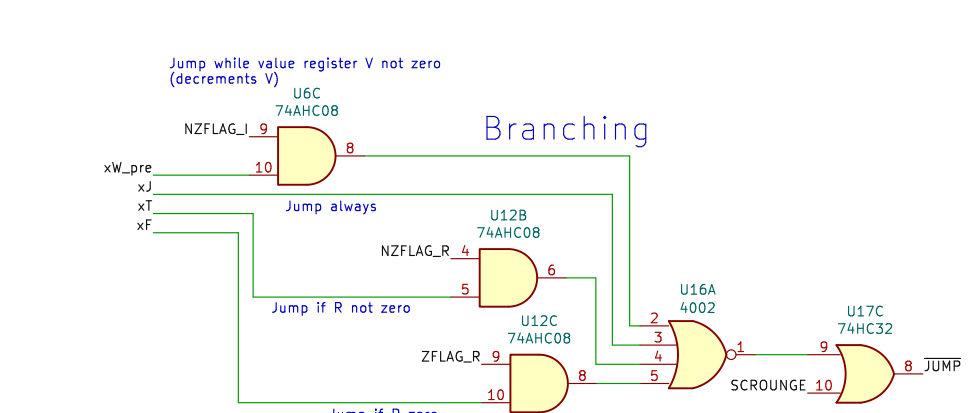
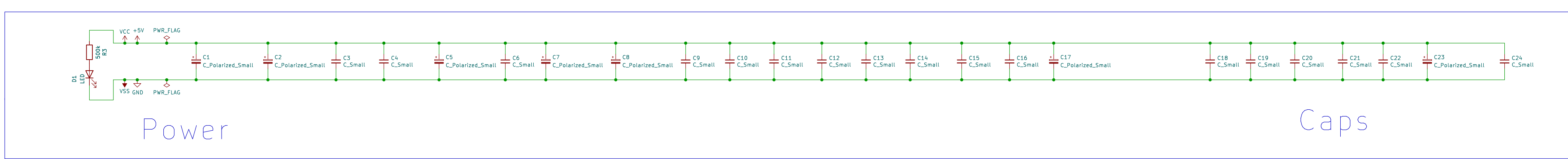
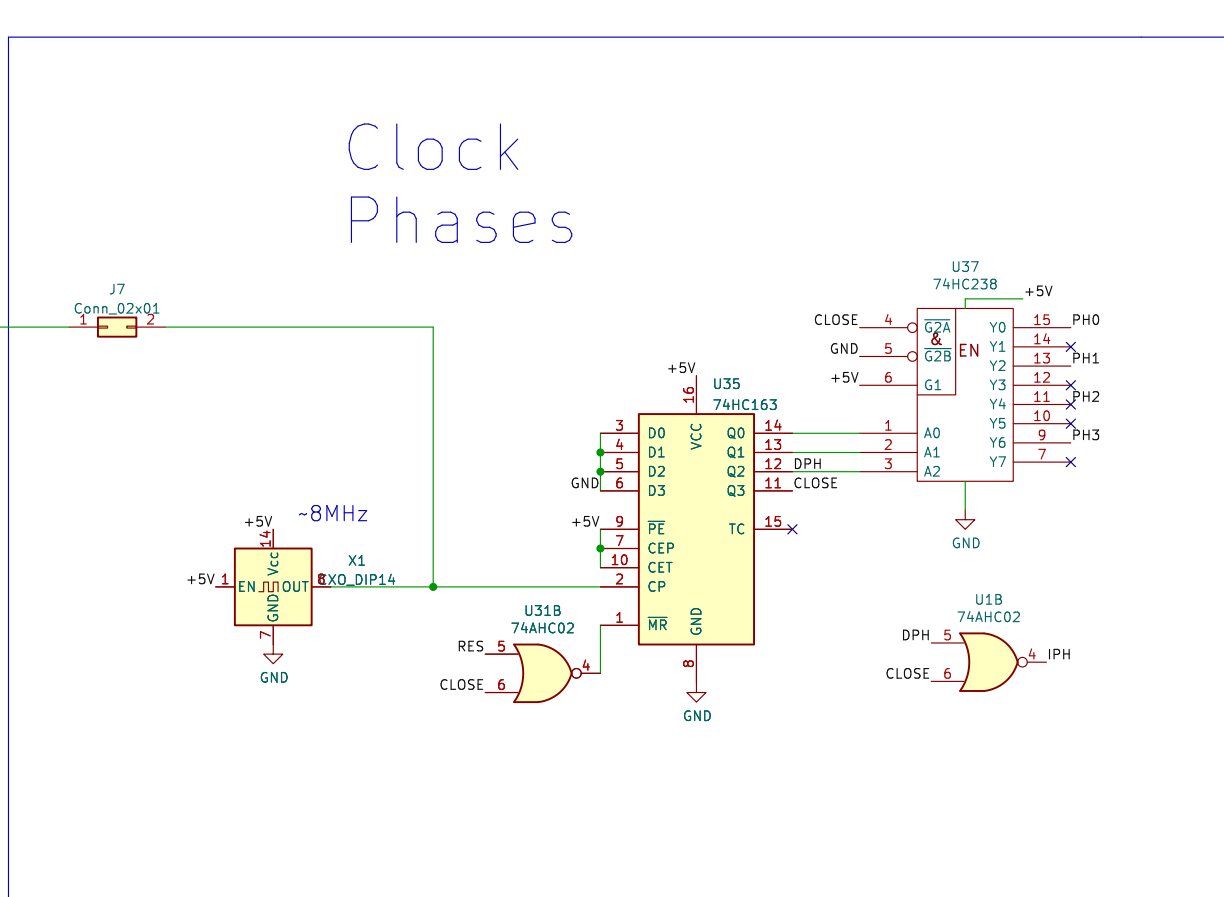
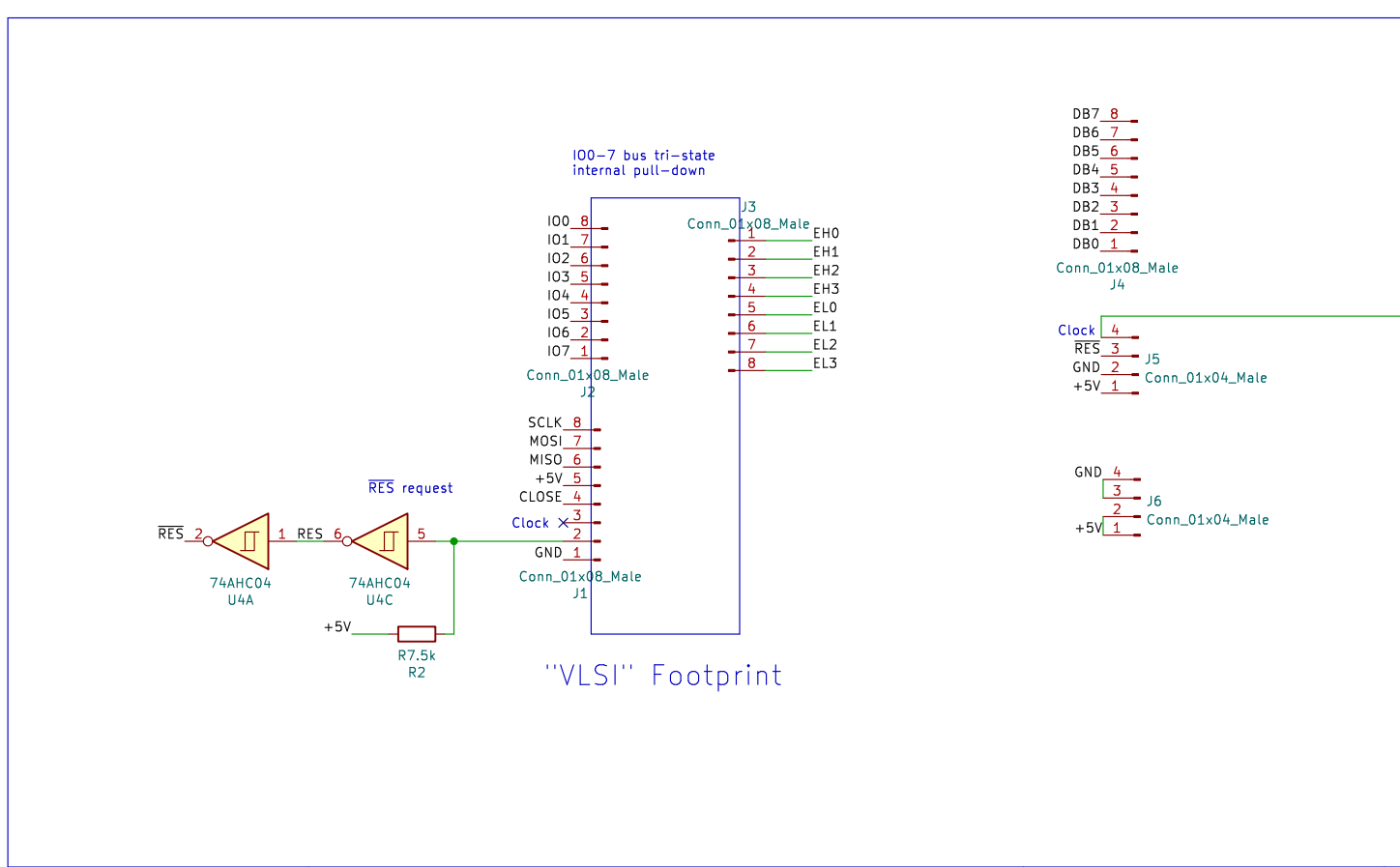


