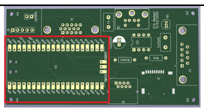
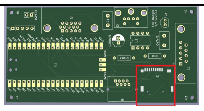
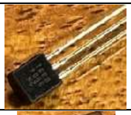
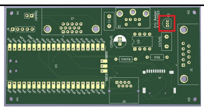
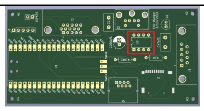
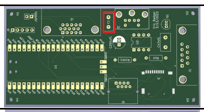
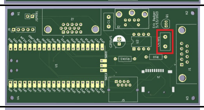
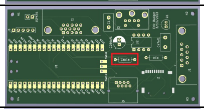
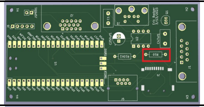
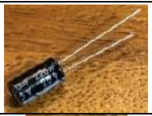
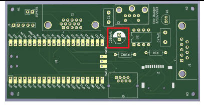
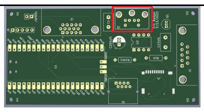
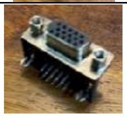
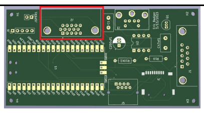
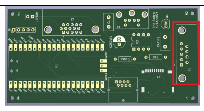
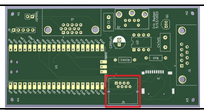
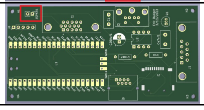
Badger6502Pico

Contents

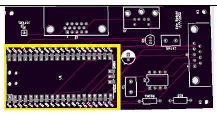
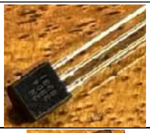
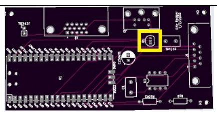
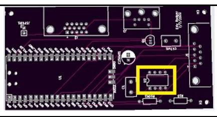
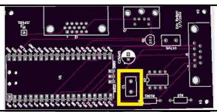
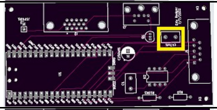
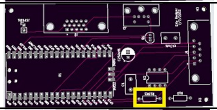
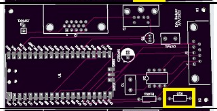
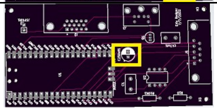
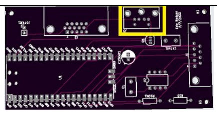
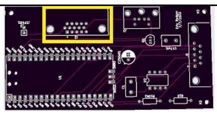
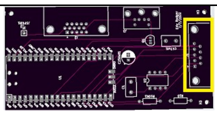
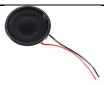
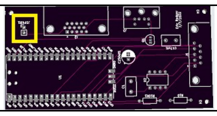
Parts and assembly - V2 ebadger6502pico

Amazon shopping list : https://www.amazon.com/hz/wishlist/ls/5XCSSBHC06ON?ref_=wl_share

Image	Part description	Notes	
	Raspberry PI Pico	Use pin headers to line up holes. Solder a couple empty pins to hold in place, remove pin headers and solder remaining. SMD solder the castellations.	
	Surface mount micro SD card.	SMD Solder to J4	
	2N3904 transistor	Solder to Q1 – direction matters, see silkscreen	
	LM386	Solder to U2, match notch with notch in silk screen. Direction matters.	
	106 capacitor	Solder into C1, direction does not matter	
	473 capacitor	Solder to C47nf1, direction does not matter	
	10K Ohm resistor	Solder to R10K1, direction does not matter	
	10 Ohm resistor	Solder to R10, direction does not matter	
	220uf capacitor	Solder to C250uf1, direction matters. See white stripe. Short lead (-) in white side	
	PS/2 keyboard port	Pops in, line up pins, press firmly but don't force. Solder to J3	
	15 pin VGA port	Press firmly when all pins lined up, do not force, insert with care. Solder to J2	
	9 pin Joystick port	Press firmly when all pins are lined up, do not force, insert with care. Solder to J1	
	RJ45 port	Solder to J5	
	Speaker	Strip wire ends and solder to JSPKR1, direction does not matter	

Parts and assembly - V1 ebadger6502pico

1) Amazon shopping list : https://www.amazon.com/hz/wishlist/ls/5XCSSBHC06ON?ref_=wl_share

