

Pulls the RDY shared line low when the HLT control signal is active.
Resume execution when the button is pressed, and also on reset.

Sheet: /Clock/HaltResume/
File: HaltResume.sch

Title:

Size: A4	Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)	Rev: A (ad8706e6)

Id: 3/39

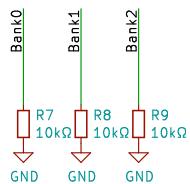
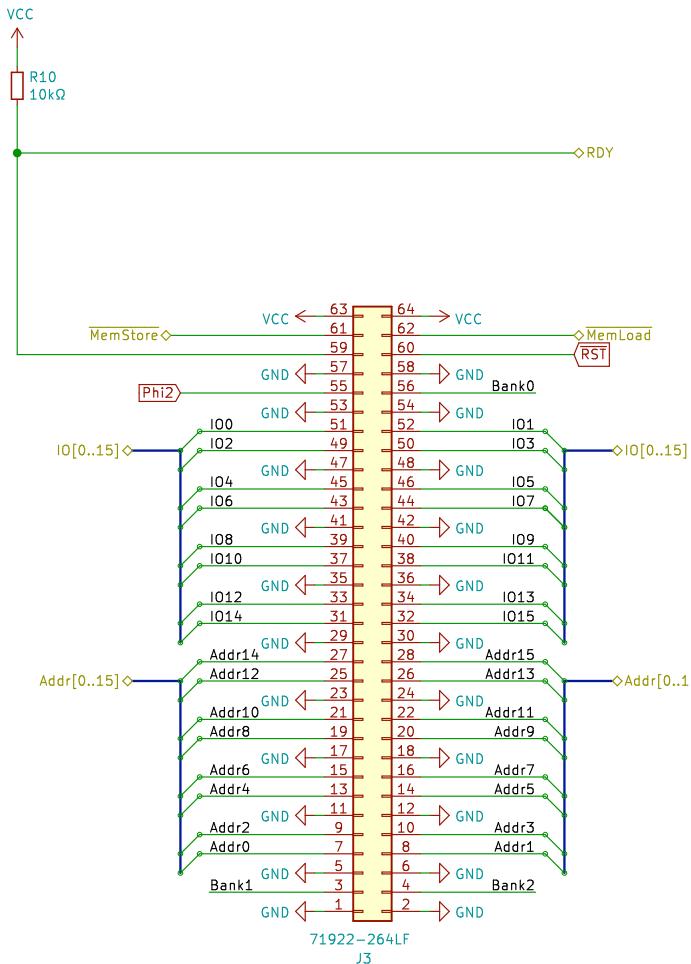
1 2 3 4 5

A

B

C

D



Peripheral devices on separate boards connect to the system bus here.

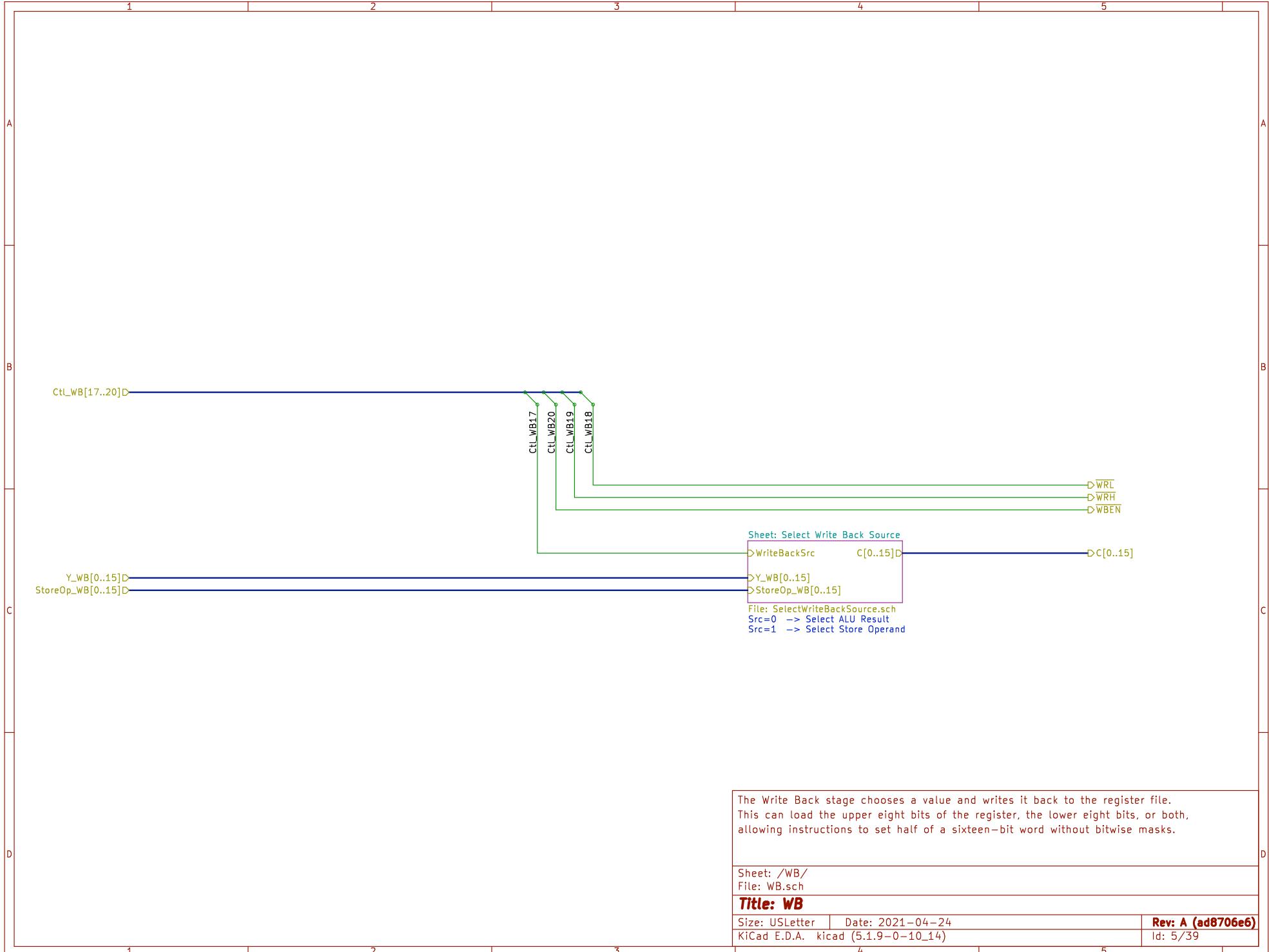
Sheet: /System Bus Connector/
File: System Bus Connector.sch