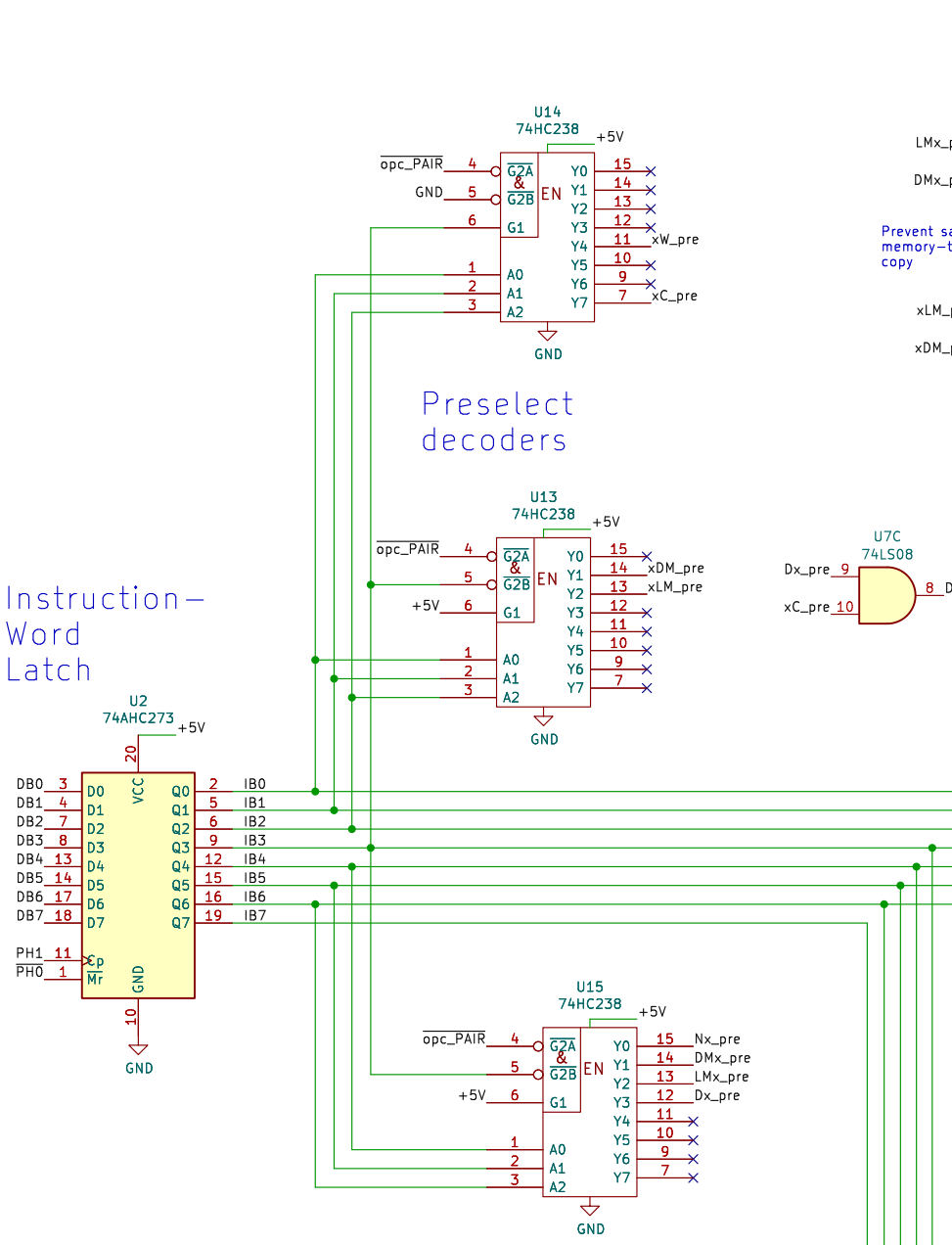
Myth Microcontroller Reference Schematics



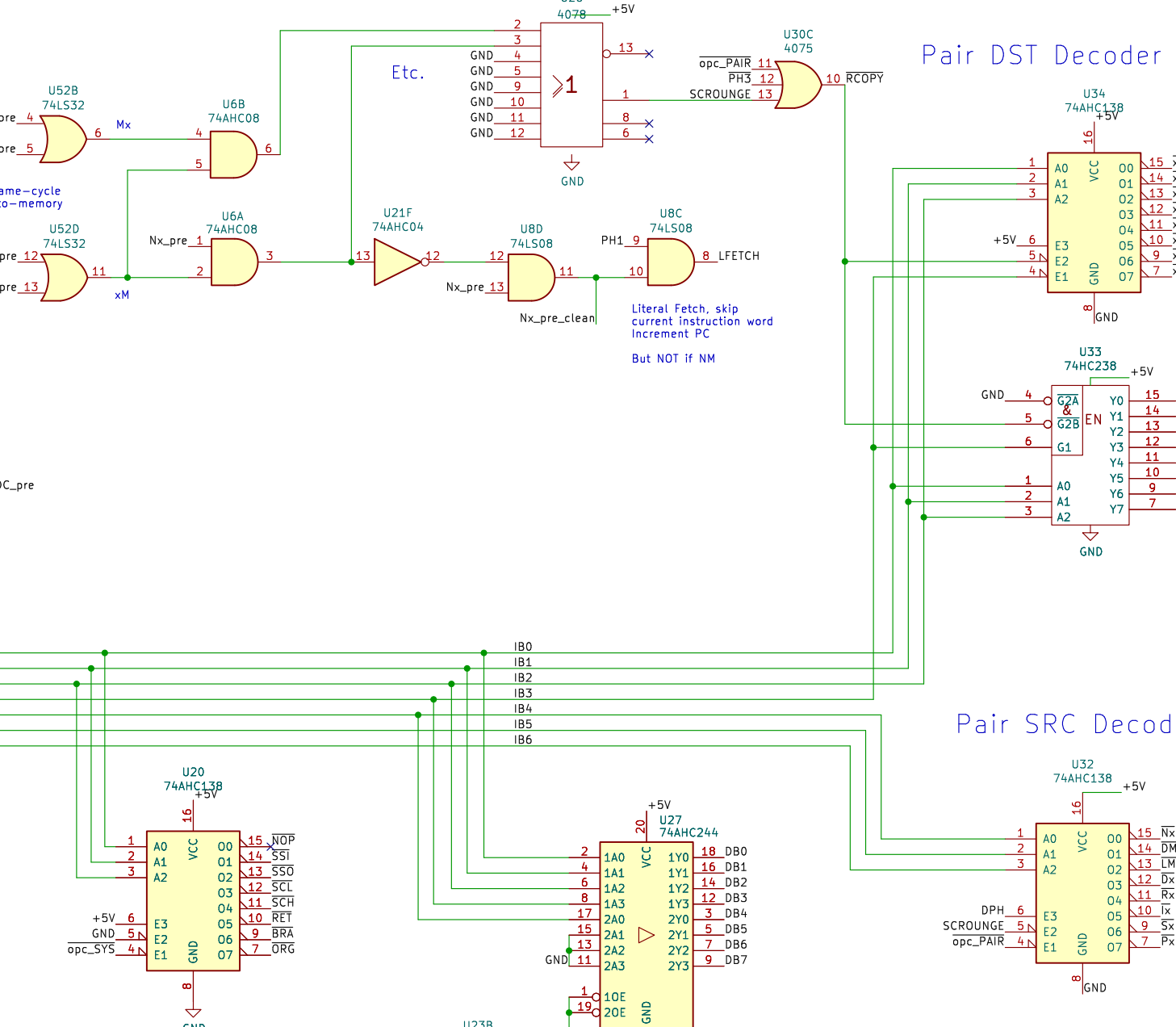
Instruction Decoder

Scrounger

Remap MM, RR, DD etc. and impracticable opcodes such as NM. Currently NOP, reserved for instruction set extension.



PHA0: Setup PC on address bus
PHA1: Latch opcode into IB0-7, predecoded
PHA2: Decode setup source file on IB or other action
PHA3: Latch source into target or other action
CLOSE: Cleanup

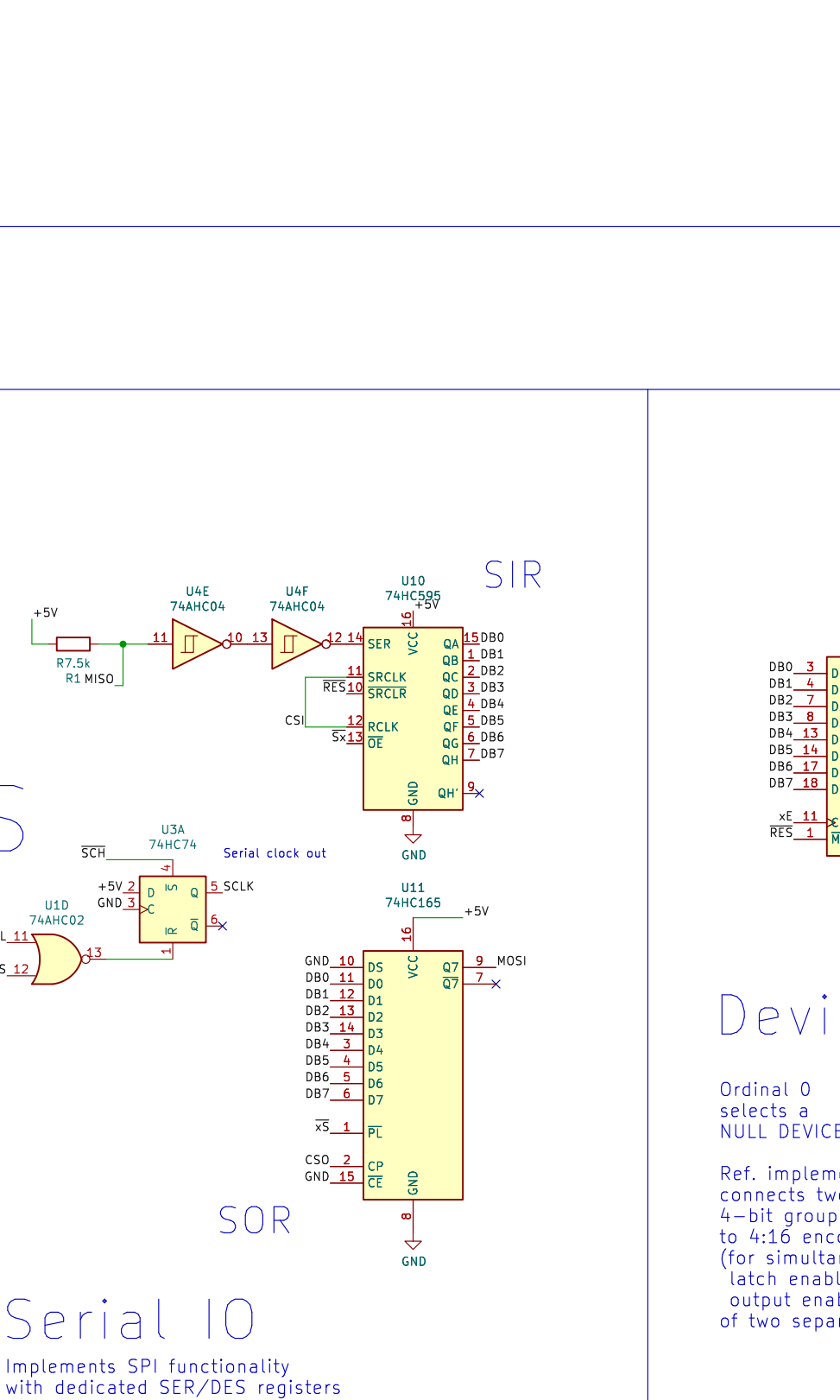
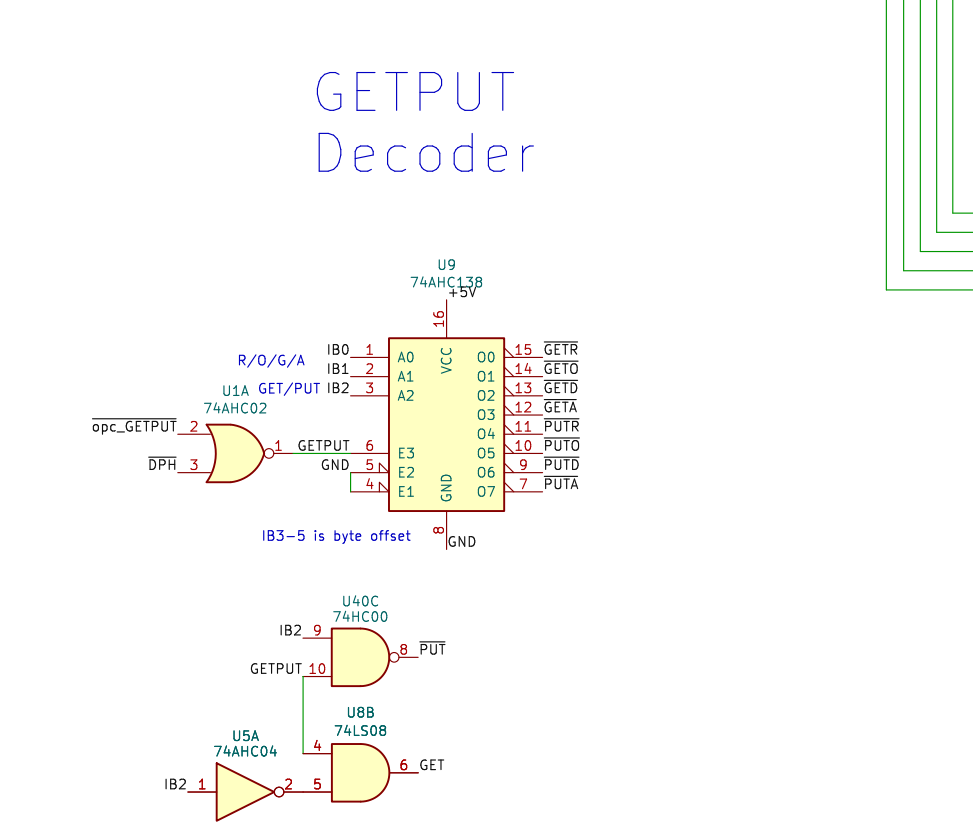


