Atmega Chips Programmed with Arduino ISP

Atmega328p:

**Name**-----------**Arduino Uno Pin**-------------------**Atmega328P Pin**

MISO--------------Digital pin - 12-------------------------------18

MOSI--------------Digital pin - 11-------------------------------17

SCK---------------Digital pin - 13-------------------------------19

Slave Select-----Digital pin - 10--------------------------------1

VCC--------------------5 V---------------------------------------7 & 20

GND-------------------GND--------------------------------------8 & 22

* Normal Arduino ISP programmer
* Boards: (add text below to boards file)

##############################################################

atmega328bb.name=ATmega328p on breadboard

atmega328bb.vid.0=0x1A86

atmega328bb.pid.0=0x7523

atmega328bb.bootloader.unlock\_bits=0x3F

atmega328bb.bootloader.lock\_bits=0x0F

atmega328bb.build.core=arduino:arduino

atmega328bb.build.board=AVR\_ATMEGA328BB

atmega328bb.build.variant=arduino:standard

atmega328bb.bootloader.tool=arduino:avrdude

atmega328bb.upload.tool=arduino:avrdude

menu.clock=Clock Speed

atmega328bb.menu.clock.1MHz=Internal 1MHz 1.8V+

atmega328bb.menu.clock.1MHz.build.mcu=atmega328p

atmega328bb.menu.clock.1MHz.build.f\_cpu=1000000L

atmega328bb.menu.clock.1MHz.upload.speed=4800

atmega328bb.menu.clock.1MHz.upload.protocol=arduino

atmega328bb.menu.clock.1MHz.bootloader.low\_fuses=0x42

atmega328bb.menu.clock.1MHz.bootloader.high\_fuses=0xDA

atmega328bb.menu.clock.1MHz.bootloader.extended\_fuses=0xFF

atmega328bb.menu.clock.1MHz.bootloader.file=atmega/optiboot\_atmega328\_1MHz\_4800.hex

atmega328bb.menu.clock.1MHz.upload.maximum\_size=31744

atmega328bb.menu.clock.8MHz=Internal 8MHz 3.3V+

atmega328bb.menu.clock.8MHz.build.mcu=atmega328p

atmega328bb.menu.clock.8MHz.build.f\_cpu=8000000L

atmega328bb.menu.clock.8MHz.upload.speed=38400

atmega328bb.menu.clock.8MHz.upload.protocol=arduino

atmega328bb.menu.clock.8MHz.bootloader.low\_fuses=0xE2

atmega328bb.menu.clock.8MHz.bootloader.high\_fuses=0xDC

atmega328bb.menu.clock.8MHz.bootloader.extended\_fuses=0xFD

atmega328bb.menu.clock.8MHz.bootloader.file=atmega/optiboot\_atmega328\_8MHz\_38400.hex

atmega328bb.menu.clock.8MHz.upload.maximum\_size=31744

RESULTS:

void setup() {

// initialize digital pin LED\_BUILTIN as an output.

pinMode(PD0, OUTPUT);

}

// the loop function runs over and over again forever

void loop() {

digitalWrite(PD0, HIGH); // turn the LED on (HIGH is the voltage level)

delay(500); // wait for a second

digitalWrite(PD0, LOW); // turn the LED off by making the voltage LOW

delay(500); // wait for a second

}

Blink uploads and works perfect (LED on pin 2) with Atmega328p on Breadboard, 1MHz clock

Blink uploads and runs 8x slow with Atmega328p on Breadboard, 8Mhz clock

Blink uploads and runs 16x slow w/ Arduino Uno

AVRDUDE:

* Change path on computer

To Get Info on chip:

avrdude -c arduino -p m328p -P COM3 -b 19200 -C "C:\Program Files (x86)\Arduino\hardware\tools\avr\etc\avrdude.conf" -v

