UNO_AVR_Programmer_V2: Working with the ATtiny44.

Introduction

The UNO_AVR_programmer was developed with the Atmega 328 family of devices. It can also program the ATtiny44 and ATtiny461 families but there are some key differences as listed below.

Hardware	**Atmega 328**	**ATtiny44/461**
Timers	T2 provides port for watch crystal	T2 not implemented
Serial comms	Hardware for UART and I2C (TWI)	USI
Ports	A and B	B, C and D

Note: The Universal serial interface (USI) can be configured as a UART or I2C.
In the absence of a watch crystal the UNO AVR programmer generates an accurate 65.536mS square wave on its SCK pin to calibrate the ATtiny internal RC oscillator.

Hardware setup

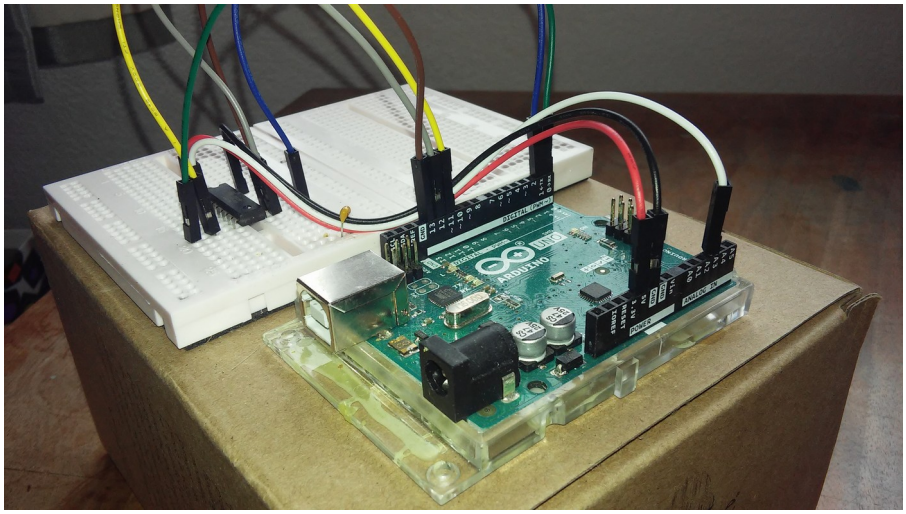


Photo of the ATtiny development setup

Wires are RS part no 791-6463

Breadboard is RS part no 102-9147

5V Decoupling has not been used with the ATtiny44

Interconnections				
Wire Colour	UNO		ATtiny 44	
				Pin
Red	5V		+5	1
Black	GND		0V	14
White	A3	PORTC3	PB3 (RESET)	4
Yellow	Digital IO 11	MOSI	MOSI PA6	7
Grey	Digital IO 12	MISO	MISO PA5	8
Brown	Digital IO 13	SCK	USCK SCL PA4	9
Blue	Digital IO 1	Tx	USI DO	8
Green	Digital IO 0	Rx	USI DI	7

Note: When programming the Attiny44/461 the Rx/Tx wires should be disconnected from the UNO since programming is not compatible with the UNO UART hardware.

An optional LED plus series 1K resistor can be connected between UNO digital IO pin 8 and 0V. This flashes as the hex file is downloaded, indicating that things are working correctly. (Pin 8 sources current rather than sinking it.)

Other resources

In addition to the Arduino application a terminal program is also recommended. My favourite is Br@y++. It can be downloaded from https://sites.google.com/site/terminalbpp/. But take care to download version 20130820, other versions may have an issue with the "scroll" button. Settings for the terminal program are: Baud rate: 38400, 8 data bits, no parity, 1 stop bit and no handshaking.

Setting up the UNO

Connect the UNO to the PC
Use the Arduino application to open UNO_AVR_Programmer_V2
Click on Tools/Port and if necessary select the required port number
Click on Sketch/Upload

Note: If the target device is already connected to the UNO disconnect the Rx and Tx wires from the UNO. The presence of the ATtiny Rx/Tx port prevents the sketch from being uploaded. It also prevents the ATtiny from being programmed.

The calibration program

A single program “Cal_Attiny_44_461” is supplied for both device families. All differences between the target devices are dealt with in the “Project” header file. It is important that the **correct #define statements are selected in this file**.

