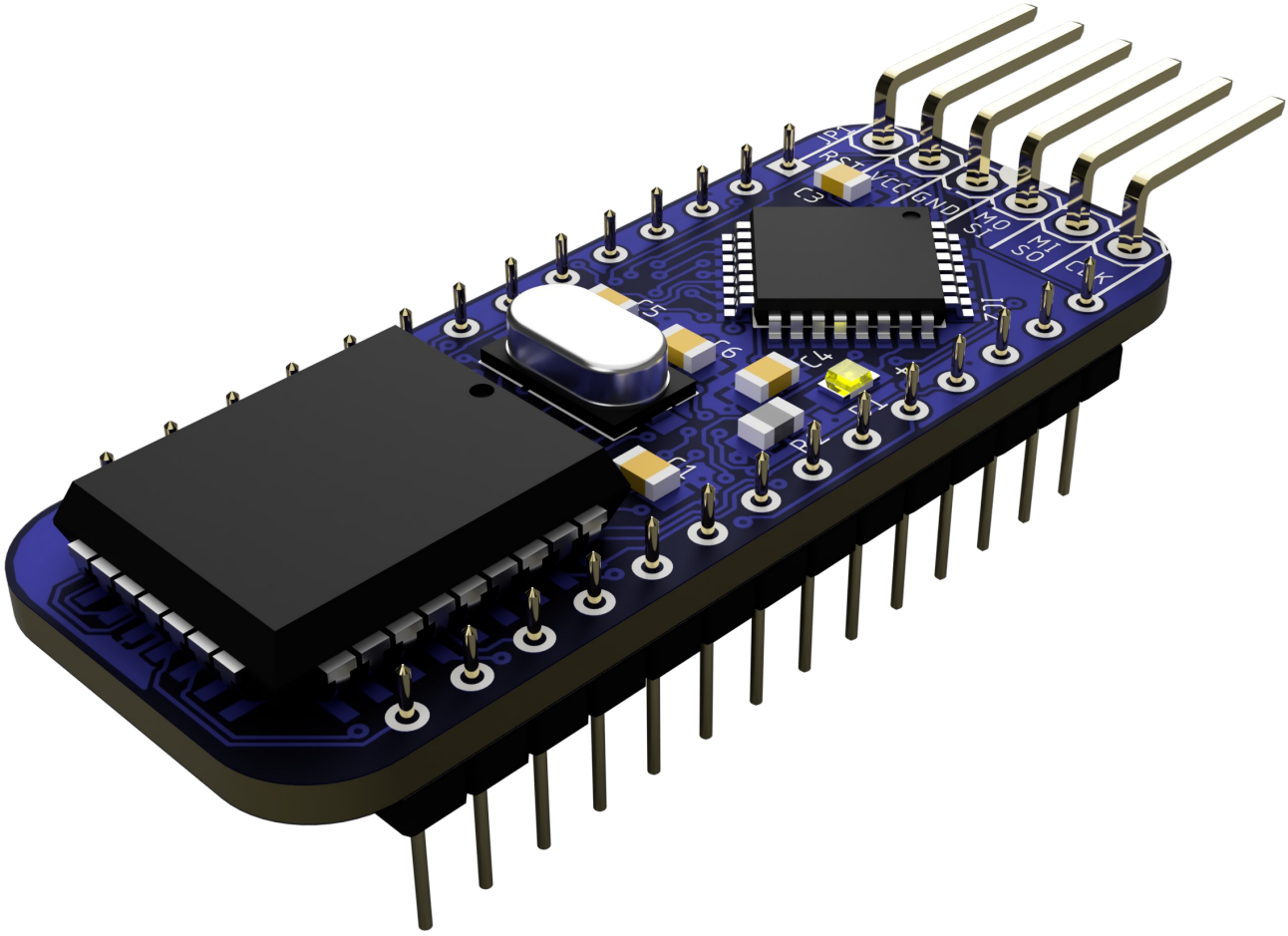
Retroninja Switchless Multi-Kernal for C64 longboard



User's Guide

Table of Contents

Functional description....4
Installing the Kernal switch....5
PCB 250407....5
PCB 250425....6
PCB 250466....6
Using the Kernal switch....7
Upgrading the firmware....7
Programming the flash chip (ROM)....7

Functional description

t power on, the MCU in the Kernal switch holds the reset line active while it reads address 0 from its internal EPROM to find out which Kernal it should select, sets the A13-A18 address pins on the flash chip accordingly and releases the reset line to make sure the computer is booted using the correct Kernal.