Image	Part description	Notes	
	Raspberry PI Pico	Use pin headers to line up holes and solder a couple empty pins to hold in place, remove pin headers and solder remaining.	
	2N3904 transistor	Solder to Q1 – direction matters, see silkscreen	
	LM386	Solder to U2, match notch with notch in silk screen. Direction matters.	
	106 capacitor	Solder into C1, direction does not matter	
	473 capacitor	Solder to C47nf1, direction does not matter	
	10K Ohm resistor	Solder to R10K1, direction does not matter	
	10 Ohm resistor	Solder to R10, direction does not matter	
	220uf capacitor	Solder to C250uf1, direction matters. See white stripe. Short lead (-) in white side	
	PS/2 keyboard port	Pops in, line up pins, press firmly but don't force. Solder to J3	
	15 pin VGA port	Press firmly when all pins lined up, don't force. Solder to J2	
	9 pin Joystick port	Press firmly when all pins are lined up, don't force. Solder to J1	
	Speaker	Strip wire ends and solder to JSPKR1, direction does not matter	

Note

Thank you for purchasing the Badger6502Pico and for your support. If in kit form, see the parts list for assembly instructions. Your feedback is appreciated, and you can always send mail to me at `eric.badger@gmail.com`. The Pico has been pre-flashed with the software.

Thanks,

Eric Badger

Required hardware:

2) VGA, PS/2 keyboard and Micro USB cable are required and not included.
3) Apply power by plugging in the Micro USB to a power source and to the box. Micro SD card is required for the drive emulators, for level editing or for loading and saving basic programs.

Software included:

1) An original and custom 6502 emulator
   a. Specification matches the ebadger6502 computer.
   b. An Apple II port of the game LodeRunner to the custom computer
      i. Includes a working level editor with save capability via Micro SD
      ii. Saves high scores onto the onboard flash.
   c. MS Basic (the CBM 2.0 variant) with Save and Load file capability.
   d. WozMon, the Apple I monitor program written by Steve Wozniak
2) An original Atari disk drive emulator
   a. UI to browse the micro SD card to select .ATR images
   b. Supports SD, ED and DD disks.
   c. Supports hard disk .ATR images up to 16MB
3) An original Commodore serial IEC disk drive emulator
   a. Currently alpha quality, YMMV
   b. Only supports reads and raw access.
   c. Does not yet support writing, formatting.

Known limitations:

1) Upon powering on, the device will try to read the Micro SD card. If a Micro SD card is not inserted, this process can take approximately 30 seconds.
2) Commodore drive emulation is not complete and is still a work in progress.
3) Atari drive emulation is better but does not yet support XEX, CAR, CAS files.
4) Both drive emulators only support 1 disk as of now.
5) Text and Graphics mode are controlled at the Pico level. The text mode is implemented like a serial terminal. As of now it's not possible in basic to control graphics or switch into graphics mode. This could change in a software update if there is demand.
6) Swapping Micro SD card requires reboot.
7) No physical reset button – to preserve the life of the device, unplug and plug in using the Type-A side of the USB cable. This reduces stress on the micro USB solder joint on the Pico.

I intend to continue work on the software including improvements for the drive emulators, possibly additional drive emulators and more. I will likely open source the project if there is interest. How great would it be if this could be a community project? If you're interested in contributing, providing feedback or adding to the wish list, please send mail to eric.badger@gmail.com.

Thanks again for your support, Eric Badger

UI considerations:

4) Device will boot directly to Lode Runner
    a. Controls: I, J, K, L control direction with up, left, down, right respectively.
    b. U, O will dig left and right respectively
    c. Ctrl+A will kill your guy if you get into an place where you can't win
    d. Ctrl+R terminates the game.
    e. Ctrl+M shows the high scores
    f. Cheats: Ctrl+N for next level, Ctrl+E for extra guy. Warning: Cheats will prevent saving your high score.
    g. Editor: While the game is in attract mode, Ctrl+E will launch the editor
    h. The "d" key in the level editor will toggle between "master disk" and "user disk". User disk is for custom levels, master disk is for built in levels. Levels can only be saved on the user disk.
    i. "e" in the editor to edit a level.
    j. "p" in the editor to play a level.
    k. While editing a level, I,J,K,M keys control the cursor, and 1-0 keys control the various blocks. Ctrl-Q will prompt to save.
5) Switch to WozMon by pressing Ctrl+Alt+Del from LodeRunner
6) Switch to Atari drive emulator by pressing Ctrl+Alt+2 from WozMon (won't work from LodeRunner)
7) Switch to Commodore drive emulator by pressing Ctrl+Alt+1 from WozMon (won't work from LodeRunner)
8) To launch Basic, type C100R and press enter from the WozMon "\" prompt.
    a. Syntax for Load/Save is [Load "filename"] or [Save "filename"] (excluding brackets) where filename is the name of the file. The quotes are required. Basic will currently load or save from the root directory of the micro SD card only
9) GPIO:
    a. 0xC050 – 0xC056 correspond to GPIO pins 0, 1, 15, 22, 28, 27, 26 respectively.
    b. When poking these addresses, the data is a bitfield with the following characteristics:
    c. If 0x80 bit is set, direction is configured for output if 0x40 is also set, otherwise is set for input.
    d. If 0x20 bit is set, 0x10 will configure the pin for pull up, if 0x10 is not set, will configure the pin for pulldown.
10) Drive Emulator navigation:
    a. When entering the drive emulator, you can navigate the file list with the arrow keys. Down, Up, Page Down and Page Up can be used to navigate the list. Pressing the left arrow will take you up a directory level. Pressing the right arrow or enter will either enter a directory or will select a file to load.   The bottom of the screen includes status messages and will indicate if a file has been loaded. For the Atari, load a disk before turning on the machine. For the Commodore, once the disk has been loaded, you can use the load command from basic to load a directory or to load a file.

Memory Map and notable addresses

Memory map:

$0000 – $BFFF RAM (48KB)

$C000 – $C0FF Devices

$C100 – $E2FF Basic ROM

$E300 – $FF00 OS

$FF00 – $FFF9 WozMon

$FFFA – $FFFB NMI vector

$FFFC – $FFFD Boot vector

$FFFE – $FFFF IRQ vector

Pokeable addresses:

$C053 - Speaker beep

$C050 – $C056 correspond to Pico GPIO 0, 1, 15, 22, 28, 27, 26

Executable Addresses:

$C100 – Basic

$C106 – Clear screen

$6000 – LodeRunner

OS Addresses:

$0380 - $03FF – Keyboard input buffer (scancodes)

$0300 - $037F – Keyboard state (scancodes)

