

UNO as UPDI programmer

Introduction

UNO_UPDI_Programmer_V1 is an Arduino project that can be uploaded to a UNO. Target devices for the programmer are the ATtiny806 and 1606. These have been chosen because they have hardware UART and I2C and are available in a SOIC package which is suitable for hand soldering.

The programming interface consists of

- +/- 5V power

- UNO pin A0 to the target reset pin

- UNO pin A4 to the target Rx port

- UNO pin A5 to the target Tx port

Connection of pins 4 and 5 is optional but provides a basic method to testing the target code.

It enables

- ATtiny806/1606 devices to be programmed with hex and text files

- The UNO to be used to connect the target UART port to its own for test purposes

Note: This UART link works at 9600Baud. Having completed tests the user may wish to increase the baud rate.

Two test projects are also supplied which were developed using Atmel Studio 7

- "Text_reader" which reads the strings programmed to flash

- "Floating_point_arithmetic" which does some simple arithmetic

The purpose of these is to enable the operation of the programmer to be checked out fairly easily.

Hex files are supplied so there is no need to have Studio 7.

A terminal program is however required. My favourite can be downloaded from <https://sites.google.com/site/terminalbpp/>. But take care to download the version 20130820, other versions may have an issue with the "scroll" button. Settings for the terminal program are: Baud rate: 28800, 8 data bits, no parity, 1 stop bit and no handshaking.

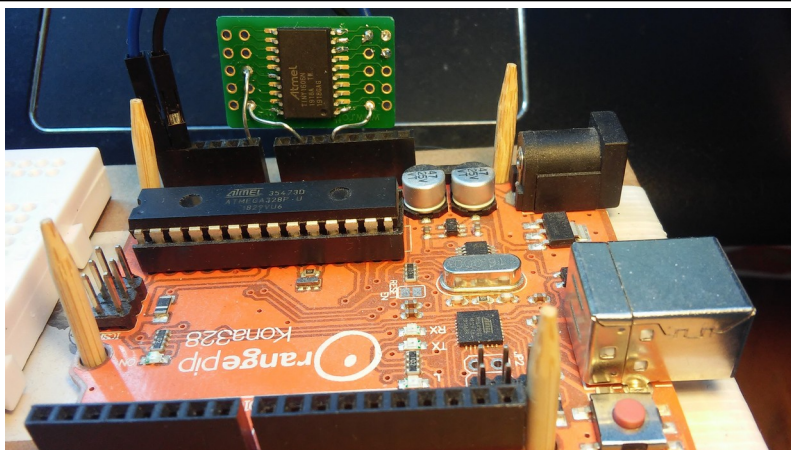


Photo of the programmer development setup.

The tinned copper wires connect power and the target reset pin.

The blue and black wires connect the target Rx/Tx UART pins

The target is mounted on a pcb available from Radio Spares as part no 728-8881

Running the programmer

Open the Arduino project and upload it to a UNO.

Connect a target device to the UNO as follows:

UNO pin A5	Target pin 9
UNO pin A4	Target in 8
UNO pin A0	Target pin 16
UNO 0V	Target pin 20
UNO 5V	Target pin 1

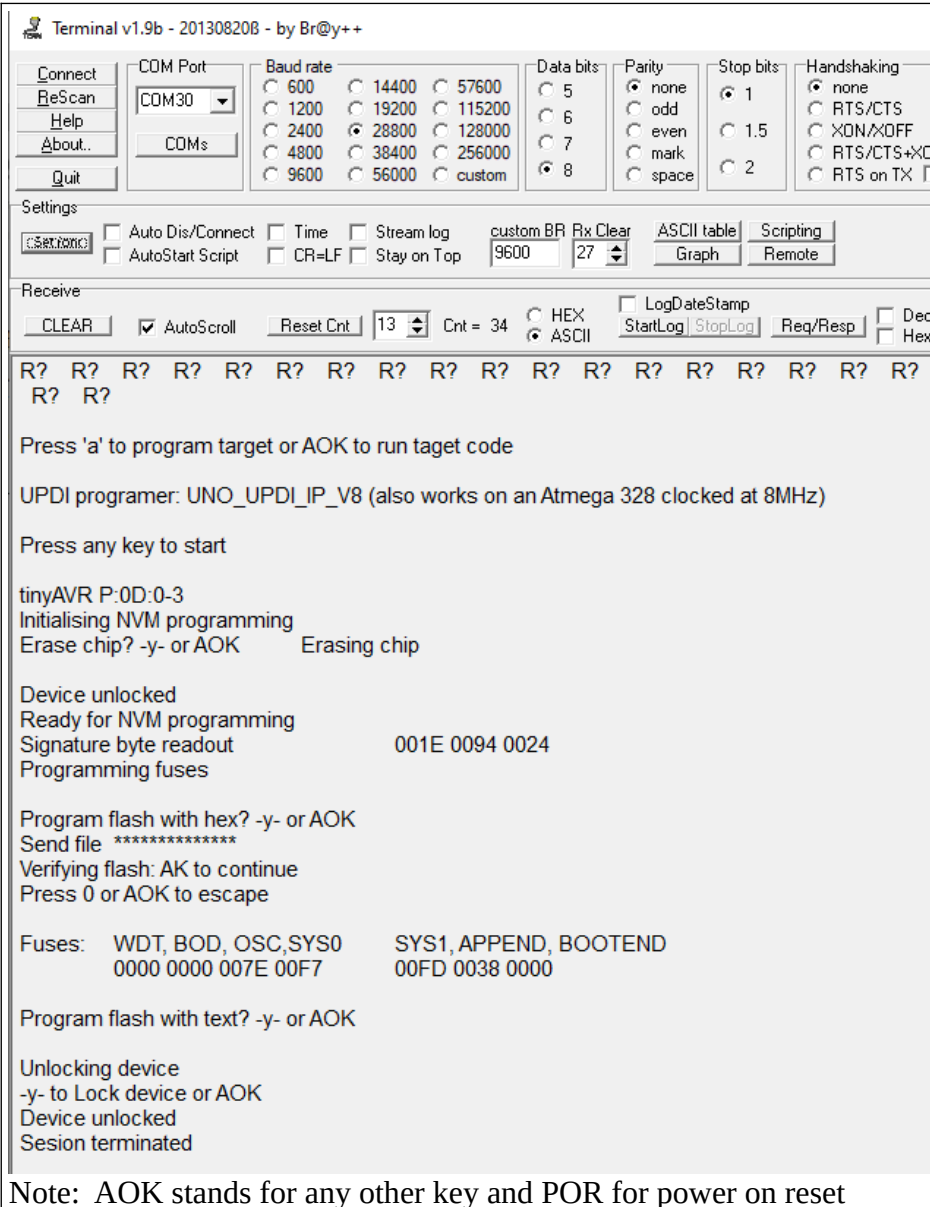
Open the terminal program:

Set it to 28800baud, 8 data bits, no parity, 1 stop bit and no handshaking

Click on the Rescan button to select the correct port
and then click on the connect button.

User prompt “R? R? R?.....” should be generated

A typical terminal session is shown below:

	Typical terminal session At the user prompt the user pressed “raa” to get started “y” to erase memory “y” to upload hex file The hex file was then uploaded and the user pressed “aa” to bypass verification “a” for no text file “a” to leave the lock bits unset and terminate the session It is also possible at this point to run the target application without changing the terminal baud rate. If a text file is to be downloaded the user will be requested to reduce the baud rate to 4800.
<p>Terminal v1.9b - 20130820B - by Br@y++</p> <p>Connect ReScan Help About... Quit</p> <p>COM Port: COM30</p> <p>Baud rate: 600 1200 2400 4800 9600 14400 28800 57600 115200 128000 256000 custom</p> <p>Data bits: 5 6 7 8</p> <p>Parity: none odd even mark space</p> <p>Stop bits: 1 1.5 2</p> <p>Handshaking: none RTS/CTS XON/XOFF RTS/CTS+XO RTS on TX</p> <p>Settings: Auto Dis/Connect Time Stream log custom BR Rx Clear ASCII table Scripting</p> <p>Receive: CLEAR AutoScroll Reset Cnt 13 Cnt = 34 HEX ASCII LogDateStamp StartLog StopLog Req/Resp Dec Hex</p> <p>R? R? R? R? R? R? R? R? R? R? R? R? R? R? R? R? R? R?</p> <p>Press 'a' to program target or AOK to run target code</p> <p>UPDI programmer: UNO_UPDI_IP_V8 (also works on an Atmega 328 clocked at 8MHz)</p> <p>Press any key to start</p> <p>tinyAVR P:0D:0-3</p> <p>Initialising NVM programming</p> <p>Erase chip? -y- or AOK Erasing chip</p> <p>Device unlocked</p> <p>Ready for NVM programming</p> <p>Signature byte readout 001E 0094 0024</p> <p>Programming fuses</p> <p>Program flash with hex? -y- or AOK</p> <p>Send file *****</p> <p>Verifying flash: AK to continue</p> <p>Press 0 or AOK to escape</p> <p>Fuses: WDT, BOD, OSC, SYS0 0000 0000 007E 00F7 SYS1, APPEND, BOOTEND 00FD 0038 0000</p> <p>Program flash with text? -y- or AOK</p> <p>Unlocking device</p> <p>-y- to Lock device or AOK</p> <p>Device unlocked</p> <p>Session terminated</p>	

Note: AOK stands for any other key and POR for power on reset

Alternative deployment scenario

Here the Arduino exports the hex file rather than uploading it to a UNO. The UNO is loaded with “UNO_AVR_Programmer_V2” which copies the hex file to an Atmega 328. The 328 runs off its internal 8MHz clock and therefore all baud rates must be halved and times doubled.

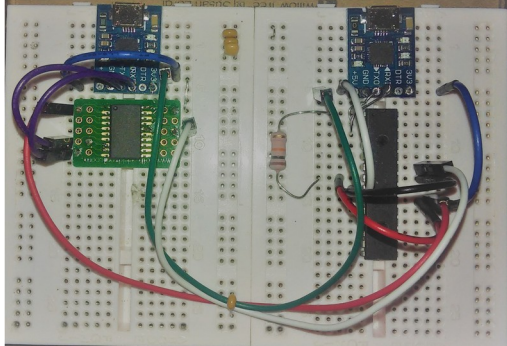


Figure shows Atmega 328 loaded with Programmer V1a and a target device.

Both devices are connected to a CP2102 micro USB bridge.

Hex/text files are uploaded by the Atmega 328 and copied to the target.

The target connection to the PC is completely independent of the Atmega 328.