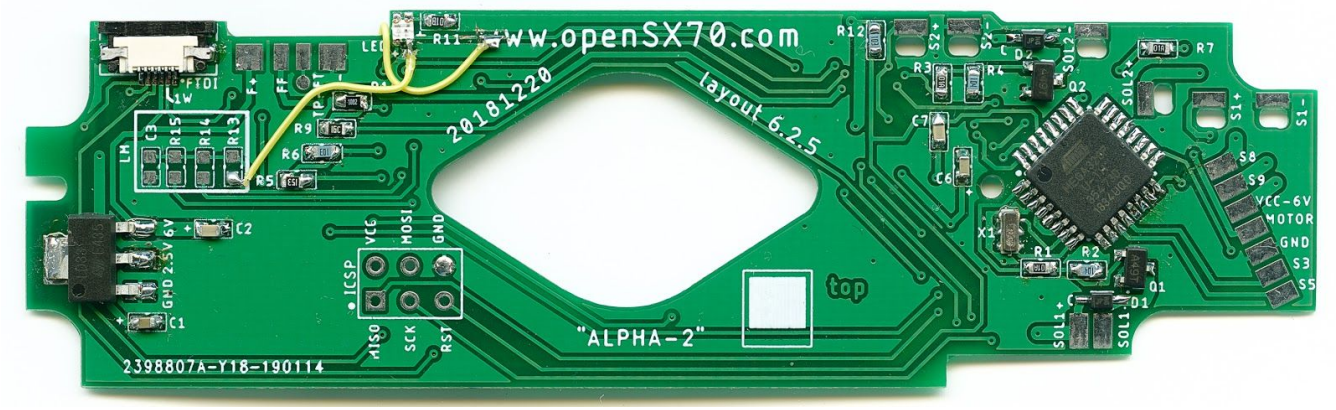
opensx70 software upgrading

version 3.1

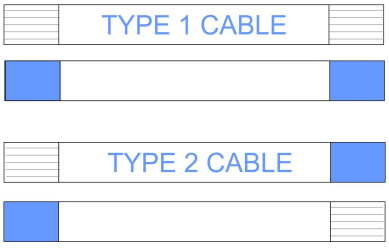
Why do I have to program my camera?
The quick answer is that you don't have to, but you probably want to upgrade to a more up-to-date version of the microcode running in your camera, especially* now that the system is highly experimental.

What do I need?
You need the openSX70 pcb, normally already mounted in the camera.



you need the cable, it can only be either type 2, now we **only** use type 2.

