The '64' in Z80SBC64 refers to Altera 7064, but as many Z80 SBC that must had been built over the years, this can easily be the 64th Z80SBC. So what interesting feature is there after all these Z80SBC? I think ROM-less, serial bootstrap may be interesting or at least not commonly done. Unlike its big brother, Z80SBCRC, this design is hobbyist friendly with through-hole components and an easy-to-solder surface mount connector.

The features of Z80SBC64 are:
* 20MHz Z80
* ROM-less
* 128K RAM in four 32K banks, battery-backed
* Altera EPM7064S CPLD in PLCC44 package
* Serial bootstrapping
* Hardware serial receive function and software serial transmit (bit bang) function
* Serial port operates at 115200 N81, no handshake
* Compact flash interface
* CP/M 2.2
* 100mm x 50mm base board
* RC2014 expansion bus

More information on Z80SBC64 here:
https://www.retrobrewcomputers.org/doku.php?id=builderpages: plasmo:z80sbc64

I think this Z80 SBC can be built for $40 and if you are willing to "take a walk on the wild side" and buy parts from eBay, it can be built for lesser than $15.

Bill

---

Very nice, where did you locate the 22mhz oscillator? I assume it's pretty critical for correct Baud rates?

- Gary

---

I think this Z80 SBC can be built for $40 and if you are willing to "take a walk on the wild side" and buy parts from eBay, it can be built for lesser than $15.

Bill
Yes, the 16x baud clock is derived from the processor clock so there is a fixed relationship (12x) between them. The exact frequency is 22.1184MHz and you can find that fairly easily on eBay. I have a whole bag of 22MHz oscillator and it is within 0.5% of actual value so I use that instead. You can also run the processor at 7.3728MHz and the serial port at 38400 or 11.059Mhz with serial port at 57600. The 12x divider is in CPLD which can be easily change to other values, but because the transmit is bit banging in software, the delay loop need to be re-calculated for different divider values.

Bill

---

Subject: Re: Z80SBC64, yet another Z80 SBC
Posted by plasmo on Tue, 18 Dec 2018 05:11:00 GMT

ASCII art benchmark running in MBASIC80 in CP/M2.2. It took 52 seconds to complete, no discernible difference in performance between bit-bang transmitter and hardware serial transmit.

The ZEXALL.COM instruction test passed 100% @22MHz in 35 minutes.

The compact flash interface is surprisingly robust. It is able to run every brands of CF disks I have, something like 18 CF disks of different capacities and brands.

I'm porting banked CP/M 3 to it. It only has two banks, one 64K and the other 32K (the last 32K is for bootstrap monitor). The banked CP/M 3 loads and seems to run OK, but I got this gibberish sign-on message and an error message "BDOS ERR:" embedded in it. Need to look into it tomorrow.