The USI routines used in this project are taken from Atmel Application note AVR307 and the accompanying source code. See http://becomingmaker.com/usi-serial-uart-attiny85/ for the links. (Note: The author of this web page is also an Osborne, a different Osborne however from the one posting this project).

Note:
UNO_AVR_Programmer_V2 runs on the UNO and has been written in Arduino.
The calibration program runs on the target and has been written in Atmel Studio 7 (its very easy with Studio 7 to change the target device). Both development environments are excellent and I would recommend that beginners become familiar with each one.

The A versions of these devices are basically drop on replacements for the non-A versions. They have the same signature bytes. However the A versions are believed to be a slightly more recent development.

The ATtiny 24 and 261 devices with only 2kB of flash are not able to host this calibration program. If required a cut down version could fairly easily be developed.

Calibrating the ATtiny44/A.

Complete the ATtiny development setup (see photo on page 1).

Ensure that the UNO Rx/Tx lines are disconnected.

Open Br@y++ and click on the "Rescan" button (top left of its window) to identify the correct port number. Then click on the connect button.

After a short pause user prompt s s s s…….. should be displayed (press UNO reset switch).

Press "s" and the UNO should respond with:

> ATtiny 44A detected.
> Press -p- to program flash, -e- for EEPROM, -r- to run target or -x- to escape.

Press P and send Hex_files/Cal_Attiny_44.hex then press zero twice to continue.

The UNO should respond with:
Config bytes (programmed): Fuses extended(if used), high, low and lock 00FF 00D7 00E2 003F

Note the importance of **pressing "P" rather than "p"** for a new device since the config bytes must be programmed.

Click on "n" when a text file is requested.

Press the UNO reset button to reinstate the s s s s…….prompt

Press "s" then "t" to start the calibration clock

Reconnect the UNO Rx/Tx wires and press any key to start calibration

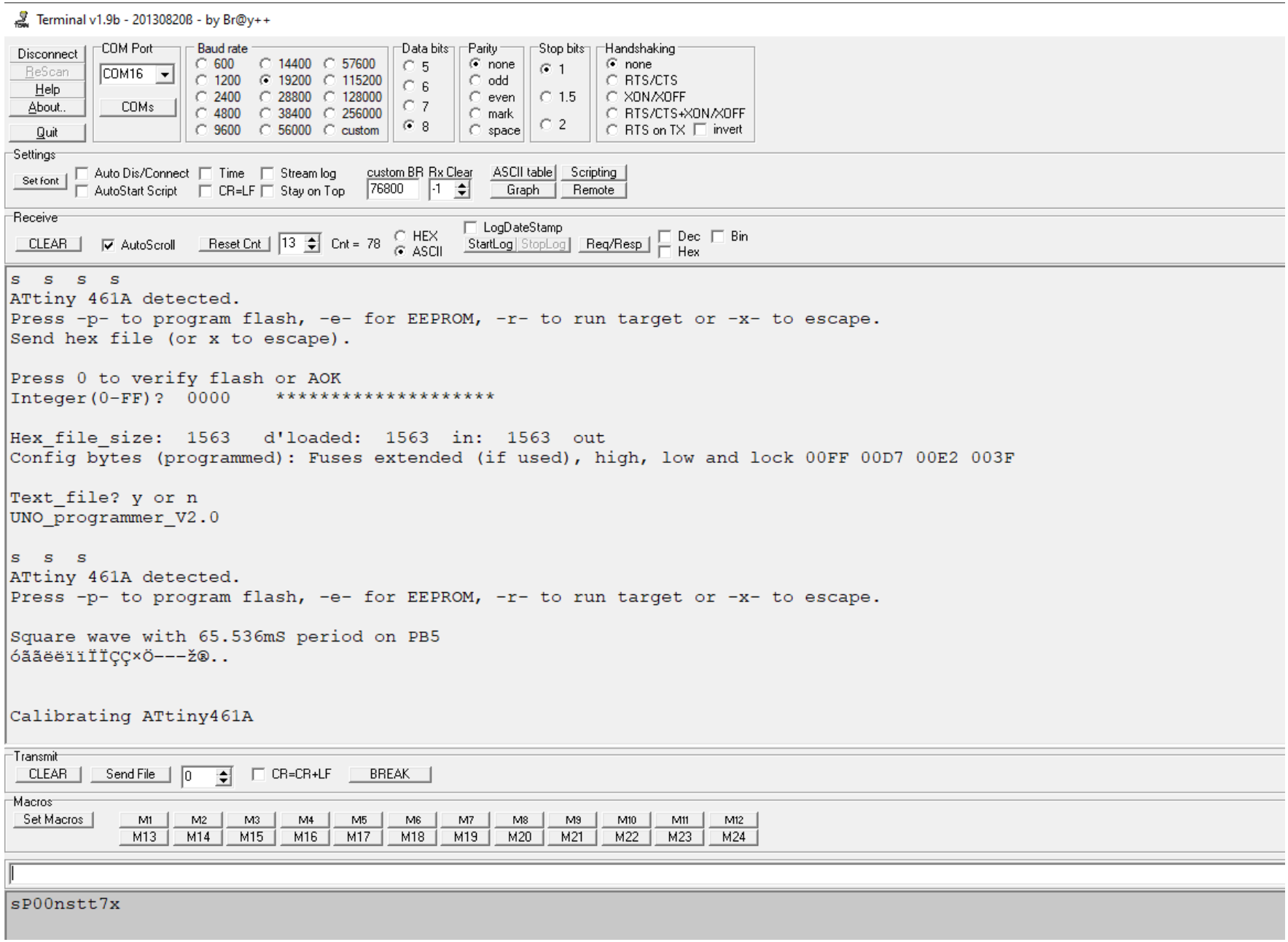
When the table of calibration values is printed out select the value with the lowest error and press "x". It will then be saved to EEPROM.