FPC cables types

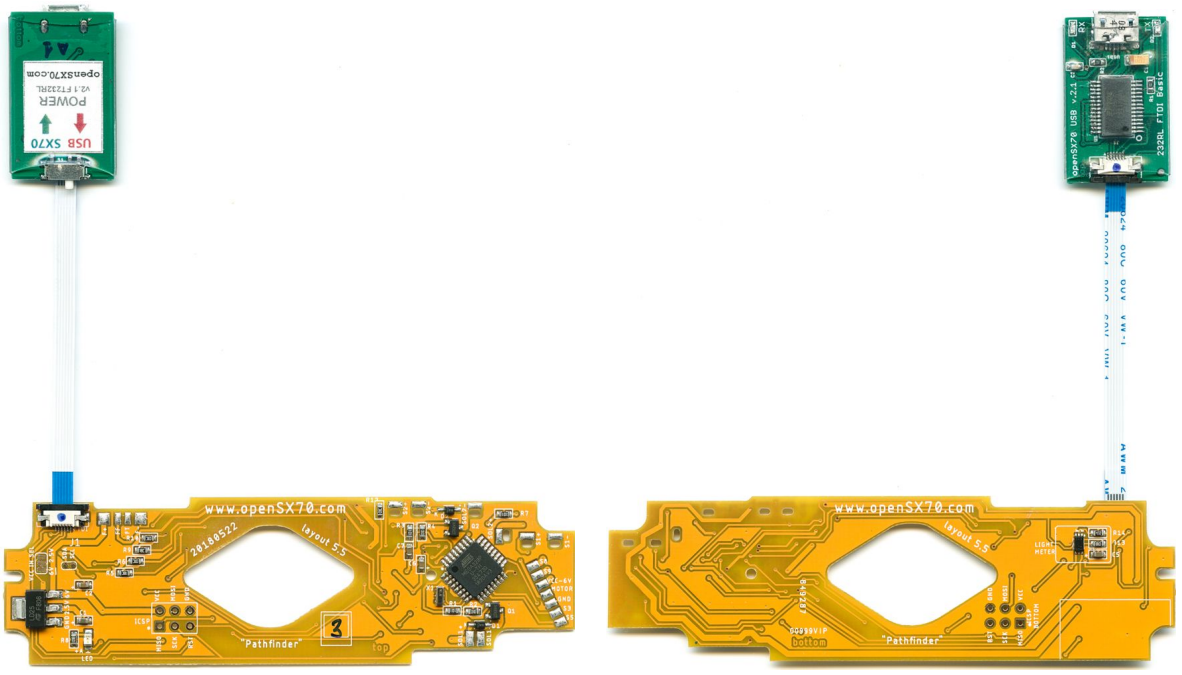


And of course the FTDI-USB adapter, now standard is type D based on the FT232RL FTDI chip:

in the "in camera" configuration getting power from SX70.

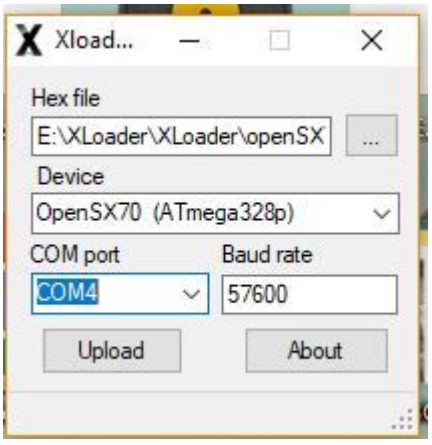
The adaptor has a switch once the main PCB is in the camera it has to be in “SX70” position. Then it will get the power from the pack on the camera.

BUT if you want to program the PCB BEFORE is installed in the camera it can be USB powered selecting “USB” in the POWER SWITCH:

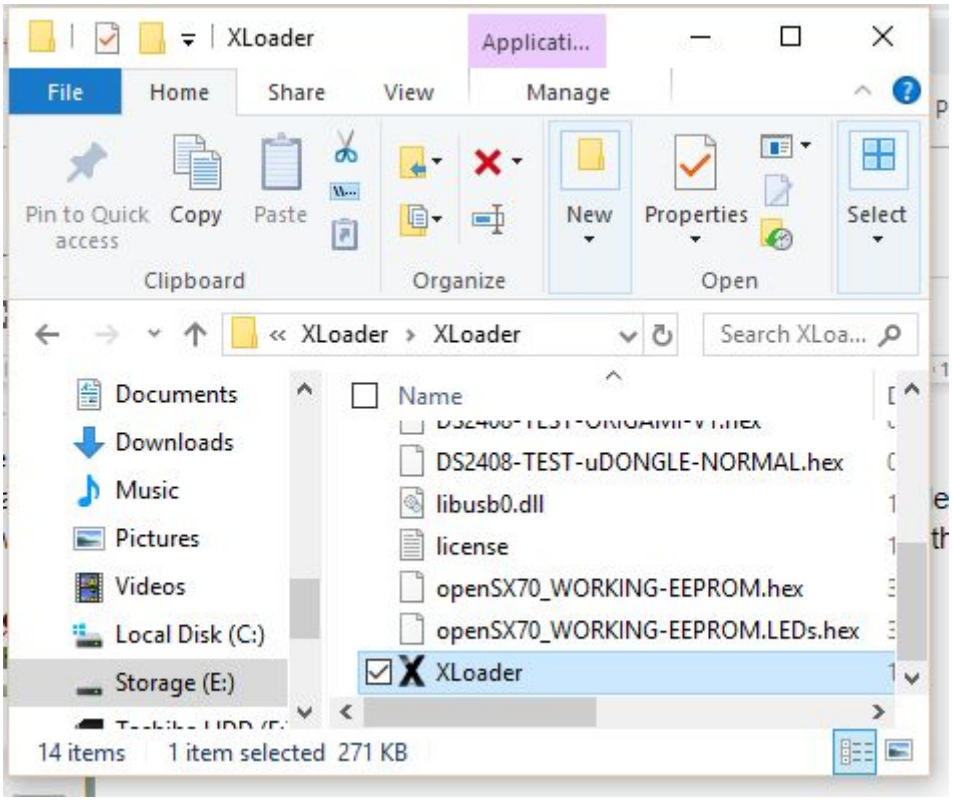


How do I connect the FTDI USB adapter?
Please read the assembly instructions as how to connect and different options depending on the connectors installed in the adapter and the camera.
You do not have to leave a FPC cable connected, but I guess it is a good idea initially as to be able to test new sketches or firmware. You can also try different things that are very simple to change on the code.
There are two methods to upload code to the camera. Please make sure you select the software suitable for your camera/dongle.
With the Xloader (PC only) you can easily upload the software as if it were a firmware. If you install and configure the Arduino IDE (Mac, PC and Linux) you can also quite easily upload the firmware but also modify small things without having to understand the code. You press Ctrl-F and search “OPTION” and check all the options.