SYS decoder

Opcode Type Decoder

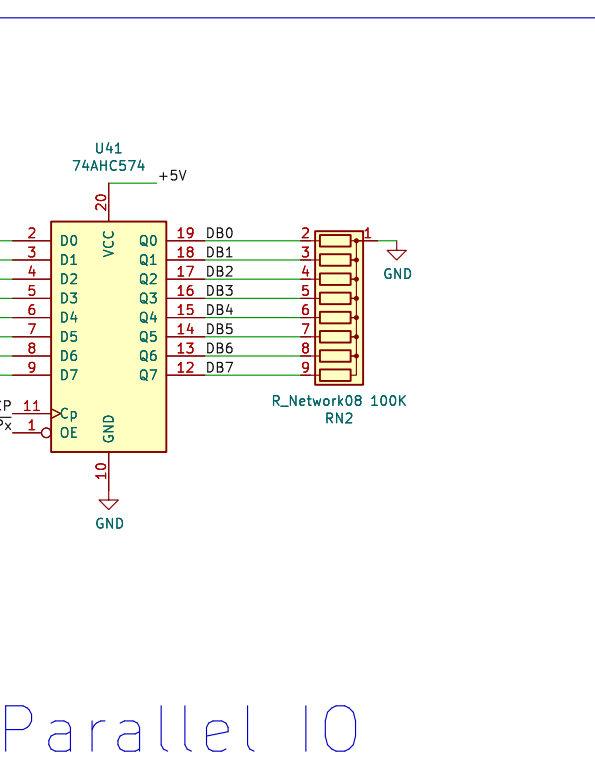
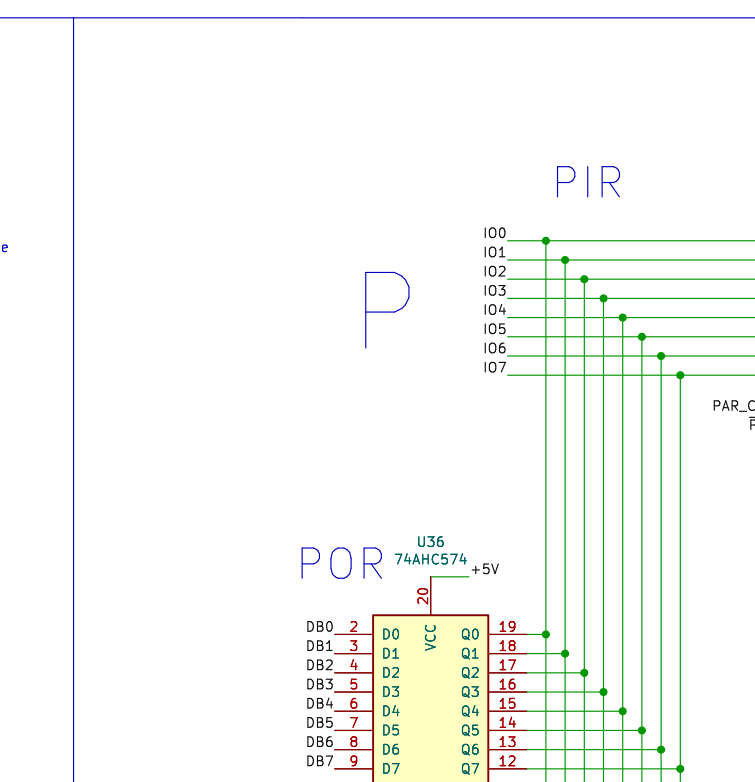
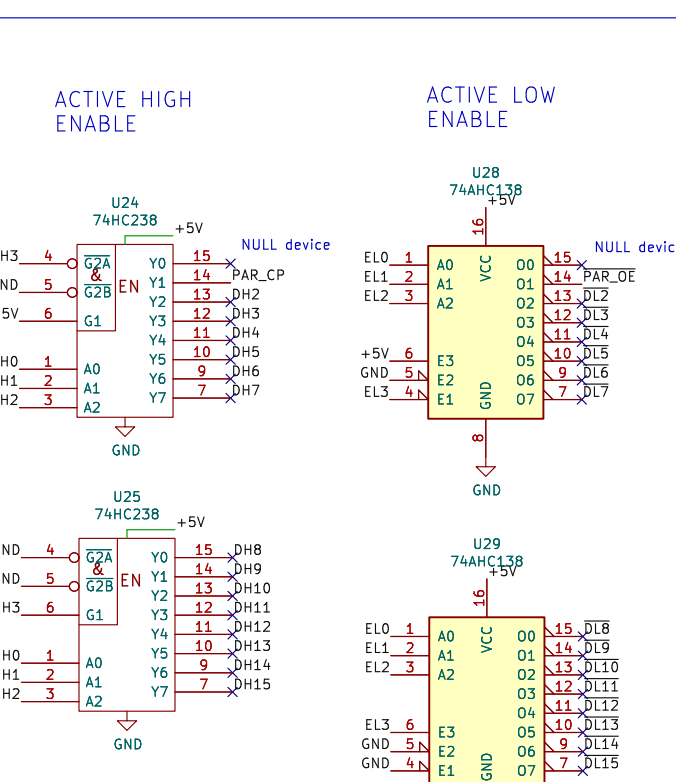
Instruction byte:
a10: OPC_SYS
else: OPC_ADJUST
OPC_ALU
OPC_TRAP
OPC_GETPUT
OPC_PAIR

See table @ SYS decoder
b2: extended sign bit b1-0: LSB
See table @ ALU
b4: MSB b3-0: LSB (remaining bits set 1)
b5-3: OFFS b2: GL b1: GP B0: RD
DEST SRC



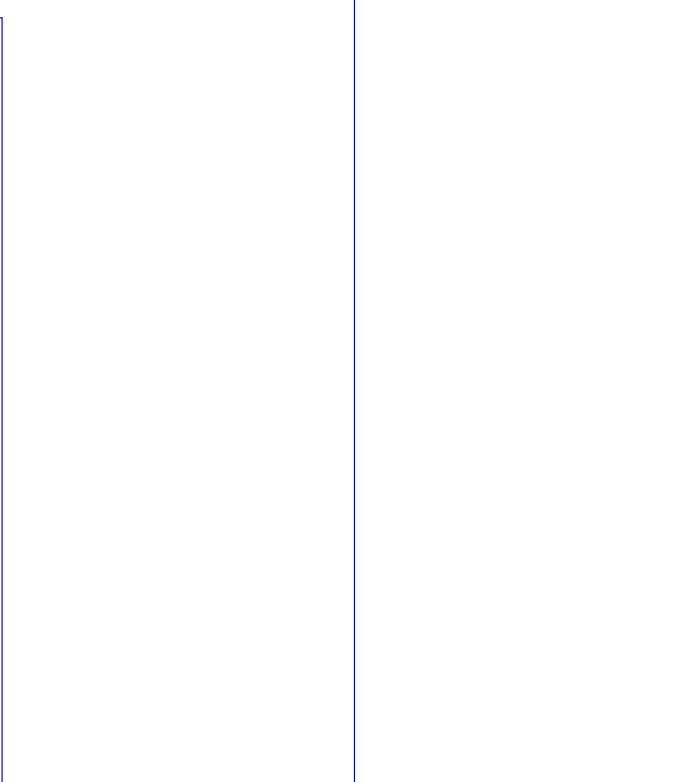
Device Enable

Original 0 selects a NULL DEVICE
Ref. implementation connects two independent 4-bit groups (high/low) to 4-bit encoders (for simultaneous latch enable / output enable of two separate devices)

