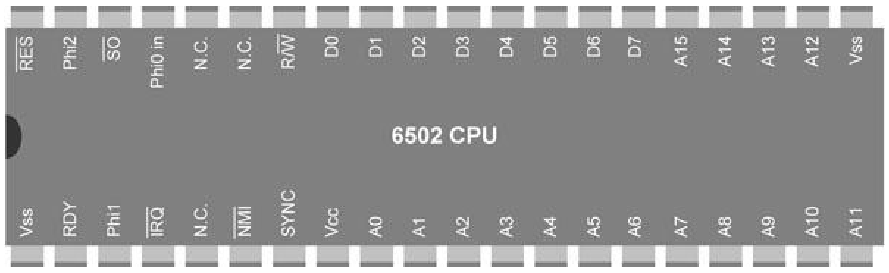


Grant's 8-chip (or 7-chip) 6502 computer

(only 7 or 6 chips if using a USB to TTL serial cable)

- a fully operational 6502 computer running BASIC can't get simpler than this!

by Grant Searle

For news and updates, follow me on Twitter:

Follow

Last update: 4th February 2014

*[FOR A **Z80** VERSION OF THE MINIMAL COMPUTER, CLICK HERE]*
*[FOR A **6809** VERSION OF THE MINIMAL COMPUTER, CLICK HERE]*

[FOR A VERSION ON A LOW-COST FPGA BOARD CLICK HERE]

Please note that you are NOT allowed to reproduce any of this page elsewhere on the Web without my permission.
WORK IN PROGRESS - REGULARLY UPDATED - PLEASE CHECK BACK IN A DAY OR TWO AND ENSURE YOU REFRESH YOUR BROWSER PAGE TO PICK UP LATEST VERSION. Thanks.

Specification

16K ROM
32K RAM
6502 Processor with a 1.8432MHz crystal (0.9216MHz or 1.8432MHz clock)
115200 Baud serial interface, RS232 specification voltage levels
Hardware serial handshaking ensures characters not lost when transferring programs from the PC to this board
Power consumption - 160mA
Microsoft BASIC
Minimal possible component count - 8 (or 7 if running the CPU at 1.8MHz) ICs and a small number of discrete components

Memory Map

0000-7FFF 32K RAM
8000-9FFF FREE SPACE (8K)
A000-BFFF SERIAL INTERFACE (minimally decoded)
C000-FFFF 16K ROM (BASIC from C000 TO DED3, serial routines FF00 to FFFF, so a large amount of free space suitable for a monitor etc)

