Very quick and dirty introduction to the technical background of optical discs

This text doesn't go deep into the details. Serves just a quick overview about *what* is done, not *how* exactly. Hope I got the stuff right – **without any warranty for correctness**. Due to my heavy concentration issues I'm still not very confident with this stuff and very slowly learning.

==

Pressed optical discs contain pits and lands (you should already know this). A binary "one" is a change from land to pit or from pit to land while a binary "zero" is no change (keep being on the same level – pit or land). Pits and lands reflect a very different amount of focused laser light (wavelength is approximately/proportional to the depth of the pits. A BD drive must contain three lasers because of this).

For CDs (later discs are similar, a bit more advanced but the same principle)
The modulated analog(!) signal generated by the optical detectors getting the laser reflection must not contain too many changes (each one has to be followed by at least two zeroes) and not too few changes (after ten zeroes there must be a one). The signal could not pass the A/D converter with too many changes and the optics might lose track with too few changes and/or the converter might become unable to determine the exact number of consecutive zeroes.

You will immediately notice that these harsh limitations make CDs unsuitable for *any* kind of computer data. To get arbitrary data into those restrictions, each normal byte (8 bits) are converted to 14 bits with a lookup table. For each 256 possibilities a byte can contain, from 0000 0000 to 1111 1111, there is a 14-bit representation fulfilling the requirements above. After each byte (14 bits) come three merge bits, set accordingly to make sure end+beginning of two consecutive bytes don't violate the conditions. This is called **EFM encoding** (eight-to-fourteen modulation).

Lastly the drive will run into trouble if the area sizes covered by pits isn't approximately equal to the area size covered by lands. A regular pattern of data, appears all the time in normal data, could wreck havoc with this balance. Hence all data is run through a pseudo-random scrambler applying XOR function. It is very unlikely to encounter normal data that would result in regular patterns after running through the scrambler (precisely what SafeDisc 2 and later abused for copy protection purposes: regular patterns **after** running through the scrambler)

What I left out are the parity codes. Reed-Solomon encoding. Huge amount of redundancy, error correction and error detection. Goes way above my head anyway. CDs have additional error correction within the sector data visible to the end user with what is called "RAW read/write". But the end user has no access to the scrambled data on the CD and to 14-bit representation of each byte. You can't *truly* read RAW data from CDs, let alone write. For DVDs we do not even have this pseudo-RAW read/write.

Now to that link above:
What the guy behind this blog entry did, was implementing the EFM+ and the scrambling of DVDs (not to be confused with CSS video "protection" which is also has the name "scrambling") and the error correction *in software*. He intercepted the data flow, which the firmware wants to send to the writing laser. Say the burner gets told to write an ISO image of a DVD, starts burning normally, behaves normally, but without the firmware knowing it, doesn't send any data to the laser. He sends different data from a different device directly to the laser. This is absolute RAW data and he has 100% control over everything, even completely different sector encoding of GC/Wii discs or no usable data at all – for drawing pictures or the BCA barcode thing.

If the whole setup fully worked like he intended, I guess this would probably have beaten almost all DVD (and maybe CD) copy protections. The PlayStation "wobble" stuff can **not** be replicated this way.