KansasFest Session - Disk Imaging with Applesauce

Presented by John K. Morris

Thursday, July 20 2017

08:45-09:15

John will discuss the development of the Applesauce floppy drive controller that enables you to connect your Apple II floppy drive directly to your modern computer via USB. He will also discuss how to use the software and various other topics around disk imaging and copy protection.

08:45 Disk imaging - I had a problem with my EDD card and I wanted to do something different. I embarked on this journey to capture disk information in a different way.
What I ended up building was a hardware solution. It is a floppy controller that allows you to connect a Disk II to your computer over USB.

The nuts and bolts: What is actually on a disk?
There is a common idea that a floppy disk contains 0s and 1s. A floppy disk only contains 1s. We have a strip of magnetism and everything faces either north or south. As the head rolls over this information, or the information rolls past the head. The head picks it up and if the bits there are north facing, it sends an ac current one way, and if they are south, it sends the current the other way.

What the drive does is, there's one chip at the heart of it: the 3470. It looks for the current to change one way or the other. SO whenever it changes from north to south or south to north, it sends a message out. That message is a 1, a one micro-second long pulse to denote one. 0s are derived by the lack of impulses over time. The A][expects a pulse every 4 micro seconds. If there isn't one happening there, when it does find one… it says' Hey it's been about 8 micro seconds, that's a zero, I don't want that'. So a gap makes 0s.

You've heard the term Flux? That is the transition north to south or south to north. The Applesauce connects up and it reads the flux transitions directly. It grabs every transition on the disk and it measures the amount of time of nanoseconds in-between these. It can take that stream and create bitstreams, nibble streams and take it all the way from being pulses to data. Basically what the floppy drive controller does.

Applesauce shown on the screen, Fast Disk Imager mode. John comments he pulled a few disks from upstairs for demonstration. he presses start and it shows tracks being copied, successful in ~12 seconds. This is "Early Games". Image file saved.

Here we have Early Games, the .DSK file is double clicked to launch with Virtual][. The game boots quickly and is shown working.

What we witnessed was many steps in that 12 seconds. It's puling the fluxes, turning them into bits, then into nibbles, then de-encoding it. This has created a disk image. This is for unprotected disks. Apple sauce has a number of techniques for data repair. If the address parts of the disk are made, it will hunt for chunks of data and bring things back.

if you have bad nibbles it'll go into the flux transitions that make up the nibbles to determine potential values that nibble could have been, then verify against a checksum, to re-create a chunk of lost data.
Many different techniques are employed.
All this relies on the fact the disk has a known structure. ProDOS, DOS 3.3 and so on. When DOS works it
It has address prologues and checksums, because real the Disk II isn't that accurate.

Question: So you're telling me when it creates the disk image that it's actually verifying the data ?
Answer: Yes.
Question: Does EDD do this?
Answer: No, EDD does not do any validating.

There are many ways to do data recovery when there is a set structure, allowing for aggressive methods.
When it comes to copy protection, it kind of turned into a wild west out there. All DOS 3.3 address chunks start with the beginning of address information a sector D95A6. Everything is logically laid out in one big loop with little keys in there. As it finds the keys, it says OK this is a sector, look at the sector number and go to read the data following it - that is your sector. Publishers didn't like that because you could copy everything. Checks sector numbers to grab data and that's your sector. Publishers first changed sector recognition with cat and mouse games. EG moved address information to D97A6. Affects Copy II plus. From there bit copiers were effective in grabbing bits / raw information as they didn't care about whether it's copying a sector or not. It just grabs the raw information.

The copy protection folks came up with, well one of many. One was an exploitation of the 3470 Drive II chipset (creates the pulses that makes the bits). They noticed you can't have more than two 00s in a row on a disk because you can't have too much time of the 3470 chip not seeing

anything. When this happens, the chip thinks it's not doing its job well so it uses an amplifier and keeps turning it up to find the data it's supposed to be finding. Until it gets to the point that it's amplified noise as random data. This is why you can't have more than two 00s in a row - this creates random bits from the amplifier.

To defeat bit copiers, publishes added areas of no transitions. Bit copies run and see the 'bits', spin the disc around again and the drive controller has made the data different. We know that this is a real disk because these bits keep changing. When you put it into Copy II plus or EDD, it runs through and it hits this no transition area and random bits spit out. And Copy II Plus thinks of those bits as real, it doesn't understand if it's real or fake. **It grabs those and cements these as being part of the image. When the software runs, it comes to this pattern of bits here Spins around 'Hey this is the same'. Spins around 'This isn't changing' leading to detection of a copied disk.**

Question from the audience - nibbles. DOS 3.3 and above has a finite six-and-two encoding (16-sectors) where logic sequences take 256 bytes / sector, spit out 342 nibbles (with checksum) to represent the bytes. This increases size but it allows for encoding without more than two zeros.