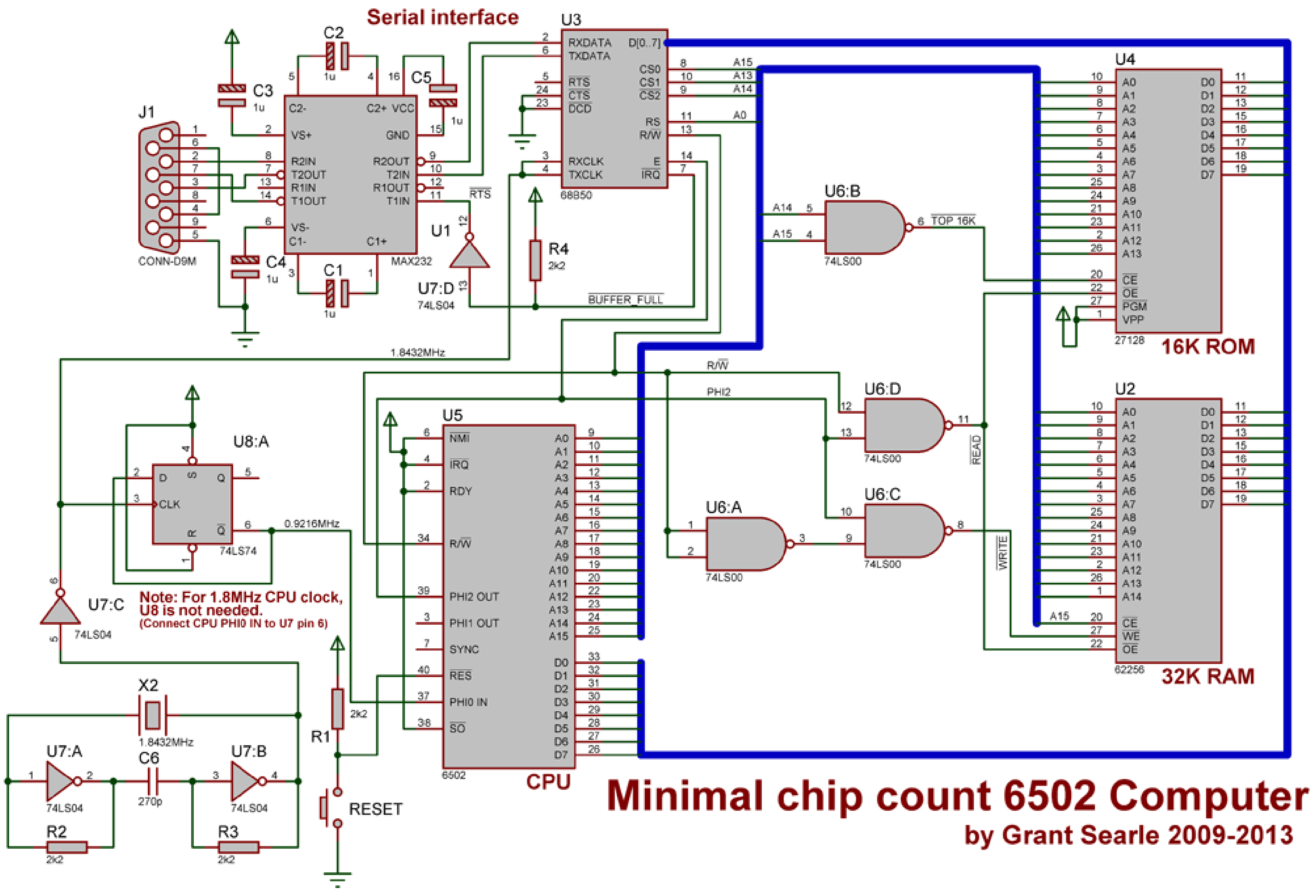
Circuit diagram

The purpose of this computer is to create the simplest possible machine with a high speed interface, good amount of RAM and also a good implementation of BASIC.
The design that I produced is shown here, and is probably the simplest 6502 circuit that can be done to fulfil what I need. This can be used as the basis of more complex machines.
The ACIA is very minimally decoded. If a more refined decoding is needed then it is straightforward to add additional gates or a 74LS138 decoder. However, even with this simple arrangement, there is still 8K of address space free for interfacing as needed.

Data I/O is high speed at 115200 baud, and if data transfer between the PC and this board is not controlled then characters will be lost. To ensure data transfer is stopped when the 6502 is not ready for it (ie. the previous byte hasn't been read) the 6850 receive interrupt is enabled. As soon as a character is received, the IRQ on this chip will go low. This is connected, via an inverter, to the /RTS serial connection. The /RTS will therefore go high as soon as a character is received. It will remain high until the character is read (a read of the data resets the IRQ status). Once it has been read, the /IRQ will go high, so the /RTS will go low, allowing the PC to resume transfer. This is repeated as each character is received, controlling the data transfer until the 6502 is ready for it.

NOTE: 8 Chips are needed if you want to run on a 1MHz CPU (0.9216MHz clock). However, only 7 chips are needed if using a 2MHz CPU (1.8432MHz clock).
I have found that ALL my 6502 1MHz chips will run at the 1.8432MHz speed, so U8 is not needed and the 7 chip design works. Additionally, all of the standard 6850 chips that I have worked perfectly at the speeds required for this circuit even though the circuit requirement is faster than their specification. Therefore, you are unlikely to have any issues. However, I would recommend you buy the "B" speed grade ACIA ie. 68B50 if possible.

Power supply pins and any (optional) decoupling capacitors are not shown and need to be connected to the appropriate power rails.

