The 3.3V regulator built into the microcontroller is documented as only supplying a max of 300mA to external circuitry.

Rather than use that, we use our own regulator in the cartridge board. It can power the microcontroller and all 3V logic chips. Then we power the microcontroller through the 3V3_OUT and GND pins.

One jumper connects 3V3_EN connects to GND to disable the board's regulator. Another jumper connects VBUS to the +5V supply to power the 5V chips.

While programming the microcontroller in-place via USB, the jumper should be on the VBUS pins. While operating in the Sega, the jumper should be on the 3V3_EN pins.

000 => register & !A3 & write = REG_WE
010 => register & A3 & write = SYNC_SET
110 => register & A3 & !write = SYNC_OE

001 => !SYNC_OE & SYNC_WE & !A1 = CMD_SET
010 => SYNC_OE & !SYNC_WE & !A1 = CMD_OE
101 => !SYNC_OE & SYNC_WE & A1 = ERR_CLR
110 => SYNC_OE & !SYNC_WE & A1 = ERR_OE

Write 16-bits to even addresses, but only low byte is accepted.

Or write 8-bits to odd addresses.

Register addresses
CMD: 0xA13000
ARG: 0xA13002
unused2: 0xA13004
unused3: 0xA13006

Command token address
0xA13008

Error token address
0xA1300A

While powering the microcontroller via USB without the Sega (during programming and testing), pull-up resistors on these lines keep inputs from floating. They can be set low for testing with jumper wires in the cartridge pin header.

5V signals from Sega

3V signals from microcontroller

5V input tolerated

Template for RPiPicoW board header alignment

File: register-file-internal.kicad_sch

File: sync-token-internal.kicad_sch

File: ethernet-internal.kicad_sch

File: kinetoscope-header-internal.kicad_sch

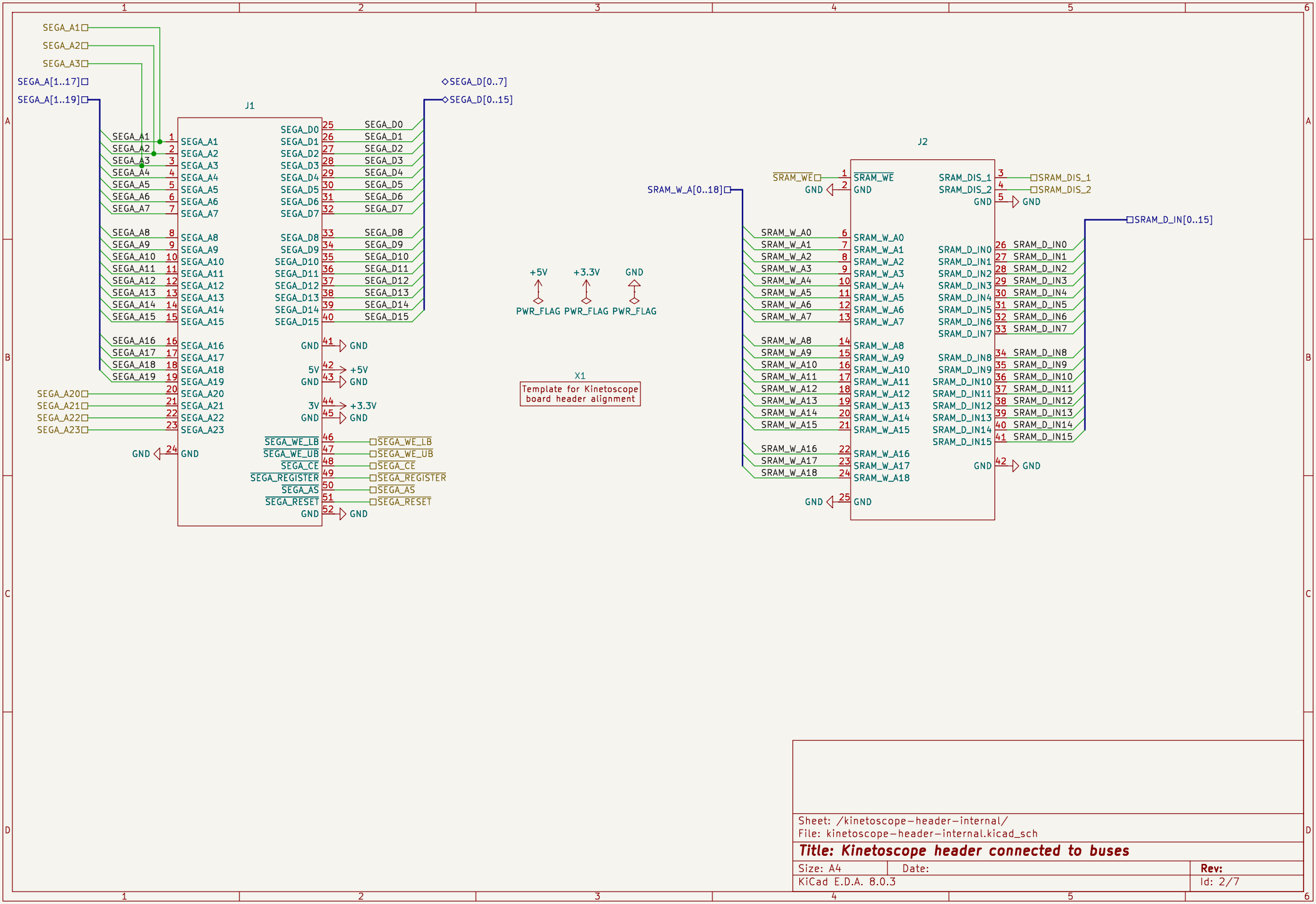
File: address-counter-internal.kicad_sch

