Don’t Worry, It’s Only Ones and Zeros

As you’re reading this, the chances are pretty good that you’re doing so with the help of a computer. You might be using a personal computer, a tablet computer, or a smart-phone. Today computers are all around us, in things most of us probably don’t even think about. Our entertainment depends on computers. We listen to music through digital recordings and digital playback. We watch movies and programs on television recorded and transmitted digitally. Even some of what we watch in movies and television programs was created digitally, never having existed in the real world.

If you drive a modern car one or more computers control the engine and the transmission. You navigate using computers, and radios. Computer controlled machines make many of the pieces of the car. We use computers to communicate, to manage our finances, to track our currency and the things around us. It’s hard to think of an area of our lives that isn’t somehow influenced by the many ways we use computers.

All of this has come to pass in just under a hundred years. With current advancements in computer hardware and software, it’s not hard to imagine a day when we will talk to our computers the same way we talk to one another. “Artificial intelligence” is an increasingly common part of our lives. Some people wonder how long it will be before computers are smarter than we are and start ruling us. Others dream of cybernetically enhancing our brains or uploading our personalities and memories into computers.

These fantasies about the future of computing remind me of other fantasy works. L.E. Modesitt Jr. has written dozens of fantasy and science fiction novels, and he continues to churn them out. In one of these novels, *The Timegod* (expanded from an earlier work titled *The Fires of Paratime*), there is a conversation between the main character and his supervisor about the nature of *polite barbarians*. The gist of the conversation expresses the idea that a barbarian uses the technology of the day without any understanding of its principles. By that standard, most of the people in the civilized world today are barbarians.

That almost seems like a counter-intuitive notion of barbarism. The first entry in the dictionary for the word *barbarian* is “a person in a savage, primitive state; uncivilized person.” I’m pretty sure that’s the meaning that normally comes to mind when we hear or

read about barbarians. The second entry is “a person without culture, refinement, or education; philistine.” This is closer to the understanding in Modesitt’s work, but the idea of *polite barbarians* takes it a bit further.

> *Baldur didn’t like cultures where the “elite” were barbarians, where only a few understood the technology that underlay the civilization.*

A *polite barbarian* may be cultured and refined. A *polite barbarian* may even be educated. They may be able to use technology and take advantage of the benefits of civilization. They may even run the civilization, but they’re still barbarians under the skin.

Politeness is nothing more than a thin veneer over the savagery and inhumanity of humanity. It’s the behavior and manners that we learn in a culture to keep other savages like ourselves from doing to us what we’d do to them if we had the power. Barbarians *need* politeness or they’ll destroy the civilization that gives them security and comfort.

That’s another notion that finds its way into works of fantasy and science fiction. We’ve created technological terrors in our modern world. Novelists and screenwriters love to write about them and the consequences of their use. We’re treated to tales of zombies (created by weaponized viruses used in biological warfare) and post-apocalyptic societies (the apocalypse* being the consequence of biological, chemical, or nuclear warfare). Time traveling robots from the future come back to destroy the people that defeat them in the future before they can grow up (in a world where artificial intelligence has run amok), and so on.

Piers Anthony is another prolific science fiction and fantasy author, and he gave us a story of a world depopulated thanks to space travel in his *Tarot* series of novels (I wouldn’t recommend this series by the way, it’s easily among his worst works). If I recall correctly (it’s been decades since I read this) “Brother Paul” is the main character in this series. He’s a monk, and he understands basic technology, which on his world has broken down and is almost lost. The series opens with him working hard to better the lives of people by fixing machinery that has broken down. He has to solve the problems with whatever materials he can find. By modern standards his technological fixes are practically ancient technology, but the people of the day can’t manage them. They’ve been reduced to barbarism by the depopulation of their world, even while powerful technology still exists to permit space travel.

In another short story that I once read (the title and the author’s name escape me), an intelligent man learns an ancient secret, long forgotten by mankind. It seems that no one in his day and age can read, write, or even perform simple mathematics. They have machines to do it all for them. As he’s working with a simple calculator he notices that the images on its display always seem to be the same whenever the calculator is performing the same

calculations. From this he works out a system of arithmetic that can be performed *without a calculator*. This is such a radical idea in his society that he's persecuted for it, until the military learns about it. *They* want to exploit it because calculations can be made without electronic machinery and that means without being traced by the enemy.

These aren't new ideas. Like I mentioned, I read these last two stories decades ago (actually, close to 40 years ago). I first read *The Timegod* almost 20 years ago. I've wanted to understand the technology I use throughout most of that time, particularly computers.