$D2 – Last key pressed

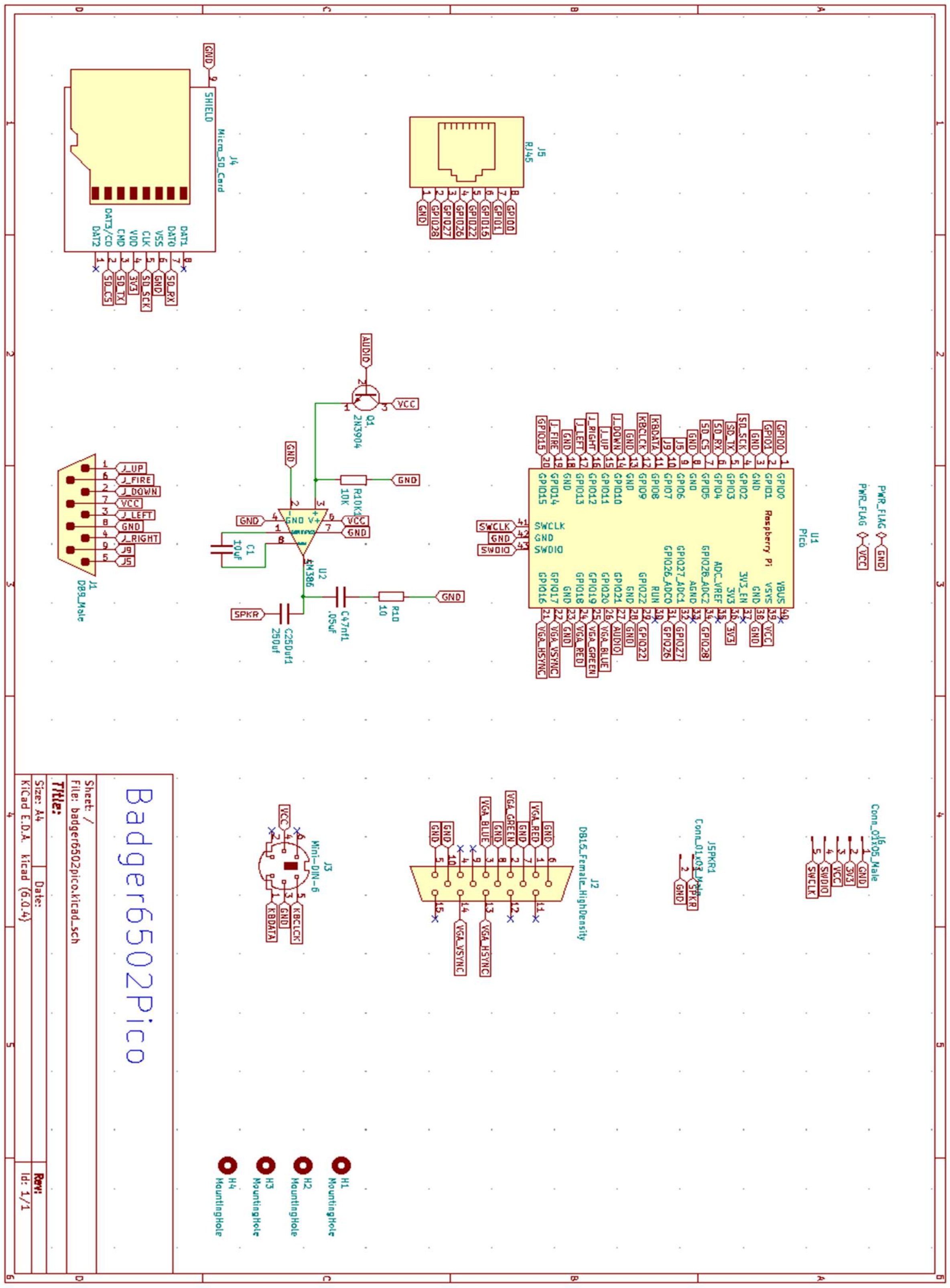
Additional resources

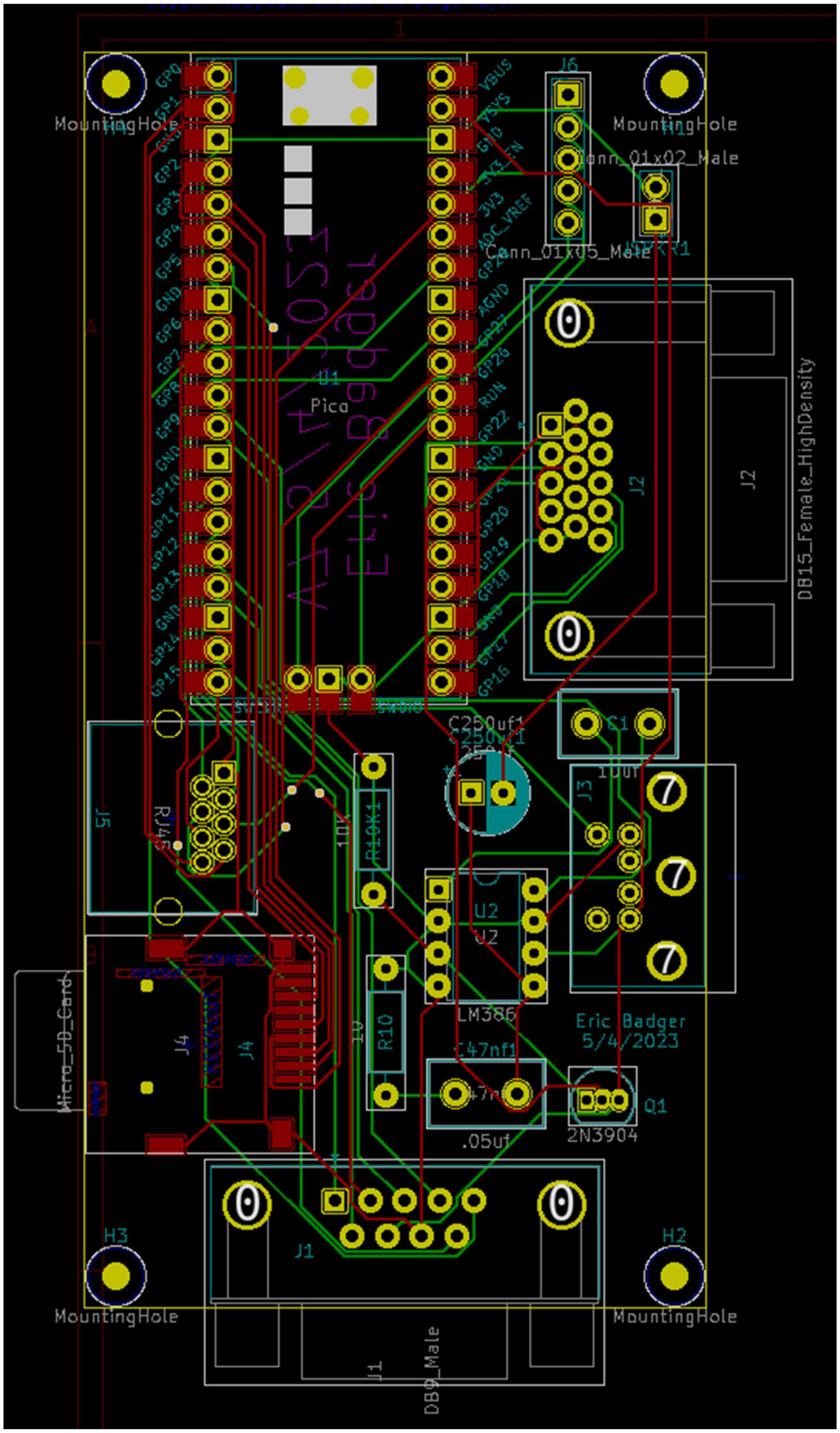
a. https://github.com/ebadger/msbasic
   i. This repository contains the WozMon and MS Basic implementation as well as much of the emulator specific code. As of now this is only good for code inspection as the badger6502pico project is also required and is not yet published.   I will do the work to publish if there is sufficient demand.

