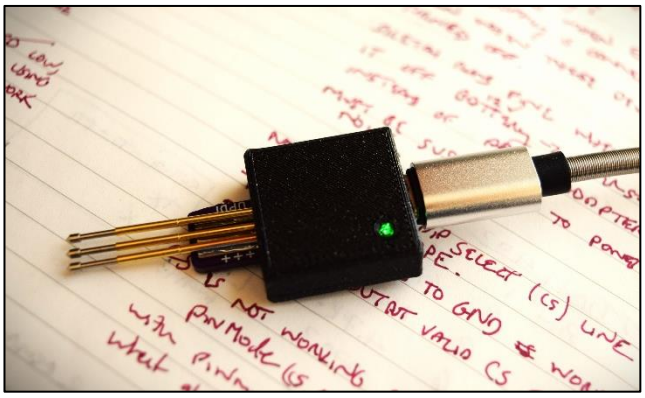
Build yourself a UPDI programmer.

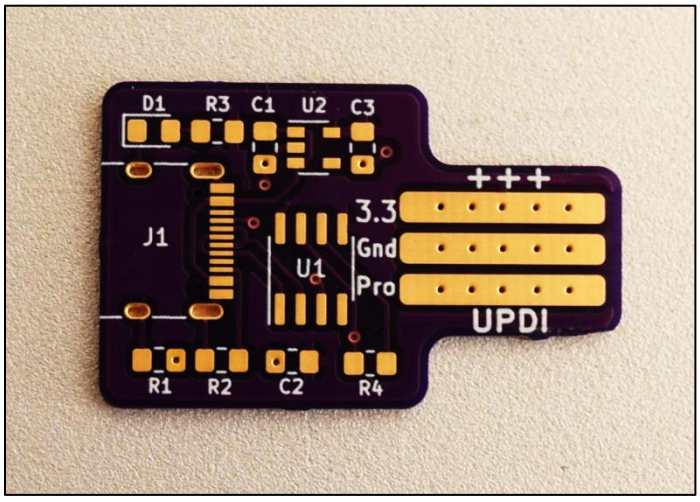


by Bruce Hall, W8BH

Introduction

The newest microcontrollers from MicroChip/Atmel eschew the familiar 6-pin ICSP interface for something new: the UPDI (Unified Program and Debug Interface). With few exceptions, you will need a UPDI programmer if you want to program newer tinyAVR, megaAVR, or AVR-Dx chips. The bad news is that the interface is proprietary and unpublished. The good news is that kind and generous souls have reverse-engineered the protocol. Even better, all you need is a USB-to-Serial adapter and a resistor. UPDI programming is easy, fast, and inexpensive.

There are many UPDI programmer examples and tutorials on the web. You can build one from a spare Nano board. If you want to make your own from a USB-to-Serial adapter, I suggest starting with the excellent SerialUPDI summary by Spence Konde.

