# Programming tinyAVR 0- and 1-series microcontrollers with Arduino

**Jason Goodman, Wheaton College**

## Introduction

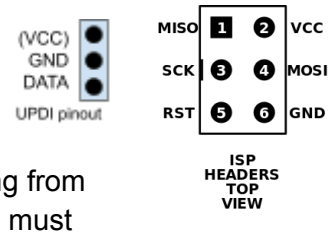
The new 0- and 1-series ATTiny microcontrollers from AVR are faster, more flexible, cheaper, provide more storage, and are easier to program than previous ATTinies. Let's learn how to program them!

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 32K (2K) | | | | | | ATtiny3216 $1.03/NA | | ATtiny3217 $1.01 |
| 16K (1K/*2K) | | | ATtiny1604 $0.66 | ATtiny1614* $0.75 | ATtiny1606 $0.90/NA | ATtiny1616* $0.94/$0.77 | ATtiny1607 $0.88 | ATtiny1617* $0.97 |
| 8K (512) | | | ATtiny804 $0.64 | ATtiny814 $0.66 | ATtiny806 $0.83/NA | ATtiny816 $0.90/$0.71 | ATtiny807 $0.83 | ATtiny817 $0.92 |
| 4K (256) | ATtiny402 $0.41 | ATtiny412 $0.46 | ATtiny404 $0.52 | ATtiny414 $0.56 | ATtiny406 $0.77/$0.62 | ATtiny416 $0.85/$0.63 | | ATtiny417 $0.81 |
| 2K (128) | ATtiny202 $0.41 | ATtiny212 £0.43 | ATtiny204 $0.52 | ATtiny214 $0.54 | | | | |
| FLASH (RAM) | 8 pins SOIC 5 I/O lines | | 14 pins SOIC 11 I/O lines | | 20 pins SOIC/QFN 17 I/O lines | | 24 pins QFN 21 I/O lines | |

This tutorial uses the Arduino platform, and Spencer Konde's megaTinyCore Arduino core. It should work on Mac, PC, or Linux.

## UPDI Programming - hardware

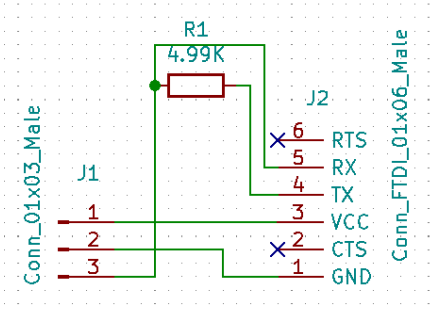
Previous ATTinies used the ISP (In-System Programming) method, which requires six wires . These new ATTiny microcontrollers use UPDI (Unified Program and Debug Interface), which requires only two: Data and Ground. However, it can be convenient to add an optional power pin, VCC, which allows you to power the board you're programming from the device that's sending the program. Without this pin, the programmee must have its own power source.







You'll send the data from the computer through a standard FTDI USB-to-serial cable. We need a converter to adapt the pins from FTDI to UPDI. Designs for this converter board can be found at the Fab Academy website (board, components, traces, interior) or on my Github. You can also easily build this on a breadboard. A basic schematic is at right: it's just a 5K resistor plus a few wires.



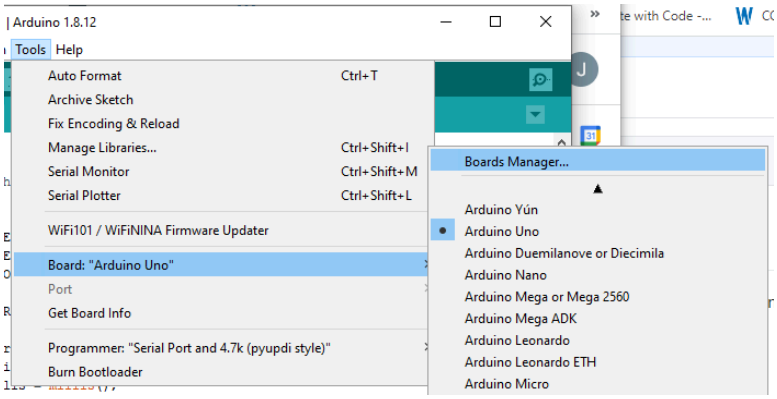

## Install Arduino

This tutorial relies on the Arduino IDE (Integrated Development Environment). It is available for Mac, PC, and Linux. Download and install the latest version from

https://www.arduino.cc/en/software . If you already have Arduino installed, check to make sure you're using the latest version: the steps below will not work on an older build.