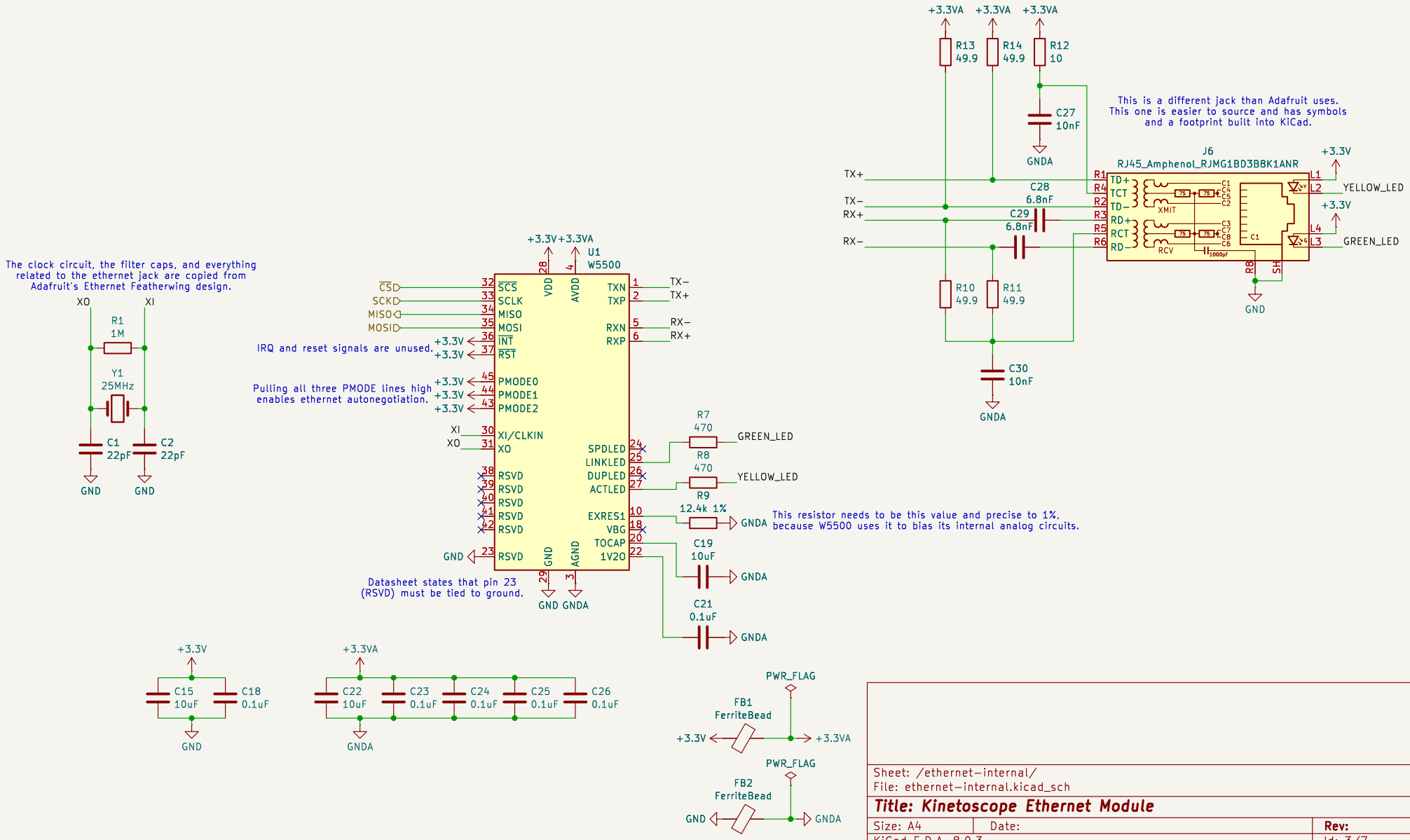
File: data-register-internal.kicad_sch

Sheet: /
File: microcontroller.kicad_sch

Title: Kinetoscope Microcontroller and Register Board

Size: A4	Date:	Rev:
KiCad E.D.A. 8.0.3		Id: 1/7

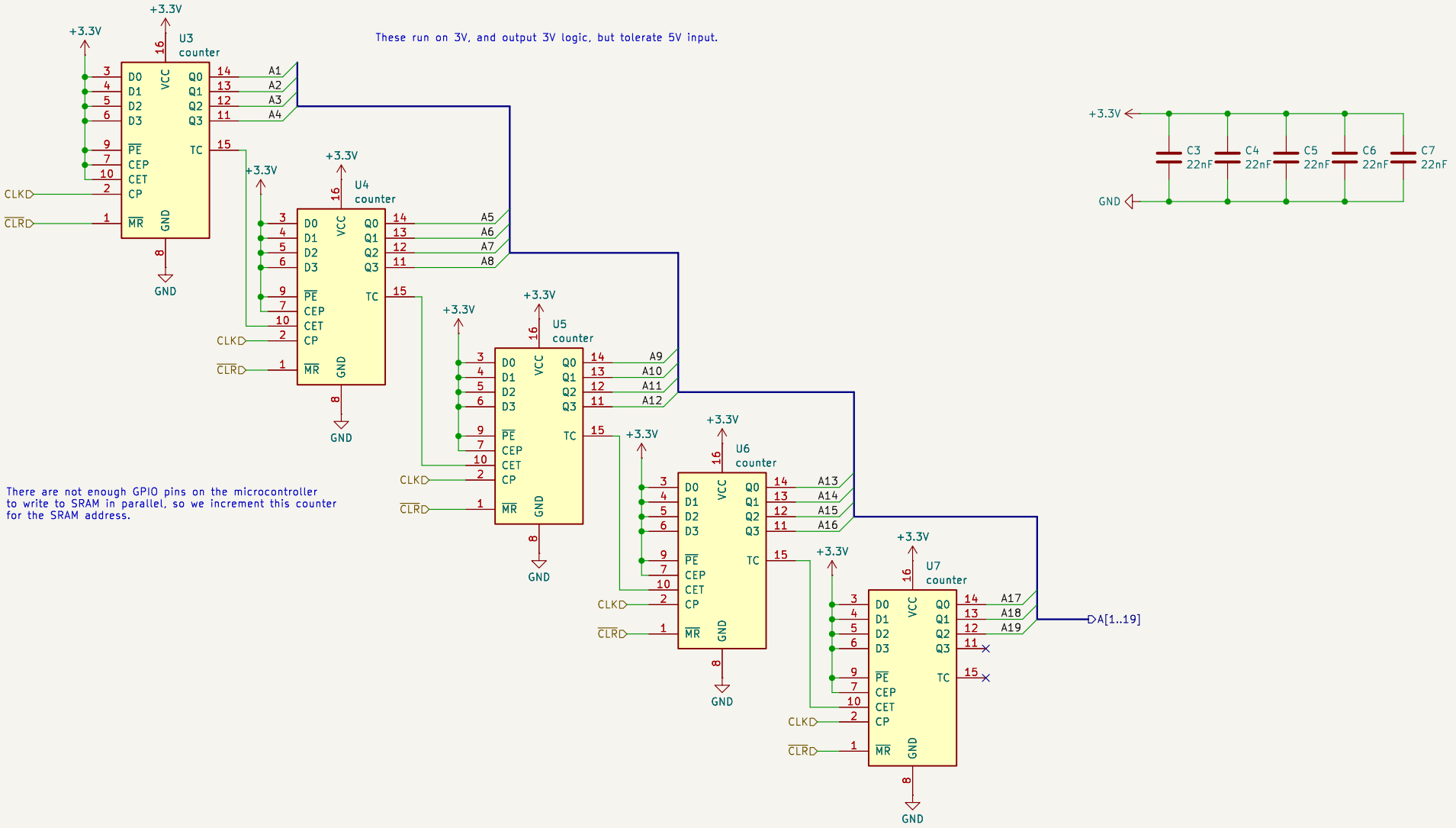

SEGA_A1
SEGA_A2
SEGA_A3
SEGA_A[1..17]
SEGA_A[1..19]
J1
SEGA_D[0..7]
SEGA_D[0..15]
SEGA_A20
SEGA_A21
SEGA_A22
SEGA_A23
GND
5V
+5V
3V
+3.3V
SEGA_WE_LB