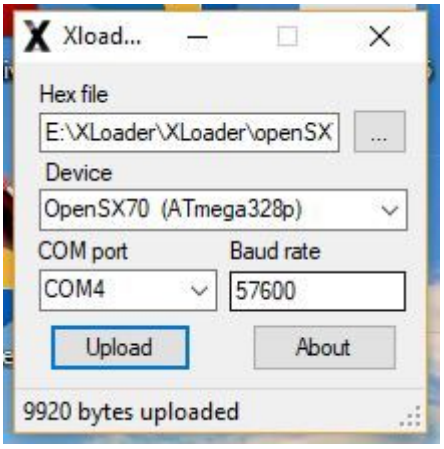
Programming from Xloader (PC)
Copy the Xloader files in the zip file to folder of your liking



execute or double click on the Xloader file:



In Hex file select the file you want to load. Device should be OpenSX70 (Atmega328p).
Select the COM port (it shows when you connect the FTDI to the PC). Do not change “Baud rate”.
And click “Upload)



Mac users: There is a similar tool for macs called HexUploader. I haven’t tested yet but it should work.

Programming from the arduino IDE
The software is not only compatible with both PC and Mac but also with Linux.
You need to install the “Development Enviroment”

https://www.arduino.cc/en/Guide/HomePage
https://learn.sparkfun.com/tutorials/installing-arduino-ide

Also the FTDI driver
https://learn.sparkfun.com/tutorials/how-to-install-ftdi-drivers/all

-installing the needed libraries

https://www.arduino.cc/en/Guihde/Libraries

The libraries you need are in the main openSX70 arduino Github repository:
https://github.com/openSX70/openSX70-Arduino-main/archive/master.zip

The code is in a folder within the same zip that came with Xloader. Or get it from Github.

Setting all up
You need the camera with an empty cart, the FPC cable properly connected to both the camera and the USB adapter and the USB cable connected to the PC. The IDE must be running and the openSX70 should be opened.
In the IDE menu, *tools* you have to check the port. A new port shoud appear once it is connected a powered. This is good. You can also test the port on your PC by only connecting the adapter without FPC. The port should appear. If it does not check that you have the proper FTDI drivers.

Then also in the *tools* menu you select *board* and then "***Arduino Pro or Pro Mini***"
Next to that in tools is *processor:* choose the option "**Atmega328P 3.3V 8 mhz"**
(Note, you might know that we are running the board at 2.5V, that is alright)

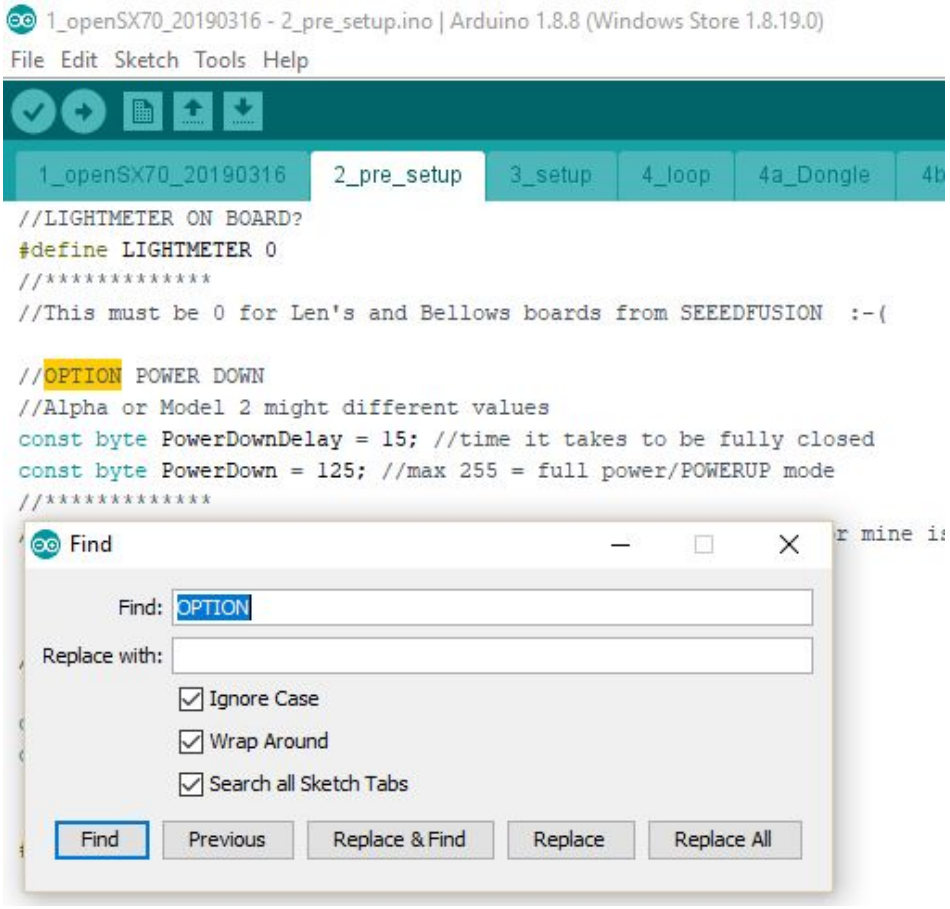
Fixing "crazy mouse" (from David Walker)

