Building a MultiComp-based Z80

Dec 2015

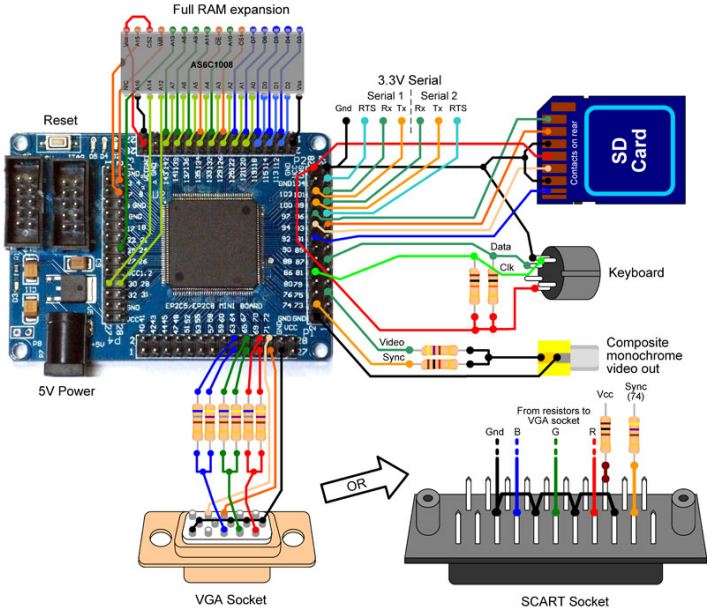
Let's see if we can build an FPGA-based system. The biggest challenge is that everything involved is completely different from software and hardware design. *It's a new ball game!*

We will need:

- some hardware (evidently, an FPGA is still a chip, just a different one)
- a design, this is essentially the schematic and wiring diagram
- a software tool chain to "synthesise" the design into a "bit stream"
- a way to upload that bit stream into the FPGA
- a connection between the FPGA and the outside world
- a healthy dose of curiosity, concentration, and time...

There are too many unknowns and new aspects to this to tackle it all at once. Fortunately, there are some nice projects on the web to help us forward with relatively little effort.

One of these is Grant Searle's amazing MultiComp "pick and mix" computer:

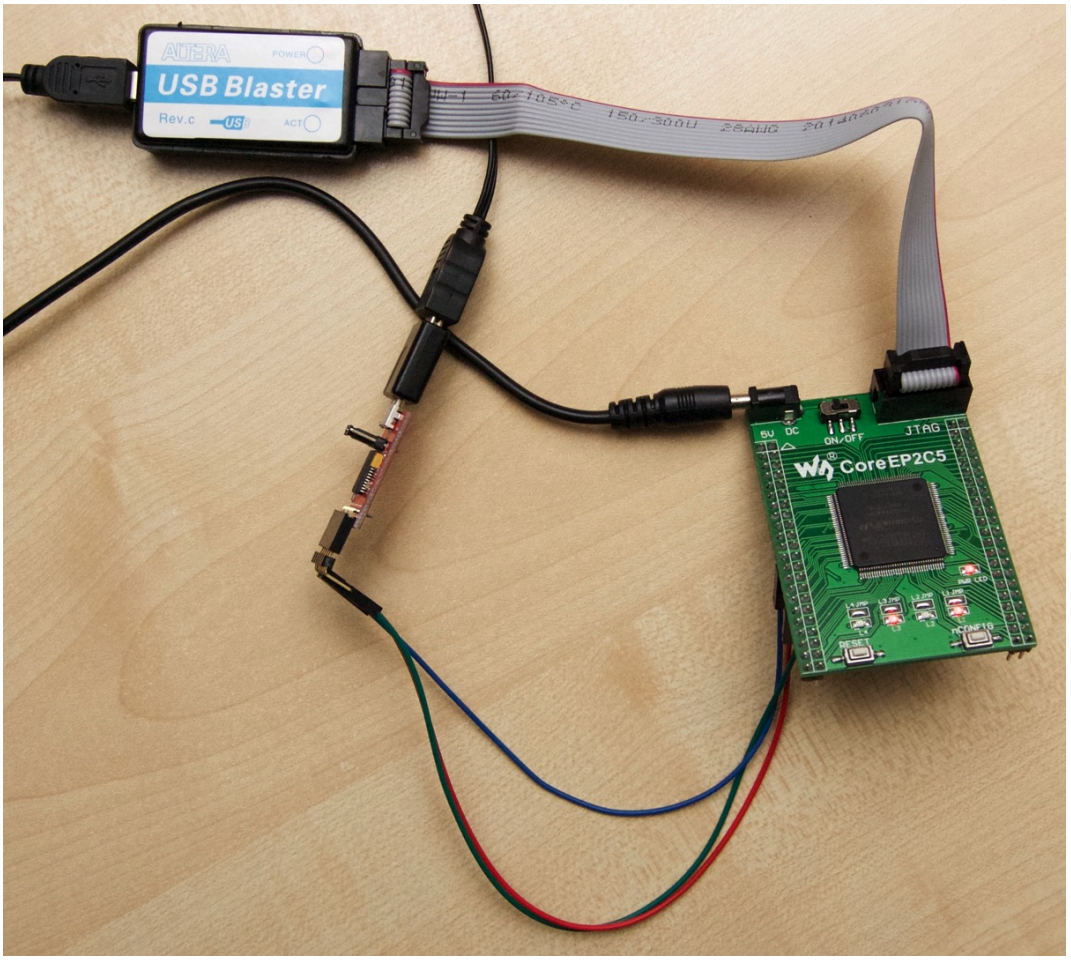


It's a fascinating example of the flexibility and richness of an FPGA, which requires only a few low-cost parts (all available from eBay, AliExpress, etc).

MultiComp can re-create an 8-bit 6502, 6809, **or** Z80 microcomputer system from the 1970's era, including multiple serial interfaces, a PS2 keyboard interface, various types of video out, plenty of RAM memory, and lots of permanent storage on an SD or µSD card.

The beauty of Grant's scheme, and the web pages he has set up with all the information you need, is that you can start with an absolutely minimal setup to test the waters and figure out all the issues listed above. The design is written in VHDL, which is one of the two main languages used for constructing FPGA designs (the other being Verilog).

This is all it takes to get to a working configuration:



In this case an EP2C5 Altera FPGA board from WaveShare, a USB Blaster to program that board, and an FTDI serial interface to talk to the final configuration. Oh, and 5V power. If you look around on eBay, you'll find several alternatives for a total cost of under €25.

There are - even with this minimal setup - still *many* choices to be made. So let's pick this:

- we will implement a Z80 CPU
- 4 KB RAM, stored inside the FPGA
- 8 KB ROM, stored inside the FPGA
- one serial port, running at 115200 baud

After a day of following Grant's instructions, figuring out all the details, getting the free "Web Pack 13.0 sp1" IDE from Altera going (requires Windows), this will be the result:

```
Z80 SBC By Grant Searle

Memory top?
Z80 BASIC Ver 4.7b
Copyright (C) 1978 by Microsoft
3763 Bytes free
Ok
```

Which, frankly, should make you fall off your chair (with a soft landing, please!). Without any soldering, *thousands* of logic elements have been tied together (via bits!) to implement a full computer not unlike the Altair and Imsai systems which launched the term "Hobby Computer" about four decades ago. It's a shame that nothing of what's going on is *visible!*

Not only does it work, it can run several times faster than the original and draws 75 mA.

By building this, we will have skimmed very lightly over all the intricacies of VHDL. There really *is* a lot of logic inside this design. It has merely been packaged to get going quickly.

But why stop here? Let's build a full-scale CP/M system, the predecessor of MS-DOS!

For simplicity's sake, we'll stick to the serial port for now. That's the beauty with FPGAs: you can extend and improve your setup gradually. *Lots of spare logic cells to play with!*

