A Tandy / RadioShack TRS-80 Model 1 Microcomputer clone.

Built in 2019 using current-production parts.

Glen Kleinschmidt
www.glensstuff.com
Jan. 2020

Introduction

Having been thoroughly bitten by the old-school 8-bit computing bug and with my Commodore PET 2001 clone (http://www.glensstuff.com/pet2001/pet2001.htm) done and dusted, it was time to start investigating potential prospects for the next project.

Released in 1977 (the year I made my own appearance), the TRS-80 Model 1 microcomputer was, for a period of time, one of the best selling, if not the best selling, personal computers for the enthusiast at home. In common with the PET 2001, a CMOS variant of the machine's original microprocessor, even after all of these intervening years and technological progress, is still in active production and readily available.

Additionally, as per the PET 2001, the machines primitive monochrome video display graphics were generated by means of straightforward TTL logic circuity rather than utilising any kind of propriety or now long-obsolete video graphics-generating integrated circuits or similar. There was one odd-ball chip thrown into the mix though – a single IC manufactured by Motorola designated the MCM6670 and rather grandiosely given the title of a "Character Generator".
This IC, not so much in the way of an actual generator, was in fact just a small mask-programmable ROM housed in an 18-pin plastic DIP. It featured 5-bit words and was presumably chosen by the hardware designer(s) of the TRS-80 for being a cheaper storage medium for the computers character set than something like the MM2716E UV-erasable PROMs that were used for storing the operating system.

Suffice to say, the TRS-80 was the perfect candidate for my next 8-bit retro computer clone project. At this juncture it's worth mentioning that, just as per my PET 2001 project, this clone is a functional replica of the original computer in the traditional hardware sense.
It's not an FPGA port or an emulator running on a Raspberry Pi and nor is it a part-for-part duplication of the original circuitry, but a complete ground-up re-design using contemporary discrete CMOS logic and memory devices, with some additional features thrown in for good measure. At the time of writing every component used in this project is a current-production part.

74HC(T) CMOS family logic entirely displaces the original LS TTL logic and great simplifications were made by using modern memory devices. The self-contained TRS-80 Model 1 keyboard unit was originally offered with as little as 4 kilobytes of system RAM, but could be expanded to a maximum of 16 kilobytes. The BASIC operating system, however, could recognise an expandable maximum of 48 kilobytes. To get the full 48 kilobytes of system RAM you needed a 16 kilobyte-equipped keyboard unit in addition to either a third-party memory unit or the official Expansion Unit; these being external devices which plug into the keyboard unit's Expansion Interface port.

The Expansion Unit was an optional accessory which, amongst other things, provided a floppy disk drive controller and a Centronics parallel printer port. The Expansion Unit could also be provisioned with the additional 32 kilobytes of RAM. Note that my clone project detailed here, except for sporting the full 48 kilobytes of system RAM, does not replicate the functions of the Expansion Unit.

A single modern static RAM chip, part # AS6C1008, provides the full 48 kilobytes of system RAM and dispenses with a great deal of address decoding logic. There's no need for arrays of single-bit-wide dynamic memory chips in this day and age! All of the address multiplexing and refresh logic associated with the original DRAM memory has therefore been conveniently dispensed with.

The circuit simplifications continue on to the systems read-only memory which stores the operating system. A single AT27C256 chip serves as the system ROM and is in fact large enough to contain both the early ("Level I") and the later, revised and expanded ("Level II") versions of the BASIC operating system; more about this later.

Back in the day if you wanted sound effects for games (or for any other purpose) you would unplug your data cassette unit and feed the "CASSOUT" signal available at pin 5 of the "Cassette I/O" port to an external audio amplifier. The TRS-80 Model 1 didn't have any dedicated hardware specifically for producing sound as such, but utilising the data cassette output register bits for this purpose soon became the established method for generating computer game beeps, zaps and primitive tunes.

My clone design incorporates an LM386 audio amplifier with volume control and a source-select function giving the additional handy utility of letting the user listen in on the read and write signals from or to the external data cassette storage device.

I also designed a universal PS/2 keyboard interface, which is accommodated on a separate, self-contained circuit board and is compatible with any TRS-80 Model 1, not just my clone design. Via a length of ribbon cable it plugs into the keyboard interface connector and permits the connection and use of any standard PS/2-protocol keyboard with the computer.

Compete schematic diagrams, bills of material and further technical descriptions follow in this document.

A complete set of Gerber files for the PCBs, ROM image binary files and firmware for the PS/2 keyboard interface are available for downloading on my website:

www.glensstuff.com

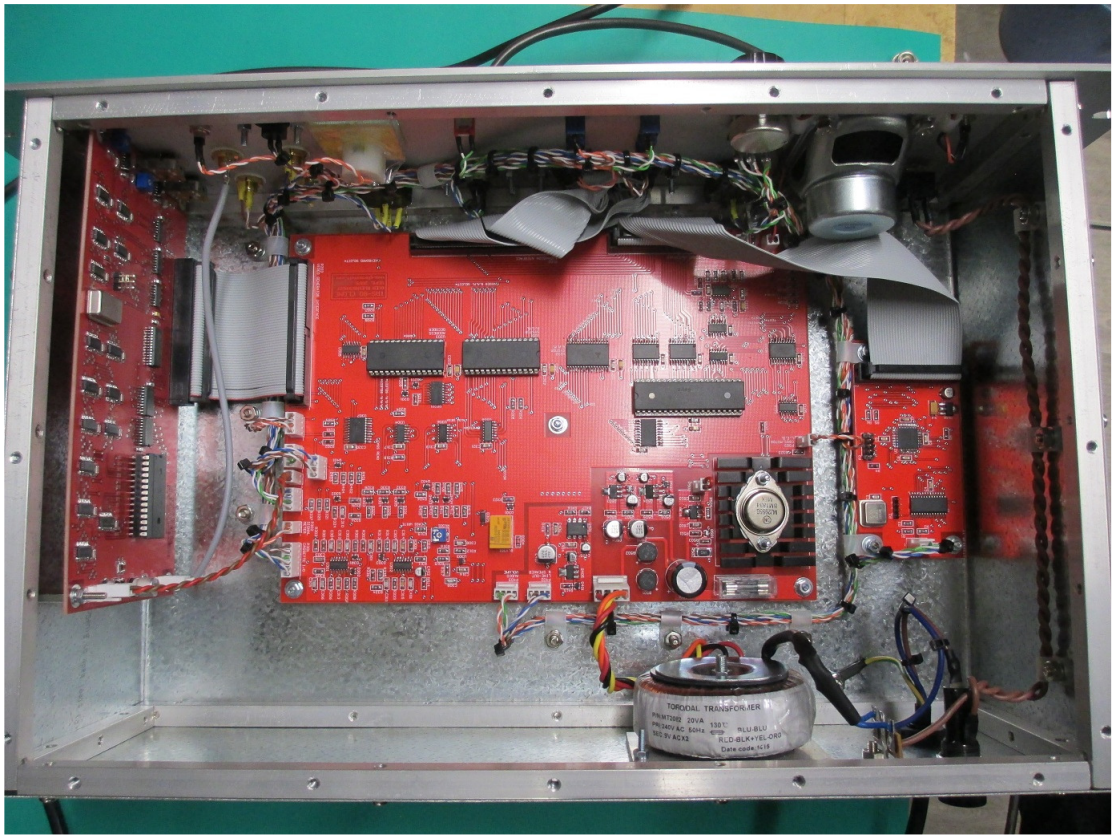
You can view a Youtube video of my competed TRS-80 Model 1 clone loading and running a version of the computer game *Sea Dragon* here:

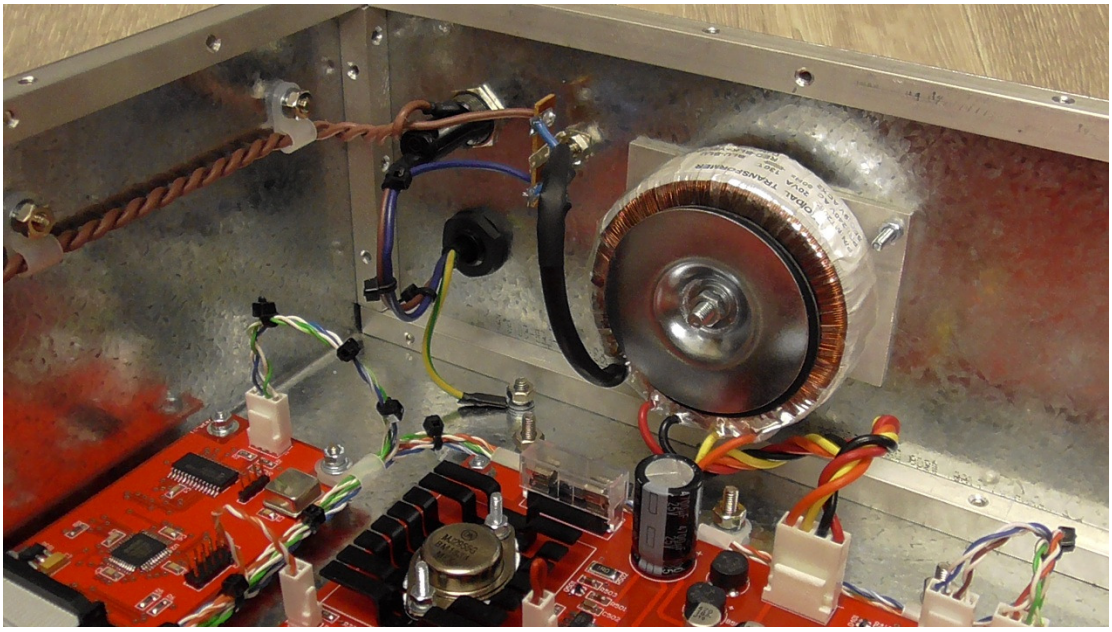
https://www.youtube.com/watch?v=LTg4eb-QqK0

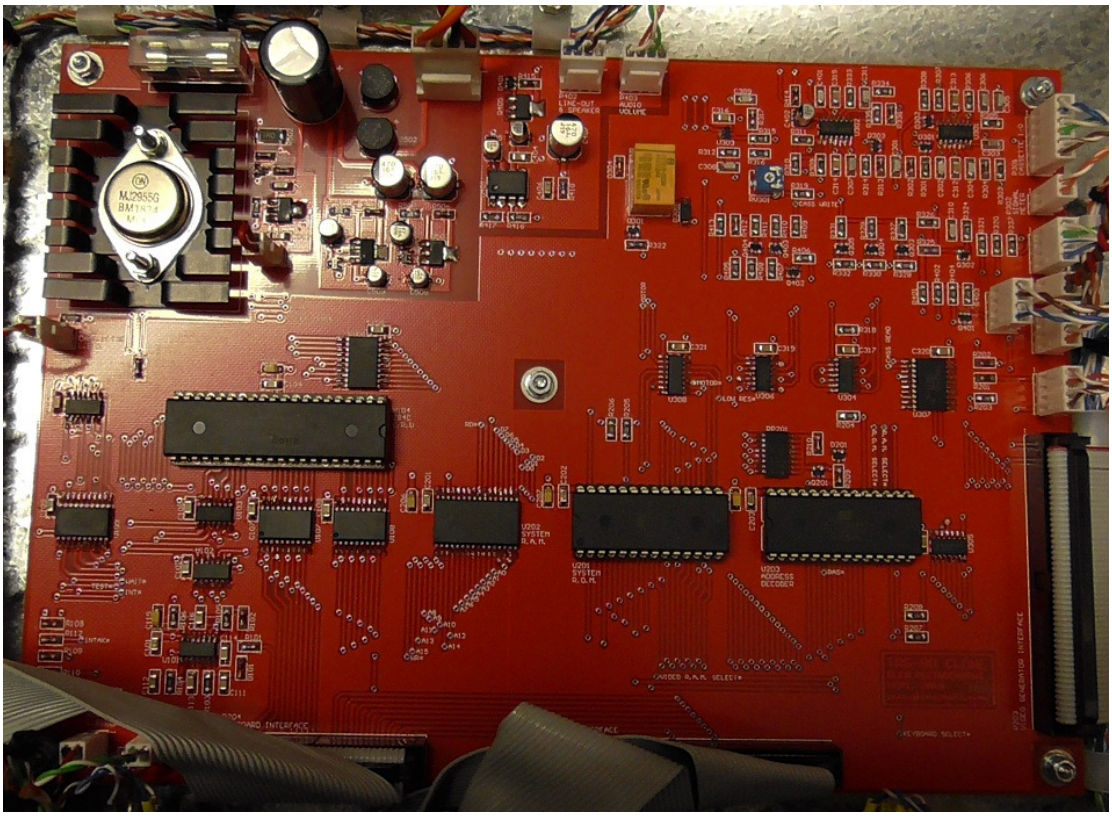
Here you can see a video of the computer in the early developmental phase, built and operational entirely on solderless breadboard! The computer at this stage was sporting 16k of RAM, ran BASIC Level I and is shown loading and running *RadioShack Flying Saucers*:

https://www.youtube.com/watch?v=dClOHNcpnVw

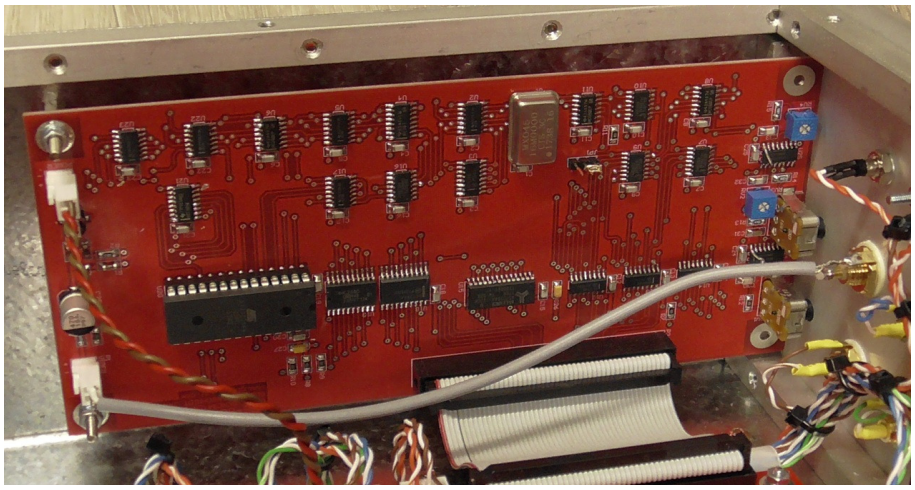

TRS-80 MODEL 1 CLONE
GLEN KLENSCHMIDT AUGUST 2019
POWER
VOLUME
LEVEL 1
LEVEL 2
BASIC
16k
32k
R.A.M.
UPPER
LOWER
CASE
ORIGINAL
UPGRADED
FONT
C.P.U. RESET
CASSETTE MOTOR
CASSETTE READ IN
CASSETTE WRITE OUT
VIDEO OUT
AUDIO OUT
RASTER POSITION HORIZONTAL
RASTER POSITION VERTICAL
OFF
S_EX
WRITE
READ
AUDIO SOURCE
EXPANSION INTERFACE
PS/2 KEYBOARD
0.2V
0.5V
1V
2V
CASSETTE WRITE LEVEL