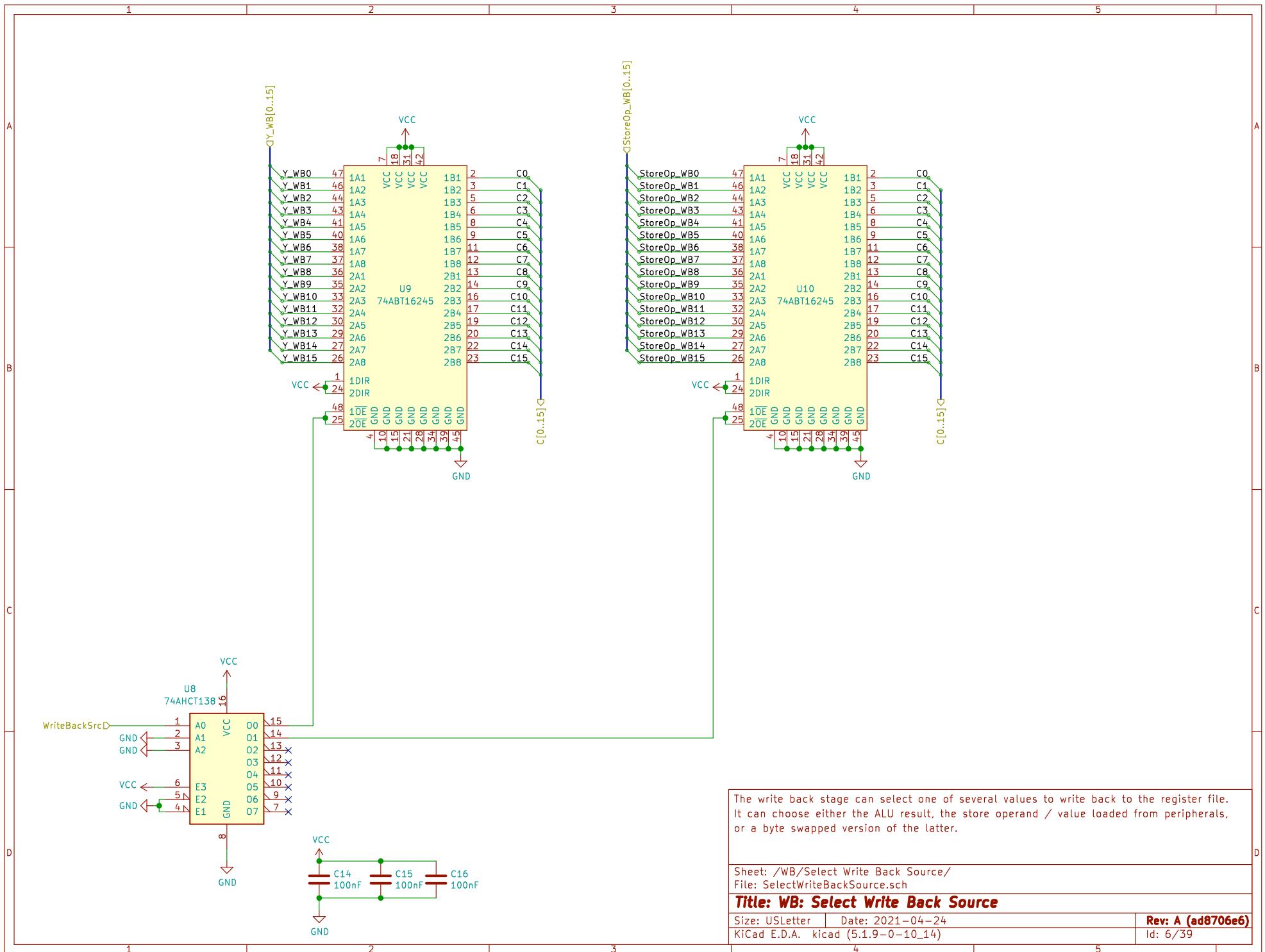
Title: System Bus Connector

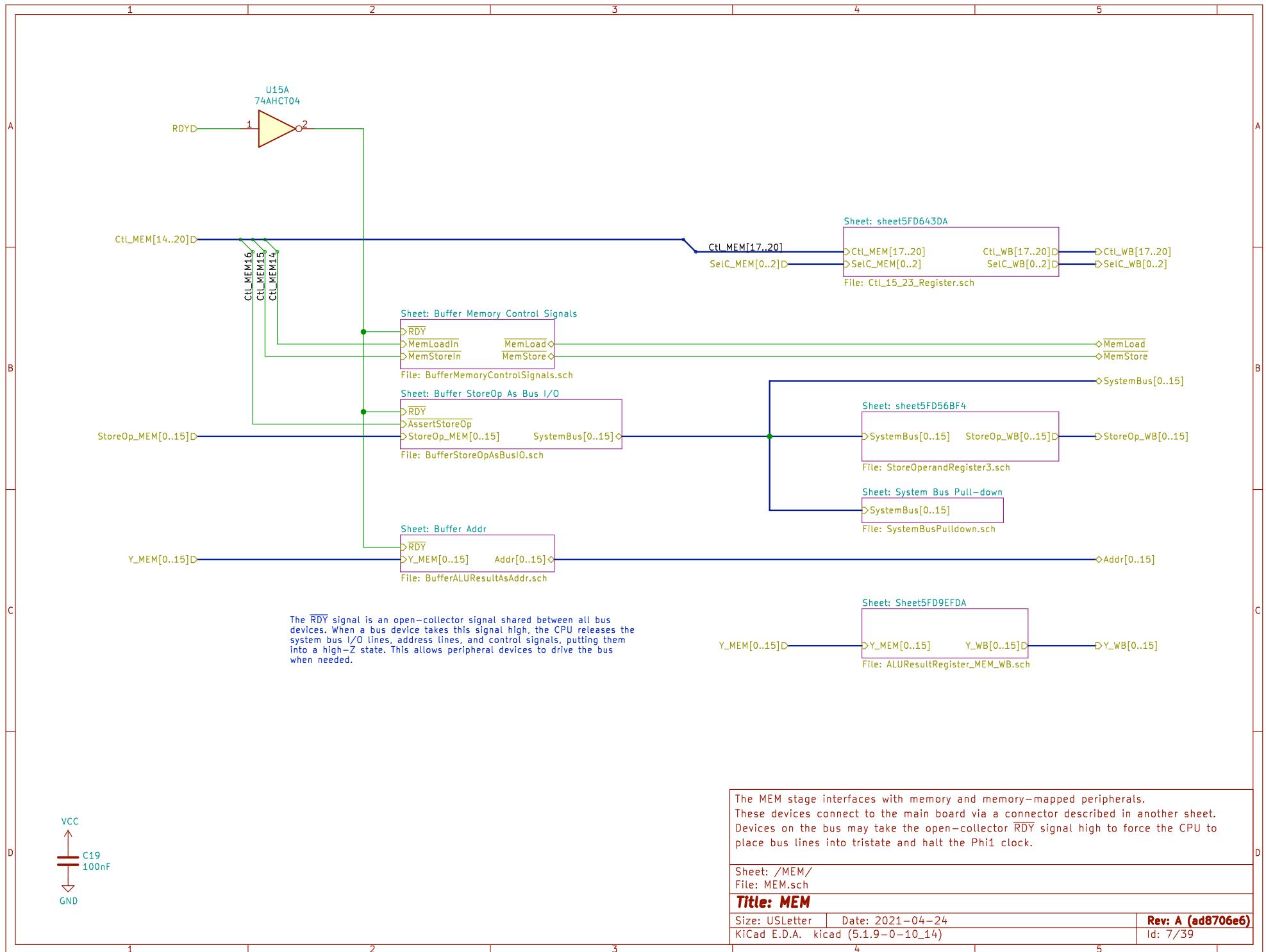
Size: USLetter | Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)