Note: After pressing the UNO reset it is necessary to wait for about 1 second before the user prompt appears.

See following page for a screen shot of Bray++.
Note:
The Baud rate is set to 19200 rather than 38400. This was just to check the operation of the USI at 19200 Baud. If using the hex files supplied with this posting leave it at 38400 Baud.

Note the user responses. These are entered in the box at the bottom left of the window. The mouse cursor must be left clicked in this region before user keypresses can be accepted.

The white single line box immediately above the user responses can be used for sending strings to the target device.


Terminal v1.9b - 20130820ß - by Br@y++
Disconnect
ReScan
Help
About..
Quit
COM Port
COM16
COMs
Baud rate
600 1200 2400 4800 9600
14400 19200 28800 38400 56000
57600 115200 128000 256000 custom
Data bits
5 6 7 8
Parity
none odd even mark space
Stop bits
1 1.5 2
Handshaking
none RTS/CTS XON/XOFF RTS/CTS+XON/XOFF RTS on TX invert
Settings
Set font
Auto Dis/Connect
AutoStart Script
Time
CR=LF
Stream log
Stay on Top
custom BR
76800
Rx Clear
-1
ASCII table
Graph
Scripting
Remote
Receive
CLEAR
AutoScroll
Reset Cnt
13
Cnt = 78
HEX
ASCII
LogDateStamp
StartLog
StopLog
Req/Resp
Dec
Hex
Bin
s s s s
ATtiny 461A detected.
Press -p- to program flash, -e- for EEPROM, -r- to run target or -x- to escape.
Send hex file (or x to escape).
Press 0 to verify flash or AOK
Integer(0-FF)? 0000 ********************
Hex_file_size: 1563 d'loaded: 1563 in: 1563 out
Config bytes (programmed): Fuses extended (if used), high, low and lock 00FF 00D7 00E2 003F
Text_file? y or n
UNO_programmer_V2.0
s s s
ATtiny 461A detected.
Press -p- to program flash, -e- for EEPROM, -r- to run target or -x- to escape.
Square wave with 65.536mS period on PB5
óããëëïïÎÎÇÇ×Ö---ž®..
Calibrating ATtiny461A
Transmit
CLEAR
Send File
0
CR=CR+LF
BREAK
Macros
Set Macros
M1 M2 M3 M4 M5 M6 M7 M8 M9 M10 M11 M12
M13 M14 M15 M16 M17 M18 M19 M20 M21 M22 M23 M24
sP00nstt7x