TOROIDAL TRANSFORMER

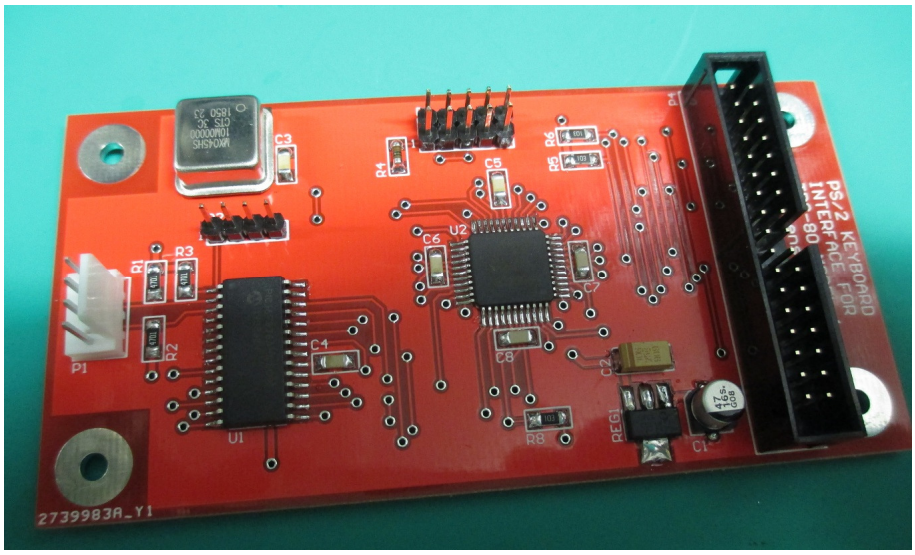


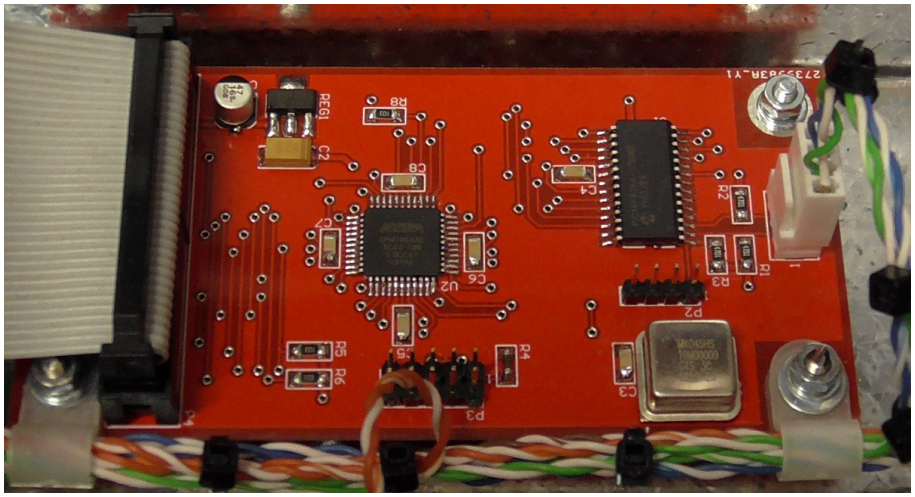


The motherboard

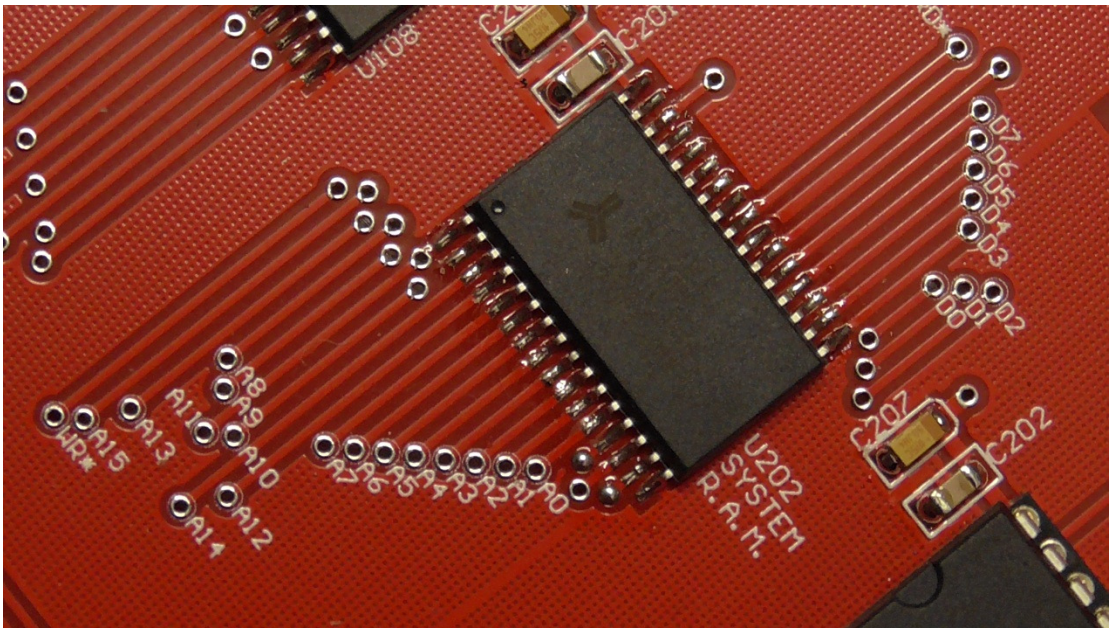


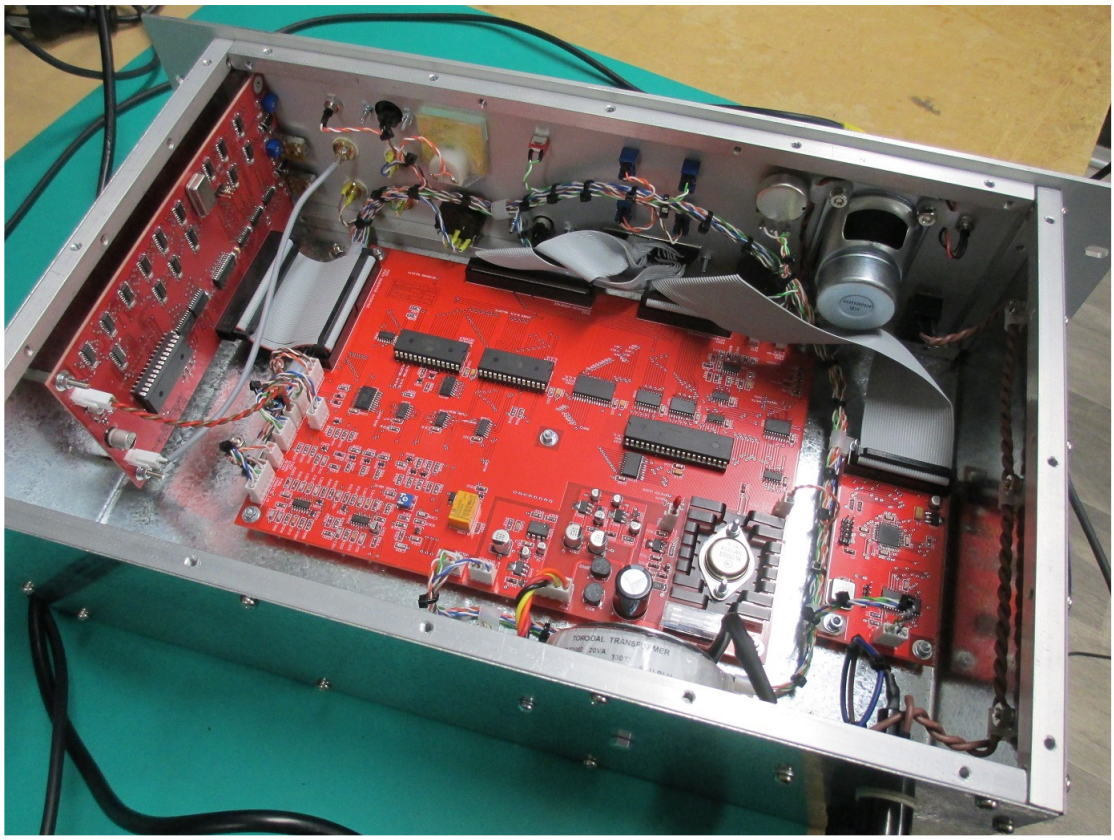
The video generator board





The PS/2 keyboard interface board


U108
C201
C202
U202
SYSTEM
R.A.M.
A8
A9
A11
A10
A13
A15
A14
A12
A7
A6
A5
A4
A3
A2
A1
A0
D7
D6
D5
D4
D3
D2
D1
D0



Hardware Overview

With the exception of some small amount of logic associated with the keyboard, all of the circuitry in the original TRS-80 Model 1 was accommodated on one large PCB. In my clone design I have split the video generation circuitry onto a separate PCB. There are thus two PCBs, designated the “motherboard” and the “video generator board”.

The motherboard contains:

The power supply circuitry,
CPU and support circuitry,
System RAM & ROM,
All peripheral I/O interface circuitry,
The audio amplifier and associated circuitry.

Video generator board

The video generator board contains all of the circuity required to generate the TRS-80’s 64 (32 in low-resolution mode) column by 16 row, character-based video display. This includes the screen, or video, display RAM, the character ROM and a 16 MHz master oscillator module from which the 1.7778 MHz CPU/system clock, as well as all video timing signals, are derived. In numerous aspects the circuitry departs from the original design fairly significantly.

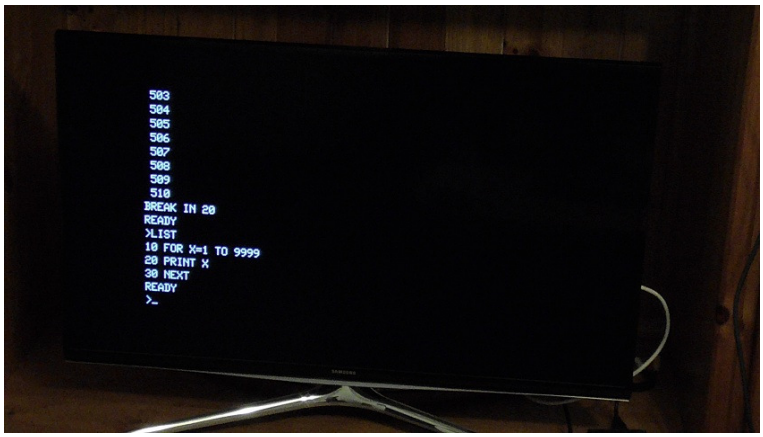
Note that in the TRS-80 the *video* RAM and ROM space is in addition to and separate from the *system* RAM and ROM space. There are 64 x 16 = 1024 bytes of system-accessible video RAM to define each individual character location. The video ROM simply contains the data defining the TRS-80’s alpha-numeric and graphical character set and is continuously and exclusively accessed by the circuitry concerned with producing the video display. The video character ROM is not hardware accessible to the CPU.

In high-resolution mode, each character location occupies a grid of 6(h) x 12(v) pixels. The complete display is therefore composed of 384(h) x 192(v) pixels. There were some weird design decisions made with the original TRS-80; for example the odd-ball 10.6445 MHz pixel-shifting clock frequency. Perhaps that was a commonly available crystal 40+ years ago, but you certainly can't buy a 10.6445 MHz crystal off the shelf today.

That high frequency meant that all 384 pixels of a complete horizontal row were serially shifted out in just 36 uS of the complete 63.13 uS line period. This means that unless displayed on a TV or monitor having a tweakable (expandable) width control for the horizontal picture size, the result will be a fairly squished up display, horizontally. An additional bother is that 10.6445 MHz is pushing the video bandwidth limitation of your typical TV-based display monitor a bit too far and it was a complaint back in the day that the high-resolution video characters were a bit ill-defined and blurry on screen; even on the re-purposed, modified and re-decaled B&W television set that was originally sold as a dedicated display monitor for the TRS-80. Maybe the Tandy Corporation just had a pre-existing stockpile of 10.6445 MHz crystals to find a use for?

To mitigate these issues and to avoid having to source an unobtanium 10.6445 MHz crystal, in my clone design I've divided the 16 MHz master clock by two to deliver a lowered pixel-shifting clock frequency of 8 MHz. After re-working the divider chains, all 384 horizontal pixels are now shifted out in 48 uS of a complete 64.5 uS line period. This results in a generated video display which fills the screen horizontally.

Jumper JP1 on the video generator board permits the field frequency to be set to either a 50 Hz or 60 Hz frame rate, for display compatibility between standards. By virtue of the way the divider chains work out, the selectable frame rates, at 49.69 Hz and 59.73 Hz, aren't precisely 50 Hz and 60 Hz, but neither were they in the original machine and they are still well within acceptable limits. I have yet to find a modern digital TV (that is those still having a composite video input), let alone any old analogue display monitor which will refuse to sync.



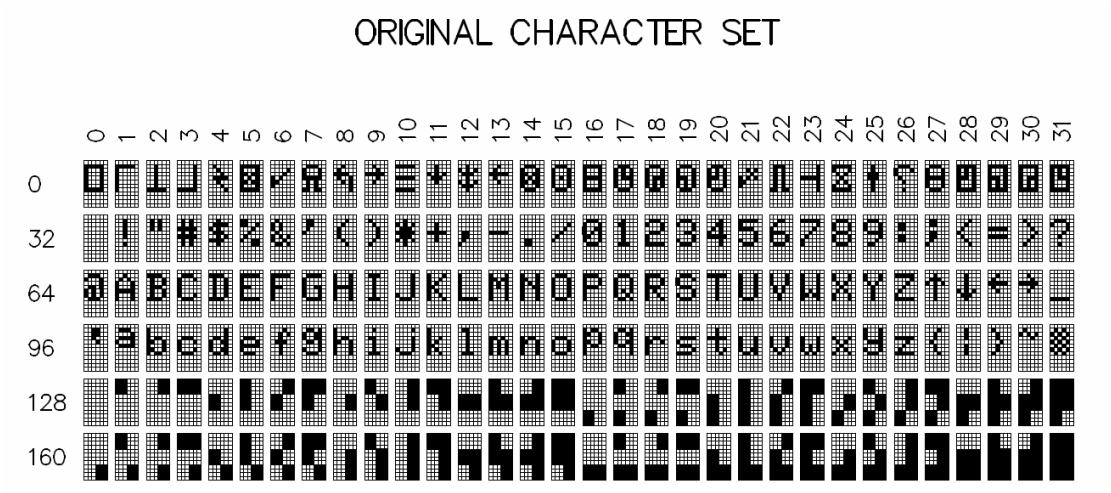
In the original TRS-80 the 10.6445 MHz master clock was divided by 6 to deliver a CPU clock signal for the Z80 microprocessor of 1.7741 MHz. In my clone design I divide the 16 MHz master clock by 9 to deliver a 1.7778 MHz clock for the Z84C microprocessor. 1.7778 MHz is more than close enough to the original frequency so as to not cause any compatibility problems with data cassette baud rates and the like. I guess that I can even boast that my clone design runs a little bit faster than the original item too. Ha!

Lower case modification

The original TRS-80 Model 1, unless modified, was incapable of displaying lower case characters; even so lower case characters were present inside the MCM6670 character ROM chip. This proved to be an issue when it came to applications a little more serious than computer games, such as word processing. In 1978 the Electric Pencil word processor program came with instructions detailing the hardware modifications required to enable the display of lower case letters of the alphabet. RadioShack itself eventually started selling a lower case conversion kit (catalogue number 26-1104) which was more advanced, coming with a replacement/substitute character ROM chip to overcome some limitations incurred when enabling the display of lower case characters with the original ROM (more on this later).

My clone design implements the lower case modification and the character ROM contains both the original and the upgraded character sets. The "CASE" switch on the front panel permits the lower case modification to be either enabled (set to "LOWER") or disabled (set to "UPPER"). The "FONT" switch permits selection of either the "ORIGINAL" or the "UPGRADED" character set.

To properly understand the ins and outs of the lower case hardware modification a good place to start is by looking at the organisation and contents of the original character set and ROM:



Character locations 0 through 127 are those defined in the originally installed MCM6670 character ROM chip. The 64 characters occupying locations 128 through 191 are the TRS-80's "graphics" characters. These characters provide chunky (3 x 4 pixel-size block) pseudo-bitmapping and were originally generated by discrete 74LS logic circuitry independently of the character ROM.

In the original hardware, the weird and not particularly useful hieroglyph-like characters occupying ROM locations 0 through 31, in addition to all of the characters (which includes the lower case letters) defined in locations 96 through 127 were not accessible.

For the 1024 bytes of video RAM, the TRS-80 used and array of 1-bit wide data-I/O static RAM chips addressed in parallel, but there were only seven of them, not the eight required for a full byte! SRAM chips were expensive back in the day and the original hardware designer, counting his beans, simply decided to do without one. With only seven true bits, we can only access 2^7 = 128 unique characters out of the 191 actually defined.

The missing SRAM chip was in the position of bit 6. Note that when bit 7 (128_{DEC}) is high we are selecting a graphics character. Bit 7 was therefore defined in the original service manual (under the title "VIDEO RAMS" on page 17) as the "graphical/alphanumeric definition" bit.

It's worth pointing out at this juncture that in the same paragraph a bit of a turkey-brained and not particularly helpful explanation was given for the derivation and function of bit 6. In lieu of the missing SRAM chip, a pseudo bit 6 was generated by NORing bits 5 and 7 with one gate of a 74LS02.

We're all good up 'till this point, but the author then went on to explain that this "is a sneaky way of squeezing a seventh ASCII bit out of six RAMs". Besides the minor niggle that the TRS-80's character encoding isn't entirely compliant with the ASCII standard, the real mischievousness here is in the insinuation that seven true bits of data were miraculously extracted from only six. Such a feat would have profound implications going well beyond the scope of this discussion and we would actually be able to access all 128 characters defined in the character ROM, rather than only a selection of 64, but no, I'm afraid this isn't the case after all.

To help properly convey how the pseudo bit 6 impacts upon the addressing of the character ROM, I have produced the following table:

Data byte	**D5** (32_{DEC})	**D7** (128_{DEC})	**D6** (64_{DEC}) D5-NOR-D7	**Actual character location addressed**
0-31	0	0	1	64-95
32-63	1	0	0	32-63
64-95	0	0	1	64-95
96-127	1	0	0	32-63
128-159	0	1	0	128-159
160-191	1	1	0	160-191
192-223	0	1	0	128-159
224-255	1	1	0	160-191

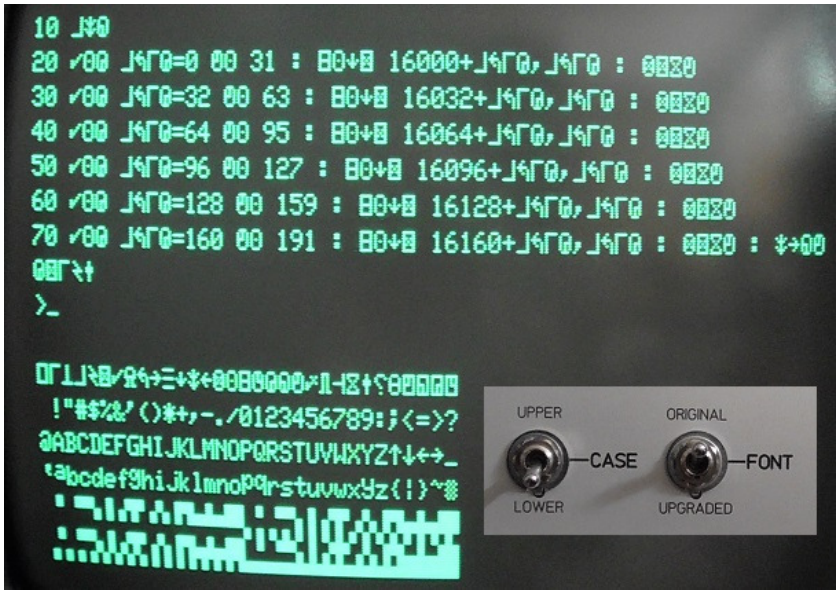
This table shows how the hieroglyph-like characters in ROM locations 0 through 31 in addition to the lower case and other characters in ROM locations 96 through 127 are barred access. Data bytes 0 through 31 don't access the same locations in the ROM, but map directly to locations 64 through 95 instead. Similarly, data bytes 96 through 127 map directly to ROM locations 32 through 63.

So, if we write a BASIC program to sequentially dump all of the addressable characters to the screen in order, we simply end up getting two complete instances of the 64 characters defined in locations 32 through 95 of the character ROM, rather than the full set of 128 unique characters:

Note that I only dumped to screen characters 0 through 191; 192 through 255 just sequentially repeats the 64 graphics characters.

The Electric Pencil lower case modification, mentioned previously, involved some track cutting, wiring and soldering in a substitute for the missing SRAM chip to deliver a true bit 6. This substitute SRAM chip was typically piggybacked on top of one of the existing ones and you had to lift up the data input and the data output pins and wire these into circuit independently. You also had to drill a hole somewhere in your computers plastic case for the mounting of a toggle switch. This toggle switch would be wired to effectively switch the lower case modification in and out of circuit, by selecting between either the true bit 6 now available (provided by the additional SRAM chip) or the original pseudo bit 6.

The reason that this switch to revert the computer to its standard mode of operation was required is made obvious by the following character set screen dump:

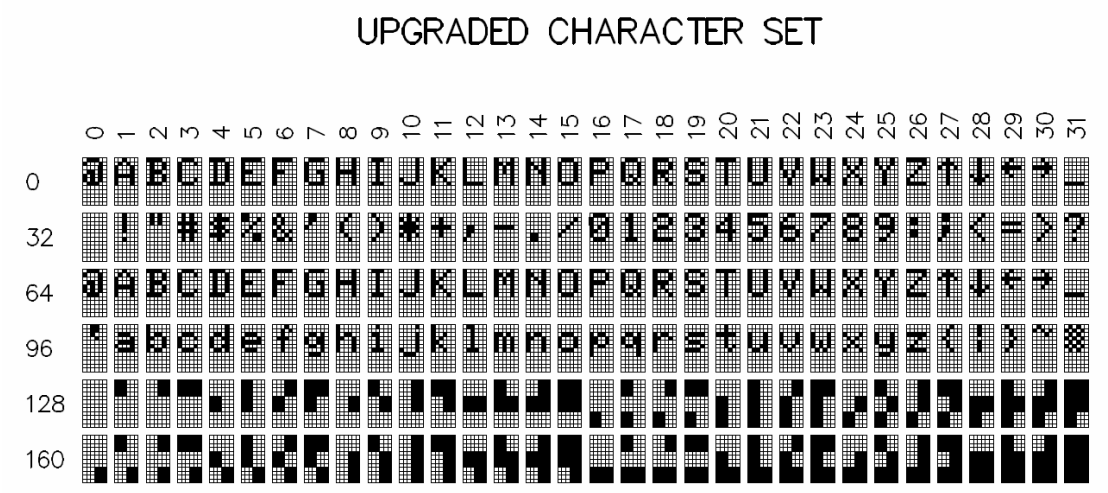


This is the exact same BASIC program and screen dump as exhibited in the previous photo, but now the "CASE" switch has been flicked to the "LOWER" position, enabling the lower case modification. As you can seen, the complete set of 128 unique characters, as defined in the character ROM, are now accessible and have been sequentially dumped to the screen (yay!), but, erm, we appear to have a problem, Houston.

