# Apple SD Floppy Drive Emulator - BMOW

Autor:
Data de publicació: 16-03-2017

December 15th, 2012 | Category: Floppy Emu

Floppy Emu is a prototype floppy disk drive emulator for vintage Macintosh computers. It emulates 400K, 800K, or 1.4 MB disks, and is compatible with everything from the original Macintosh 128K through the Mac II series and Power Macintosh line.

You can buy a pre-assembled and tested Floppy Emu, or if you're comfortable with surface mount soldering and have an AVR programming tool, you can build a Floppy Emu at home. Be warned there are several SMD chips with fine-pitch 0.5 mm pin spacing. These can be soldered by hand, but it's challenging work.

Floppy Emu is released under the Creative Commons Attribution-NonCommercial 3.0 license. You're free to build one for personal use, or adapt and share the design for other non-commercial projects, but please don't build a batch of Floppy Emus and sell them on eBay. Now with that out of the way, let's get started!

Download the Floppy Emu source files

What You'll Need

A Windows PC
AVR Studio 5 software from Atmel. This is free software. AVR Studio 6 probably works too, but I haven't tried it. AVRDUDE should work as well.
An AVR programmer, such as the AVRISP mkII
An SD or SDHC card that's fast when transferring small data blocks. Lower capacity cards tend to perform better at small transfers – try a card with 2GB capacity or less.
Soldering iron
Patience

Making a PCB

To begin building your Floppy Emu, you'll need a printed circuit board. Send me an email to ask if I have any extra PCBs available. Otherwise, there are many on-line services that will make these for you, using a set of layout files called Gerbers. You'll find the Floppy Emu Gerbers in the file archive, in the directory eaglefloppyemugerbers.zip. I recommend using the Dorkbot PDX PCB Order service to get the PCB manufactured. Send them the Gerber files, and they'll make three copies of the PCB for about $30, with a turn-around time of a few weeks. The guy who runs the PCB service is very helpful if you have questions.

Getting the Parts

While you're waiting for the PCB, you can locate all the other parts you'll need to assemble the Floppy Emu. You'll find a complete parts list in the file archive. The total parts cost (not including the PCB) should be about $40-$50 when ordering in single unit quantities, with taxes and shipping costs.

One of the required parts is a male DB-19 connector, which mates with the Mac's external floppy port. These can be very difficult to find, and IEC is the only supplier I know that has them. If you can't find a DB-19 solder type connector, you can still use the Floppy Emu with its alternate IDC20 connector, which is readily available from electronics suppliers. With the IDC20, there are several different connection options:

Connect the IDC20 to the internal floppy connector inside your Mac (you will need an IDC20 cable)
Connect the IDC20 to the circuitry inside a real external Apple floppy drive enclosure, after removing the old drive's guts
Use an Apple II DB-19 to IDC20 cable

If you don't already have one, you'll also need an AVR programmer like the AVRISP mkII.

Assembly

Here's where things get fun! Refer to the file board-layout.png for placement information, or check the schematics.

The first step is to solder and test the AVR.

Solder the AVR chip. Pay attention to the orientation – the dot in the corner of the chip should match the dot on the silkscreen. I recommend using the drag solder technique. Begin by soldering two pins on opposite corners of the chip to tack it in place. Verify that the pins are perfectly aligned with the pads on the board before proceeding. Now goob an absurd amount of solder onto the remaining pins. You will probably bridge most of the pins together – that's OK. Finally, go back with a solder wick and flux, and wick away all the excess solder. Surface tension will hold the solder onto the pins and pads where it belongs, while the excess is magically wicked away. Use a 10x jeweler's loupe to visually inspect the pins for tiny bridges or gaps.
Solder the crystal X1, the 18pF capacitors C12-C13, and the AVR decoupling capacitors C4-C7 (these are on the back of the board).
Also solder the DB-19 connector. It fits along the edge of the board, with the thickness of the board fitting between the two rows of solder cups.

The AVR microcontroller should now be functional. You can use AVR Studio to test it out.

Install the AVR Studio 5 software on your PC and run it.
Connect the Floppy Emu board to your Mac, and turn on the Mac.
Connect the AVRISP mkII programmer to your PC.
Press the ISP connector into the board, but do not solder it yet. If it is soldered too soon, it will be in the way of your iron later.
Connect the AVRISP's programming cable to the ISP connector on the board.
Click on the AVR Programming button in the toolbar. It looks like a chip with a lightning bolt.
In the programming dialog, set Tool to AVRISP mkII, Device to ATmega1284P, and Interface to ISP
Click the Apply button
Where it says Device ID at the top, click the Read button. Push the ISP connector firmly sideways while you do this, to ensure its pins make electrical contact with the board.

If everything is working, the Device ID field should now display 0x1E 0x97 0x05. If it does not, or you encounter an error, review the steps above, then go back and carefully re-check all the solder connections. Do not proceed further until you are able to retrieve the AVR device ID.

The next step is to program the AVR with the Floppy Emu software and bootloader.

