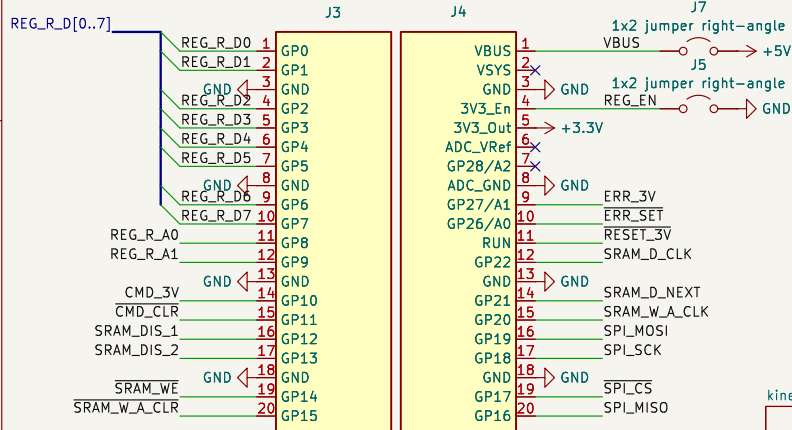The 3.3V regulator built into the microcontroller is documented as only supplying a max of 300mA to external circuitry.

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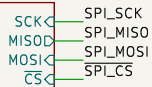
One jumper connects 3V3_EN connects to GND to disable the board's regulator. Another jumper connects VBUS to the +5V supply to power the 5V chips.

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.
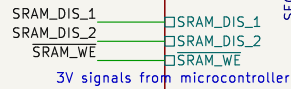
J3

REG_R_D0	1	GP0
REG_R_D1	2	GP1
GND	3	GND
REG_R_D2	4	GP2
REG_R_D3	5	GP3
REG_R_D4	6	GP4
REG_R_D5	7	GP5
GND	8	GND
REG_R_D6	9	GP6
REG_R_D7	10	GP7
REG_R_A0	11	GP8
REG_R_A1	12	GP9
GND	13	GND
CMD_3V	14	GP10
CMD_CLR	15	GP11
SRAM_DIS_1	16	GP12
SRAM_DIS_2	17	GP13
GND	18	GND
SRAM_WE	19	GP14
SRAM_W_A_CLR	20	GP15

REG_R_D[0..7]

J4

VBUS	1	VBUS
	2	VSYS
GND	3	GND
GND	4	3V3_En
+3.3V	5	3V3_Out
	6	ADC_VRef
	7	GP28/A2
GND	8	ADC_GND
ERR_3V	9	GP27/A1
ERR_SET	10	GP26/A0
RESET_3V	11	RUN
SRAM_D_CLK	12	GP22
SRAM_D_NEXT	13	GND
SRAM_W_A_CLK	14	GP21
SPI_MOSI	15	GP20
SPI_SCK	16	GP19
	17	GP18
GND	18	GND
SPI_CS	19	GP17
SPI_MISO	20	GP16

J7 1x2 jumper right-angle
VBUS — +5V

J5 1x2 jumper right-angle
GND — GND
REG_EN — GND

X2
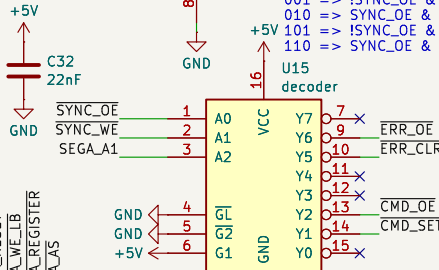Template for RPiPicoW board header alignment

ethernet-internal
SCK — SPI_SCK
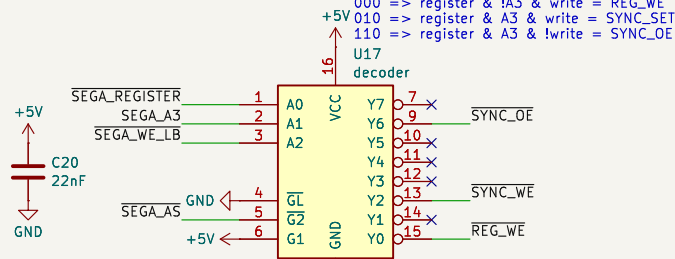MISO — SPI_MISO
MOSI — SPI_MOSI
CS — SPI_CS
File: ethernet-internal.kicad_sch

kinetoscope-header-internal

SEGA_D[0..7]