SEGA_WE_UB
SEGA_CE
SEGA_REGISTER
SEGA_AS
SEGA_RESET
PWR_FLAG
X1
Template for Kinetoscope board header alignment
J2
SRAM_W_A[0..18]
SRAM_WE
SRAM_DIS_1
SRAM_DIS_2
SRAM_D_IN[0..15]
Sheet: /kinetoscope-header-internal/
File: kinetoscope-header-internal.kicad_sch
Title: Kinetoscope header connected to buses
Size: A4
Date:
Rev:
KiCad E.D.A. 8.0.3
Id: 2/7



Sheet: /ethernet-internal/
File: ethernet-internal.kicad_sch

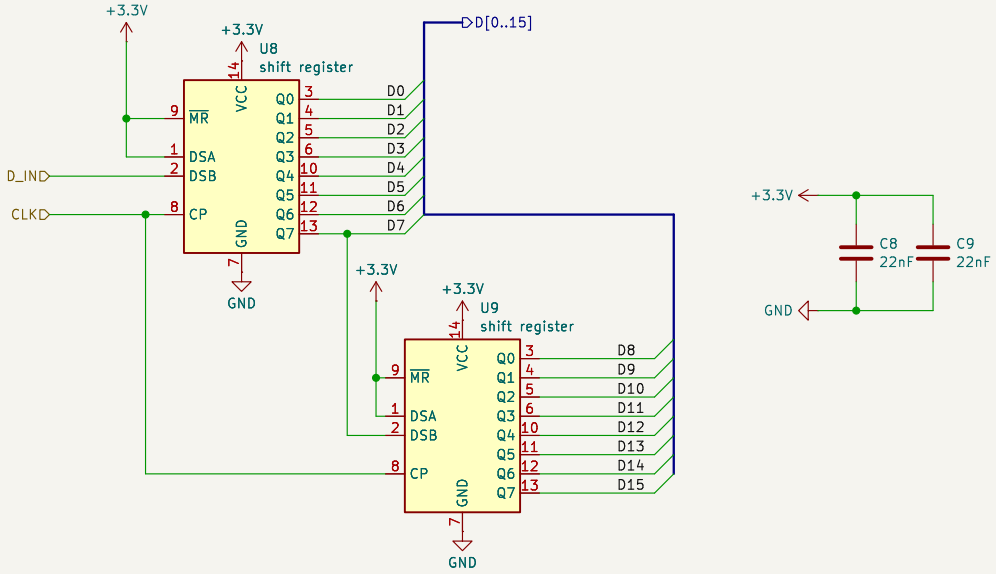
Title: Kinetoscope Ethernet Module

Size: A4	Date:	Rev:
KiCad E.D.A. 8.0.3		Id: 3/7



Sheet: /address-counter-internal/	
File: address-counter-internal.kicad_sch	
Title: Kinetoscope SRAM Address Counter	
Size: A4	Date:
KiCad E.D.A. 8.0.3	Id: 4/7

There are not enough GPIO pins on the microcontroller
to write to SRAM in parallel, so we fill this register serially.