Default favrdude: safemode: lfuse reads as 62 (0110 0010)

avrdude: safemode: hfuse reads as D9 (1101 1001)

avrdude: safemode: efuse reads as FF (1111 1111)

https://www.engbedded.com/fusecalc/

LFUSE: Bit 7 first, Bit 0 last

Bit-7 : CKDIV8 : When set divides the clock speed by 8

Bit-6 : CKOUT : When set clock pulses are output on PB0 (Pin 14)

Bit-5 : SUT1 : Startup time delay

Bit-4 : SUT0 : Startup time delay

Bit-3 : CKSEL3 : Set the clock source

Bit-2 : CKSEL2 : Set the clock source

Bit-1 : CKSEL1 : Set the clock source

Bit-0 : CKSEL0 : Set the clock source

To Change Fuse 8Mz:

avrdude -c arduino -p m328p -P COM3 -b 19200 -C "C:\Program Files (x86)\Arduino\hardware\tools\avr\etc\avrdude.conf" -U lfuse:w:0xe2:m

0XE2: (1110 0010)

-U <memtype>:r|w|v:<filename>[:format]

To Change Fuse 16Mz:

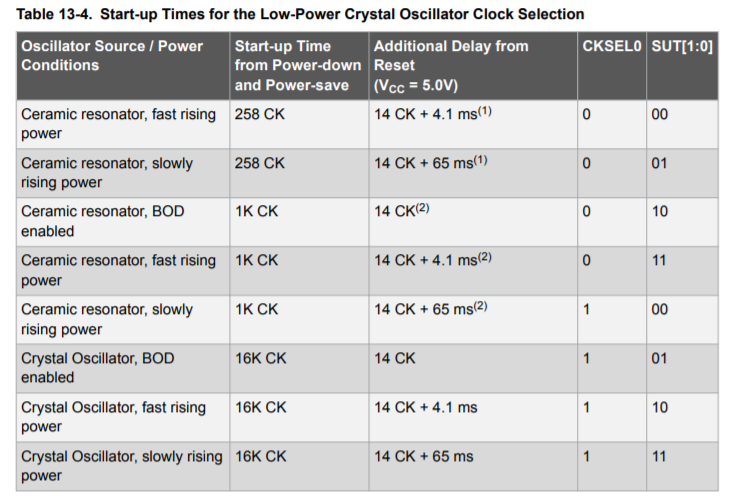
avrdude -c arduino -p m328p -P COM3 -b 19200 -C "C:\Program Files (x86)\Arduino\hardware\tools\avr\etc\avrdude.conf" -U lfuse:w:0xff:m -U efuse:w:0xfd:m

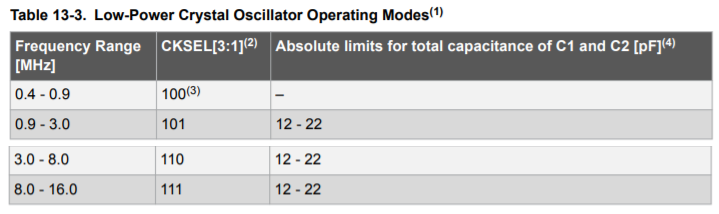
UNO SETTINGS BURNED WITH BOOTLOADER: (MAY NOT WANT THIS!)

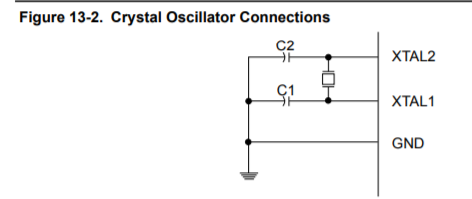
uno.bootloader.low\_fuses=0xff uno.bootloader.high\_fuses=0xde uno.bootloader.extended\_fuses=0x05