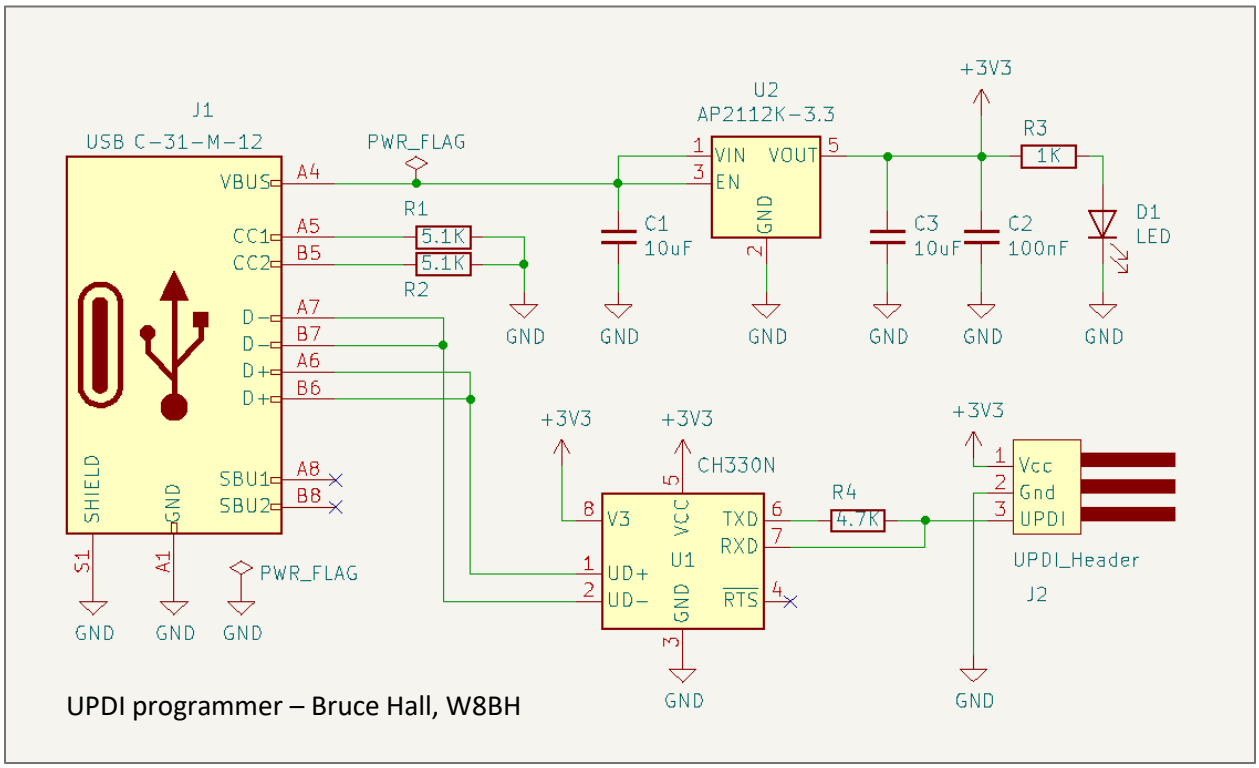


I chose to make mine from scratch. The result is a small dongle with three pogo-pins. To use it, press the pogo pins onto an unpopulated 3-pin PCB header (or machine pin socket). It's quick and easy, with no programming wires to connect and disconnect.

Read on if you feel comfortable with SMD soldering and want to build one yourself.

Circuit Description

Look at the schematic below. The world is moving to USB-C, so I chose a USB-C jack for the input. USB-C is capable of suppling power at different voltage and current levels, depending on the needs of the device. Two 5.1K resistors on the USB control lines tell the upstream "provider" to supply 5 volts. You could draw up to 3A in this mode, but few if any computers are able to supply that much current. The downstream device can safely use up to 500mA.

