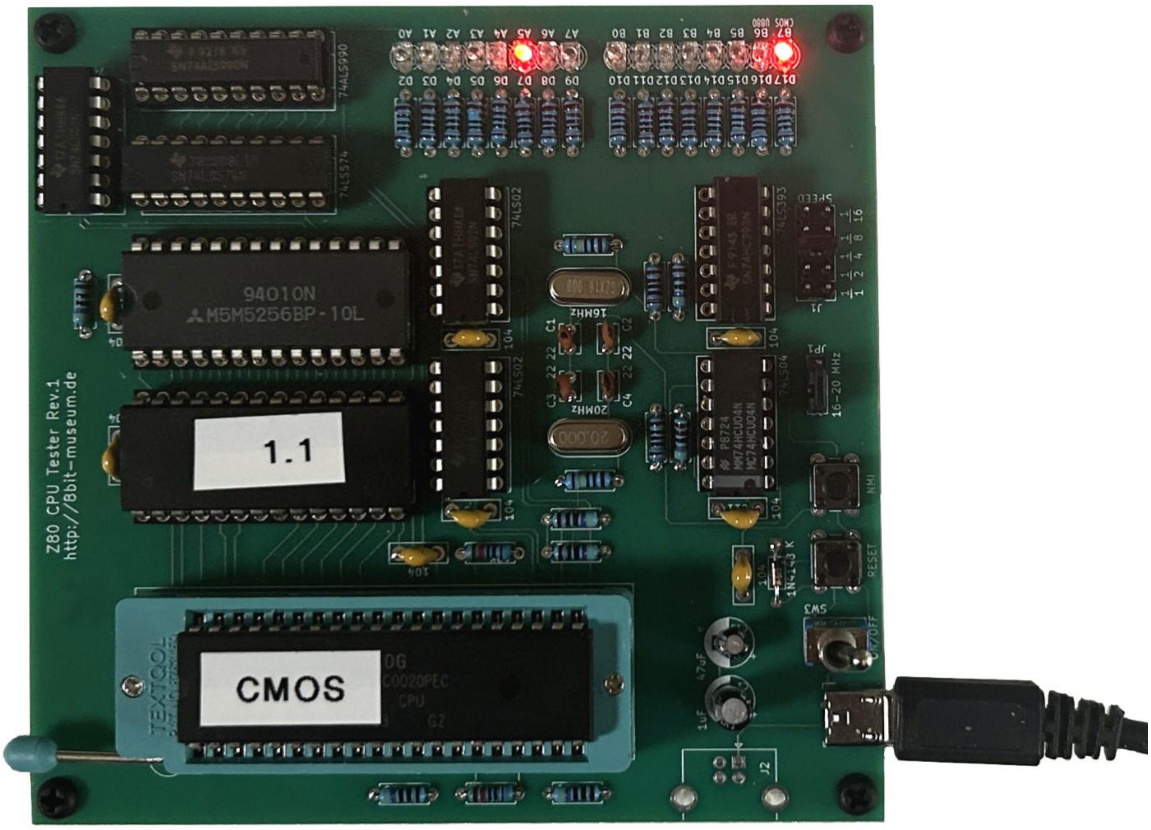
Version: 2024-07-24

Z80 CPU TESTER

http://8bit-museum.de

Manual



You use the hardware and software at your own risk. No liability is accepted for any damage that may arise when using the software and hardware. No guarantee is given for the correct functionality of the functions provided. There is also no entitlement to a bug fix, even if the developer tries to fix bugs as quickly as possible.

This page is intentionally left blank

Table of Contents

This page is intentionally left blank

1 Usage

1.1 Introduction

With the Z80 CPU Tester, Z80 CPUs and compatible CPUs can not only be tested, but it also tries to detect the CPU manufacturer and technology (NMOS/CMOS).

It tries

- to identify the CPU technology (NMOS/CMOS),
- to identify the CPU manufacturer, and
- to test whether the CPU is still functional

and helps to find out the maximum possible CPU frequency.

The following is not tested:

- Interrupts,
- WAIT and HALT inputs signals,
- M1 and RFSH output signals, and
- BUSRQ and BUSACK signals

The hardware has following features:

- 32kb EPROM (27C256), alternatively an EEPROM can be used (AT29C256)
- 32kb SRAM
- output ports (1x unidirectional, 1x read-back) with 16 LEDs
- RESET and NMI button
- 16 MHz / 20 MHz switchable
- 1, 1/2, 1/4, 1/8, 1/16 multiplicator (1-16 MHz / 1.25-20 MHz)
- USB powered
- THT for easy soldering

1.2 Repository and Licenses

The Gerber files for creating the board and the required software are available on Github. Alternatively, the board can also be obtained from me.