SRAM_DIS_1 — SRAM_DIS_1
SRAM_DIS_2 — SRAM_DIS_2
SRAM_WE — SRAM_WE

3V signals from microcontroller

SEGA_D[0..7], SEGA_A1, SEGA_A2, SEGA_A3, SEGA_RESET, SEGA_WE_LB, SEGA_REGISTER, SEGA_AS
5V signals from Sega

SEGA_OE, SEGA_CE, SEGA_WE_UB, SEGA_D[0..15], SEGA_A[1..17], SEGA_A[1..19], SEGA_A20, SEGA_A21, SEGA_A22, SEGA_A23
SRAM_W_A[0..18], SRAM_D_IN[0..15]

File: kinetoscope-header-internal.kicad_sch

address-counter-internal
CLR — SRAM_W_A_CLR
CLK — SRAM_W_A_CLK
A[1..19] — SRAM_W_A[0..18]
File: address-counter-internal.kicad_sch
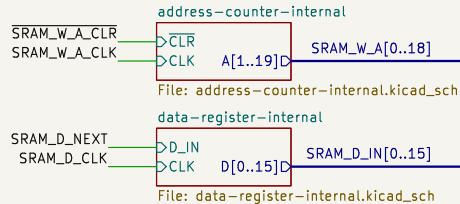
data-register-internal
D_IN — SRAM_D_NEXT
CLK — SRAM_D_CLK
D[0..15] — SRAM_D_IN[0..15]
File: data-register-internal.kicad_sch

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

U17 decoder

+5V

SEGA_REGISTER	1	A0
SEGA_A3	2	A1
SEGA_WE_LB	3	A2
SEGA_AS	4	GL
GND	5	G2
+5V	6	G1

Y7 7 —
Y6 9 — SYNC_OE
Y5 10 —
Y4 11 —
Y3 12 —
Y2 13 — SYNC_WE
Y1 14 —
Y0 15 — REG_WE

C20 22nF
GND

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

U15 decoder

+5V

SYNC_OE	1	A0
SYNC_WE	2	A1
SEGA_A1	3	A2
GND	4	GL
GND	5	G2
+5V	6	G1

Y7 7 —
Y6 9 — ERR_OE
Y5 10 — ERR_CLR
Y4 11 —
Y3 12 —
Y2 13 — CMD_OE
Y1 14 — CMD_SET
Y0 15 —

C32 22nF
GND

register-file-internal
REG_R_D[0..7] — D_OUT[0..7]
REG_R_A0 — R_A0
REG_R_A1 — R_A1
D_IN[0..7] — SEGA_D[0..7]
W_A0 — SEGA_A1
W_A1 — SEGA_A2
WE — REG_WE
File: register-file-internal.kicad_sch

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

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

Or write 8-bits to odd addresses.

Command token address 0xA13008

Error token address 0xA1300A

sync-token-internal
5V signal — CMD_SET
3V signal — CMD_CLR
5V signal — CMD_OE
3V signal — ERR_SET
5V signal — ERR_CLR
5V signal — ERR_OE
CMD_3V_ALWAYS_ON — CMD_3V
OUTPUT_5V — SEGA_D0
ERR_3V_ALWAYS_ON — ERR_3V
File: sync-token-internal.kicad_sch
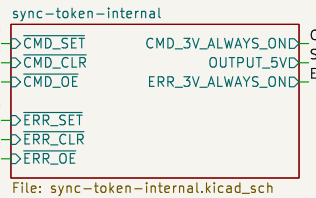
While powering the microcontroller via USB without the Sega (during programming and testing), pull-up resistors on these lines keep inputs from floating. They can be set low for testing with jumper wires in the cartridge pin header.

+5V R15 10k SEGA_RESET
+5V R16 10k SEGA_A1
+5V R17 10k SEGA_A2
+5V R2 10k SEGA_A3
+5V R18 10k SEGA_WE_LB
+5V R19 10k SEGA_REGISTER
+5V R20 10k SEGA_AS

U16 3.3V <=> 5V
+3.3V
SEGA_RESET — A
OE — 1
GND — 2
Y — RESET_3V
5V input tolerated
GND
+3.3V C17 22nF GND

J1

SEGA_A1	1	SEGA_A1	
SEGA_A2	2	SEGA_A2	
SEGA_A3	3	SEGA_A3	
SEGA_A4	4	SEGA_A4	
SEGA_A5	5	SEGA_A5	
SEGA_A6	6	SEGA_A6	
SEGA_A7	7	SEGA_A7	
SEGA_A8	8	SEGA_A8	
SEGA_A9	9	SEGA_A9	
SEGA_A10	10	SEGA_A10	
SEGA_A11	11	SEGA_A11	
SEGA_A12	12	SEGA_A12	
SEGA_A13	13	SEGA_A13	
SEGA_A14	14	SEGA_A14	
SEGA_A15	15	SEGA_A15	
SEGA_A16	16	SEGA_A16	
SEGA_A17	17	SEGA_A17	
SEGA_A18	18	SEGA_A18	
SEGA_A19	19	SEGA_A19	
SEGA_A20	20	SEGA_A20	
SEGA_A21	21	SEGA_A21	
SEGA_A22	22	SEGA_A22	
SEGA_A23	23	SEGA_A23	