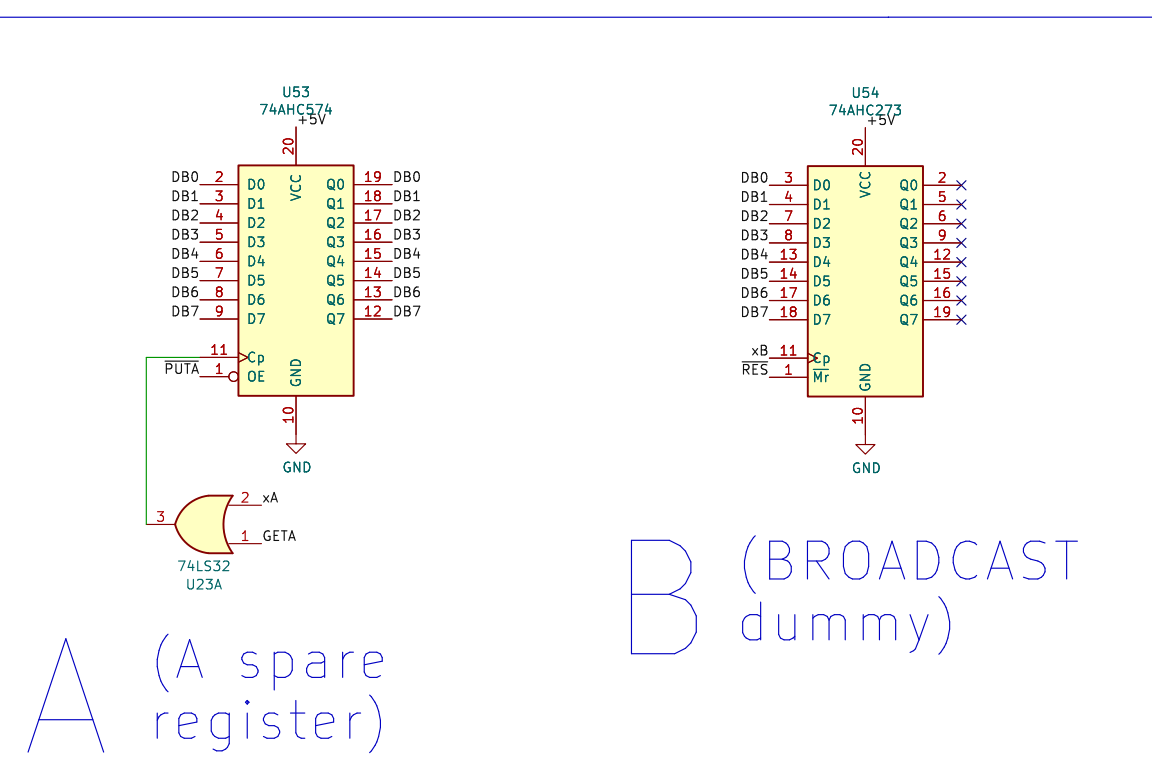
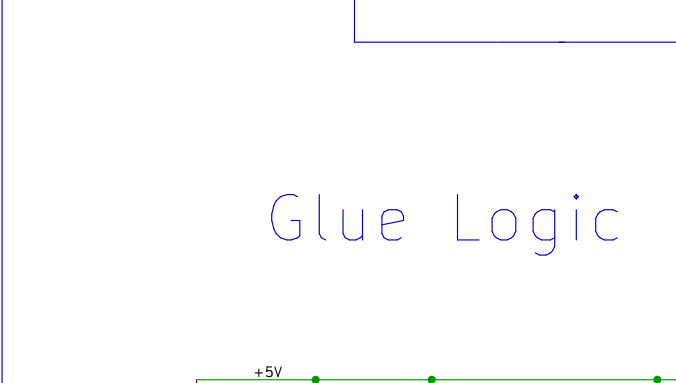


Parallel IO

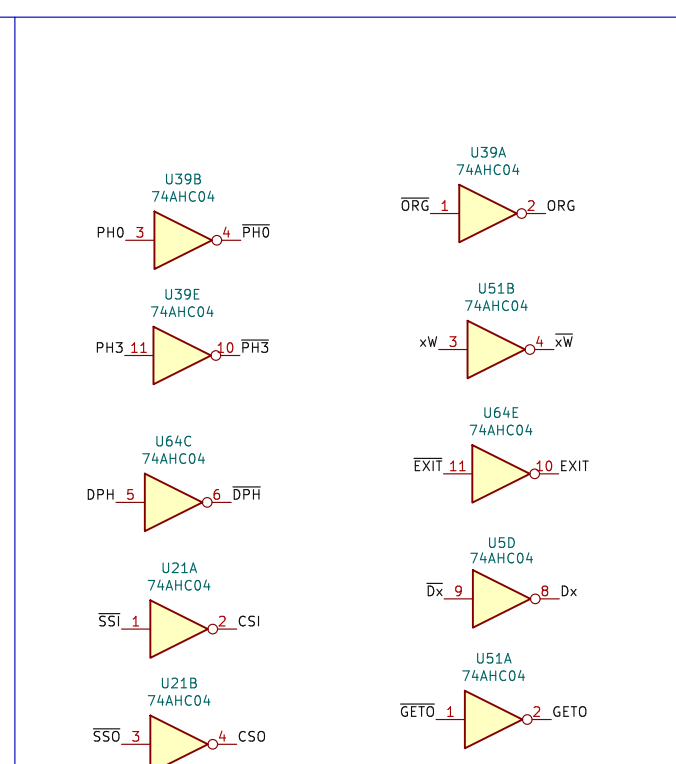
Implement an 8-bit parallel tri-state interface bus with dedicated in/out ports



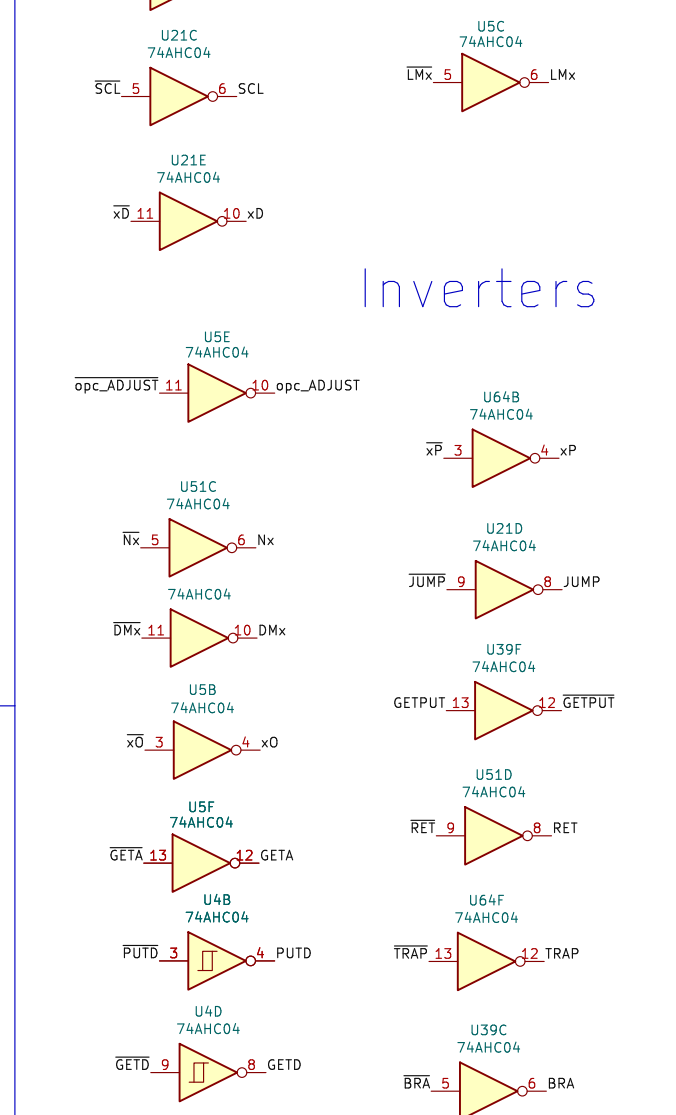
Glue Logic



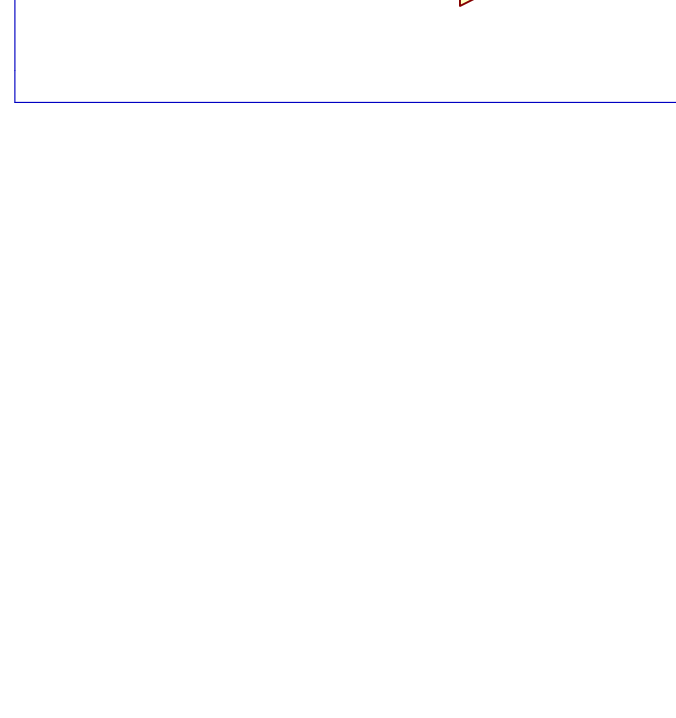
B (BROADCAST dummy)