Level II BASIC was originally written with the presumption that data values 0 through 31 will address ROM locations 64 through 95. As a result the BASIC program listing has turned into incomprehensible gobbledygook for anyone who isn't a savant, as the computer is now printing text to the screen using the hieroglyphs.

When the computer running Level II BASIC wishes to display "C" in a specific location on the screen, it programs the corresponding location in the video RAM with a data value of 3_{DEC}, which was previously redirected to address location 67 of the character ROM; that being the location defining the character "C". But now, with the lower case modification enabled, data values in the video RAM directly address the same locations in the character ROM, so instead of "C" we actually get the mirror-image "L" that is defined in character ROM location number 3.

The cure for this problem is a simple one; when not using the Electric Pencil word processor you can simply switch the lower case modification out of circuit, or you can simply update the character ROM to deliver the correct characters to the screen regardless. The latter was achieved by the updated/replacement character ROM of the RadioShack lower case modification kit mentioned earlier:



In the upgraded character ROM the 32 weird hieroglyph-like characters originally in locations 0 through 31 are deleted and substituted with a copy/repeat of the 32 characters of locations 64 through 95. With this character ROM, when the BASIC operating system calls for the display of character number 3 is gets its “C”, whether the lower case modification is enabled or not.

Now here is a repeat of the character set screen dump, but this time the “FONT” switch has been flicked to “UPGRADED”, enabling the page of the character ROM programmed with the upgraded character set; as though a genie has waved his magic wand (I’m fairly sure I’ve seen at least one genie wield a magic wand and that mastery of this instrument isn’t solely the purview of fairies), the BASIC listing is transformed back into legibility, even so the lower case modification remains enabled:

Another thing that had been “fixed” with the upgraded character ROM, which you’ve probably noticed already, is that the lower case letters of the alphabet have been revised. They are more aesthetically pleasing now and the letters “g”, “j”, “p”, “q” and “y” have single-line descenders this time ‘round.

The original MCM6670 character ROM chip was mask-programmable. It was sold with either a default, pre-defined character set, or, given some specific minimum order quantity, Motorola could custom-program the device to your personal specification. The latter I guess is obviously how RadioShack procured their upgraded character ROM and maybe their original character ROMs too, as there are a few deviations from the character set graphically defined in the MCM6670’s original datasheet.

There was good reason why the lower case letters of this chip’s default character set had that funky look without the descenders though; it wasn’t just due to the weird artistic bent of some integrated circuit designer. The MCM6670 had 5 bit words and eight such words (or “bytes” incomplete by 3 bits) defined each character. Each character thus occupied a 5(h) x 8(v) pixel grid, but only 7 of the 8 available rows were generally used in practice. The chip was referred to in Motorola literature as a “….5 x 7 Character Generator”.

When defining alphanumeric characters, you had to leave either the topmost or the bottommost row of pixels of all of your defined characters blank to ensure that, between lines of displayed text, there would be at least one pixel of vertical separation. This is why no characters are higher than 7 of the 8 available pixels. To implement descenders on the lower case characters with acceptable legibility, all of the 8 pixels available vertically need to be utilised.

The TRS-80 can get away with using all 8 of those vertical pixels though, and we can have our lower case descenders. This is because the TRS-80’s video hardware unconventionally puts four additional and permanently blank horizontal scan lines under each and every non-graphical character.

The video generation circuitry of my clone design has been simplified significantly over the original circuitry. A single SRAM chip with an 8 bit-wide data bus (U15) serves as the video RAM. NOR gate U16D generates the pseudo bit 6 to emulate the original operation, while quad NAND gate U17 comprises a 2-input multiplexer/data selector which selects between either the pseudo bit 6 or the true bit 6 available directly from the video RAM. When the “lower case” select line is unasserted, it is pulled low by resistor R10. In this state U17 selects and addresses the Character ROM using the pseudo bit 6. When the “CASE” switch is closed, by being flicked to the “LOWER” position, the “lower case” line is switched high and U17 selects and addresses the Character ROM using the true bit 6; lower case mode is enabled.

To knock about five discrete logic chips out of the design, I did away with the discrete logic circuitry for generating the graphics characters and have instead defined all displayable characters in the character ROM, U20. Address line A12 of U20 is used as a page-select line, under the control of the “FONT” switch. The first 4 kilobyte page of U20 contains a replication of the original character set, while the second 4 kilobyte page contains the upgraded character set.

Rock stable composite sync generation and manual control over the raster positioning is provided by a proper pair of monostable ICs (75HC4538), U24 and U25. The original TRS-80 used a funky arrangement of R-C networks and 74C04 inverters, which proved troublesome over time.

Memory, address decode and BASIC

Referring to page 2 of the motherboard schematic set, U201 is the system ROM. This 32 kilobyte memory contains both versions of the BASIC operating system originally produced for the TRS-80 Model 1.

U201 is programmed with the original “Level I” BASIC in the first 4 kilobytes of the first 16 kilobyte page and the upgraded “Level II” BASIC in the first 12 kilobytes of the other. Page selection is decided by the “BASIC” selector switch which connects to the motherboard via header P202. When the “BASIC” selector switch is open, R204 keeps address pin A14 of U201 low, selecting the lower 16 kilobyte page and thus causing the TRS-80 to run Level I BASIC. Closing the “BASIC” selector switch pulls A14 high and selects Level II BASIC.

All of the computers memory address decoding logic, aside from that for the data cassette I/O register, has been simplified to a ROM look-up table stored in U203. The four separate areas of addressable memory under the direction of U203 are the:

System ROM
System RAM
Video RAM
And the memory-mapped keyboard interface.

The “R.A.M.” switch on the front panel asserts a page select on the address decoder (U203) and permits the user to select between an addressable system RAM quantity of either 16 or 48 kilobytes.

The BASIC command “PRINT MEM” returns the number of “free” system RAM bytes available above that reserved for the functions of the operating system, which, apparently, is 812 bytes for BASIC Level II. In all honesty I’m not too sure how useful this feature will be in practice, but if anyone who might replicate my clone design for themself would like to simulate running out of RAM at the limit that was originally imposed upon the keyboard unit itself, then there you go!

From TTL to CMOS and booting in Level II BASIC

Referring to sheet #2 of the Motherboard schematic ("Memory and Address Decode"), resistor array RP201 provides a passive, TTL-level logic-high pull-up to all data bus lines.

In the original TRS-80, a trio of TTL 74LS367 Hex Bus Drivers, working together as transceivers, buffered the bi-directional 8-bit data port of the Z80 microprocessor to the outside world. In my clone design this function is performed by a single CMOS 74HCT245 Octal Transceiver chip.

Unlike the TTL inputs of 74LS-series logic ICs, the CMOS inputs of 74HC-series ICs do not effectively have internal pull-up resistors by virtue of the structure of their internal architecture; if left unconnected, 74LS-series inputs float high. When any one of the high-impedance inputs of a 74HCT245 is left to float on its own accord, said input will assume an arbitrary voltage level which, for most practical purposes, can be thought to depend upon the phase of the moon as much as anything else.

When the TRS-80 boots running Level II BASIC, one of the first things it does is read the data bus whilst not addressing any of the memory devices or registers contained within the keyboard unit. It does this to check for the presence of that, back in the day coveted, optional external accessory; the Expansion Unit. If the data bus returns any value other than 0x00 or 0xFF the operating system will assume that an Expansion Unit is connected and it will then attempt to talk with the floppy disk controller present in said Expansion Unit.

When the operating system performs this check without an Expansion Unit connected, the data bus is essentially floating as, as already mentioned, no memory devices or registers internal to the keyboard unit are being addressed; simply put, nothing is receiving a request to put data onto the bus. In this state, the 74LS367 chips would reliably present a data bus value of 0xFF to the microprocessor, by virtue of their internal pull-ups which cause all non-driven inputs to float high. The operating system accepts 0xFF as an indication that an expansion unit isn't connected to the computers Expansion Port and it then proceeds to boot into BASIC.

Here I have detailed for you, dear reader, an unforseen and potential gotcha when it comes to re-designing old TTL computer hardware with CMOS. It was when breadboarding my clone circuitry (yes, I really did build the entire computer on solderless breadboard) that I discovered the above detailed quirk of the boot sequence for Level II BASIC.

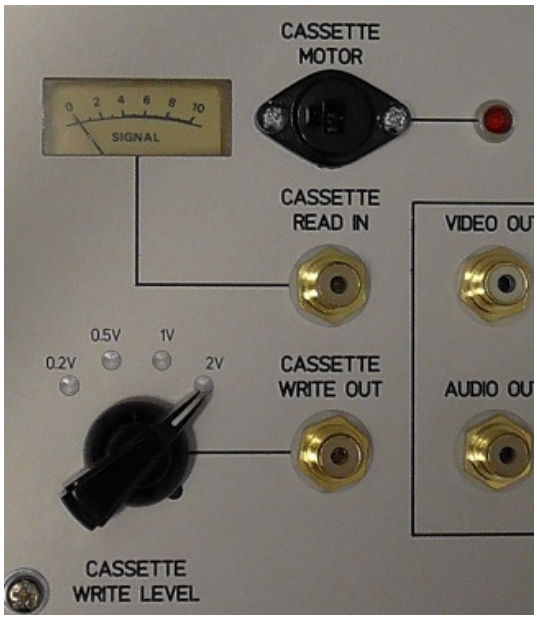
My CMOS 74HCT245 transceiver chip, rather than giving a reliable 0xFF, was presenting random byte values to the microprocessor during the test for the Expansion Unit at boot up, causing Level II BASIC to hang practically every time I turned the power on. Failing the 0xFF test, the computer erroneously assumed that an Expansion Unit was connected and it then became stuck in a hopeless loop trying talk with an external floppy disk controller.

Due to good fortune, however, this wasn't cause to angrily swipe the whole convoluted mess of breadboard from the bench, abandoning all hope and writing off many hours of work and the endeavour of an all-CMOS clone as a pipedream and lost cause. All I had to do was add the passive pull-ups to the data bus lines as I have detailed here in the final design. Phew!

Data cassette interface.

The TRS-80 Model 1 keyboard unit did all of it's loading and storage of programs and data from and to an external data cassette recorder. This was just a conventional audio cassette recorder which RadioShack re-decaled, rewired and sold as a dedicated peripheral to plug into the 5-pin DIN connector that was the TRS-80's "Cassette I/O" port.

Given that practically any half decent audio-frequency recording device can be adapted to perform program and data storage for the TRS-80, I wanted a little more flexibility over the cassette interface designed into the original computer for my clone implementation. Different recording devices have different record-input signal sensitivities and output signal levels and drive capability, so I wanted a "cassette" interface which could reasonably accommodate these varied requirements.



The original design was notorious for its unreliability; for example the playback signal level from the cassette recorder was fairly critical and needed to be set to be within a relatively narrow window for reliable operation. The original cassette interface hardware design was based on some sound ideas, but the implementation just wasn't particularly great.

I wanted none of any of that and at risk of sounding big-headed I will state that my interface circuitry works as flexibly and reliably as anyone could hope for. I have most of my programs and data saved as either MP3 or WAV files, recorded and played back via digital recording devices; namely either a small, portable pocket audio recorder or the desktop PC which resides in my electronics shack.

Programs and data to be loaded into the computer are received via the "CASSETTE READ IN" RCA jack on the front panel. This audio input is buffered by voltage-follower U301A and presents a high input impedance of 100k and therefore negligible loading of the source. In the original TRS-80 the signal input was terminated by a 100R resistor. This was done because no form of buffering was employed and an adequately low source impedance was required for predictable operation of the active filter stage that followed. This is all fine and dandy, so long as the playback device has no issue driving a 100 ohm load (which means that it must have a rather low output impedance already); so not overly flexible.

The TRS-80 saves data by sending out a stream of pulses which repeat at a fixed interval dependant upon the baud rate (250 baud for Level I BASIC and 500 baud for Level II BASIC). Some of the pulses in this repetitive stream go missing though; each one of these represents a logical zero. The pulses that do not go missing are originally produced as a squarewave-approximation of one cycle of a sinewave; two of the bits of the data cassette output register operate as a crude 3-level DAC in combination with an external resistor network.

The recorded pulses, received from a playback device, are bandpass filtered by the combination of the low-pass filter based on U301B followed by the high-pass filter based on U301C. These are both 2-pole Butterworth filters and the corners frequencies are 7 kHz and 200 Hz respectively.

This 200-7000 Hz bandwidth simultaneously passes the wanted signal without unnecessary distortion whist adequately rejecting extraneous junk. I've found that the audio output of my digital playback devices (I'm looking at you, desktop PC) can contain a great deal of high frequency digital noise and interference. A 5m-long audio hook-up lead seems to act as a decent antenna for the emissions of a nasty compact fluorescent lamp too. It only takes one blink/blip of HF interference to poof that 250 baud digital load, which you've perceivably been waiting ages and ages for to finish already, into oblivion.

The received pulses are next full-wave rectified and converted to logic-level signals by splicing comparator U303. The splicing threshold/level is automatically maintained at approximately two-thirds of the peak amplitude of the rectified pulses. This means that the pulse detection works reliably and is relatively insensitive to received signal amplitude over a large dynamic range.

So that I'm not flying blind when connecting up and setting the signal level of an external signal source, I utilised the splicing threshold detector to perform the additional function and utility of driving a small (250 uA FSD) analogue panel meter to give an indication of received signal level. It's handy to have this simple indicator to instantly let you know that both a signal of adequate level is actually being received and that you aren't grossly overdriving the input.

I've reliably loaded programs in both a state of overload and with a signal meter indication of substantially less than one out of ten, but I generally just set for a deflection to around about half scale.

When saving data or programs, the serial data pulse stream to be recorded is sent out of the front panel RCA jack labelled "CASSETTE WRITE OUT". The four-position "CASSETTE WRITE LEVEL" switch permits the signal amplitude to be set to one of the four labelled levels; these being 0.2V, 0.5V, 1V or 2V. These are unloaded peak-peak amplitudes and they drop to approximately half with of 600 ohm load.

Whenever the TRS-80 sends data out of the cassette port with the intent that it be recorded, a bit is set in the cassette port output register (integrated circuit U308) to activate the cassette motor relay. The normally-open contacts of this relay are accessible in my clone design via the 2-pin DIN connector on the front panel, logically labelled "CASSETTE MOTOR".

I've already mentioned that I mostly use digital recording and storage devices for my programs, where there's not much in the way of an electric motor to put under the command of the computer, so it might be wondered why I bothered to replicate this level of original functionality. Well, I happen to have an old reel-to-reel tape recorder that I would eventually like to press into data storage service, just for the alluring inconvenience and unpracticality of it all.

The red LED on the front panel associated with the "CASSETTE MOTOR" DIN connector lights up whenever the cassette motor relay is energized.

Audio line-out and power amplifier

The audio circuitry located on the motherboard is quite simple. Referring to page 4 of the motherboard schematic set, the signal input to the audio power amplifier is under the command of the "AUDIO SOURCE" select switch (connecting to header P401), which has the following four positions:

- **OFF**
  No audio source is selected and no sounds will emanate from the loudspeaker, even with the volume control turned all the way up.

- **S_FX**
  An abbreviation for Sound Effects. This position selects the 2-bit DAC for the cassette port write output (which was universally repurposed by the writers of computer games and hackers alike for sound production) as the signal source. Rather cleverly though, the audio is muted when the register bit that activates the relay which controls the data cassette drive motor is set. This means that you will not hear the screeching sounds of either the incoming or outgoing serial data streams when loading or saving programs or data.

- **WRITE**
  Similar to S_FX, but this time the audio is un-muted only when the data cassette motor relay is energised. This means that you will only hear those outgoing serial data streams that are sent when saving programs or data.

- **READ**
  The opposite of WRITE; you will hear the serial data streams that are received via the data cassette port whilst loading either programs or data. The serial input data flip-flop, whose reset state is controlled by the microprocessor, is selected as the signal source. This means that you will only hear incoming serial data when the computer itself is listening for it.

A good old LM386 audio power amplifier, U401, is utilised for speaker-driving duties and op-amp U302D serves as a line-level output buffer. The line-level output signal connects to the "AUDIO OUT" RCA jack mounted on the front panel of my clone and serves as an output

which can handily drive, say, the audio input of a video monitor or an external power amplifier, if desired. In this case the computers own internal amplifier and speaker can be muted by turning the volume control down to zero; the volume control does not affect the line-level output.

Power supply

There's not much to say about the power supply circuitry. A regulated +5V supply rail is provided for the logic circuitry and a pair of well filtered, though unregulated, supply potentials of approximately +/- 10V are provided for the analogue circuitry of the data cassette interface. The +5V regulator has reserve capacity for powering peripheral devices via the Expansion Port.

A surface-mount, SOT-223 version of the old-school LM7805 linear regulator takes care of the regulated +5V rail, in conjunction with helper transistor Q502, which passes the majority of the load current and dissipates almost all of the regulators wasted power. Q501 serves as a current limiter to protect Q502.

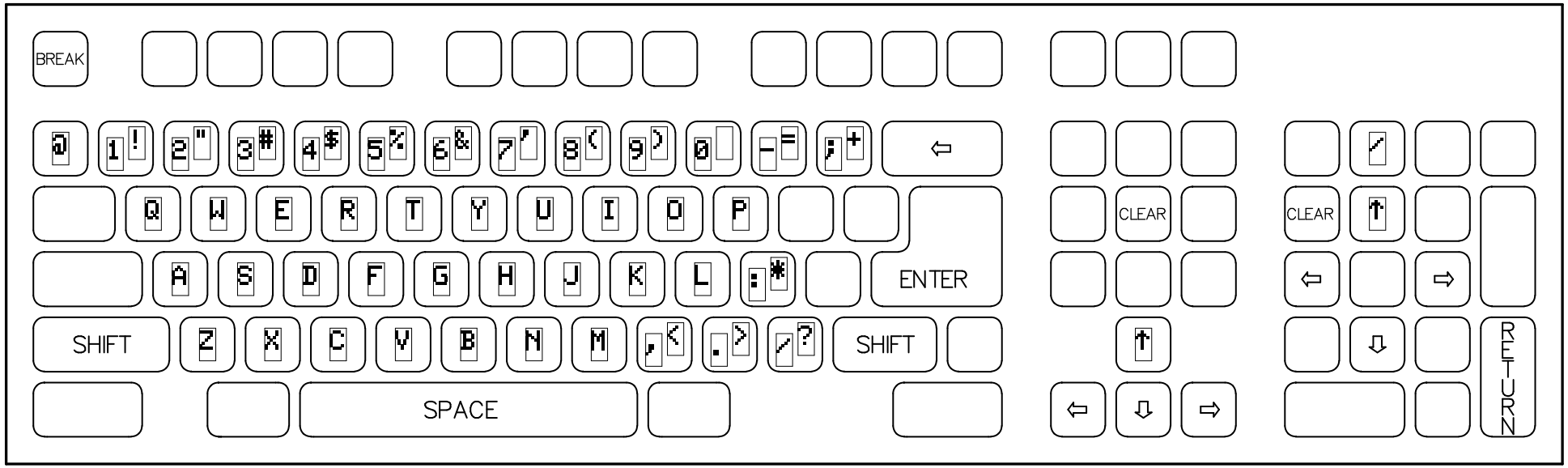
Header P502 serves as an isolation jumper for the +5V rail so that operation of the +5V regulator can be tested and verified without risk of frying the computers logic circuitry. The regulator is short-circuit proof. Operation is checked by ensuring the isolation jumper P502 has not yet been installed, applying power, checking the +5V rail and then shorting the collector of Q502 to ground. If fuse F501 doesn't blow and the collector of Q502 returns to +5V once the short to ground is removed, then the regulator can be assumed to be functional.