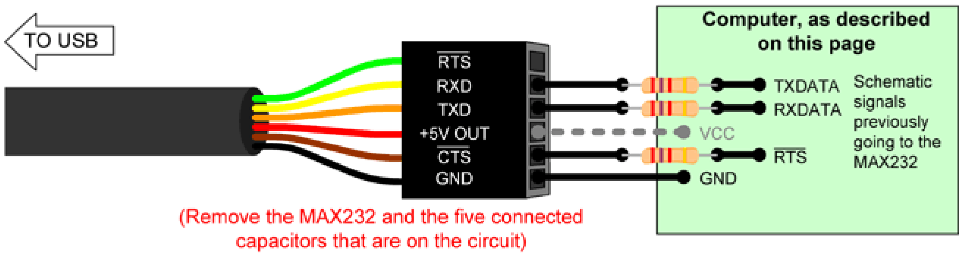


A MAX202 could be used instead of the MAX232, in which case C1-C5 should be 0.1uF.

Alternative serial (and power) connection

The circuit assumes a connection to a standard RS232 port. However, this could also be connected to a USB to TTL (5V) serial cable instead. If doing this, the MAX232, capacitors C1 to C5 and the serial connector are not needed. This therefore reduces the chip count by 1.

Using a USB to 5V (TTL) serial cable instead of an RS232 interface



TO USB
RTS
RXD
TXD
+5V OUT
CTS
GND
Computer, as described on this page
TXDATA
RXDATA
VCC
RTS
GND
Schematic signals previously going to the MAX232
(Remove the MAX232 and the five connected capacitors that are on the circuit)

Resistors are present to safely limit current along the cable if the USB or computer is not powered. These are not needed if using +5V OUT to VCC to power the computer - connect directly instead.

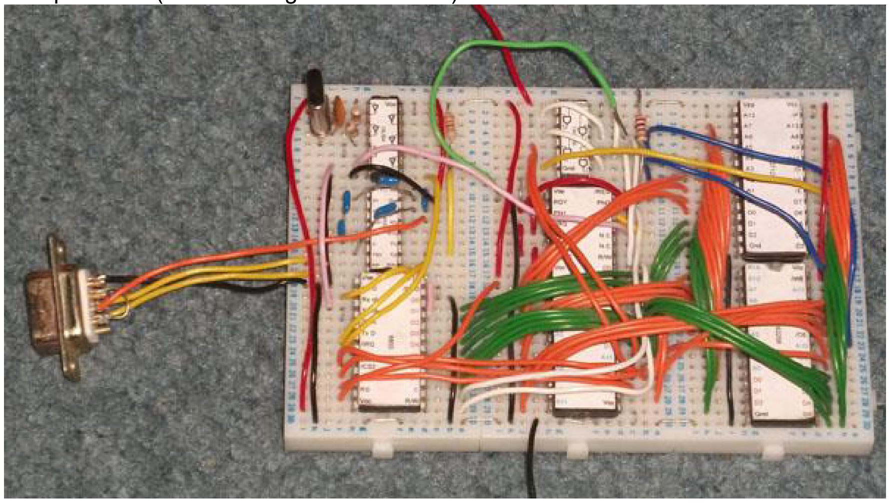
By using a USB to serial cable the board can also be powered from the USB port down the same cable (a powered hub could be used if higher current needed), eliminating the need for a separate power supply for this board. Just connect the +5V out on the cable to the Vcc supply on the board, as shown on the dotted line.

IMPORTANT - only connect Vcc to the USB cable if there is no other power supplied to the board.
The 2k7 resistors are present in the diagram because the board may be powered but not plugged in to the USB cable, or the USB cable could be plugged in and active without power to the board. These limit the current to avoid power being drawn through the interface pins. If always powering the board via the cable then no resistors are necessary.