b. https://github.com/ebadger/BadgerFrotz
   i. BadgerFrotz is a port of the PC Z-machine emulator Frotz to the Pi Pico.
   ii. This project is designed to run on the ebadger6502pico hardware, and could serve as a useful guide if you'd like to adapt the hardware for some other project
   iii. Pretty useful as it effectively implements a VGA terminal.
   iv. Also fun since it can run most of the old Z-machine games.
   v. Includes Colossal Cave Adventure. Most of the old Infocom games are playable, but you'll need to go find the .z files, convert them into c-data structure to play them. (See binaries.h)

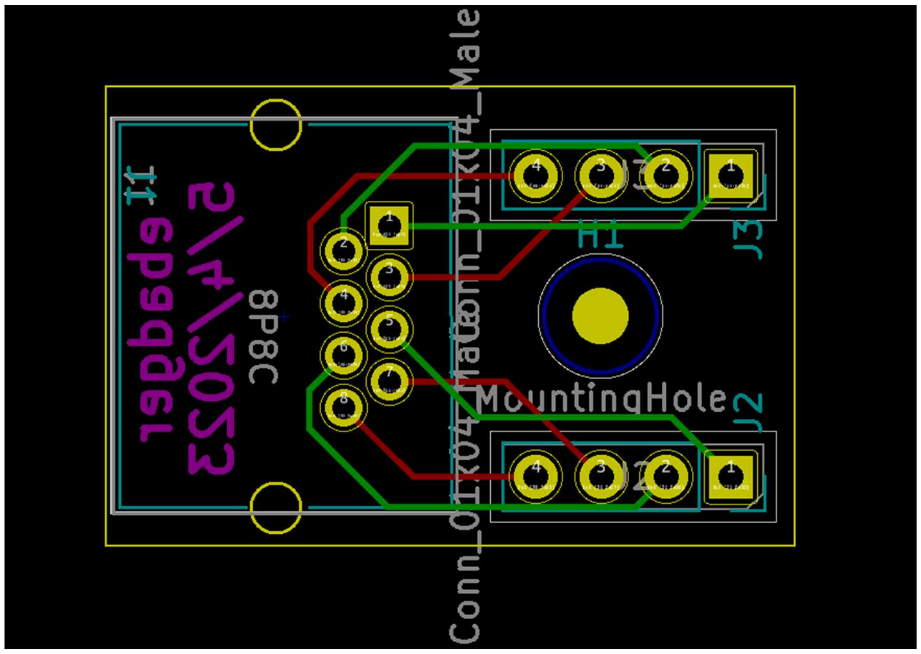
c. http://vintage-basic.net/games.html
   i. An archive of basic games to type in