I recommend using Arduino for all your ATTiny development, unless you have a pressing reason not to. Compiling software, installing it on the Tiny, and upgrades and support are much easier, and there's a ton of online resources to help you learn.

## Install megaTinyCore




"Arduino" is both a development environment that runs on your computer, and a "core" -- a set of toolkit code that runs on your microcontroller. Each microcontroller needs its own version of the Arduino core. For the AVR 0- and 1-series, that's megaTinyCore by Spence Konde

(https://github.com/SpenceKonde/megaTinyCore )

Instructions on installing megaTinyCore are here:
https://github.com/SpenceKonde/megaTinyCore/blob/master/Installation.md

1) Open File / Preferences in Arduino IDE
2) Add http://drazzy.com/package_drazzy.com_index.json to the list of "Additional Board Manager URLs"
3) In the Tools / Board menu, choose "Boards Manager…"
4) Find megaTinyCore and click "Install". This may take a while.
5) Close the Board Manager.
6) In the Tools / Board menu, choose "megaTinyCore" and pick the particular microcontroller you're using.
7) In the Tools / Programmer menu, choose "SerialUPDI SLOW - 57600 baud, any platform, any adapter)". This option only appears when you have chosen a megaTinyCore microcontroller.

## Connections

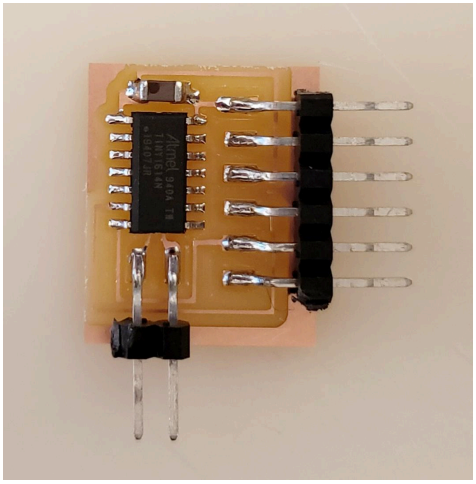
Plug your "SAMD UPDI Breakaway" board you built during Electronics Production week ago into your computer, and attach the FTDI-to-UPDI adapter section. If you don't have a working one, you can use a commercial USB-to-FTDI cable with a FTDI-to-UPDI converter attached to that. Make sure the converter is plugged in the right way! Now connect the UPDI converter to the programming pins on your 1-series microcontroller board. Once again, make sure the pins match power, ground, and signal.

In the Arduino app, choose Tools / Port, and select the serial port corresponding to your FTDI cable (should be only one option on most modern Macs and PCs).

## Programming


Load up an Arduino sketch (for example, Neil Gershenfeld's echo.ino sketch for the ATTiny 1614 (which should work on other 0- and 1-series chips): http://academy.cba.mit.edu/classes/embedded_programming/t1614/hello.t1614.echo.ino ). Click "upload using programmer" on the sketch window. Hopefully the code should be compiled with no errors, and then sent to your board.

## Serial Port Communications



Your microcontroller may want to talk to your PC directly (for example, that's what echo.ino is supposed to do.) You can't use UPDI for that, it's for programming only! You must design and build a separate FTDI connection for your board. For example, in the ATTiny1614 board at right, the bottom two pins are for UPDI programming to set up the board; the six pins on the right are for FTDI serial communication with Arduino once the board is operating.

To talk with the board, choose Tools / Serial Monitor in Arduino.

## For more information:

There is a ton of information about general Arduino programming on the Web. Most of these are for commercially-made Arduino boards, but the programming is mostly the same. I recommend the tutorials at Adafruit and Sparkfun.