SEGA_A1
SEGA_A2
SEGA_A3
SEGA_A[1..17]
SEGA_A[1..19]

SEGA_D[0..7]
SEGA_D[0..15]

SEGA_D0 25 SEGA_D0
SEGA_D1 26 SEGA_D1
SEGA_D2 27 SEGA_D2
SEGA_D3 28 SEGA_D3
SEGA_D4 29 SEGA_D4
SEGA_D5 30 SEGA_D5
SEGA_D6 31 SEGA_D6
SEGA_D7 32 SEGA_D7
SEGA_D8 33 SEGA_D8
SEGA_D9 34 SEGA_D9
SEGA_D10 35 SEGA_D10
SEGA_D11 36 SEGA_D11
SEGA_D12 37 SEGA_D12
SEGA_D13 38 SEGA_D13
SEGA_D14 39 SEGA_D14
SEGA_D15 40 SEGA_D15

GND 41 GND
5V 42 +5V
GND 43 GND
3V 44 +3.3V

SEGA_OE 45 SEGA_OE
SEGA_WE_LB 46 SEGA_WE_LB
SEGA_WE_UB 47 SEGA_WE_UB
SEGA_CE 48 SEGA_CE
SEGA_REGISTER 49 SEGA_REGISTER
SEGA_AS 50 SEGA_AS
SEGA_RESET 51 SEGA_RESET
GND 52 GND

GND 24 GND

X1

+5V +3.3V GND

PWR_FLAG PWR_FLAG PWR_FLAG

Template for Kinetoscope
board header alignment

J2

SRAM_WE 1 SRAM_WE
GND 2 GND

SRAM_DIS_1 3 SRAM_DIS_1
SRAM_DIS_2 4 SRAM_DIS_2
GND 5 GND

SRAM_W_A[0..18]

SRAM_D_IN[0..15]

SRAM_W_A0 6 SRAM_W_A0
SRAM_W_A1 7 SRAM_W_A1
SRAM_W_A2 8 SRAM_W_A2
SRAM_W_A3 9 SRAM_W_A3
SRAM_W_A4 10 SRAM_W_A4
SRAM_W_A5 11 SRAM_W_A5
SRAM_W_A6 12 SRAM_W_A6
SRAM_W_A7 13 SRAM_W_A7
SRAM_W_A8 14 SRAM_W_A8
SRAM_W_A9 15 SRAM_W_A9
SRAM_W_A10 16 SRAM_W_A10
SRAM_W_A11 17 SRAM_W_A11
SRAM_W_A12 18 SRAM_W_A12
SRAM_W_A13 19 SRAM_W_A13
SRAM_W_A14 20 SRAM_W_A14
SRAM_W_A15 21 SRAM_W_A15
SRAM_W_A16 22 SRAM_W_A16
SRAM_W_A17 23 SRAM_W_A17
SRAM_W_A18 24 SRAM_W_A18

SRAM_D_IN0 26 SRAM_D_IN0
SRAM_D_IN1 27 SRAM_D_IN1
SRAM_D_IN2 28 SRAM_D_IN2
SRAM_D_IN3 29 SRAM_D_IN3
SRAM_D_IN4 30 SRAM_D_IN4
SRAM_D_IN5 31 SRAM_D_IN5
SRAM_D_IN6 32 SRAM_D_IN6
SRAM_D_IN7 33 SRAM_D_IN7
SRAM_D_IN8 34 SRAM_D_IN8
SRAM_D_IN9 35 SRAM_D_IN9
SRAM_D_IN10 36 SRAM_D_IN10
SRAM_D_IN11 37 SRAM_D_IN11
SRAM_D_IN12 38 SRAM_D_IN12
SRAM_D_IN13 39 SRAM_D_IN13
SRAM_D_IN14 40 SRAM_D_IN14
SRAM_D_IN15 41 SRAM_D_IN15