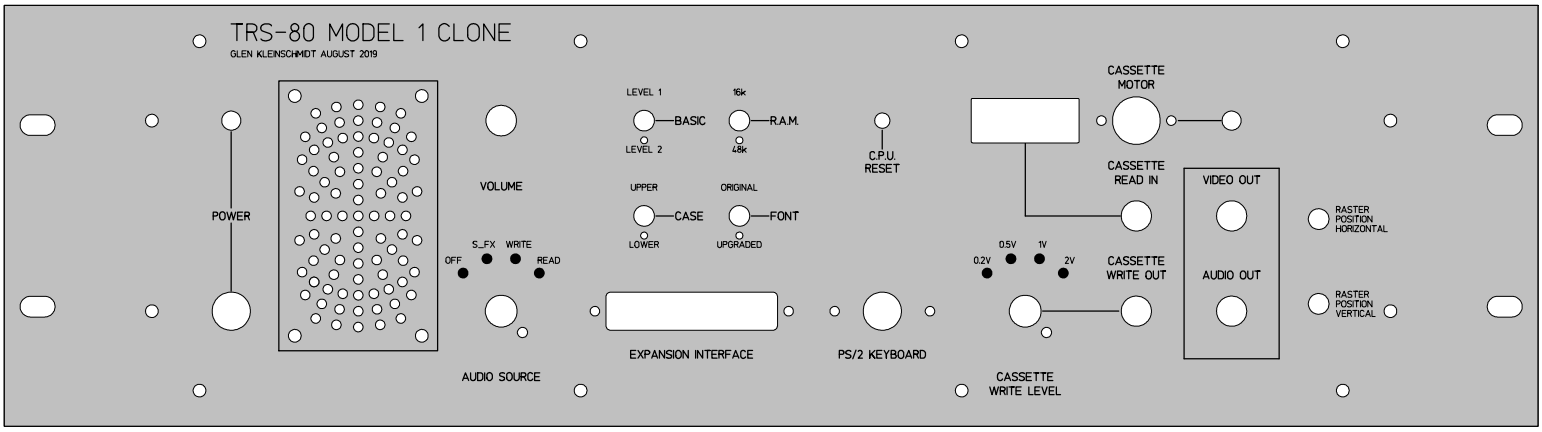
PS/2 keyboard interface

The motherboard of my clone design is compatible with a standard TRS-80 keyboard. It would actually work with an original unit so long as an adaptor cable is soldered up to connect to the 34-pin, double-row boxed header that I've used as the keyboard interface connector.

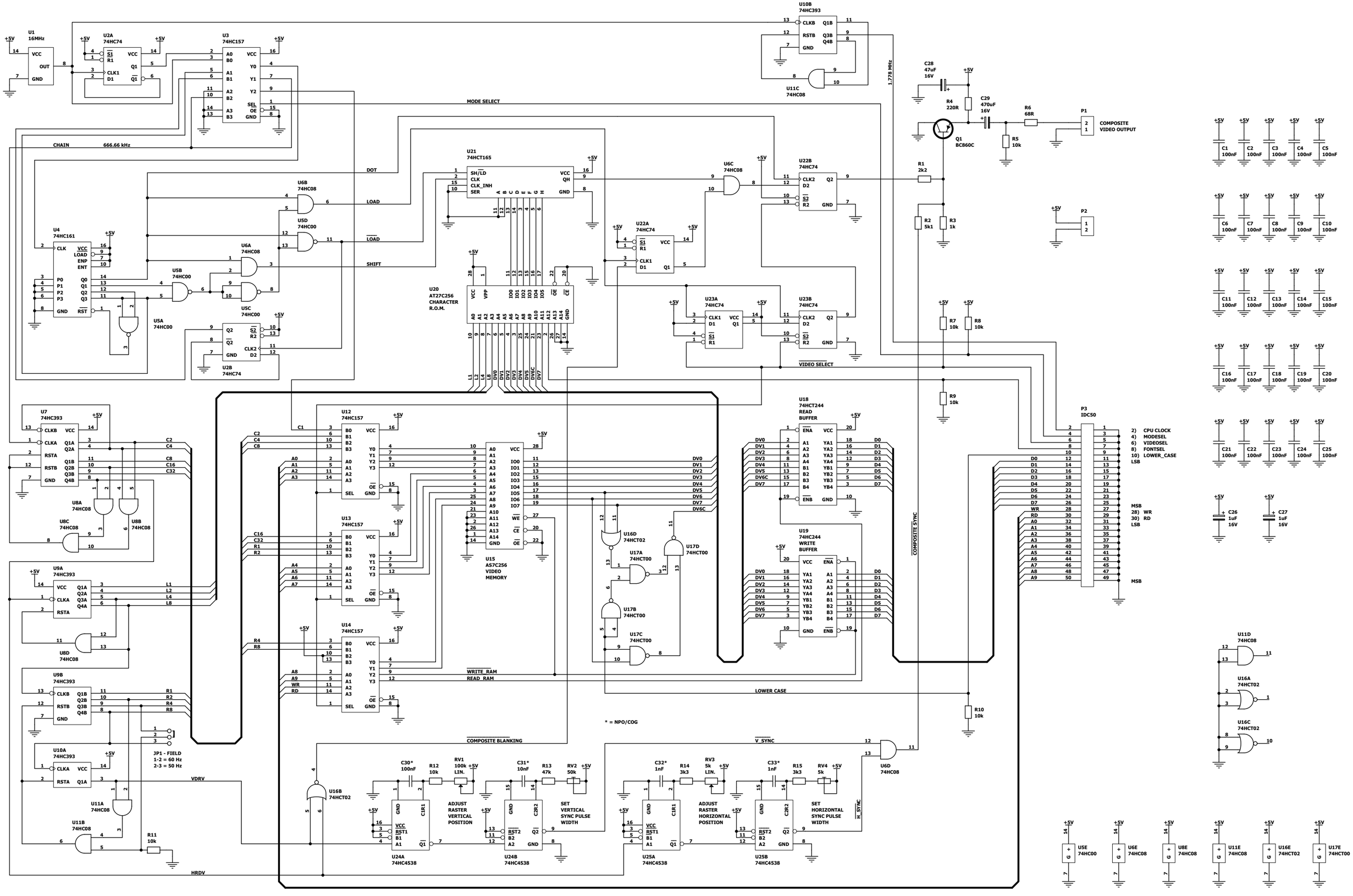
I didn't think that the best option for my clone build, however, was to try to procure an original keyboard assembly or to try to replicate one in current-production electro-mechanical hardware. The simplest and most practical solution was to design a little adaptor board to interface a standard PS/2 keyboard to the TRS-80. This adaptor board which I have designed is universal in the sense that it can interface a PS/2 keyboard to any TRS-80 Model 1, not just my clone design.

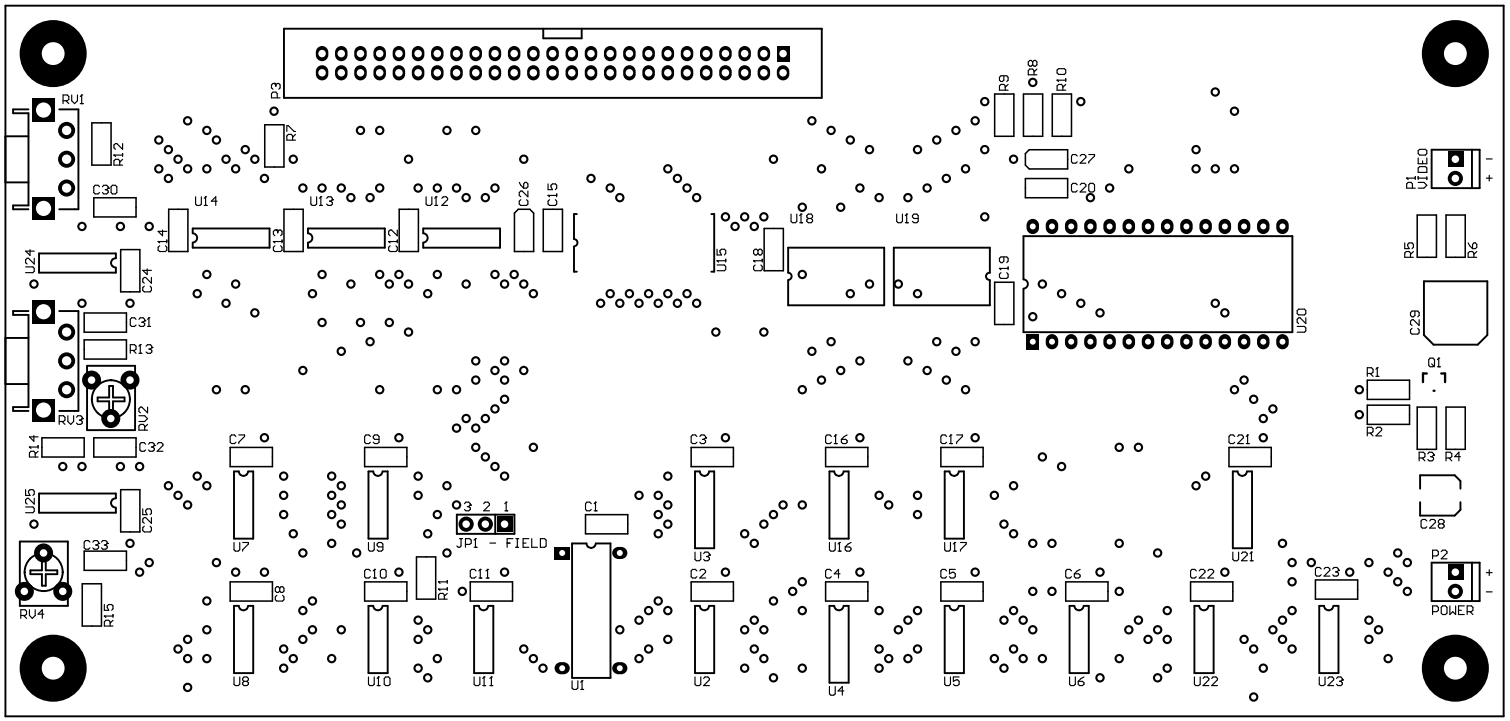
No all of the TRS-80's original keys copy directly over to the PS/2 keyboard layout. For example, BREAK and CLEAR have been assigned to ESC and HOME respectively. There are a few other differences and on the page following this written description is a graphical keyboard legend (standard US layout) which I have produced as a typing aide. I printed this legend out and laminated it, to serve as a handy lookup and reminder card when typing. The differences are few, however, and easily remembered once you get typing.

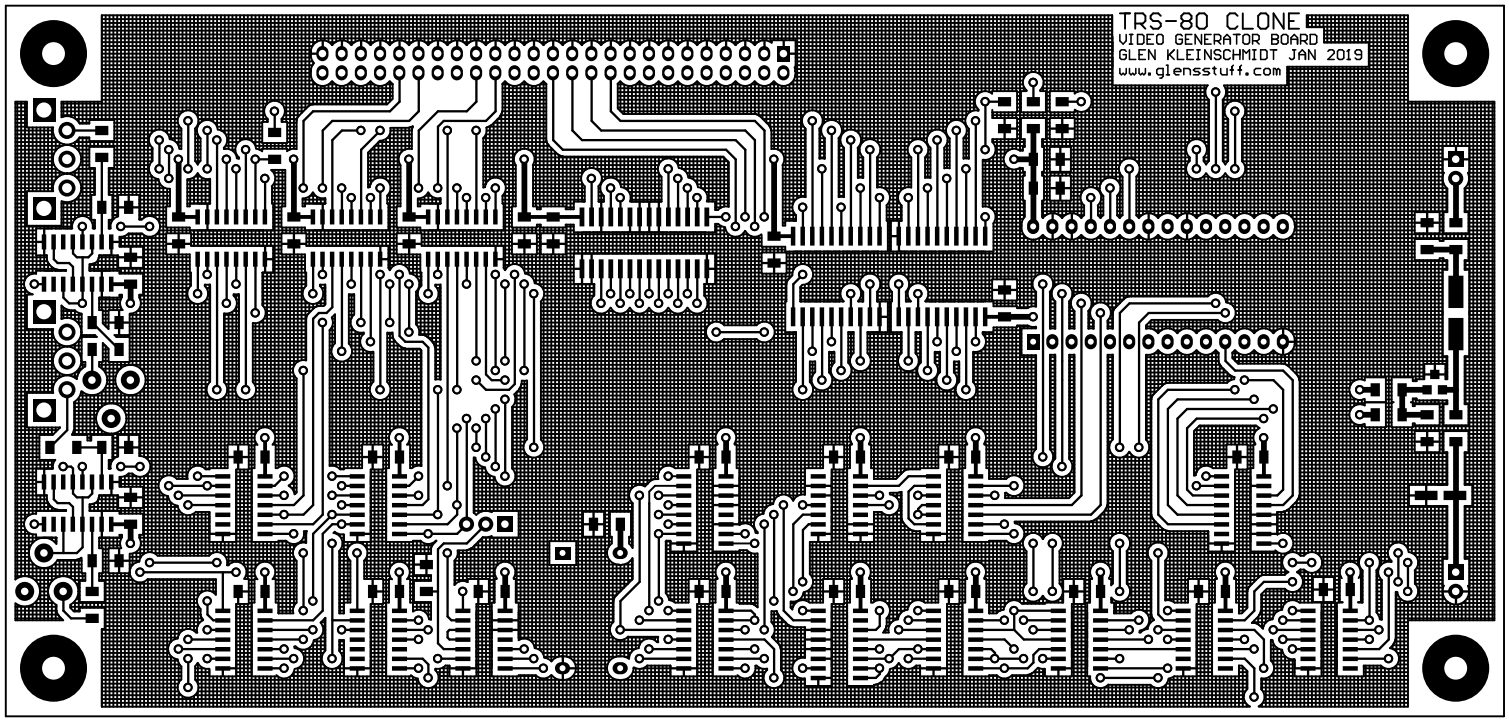

BREAK
@
1 !
2 "
3 #
4 $
5 %
6 &
7 '
8 (
9)
0
- =
; +
⇦
Q W E R T Y U I O P
A S D F G H J K L : *
ENTER
SHIFT
Z X C V B N M , < . > / ?
SHIFT
SPACE
CLEAR
CLEAR
RETURN

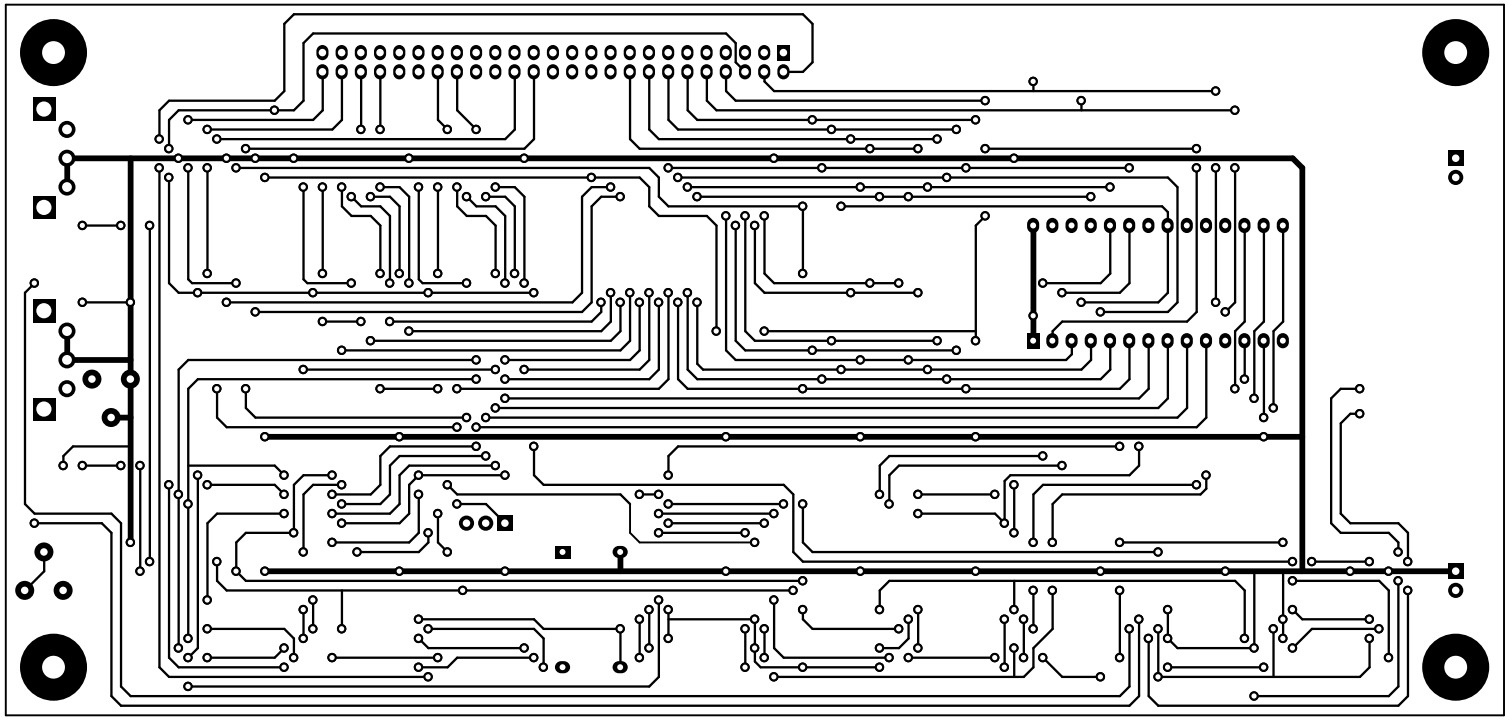

TRS-80 MODEL 1 CLONE
GLEN KLEINSCHMIDT AUGUST 2019
POWER
VOLUME
S_FX
WRITE
OFF
READ
AUDIO SOURCE
LEVEL 1
16k
BASIC
R.A.M.
LEVEL 2
48k
UPPER
ORIGINAL
CASE
FONT
LOWER
UPGRADED
EXPANSION INTERFACE
C.P.U.
RESET
PS/2 KEYBOARD
CASSETTE
MOTOR
CASSETTE
READ IN
0.2V
0.5V
1V
2V
CASSETTE
WRITE OUT
CASSETTE
WRITE LEVEL
VIDEO OUT
AUDIO OUT
RASTER
POSITION
HORIZONTAL
RASTER
POSITION
VERTICAL