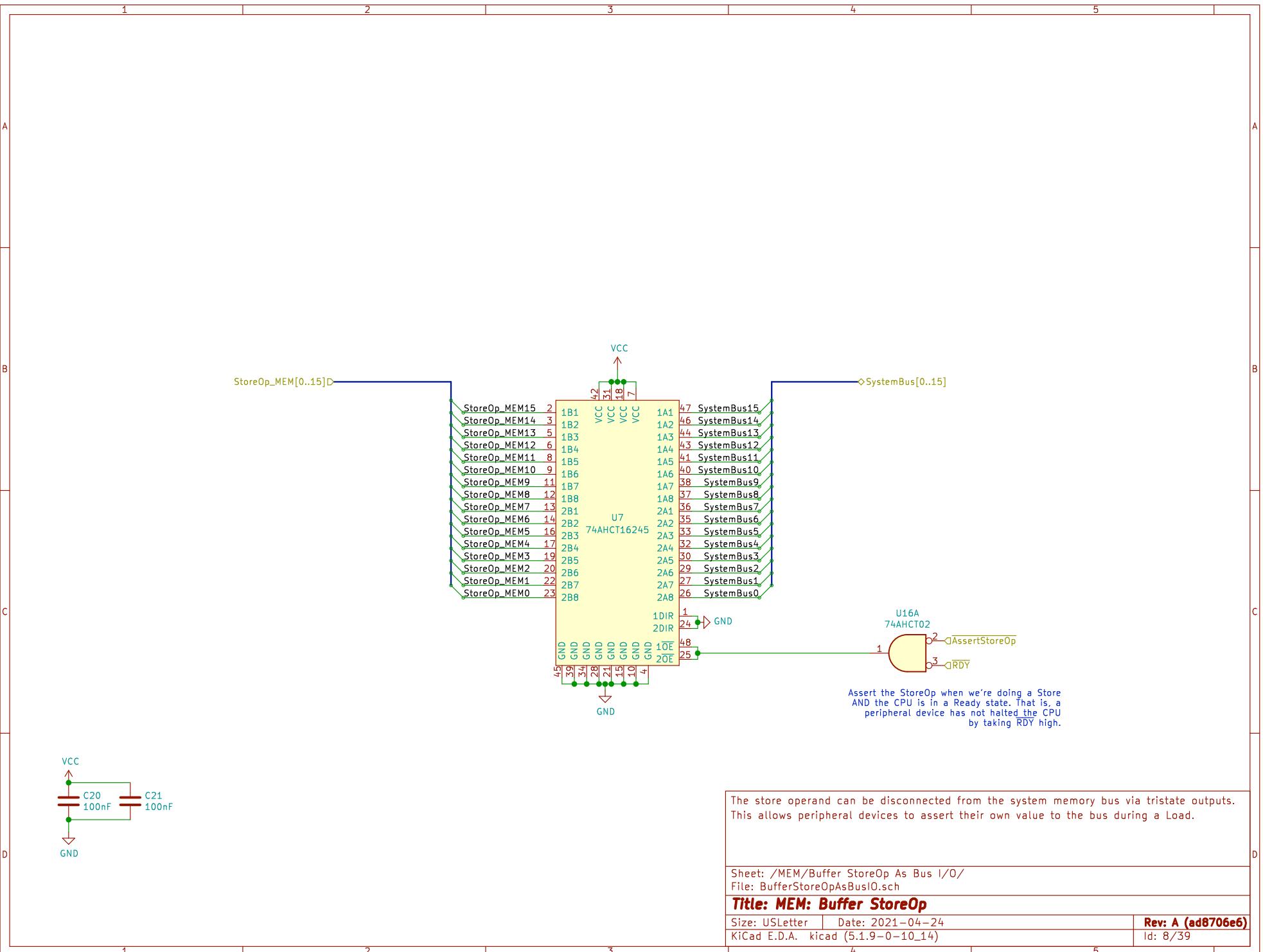
Rev: A (ad8706e6)
Id: 4/39

1 2 3 4 5









A

A

B

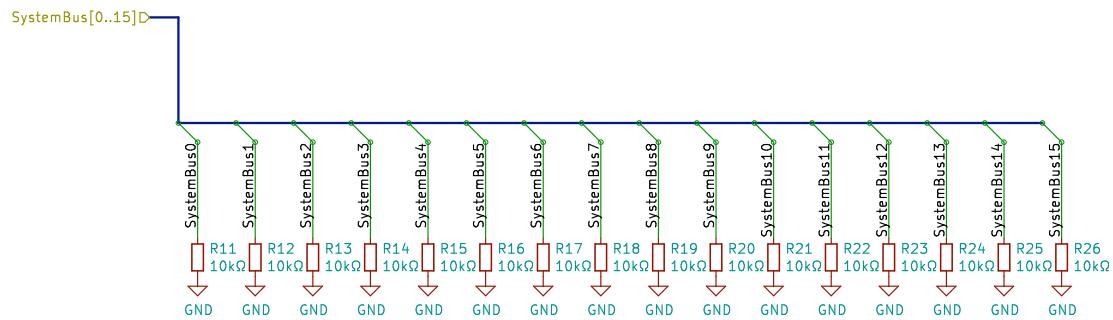
B

C

C

D

D



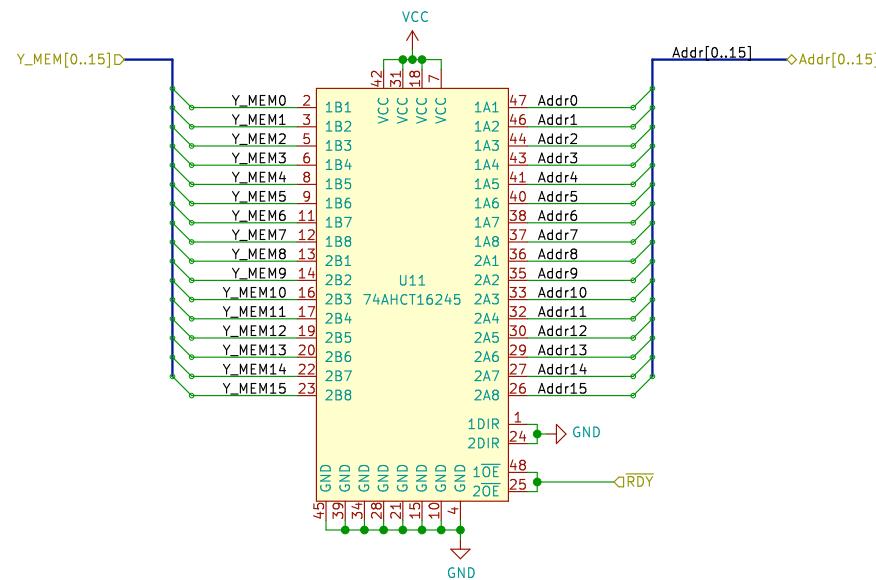
The bus needs pull-down resistors since these lines may otherwise float. sometimes.

Sheet: /MEM/System Bus Pull-down/
File: SystemBusPulldown.sch

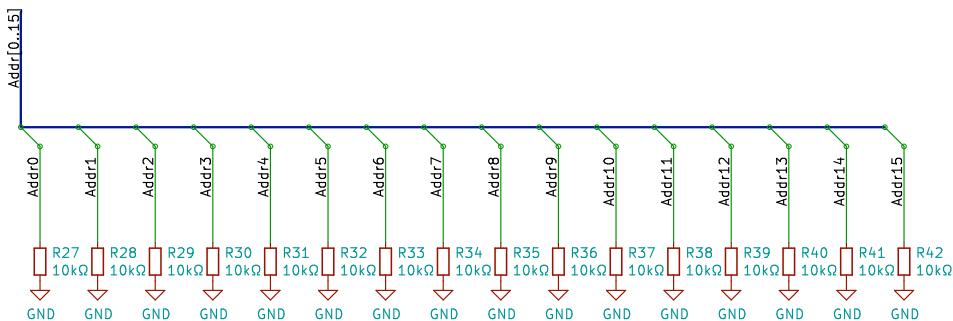
Title: MEM: System Bus Pull-down Resistors

Size: USLetter Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)

Rev: A (ad8706e6)
Id: 9/39



VCC
C23
100nF
GND



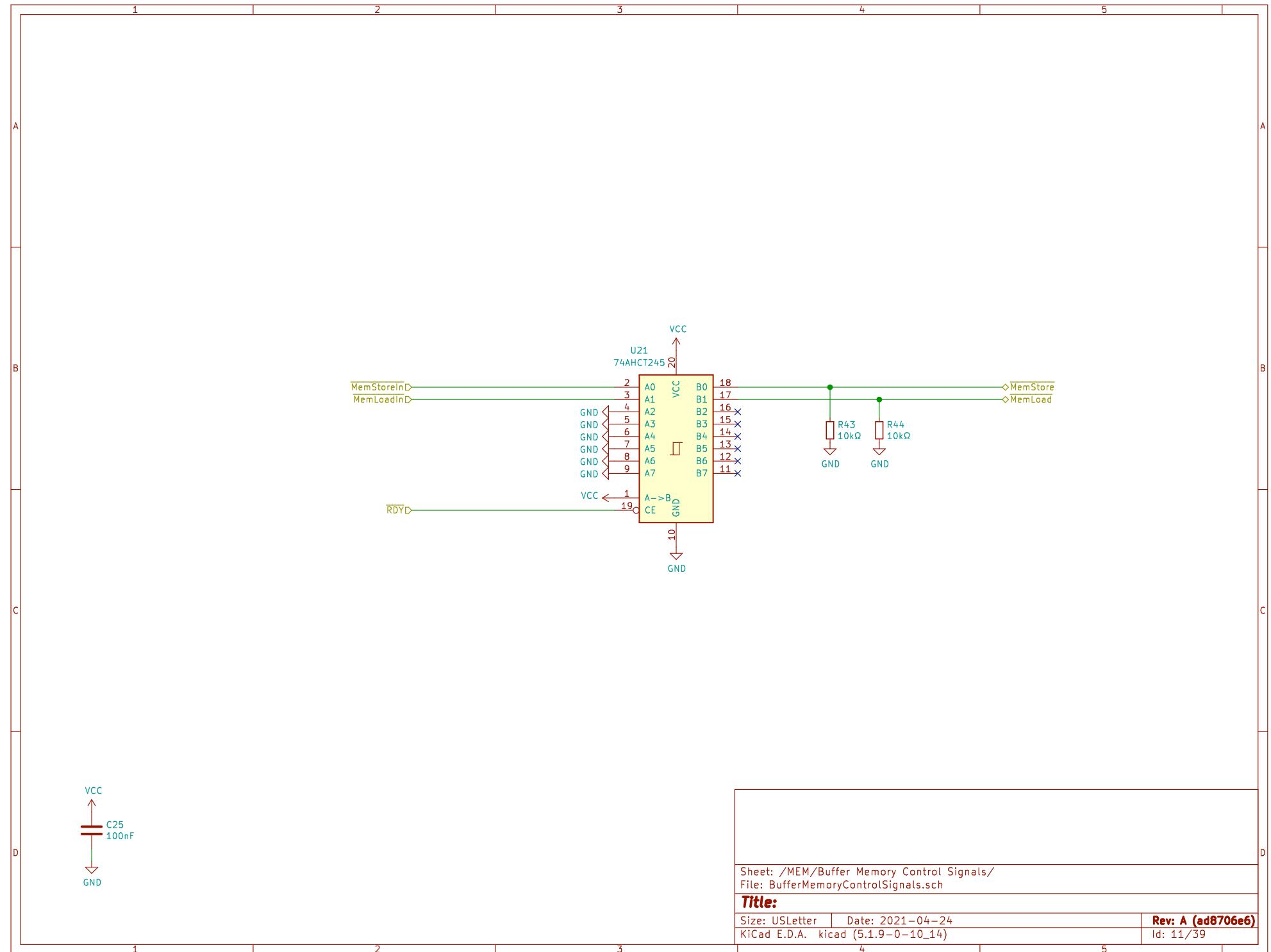
Buffer the effective address before it leaves the main board.