Bill

b>mbasic80 asciiart.txt
BASIC-80 Rev. 5.21
[CP/M Version]
Copyright 1977-1981 (C) by Microsoft
Created: 28-Jul-81
32824 Bytes free
0000000111111111111111112222223347E7AB322222111100000000000000000000000000
00
000001111111111111111122222233357BF7543322221111100000000000000000000000
00
0001111111111111111122222233334469                 D   63221111110000000000000000
011111111111111122222233334457DB                    85332111111100000000000000000
011111111111111122222233334469 D978 BCF    DF9 6556F4221111100000000000000000
01111111111111111122222333334469 11111111111111111122222333334469 E7AB32222211110000000000000000
00
000111111111111111112222223333445C 6433322221111110000000000000000000000000
01111111111111111111111122222233334456C 65443332221111110000000000000000000000000
1111111111111111111122222333344556C 65443332221111110000000000000000000000000
111111111111111122222333334469 D978 BCF    DF9 6556F4221111100000000000000000
11111111111111111122222333334469 11111111111111111122222333334469 E7AB32222211110000000000000000
00
000111111111111111112222223333445C 6433322221111110000000000000000000000000
01111111111111111111111122222233334456C 65443332221111110000000000000000000000000
1111111111111111111122222333344556C 65443332221111110000000000000000000000000
111111111111111122222333334469 D978 BCF    DF9 6556F4221111100000000000000000
11111111111111111122222333334469 11111111111111111122222333334469 E7AB32222211110000000000000000
00
000111111111111111112222223333445C 6433322221111110000000000000000000000000
01111111111111111111111122222233334456C 65443332221111110000000000000000000000000
1111111111111111111122222333344556C 65443332221111110000000000000000000000000
111111111111111122222333334469 D978 BCF    DF9 6556F4221111100000000000000000
11111111111111111122222333334469 11111111111111111122222333334469 E7AB32222211110000000000000000
00
b>zexall
Z80 instruction exerciser
<adc,sbc> hl,<bc,de,hl,sp>.... OK
add hl,<bc,de,hl,sp>......... OK
add ix,<bc,de,ix,sp>......... OK
add iy,<bc,de,iy,sp>......... OK
aluop a,nn.................. OK
aluop a,<b,c,d,e,h,l,(hl),a>.. OK
aluop a,<ixh,ixl,iyh,iyl>.... OK
aluop a,(<ix,iy>+1)........... OK
bit n,(<ix,iy>+1)............ OK
bit n,<b,c,d,e,h,l,(hl),a>.... OK
cpd<r>...................... OK
cpi<r>...................... OK
<daa,cpl,scf,ccf>........... OK
<inc,dec> a............... OK
<inc,dec> b............... OK
<inc,dec> bc............... OK
<inc,dec> c............... OK
<inc,dec> d............... OK
<inc,dec> de............... OK
<inc,dec> e............... OK
<inc,dec> h............... OK
<inc,dec> hl............... OK
<inc,dec> ix............... OK
<inc,dec> iy............... OK
<inc,dec> l............... OK
<inc,dec> (hl)..............  OK
<inc,dec> sp..............  OK
<inc,dec> (<ix,iy>+1)....... OK
<inc,dec> ixh.............  OK
<inc,dec> ixl.............  OK
<inc,dec> iyh.............  OK
<inc,dec> iyl.............  OK
ld <bc,de>,(nnnn).........  OK
ld hl,(nnnn)...............  OK
ld sp,(nnnn)...............  OK
ld <ix,iy>,(nnnn).........  OK
ld (nnnn),<bc,de>.........  OK
ld (nnnn),hl..............  OK
ld (nnnn),sp..............  OK
ld (nnnn),<ix,iy>.........  OK
ld <bc,de,hl,sp>,nnnn.....  OK
ld <ix,iy>,nnnn..........  OK
ld a,(bc),(de)............  OK
ld <b,c,d,e,h,l,(hl),a>,nn...  OK
ld (<ix,iy>+1),nnn.......  OK
ld <b,c,d,e>,(<ix,iy>+1).... OK
ld <h,l>,(<ix,iy>+1)....... OK
ld a,(<ix,iy>+1)..........  OK
ld <ixh,ixl,iyh,iyl>,nn...  OK
ld <bcdehla>,<bcdehla>...... OK
ld <bcdexya>,<bcdexya>...... OK
ld a,(nnnn) / ld (nnnn),a...  OK
ldd<r> (1).................. OK
ldd<r> (2).................. OK
ldi<r> (1).................. OK
ldi<r> (2).................. OK
neg........................  OK
<rrd,rld>................... OK
<rlca,rrca,rla,rra>........  OK
shf/rot (<ix,iy>+1)........  OK
shf/rot <b,c,d,e,h,l,(hl),a>.. OK
<set,res> n,<bcdehl(hl)a>.... OK
<set,res> n,<ix,iy,+1>...... OK
ld (<ix,iy>+1),<b,c,d,e>..... OK
ld (<ix,iy,+1),<h,l>......... OK
ld (<ix,iy,+1),a............ OK
ld <bcd,de>,a............  OK
Tests complete
b>^C
b>^C
b>a:
a>cpmldr
Boot LDRBIOS

CP/M V3.0 Loader
Copyright (C) 1982, Digital Research

BDOS ERR:
BNKBIOS3 SPR F300 0500
BNKBDOS3 SPR 7700 0900
RESBDOS3 SPR ED00 0600
BNKBDOS3 SPR 4900 2E00

59K TPA
Copyright 1979 (c) by Digital Research
CP/M 3 for Z80SBCRC, no RAMdisk 11/9/18 Banked version
A>dir
A: HELP COM : HELP HLP : HEXCOM COM : HIST UTL : HEXCOM CPM
A: PIP COM : PORTS LIB : RANDOM ASM : README 1ST : RESBDOS3 SPR
A: RAMAC COM : SCB ASM : SETDEF COM : SHOW COM : SID COM
A: BDOS3 SPR : BIOSKRNL ASM : BNKBBDOS3 SPR : CALLVERS ASM : LDRBIOS REL
A: CBIOSS REL : SUBMIT COM : TYPE COM : CPMLDR SYM : BNKBIOS3 SPR
Subject: Re: Z80SBC64, yet another Z80 SBC
Posted by plasmo on Sat, 02 Feb 2019 15:18:23 GMT