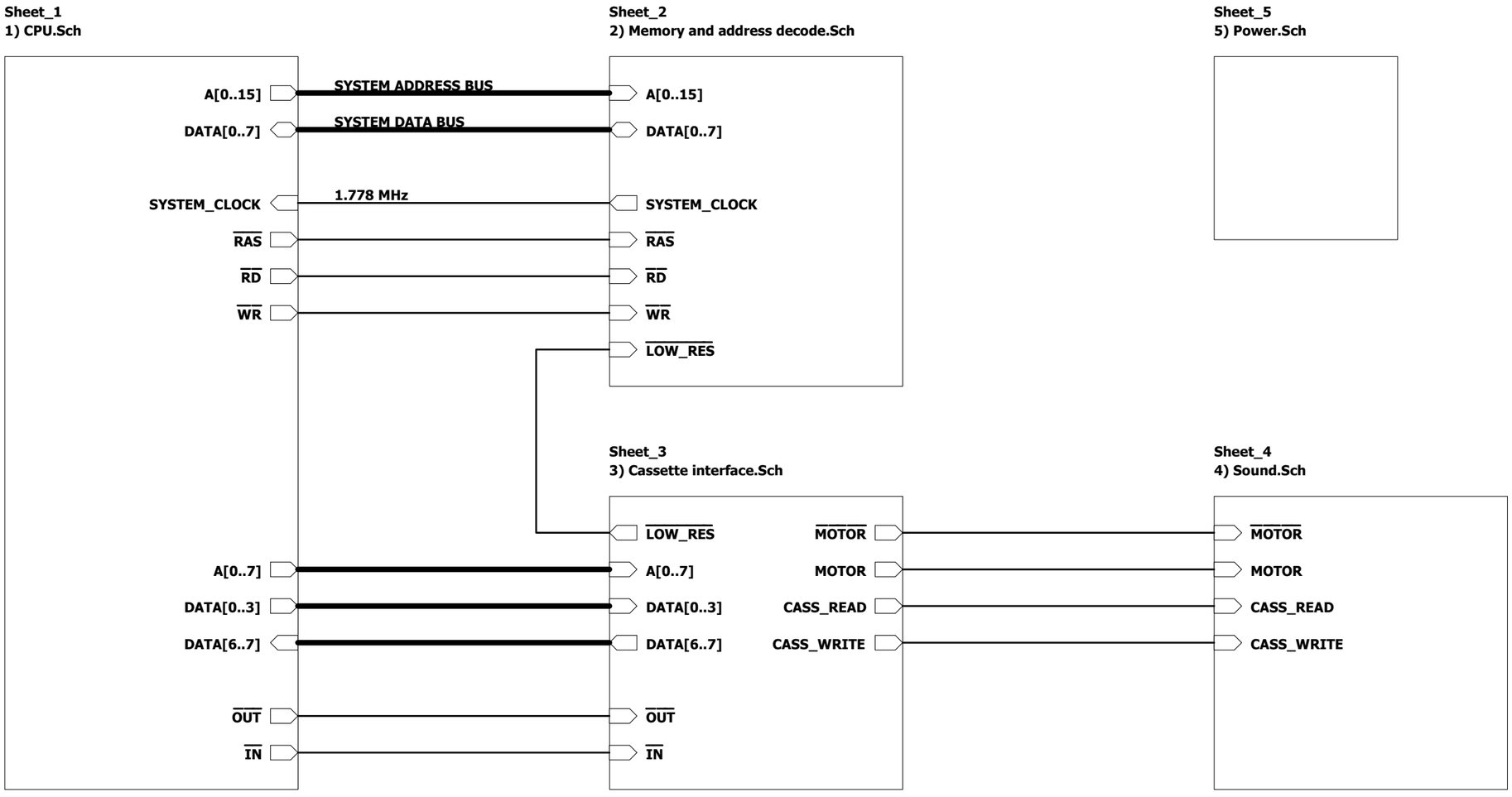
VIDEO GENERATOR BOARD



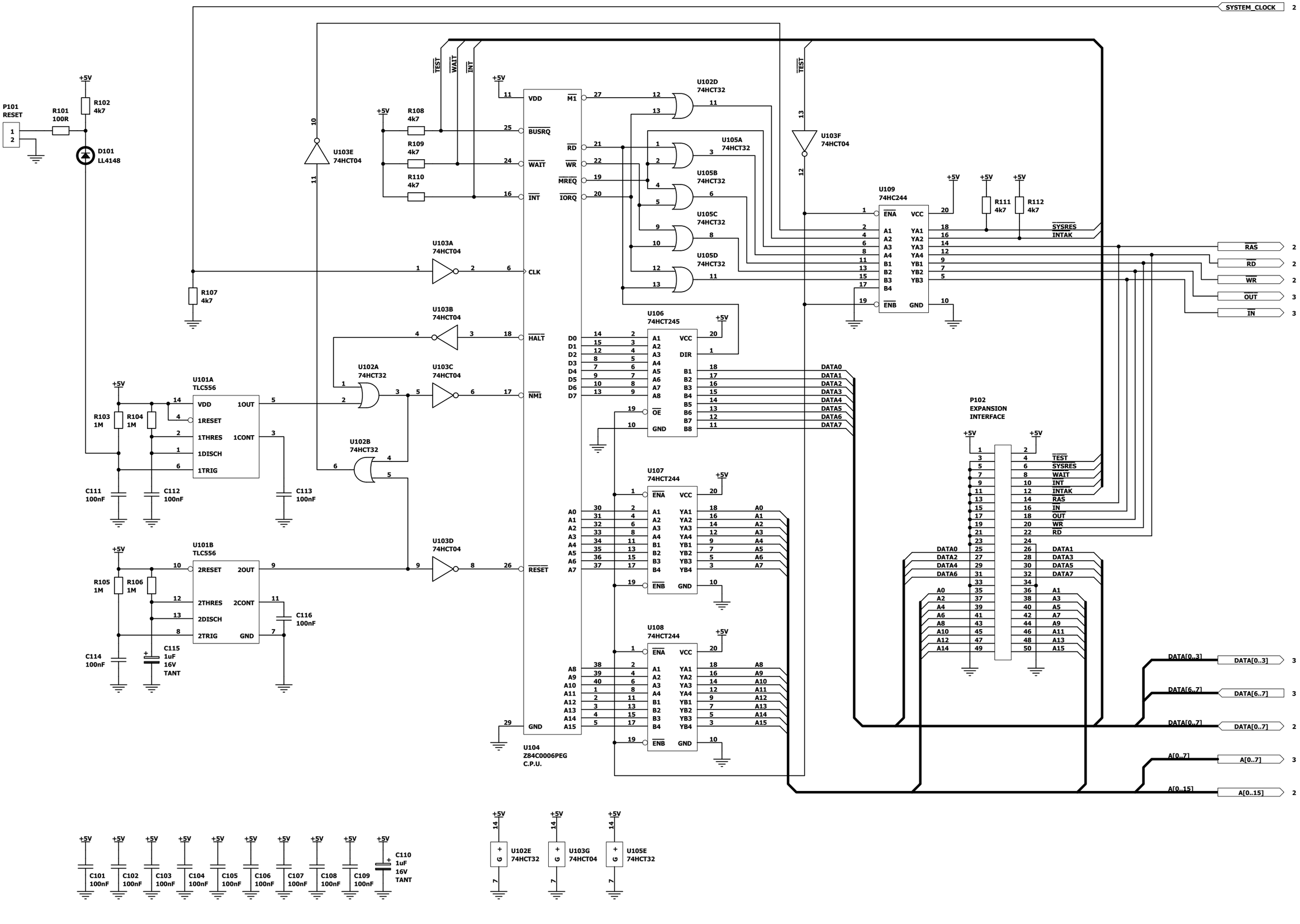

P3
RV1
R12
C30
U14
U13
U12
C26
C15
R7
R9
R8
R10
C27
C20
U8
U19
U15
C18
C19
U20
P1
VIDEO
R5
R6
C29
Q1
R1
R2
R3
R4
C28
U24
C24
C31
R13
RV3
RV2
R14
C32
C7
C9
C3
C16
C17
C21
U25
C25
C33
RV4
R15
JP1 - FIELD
3 2 1
C1
U7
U9
U3
U16
U17
U21
C8
C10
R11
C11
C2
C4
C5
C6
C22
C23
P2
POWER
U8
U10
U11
U1
U2
U4
U5
U6
U22
U23


TRS-80 CLONE
VIDEO GENERATOR BOARD
GLEN KLEINSCHMIDT JAN 2019
www.glensstuff.com

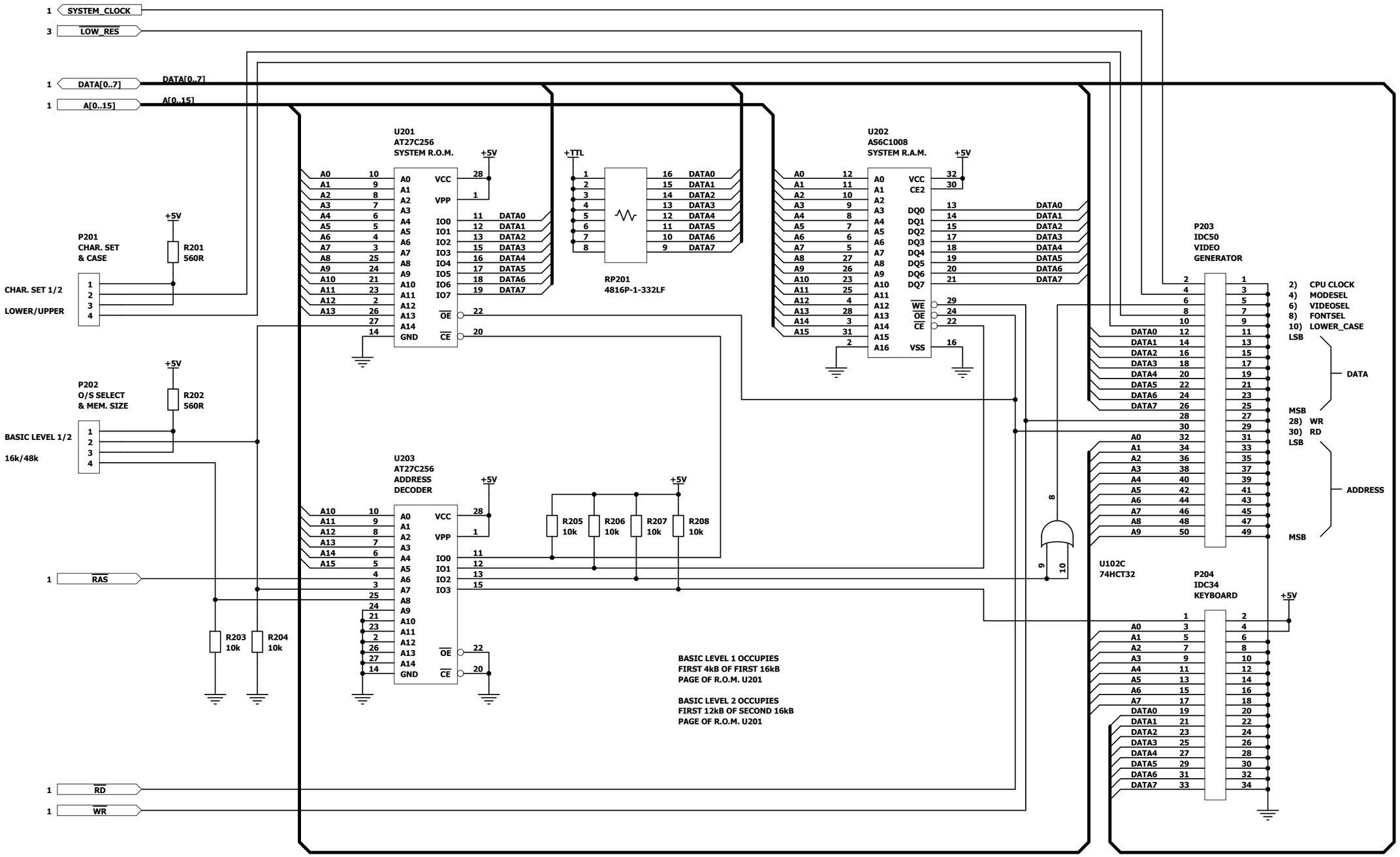


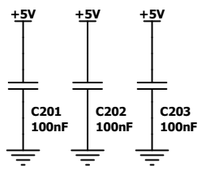

Sheet_1
1) CPU.Sch
A[0..15]
DATA[0..7]
SYSTEM_CLOCK
RAS
RD
WR
A[0..7]
DATA[0..3]
DATA[6..7]
OUT
IN
SYSTEM ADDRESS BUS
SYSTEM DATA BUS
1.778 MHz
Sheet_2
2) Memory and address decode.Sch
A[0..15]
DATA[0..7]
SYSTEM_CLOCK
RAS
RD
WR
LOW_RES
Sheet_3
3) Cassette interface.Sch
LOW_RES
A[0..7]
DATA[0..3]
DATA[6..7]
OUT
IN
MOTOR
CASS_READ
CASS_WRITE
Sheet_4
4) Sound.Sch
MOTOR
CASS_READ
CASS_WRITE
Sheet_5
5) Power.Sch

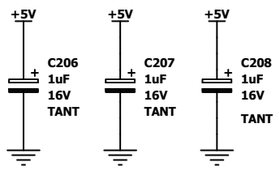
1) CPU

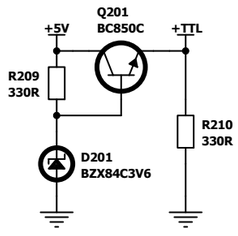

SYSTEM_CLOCK
P101 RESET
R101 100R
R102 4k7
D101 LL4148
R107 4k7
U103E 74HCT04
R108 4k7
R109 4k7
R110 4k7
TEST
WAIT
INT
U102D 74HCT32
U105A 74HCT32
U105B 74HCT32
U105C 74HCT32
U105D 74HCT32
U103F 74HCT04
U109 74HC244
R111 4k7
R112 4k7
SYSRES
INTAK
RAS
RD
WR
OUT
IN
U103A 74HCT04
U103B 74HCT04
U103C 74HCT04
U102A 74HCT32
U102B 74HCT32
U103D 74HCT04
U106 74HCT245
DATA0
DATA1
DATA2
DATA3
DATA4
DATA5
DATA6
DATA7
P102 EXPANSION INTERFACE
U101A TLC556
R103 1M
R104 1M
C111 100nF
C112 100nF
C113 100nF
U101B TLC556
R105 1M
R106 1M
C114 100nF
C115 1uF 16V TANT
C116 100nF
U107 74HCT244
U108 74HCT244
U104 Z84C0006PEG C.P.U.
CLK
HALT
NMI
RESET
BUSRQ
M1
RD
WR
MREQ
IORQ
DATA[0..3]
DATA[4..7]
DATA[0..7]
A[0..7]
A[0..15]
C101 100nF
C102 100nF
C103 100nF
C104 100nF
C105 100nF
C106 100nF
C107 100nF
C108 100nF
C109 100nF
C110 1uF 16V TANT
U102E 74HCT32
U103G 74HCT04
U105E 74HCT32

