# FloppyControl getting started

FloppyControl is hardware and software to capture raw floppy through usb to a windows application. The Windows application can convert the raw digital flux data into mfm, sectors and finally into a disk image.

The hardware consists of an Arduino Duo (32 bits 84MHz ARM mcu) with a native usb port for fast data transfers.

The software consists of an application written in .net 4.5 winforms c# application that’s compatible with Windows 7 to Windows 10. This application controls the hardware to move the head and capture the data.

What you need:

* Arduino Due (Chinese clone works but original is recommended to avoid any issues)
* PCB shield for the Arduino (screw terminal version or male molex connector for the step stick version)
* 5 SMD 1205 resistors of the value 150 to 470 ohms. It’s very easy to solder with any type of soldering iron that’s suitable for soldering PCBs
* 34 pin dual row floppy connector (male)
* 5x 8pin single in line header (male)
* 1x 10pin single in line header (male)
* 1x 4pin single in line header (female or male, your choice). For the direct connection of the stepper motor.
* Screw terminal (optional)
* Step stick (optional)
* Floppy cable (straight or with twist both work)
* Molex splitter cable with floppy drive power connector
* Power supply 12V and 5V for the step stick and floppy drive. It can’t be powered from USB unfortunately. I used an Aliexpress psu with both 12 and 5V and a build in Molex connector which works fine and costs about 4 dollars/euros (search for Malloom 2019 New Arrival Power Supply 12V And 5V AC Adapter For Hard Disk Drive CD DVD-ROM)
* Windows PC
* Good quality micro usb cable
* Computer mains power cable
* Soldering iron, solder, maybe some flux
* Small pointed tweezers for placing the smd resistors
* Flux remover like flux off, IP, denatured alcohol or acetone
* Paper towel
* Clamp to put the pcb in for convenient soldering.

I can solder and test the PCB for you for about 30 euros without p&p.

## Soldering

1. Solder the resistors first. Whet one pad, then put the resistor on the pad and solder one side, then the other side.
2. Solder the Arduino connectors to the bottom of the board (non silkscreen/text side) so that the longer leads point downward and the shorter leads are through the holes of the pcb. For better alignment, place the connectors on an Arduino Due
3. Solder the top connectors, the order doesn’t matte
4. Clean the solder joints with flux off, IP, denatured alcohol or acetone, use a tooth brush. Use a paper towel to remove excess liquid. I use the toothbrush to pad down the towel onto the through hole pins without rubbing. Two or three times usually is enough to get all the flux and liquid residues off.
5. If you want to control the stepper motor (for capturing off center disks, disks with reglued disk donuts) unsolder the wires of the stepper motor and solder some wires with a 4 pin connector. Check if the coils are correct (ie pin 1 and 2 should be one coil, pin 2 and 3 should be the other). If the stepper motor doesn’t turn properly adjust the current (should be around 0.3 to 0.45V as measured from the potentiometer wiper and ground). If it still doesn’t turn swap one coils wires, so swap pin 1 with pin 2. If the stepper motor runs in the wrong direction turn the connector 180 degrees, so pin 1 becomes pin 4 and vice versa.

# Electrical testing

1. Check with a multi meter that there’s no short circuit from 3v3 to GND, 12V to GND
2. Check that the resistors measure the correct value when probing the through hole pins of the connectors. ie put the positive lead on 3v3 and probe each resistor on the floppy connector.
3. Check the voltages of the power supply and pinouts. 12V on the screw connector and 5V (and 12V but is rarely used) to the floppy drive
4. Check if none of the pins of the psu has a high voltage (110V) with regards to the PC ground (use the usb shield for example). Some cheap Chinese psu’s aren’t safe to use. There shouldn’t be any voltage, so under 1V should be ok.

# Programming the Arduino Due

1. Install the Arduino Ide (I used v. 1.8.9, the latest version at the time of writing)
2. Before plugging the shield on the Arduino Due, connect the Due to the PC. Use the usb connector closest to the barrel power connector (on the bottom of the pcb it’s named programming atmega16u2).
3. When the board is plugged in, the Arduino Ide should show a message at the bottom that you need to install the board, please do so by clicking on the bottom and clicking install in the following screen. This can take a few minutes and may appear to hang but please be patient).
4. Once the board is detected, you can select Arduino Due in the menu Tools > Board > Arduino Due programming port (at the bottom of the list)
5. Select the correct com port.
6. Click the arrow icon at the top of the screen to program the Arduino. This should take about 10 seconds.
7. Once it’s reported that it’s done at the bottom of the log screen, unplug the micro usb cable from the Arduino and connect the other usb connector (marked as native USB =>SAM3x on the bottom of the pcb).
8. Select the board Arduino Due (native port). And select the appropriate com port.
9. Go to Tools > Serial Monitor
10. Select No Line ending at the bottom, baud speed is irrelevant.
11. Type the lowercase ‘i’ without quotes and hit enter. This should show:
    1. Arduino Due working!!!
12. The board is working!
13. Now plugin the shield, place the step stick

# Setting up FloppyControl App

Now that the board is tested and working, open FloppyControl app.

1. In the menu click Settings, select the same com port and click save.
2. Depending if you use the step stick, use the preset Direct step if you’re not using a step stick or ‘Step stick preset’ if you do. For normal disks no microstepping is needed. If you’re capturing a reglued disk, use the glued preset, it will microstep between the tracks and make a full recovery from glued disk donuts which are often off center.
3. Before each capture it’s advisable to set the correct recovery Base filename. It will create a folder of that name in the main recovery folder which can be set in the settings panel (menu Options > Settings). Each capture will have the basename with track start and end and an incremental number appended to it. Each time a file is saved the number is incremented to make sure no data is ever overwritten.
4. Each new start capture will clear the buffer and at the end of the capture the file is saved to the harddisk. You can click the ‘Explore here’ to open an explorer window at the file location.
5. Click the capture button and the capture should start immediately. The floppy will seek track 00 and start capturing from there.
6. Before processing can start the peaks of the histogram needs to be set. Click on the histogram if you see that the first peak is highest, then the second is the second highest. If not, scroll using the slider to a point where this is the case. Now click on the histogram. The triangles will move to the peaks automatically (or slightly shifted, as this has been found optimal for detecting the sectors better).
7. Determine if the captured data is from PC or Amiga and press the respective Process buttons.

Congratulations, you’ve successfully setup the workflow to start recovering your floppy disks!