If at any time, restore is pressed and held for a few seconds the MCU will detect this and switch to Kernal 0 which is a menu Kernal. For this application I created a custom Mini-Kernal for the C64 that displays a list of Kernals to choose from.

When the user selects a Kernal image from the menu, the Mini-Kernal writes a predefined switching command to RAM along with a byte that indicates the selected image number. It does this over and over. Example of the command: RNROM64#1 where 1 is a byte with value 0x01 that tells the switch that we want to switch to kernal image 1.
The Kernal switch now captures the command from the data bus, writes the selected kernal number to the MCU EPROM for future use, then it switches the address pins A13-A18 accordingly. It then resets the computer using the new Kernal by pulling the reset line. The computer will continue to start up using this new Kernal until a new choice is made.

The flash is populated with the Mini-Kernal followed by up to 10*8 KB Kernals. It would in theory be possible to add up to 63 kernals if both the Mini-Kernal and the MCU firmware was modified in some way to accommodate that many Kernal images.
Here's an example layout that fills out an SST39SF010A:

Mini-Kernal	CBM Kernal	JiffyDOS US	JiffyDOS SE	JiffyDOS DK	JaffyDOS US	JaffyDOS SE	SpeedDOS	SpeedDOS	DolphinDOS	DolphinDOS	Mini-Kernal	Mini-Kernal	Mini-Kernal	Mini-Kernal	Mini-Kernal
8kB	8kB	8kB	8kB	8kB	8kB	8kB	8kB	8kB	8kB	8kB	8kB	8kB	8kB	8kB	8kB

If the same layout would be used with a larger flash chip it should be duplicated to fill out the flash.

This C64 Kernal switch only suits the C64 longboards which have a 24pin ROM and has basic in a separate ROM. All C64 motherboards are longboards except PCB 250469.
The C64 shortboard (PCB 250469) have 28-pin 27128 type Kernal ROMs and I have another Multi-ROM version that is compatible with those at:
https://github.com/RetroNynjah/Switchless-Multi-ROM-for-27128-27256

Installing the Kernal switch

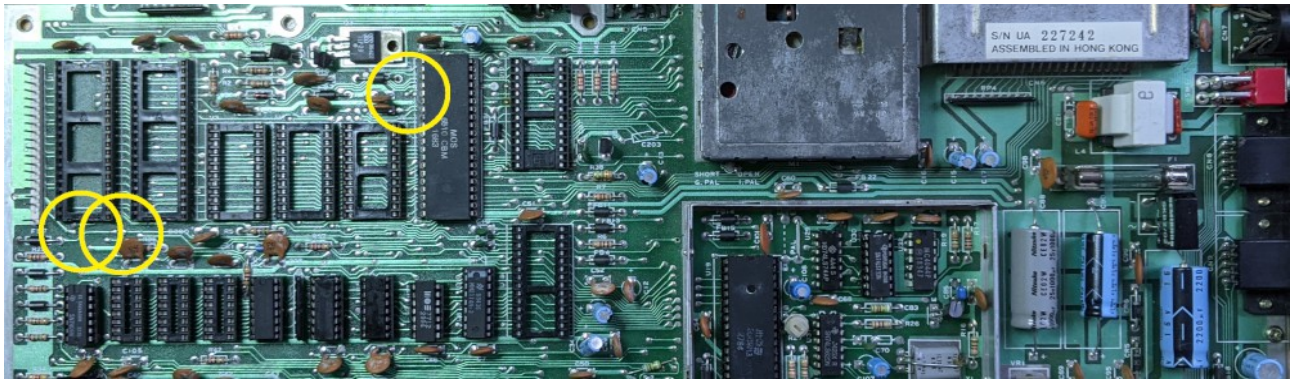
The below pins on the switch must be connected to the right signals on the motherboard.

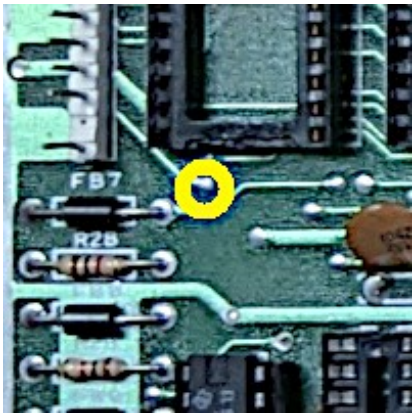
Switch pin	Computer signal	Example of locations
CLK	R/W	6510 pin 38, 6526 pin 22
MISO	RESTORE	KBD connector pin 3
MOSI	RESET	6510 pin 40, 6526 pin 34

The connections can be made on the chips using test clips but it's recommended to solder pin headers to vias on the motherboard and connect to the pin headers using Dupont wires or solder one end of Dupont cables directly to the vias for a permanent installation.
If you want to solder pin headers to the vias you can try inserting them from the top side without taking the board out. Try heating the pins with the soldering iron while pushing them into the via using pliers but to get the best possible connection of the pins they should be soldered from the bottom of the motherboard.