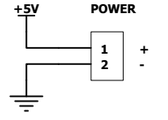
2) MEMORY AND ADDRESS DECODE



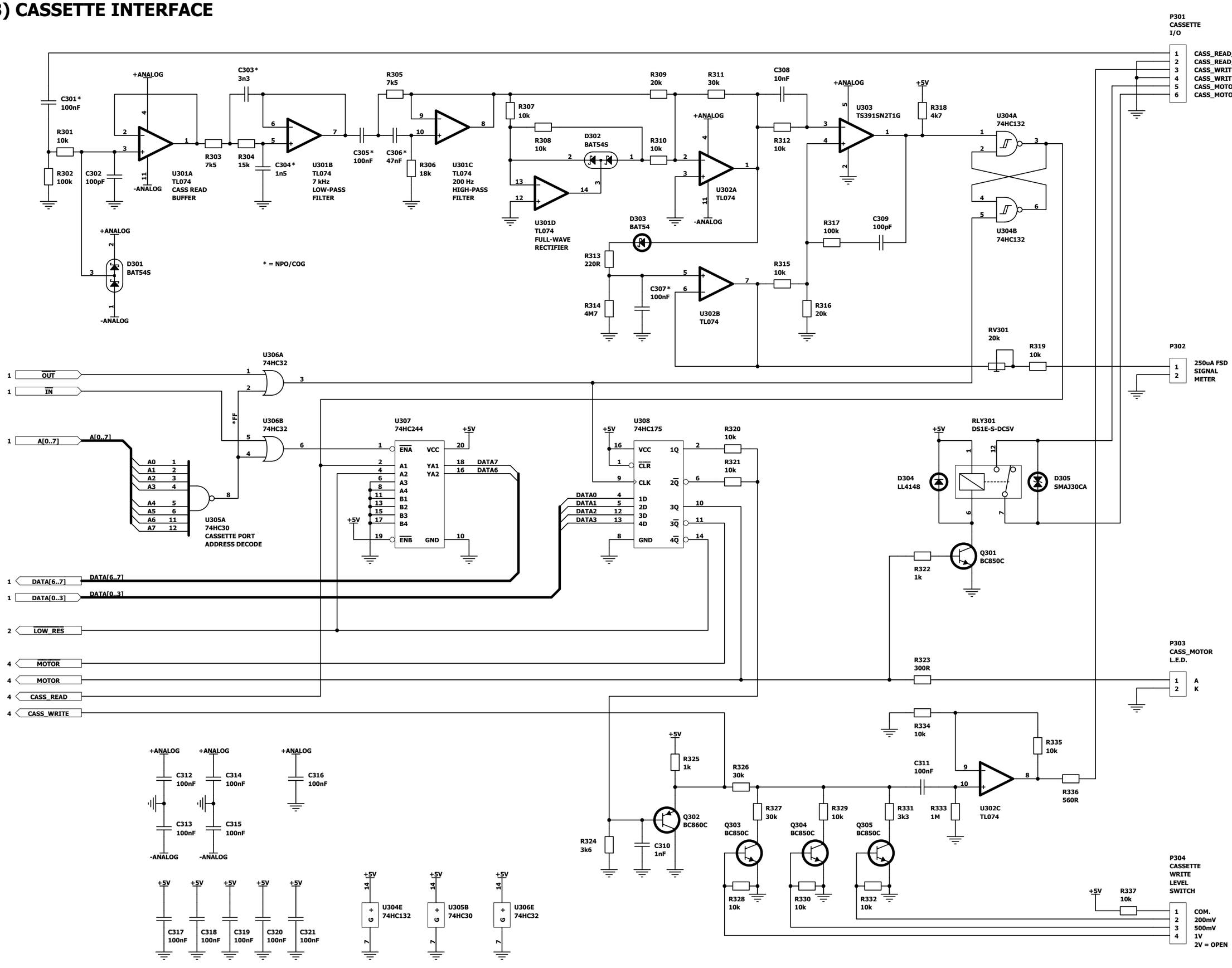




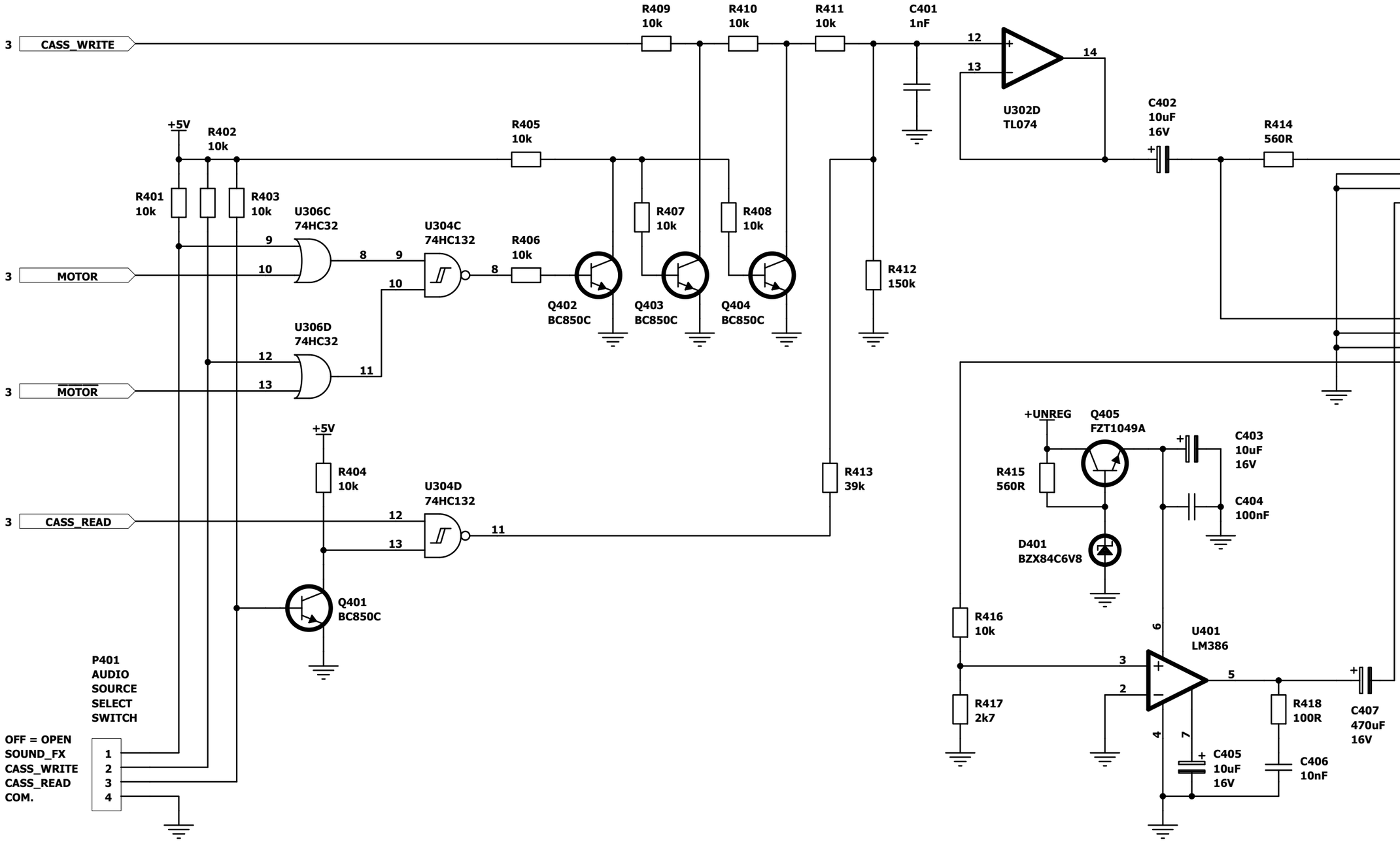




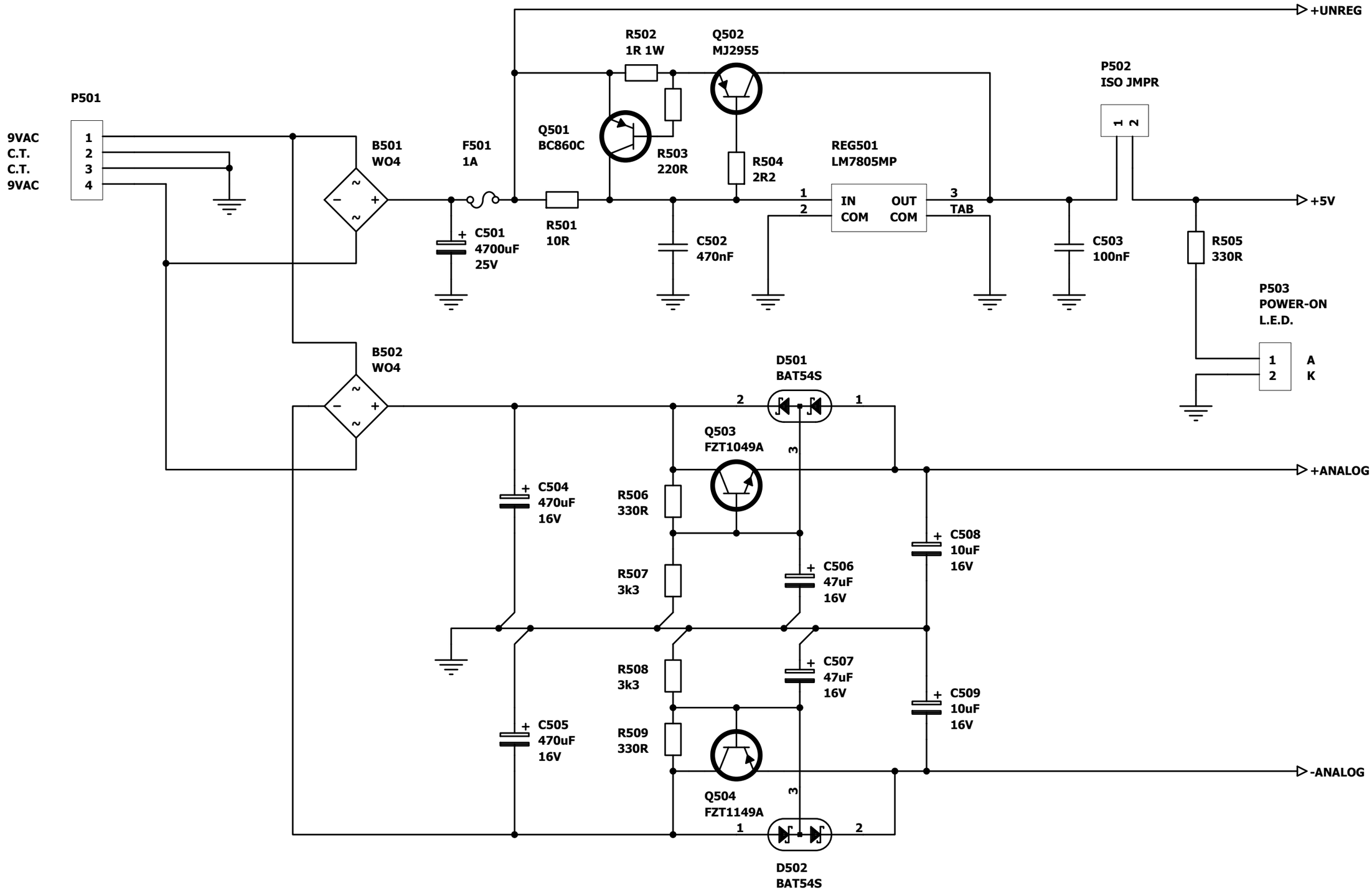
3) CASSETTE INTERFACE

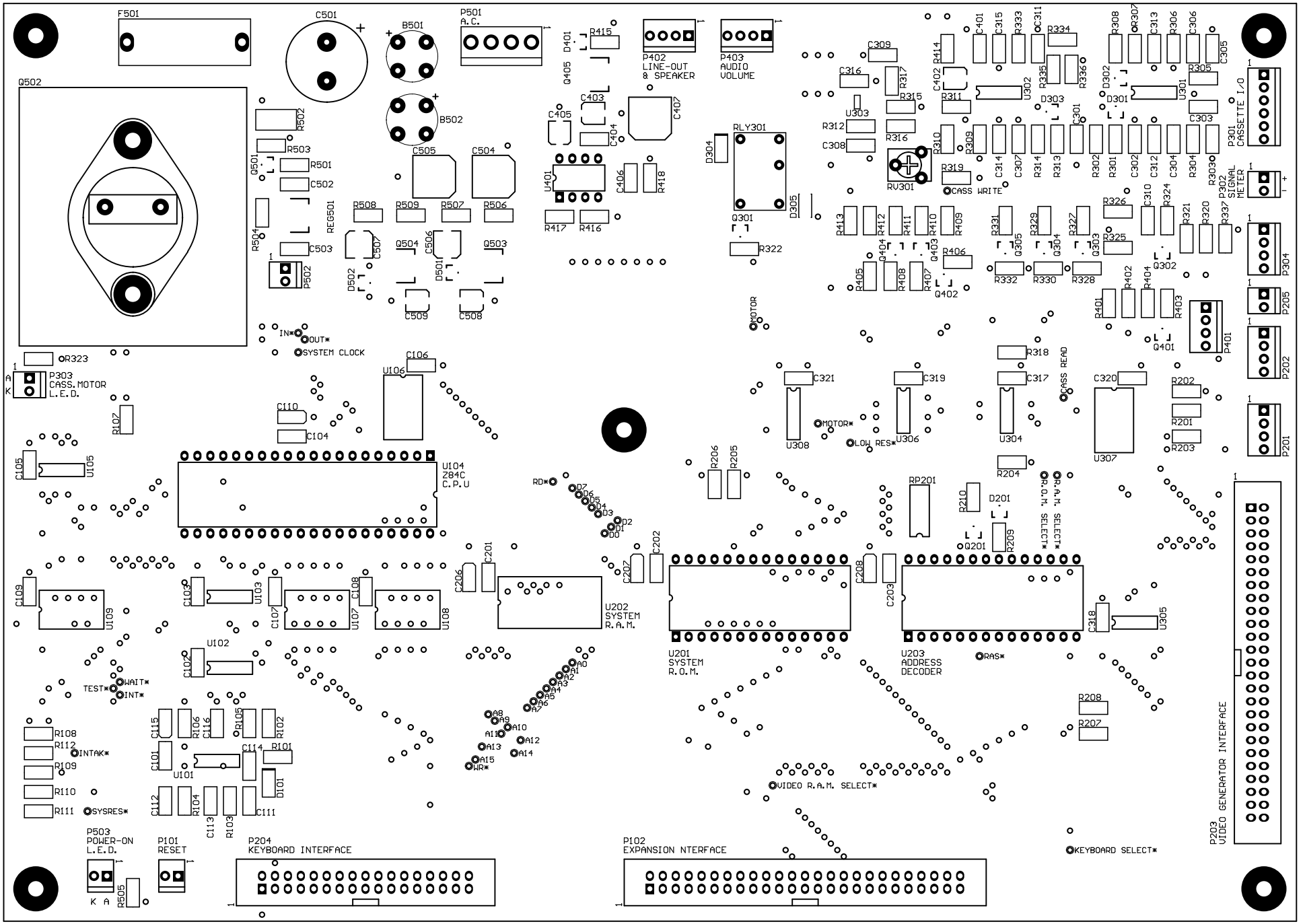


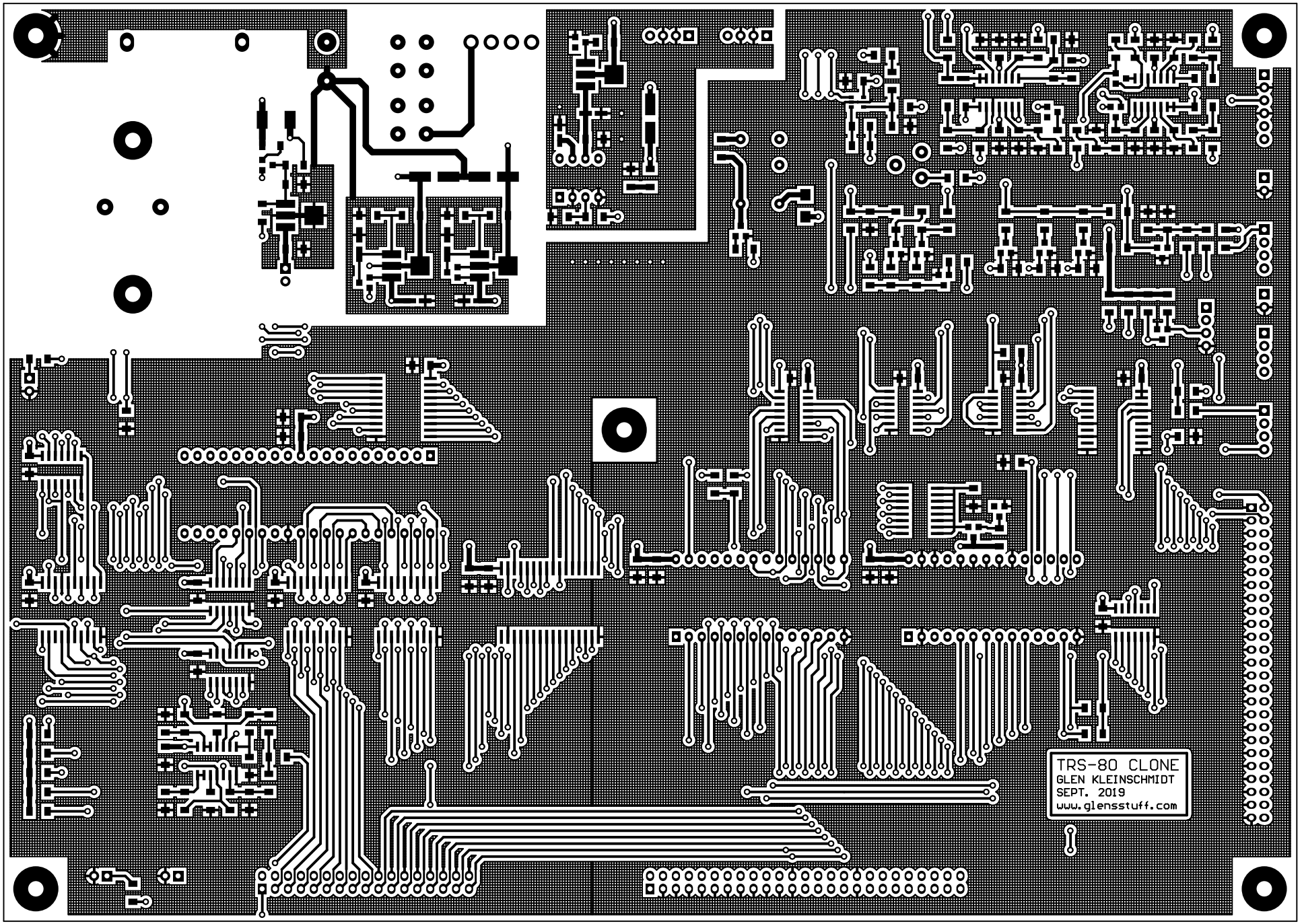
4) SOUND

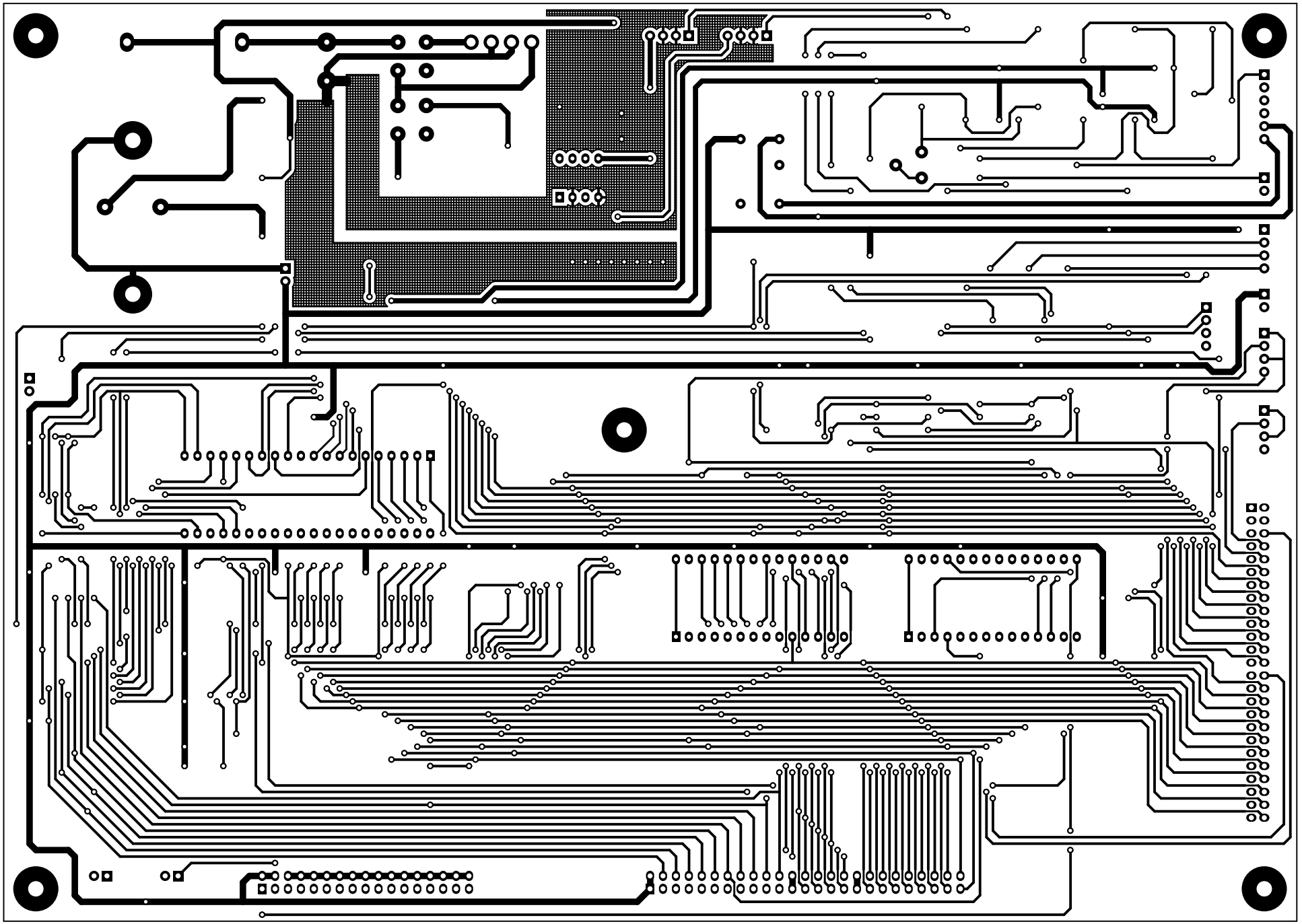

3 CASS_WRITE
R409 10k
R410 10k
R411 10k
C401 1nF
12
13
14
U302D TL074
C402 10uF 16V
R414 560R
P402 AUDIO OUTPUT
1
2
3
4
LINE (1Vp-p)
SPEAKER (8R)
+5V
R402 10k
R405 10k
R401 10k
R403 10k
U306C 74HC32
U304C 74HC132
R406 10k
R407 10k
R408 10k
3 MOTOR
Q402 BC850C
Q403 BC850C
Q404 BC850C
R412 150k
P403 VOLUME CONTROL
U306D 74HC32
3 MOTOR
+5V
R404 10k
U304D 74HC132
R413 39k
3 CASS_READ
Q401 BC850C
+UNREG
Q405 FZT1049A
C403 10uF 16V
R415 560R
C404 100nF
D401 BZX84C6V8
R416 10k
U401 LM386
R417 2k7
R418 100R
C407 470uF 16V
C405 10uF 16V
C406 10nF
P401 AUDIO SOURCE SELECT SWITCH
OFF = OPEN
SOUND_FX
CASS_WRITE
CASS_READ
COM.
1
2
3
4