Sheet: /data-register-internal/
File: data-register-internal.kicad_sch

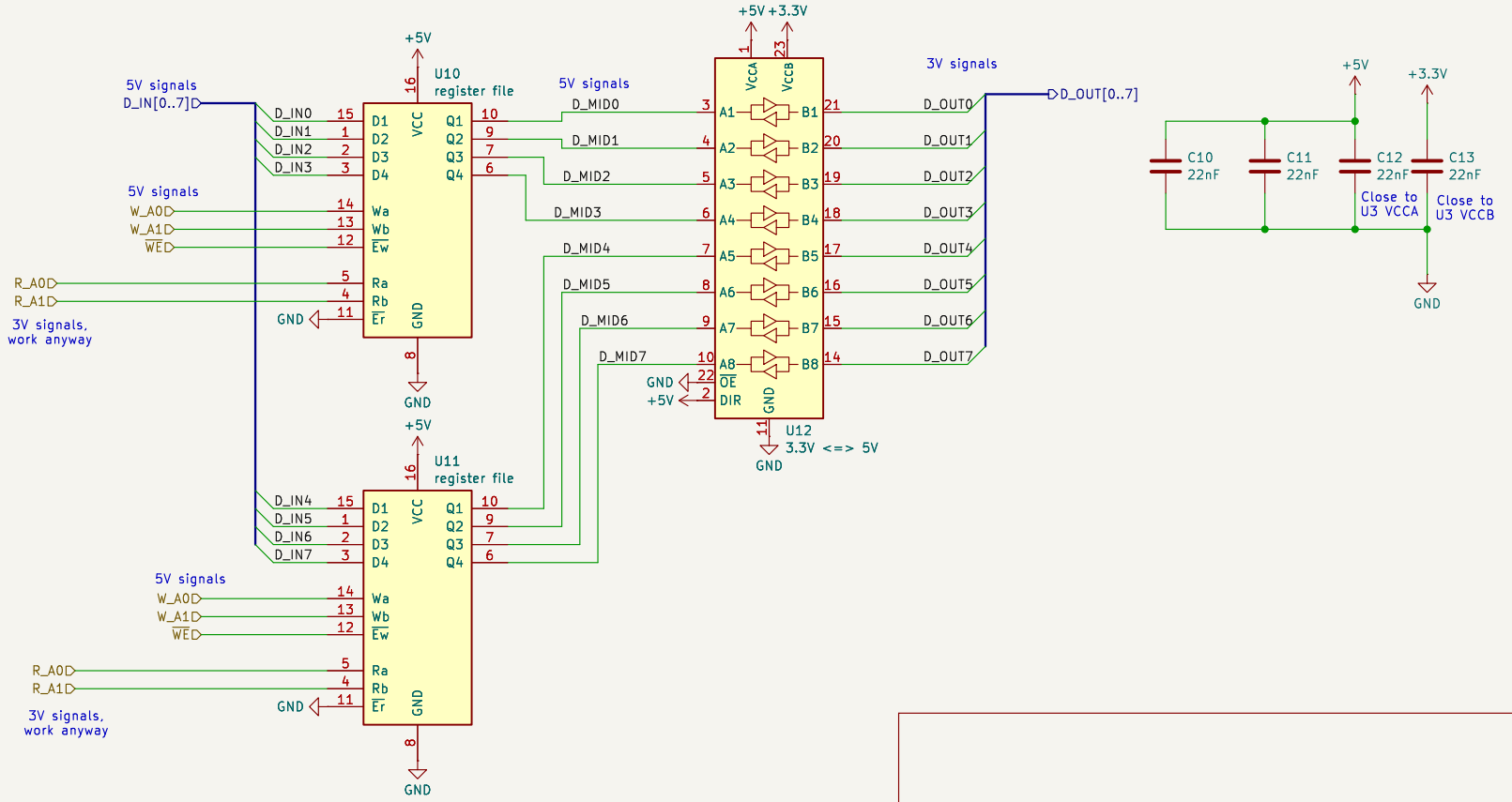
Title: Kinetoscope SRAM Data Register

Size: A4	Date:	Rev:
KiCad E.D.A. 8.0.3		Id: 5/7

This is a set of four 8-bit registers that can be independently read and written by the two controllers (one reads, one writes).