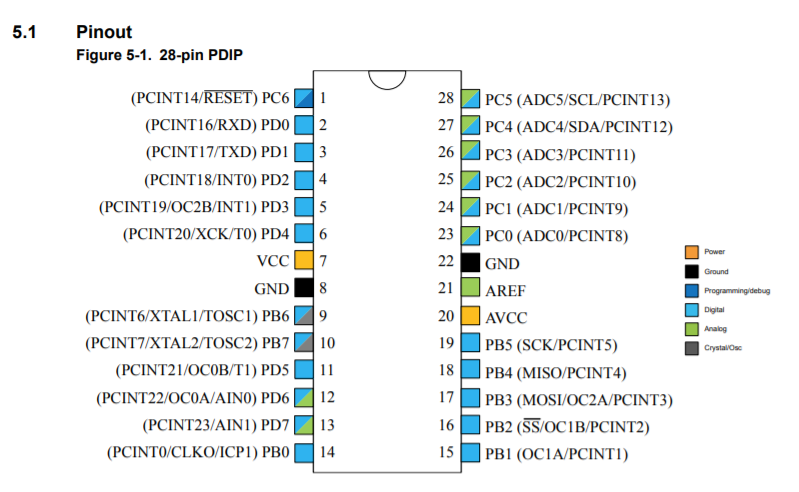
Low Fuse: FF

11111111









HI FUSE: xDE DO NOT CHANGE UNLESS USING BOOTLOADER!!!!

1101 1110

Preset: D9 (1101 1001)

Bit-7 : RSTDISBL : External Reset disable

Bit-6 : DWEN : Debug Wire enable

Bit-5 : SPIEN : Enable serial programming and data downloading

Bit-4 : WDTON : Watchdog timer always on

Bit-3 : EESAVE : Preserve EEPROM memory through chip erase

Bit-2 : BOOTSZ1 : Sets the bootloader memory size

Bit-1 : BOOTSZ0 : Sets the bootloader momory size

Bit-0 : BOOTRST : Select the reset vector

Extended Fuse: x05

0000 0101

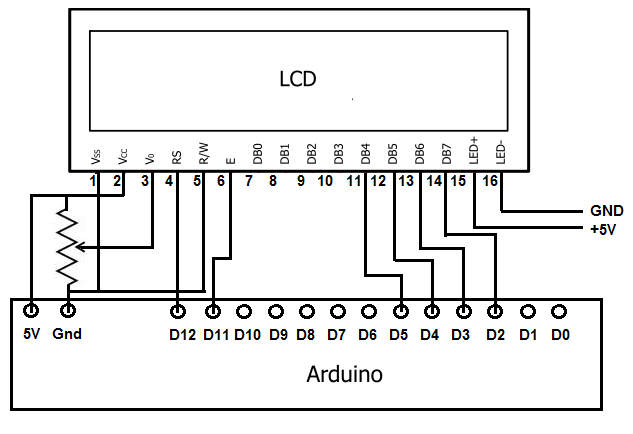
Preset: xFF (1111 1111)

Brown Out Detect at 2.7V xFD 1111 1101 or x05 0000 0101:

per Fuse Calculator: \* Note that some numerical values refer to fuses containing undefined bits (set to '1' here). Depending on the target device these fuse bits will be read either as '0' or '1'. Verification errors will occur if the values are read back with undefined bits set to '0'. Everything is fine if the values read from the device are either the same as programmed, or the following values (undefined set to '0'): Extended: 0x05.

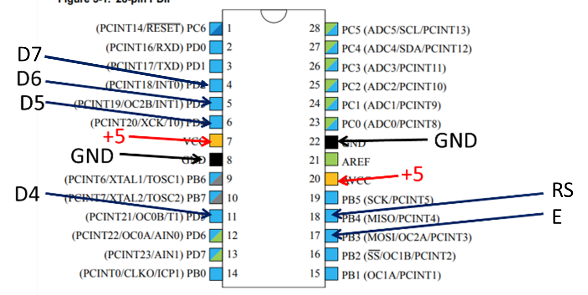
TESTING LCD DISPLAY:

**Normal test (Arduino Hello World)**

  
<http://www.learningaboutelectronics.com/Articles/Arduino-HD44780-circuit.php>

**Atmega328p 8MHz Test**

A: +5 (current limiting resistor in series)  
K: GND  
R/W: GND  
RS: Arduino D12: PB4: pin 18  
E: Arduino D11: PB3: pin 17  
D4: Arduino D5: PD5: pin 11  
D5: Arduino D4: PD4: pin 6  
D6: Arduino D3: PD3: pin 5  
D7: Arduino D2: PD2: pin 4  
VSS: GND, VDD: +5, Vo: wiper on 10k potentiometer



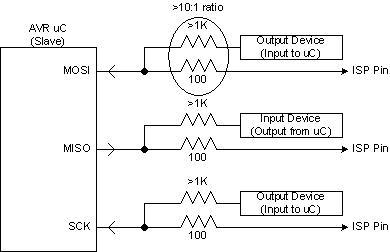
**WORKS!!!!**

Real-Time Programming Board?

So moving the atmega chip back and forth is not going to work. Just too difficult to plug the board into a programmer and then unplug and wire up to test. If the ISP pins don’t overlap with other pins used in the sketch it’s fine, but that means four pins (Reset, MISO, MOSI, SCK) are tied up.

Seems like a better approach is to have the programmer board also be a breakout board with connections for all the atmega pins. But then we need to tristate or disconnect the programmer pins during testing, and connect those same pins during programming. Looks like resistors can have the same function (see pic below).

A tristate buffer should work for the ISP connections since we know the directions (tristate is directional with in -> out flow).



But for the breakout connections we don’t know if they’ll be used for inputs or outputs. A bilateral switch should work for those pins.

A single mechanical switch should toggle the bilateral and tristate so when one is enabled, the other is disabled and vice versa.

Tested Neopixel String w/ all 3 pins driving: MISO, MOSI, SCK Worked fine with all three NOTE USE 330 Ohm resistor between Atmega328p and Neopixel Digital Input, also need 1000uF capacitor between +5 and GND

Code: Strandtest from Neopixel Library

Tested TFT board with:

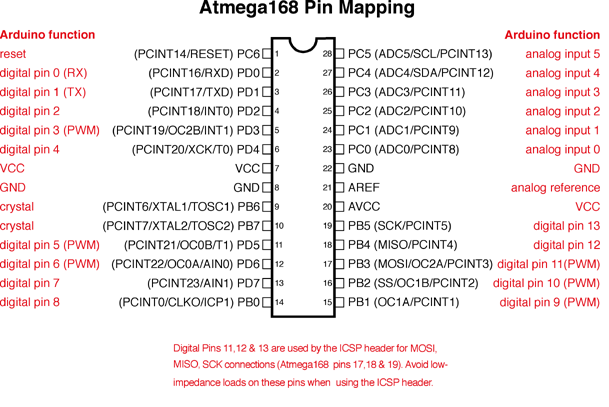
CS Arduino 10 🡪 PB2  
 RST Arduino 9 🡪 PB1  
 D/C Arduino 8 🡪 PB0  
 Vin 5V, GND GND   
 SCK Arduino 13 🡪 PB5  
 MOSI Arduino 11 🡪 PB3

