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

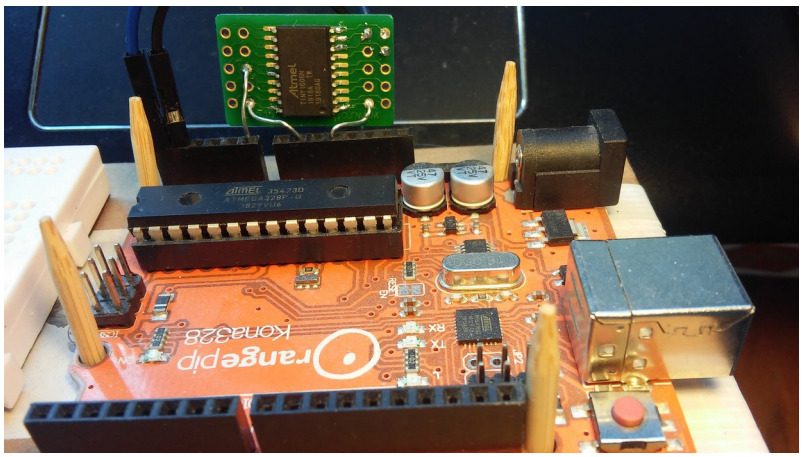
Note: This UART link woks at 9600Baud. Having completed tests the user may wish to increase the baud rate.

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 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.

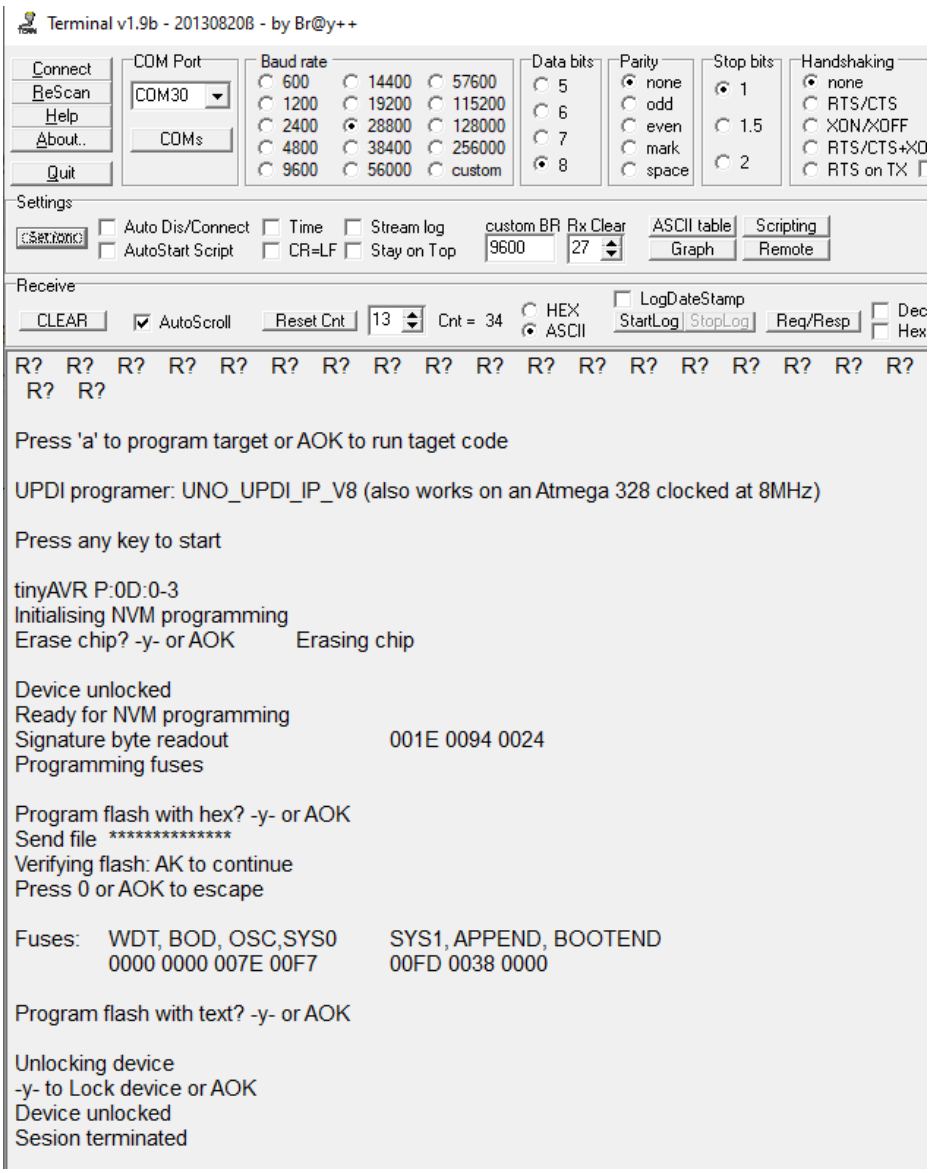
Connect a target device to the UNO as follows:

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:



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

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.

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.

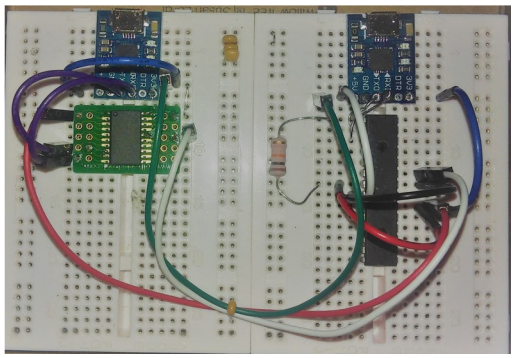


Figure shows Atmega 328 loaded with Programmer V1a and a target device.
Both devices are connected to a CP2102 micro USB bridge.
Hex/text files are uploaded by the Atmega 328 and copied to the target.
The target connection to the PC in completely independent of the Atmega 328.