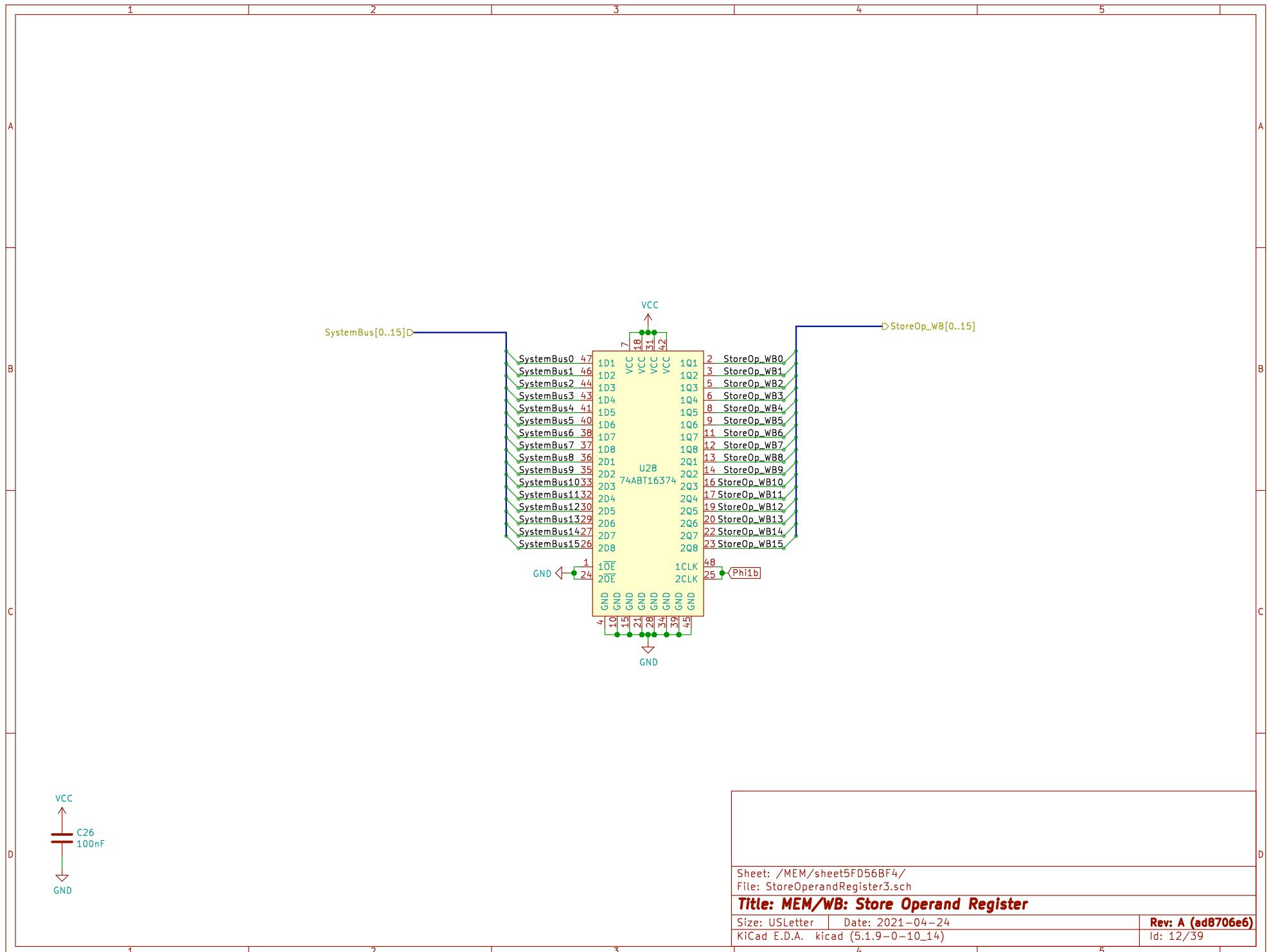
Sheet: /MEM/Buffer Addr/
File: BufferALUREsultAsAddr.sch

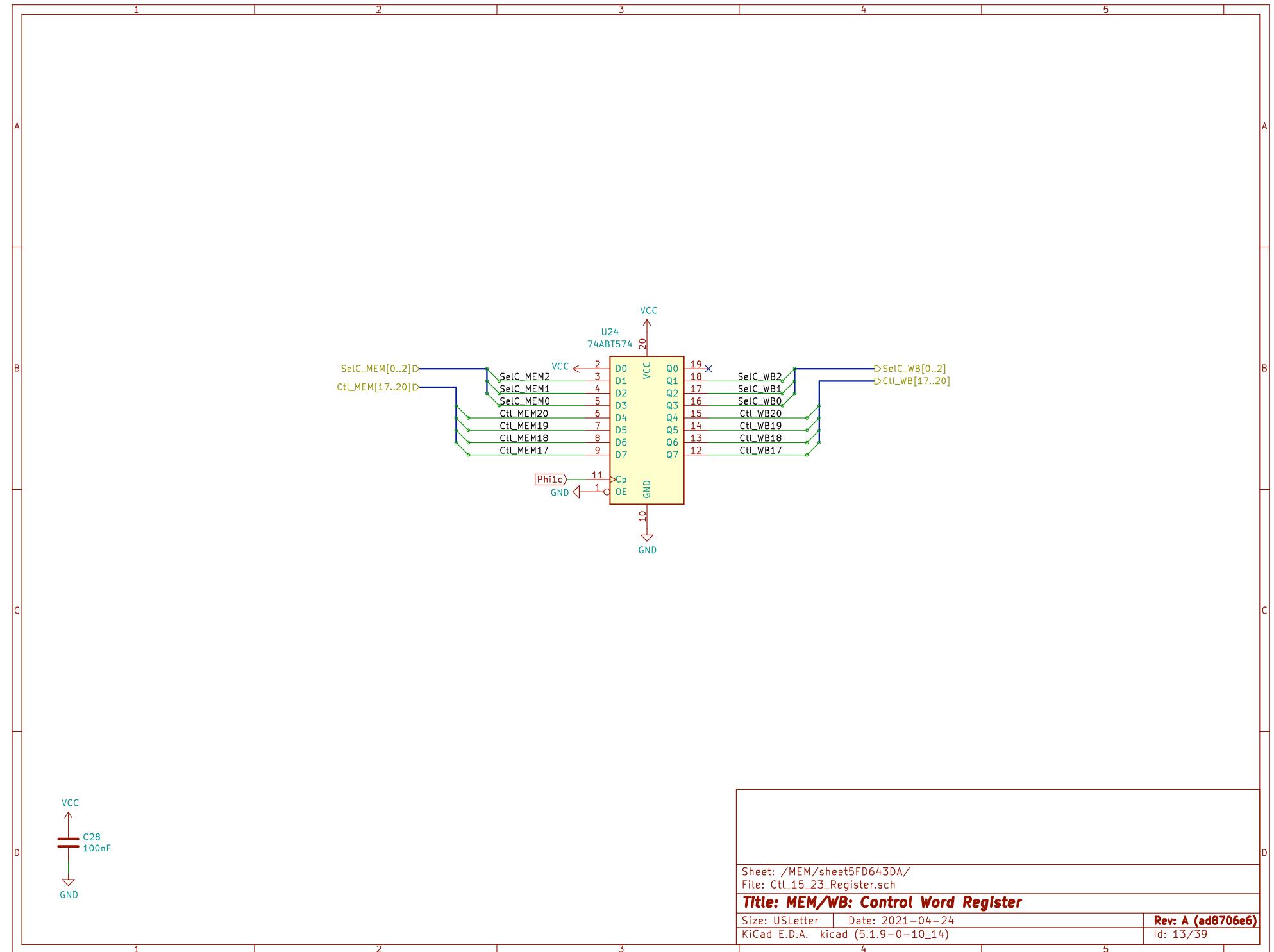
Title: MEM: Buffer ALUResult

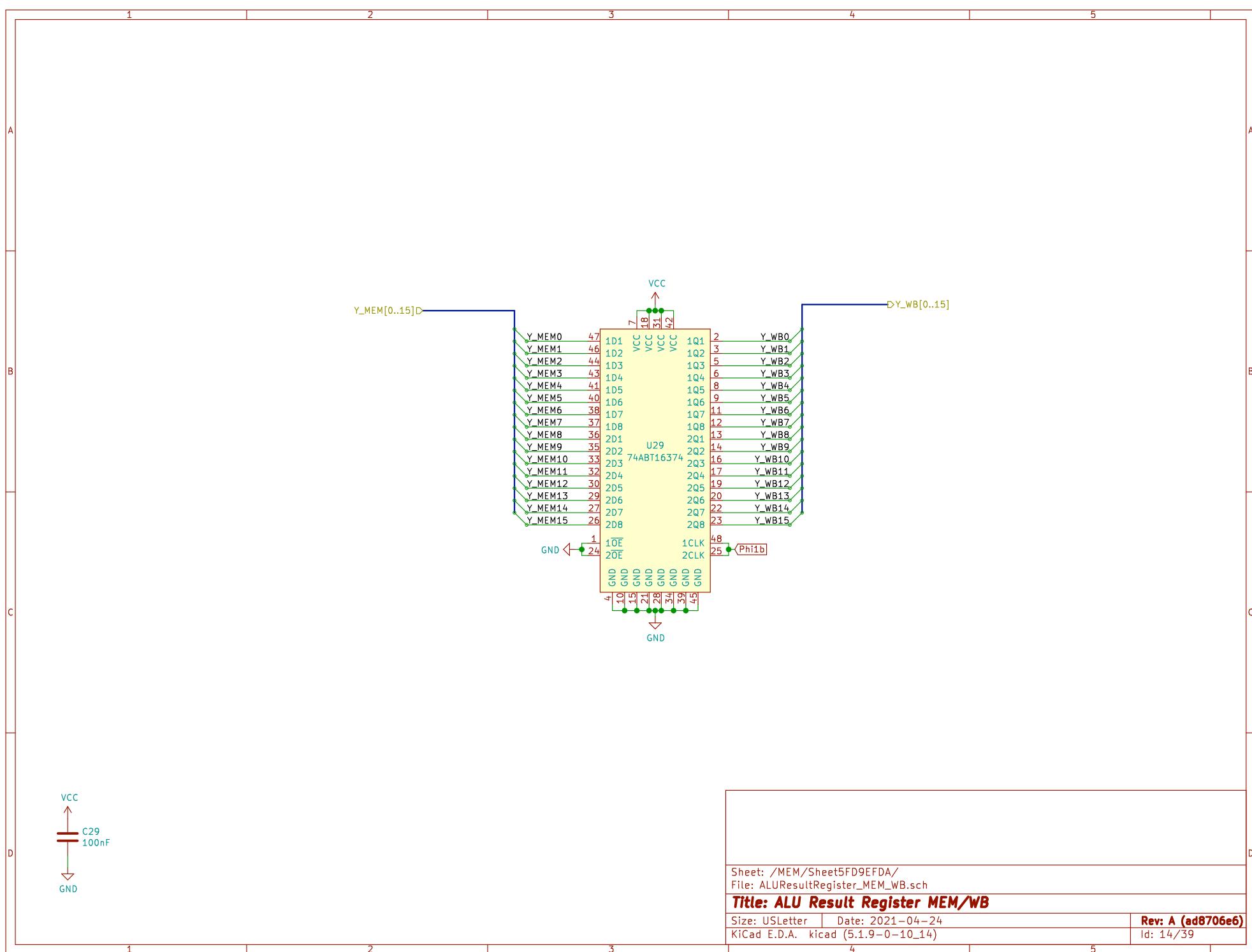
Size: USLetter | Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)