The below pictures show examples of suitable vias for different motherboards.

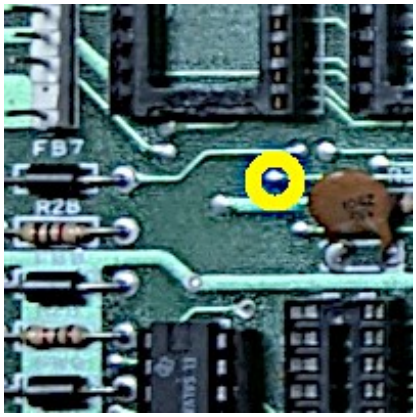
PCB 250407





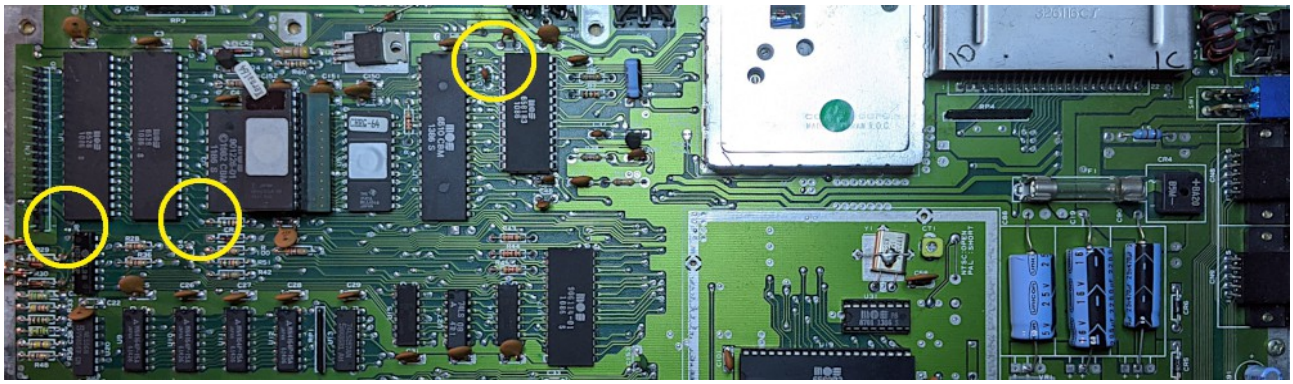
RESTORE

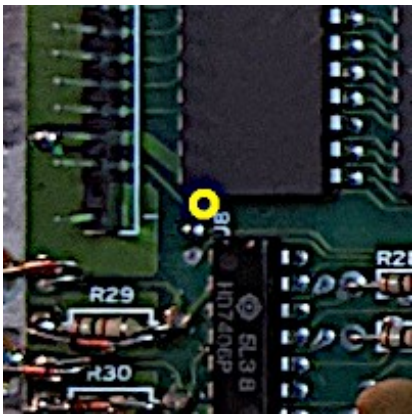
R/W



RESET

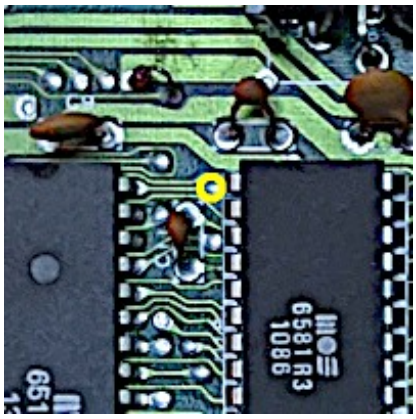
PCB 250425





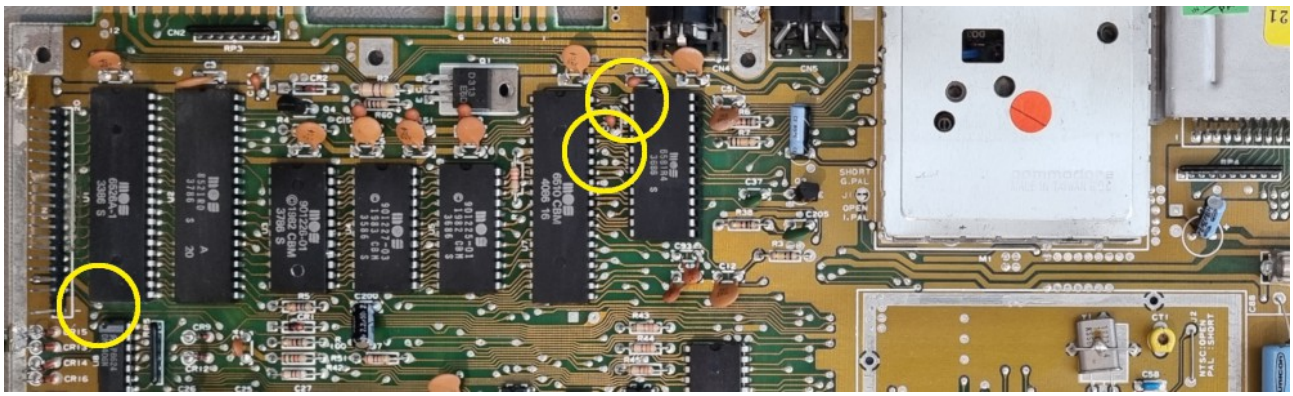
RESTORE

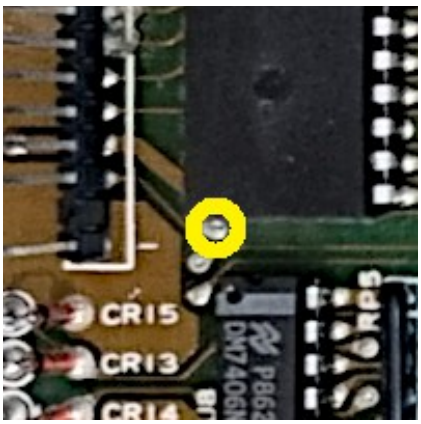
R/W



RESET

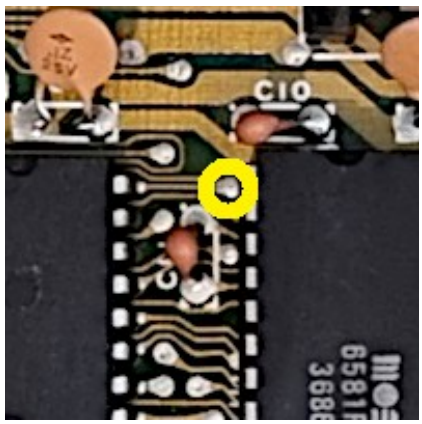
PCB 250466





RESTORE

R/W



RESET

Using the Kernal switch

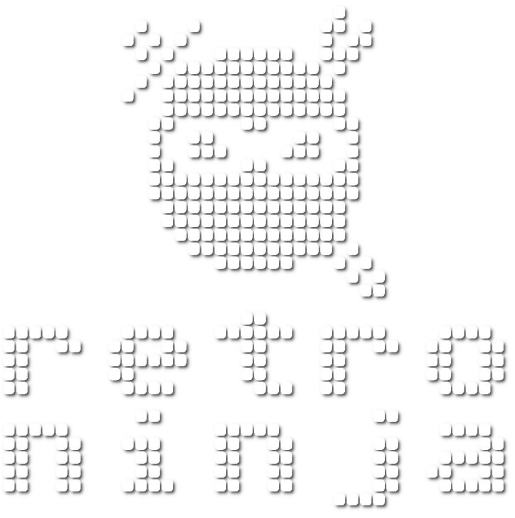
Hold the restore key for about three seconds until the kernal menu is activated. Once in menu you can use cursor up/down to move to the Kernal you want to switch to and press return. The switch resets the computer with the new Kernal and saves your choice for next power-on.

Upgrading the firmware

The firmware in the microcontroller can be upgraded using the 6-pin ISP header. The source code is written for Arduino and programming the firmware can be done directly from the Arduino IDE using an ISP programmer.
If your device is using an Atmega328 it can be programmed as an Arduino UNO. The other variants require custom board definitions. I have used the MiniCore from MCUdude (https://github.com/MCUdude/MiniCore) with default board settings.

If you can't or don't want to use Arduino I have some precompiled hex files for the Kernal switch and drive ROM switch along with fuse configurations and syntax examples for how to program them using avrdude. These can be found under the Applications folder in the GitHub repository at:
https://github.com/RetroNynjah/Switchless-Multi-ROM-for-2364

Programming the flash chip (ROM)

The flash chip needs to be programmed before soldering it. If you need to reprogram the flash later it must first be desoldered and then resoldered again after reprogramming.

The flash chip should contain the Mini-Kernal image followed by all switchable kernal images in correct order. All images needs to have a size of exactly 8KB.