5) POWER

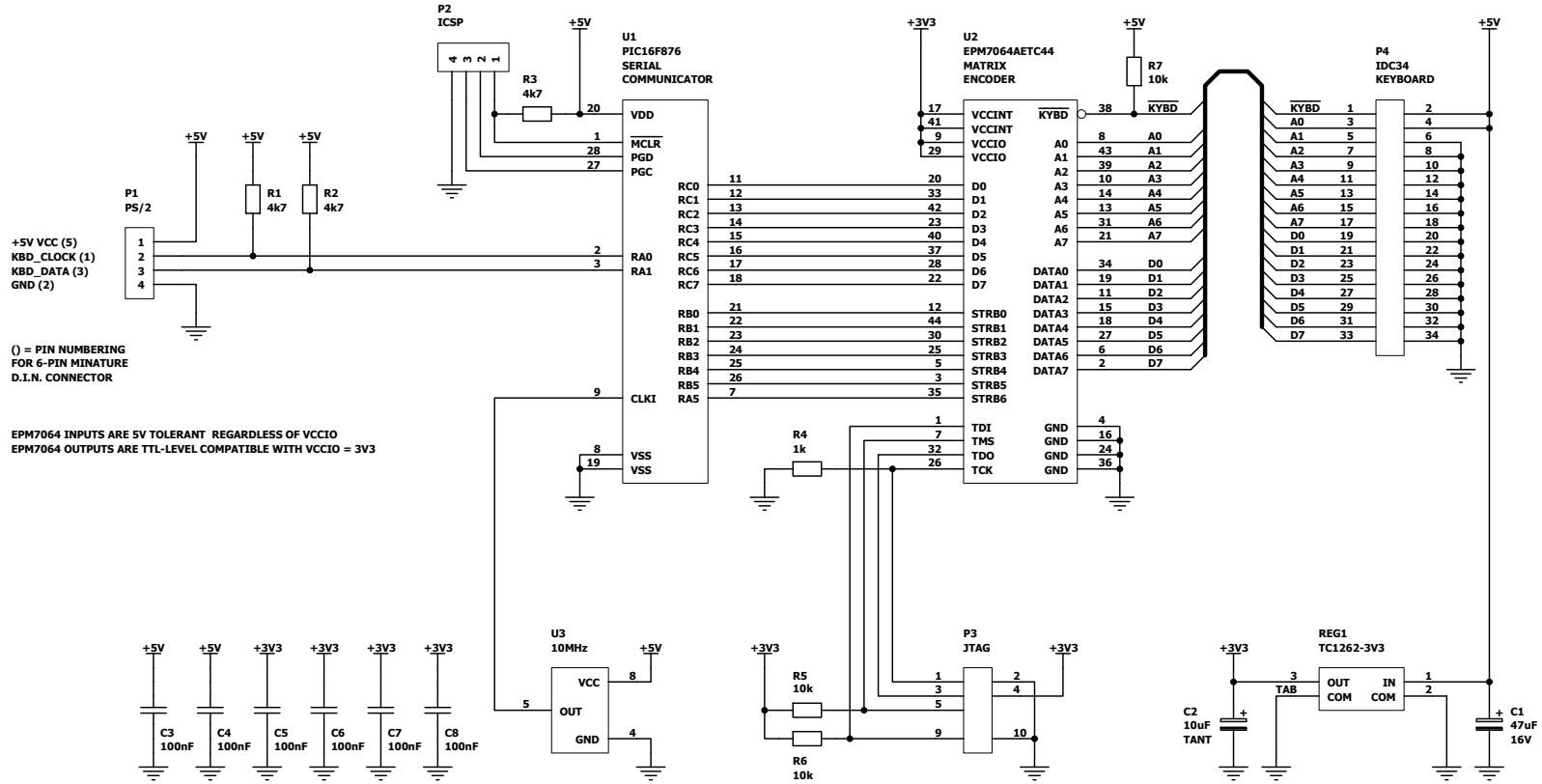

P501
9VAC
C.T.
C.T.
9VAC
1
2
3
4
B501
WO4
F501
1A
C501
4700uF
25V
R501
10R
Q501
BC860C
R502
1R 1W
R503
220R
Q502
MJ2955
R504
2R2
REG501
LM7805MP
IN
COM
OUT
COM
TAB
C502
470nF
+UNREG
P502
ISO JMPR
C503
100nF
+5V
R505
330R
P503
POWER-ON
L.E.D.
A
K
B502
WO4
D501
BAT54S
Q503
FZT1049A
+ANALOG
C504
470uF
16V
R506
330R
R507
3k3
C506
47uF
16V
C508
10uF
16V
R508
3k3
C507
47uF
16V
C509
10uF
16V
C505
470uF
16V
R509
330R
-ANALOG
Q504
FZT1149A
D502
BAT54S

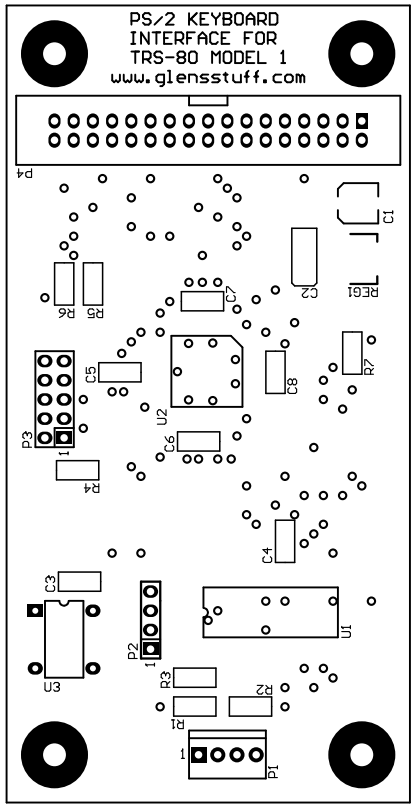

F501
C501
B501
P501
A.C.
P402
LINE-OUT
& SPEAKER
P403
AUDIO
VOLUME
Q502
B502
C505
C504
R502
R503
R501
C502
REG501
R504
C503
P502
R508
R509
R507
R506
Q504
Q503
C506
C507
D502
D501
C509
C508
R415
D401
Q405
C403
C405
C404
C407
U401
C406
R418
R417
R416
RLY301
D304
D305
Q301
R322
C309
C316
R317
R315
U303
R316
R312
C308
RV301
R310
R319
CASS WRITE
R414
C402
C401
C315
R333
C311
R334
R335
R336
U302
D303
R308
R307
C313
R306
C306
C305
D302
D301
U301
R305
C303
P301
CASSETTE I/O
R309
C314
C307
R314
R313
C301
R302
R301
C302
C312
C304
R304
R303
P302
SIGNAL
METER
R413
R412
R411
R410
R409
R331
R329
R327
C310
R324
R326
R321
R320
R337
P304
Q404
Q403
R406
Q305
Q304
Q303
R325
Q302
R405
R408
R407
Q402
R332
R330
R328
R401
R402
R404
R403
Q401
P401
P205
P202
P201
IN*
OUT*
SYSTEM CLOCK
MOTOR
R323
P303
CASS. MOTOR
L.E.D.
C106
U106
C321
C319
R318
C317
CASS READ
C320
R202
R201
R203
U308
MOTOR*
LOW RES*
U306
U304
U307
C110
C104
R107
C105
U105
U104
Z84C
C.P.U
R206
R205
RD*
D7
D6
D5
D4
D3
D2
D1
D0
RP201
R210
D201
Q201
R209
R.A.M. SELECT*
R.O.M. SELECT*
R204
C109
U109
C103
U103
C107
C108
U107
U108
C206
C201
C202
C207
U202
SYSTEM
R.A.M.
C208
C203
C318
U305
U102
C102
U201
SYSTEM
R.O.M.
U203
ADDRESS
DECODER
RAS*
TEST*
WAIT*
INT*
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15
WR*
R108
R112
INTAK*
R109
R110
R111
SYSRES*
C115
R106
C116
R105
R102
C101
C114
R101
U101
D101
C112
R104
C113
R103
C111
R208
R207
VIDEO R.A.M. SELECT*
KEYBOARD SELECT*
P203
VIDEO GENERATOR INTERFACE
P503
POWER-ON
L.E.D.
K A
R505
P101
RESET
P204
KEYBOARD INTERFACE
P102
EXPANSION NTERFACE

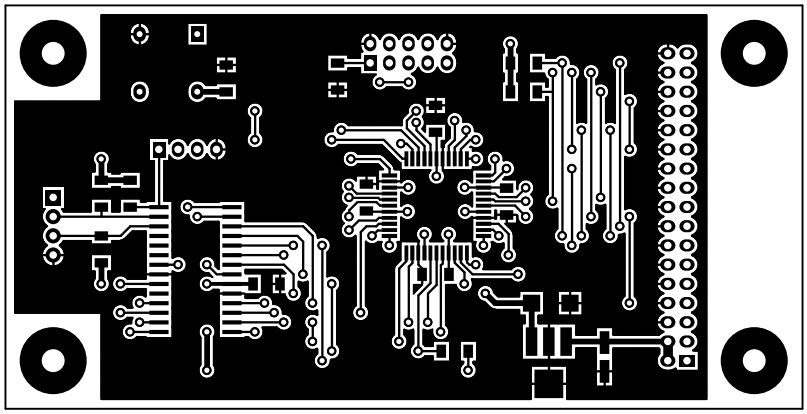

TRS-80 CLONE
GLEN KLEINSCHMIDT
SEPT. 2019
www.glensstuff.com

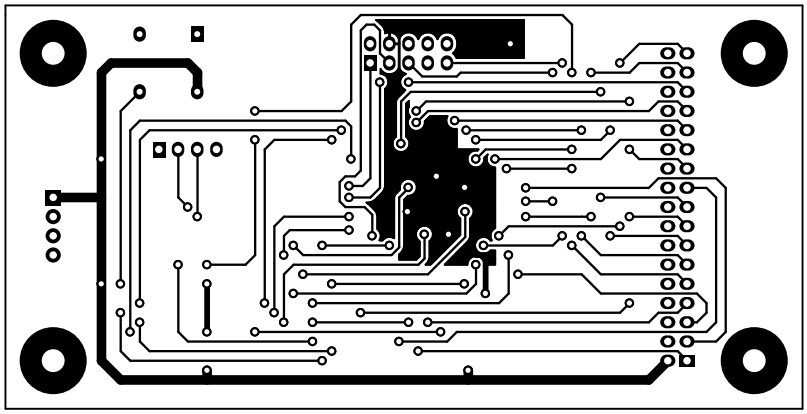


TANDY TRS-80 CLONE
PS/2 KEYBOARD INTERFACE

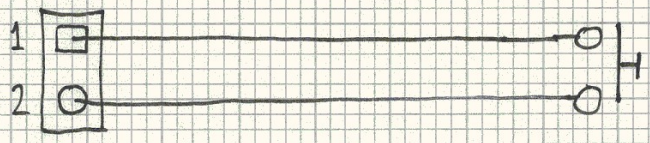



PS/2 KEYBOARD
INTERFACE FOR
TRS-80 MODEL 1
www.glensstuff.com
P4
C1
C2
R6
R5
C7
R7
C5
C8
U2
P3
1
C6
R4
C4
C3
U1
P2
1
U3
R3
R2
R1
1
P1



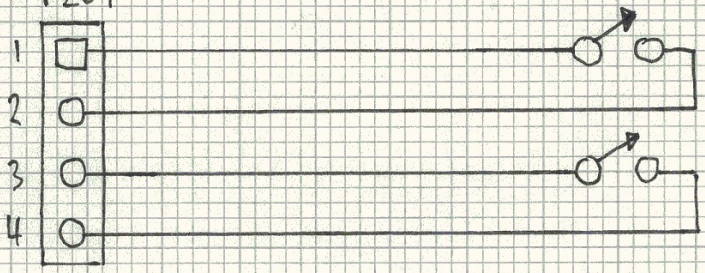


P101



"C.P.U. RESET"
NORMALLY OPEN
MOMENTARY PUSH BUTTON

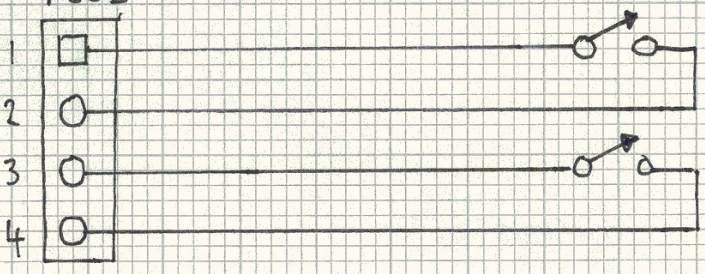
P201



"FONT"
OPEN = "ORIGINAL" CHARACTER SET
CLOSED = "UPGRADED" CHARACTER SET

"CASE"
OPEN = "UPPER"
CLOSED = "LOWER" – LOWER CASE MOD. ENABLED.

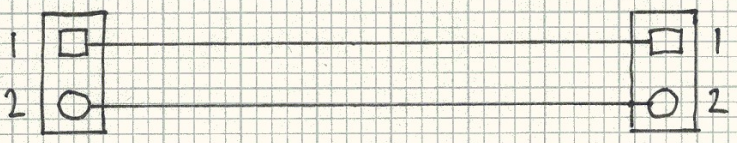
P202



"BASIC"
OPEN = "LEVEL I"
CLOSED = "LEVEL II"

"R.A.M."
OPEN = "16K"
CLOSED = "48K"

