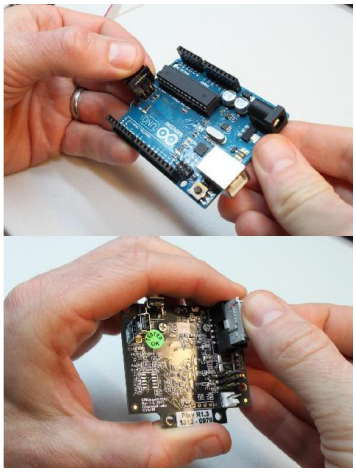
Hooking up Pixy to a Microcontroller like an Arduino

Pixy is meant to talk to a microcontroller, and Pixy comes out of the box ready to talk to an Arduino. It sends block information to Arduino at 1 Mbits/second which means Pixy can send more than 6000 detected objects per second or 135 detected objects per frame (Pixy can process 50 frames per second.)

OK, to get Pixy and Arduino talking to each other, use the supplied Arduino cable to connect Pixy to your Arduino.



Next, download the Arduino library "arduino_pixy-x.y.z.zip" here. Bring up the Arduino IDE and import the Pixy library by selecting **Sketch➔Import Library** in the Arduino IDE, and then browsing to the Pixy.zip file that you just downloaded.

Next, load the "hello_world" example by selecting it in **File➔Examples➔Pixy**. Upload it and bring up the **Serial Monitor**. You should see messages printed that look similar to this:

```
Detected 1:
  block 0: sig: 1 x: 159 y: 109 width: 61 height: 61
Detected 1:
  block 0: sig: 1 x: 173 y: 114 width: 60 height: 61
Detected 1:
  block 0: sig: 1 x: 146 y: 111 width: 70 height: 65
...
```

Note, this example will only print messages if Pixy is running the "default program" and an object that matches one of its color signatures is visible.

Arduino API

Using Pixy with Arduino is really simple. You simply include the SPI and Pixy headers:

```
#include <SPI.h>
#include <Pixy.h>
```

And make a global instance of Pixy by putting this little guy outside your setup() and loop() functions:

```
Pixy pixy;
```

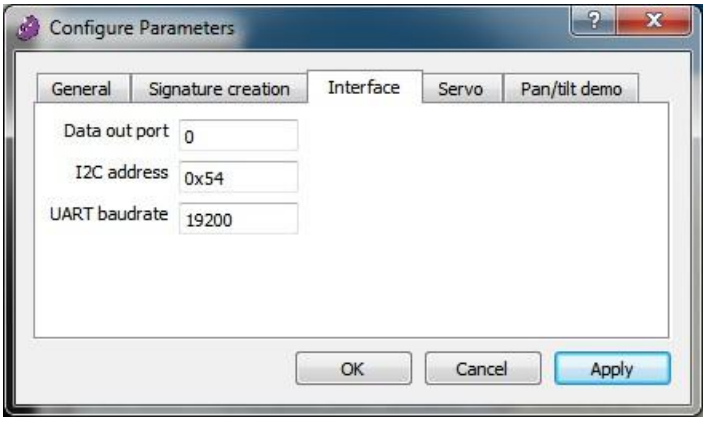
The API consists of one call: `getBlocks()`, which returns the number of objects Pixy has detected. You can then look in the `pixy.blocks[]` array for information about each detected object (one array member for each detected object.) Each array member (`i`) contains the following fields:

- `pixy.blocks[i].signature` The signature number of the detected object (1-7)
- `pixy.blocks[i].x` The x location of the center of the detected object (0 to 319)
- `pixy.blocks[i].y` The y location of the center of the detected object (0 to 199)
- `pixy.blocks[i].width` The width of the detected object (1 to 320)
- `pixy.blocks[i].height` The height of the detected object (1 to 200)
- `pixy.blocks[i].print()` A member function that prints the detected object information to the serial port

So it's simple to talk to Pixy with your Arduino!

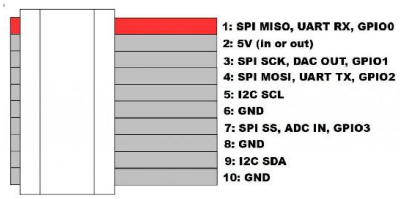
Other Microcontrollers or Devices

Pixy will output objects that it detects through one of several interfaces that you choose, and it will do this every 20 ms. It supports SPI, I2C, UART, and analog/digital I/O through its 10-pin I/O connector. Pixy also supports USB 2.0 through its mini-USB connector. You can configure which interface Pixy uses through the configure dialog in PixyMon. The "Data out port" parameter in the "Interface" tab determines the output port.