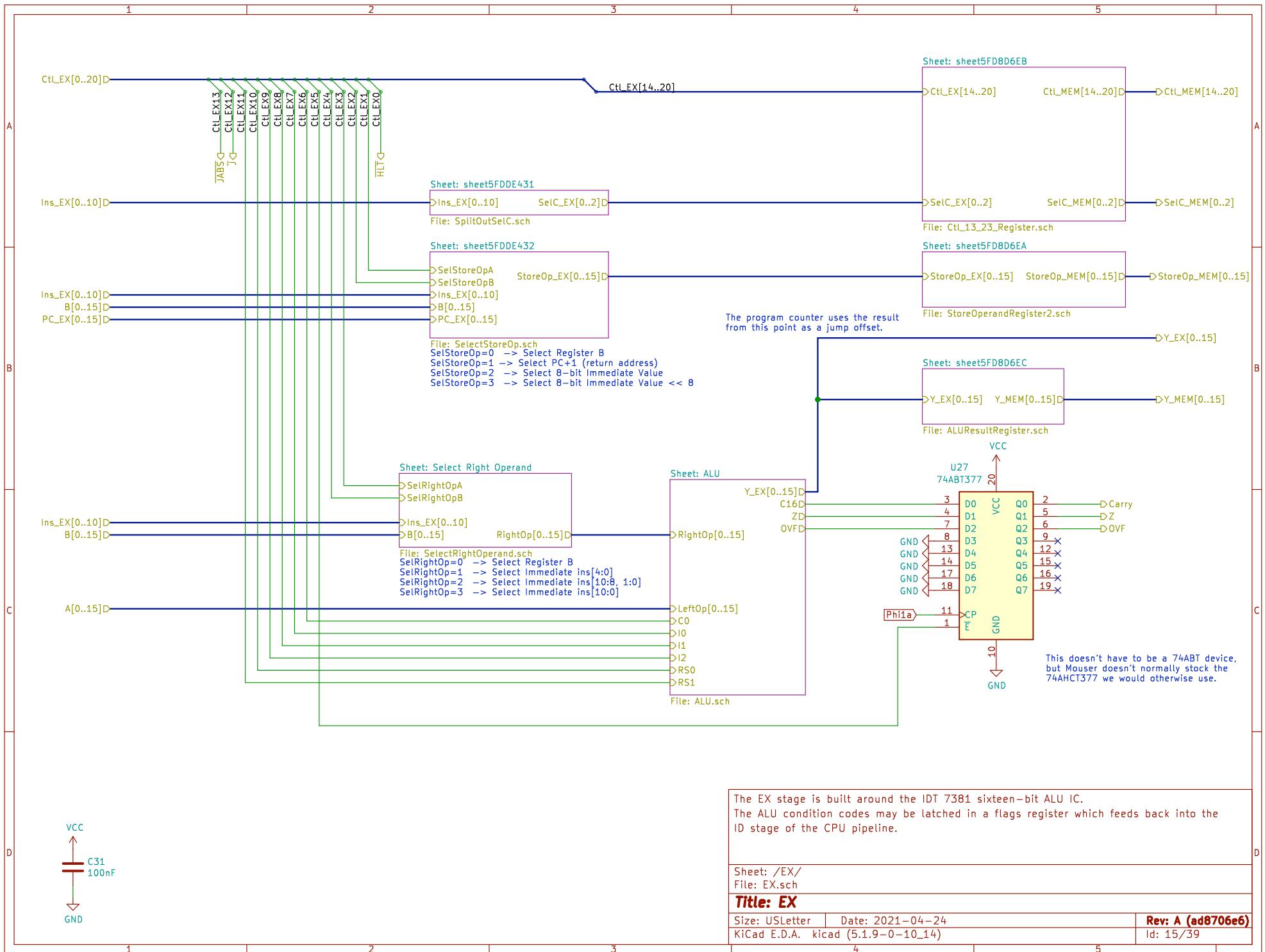
Rev: A (ad8706e6)
Id: 10/39

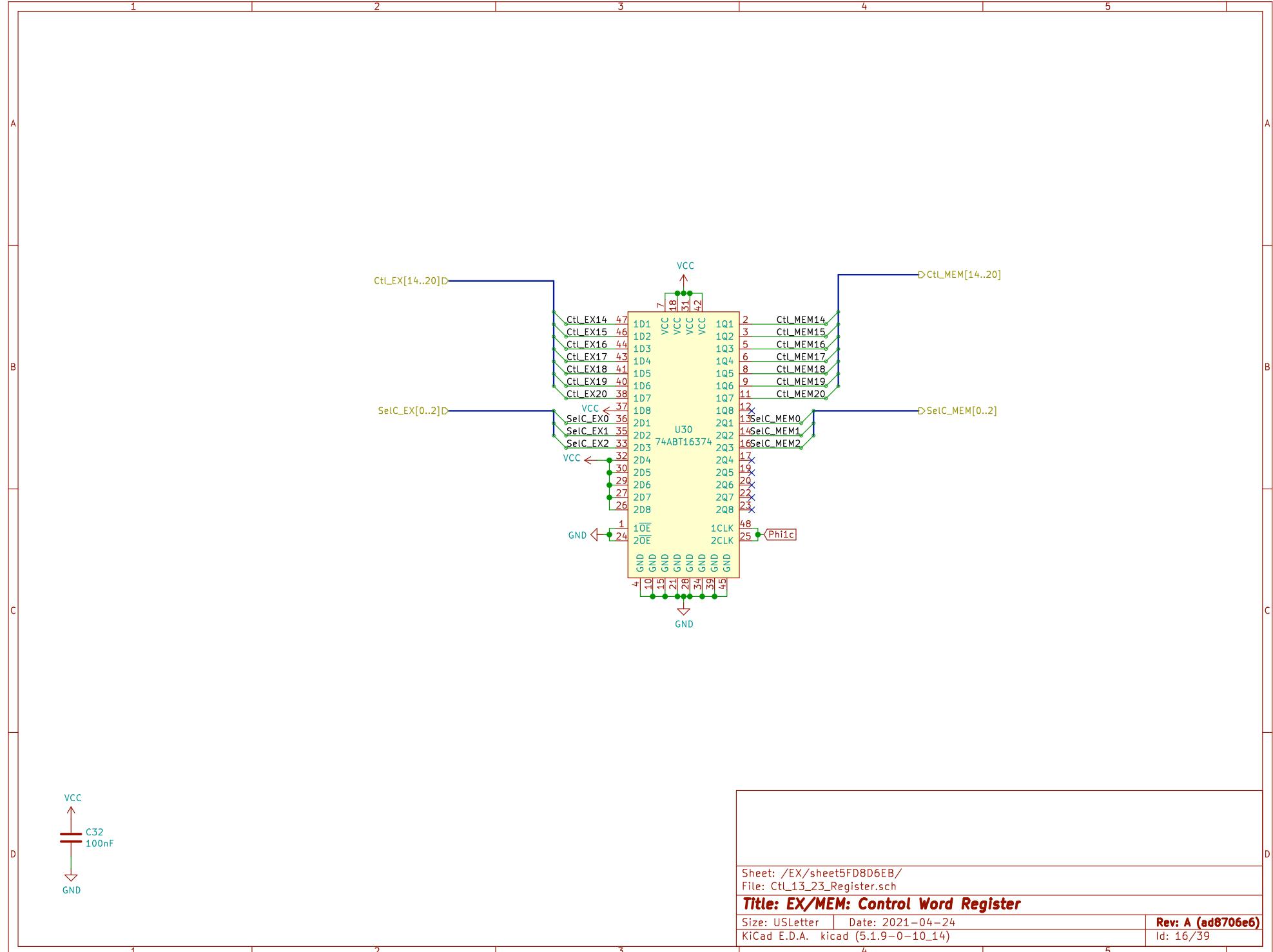


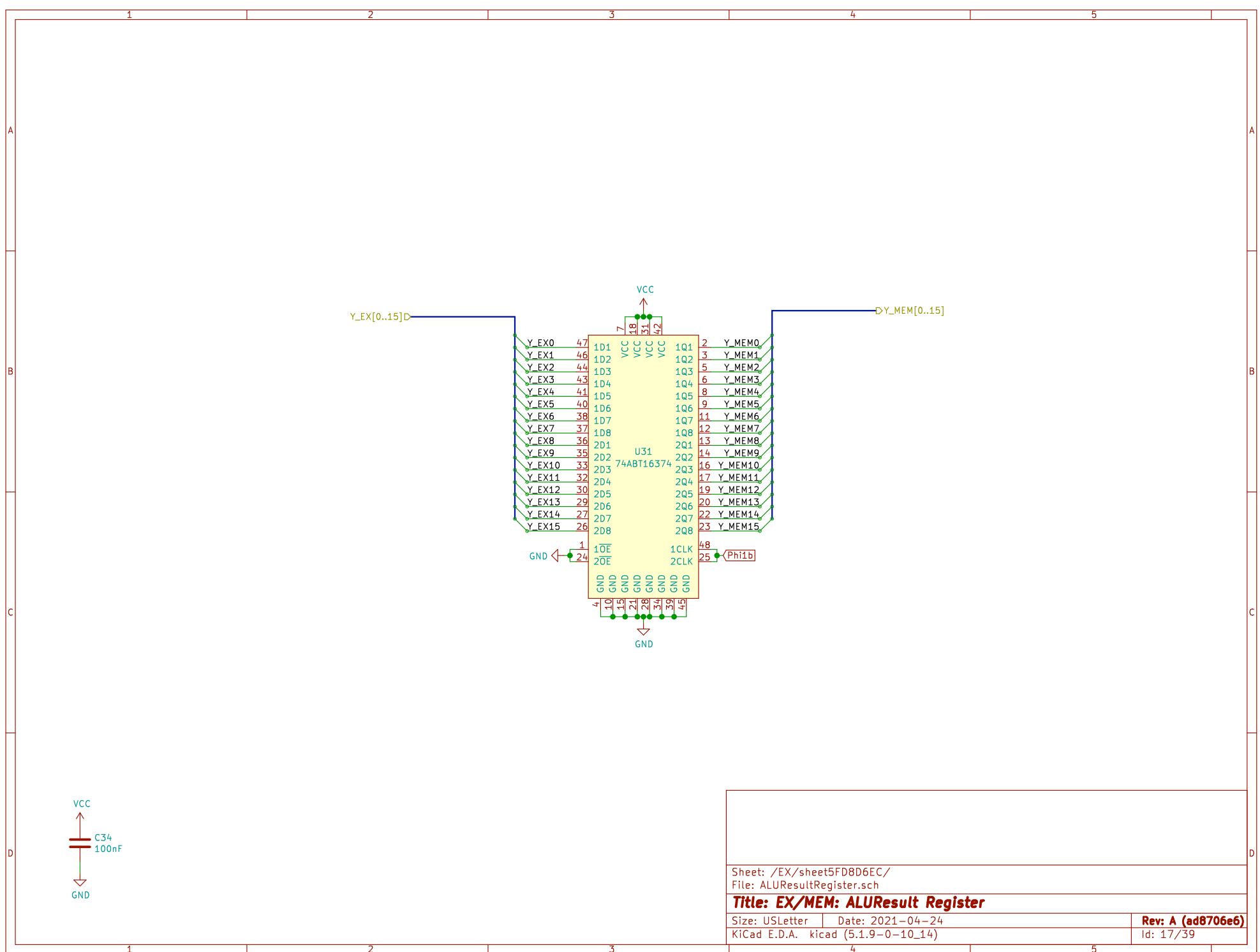












1 2 3 4 5

A

A

B

B

C

C

D

D



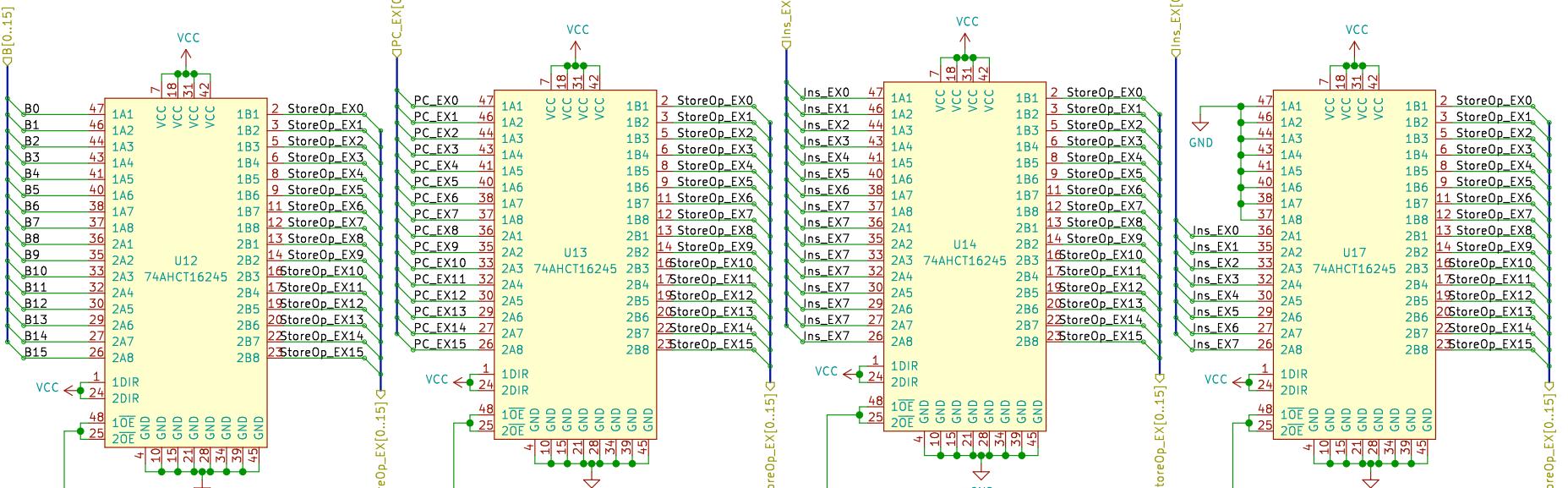
Sheet: /EX/sheet5FDDE431/
File: SplitOutSelC.sch

Title:

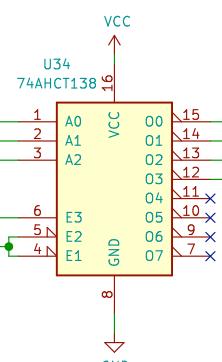
Size: USLetter | Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)