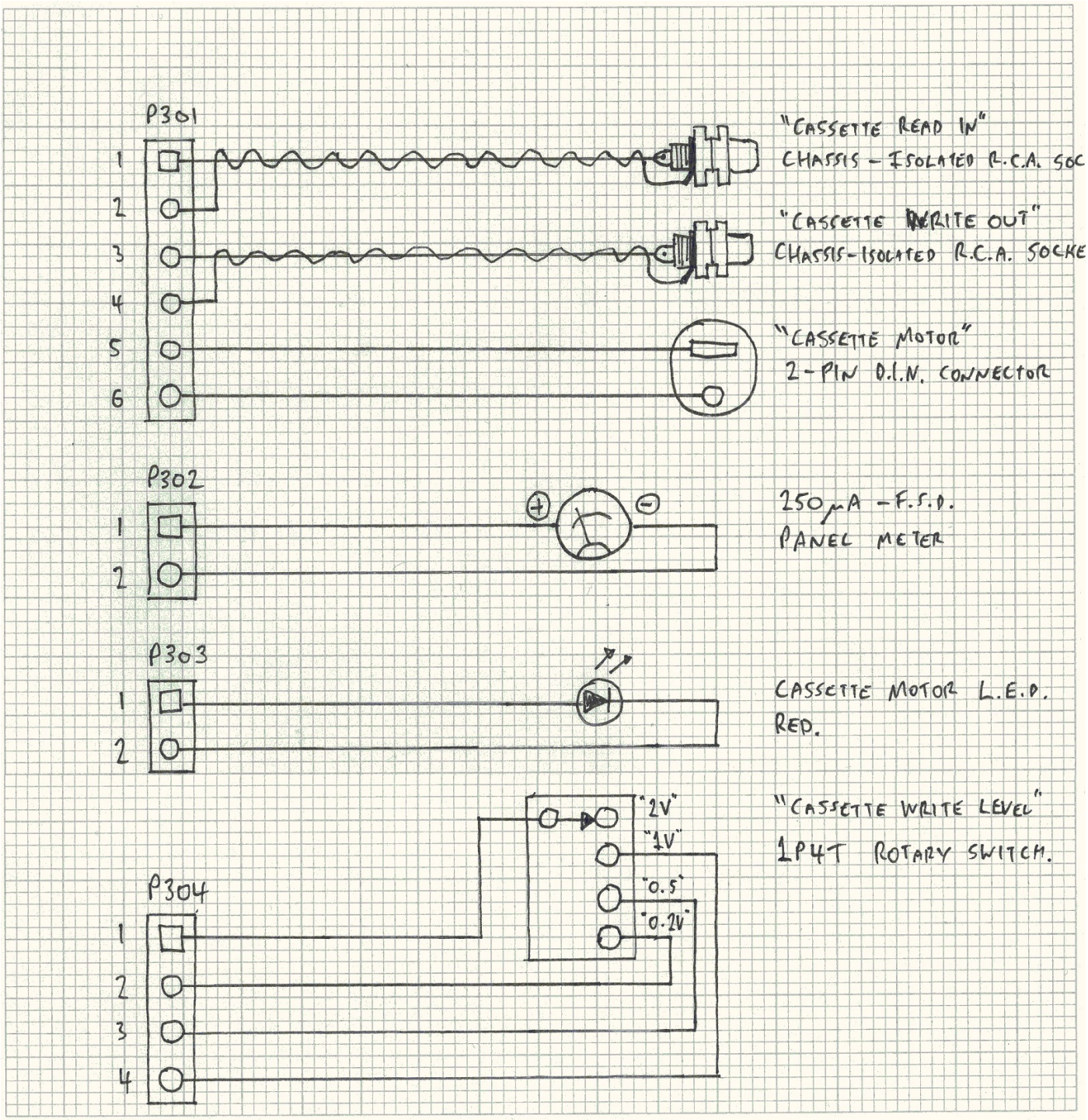
P205

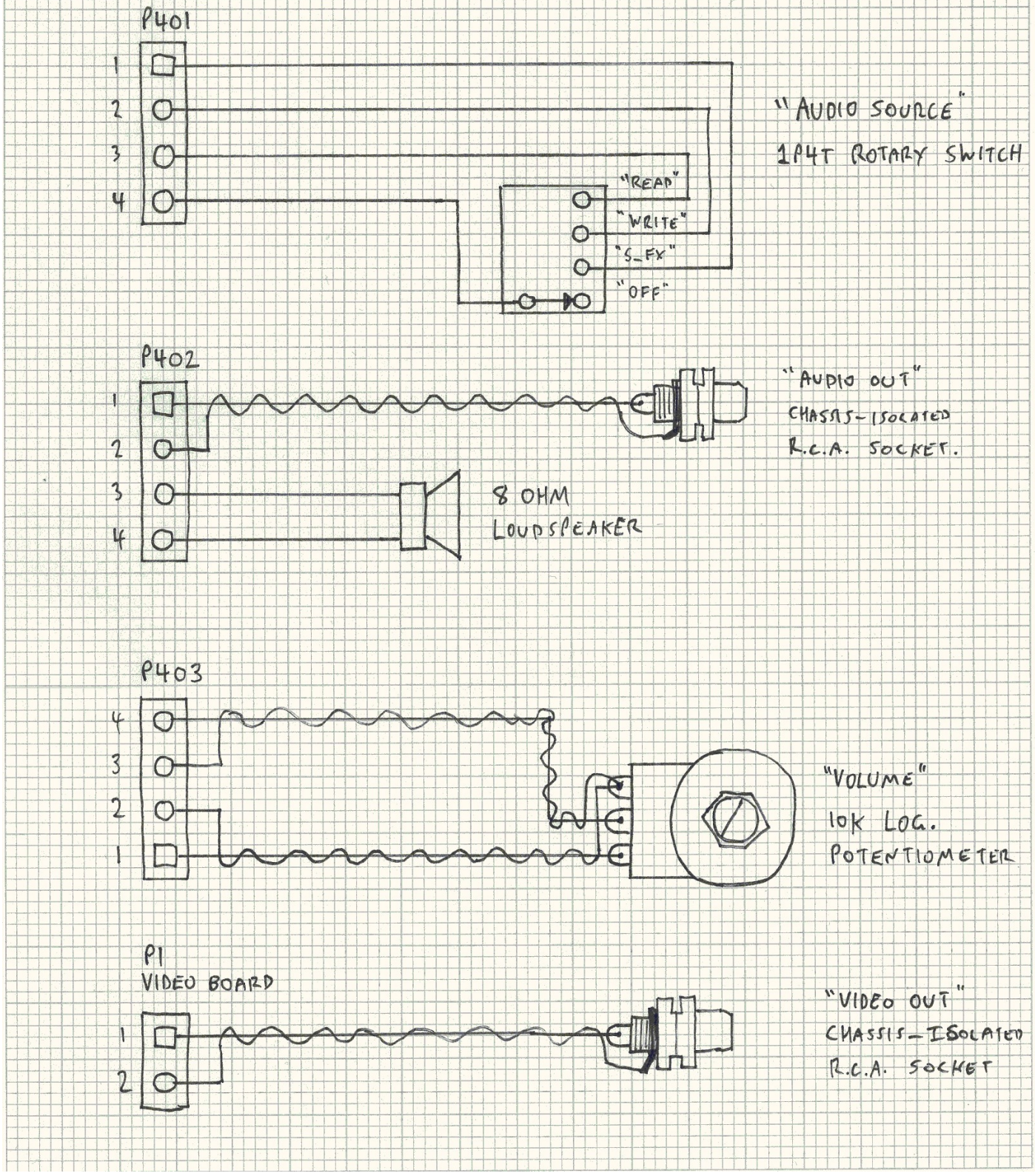


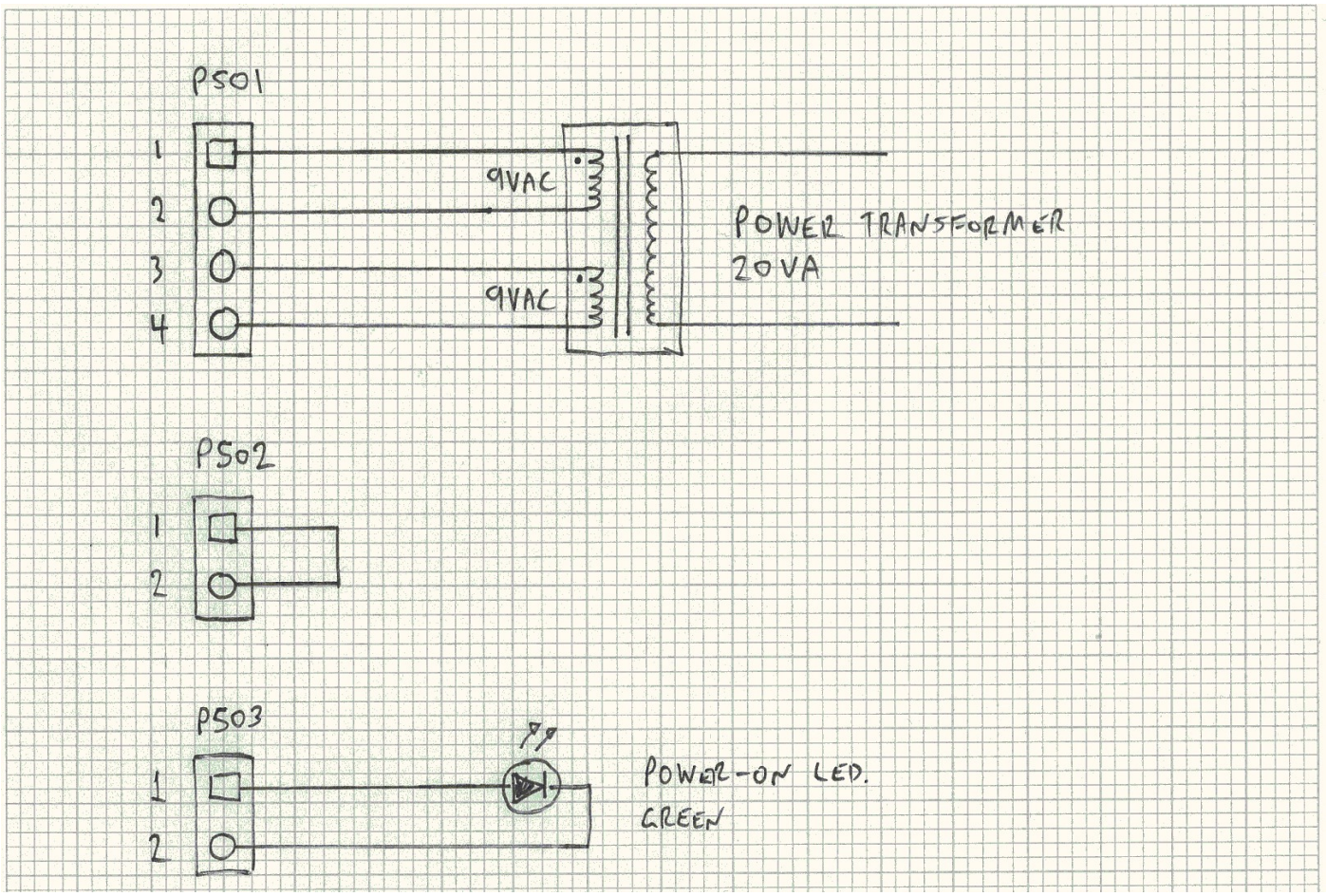
P2 OF VIDEO GENERATOR BOARD

P203 ON MOTHERBOARD CONNECTS TO P3 ON VIDEO GENERATOR BOARD VIA 50-WAY RIBBON CABLE.

P204 ON MOTHERBOARD CONNECTS TO P4 ON PS/2 KEYBOARD INTERFACE BOARD VIA 34-WAY RIBBON CABLE.


P301
1
2
3
4
5
6
"CASSETTE READ IN"
CHASSIS - ISOLATED R.C.A. SOC
"CASSETTE WRITE OUT"
CHASSIS-ISOLATED R.C.A. SOCKE
"CASSETTE MOTOR"
2-PIN D.I.N. CONNECTOR
P302
1
2
+
-
250μA - F.S.D.
PANEL METER
P303
1
2
CASSETTE MOTOR L.E.D.
RED.
P304
1
2
3
4
"2V"
"1V"
"0.5"
"0.2V"
"CASSETTE WRITE LEVEL"
1P4T ROTARY SWITCH.


P401
1
2
3
4
"READ"
"WRITE"
"S_FX"
"OFF"
"AUDIO SOURCE"
1P4T ROTARY SWITCH
P402
1
2
3
4
"AUDIO OUT"
CHASSIS-ISOLATED
R.C.A. SOCKET.
8 OHM
LOUDSPEAKER
P403
4
3
2
1
"VOLUME"
10K LOG.
POTENTIOMETER
P1
VIDEO BOARD
1
2
"VIDEO OUT"
CHASSIS-ISOLATED
R.C.A. SOCKET


PS01
1
2
3
4
9VAC
9VAC
POWER TRANSFORMER
20VA
PS02
1
2
PS03
1
2
POWER-ON LED.
GREEN

Parts for Motherboard PCB

Designator	Component type	Value	Part #	Manufacturer	Package	Description	Quantity
R502	Resistor	1R 1W		Various	2512	Chip, thin film	1
R504	Resistor	2R2		Various	1206	Chip, thin film	1
R501	Resistor	10R		Various	1206	Chip, thin film	1
R101, R418	Resistor	100R		Various	1206	Chip, thin film	2
R313, R503	Resistor	220R		Various	1206	Chip, thin film	2
R323	Resistor	300R		Various	1206	Chip, thin film	1
R209, R210, R505, R506, R509	Resistor	330R		Various	1206	Chip, thin film	5
R201, R202, R336, R414, R415	Resistor	560R		Various	1206	Chip, thin film	5
R322, R325	Resistor	1k		Various	1206	Chip, thin film	2
R417	Resistor	2k7		Various	1206	Chip, thin film	1
R507, R508, R331	Resistor	3k3		Various	1206	Chip, thin film	3
RP201	Resistor array	3k3 x 8	4816P-1-332LF	Bournes	16 pin SMT		1
R324	Resistor	3k6		Various	1206	Chip, thin film	1
R102, R107, R108, R109, R110, R111, R112, R318	Resistor	4k7		Various	1206	Chip, thin film	8
R303, R305	Resistor	7k5		Various	1206	Chip, thin film	2
R203, R204, R205, R206, R207, R208, R301, R307, R308, R310, R312, R315, R319, R320, R321, R328, R329, R330, R332, R334, R335, R337, R401, R402, R403, R404, R405, R406, R407, R408, R409, R410, R411, R416	Resistor	10k		Various	1206	Chip, thin film	34
R304	Resistor	15k		Various	1206	Chip, thin film	1
R306	Resistor	18k		Various	1206	Chip, thin film	1
R309, R316	Resistor	20k		Various	1206	Chip, thin film	2
R311, R326, R327	Resistor	30k		Various	1206	Chip, thin film	3
R413	Resistor	39k		Various	1206	Chip, thin film	1
R302, R317	Resistor	100k		Various	1206	Chip, thin film	2
R412	Resistor	150k		Various	1206	Chip, thin film	1
R103, R104, R105, R106, R333	Resistor	1M		Various	1206	Chip, thin film	5
R314	Resistor	4M7		Various	1206	Chip, thin film	1
RV301	Trimpot	20k	3362F-1-203LF	Bournes		Top adjust	1

C302, C309	Capacitor	100pF		Various	1206	Ceramic	2
C310, C401	Capacitor	1nF		Various	1206	Ceramic	2
C304	Capacitor	1n5		Various	1206	NPO/COG	1
C303	Capacitor	3n3		Various	1206	NPO/COG	1
C308, C406	Capacitor	10nF		Various	1206	Ceramic	2
C306	Capacitor	47nF		Various	1206	NPO/COG	1
C301, C305, C307	Capacitor	100nF		Various	1206	NPO/COG	3
C101, C102, C103, C104, C105, C106, C107, C108, C109, C111, C112, C113, C114, C116, C201, C202, C203, C311, C312, C313, C314, C315, C316, C317, C318, C319, C320, C321, C404, C503	Capacitor	100nF		Various	1206	Ceramic	30
C502	Capacitor	470nF		Various	1206	Ceramic	1
C110, C115, C206, C207, C208	Capacitor	1uF / 16V		Various	1206	Tantalum	5
C402, C403, C405, C508, C509	Capacitor	10uF / 16V	EEE-1CA100SR	Panasonic	SMT	Electrolytic	5
C506, C507	Capacitor	47uF / 16V	EEE-1CA470WR	Panasonic	SMT	Electrolytic	2
C407, C504, C505	Capacitor	470uF / 16V	EEE-1CA471UP	Panasonic	SMT	Electrolytic	3
C501	Capacitor	4700uF / 25V		Various	Through hole. Pitch = 7.5mm. Diameter = 16mm.	Electrolytic	1

U103	I.C.		74HCT04	Various	SOIC	Hex inverter	1
U305	I.C.		74HC30	Various	SOIC	8-input NAND gate	1
U306	I.C.		74HC32	Various	SOIC	Quad 2-input OR	1
U304	I.C.		74HC132	Various	SOIC	Quad 2-input NAND S/T	1
U308	I.C.		74HC175	Various	SOIC	Quad D-type flip-flop	1
U109, U307	I.C.		74HC244	Various	SOIC - WIDE	Octal buffer / driver	2
U102, U105	I.C.		74HCT32	Various	SOIC	Quad 2-input OR	2
U107, U108	I.C.		74HCT244	Various	SOIC - WIDE	Octal buffer / driver	2
U106	I.C.		74HCT245	Various	SOIC - WIDE	Octal bus transceiver	1
U202	I.C.		AS6C1008	Alliance Memory	SOP	128k x 8 SRAM	1
U201, U203	I.C.		AT27C256	Atmel	DIP	32k x 8 OTP ROM	2
U104	I.C.		Z84C0006PEG	Zilog	DIP	Microprocessor	1
U401	I.C.		LM386	Texas Instruments	DIP	Audio power amplifier	1
REG501	I.C.		LM7805MP	Texas Instruments	SOT-223	+5V fixed regulator	1
U301, U302	I.C.		TL074	Various	SOIC	Quad JFET-input opamp	2
U101	I.C.		TLC556	Texas Instruments	SOIC	Dual timer	1
U303	I.C.		TS391SN2T1G	On Semiconductor	TSOP-5	Comparator	1

B501, B502	Bridge rectifier		W04	Various	TH		2
D305	TVS diode		SMAJ30CA	Various	DO214AC		1
D303	Single diode		BAT54	Various	SOT-23		1
D301, D302, D501, D502	Dual diode		BAT54S	Various	SOT-23		4
D101, D304	Diode		LL4148	Various	SOD-80		2
D201	Zener diode		BZX84C3V6	Various	SOT-23		1
D401	Zener diode		BZX84C6V8	Various	SOT-23		1
Q201, Q301, Q303, Q304, Q305, Q401, Q402, Q403, Q404	Transistor		BC850C	Various	SOT-23	NPN	9
Q302, Q501	Transistor		BC860C	Various	SOT-23	PNP	2
Q405, Q503	Transistor		FZT1049A	Diodes Inc.	SOT-223	NPN	2
Q504	Transistor		FZT1149A	Diodes Inc.	SOT-223	PNP	1
Q502	Transistor		MJ2955	Various	TO-3	PNP	1

P501	Connector		640445-4	TE Connectivity	TH	3.96mm-pitch header, 4-way	1

P101, P205, P302, P303, P502, P503	Connector		0022272021	Molex	TH	KK-254 vertical header, 2-way	6
P201, P202, P304, P401, P402, P403	Connector		0022272041	Molex	TH	KK-254 vertical header, 4-way	6
P301	Connector		0022272061	Molex	TH	KK-254 vertical header, 6-way	1
P204	Connector			Various	TH	34-way IDC boxed header	1
P102, P203	Connector			Various	TH	50-way IDC boxed header	2

F501	Fuse holder		64600001003	Littlefuse	TH	5x20mm PCB mount	1
	Cover		64800001009	Littlefuse		For fuse holder F501	1
RLY301	Relay		DS1E-S-DC5V	Panasonic	TH	5VDC coil, SPDT	1
	Heatsink		506007B00000G	Aavid		Board Level Heatsink for TO-3	1

Q502 mounted with M3 hardware. Machined-pin sockets recommended for all DIP integrated circuits.

Parts for Video Generator PCB

Designator	Component type	Value	Part #	Manufacturer	Package	Description	Quantity
R6	Resistor	68R		Various	1206	Chip, thin film	1
R4	Resistor	220R		Various	1206	Chip, thin film	1
R3	Resistor	1k		Various	1206	Chip, thin film	1
R1	Resistor	2k2		Various	1206	Chip, thin film	1
R14, R15	Resistor	3k3		Various	1206	Chip, thin film	2
R2	Resistor	5k1		Various	1206	Chip, thin film	1
R5, R7, R8, R9, R10, R11, R12	Resistor	10k		Various	1206	Chip, thin film	7
R13	Resistor	47k		Various	1206	Chip, thin film	1
RV4	Trimpot	5k	3362F-1-502LF	Bournes		Cermet, top adjust	1
RV2	Trimpot	50k	3362F-1-503LF	Bournes		Cermet, top adjust	1
RV3	Potentiometer	5k linear	296XD	CTS		PCB mount, right angle	1
RV1	Potentiometer	100k linear	296XD	CTS		PCB mount, right angle	1

C32, C33	Capacitor	1nF		Various	1206	NPO/COG	2
C31	Capacitor	10nF		Various	1206	NPO/COG	1
C30	Capacitor	100nF		Various	1206	NPO/COG	1
C1, C2, C3, C5, C5, C6, C7, C8, C9, C10, C11, C12, C13, C14, C15, C16, C17, C18, C19, C20, C21, C22, C23, C24, C25	Capacitor	100nF		Various	1206	Ceramic	25
C26, C27	Capacitor	1uF / 16V		Various	1206	Tantalum	2
C28	Capacitor	47uF / 16V	EEE-1CA470WR	Panasonic	SMT	Electrolytic	1
C29	Capacitor	470uF / 16V	EEE-1CA471UP	Panasonic	SMT	Electrolytic	1

U5	I.C.		74HC00	Various	SOIC	Quad NAND gate	1
U17	I.C.		74HCT00	Various	SOIC	Quad NAND gate	1
U6, U8, U11	I.C.		74HC08	Various	SOIC	Quad AND gate	3
U2, U22, U23	I.C.		74HC74	Various	SOIC	Dual D flip-flop	3
U3, U12, U13, U14	I.C.		74HC157	Various	SOIC	Data selector / multiplexer	4
U4	I.C.		74HC161	Various	SOIC	4-bit synchronous counter	1
U7, U9, U10	I.C.		74HC393	Various	SOIC	Dual counter	3
U24, U25	I.C.		74HC4538	Various	SOIC	Dual monostable	2
U16	I.C.		74HCT02	Various	SOIC	Quad 2-input NOR	1
U21	I.C.		74HCT165	Various	SOIC	Shift register	1
U19	I.C.		74HC244	Various	SOIC - WIDE	Octal buffer / driver	1
U18	I.C.		74HCT244	Various	SOIC - WIDE	Octal buffer / driver	1
U15	I.C.		AS7C256	Alliance Memory	SOJ	32k x 8 SRAM	1
U20	I.C.		AT27C256	Atmel	DIP	32k x 8 OTP ROM	1
U1	I.C.		MX045-3C-16M0000	CTS Elec. Comp.	DIP	16 MHz oscillator, 14 pin DIP	1

Q1	Transistor		BC860C	Various	SOT-23	PNP	1

P1, P2	Connector		0022272021	Molex	TH	KK-254 vertical header, 2-way	2
P3	Connector			Various	TH	50-way IDC boxed header	1
JP1_A	Jumper		SPC02SYAN	Sullins		0.1" pitch	1
JP1_B	3 pin header			Various	TH	Single row 0.1" pitch male pins	1

Machined-pin socket recommended for U20.

Parts for Keyboard Interface PCB

Designator	Component type	Value	Part #	Manufacturer	Package	Description	Quantity
R4	Resistor	1k		Various	1206	Chip, thin film	1
R1, R2, R3	Resistor	4k7		Various	1206	Chip, thin film	3
R5, R6, R7	Resistor	10k		Various	1206	Chip, thin film	3

C3, C4, C5, C6, C7, C8	Capacitor	100nF		Various	1206	Ceramic	6
C1	Capacitor	47uF / 16V	EEE-1CA470WR	Panasonic	SMT	Electrolytic	1
C2	Capacitor	10uF / 25V	T491C106K025AT	Kemet	2312	Tantalum ESR 1.5 ohms	1

U1	I.C.		PIC16F876A	Microchip	SOIC - WIDE	Microcontroller	1
U2	I.C.		EPM7064AETC44	Intel	TQFP-44	CPLD	1
U3	I.C.		MXO45HS-3C	CTS Elec. Comp.	DIP	10 MHz oscillator, 8 pin DIP	1
REG1	I.C.		TC1262-3V3	Microchip	SOT-223	3.3V regulator	1

P4	Connector			Various	TH	34-way IDC boxed header	1
P1	Connector		0022272041	Molex	TH	KK-254 vertical header, 4-way	1
P2	4 pin header			Various	TH	Single row 0.1" pitch male pins	1
P3	8 pin header			Various	TH	Double row 0.1" pitch male pins	1