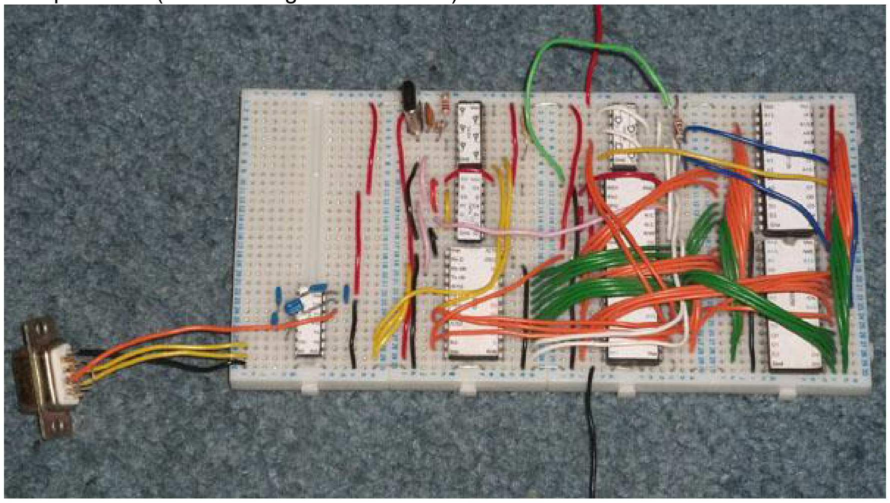
Prototypes

The circuit shown on this page is built here. As you can see, I have attached labels (download HERE) to the chips to allow the wiring to be very straightforward and error-free. Orange wiring is the address lines, green is the data lines, red and black are power lines and yellow, white and pink are the other connections.

7 chip version (CPU running at 1.8432MHz):



8 chip version (CPU running at 0.9216MHz):



ROM BASIC assembly listing and ROM HEX file

Taken from the "pagetable.com" site (http://www.pagetable.com/?p=46), I modified the listings to remove the "SEGMENT" style coding and returned it to using ORG statements. The segment coding would not show a proper "listing" file (address spaces and jump locations were not set). Using ORGs, the listing then became a useable file. I also brought all of the core BASIC code back into one file so that I can see it and maintain it properly.

The BASIC is (*currently - may change*) Microsoft BASIC for the OSI computer, relocated to $C000, with the serial I/O and control-C routines modified to point to my serial handler routines.

I wrote the serial handler to control the text I/O, along with suitable Control-C break handling.

Update 4th February 2014 : Fixed delete key to use normal delete instead of "_" character (OSI BASIC)

The attached ZIP file contains the following:

Source code
osi_bas.s <== the Microsoft OSI BASIC and I/O routines SOURCE all in a single file. My I/O routines are at the end of it.

Files to allow the source to be assembled on a Windows based machine
assemble.bat <== double click to assemble osi_bas.s and link to binary file "osi_bas.bin". This should be exactly 16K
osi_bas.cfg <== configuration file for the linker (ensure ORG and entries in this file match if you change any)
ca65.exe <== the assembler from the cc65 package. Use this. The new version on the cc65 site crashes!
ld65.exe <== linker from the cc65 package

Output files
osi_bas.bin <== the ROM fine in pure binary
osi_bas.lst <== Assembly listing file
rom.hex <== the ROM fine in standard INTEL-HEX format

To allow simple re-assembly, extract all files to the same folder. Freeware utilities are available to convert the "bin" file to HEX or s19 (etc) - use your internet search tool to find.

All source code, assembler binaries and the HEX dump of the ROM is here. It is in standard INTEL-HEX format for uploading to a suitable programmer.

Acknowledgements
All work done by original authors are respectfully recognised.
Original split source code from http://www.pagetable.com/?p=46
Assembler and linker from http://www.cc65.org/ (older version - cc65-win32-2.13.3-1)

ROM BASIC - details of what has been included/excluded

INCLUDED TOKENS

END, FOR, NEXT, DATA, INPUT, DIM, READ, LET, GOTO, RUN, IF, RESTORE, GOSUB, RETURN, REM, STOP, ON, NULL, WAIT, DEF, POKE, PRINT, CONT, LIST, CLEAR, NEW, TAB(, TO, FN, SPC(, THEN, NOT, STEP

SGN, INT, ABS, USR, FRE, POS, SQR, RND, LOG, EXP, COS, SIN, TAN, ATN, PEEK, LEN, STR$, VAL, ASC, CHR$, LEFT$, RIGHT$, MID$

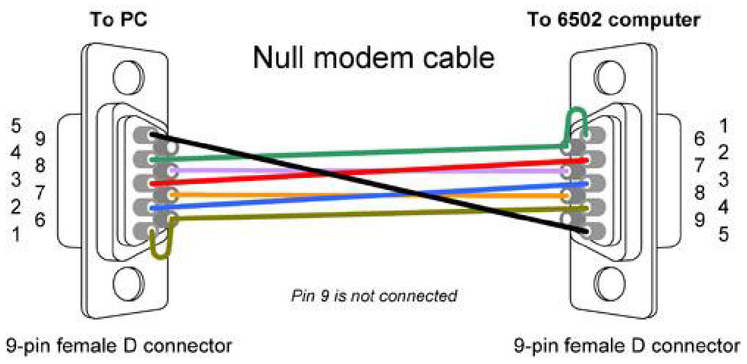
+, -, *, /, ^, AND, OR, >, +, <

UNPROGRAMMED TOKENS

LOAD, SAVE

Powering-up

You will need a suitable connection to a PC or other terminal display. You will probably need a null-modem cable to connect this to a PC serial port. Here is my recommended wiring.