Rev: A (ad8706e6)
Id: 18/39

1 2 3 4 5



SelStoreOp=0 → Select Register B
 SelStoreOp=1 → Select PC+1 (return address)
 SelStoreOp=2 → Select 8-bit Immediate Value
 SelStoreOp=3 → Select 8-bit Immediate Value << 8



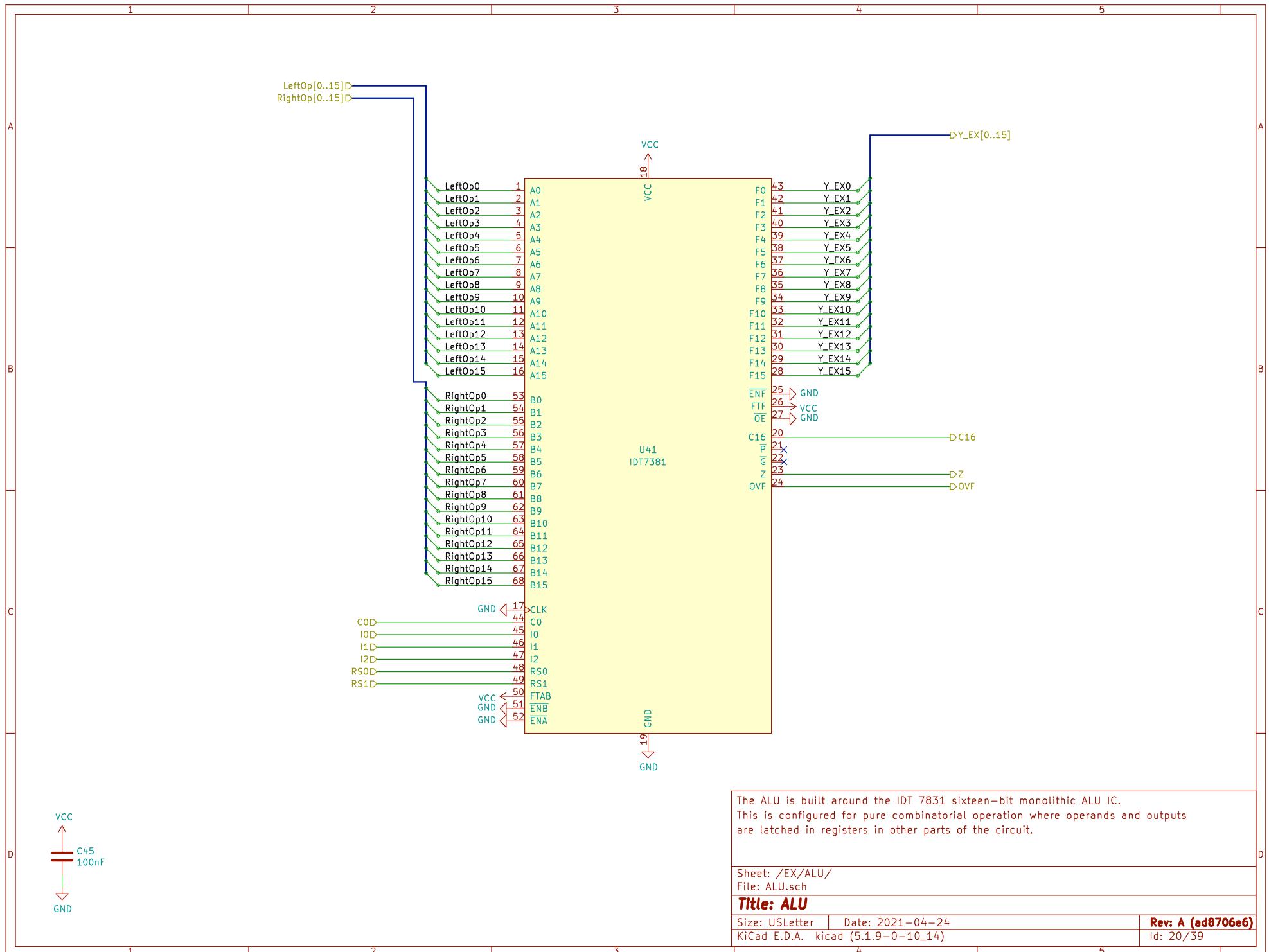
The Store Operand can be taken from either the value on port B of the register file, the program counter value, or an eight-bit immediate value.