GND 42 GND

GND 25 GND

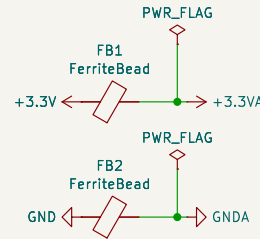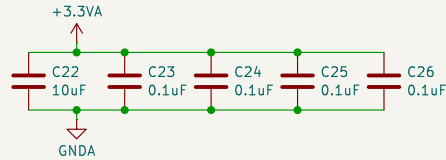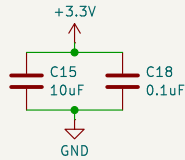**Annotations / text labels on schematic:**

The clock circuit, the filter caps, and everything related to the ethernet jack are copied from Adafruit's Ethernet Featherwing design.
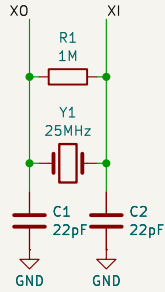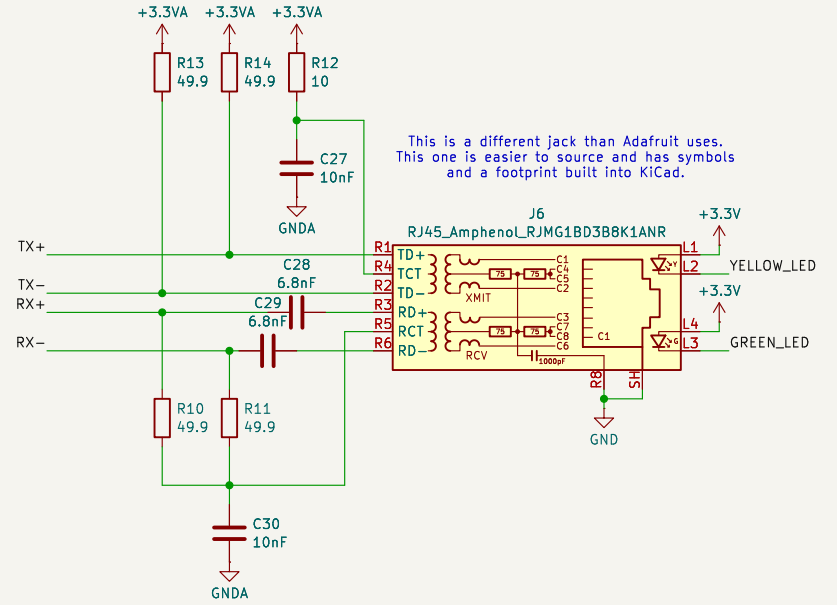
IRQ and reset signals are unused.

Pulling all three PMODE lines high enables ethernet autonegotiation.

Datasheet states that pin 23 (RSVD) must be tied to ground.

This is a different jack than Adafruit uses. This one is easier to source and has symbols and a footprint built into KiCad.

This resistor needs to be this value and precise to 1%, because W5500 uses it to bias its internal analog circuits.

Components:

- R1 1M
- Y1 25MHz
- C1 22pF
- C2 22pF
- U1 W5500
- R7 470 — GREEN_LED
- R8 470 — YELLOW_LED
- R9 — 12.4k 1%
- C19 10uF
- C21 0.1uF
- R13 49.9
- R14 49.9
- R12 10
- C27 10nF
- C28 6.8nF
- C29 6.8nF
- R10 49.9
- R11 49.9
- C30 10nF
- J6 RJ45_Amphenol_RJMG1BD3B8K1ANR
- C15 10uF
- C18 0.1uF
- C22 10uF
- C23 0.1uF
- C24 0.1uF
- C25 0.1uF
- C26 0.1uF
- FB1 FerriteBead
- FB2 FerriteBead

Signal labels: CSD, SCKD, MISO, MOSID, TX−, TXP, RXN, RXP, TX+, TX−, RX+, RX−, YELLOW_LED, GREEN_LED, +3.3V, +3.3VA, GND, GNDA, PWR_FLAG

These run on 3V, and output 3V logic, but tolerate 5V input.

+3.3V

U3
counter

3	D0	Q0	14	A1
4	D1	Q1	13	A2
5	D2	Q2	12	A3
6	D3	Q3	11	A4
9	PE̅		
7	CEP	TC	15
10	CET		
2	CP		
1	MR̅		

VCC 16

GND 8

CLK
CLR̅

+3.3V

U4
counter

| 3 | D0 | Q0 | 14 | A5
| 4 | D1 | Q1 | 13 | A6
| 5 | D2 | Q2 | 12 | A7
| 6 | D3 | Q3 | 11 | A8
9	PE̅		
7	CEP	TC	15
10	CET		
2	CP		
1	MR̅		