09:00
With DOS 3.2, 13-sector disks they used. They required five-and-three encoding to make 410 nibbles. The rules for earlier encoding and state machine chip was that there could only be one zero. Moving on to DOS 3.3 they came up with six and two to shrink packets from 410 to 342 to gain room for the additional three sectors. An intermediate stage in there solely to convert into binary. How do you store a 0?!

Then questions is presented - how do you actually store 0?
So they came up for a way to make bit copies suffer, a lot.

When you have a bit stream you have to establish what is real and what is fake. Each bit is assumed to take four-microseconds. Transitions are variable, drive speed is constantly wiggling around. In fact, it's a miracle its works at all!

Woz did an amazing job taking the components to make what he did.
Protected disks need flux imaging.

Software demonstration: Imaging of "Mastertype", protected, with Applesauce. Introducing flux imaging, this records every flux transition on the disk and the number of microseconds between

each. Applesauce can locate where every flux transitions exists on the media itself. It has as special sensor to be mounted inside a Drive II which watches the spindle to keep the alignment of tracks. A visual representation is shown, representing the density of flux across a 5.25" disk media. The grey intensity of each pixel in Applesauce shows flux. The more you have the brighter it will be.

Question: What do you do with areas of no transitions?
John: Real bits we can get fine but we still need to deal with the problems of fake bits (from amplification). Luckily real bits don't really move around as they maintain a consistent timing to their neighbours. Allows for determination of what lines up - **Applesauce grabs five copies of each quarter track**. Comparison - is this consistent, is the timing between these bits consistent? When it gets to an area of no transitions it can see solid and random bits. Random bits being thrown in where they dance around and number change, turns into a big garble Looking over multiple reads, things move around inconsistently. Applesauce will see these bits are being inconsistent and find the the extent of these inconsistencies (generated by 3470) and eliminate the fake bits. So it will pull the false things out.

*Jeremy comments EDD is just ONE copy of quarter tracks and therefore can make no comparison. EDD is at the mercy of the drive controller giving it accurate bits in one pass.

09:08 Flux transitions shown on screen. Track 0 on this outside edge …. we have a bunch of bits in a stream but where things are starting? That is done with sync nibbles FF followed by 00 which delays the system. Four of these in a row tells the logic to bring it into sync. Four sets of sync nibbles guarantee aligned of data.

These vertical lines here are sync areas. This is where it wants to make sure the data following it is properly
These sync fields allow for data to be read with correct synchronisation. This effectively leaves watermarks. On a regular disk II, all these are skewed as it has no idea of the position of the disk when it writes a sector. 'OK next sector', writing a whole track. By having each sector aligned this way (on slide) shows high quality equipment generated this floppy, possibly a whole set of tracks written once. There area lot of variance in drive speed - this can be managed. It'll write a big area of sync, come around and overwrite the end to make sure a long set of bits all around. It's the safest way to do that.

Referring to the on screen flux image - a lot of nothing is on track 0 which is unusual but important to starting things up. Applesauce generated a RAW file / disk imaging complete 09:11.
We want to improve the ability to recognise and analyse disks even with older disks failing. The work that Mark is doing is trying to save them before they are gone. This is another way to do that.

A reminder that five flux transitions are capture for later processing. You can get enough information resulting in intact protection. Johns explains his excitement when finding new protection schemes and exploring the craziness that publishers went to. The brilliance is far more interesting than the content of the disc itself. “Sometimes I look at a disc, wow what are they doing here? I spend two hours spelunking a disk… it’s truly fascinating and really an art form. Usually far more interesting than the content of the data!”.

A2R file is generated at approximately 35mb - from this we can generate other formats. This A2R process is future proof. No more interpretation of data, we are saving real flux transitions in it’s lowest, raw form.
Reminder: Nibs can contain some protection but when it comes down to it there isn’t a format that exists (until A2R) that maintains protection. A2R can extract everything and generate different formats of disk images. The raw is the raw timing of flux transition - no questions involved “is this a zero?”. Raw bits captured in their lowest form.

Choplifer with spiral tracking demonstrated. Disc author did not use sectors.

Spiral pattern is shown - comment 1x1px PNG images, representing progress are saved. Realtime. black 0 white ~255. Under construction.

Sporadic pattern shown (game Frogger). Entire game exists on the edge with really tightly packed, interleaved quarter tracks. Insanity

Hardware is straightforward, leaving software to do heavy lifting. Accurate to approximately 125 nano seconds.

Question: When can we buy this? John jokes a few things are exploding /… fire bad …. / more things to do in standardisation of images so lets give it a month.

Question: Alternative track lengths.

John: No reason why it couldn't image up to 40 tracks but currently 35 track limit.

Question: Can this be used portions of unprotected damaged discs?
john: Yes. when it finds damaged areas, it can point it out and show you what flux transitions do exist. A new tool is being worked on, like a nibble editor, to look at individual flux transitions. From this it can interpolate what should be there (in .DSK creation).

Question: Unidisk
john: Disk II allows for sync sensor, Unidisk is too small to accommodate this. Sync sensor goes und

Presentation ends 09:20