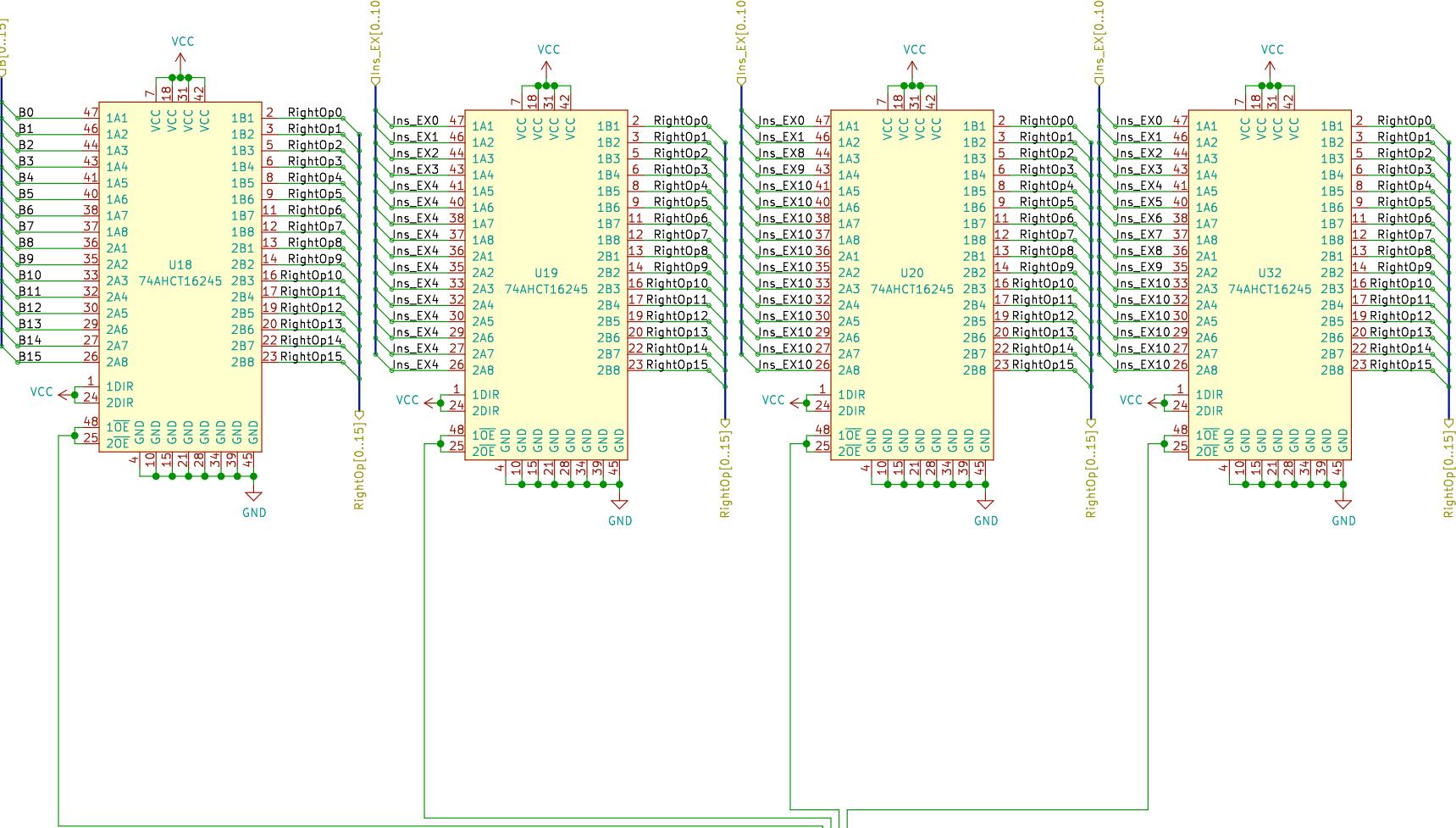
Sheet: /EX/sheet5FDDE432/
 File: SelectStoreOp.sch

Title: Select Store Operand

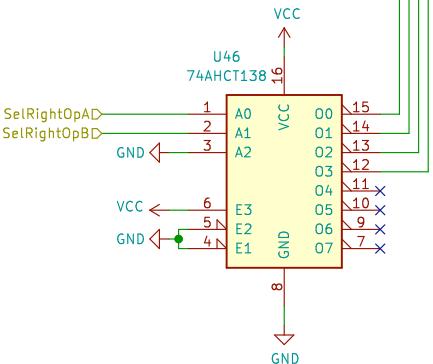
Size: USLetter Date: 2021-04-24
 KiCad E.D.A. kicad (5.1.9-0-10_14)

Rev: A (ad8706e6)
 Id: 19/39





$SelRightOp=0$ -> Select Register B
 $SelRightOp=1$ -> Select Immediate ins[4:0]
 $SelRightOp=2$ -> Select Immediate ins[10:8, 1:0]
 $SelRightOp=3$ -> Select Immediate ins[10:0]



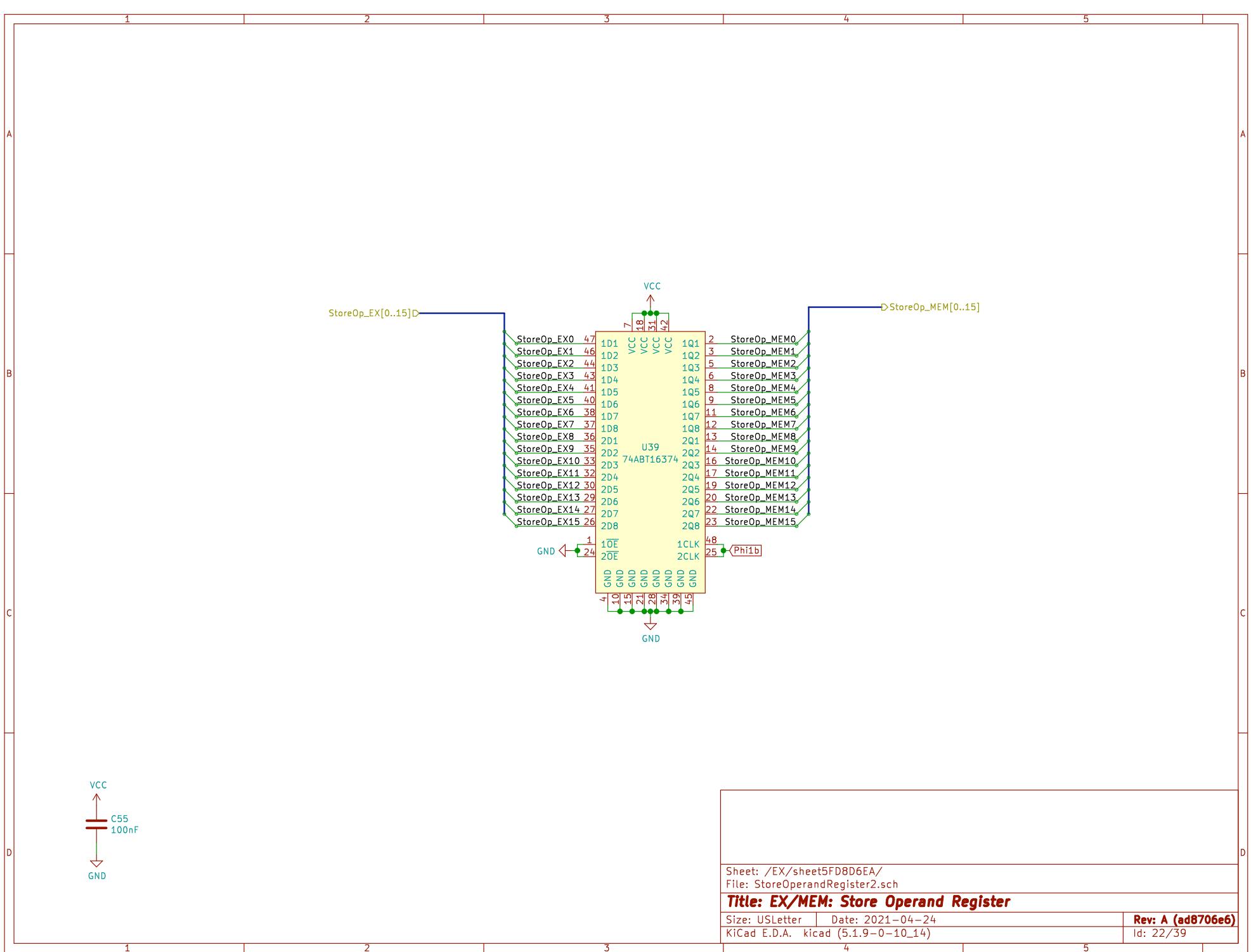
Select the right operand to the ALU
The right operand can either take the value from the B port of the register file, or from an immediate value contained in the instruction word.