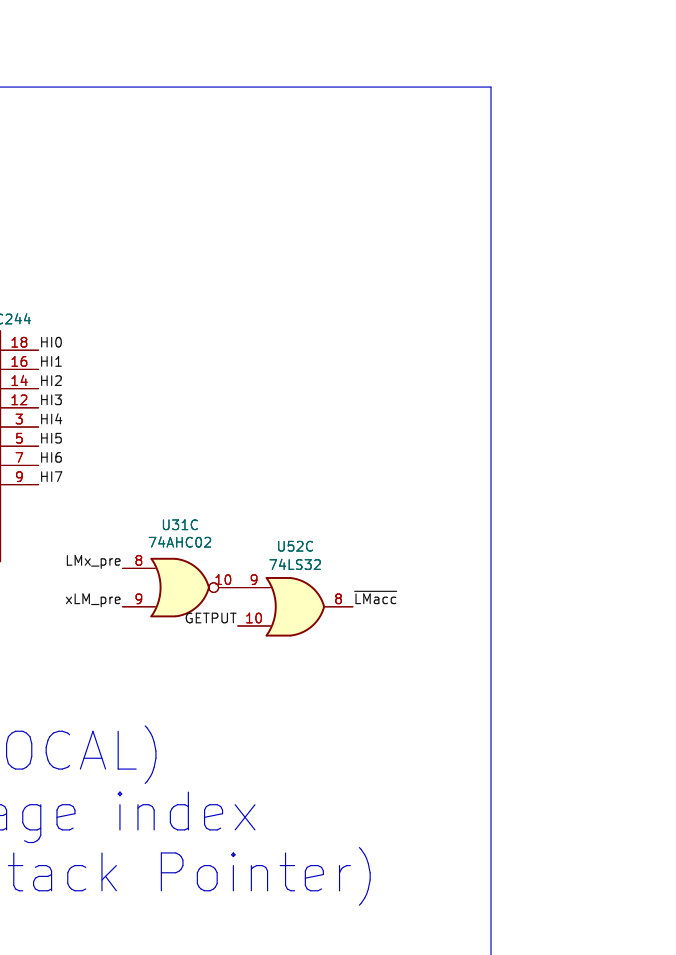
J (JUMP register) (Program Counter)



(Relative addressing)



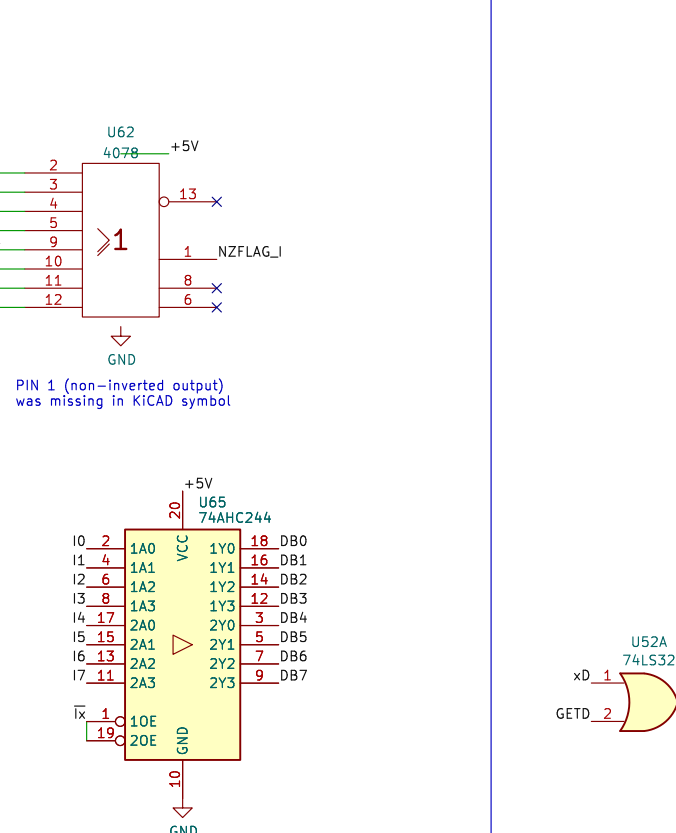
J (the program counter) is cleared/set to 0 during TRAP or CALL. Execution starts at offset zero, the "head" of the destination page.



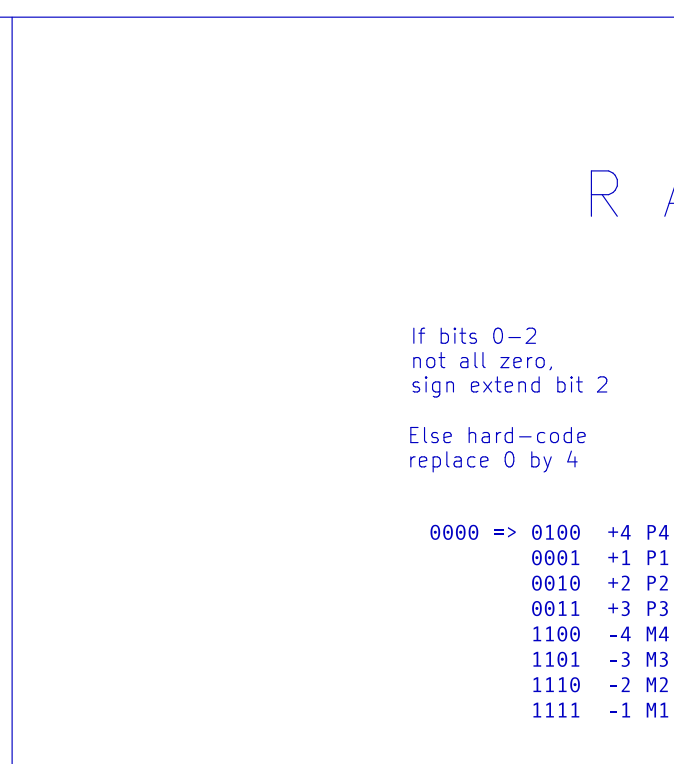
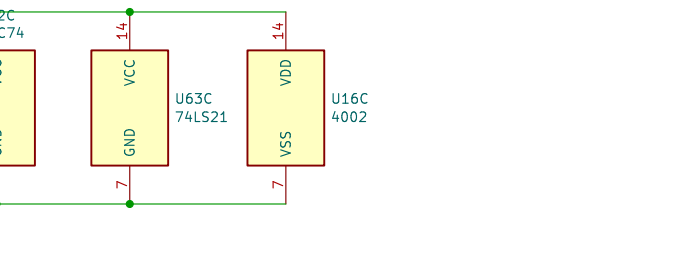
Delay EXIT until CLOSE

Automatic call stack CALL: Enter stack frame RET: Resume previous frame

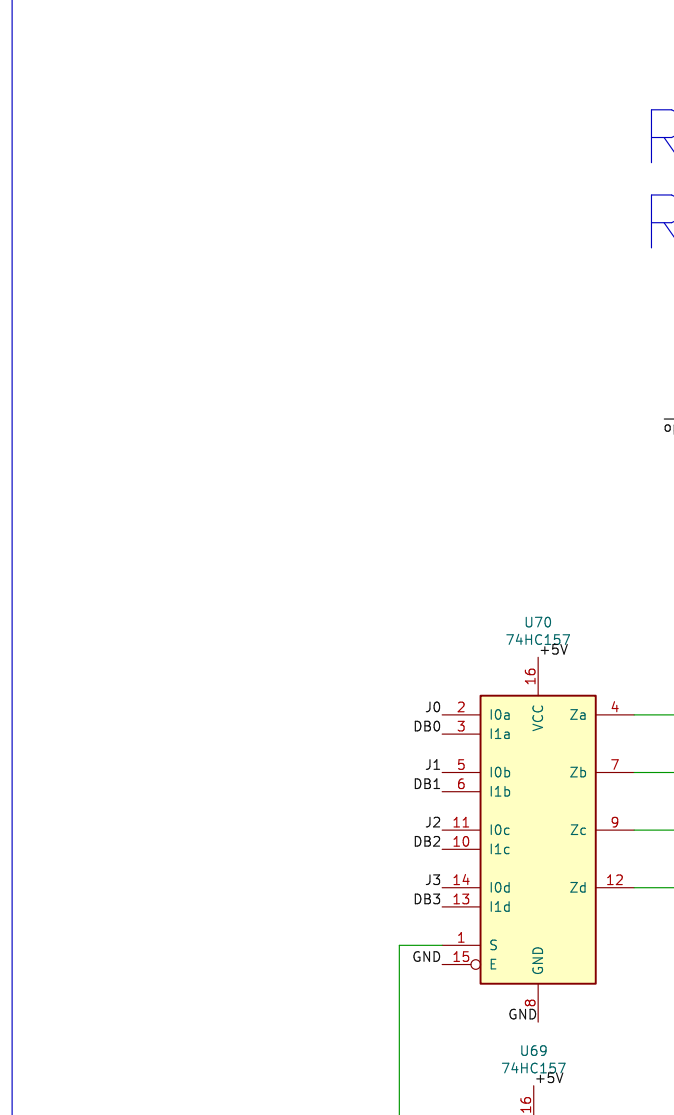
L (LOCAL) Page index (Stack Pointer)



I (INNER Loop Register)

