Sanguino.cc

so bleeding edge, we had to make it red.

Main
Home
Get It
Build It
Use It
Hardware
Contact Us

Friends
Arduino
RepRap
NYC Resistor
RRRF
Lady Ada

Designed By:
Zach Hoeken

Use Your Sanguino

Setup Arduino Host Software

The Arduino host software does not support the Sanguino out of the box, but it is very easy to add support for the Sanguino to the host software. It as little as 5 minutes, you will be uploading your sketches directly to your board.

Choose the instructions for your operating system:

- Software installation for Mac OSX
- Software installation for Windows
- Software installation for Linux

The best part is that you can now use your Arduino software to upload programs to your Arduino or your Sanguino, by simply choosing the appropriate board from the 'Board' menu.

USB -> TTL Cable

The Sanguino follows the path of many other Arduino clones, such as the Boarduino by omitting the onboard USB->TTL chip. That chip is expensive and only available as a tiny, SMD only chip which is hard to solder.

Instead, we provide the serial interface via a 6-pin header on one end of the board. Using a $20 USB to TTL converter it functions exactly like an Arduino board. The upside to this is that if you have multiple boards with this standard 6-pin configuration, you can re-use the cable and save money.

Suppliers

- RepRap Research Foundation

- Adafruit Industries
- MAKE Store
- Mouser

Program for it

The Sanguino strives to be as compatible with the Arduino software as possible. Obviously with a different chip that has more pins and functionality, there will be some slight differences. Below, we will explain the major differences and how programming the Sanguino is different.

The Arduino reference has all sorts of information that is directly applicable to your board. It helps if you're familiar with it.

pinMode(), digitalRead(), digitalWrite()

The digital pin functions are identical to the Arduino, with one difference: there are more of them!

The Sanguino has 24 normal digital pins, numbered 0-23. As with the Arduino, you can also use Analog pins as Digital pins. Digital pins 24-31 correspond to Analog pins 7-0. Its sort of backwards, but that means Analog 0 is Digital 31, and Analog 7 is Digital 24.

Some of the digital pins have extra features you may want to keep in mind when choosing pins:

- Digital 2 - Interrupt 2
- Digital 3 - PWM
- Digital 4 - PWM
- Digital 5 - MOSI (used for SPI)
- Digital 6 - MISO (used for SPI)
- DIgital 7 - SCK (used for SPI)
- Digital 8 - Rx0 (this is used by the USB<->Serial port)
- Digital 9 - Tx0 (this is used by the USB<->Serial port)
- Digital 10 - Rx1 (this shared with the extra serial connection) / Interrupt 0
- Digital 11 - Tx1 (this shared with the extra serial connection) / Interrupt 1
- Digital 12 - PWM
- Digital 13 - PWM
- Digital 14 - PWM
- Digital 15 - PWM
- Digital 16 - SCL (used for I2C)
- Digital 17 - SDA (used for I2C)

These extra features are optional, and the pins function as digital I/O pins by default.

analogRead()

The analogRead() pin function is identical to the Arduino, with one difference: there are more of them!

The analog pins on the Sanguino are numbered 0-7. You have 2 more analog pins to work with.

analogWrite()

The analogWrite() function is identical to the Arduino, but the pins have switched around.

The PWM capable pins on the arduino are: 3, 4, 12, 13, 14, 15.

attachInterrupt(), detachInterrupt()

The Sanguino attachInterrupt() and detachInterrupt() functions are identical to the Arduino functions, but there is one more external interrupt pin.

The interrupt capable pins are correspond to these digital pins:

Interrupt	Digital Pin
0	10
1	11
2	2

Serial

The atmega644p chip is super awesome because it provides a 2nd USART, which you can use for serial communication. Using the This means that you can easily talk serial to your computer, and on a completely different serial link, you can talk to:

- another Sanguino or Arduino.
- a GPS unit.
- something else that talks serial.

Using the first/normal serial port is exactly the same as with the normal Arduino.

Using the additional serial port is easy: you include the appropriate header file, and then use the serial object as you would the normal serial port. Wiring it up to your device is up to you, however. ;)

To use the 2nd serial port, just start writing code that uses the *Serial1* object.

```
void setup()
{
  Serial1.begin(19200);
  Serial1.println("Hello world!");
}
```