> In the ftdiport.inf file, there's a line that says
>
> [FtdiPort.NT.HW.AddReg]
>
> HKR,,"UpperFilters",0x00010000,"serenum"
>
> just put a ; before the second line
>
> and it stops the port enumerating as a serial mouse

Also look here:
https://stackoverflow.com/questions/9226082/device-misdetected-as-serial-mouse

Uploading a new sketch
Just like in a regular Arduino you have to press the right pointing arrow or simply CTRL-U. Of course, as mentioned, you need to have the sketch open in the ide.

Do I need to be a programmer?
Of course you don't, you just have to follow the steps outline before and you will easily upgrade your camera to the latest software.
But there are some things that you might want to test once the sketch is open in the IDE.
You can for example, easily change the shutter speed, or the time you have to keep the red button pressed for the timer delay or things like that. Just press CTRL-F and search of OPTION



There are many things you can change:

Shutter Speeds:

//OPTION
int ShutterConstant = 22 ;
int ShutterSpeed[] = { 1, 4, 6, 7, 11, 17, 22, 34, 42, 80, 140, 290, AUTO600, AUTO100, POST, POSB };
//int ShutterSpeed[] = { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F };
//int ShutterSpeed[] = { EV17, EV16, EV15, EV14, EV13, EV12, EV11.5, EV11, EV10.5, EV10, EV9, EV8, AUTO600, AUTO100, T, B };

//reduced speeds from 25 (slot5) to compensate flash firing

int FastestFlashSpeed = 25 + ShutterConstant;

// to change the speed in the slot position just change the number corresponding.

Red button behavior options:

//OPTION RED Button timing variables
int debounce = 20; // ms debounce period to prevent flickering when pressing or releasing the button
int DCgap = 250; // max ms between clicks for a double click event
int holdTime = 500; // ms hold period: how long to wait for press+hold event

LED remaining shots "counter":

//OPTION this is the so called LED counter coment auto this part to disable the counter.
//LED COUNTER

```
  //if (digitalRead(S8) != HIGH || digitalRead(S9) != LOW)
  if (digitalRead(S8) == LOW && digitalRead(S9) == LOW) //NORMAL OPERATION
{
simpleBlink (8 - (EEPROM.read (4)));
}
```

Solenoid 1 options:

```
//OPTION POWER DOWN
//Alpha or Model 2 might different values
const byte PowerDownDelay = 15; //time it takes to be fully closed
const byte PowerDown = 125; //max 255 = full power/POWERUP mode
```

This options are important if you are having trouble with the shutter. There is a power-saving mode (in the original camera, called Power down) so the camera gives all the power (255) to close the shutter, then waits for a few ms specified in the PowerDownDelay value. You might have to wait longer for actually closing. Then it goes to PowerDown in this case is half the power 125, specified in the PowerDown value. You can also change (usually increase) this value if you have trouble with the shutter.
I have not figured out nor tested enough to make this work with original/model2 cameras. Pushing the BC/AC S4 mechanical switch affects speed. You are on your own here, but if you figure something out let me know!

Origami V1 dongles:

```
//OPTION for origamiV1 dongles
#define origamiV1 0
```

change to: #define origamiV1 1

Other options:

To remove the blinking “counter” on opening change to 0.
#define LEDCOUNTER 1

If your board does not have viewfinder LEDs turn to 0.
#define VFled 1

This will make the software “manual” if your PCB doesn’t have a TSL235 sensonr.
#define LIGHTMETER 1

Alternatives to Arduino IDE:
https://www.visualmicro.com/page/Arduino-Visual-Studio-Downloads.aspx

This is an add-on to the regular Microsoft visual Studio. For instance for VS2019.

https://visualstudio.microsoft.com/downloads/