCLUDING ANY IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY DIRECT OR INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND RELATED TO THESE MATERIALS OR ACCOMPANYING INFORMATION PROVIDED TO YOU BY MICROCHIP OR OTHER THIRD PARTIES, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBLITY OF SUCH DAMAGES OR THE DAMAGES ARE FORESEEABLE. PLEASE BE AWARE THAT IMPLEMENTATION OF INTELLECTUAL PROPERTY PRESENTED HERE MAY REQUIRE A LICENSE FROM THIRD PARTIES.

TRADEMARKS:

The Microchip name and logo, the Microchip logo, AnyRate, AVR, AVR logo, AVR Freaks, BeaconThings, BitCloud, CryptoMemory, CryptoRF, dsPIC, FlashFlex, flexPWR, Heldo, JukeBlox, KEELOQ, KEELOQ logo, Kleer, LANCheck, LINK MD, maXStylus, maXTouch, MediaLB, megaAVR, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, Prochip Designer, QTouch, RightTouch, SAM-BA, SpyNIC, SST, SST Logo, SuperFlash, tinyAVR, UNI/O, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

ClockWorks, The Embedded Control Solutions Company, EtherSynch, Hyper Speed Control, HyperLight Load, IntelliMOS, mTouch, Precision Edge, and Quiet-Wire are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, BodyCom, chipKIT, chipKIT logo, CodeGuard, CryptoAuthentication, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, EtherGREEN, In-Circuit Serial Programming, ICSP, Inter-Chip Connectivity, JitterBlocker, KleerNet, KleerNet logo, Mindi, MiWi, motorBench, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, PureSilicon, QMatrix, RightTouch logo, REAL ICE, Ripple Blocker, SAM-ICE, Serial Quad I/O, SMART-I.S., SQI, SuperSwitcher, SuperSwitcher II, Total Endurance, TSHARC, USBCheck, VariSense, ViewSpan, WiperLock, Wireless DNA, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

Silicon Storage Technology is a registered trademark of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

All other trademarks mentioned herein are property of their respective companies.