If you hover the mouse pointer over the "Data out port" text, a help string will be displayed that describes which value corresponds to which type of port.

- 0: SPI - this is the default port that uses 3 wires (pins 1, 3, and 4 of the I/O connector) and is used to communicate with Arduino
- 1: I2C - this is a multi-drop 2-wire port (pins 5 and 9 of the I/O connector) that allows a single master to communicate with up to 127 slaves (up to 127 Pixys).
- 2: UART - this is the common "serial port" (pins 1 and 4 of the I/O connector). Pixy receives data via pin 1 (input) and transmits data via pin 4 (output).
- 3: analog/digital x - this will output the x value of the largest detected object as an analog value between 0 and 3.3V (pin 3). It also outputs whether an object is detected or not as a digital signal (pin 1 of the I/O connector).
- 4: analog/digital y - this will output the y value of the largest detected object as an analog value between 0 and 3.3V (pin 3). It also outputs whether an object is detected or not as a digital signal (pin 1 of the I/O connector).



Technical notes on the interfaces

All serial interfaces use the same protocol. It may be helpful the refer to the Pixy's schematic.

SPI

The SPI interface operates as an SPI slave. It is designed around the Arduino's ICSP port, which doesn't have a slave select signal. The default data rate is 1 mbits/sec, but this can be increased by modifying the Pixy.h file in the Pixy Arduino library. The protocol has checksums to deal with bit errors, but bear in mind that the ribbon cable isn't shielded!

I2C

- The I2C interface operates as an I2C slave and requires polling.
- There are somewhat weak 4.7K pullups to 5V on SDA and SCL signals, via R14 and R15.
- I2C signals are 5V tolerant
- The I2C address can be configured in the "Interface" tab of the Configure Parameters dialog in PixyMon

There is an Arduino example that uses I2C. Run it by selecting **File➜Examples➜i2c** in the Arduino IDE. You'll also need to make a special cable that connects:

1. Arduino SDA to pin 9 of Pixy's I/O connector
2. Arduino SCL to pin 5 of Pixy's I/O connector
3. Arduino GND to pin 6, 8 or 10 of Pixy's I/O connector

When talking to more than one Pixy, you will need to configure a different I2C address for each Pixy so they don't step on each other. You can make a "multi-crimp cable", meaning you can take a 10-conductor ribbon cable and crimp N 10-pin IDC connectors to it and plug into to your N Pixys. That is, when selecting I2C as an interface, all signals on Pixy's I/O connector go into a high-impedance state and won't interfere with each other, waste power, etc.

UART serial

- The UART interface is 8 data bits, 1 stop bit, no parity, no handshaking
- The baudrate can be configured in the "Interface" tab of the Configure Parameters dialog in PixyMon
- RX signal (pin 1) is 5V tolerant input

- TX signal (pin 4) is 0 to 3.3V signal output
- Baudrates up to 230 kbaud supported.

There is an Arduino example that uses UART serial. Run it by selecting **File➔Examples➔uart** in the Arduino IDE. You'll also need to make a special cable that connects:

1. Arduino TX to pin 1 of Pixy's I/O connector
2. Arduino RX to pin 4 of Pixy's I/O connector
3. Arduino GND to pin 6, 8 or 10 of Pixy's I/O connector

Analog/digital output

Pixy has a single analog (DAC) output, so there are two modes for analog/digital output. Mode 3 outputs the x value of the center of the biggest detected object to pin 3 of the I/O connector. Mode 4 outputs the y value of the biggest detected object to pin 3 of the I/O connector. Pin 1 goes high (3.3V) when an object is detected, and low (0V) when no object is detected.

- Pin 1 is 0 to 3.3V signal output, can source/sink 5 mA.
- Pin 3 is 0 to 3.3V signal output, roughly 200 ohm impedance.
- Pin 3 voltage is directly, linearly proportional to the object position in the image (depending on the mode)
- In mode 3 (x mode) pin 3 is 0V if object is on the far left of the image and 3.3V is object on the far right (PixyMon perspective).
- In mode 4 (y mode) pin 3 is 0V if object is on the bottom of the image and 3.3V is object on the top (PixyMon perspective).