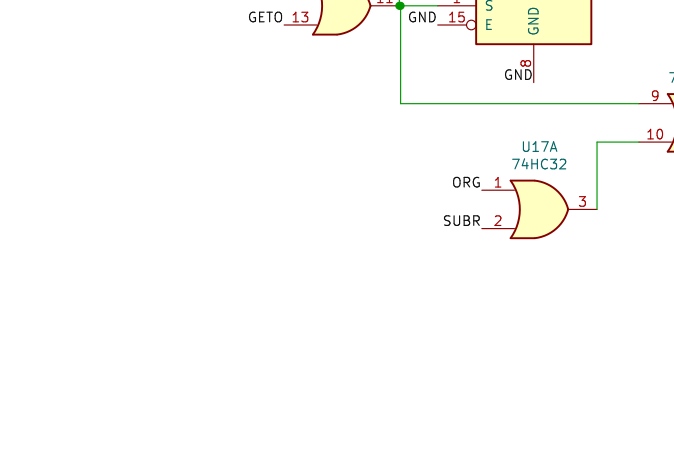


R (result) Register

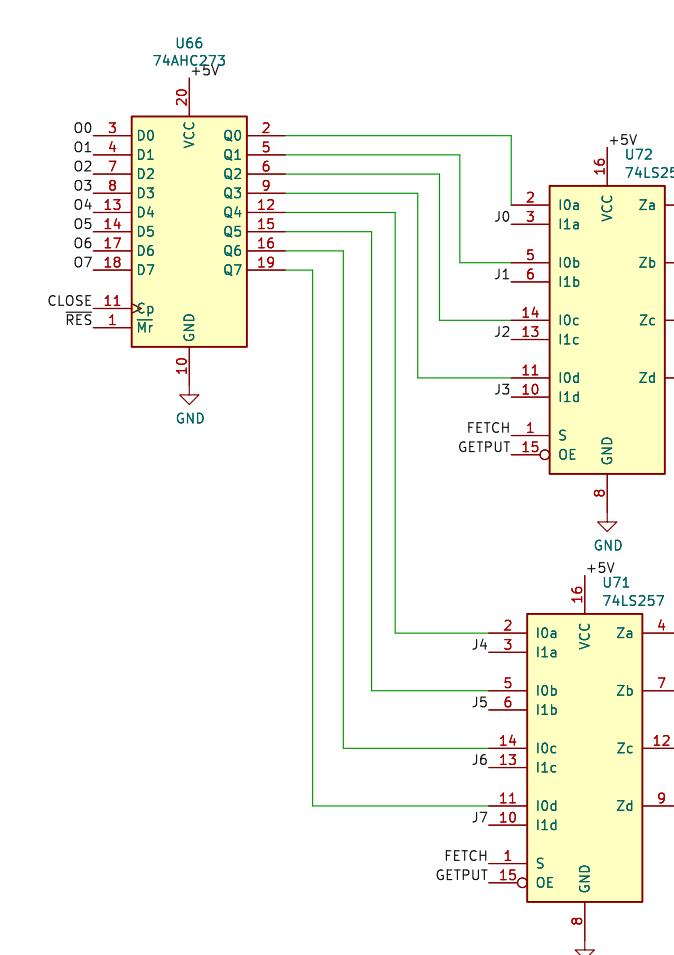


0 provides the return offset during RET receives the return offset during SUBR can be GET and PUT provides all address offsets except FETCH can be stored into is the implied 2nd ALU input operand

O (OFFSET) Operand Register



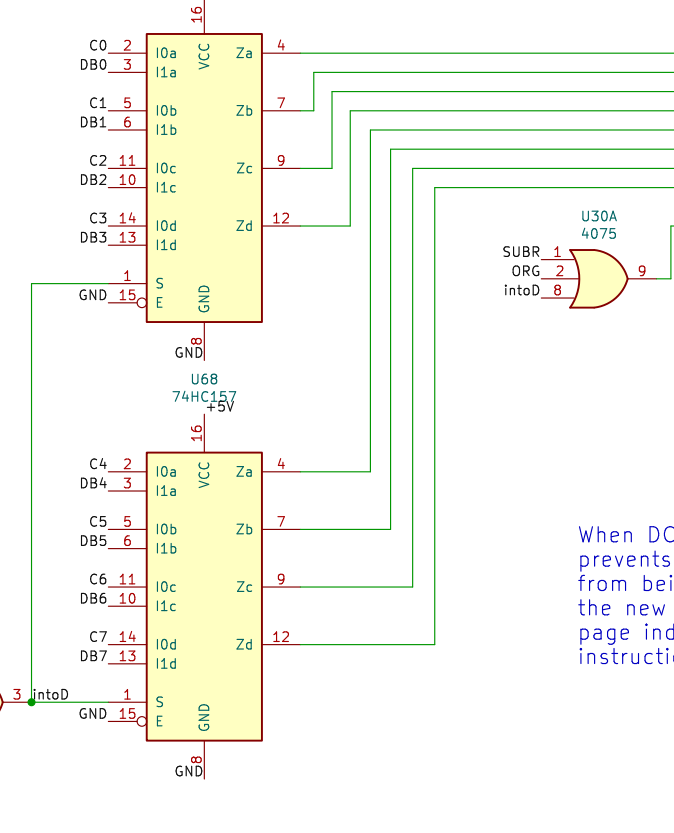
0 DIR ("identity" R / R=0)
1 DIR ("identity" O / R=0)
2 DIR ("Ones Complement" R / R=0)
3 DIR ("Ones Complement" O / R=0)
4 SLR ("Shift left New / R=0<<1")
5 SLR ("Shift left New / R=0<<1")
6 SRR ("Shift right R / R=0>>1")
7 SRR ("Shift right R / R=0>>1")
8 AND (R=R&O)
9 OR (R=R|O)
10 XOR (R=R^O)
11 ADD (Add / R=R+O, bits 0-7)
12 CAR ("Carry Bit" / R=Carry 8th bit of R+O 00h or 01h)
13 RLO ("Flag: R less than O" / R=(R<O)? 0xFF:0)
14 RCO ("Flag: R equals O" / R=(R==O)? 0xFF:0)
15 RGO ("Flag: R greater than O" / R=(R>O)? 0xFF:0)



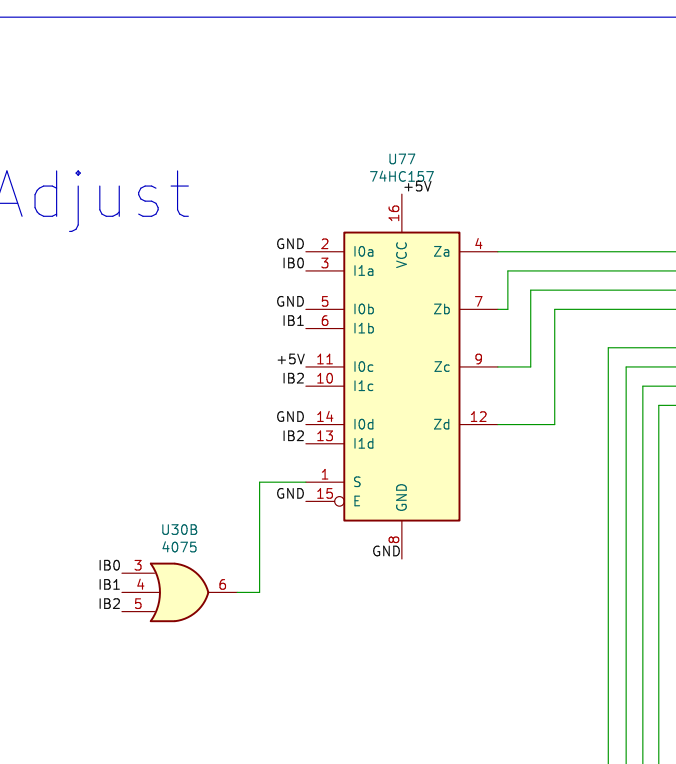
Memory M

16K

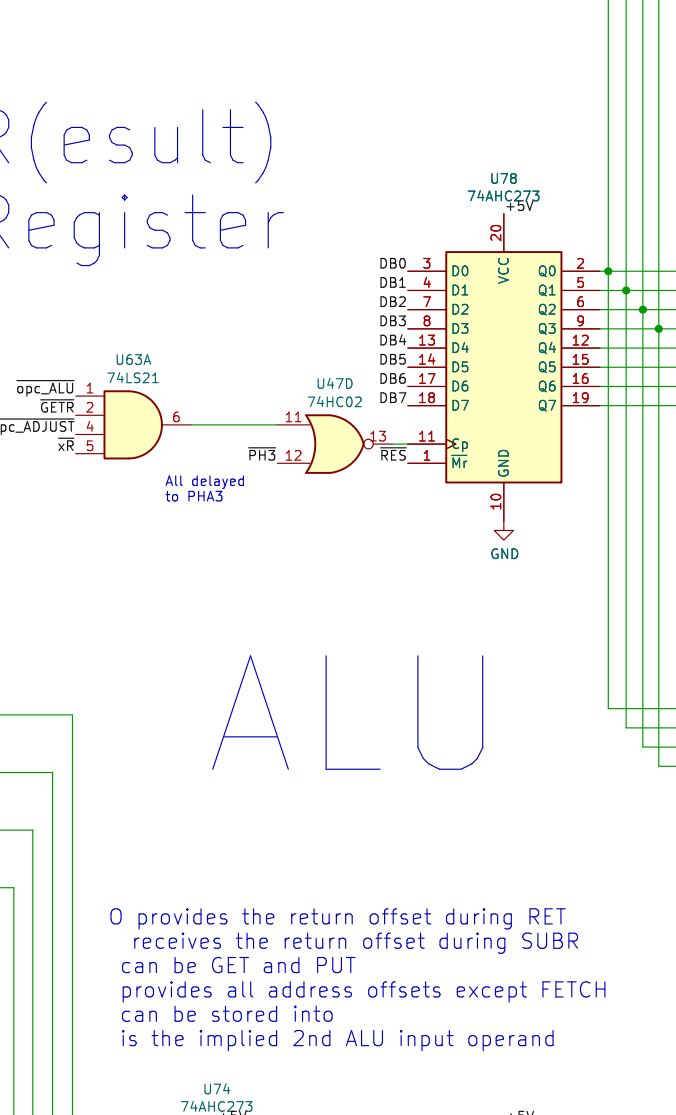
Select Program Memory during Code Phase (P0) and Data Phase (P1)