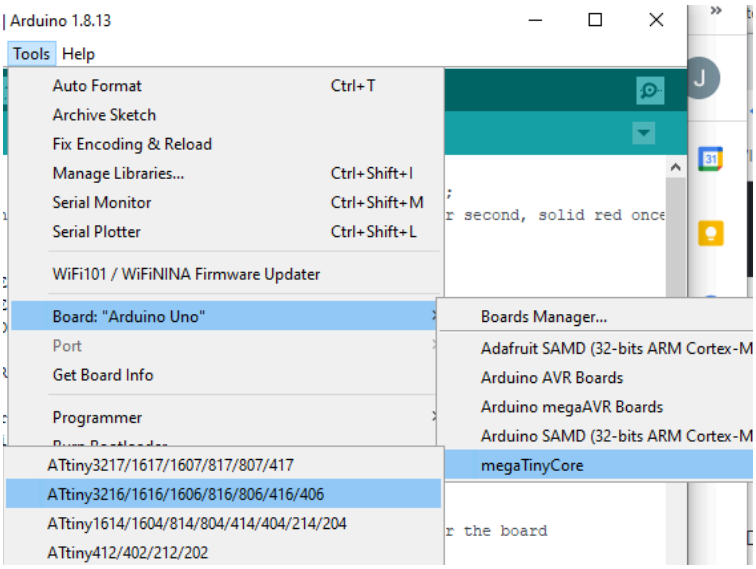
For info about the ATTiny 0- and 1-series chips, see Wikipedia.