The registers are assumed to written by a device with a shared data bus (Sega Genesis) and read by a device with dedicated pins (microcontroller). The read output is always enabled.

Everything runs on +5V, but must be compatible with 3.3V inputs. If using a slightly different part number, please check input voltages in the data sheet. The data outputs are 3.3V.



Sheet: /register-file-internal/
File: register-file-internal.kicad_sch

Title: Kinetoscope Register File connected to buses

Size: A4	Date:	Rev:
KiCad E.D.A. 8.0.3		Id: 6/7

This is a synchronization token between the M68k and the microcontroller.

The token will be cleared on boot/reset.

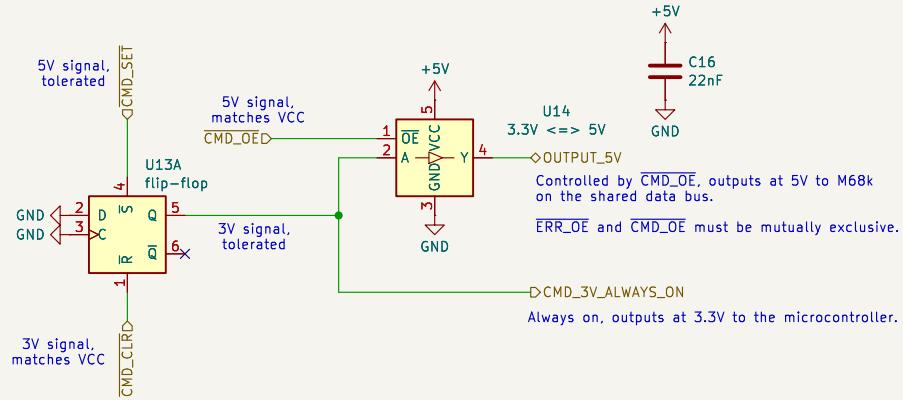
The M68k will be able to set the token (CMD_SET), indicating to the microcontroller that a command is ready to be executed.

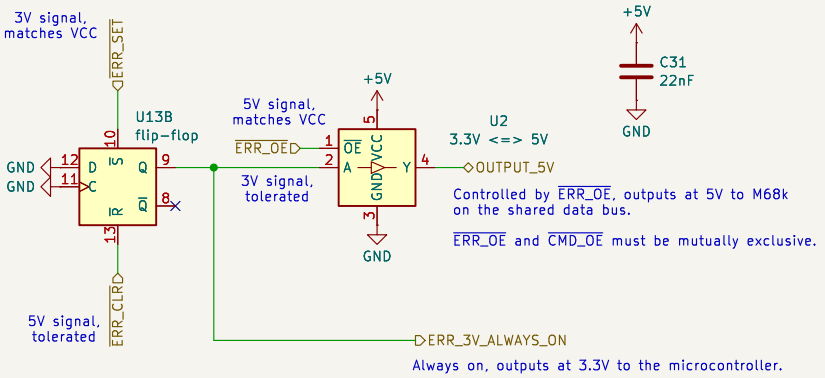
The microcontroller will be able to clear the token (CMD_CLR), indicating to the M68k that the command has been executed.

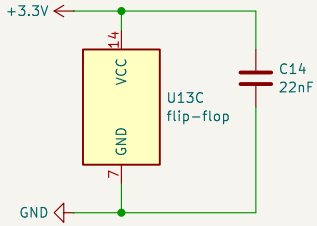
The M68k and microcontroller both will be able to read the current state of the token. The M68k reads a 5V version of the signal, and the microcontroller reads a 3.3V version of the signal.

Command codes and arguments are passed through a separate register file.

The microcontroller can also flag an error ERR_SET to the M68k, who can clear it with ERR_CLR after noticing.







Sheet: /sync-token-internal/		
File: sync-token-internal.kicad_sch		
Title: Kinetoscope Sync Token		
Size: A4	Date:	**Rev:**
KiCad E.D.A. 8.0.3		Id: 7/7