Code: graphicstest from Adafruit ST7735 and ST7789 library

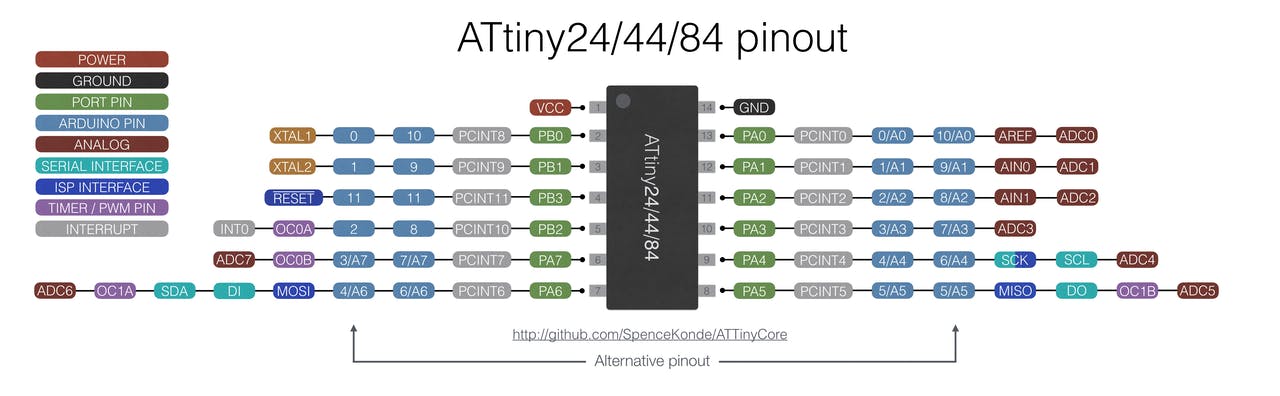
Uncomment:   
// For 1.14", 1.3", 1.54", 1.69", and 2.0" TFT with ST7789:  
Adafruit\_ST7789 tft = Adafruit\_ST7789(TFT\_CS, TFT\_DC, TFT\_RST);

// OR use this initializer (uncomment) if using a 2.0" 320x240 TFT:   
tft.init(240, 320); // Init ST7789 320x240

Worked



ATTINY TESTING: ATTINY84-20PU



Test with Board Attiny84, 1 MHz internal clock, Blink program, Arduino pin 10 (PB0, Attiny pin 2)

Works!

Test w/ Arduino pin 6 also equals MOSI… (PA6, Attiny pin 7)

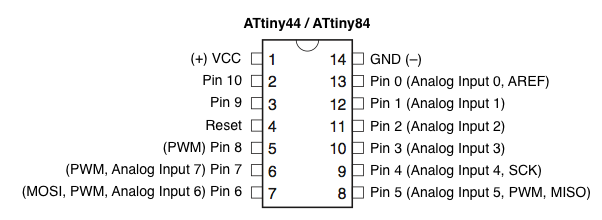
Works!

Test w/ Arduino pin 4 also equals SCK… (PA4, Attiny pin 9)

Works!

Test w/ Arduino pin 5 also equals MISO… (PA4, Attiny pin 8)

Works!



Neopixel: doesn’t compile

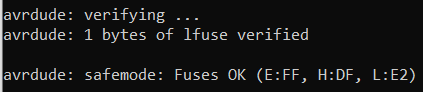
TFT: doesn’t compile

LCD: works:

A: +5 (current limiting resistor in series)  
K: GND  
R/W: GND  
RS: Arduino D10: PB0: pin 2  
E: Arduino D9: PB1: pin 3  
D4: Arduino D8: PB2: pin 5  
D5: Arduino D7: PA7: pin 6  
D6: Arduino D6: PA6/MOSI: pin 7  
D7: Arduino D5: PA5/MISO: pin 8  
VSS: GND, VDD: +5, Vo: wiper on 10k potentiometer

ATTiny as Twinkle… change low fuse so doesn’t divide by 8

avrdude -c arduino -p t84 -P COM3 -b 19200 -C "C:\Program Files (x86)\Arduino\hardware\tools\avr\etc\avrdude.conf" -U lfuse:w:0xE2:m



Blink program works as expected with Internal 8Mz clock selected

Code: <https://raw.githubusercontent.com/sparkfun/LilyTiny_LilyTwinkle/master/Firmware/LilyTiny/LilyTiny.ino>

Changed a bit…

#define fadePin 7 // Physical Pin 6

#define HEARTPIN 8 // Physical Pin 5

// variables for blink pin (pin 2) // Physical pin 11

int blinkPin = 2;

// variables for twinkle pin (pin 3) // Physical Pin 10

int twinklePin = 3;

Uploaded/tested… works. Funnest ones are fadePin / twinklePin (physical 6/10)