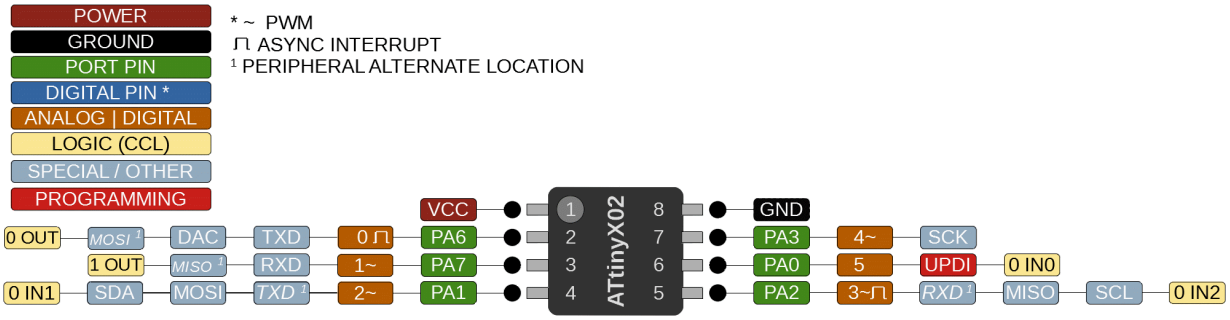
**A guide to ATTiny part numbers:**

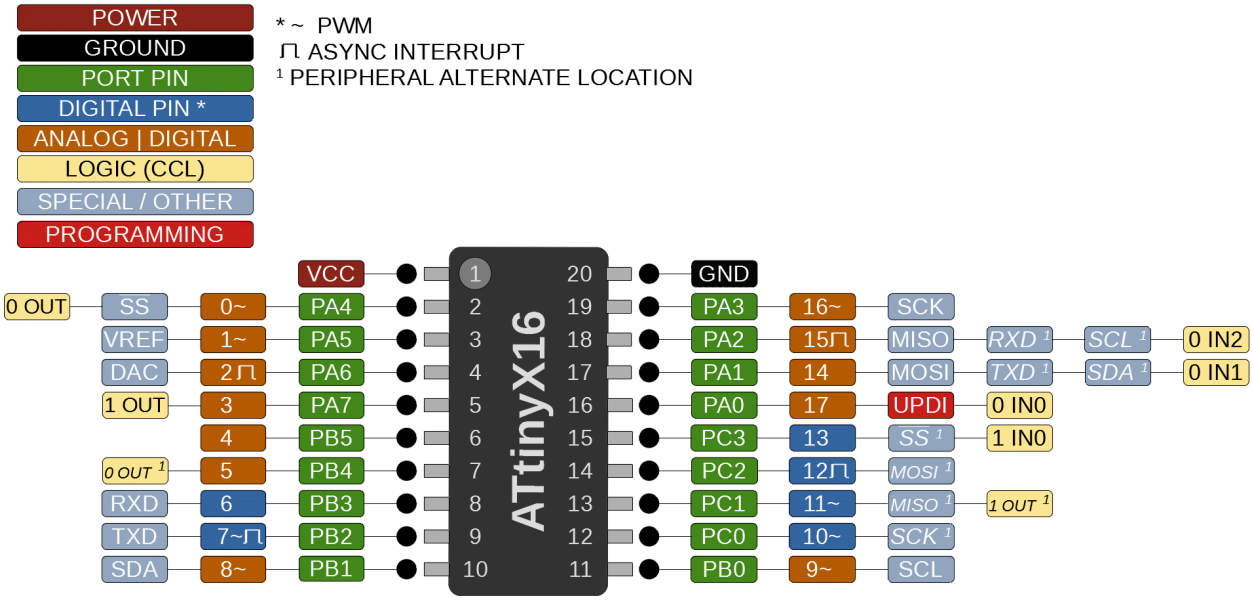
`ATTinyXXYZ`

XX: memory size (2, 4, 8, 16, 32 kb)
Y: Series. 0-series generally has fewer features than 1-series, but the details vary from chip to chip.
Z: Total number of pins. 2: 8 pins. 4: 14 pins. 6: 20 pins. 7: 20 pins.

For more information on individual chips, their features, and how to use them in Arduino, see https://github.com/SpenceKonde/megaTinyCore/tree/master/megaavr/extras. Example pinouts:


Arduino 1.8.13
Tools
Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Board: "Arduino Uno"
Port
Get Board Info
Programmer
Boards Manager...
Adafruit SAMD (32-bits ARM Cortex-M
Arduino AVR Boards
Arduino megaAVR Boards
Arduino SAMD (32-bits ARM Cortex-M
megaTinyCore
ATtiny3217/1617/1607/817/807/417
ATtiny3216/1616/1606/816/806/416/406
ATtiny1614/1604/814/804/414/404/214/204
ATtiny412/402/212/202
r second, solid red once
r the board


POWER
GROUND
PORT PIN
DIGITAL PIN *
ANALOG | DIGITAL
LOGIC (CCL)
SPECIAL / OTHER
PROGRAMMING
* ~ PWM
⎍ ASYNC INTERRUPT
[1] PERIPHERAL ALTERNATE LOCATION
ATtinyX02
VCC
GND
0 OUT
MOSI [1]
DAC
TXD
0 ⎍
PA6
4~
SCK
1 OUT
MISO [1]
RXD
1~
PA7
5
UPDI
0 IN0
0 IN1
SDA
MOSI
TXD [1]
2~
PA1
3~⎍
RXD [1]
MISO
SCL
0 IN2


POWER
GROUND
PORT PIN
DIGITAL PIN *
ANALOG | DIGITAL
LOGIC (CCL)
SPECIAL / OTHER
PROGRAMMING
* ~ PWM
⎍ ASYNC INTERRUPT
[1] PERIPHERAL ALTERNATE LOCATION
ATtinyX16
VCC
GND
0 OUT
SS
0~
PA4
16~
SCK
VREF
1~
PA5
15⎍
MISO
RXD [1]
SCL [1]
0 IN2
DAC
2⎍
PA6
14
MOSI
TXD [1]
SDA [1]
0 IN1
1 OUT
3
PA7
17
UPDI
0 IN0
4
PB5
13
SS [1]
1 IN0
0 OUT [1]
5
PB4
12⎍
MOSI [1]
RXD
6
PB3
11~
MISO [1]
1 OUT [1]
TXD
7~⎍
PB2
10~
SCK [1]
SDA
8~
PB1
9~
SCL