mfm_write can write an emulator file to a disk drive. I got it to the state I could write the disk I needed but it is not a finished program. Currently it can only write an entire disk at once. You need to edit mfm_write.c main routine to set the parameters for your drive. I have not fixed command line parsing.

These command line options will be supported at some time along with options for setting the write precompensation. Currently only emulation_file, version, and quiet work. Emulation_file must be specified.

--begin_time -b #
The number of nanoseconds to delay from index to start reading track
--cylinders -c #
The number of cylinders.
--drive -d #
Drive number to select for reading. Only valid for read command. Drives are number 1 to 4.
--emulation_file -m filename
File name to write emulation bit data to. No file created if not specified
--heads -h #
The number of heads.
--quiet -q #h
Bit mask to select which messages don't print. 0 is print all messages. Default is 1 (no debug messages). Higher bits are more important messages in general.
--unbuffered_seek -u
Use unbuffered/ST506 seeks. Default is buffered/ST412.
--version -v
Print program version number.

To work mfm_read-00A0.dts line

```
0x190 0x07 // OUT P9_31 = gpio3_14
```

needs to be commented out and this line uncommented before running setup_mfm_read

```
//0x190 0x2d // OUT P9_31 = pr1_pru0_pru_30_0
```

Use mfm_read to verify the disk is properly written. The first attempt had a couple tracks that seem to be written to the wrong head. The next run worked ok. This program does not do anything to avoid using bad locations on the disk,