Open the AVR Programming dialog again.
From the list at the left, click Fuses to display fuse information.
Set the fuses:

BODLEVEL – disabled

OCDEN – off
JTAGEN – off
SPIEN – on
WDTON – off
EESAVE – off
BOOTSZ – 2048W_F800
BOOTRST – on
CKDIV8 – off
CKOUT – on
SUT_CKSEL – EXTXOSC_8MHz_XX_16KCK_65MS

These settings should result in the fuse bytes being Extended: 0xFF, High: 0xDA, Low: 0xBF.

Click the Program button in the fuse panel. Push against the ISP connector with your finger while you do this.
From the list at the left, click on Memories to display AVR memory information
Under Flash, verify that “Erase device before programming” is checked
For the Flash filename, choose AVRreleasemerged.hex from the file archive
Under Flash, click the Program button. Push against the ISP connector with your finger while you do this.
Remove the ISP connector from the board.

Congratulations, the AVR setup is now complete! The next step is the Nokia LCD.

Solder the voltage regulator, and the regulator capacitors C10 and C11. Important: C11 is a 33uF polarized tantalum capacitor. The positive side of the capacitor should be on the right, towards the upper LCD connector.
Solder the level converter (the 74LVC244), its decoupling capacitor C8, and the 10K resistor R4.
If you want backlighting for the LCD, solder the backlight resistor R5. Personally I think it looks better without backlighting, so I leave R5 empty.
Solder the 8-pin male 0.1 inch header to the bottom of the Nokia LCD, and the 2-pin header to the two middle holes on the top of the LCD. The thicker part of the LCD bezel should be at the top.
Solder the 8-pin female header to the bottom of the board, and the 2-pin female header to the top of the board.
Connect the LCD to the Floppy Emu board.

The LCD should now be functional. Connect the Floppy Emu to your Mac, and turn the Mac on. If everything is working, the LCD should show a version number screen, followed by “SD Card init error”.

The SD card reader is next.

Solder the SD connector, and its decoupling capacitor C9.
Insert the SD card into your PC.
Copy some Macintosh disk image files onto the SD card.
Put the SD card into the Floppy Emu.

Connect the Floppy Emu to you Mac, and turn the Mac on. If everything is working, the LCD should show a menu listing of the disk images on the SD card.

The final assembly step is the CPLD.

Solder the CPLD. Pay attention to the orientation – the dot in the corner of the chip should match the dot on the silkscreen. Use the same technique for soldering the CPLD that you used for the AVR.
Solder the CPLD’s decoupling capacitors, C1-C3. These are on the back of the board.
Solder the LEDs and their resistors, R1 and R3. The silkscreen shows the proper orientation for the LEDs.
Solder the four tactile switches.
Solder the remaining connectors into place.

Now it’s time to program the CPLD. The AVR programs the CPLD indirectly, using a firmware file stored on the SD card, so no JTAG tools are needed.

Ensure the file firmware.xvf is on the SD card. This is part of the Floppy Emu source files archive.
Put the SD card into the Floppy Emu.
While your Mac is turned off, connect the Floppy Emu to the Mac.
Hold down both the PREV and NEXT buttons, then turn on the Mac.
Follow the instructions on the LCD to load the firmware file to the CPLD.

That's it! Turn the Mac off, then on once more, and you're done.

Using the Floppy Emu

When it's turned on, the Floppy Emu scans the SD card for files with a .dsk, .img, or .image extension, and shows a menu of available disk images. Use the PREV/NEXT buttons to select a disk image file, then press the SELECT button to insert it into the emulated disk drive. After the disk is inserted, the LCD display shows the current track number and active side of the drive.

The Floppy Emu expects 400K, 800K, or 1.4MB disk images in either raw format (the .dsk images typically used with Macintosh emulators), or DiskCopy 4.2 .image format. Raw image files support reading and writing to the emulated floppy disk. DiskCopy 4.2 images are read-only.

Floppy Emu supports normal sector-by-sector writing, such as copying files in the Finder, or saving data from within a program. It does not support formatting the emulated floppy, or using it with format-and-write disk copy tools. If you need a blank disk image file, create one on your PC and then copy it to the SD card.

Extending Floppy Emu

The C source code for the microcontroller program is provided in the file archive. You can build it with AVR Studio, generate a new floppyemu.hex and merged.hex file, and load it to the microcontroller using the AVRISP mkII.

The Verilog source code for the CPLD is also provided. You'll need Xilinx's free ISE WebPACK to build it. Export an XSVF file, and rename it to firmware.xvf. Then copy the file to the SD card, and load it onto the CPLD as you did before.

Happy hacking!

Read 30 comments and join the conversation

30 Comments so far

bbraun December 16th, 2012 1:31 pm

A huge thanks for this! I know it takes forever to do all the documentation to bring others up to speed. Thanks a bunch for doing all that.
I've ordered PCBs and parts!

jebug29 December 22nd, 2012 8:26 pm

._. Now I have to find someone who knows how to solder for me. Hopefully someone here will get permission to make

and sell these on eBay, with credit and a portion of  money to the guy who made it.

petersieg December 23rd, 2012 3:44 pm

Looks like I have to get some parts and pcb’s made