Connect to a PC or similar terminal, with **115200 baud, 8 bits, hardware handshake, no parity, 1 stop bit.**

Power on the board and press the RESET button. You will see the following message on the terminal:

```
Cold [C] or warm [W] start?
```

Press C.The following will then be displayed:

```
MEMORY SIZE?
```

Press the [Enter] key to use all available. After a few seconds (memory test) the following will appear on the terminal:

```
TERMINAL WIDTH?
```

Press the [Enter] key to use the default. You will then be presented with the BASIC startup message:

```
 32255 BYTES FREE

OSI 6502 BASIC VERSION 1.0 REV 3.2
COPYRIGHT 1977 BY MICROSOFT CO.

OK
```

The machine is now ready to use.

Typing

`PRINT FRE(0)`

will give (current ROM version, may be slightly different with future releases)...

`32253`

`OK`

This shows that the 32K RAM has been recognised correctly.

...the rest is up to you.

Startup sequence and a simple program entered and run on the terminal is shown here...

```
Cold [C] or warm [W] start?

MEMORY SIZE?
TERMINAL WIDTH?

 32255 BYTES FREE

OSI 6502 BASIC VERSION 1.0 REV 3.2
COPYRIGHT 1977 BY MICROSOFT CO.

OK
10 FOR A=0 TO 6.2 STEP 0.2
20 PRINT TAB(40+SIN(A)*20);"*"
30 NEXT A
RUN
                                        *
                                           *
                                               *
                                                   *
                                                      *
                                                        *
                                                          *
                                                           *
                                                           *
                                                           *
                                                          *
                                                        *
                                                     *
                                                  *
                                              *
                                          *
                                      *
                                  *
                               *
                           *
                        *
                      *
                    *
                    *
                    *
                    *
                      *
                        *
                           *
                              *
                                  *
                                      *

OK
```

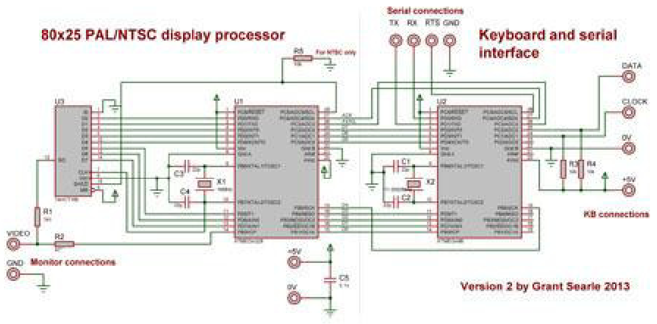
Loading and saving

Save
Type LIST but don't press enter.
In your terminal window on the PC, or whatever you are using to operate this board, enable the capture to a text file.
Press ENTER.
The listing will then be spooled to the text file on your PC.
Once the listing is complete, end the capture. The contents of the program will then be on the PC for later use.

Load
Most terminal programs have a method to pass a text file to the destination.
Ensure the board is ready to receive characters and then transfer the text file from the PC to the board using the terminal.
The data transfer is automatically suspended if the 6502 board is not ready, so no characters are lost providing hardware (CTS/RTS) handshaking is enabled on the PC.

Using a keyboard and monitor instead of the PC interface

This board, as for my other boards, can use my monitor and keyboard interface here



Links

The following links were very useful to me...

pagetable.com - BASIC assembly code - http://www.pagetable.com/?p=46

My other pages

CLICK HERE TO GO TO MY MAIN PAGE FOR MY OTHER PROJECTS

I hope this page has been useful.

Grant.

To contact me, my current eMail address can be found here. Please note that this address may change to avoid spam.