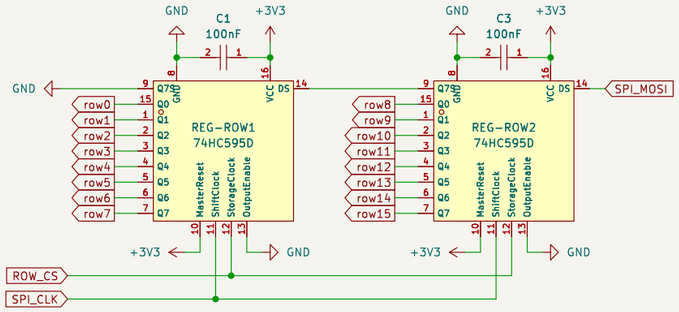
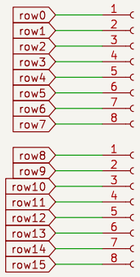
Row shift registers

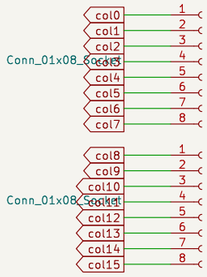


74HC595 notes
- When we spi_start(ROW_CS), ROW_CS goes LOW — StorageClock is primed
- spi_transmit(message) — data is staged!
- spi_stop(ROW_CS), ROW_CS goes HIGH — StorageClock rises, triggering Latch
- on Latch, the data is moved to prod! and pins go live with your bits.
- except we are going to also manually trigger latch to ensure no bit is left behind

Row pins

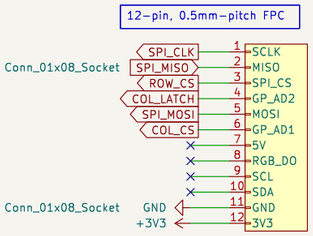


Col pins



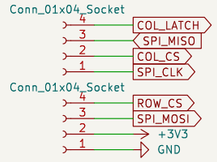
VIK interconnect

12-pin, 0.5mm-pitch FPC

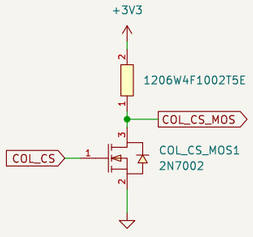


VIK alternate

2x 4-pin 2.54 header as VIK alternate

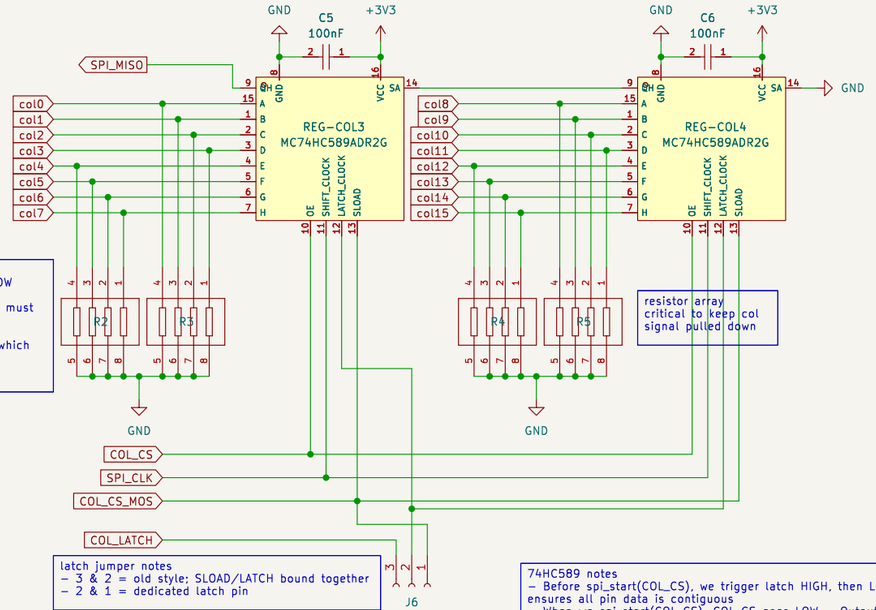


COL_CS mosfet



mosfet notes
- when we activate COL_CS, spi_start() pulls COL_CS LOW
- this causes the mosfet switch to open
- CS_COL_MOS goes HIGH (for SLOAD to do its thing, it must be HIGH)
- spi_read()
- COL_CS is deactivated — goes HIGH — by spi_stop() which causes COL_CS_MOS to go LOW

Column shift registers



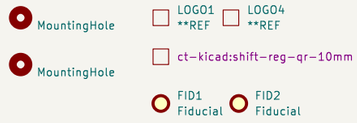
resistor array
critical to keep col signal pulled down

latch jumper notes
- 3 & 2 = old style; SLOAD/LATCH bound together
- 2 & 1 = dedicated latch pin

74HC589 notes
- Before spi_start(COL_CS), we trigger latch HIGH, then LOW; this ensures all pin data is contiguous
- When we spi_start(COL_CS), COL_CS goes LOW — OutputEnable is activated
- COL_CS_MOS is HIGH, so SLoad is activated
- Data flows out into MISO
- spi_stop(COL_CS), COL_CS goes HIGH, OutputEnable is deactivated

Design note: Cyboard's flex-pcb system is ROW2COL.

Bits and bobs



Sheet: /
File: shift-register-breakout-pcb.kicad_sch

Title:

Size: A3	Date:	**Rev:**
KiCad E.D.A. 9.0.0		Id: 1/1