The repository can be found at

https://github.com/slabbi/Z80-CPU-Tester

Regarding the licenses:

- The hardware can be used freely for private purposes. However, commercial use is prohibited.
- The software, if it was written by me, is subject to the permissive MIT license. However, third-party code may use other licenses, so please note the information in the source code.

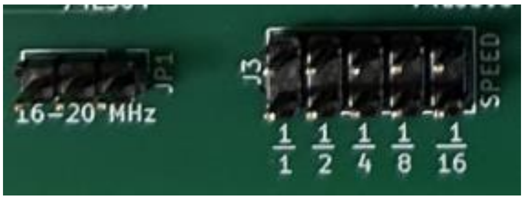
1.3 Performed tests

Following tests are performed:

- memory access, load 16 bit register load
- RL, RR, load register to register
- ADD, SUB
- PUSH, POP, SBC
- ADD, SUB, INC, DEC
- some 16 bit multiplications (ADD, RL)
- some 32 bit multiplications (ADD, ADC, SBC, PUSH, POP, RRA, EX)
- calculates some square roots (BIT, RL, ADD, SUB)
- plays Towers of Hanoi (PUSH, POP, CALL)
- calculates Pi (EX, EXX, IX, IY, INC, DEC, ADD, ADC, SBC, SRL, RR, PUSH, CALL)

Pi is calculated to 100 digits. This takes about 30 seconds at 4 Mhz. When you test CPUs at lower speeds the number of calculated digits should be reduced.

1.4 Setting the clock frequency



Two jumpers must be set:

The clock source is selected on the left (“16-20 MHz”). If the jumper is in the left position, the 16 MHz crystal is selected, and the 20 MHz crystal is selected on the right (“Speed”).

The clock frequency divider is selected on the right (“Speed”). It can be divided down to 1/16 of a clock, i.e. you get frequencies of 1, 2, 4, 8, 16 MHz or 1.25, 2.5, 5, 10, 20 MHz.

1.5 Display of test results

There are two ports (A and B) that are used to display status and results using 16 LEDs.

1.5.1 Output “Port B”

Port B displays the identified technology and manufacturer:

```
CU00tttt (C = CMOS, U = UB880, tttt = type)
```

CMOS-LED (C)

The LED displays the result of the “OUT (C),0” test.

- When "on", a CMOS CPU has been detected (OUT (C),0 outputs $ff).
- When "blinking", a CMOS CPU has been detected (OUT (C),0 outputs $00).
- When "off", a NMOS CPU has been detected (OUT (C),0 outputs $00).

Note: The Sharp LH5080A (CMOS version of LH0080A) "fails" the CMOS test, and the undocumented OUT (C),0 instruction behaves the same way it does on NMOS CPUs. The LED may blink after all functional tests are completed; it will not blink when the functional tests are still in progress.

U880-LED (U)

The LED displays the result of the U880 test.

- When "on", a U880 CPU has been detected (U880, UA880, UB880 and similar), otherwise it is "off"

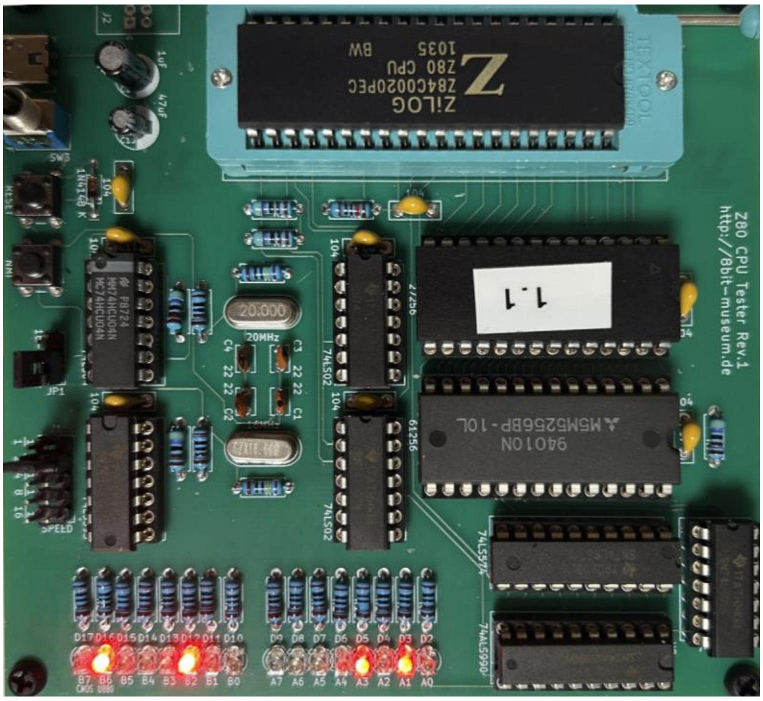
Type (tttt)

The lower four LEDs display the identified CPU. The test cannot guarantee 100% accuracy because the detection of the CPU is based on the evaluation of undocumented behavior of the CPU.