VCC 16

GND 8

GND

CLK
CLR̅

+3.3V

U5
counter

| 3 | D0 | Q0 | 14 | A9
| 4 | D1 | Q1 | 13 | A10
| 5 | D2 | Q2 | 12 | A11
| 6 | D3 | Q3 | 11 | A12
9	PE̅		
7	CEP	TC	15
10	CET		
2	CP		
1	MR̅		

VCC 16

GND 8

GND

CLK
CLR̅

+3.3V

U6
counter

| 3 | D0 | Q0 | 14 | A13
| 4 | D1 | Q1 | 13 | A14
| 5 | D2 | Q2 | 12 | A15
| 6 | D3 | Q3 | 11 | A16
9	PE̅		
7	CEP	TC	15
10	CET		
2	CP		
1	MR̅		

VCC 16

GND 8

GND

CLK
CLR̅

+3.3V

U7
counter

| 3 | D0 | Q0 | 14 | A17
| 4 | D1 | Q1 | 13 | A18
| 5 | D2 | Q2 | 12 | A19
6	D3	Q3	11
9	PE̅		
7	CEP	TC	15
10	CET		
2	CP		
1	MR̅		

VCC 16

GND 8

GND

CLK
CLR̅

A[1..19]

There are not enough GPIO pins on the microcontroller to write to SRAM in parallel, so we increment this counter for the SRAM address.

+3.3V

| C3 | C4 | C5 | C6 | C7 |
| 22nF | 22nF | 22nF | 22nF | 22nF |

GND

Sheet: /address-counter-internal/
File: address-counter-internal.kicad_sch

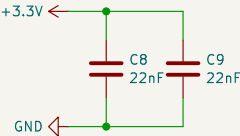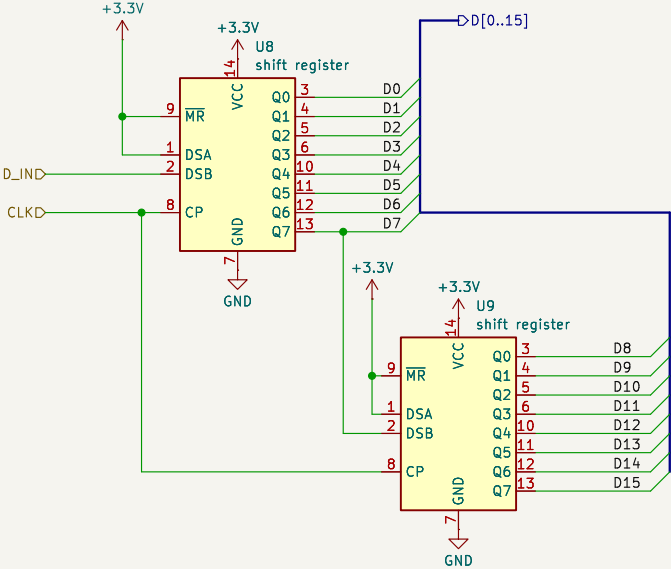**Title: Kinetoscope SRAM Address Counter**

Size: A4   Date:

KiCad E.D.A. 8.0.3

Rev:

Id: 4/7
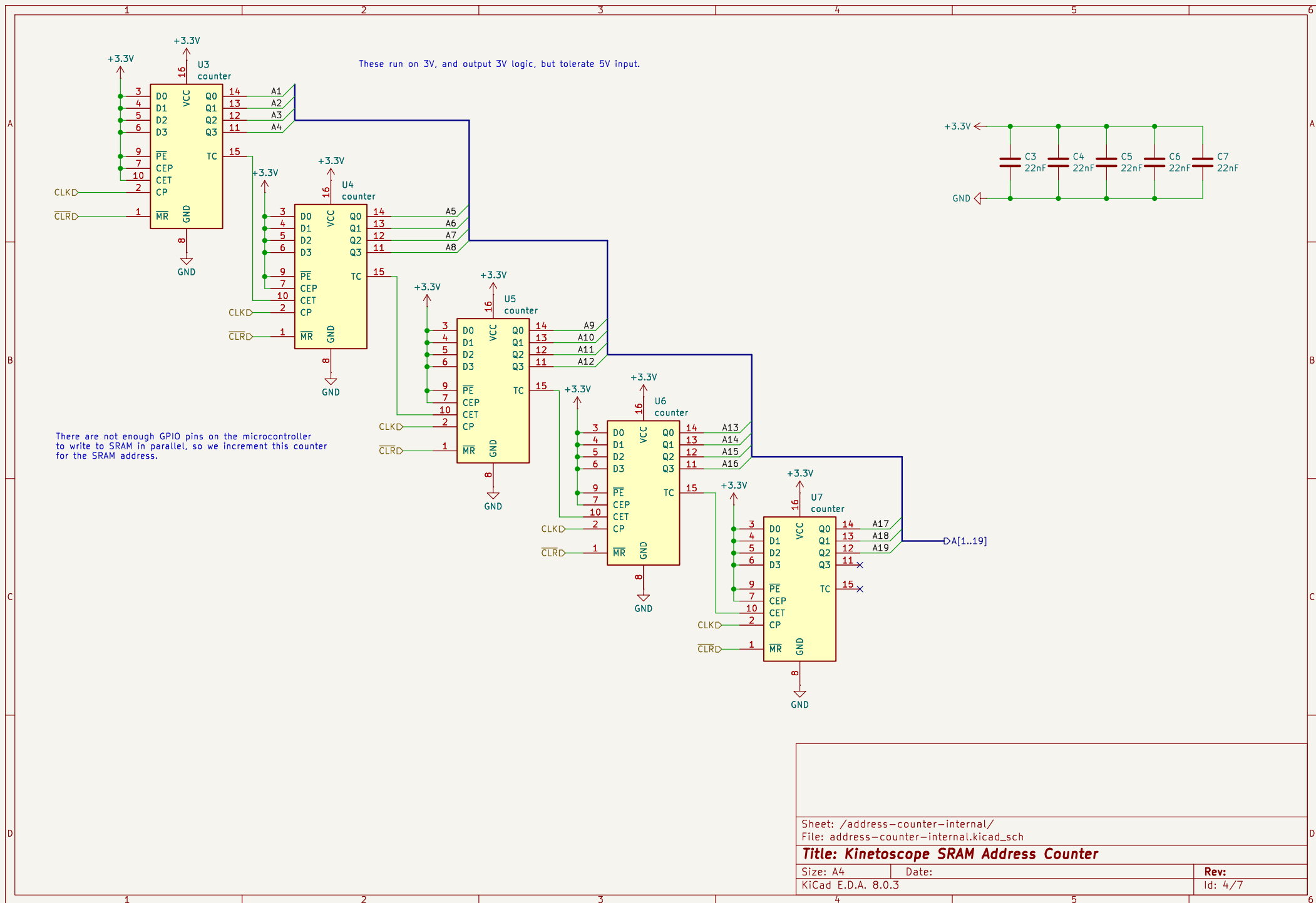
There are not enough GPIO pins on the microcontroller
to write to SRAM in parallel, so we fill this register serially.

D[0..15]