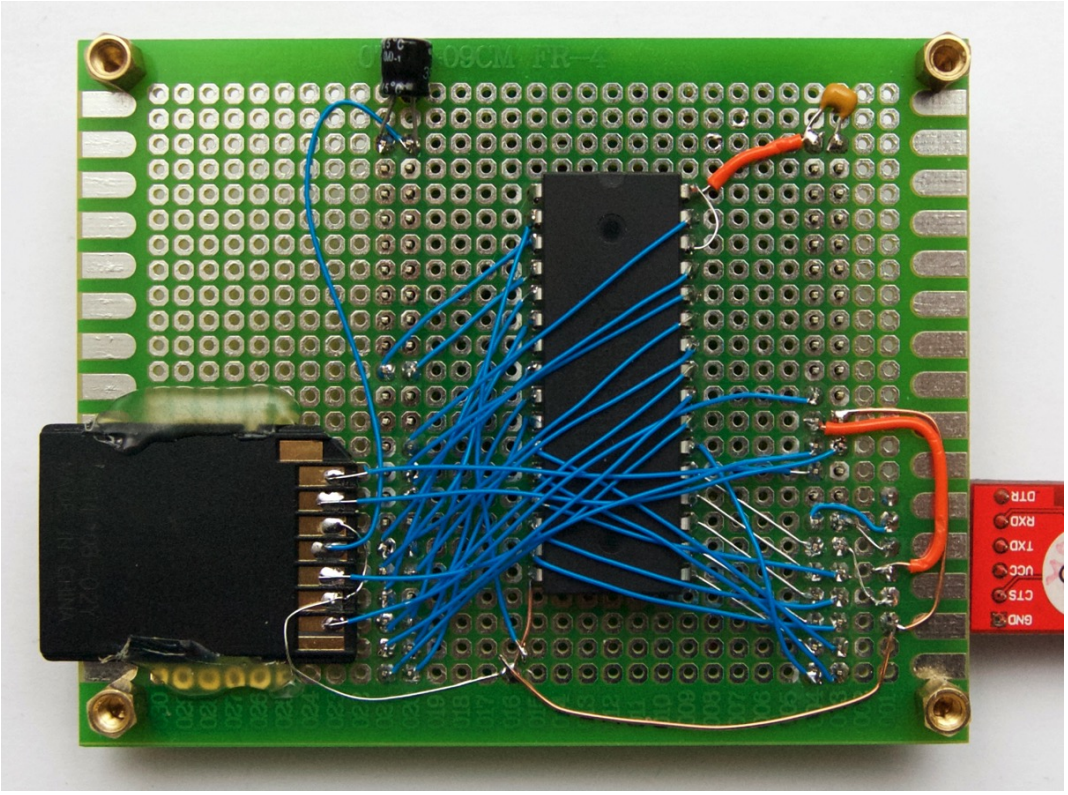
To run CP/M, we need more RAM, and we need some permanent storage. Instead of those old clunky floppy disk drives, Grant has set up an interface to an SD (or µSD card). Which is not only infinitely smaller, and considerably faster, it also has immense storage capacity compared to floppies. For now, we'll get *sixteen* 8 MB "drives", called A: to P: (so now you know where Windows got its conventions from...). Note that the SD card is not formatted as a "VFAT" disk we can read elsewhere - it's used as 16 x 8 MB of raw disk blocks.

Let's add the RAM and SD card, using point-to-point wiring since this is a one-off project.

The RAM is a single 32-pin PDIP chip, containing 1 Mbit (128K x 8) of static RAM, which is easy to interface. Search for "SRAM PDIP32 AS6C1008-55PCN" on eBay, for example. We'll only use half of that RAM, since the Z80 can't address more than 64 KB at once.

For the SD card, we'll actually use a µSD-to-SD adapter, as this makes a perfect "socket":

We can simply solder it in, and use it to insert a 2 GB µSD - it even has a "lock" switch!



All wired up, and fastened with hot glue. Note the SD adapter sticking out for easy access. There are also some extra capacitors, to help avoid spikes and supply issues. Just in case.

Here is the result of all that hard work, again following Grant's Z80 CP/M instructions:

```
Press [SPACE] to activate console

CP/M Boot ROM 2.0 by G. Searle

BC or BW - ROM BASIC Cold/Warm
X        - Boot CP/M (load $D000-$FFFF)

:nnnn... - Load Intel-Hex file record
Gnnnn    - Run loc nnnn

>b
Cold or warm?

Memory top?
Z80 BASIC Ver 4.7b
Copyright (C) 1978 by Microsoft
52755 Bytes free
Ok
```