Hardware diagrams



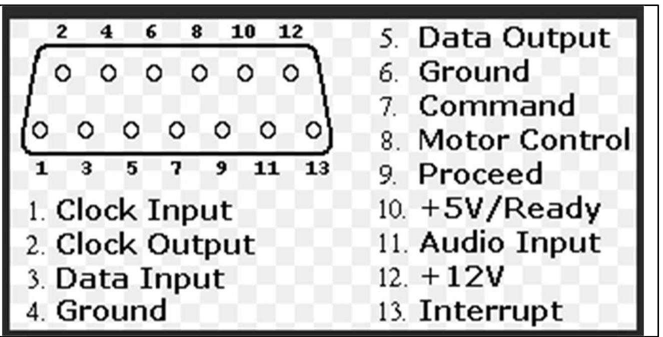
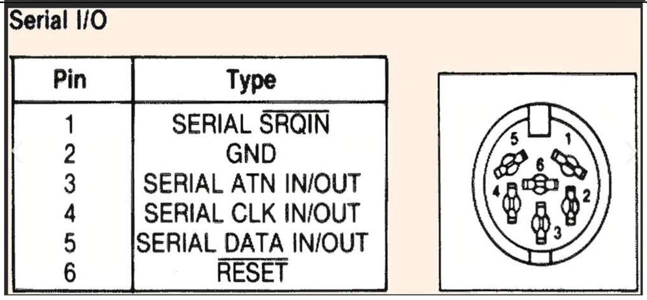


GP0
GP1
GND
GP2
GP3
GP4
GP5
GND
GP6
GP7
GP8
GP9
GND
GP10
GP11
GP12
GP13
GND
GP14
GP15
VBUS
VSYS
GND
3V3_EN
3V3
ADC_VREF
GP28
AGND
GP27
GP26
RUN
GP22
GND
GP21
GP20
GP19
GP18
GND
GP17
GP16
U1
Pico
SWCLK
SWDIO
MountingHole
H1
H2
H3
H4
J6
Conn_01x02_Male
Conn_01x05_Male
J2
DB15_Female_HighDensity
C1
10uf
C250uf1
250uf
J3
J5
RJ45
10K
R10K1
U2
LM386
R10
1V
C47nf1
.05uf
Q1
2N3904
J4
Micro_SD_Card
J1
DB9_Male
Eric Badger
5/4/2023



GPIO RJ-45 adapter pin mapping:

J3	J2
1) Ground 2) GPIO 28 3) GPIO 27 4) GPIO 26	1) GPIO 22 2) GPIO 15 3) GPIO 0 4) GPIO 1

<u>For Atari SIO</u> 22 = Command (7) 15 = Clock Output (2) 0 = Data Input (to CPU) (3) 27 = Ready (10) 26 = Data Output (from CPU) (5) Diagram is wire side	 1. Clock Input 2. Clock Output 3. Data Input 4. Ground 5. Data Output 6. Ground 7. Command 8. Motor Control 9. Proceed 10. +5V/Ready 11. Audio Input 12. +12V 13. Interrupt
<u>For Commodore Serial IEC</u> 28 – Serial SRQIN (1) 27 – Serial ATN (3) 26 – Reset (6) 22 – Serial CLK (4) 15 – Serial Data (5) Diagram is wire side	 Serial I/O Pin / Type 1 SERIAL SRQIN 2 GND 3 SERIAL ATN IN/OUT 4 SERIAL CLK IN/OUT 5 SERIAL DATA IN/OUT 6 RESET

Basic Programs

Buzz

```
5 REM BUZZ SPEAKER
10 FOR I=1 TO 100
20 POKE 49251,1
30 NEXT I
```

Blinky

```
10 REM SET ALL GPIO FOR OUTPUT AND PULL UP
20 FOR I=49232 TO 49239
30 POKE I,240
40 NEXT I
50 DELAY=512
55 REM *** FLIP THE GPIO ON AND OFF ****
60 FOR I=49232 TO 49239
70 POKE I,1
80 GOSUB 1000
90 POKE I,0
100 NEXT I
110 GOTO 60
1000 FOR X=1 TO DELAY : NEXT X
1010 RETURN
```

LastKey

```
10 REM * THE LAST ASCII KEY VALUE RECEIVED
20 PRINT PEEK(210)
30 GOTO 20
```