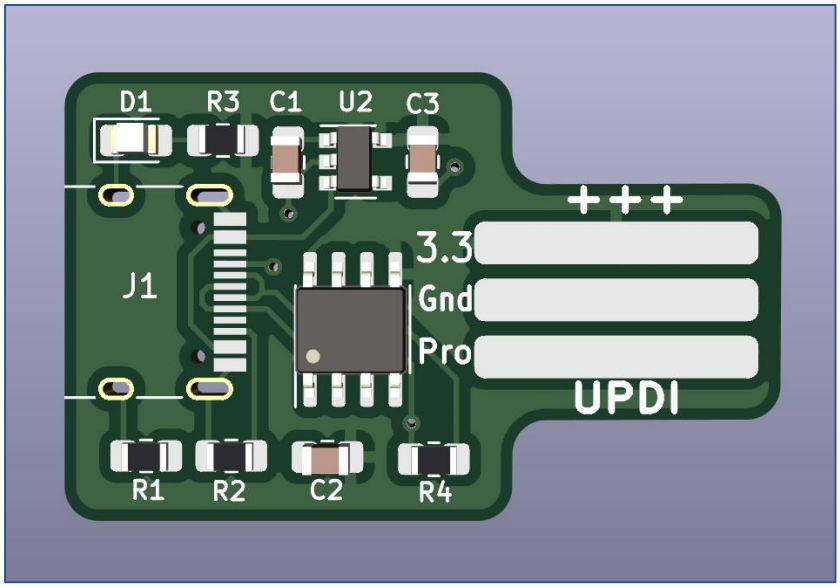


The heart of the programmer is the CH340N, a very cheap and reliable USB-to-Serial converter chip from China. You can't buy it at Digikey or Mouser, so take your chances at eBay and AliExpress. I ordered mine from LCSC Electronics (10 for $4.80 + shipping). It is an 8-pin small outline package (SOP) part that is easy to solder. This part requires virtually no external components: apply a USB signal on one side and get serial data on the other. Note that the CH330N (now obsolete) and CH340N are interchangeable.

For the power supply I chose the AP2112K-3.3 regulator chip, flanked by capacitors on the input and output. This converts 5 volts from the USB bus to 3.3V for your project. The regulator is rated for 600mA output. A power-on LED completes the supply.

The final part of the programmer is the UPDI interface itself. The TX and RX pins from the CH340N must be combined to form a single UPDI programming pin. Spence Konde's summary shows several methods in which the data pins can be successfully combined. I chose to use a single 4.7K resistor, R4. A Schottky diode will also work.

Assembly



The PCB is a 20 x 32mm two-layer board, with long pads at one end for soldering the pogo-pins. Pogo-pin type "E2" provides a conical point that works well with standard PCB holes.

Attach the USB jack first, which has small pins that I find easiest to solder under a magnifying glass or microscope. Next, solder the AP2112 and CH340 ICs. Finally, add the eight remaining 0805-sized SMD components, working from left to right.

Soldering the pogo-pins is the most challenging step, especially if you are trying to align the pins manually. If you are in a hurry, align the pins over their pads and secure them together with a small block of Styrofoam. However, I suggest creating a jig: A 3D-printed pin alignment block will keep your pins straight and correctly spaced.

Solder the center pin first since access to its pad is blocked after the outer pins are placed. Leave 2mm at the top of the pad so that you can tack solder the top of the pin. Check alignment before fully soldering the center pin. Next, align the two outer pins over their pads, parallel to the center pin, and solder them in place. The 3D-printed block can melt, so don't excessively heat the pins.

If you don't want to use pogo pins, you can use a 3-pin male header (or 3-wire cable, or female header) instead. It's up to you.

Operation

Connect a USB-C cable from the jack to the USB port on your PC. The LED on the programming board should illuminate, indicating 3.3V power. If you are using a Windows PC, open the Device Manager. Under Ports (COM & LPT), you should see "USB-SERIAL CH340 (COMx)", where x is the com port number.

Arduino 2.0 is not yet supported, so use version 1.8.19 or less. In the Arduino IDE, install the appropriate board package for the device you are programming. For example, the I use Spence Konde's board packages for programming the ATtiny402:

1. Go to File > Preferences > Additional Boards Manager URLs
2. Add the following line URL: http://drazzy.com/package_drazzy.com_index.json
3. Go to Tools > Board > Board Manager
4. Find "megaTinyCore" by Spence Konde and install it.
5. Go to Tools > Board > Board > megaTinyCore. Select "ATtiny 402/412/202/212"
6. Go to Tools > Chip and select "ATtiny402".
7. Go to Tools > Port and select your programmer's serial port
8. Go to Tools > Programmer and select "SerialUPDI – 230400 baud"

That's it. If you get the error message "UPDI initialization failed", it means there is a communication problem between the CH340N and the MCU. Make sure there are no extra components on the UPDI data line. For example, the Adafruit ATtiny816 breakout board has an unexpected 10K pullup resistor on the UPDI line. I could not program the breakout board until I removed this pullup resistor.

Resources

- PCB Gerber Files
- Case & Pin Alignment Block STL files

Last updated: 7 Mar 2023