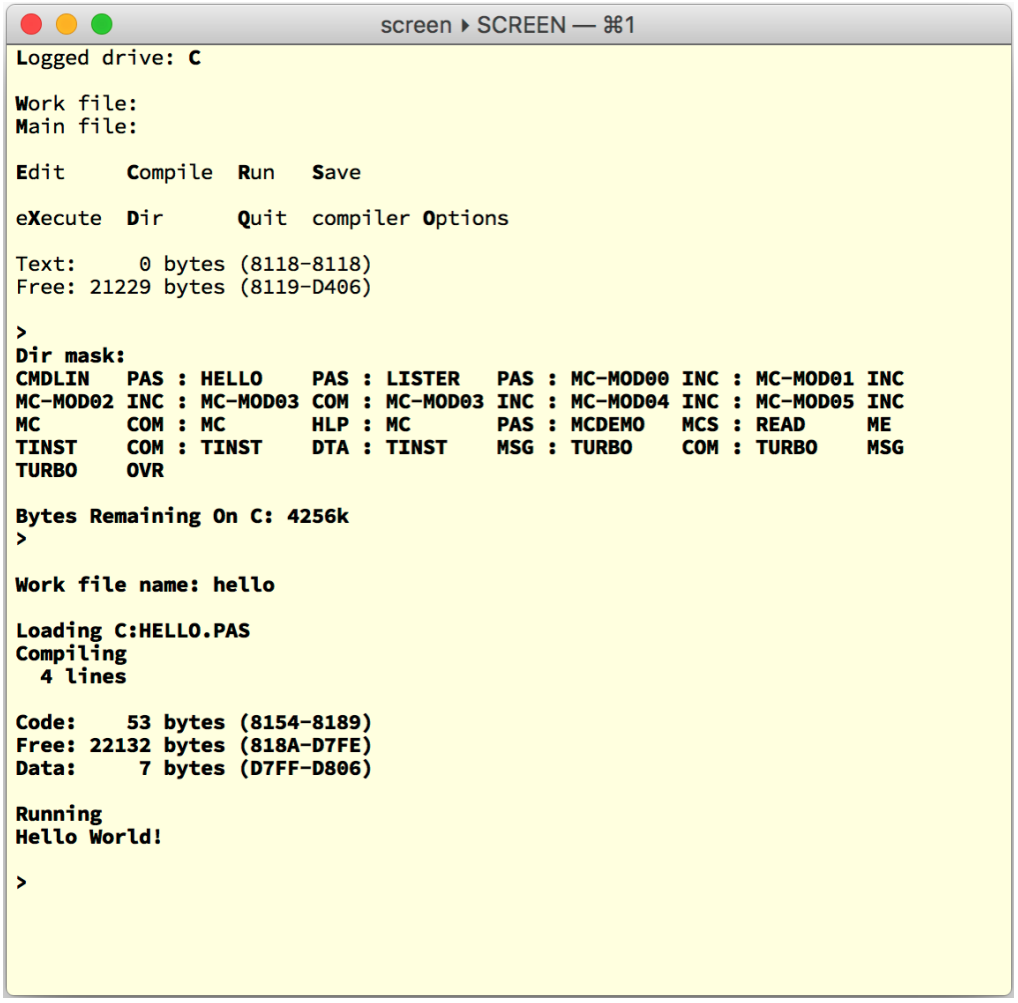
And then, jumping through a few more hoops to get the CP/M system loaded onto that µSD card (lots of chickens and eggs, all nicely resolved when following the instructions), we get:

```
Press [SPACE] to activate console
[...]
>x
Boot CP/M?
Loading CP/M...
CP/M BIOS 2.0 by G. Searle 2013

CP/M 2.2 (c) 1979 by Digital Research

A>type files.txt
```

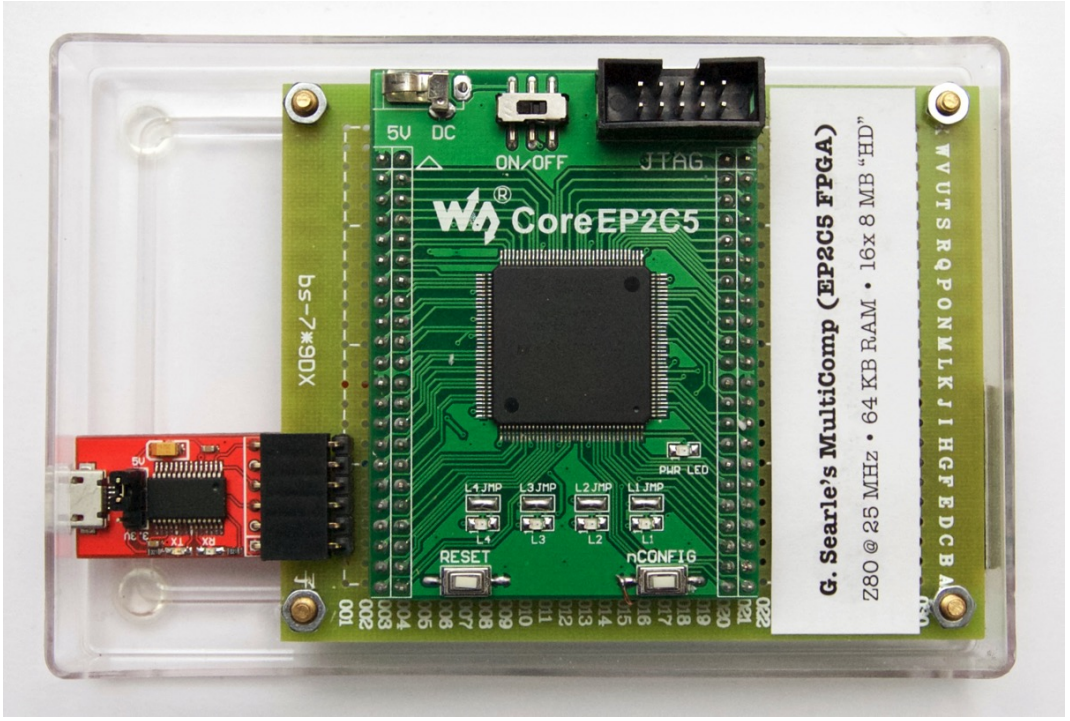
```
A:               B:               C:               D:
------------------------------------------------------------------
0                0_GAMES          0_OLDUTILS       0_NEWUTILS
1_BDS_TINY_C     1_MUMATHSIMP     1_F80M80BASIC    1_ROMS
2_APL            2_CROSSTALK      2_AZTEC_C_106D   2_ZSYSTEM
3_JANUS_ADA15    3_QTERM43        3_TPASCAL3       3_MICROPRO
4_MS_COBOL       4_CLINK          4_DXFORTH401     4_MULTIPLAN
5_PILOT          5_SUPERSFTUTIL   5_PLI14          5_DBASEII
6_SYSLIB         6_RCPM           6_ALGOLM         6_DWG_APPS
7_BBC BASIC      7_DDTZ SOURCES   7_SUPERCALC      8_MICROSHELL
------------------------------------------------------------------

A>
```