Spare Units

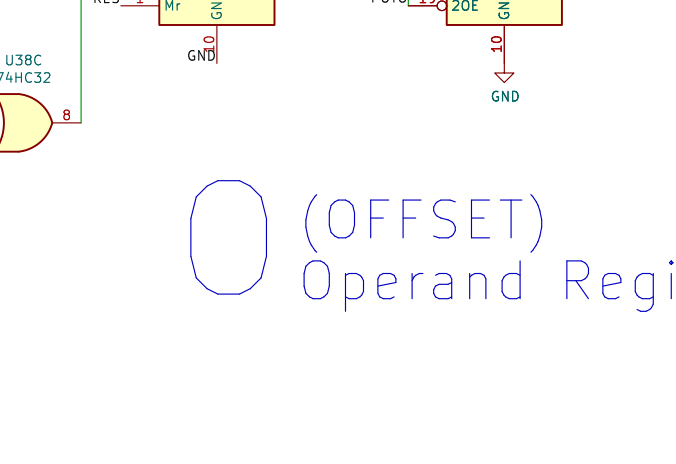


Zero detection



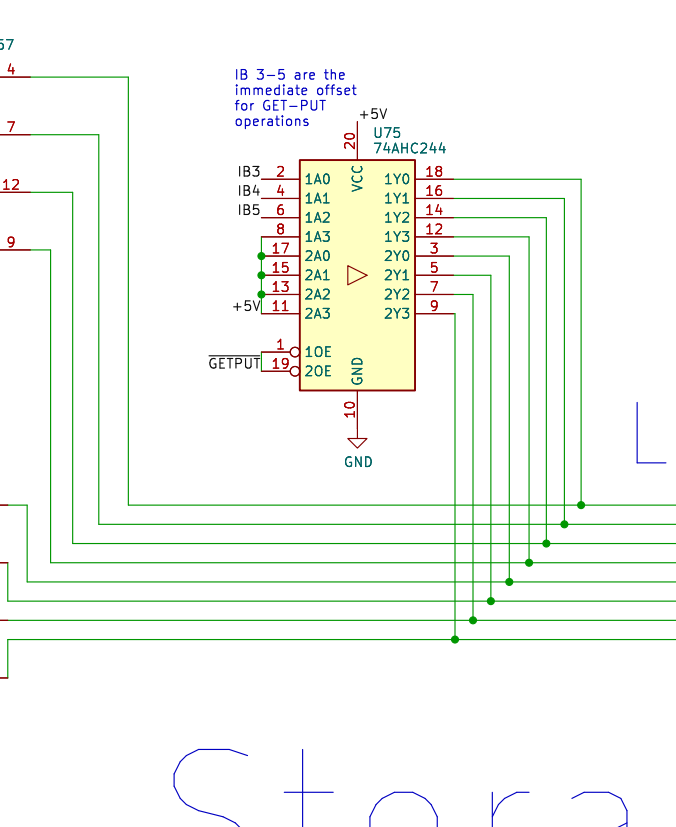
BLURS LUT ROM layout: 16 maps @256x256

BLURS LUT ROM layout



0 DIR ("identity" R / R=0)
1 DIR ("identity" O / R=0)
2 DIR ("Ones Complement" R / R=0)
3 DIR ("Ones Complement" O / R=0)
4 SLR ("Shift left New / R=0<<1")
5 SLR ("Shift left New / R=0<<1")
6 SRR ("Shift right R / R=0>>1")
7 SRR ("Shift right R / R=0>>1")
8 AND (R=R&O)
9 OR (R=R|O)
10 XOR (R=R^O)
11 ADD (Add / R=R+O, bits 0-7)
12 CAR ("Carry Bit" / R=Carry 8th bit of R+O 00h or 01h)
13 RLO ("Flag: R less than O" / R=(R<O)? 0xFF:0)
14 RCO ("Flag: R equals O" / R=(R==O)? 0xFF:0)
15 RGO ("Flag: R greater than O" / R=(R>O)? 0xFF:0)

Storage



D (DATA) Page index

When DC is executed, prevents the DS output from being updated with the new return address page index, before the instruction is done.

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

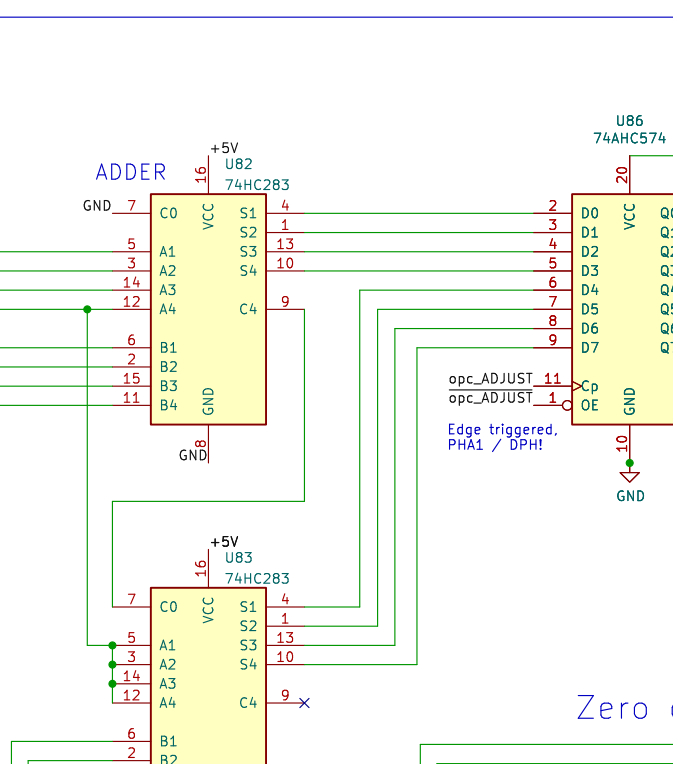
Implements SPI functionality with dedicated SER/DES registers

Serial IO

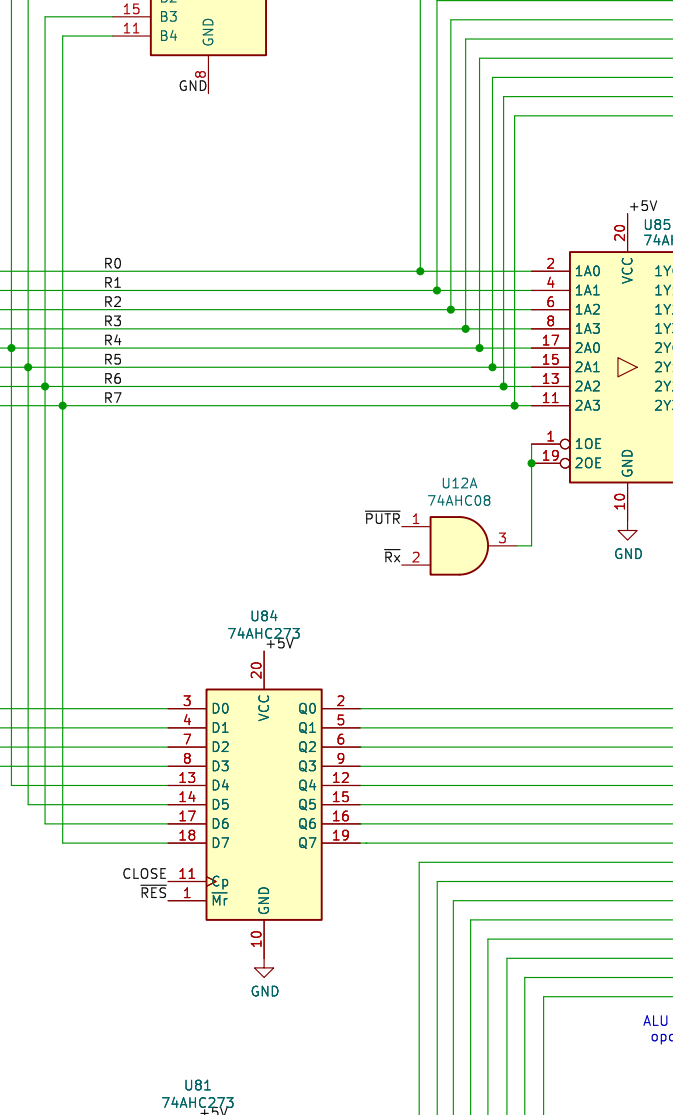
Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers

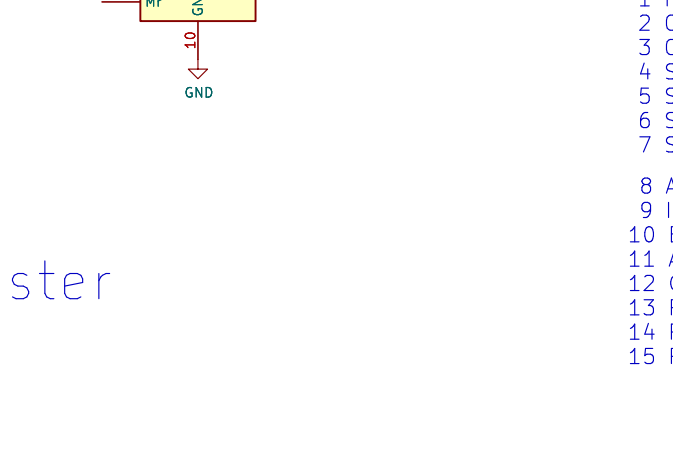


ADDER



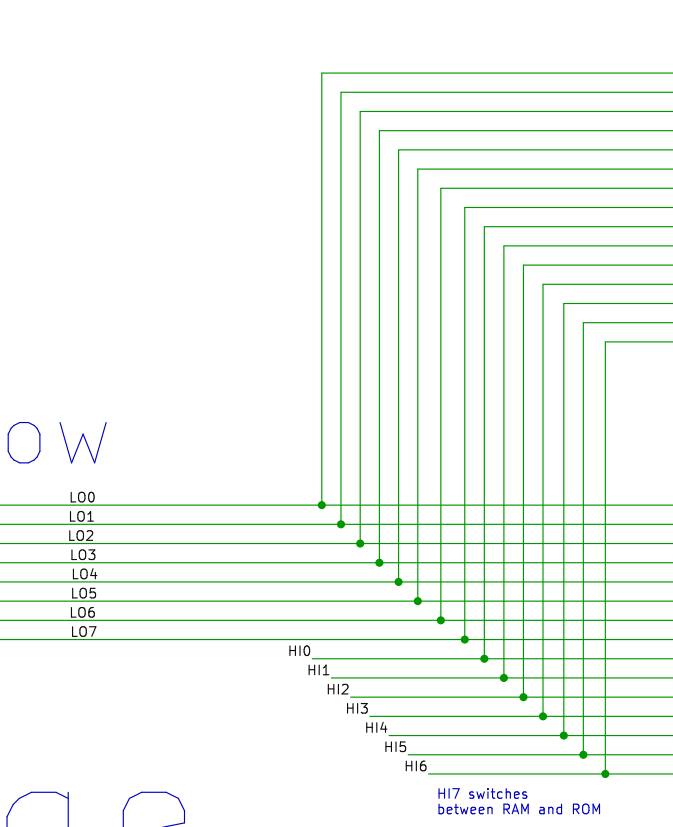
0 DIR ("identity" R / R=0)
1 DIR ("identity" O / R=0)
2 DIR ("Ones Complement" R / R=0)
3 DIR ("Ones Complement" O / R=0)
4 SLR ("Shift left New / R=0<<1")
5 SLR ("Shift left New / R=0<<1")
6 SRR ("Shift right R / R=0>>1")
7 SRR ("Shift right R / R=0>>1")
8 AND (R=R&O)
9 OR (R=R|O)
10 XOR (R=R^O)
11 ADD (Add / R=R+O, bits 0-7)
12 CAR ("Carry Bit" / R=Carry 8th bit of R+O 00h or 01h)
13 RLO ("Flag: R less than O" / R=(R<O)? 0xFF:0)
14 RCO ("Flag: R equals O" / R=(R==O)? 0xFF:0)
15 RGO ("Flag: R greater than O" / R=(R>O)? 0xFF:0)

BLURS LUT ROM layout



0 DIR ("identity" R / R=0)
1 DIR ("identity" O / R=0)
2 DIR ("Ones Complement" R / R=0)
3 DIR ("Ones Complement" O / R=0)
4 SLR ("Shift left New / R=0<<1")
5 SLR ("Shift left New / R=0<<1")
6 SRR ("Shift right R / R=0>>1")
7 SRR ("Shift right R / R=0>>1")
8 AND (R=R&O)
9 OR (R=R|O)
10 XOR (R=R^O)
11 ADD (Add / R=R+O, bits 0-7)
12 CAR ("Carry Bit" / R=Carry 8th bit of R+O 00h or 01h)
13 RLO ("Flag: R less than O" / R=(R<O)? 0xFF:0)
14 RCO ("Flag: R equals O" / R=(R==O)? 0xFF:0)
15 RGO ("Flag: R greater than O" / R=(R>O)? 0xFF:0)

Storage



D (DATA) Page index

When DC is executed, prevents the DS output from being updated with the new return address page index, before the instruction is done.