0000	not used
0001	Z180
0010	Z280
0011	EZ80
0100	U880 (newer; MME U880, Thesys Z80, Microelectronica MMN 80CPU)
0101	U880 (older; MME U880)
0110	SHARP LH5080
0111	NMOS Z80 (Zilog Z80, Zilog Z08400 or similar NMOS CPU, Mosstek MK3880N, SGS/ST Z8400, Sharp LH0080A, KR1858VM1)
1000	NEC D780C (NEC D780C, GoldStar Z8400, possibly KR1858VM1)
1001	KR1858VM1 (overclocked)
1010	Unknown NMOS Z80 Clone
1011	CMOS Z80 (Zilog Z84C00)
1100	Toshiba Z80 (Toshiba TMPZ84C00AP, ST Z84C00AB)
1101	NEC D70008AC
1110	Unknown CMOS Z80 Clone
1111	NEC Z80 Clone (NMOS)

Example

Zilog Z84C0020PEC from a Chinese marketplace. It is a U880 (running with 10 MHz):



The identified U880 displays 01000100 (Port B). “U” is set, “tttt” is 0100.

1.5.2 Output “Port A”

Port A displays the current status of the functional testing. While port B displays the result immediately, the functional testing takes some time, so port A displays the test number of the test progress.

When functional testing is completed:

- the number of the failed test is displayed with a blinking bit 7, or
- a running light shows a successful result (no error).

In the picture above, port A displays that the functional test no 10 is still in progress.

1.5.3 Input “Port A”

Port A can be used to read-back the value stored in the latch register of Port A.

1.5.4 Output XF results and XF/YF counters

The CPU identification is based on the evaluation of undocumented behavior of the XF/YF flags. The internal XY result and the XF/XF counters can be displayed by pressing the NMI button.

After pressing the NMI button, the CPU tester will - after a short time - blink three times (alternating "xxxxoooo ooooxxxx" patterns). Then consecutively displays

- the XFRESULT (Port A)
- the XFCOUNTER (Port B/A)
- the YFCOUNTER (Port B/A)
- the FLAGS result (Port A) for A = 0, F = FF / (result can be $00, $08, $20, $28).

The output will be pretty fast (depending on the CPU clock), so you should record the output with your mobile phone to analyze it later.

XFRESULT encoding

- A[7:6] - YF result of F = 0, A = C | 0x20 & 0xF7 (F.5 set, F.3 reset)
- A[5:4] - XF result of F = 0, A = C | 0x08 & 0xDF (F.3 set, F.5 reset)
- A[3:2] - YF result of F = C | 0x20 & 0xF7, A = 0 (F.5 set, F.3 reset)
- A[1:0] - XF result of F = C | 0x08 & 0xDF, A = 0 (F.3 set, F.5 reset)

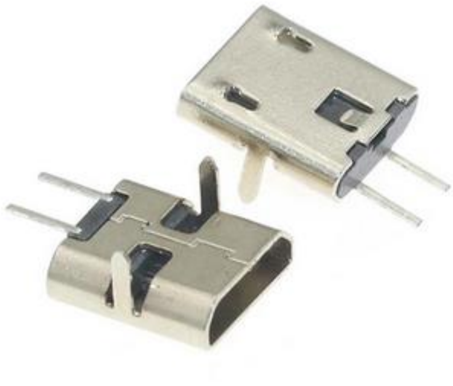
Where the result bits set as follows:

- 00 - flag always set as 0
- 11 - flag always set as 1
- 01 - flag most of the time set as 0
- 10 - flag most of the time set as 1

Note: YF aka F.5, XF aka F.3

2 Bill of materials (BOM) Addendum

2.1 USB-Socket



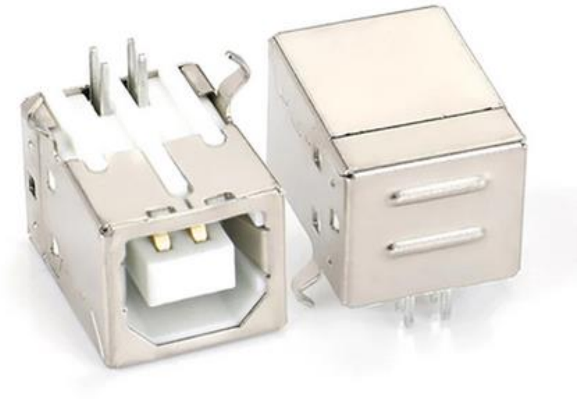
e.g. from AliExpress:

https://de.aliexpress.com/item/1005003018898705.html

https://de.aliexpress.com/item/1005001629294526.html

https://de.aliexpress.com/item/1005004836287679.html

https://de.aliexpress.com/item/4000838345540.html



e.g. from AliExpress:

https://www.aliexpress.com/item/32834923469.html