In the summer of 2003 I quit working at Microsoft. Before I left, I purchased a copy of Charles Petzold's book *Code: The Hidden Language of Computer Hardware and Software* at the company store. I read it fairly quickly and was amazed at how simple the basic concepts behind computers and automation really are. At the top of page 237 Charles Petzold says "I've mentioned several times that all the hardware to build these devices was available over a hundred years ago." He wrote the book in 1999 and published it in 2000. The hardware he was talking about might look like this.

Western Electric Relay – c. 1880s ©The Telegraph Office

That's a telegraph relay. The reason the telegraph relay exists is to extend the practical range of telegraphy. Despite being a good conductor of electricity, copper wire still has some resistance to the flow of electricity. The longer the circuit, the more resistance the wire has. Over long distances the current becomes so weak that it simply can't drive the electromagnet that activates the sounder in the telegraph receiver. The relay works by using electromagnetism to close a switch in another circuit with a separate power supply. When a telegraph signal passes through the electromagnet it pulls on the metal in the switch, closing the other circuit, and allowing power to flow through it. When the signal stops the electromagnet releases the switch, opening it, and stopping power from flowing

through the other circuit. You've probably got similar circuitry in your car enabling you to turn your headlights on and off without receiving a massive shock every time you do it.

The book tells us how it is possible to build a computer using telegraph relays. On page 203, after a lengthy discussion of feedback circuits and flip-flops we've can conceptually build 65,536 bytes of random access memory using nothing but telegraph relays and wire. We also run into this paragraph (emphasis mine)

> *As I implied earlier, 64 KB was a common amount of memory in personal computers purchased around 1980, although it wasn't constructed from telegraph relays. But could you really build such a thing using relays?* ***I trust you won't consider it.*** *Our design requires nine relays for each bit of memory, so the total 64K x 8 RAM array requires almost 5 million of them!*

and on page 237 (again)

> *Building a 64-KB RAM array from five million telegraph relays would have been as absurd one hundred years ago as it is now.*

I'm not afraid of a little absurdity, and I took Mr. Petzold's words as a bit of a challenge. I spent the next three months trying to write a software simulation of a computer built from telegraph relays in C#. My first efforts were fairly crude, clunky and slow (not unlike real telegraph relays). I did manage to create a very simple simulation that worked with 256 bytes of simulated ram, an address counter and an adder. About a year later I tried it again and had a crudely working simulation. Then in 2007 I managed to write a complete implementation of the computer described in chapter 17 of the book. That took me a couple of weeks. Six months later, in 2008 I rewrote the entire thing in C++. Sometime later, the computer that I was using to write this software crashed, and I lost the source code for it all.[†]

On February 9, 2014 I started yet another version of the same project, this time as an open-source project in Java. I finished working on it on February 10, 2015. The project page for that version, including the source code and a downloadable executable .jar file is available at https://sourceforge.net/projects/electromechanicalcomputer/.

I had planned to add more features to the simulation, including "hardware" multiply and divide instructions, and the ability to load programs from disk, features that were in the 2007 version of the code, but I lost interest for a while.

Last week, I found an old archive of the original source code. I spent some time over the weekend updating it. That version of the code included some shift and rotate instructions, but as I looked at what they actually did I realized that they weren't particularly useful. I

spent a day rethinking them, and re-implemented them to emulate the rotate instructions from the 8080 microprocessor. Then I spent some time working on a couple of new multiplication programs for the simulated computer. These programs are considerably faster than the multiplication by repeated addition programs I wrote for the 2007 implementation. I'll be posting the updated sources for both the C++ version (updated so that it can compile in Visual Studio 2015) and the C# version (updated so that it can compile in Visual Studio 2015 AND with the new instructions) here in the next couple of days.

I'm working on an overhaul of the instruction set for the computer. I'm going to take the now existing instruction set and change the binary codes for them to match the equivalent instructions on the 8080. Then I'll have to re-assemble the programs I wrote for the simulation based on the updated instruction set. That will give me room in the instruction set to implement a few more features that I've already started working on, including multiple registers for even faster programs.

---

[*] Despite popular notions, the word Apocalypse didn't originally mean the end of the world, or have anything the do with catastrophe. Instead, the word meant "revelation". It likely took on the currently accepted meaning because much of the content of John's Apocalypse (The book of Revelation in the Bible) has to do with the final events of this world.

[†] Well, I *thought* I lost it all. As I describe in this post, I found an old archive I had of it just a few days ago.