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

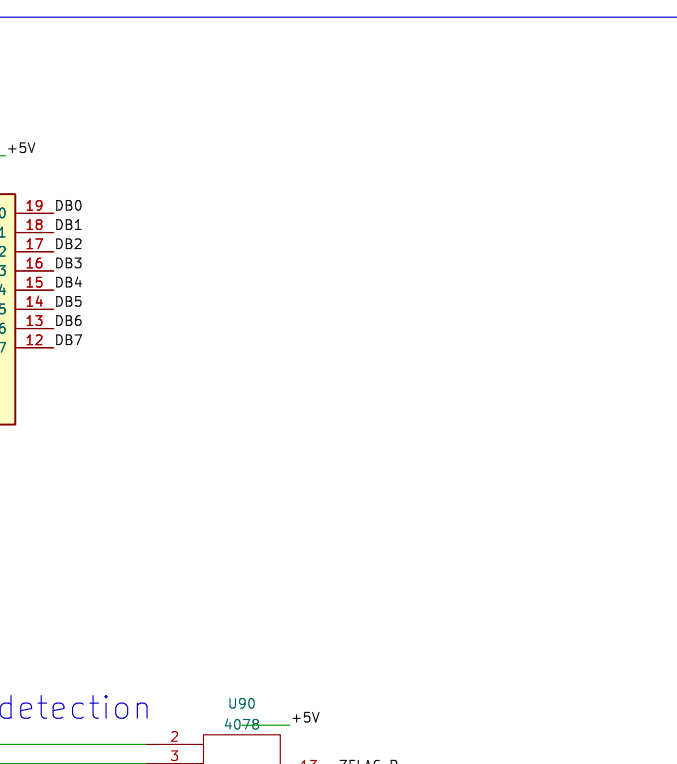
Implements SPI functionality with dedicated SER/DES registers

Serial IO

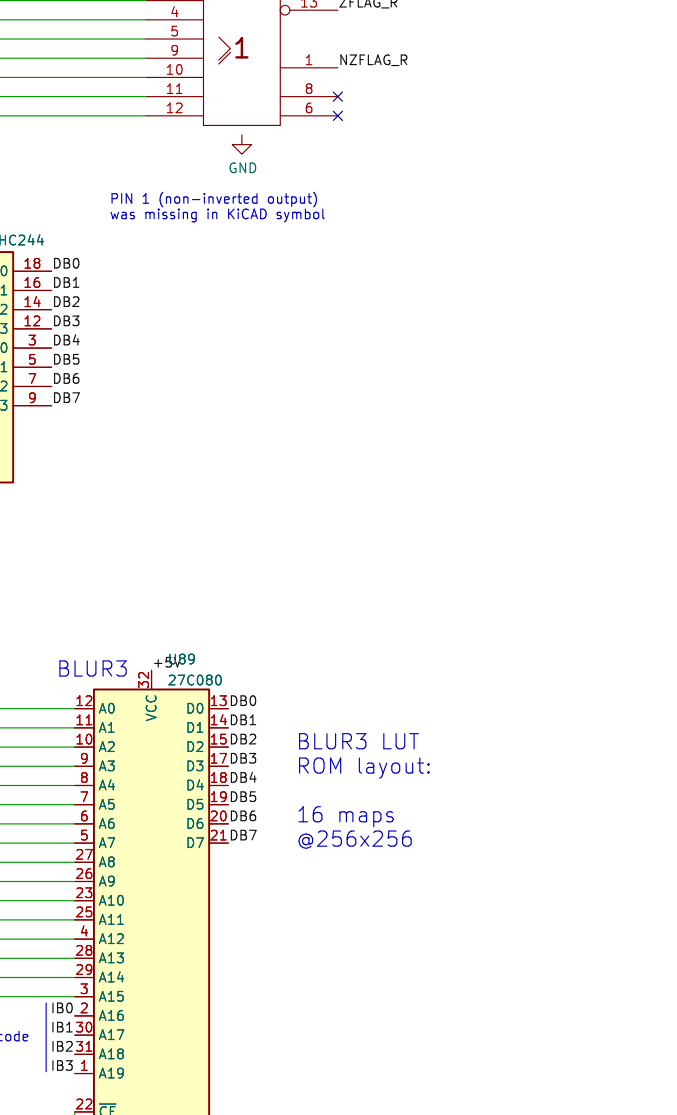
Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers

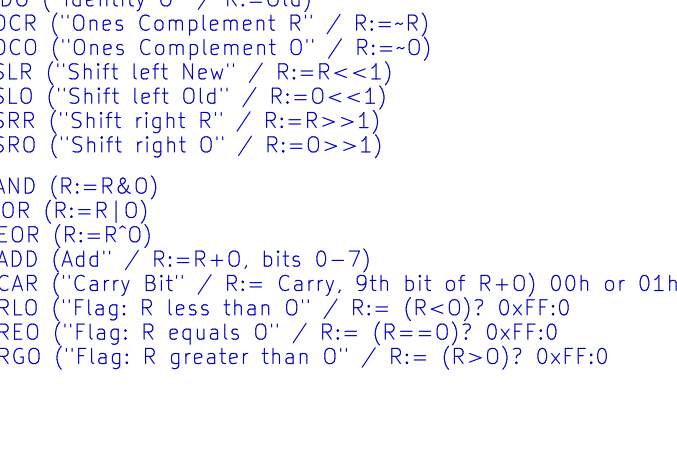


ADDER



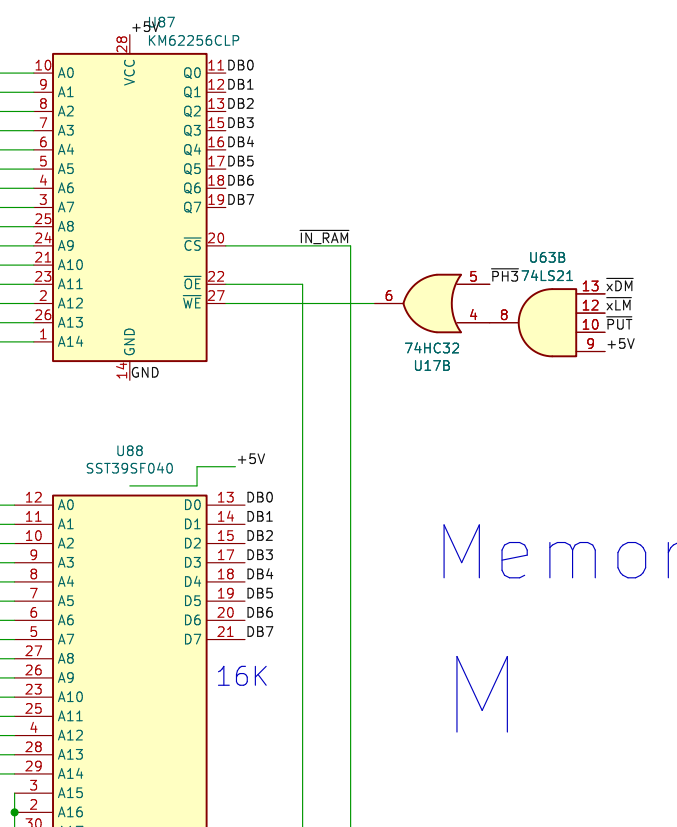
0 DIR ("identity" R / R=0)
1 DIR ("identity" O / R=0)
2 DIR ("Ones Complement" R / R=0)
3 DIR ("Ones Complement" O / R=0)
4 SLR ("Shift left New / R=0<<1")
5 SLR ("Shift left New / R=0<<1")
6 SRR ("Shift right R / R=0>>1")
7 SRR ("Shift right R / R=0>>1")
8 AND (R=R&O)
9 OR (R=R|O)
10 XOR (R=R^O)
11 ADD (Add / R=R+O, bits 0-7)
12 CAR ("Carry Bit" / R=Carry 8th bit of R+O 00h or 01h)
13 RLO ("Flag: R less than O" / R=(R<O)? 0xFF:0)
14 RCO ("Flag: R equals O" / R=(R==O)? 0xFF:0)
15 RGO ("Flag: R greater than O" / R=(R>O)? 0xFF:0)

BLURS LUT ROM layout



0 DIR ("identity" R / R=0)
1 DIR ("identity" O / R=0)
2 DIR ("Ones Complement" R / R=0)
3 DIR ("Ones Complement" O / R=0)
4 SLR ("Shift left New / R=0<<1")
5 SLR ("Shift left New / R=0<<1")
6 SRR ("Shift right R / R=0>>1")
7 SRR ("Shift right R / R=0>>1")
8 AND (R=R&O)
9 OR (R=R|O)
10 XOR (R=R^O)
11 ADD (Add / R=R+O, bits 0-7)
12 CAR ("Carry Bit" / R=Carry 8th bit of R+O 00h or 01h)
13 RLO ("Flag: R less than O" / R=(R<O)? 0xFF:0)
14 RCO ("Flag: R equals O" / R=(R==O)? 0xFF:0)
15 RGO ("Flag: R greater than O" / R=(R>O)? 0xFF:0)

Storage



D (DATA) Page index

When DC is executed, prevents the DS output from being updated with the new return address page index, before the instruction is done.

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers

Serial IO

Implements SPI functionality with dedicated SER/DES registers