U8
shift register

+3.3V
VCC 14

9	MR		Q0	3	D0
			Q1	4	D1
			Q2	5	D2
1	DSA		Q3	6	D3
2	DSB		Q4	10	D4
			Q5	11	D5
8	CP		Q6	12	D6
			Q7	13	D7

D_IN

CLK

+3.3V

GND 7
GND

U9
shift register

+3.3V
VCC 14

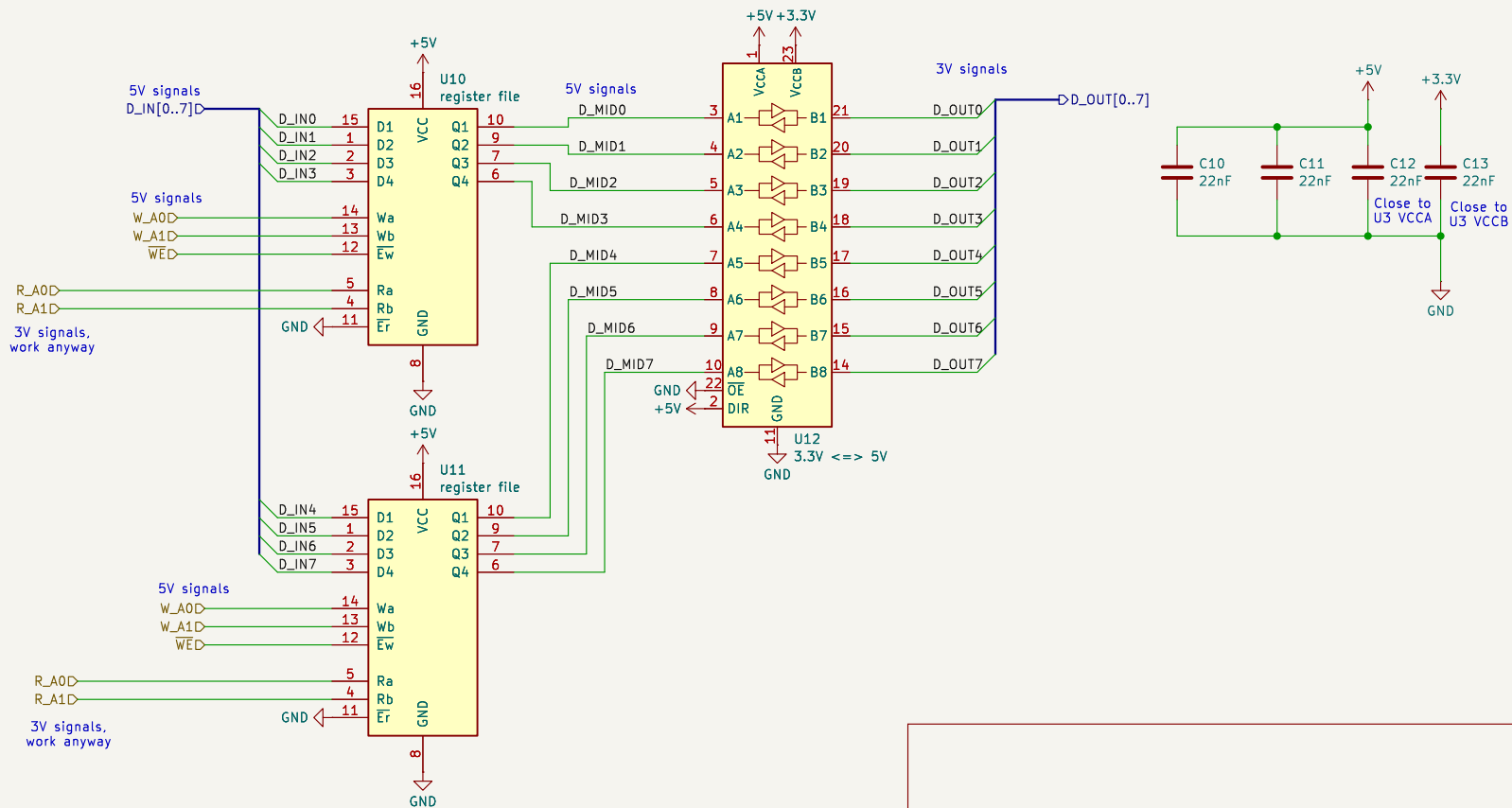| 9 | MR | | Q0 | 3 | D8 |
			Q1	4	D9
			Q2	5	D10
1	DSA		Q3	6	D11
2	DSB		Q4	10	D12
			Q5	11	D13
8	CP		Q6	12	D14
			Q7	13	D15

+3.3V

GND 7
GND

+3.3V

C8
22nF

C9
22nF

GND

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.

+5V

U10
register file

5V signals
D_IN[0..7]

		D_IN0	15	D1	VCC	Q1	10	D_MID0
		D_IN1	1	D2		Q2	9	D_MID1
		D_IN2	2	D3		Q3	7	D_MID2
		D_IN3	3	D4		Q4	6	D_MID3

5V signals

5V signals
W_A0
W_A1
WE

14	Wa
13	Wb
12	Ew

R_A0
R_A1

5	Ra
4	Rb
11	Er

GND

3V signals,
work anyway

GND

8

GND

+5V +3.3V

VccA VccB
1   23

U12
3.3V <=> 5V

D_MID0	3	A1	B1	21	D_OUT0
D_MID1	4	A2	B2	20	D_OUT1
D_MID2	5	A3	B3	19	D_OUT2
D_MID3	6	A4	B4	18	D_OUT3
D_MID4	7	A5	B5	17	D_OUT4
D_MID5	8	A6	B6	16	D_OUT5
D_MID6	9	A7	B7	15	D_OUT6
D_MID7	10	A8	B8	14	D_OUT7

GND 22 OE
+5V 2   DIR

GND 11

GND

3V signals
D_OUT[0..7]