The problem with Z80 running at 22MHz is compatible peripherals are hard to find. So while Z80SBC64 is electrically compatible with RC2014 bus, I have to replace the 22Mhz oscillator with 7.37MHz oscillator to be actually compatible with the various RC2014 boards out there. 22MHz Z80 is responsive and it should be adequate to run C programs and has an operating system like Fuzix. Native CP/M C compiler like Hitech 3.09 runs quite quickly on 22MHz Z80 so it is a shame to reduce its performance to 1/3.

The alternative is to develop new set of RC2014 boards that can run at 22MHz and backward compatible to 7.37MHz. To that end, I redesign Z80SBC64 into a 102mmX102mm motherboard with 3 RC2014 connectors and name it Z80MB64. All the software that ran on Z80SBC64 will also run on Z80MB64. Built a qualification lot of 6 Z80MB64 and they all work, so I think the design is good.

https://www.retrobrewcomputers.org/doku.php?id=builderpages:plasmo:z80mb64

Bill

File Attachments
1) Z80MB64_qual_lot_of_6.jpg, downloaded 793 times

Subject: Re: Z80SBC64, yet another Z80 SBC
Posted by etchedpixels on Mon, 04 Feb 2019 14:43:14 GMT

Possibly a dumb question but I'm mostly a software guy. Would it be practical to run 1/3rd clock into the RC2014 motherboard and generate some wait states on I/O cycles (either all of them or better yet all but the onboard I/O) ?

Subject: Re: Z80SBC64, yet another Z80 SBC
Posted by plasmo on Mon, 04 Feb 2019 16:04:58 GMT

Since the board already has all the RAM & ROM it needs, the main issue with 22MHz bus clock is the setup/access/hold time to I/O modules. Access time can be resolved by adding wait state, setup and hold time are more difficult. Interrupt timing and daisy chaining are also an issue. Depending on how a particular board is implemented, it may work or partially work with 22MHz bus clock, but it is a try-and-error process.
Subject: Re: Z80SBC64, yet another Z80 SBC
Posted by mikemac on Mon, 04 Feb 2019 16:19:11 GMT

Dumb question: you describe this as having 3 RC2014 ports (P5, P6, P7). What's P1? It looks like a fourth one unless there's some really hard to see difference on the schematic. Is P1 for a different CPU card?

Subject: Re: Z80SBC64, yet another Z80 SBC
Posted by plasmo on Mon, 04 Feb 2019 16:32:13 GMT

P1 is known as a "mistake" ;-) It was to be the 4th RC2014 connector, but it is too close to the CF adapter. I may figure out a use for it, turning the mistake into a "feature", so I'm keeping it as is. Bill

Subject: Re: Z80SBC64, yet another Z80 SBC
Posted by mikemac on Mon, 04 Feb 2019 16:47:57 GMT

Whoops! Sorry about that. Didn't mean to embarrass you in front of the group.

You are getting pretty good at sticking those Altera 7000S chips in everywhere. I see you've gotten rid of the external serial port and sucked it into the 7000S too. The earlier 68K design I look at in detail still had the external UART.

And using that DS1210 for battery backup of RAM is cute. It took me a few head scratches to figure out what it was doing until I noticed the greyed out "32 BAT" pin!

Keep up the good work!!! And I'm in awe of how many projects you manage to keep in the air at one time!

Subject: Re: Z80SBC64, yet another Z80 SBC
Posted by plasmo on Mon, 04 Feb 2019 17:00:49 GMT

No problem, mistakes are like gravity, the bigger the project, the bigger the pull of mistakes. You may remember this photo from my early Tiny68K effort. Now that's a mistake, hard to make a 'feature' out of it!
Subject: Re: Z80SBC64, yet another Z80 SBC
Posted by mikemac on Mon, 04 Feb 2019 17:58:09 GMT

Smaller PCB!!!

Subject: Re: Z80SBC64, yet another Z80 SBC
Posted by etchedpixels on Mon, 04 Feb 2019 18:58:52 GMT

And I thought it was the one you soldered to the back of the board so that with the board vertical it fitted over the hard disk bays

Alan