This is running from an SD disk image by Oscar Vermeulen, who built this setup and then built an image with lots of CP/M-era software - see his Obsolescence Guaranteed page.

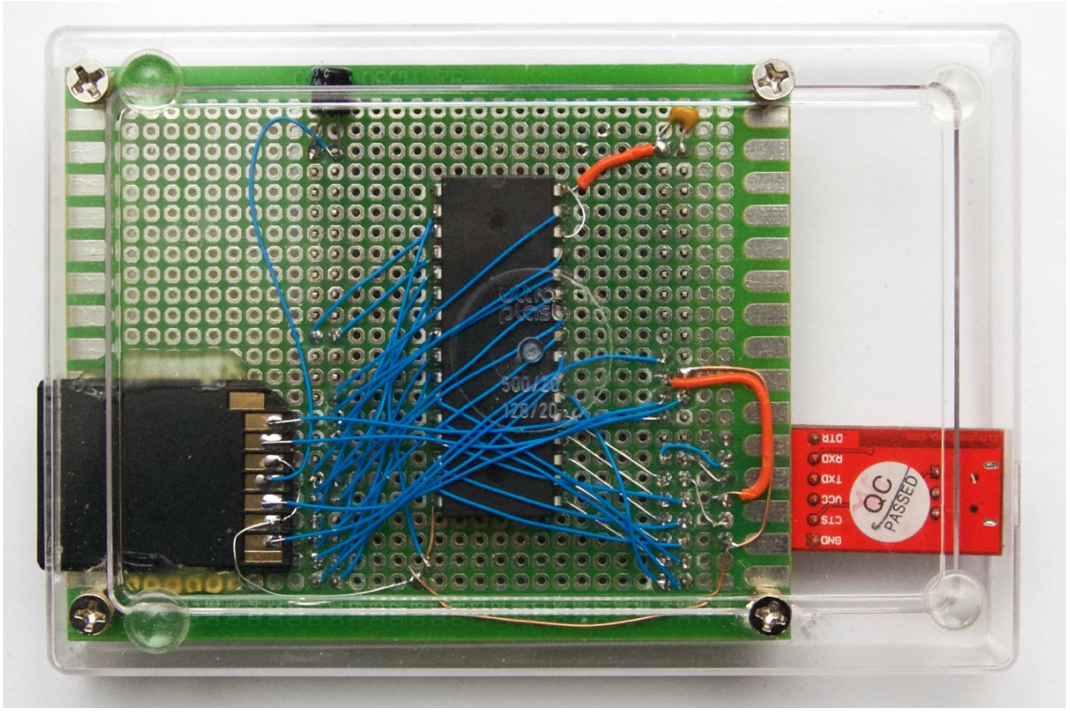
Here's Turbo Pascal 3.0, one of the miracle tools for the 64 KB world of the 8-bit era:



And here is the final build, fitted into a nice geek-friendly clear plastic box:



Finally, once again, the underside with all the gory details:



As noted on the box: the Z80 "soft core" is running at 25 MHz, over 6x the original 4 MHz. If you prefer a 6502 setup or a 6809: simply upload a different bit stream to the FPGA ...

Yes, things really have changed over a period of 4 decades!