+5V

U11
register file

		D_IN4	15	D1	VCC	Q1	10
		D_IN5	1	D2		Q2	9
		D_IN6	2	D3		Q3	7
		D_IN7	3	D4		Q4	6

5V signals
W_A0
W_A1
WE

14	Wa
13	Wb
12	Ew

R_A0
R_A1

5	Ra
4	Rb
11	Er

GND

3V signals,
work anyway

GND

8

GND

+5V   +3.3V

| C10 | C11 | C12 | C13 |
| 22nF | 22nF | 22nF | 22nF |

Close to   Close to
U3 VCCA   U3 VCCB

GND

This is a synchronization token between the M68k and the microcontroller.
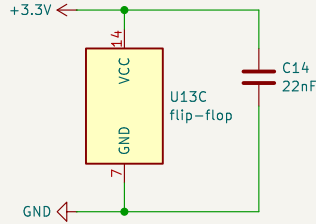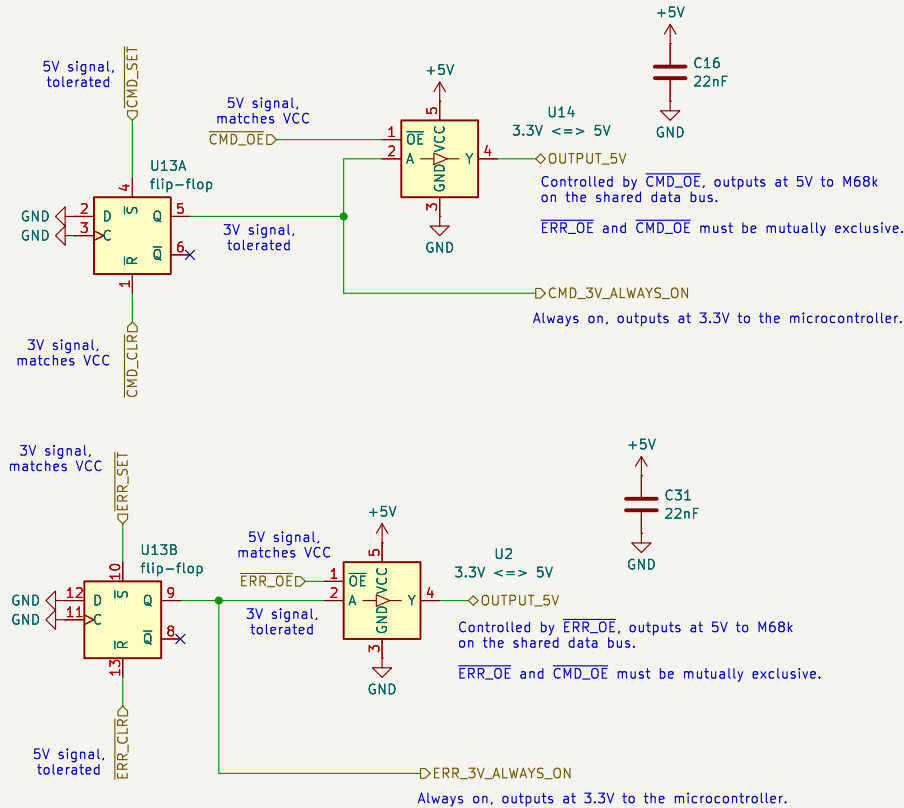
The token will be cleared on boot/reset.

The M68k will be able to set the token ($\overline{\text{CMD\_SET}}$), indicating to the microcontroller that a command is ready to be executed.

The microcontroller will be able to clear the token ($\overline{\text{CMD\_CLR}}$), 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 $\overline{\text{ERR\_SET}}$ to the M68k, who can clear it with $\overline{\text{ERR\_CLR}}$ after noticing.

U13A flip-flop

U14 3.3V <=> 5V

OUTPUT_5V

Controlled by $\overline{\text{CMD\_OE}}$, outputs at 5V to M68k on the shared data bus.

$\overline{\text{ERR\_OE}}$ and $\overline{\text{CMD\_OE}}$ must be mutually exclusive.

5V signal, tolerated

5V signal, matches VCC

$\overline{\text{CMD\_OE}}$

3V signal, tolerated

3V signal, tolerated

CMD_3V_ALWAYS_ON

Always on, outputs at 3.3V to the microcontroller.

GND
GND

$\overline{\text{CMD\_SET}}$

$\overline{\text{CMD\_CLR}}$

3V signal, matches VCC

C16 22nF

+5V

GND

U13B flip-flop

U2 3.3V <=> 5V

OUTPUT_5V

Controlled by $\overline{\text{ERR\_OE}}$, outputs at 5V to M68k on the shared data bus.

$\overline{\text{ERR\_OE}}$ and $\overline{\text{CMD\_OE}}$ must be mutually exclusive.

3V signal, matches VCC

5V signal, matches VCC

$\overline{\text{ERR\_OE}}$

3V signal, tolerated

5V signal, tolerated

ERR_3V_ALWAYS_ON

Always on, outputs at 3.3V to the microcontroller.

GND
GND

$\overline{\text{ERR\_SET}}$

$\overline{\text{ERR\_CLR}}$

C31 22nF

+5V

GND

U13C flip-flop

+3.3V

GND

C14 22nF

Title: Kinetoscope Sync Token

Size: A4

Date:

Rev:

KiCad E.D.A. 8.0.3

Id: 7/7