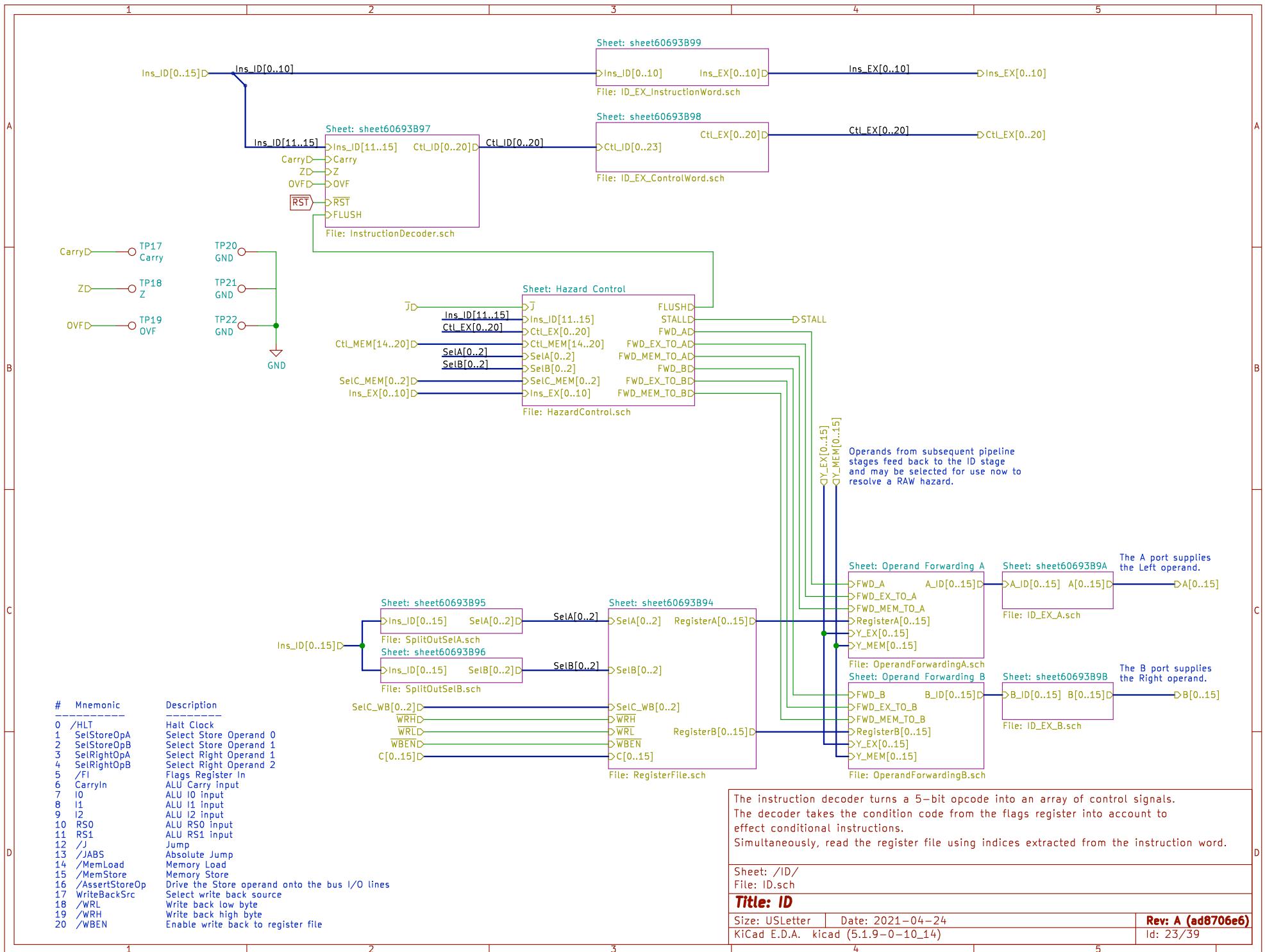
Sheet: /EX/Select Right Operand/
File: SelectRightOperand.sch

Title: Select Right Operand

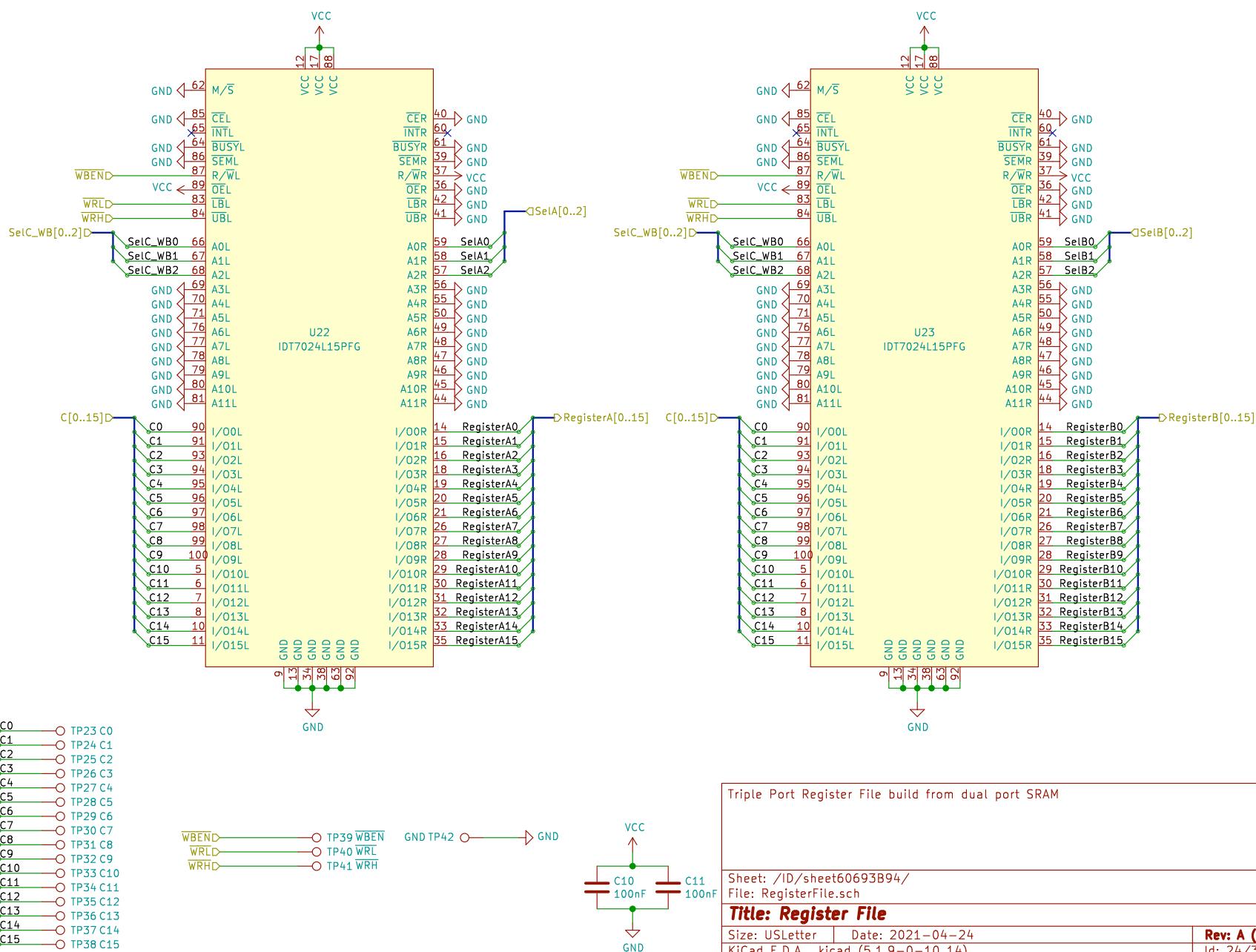
Size: USLetter Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)

Rev: A (ad8706e6)
Id: 21/39





Both dual port SRAMs are configured in Slave mode. This disables the on board contention arbitration logic. Per application note, AN-91, a simultaneous read and write to the same cell will flow through from one port to another after a short delay.



1 2 3 4 5

A

A

B

B

C

C

D

D



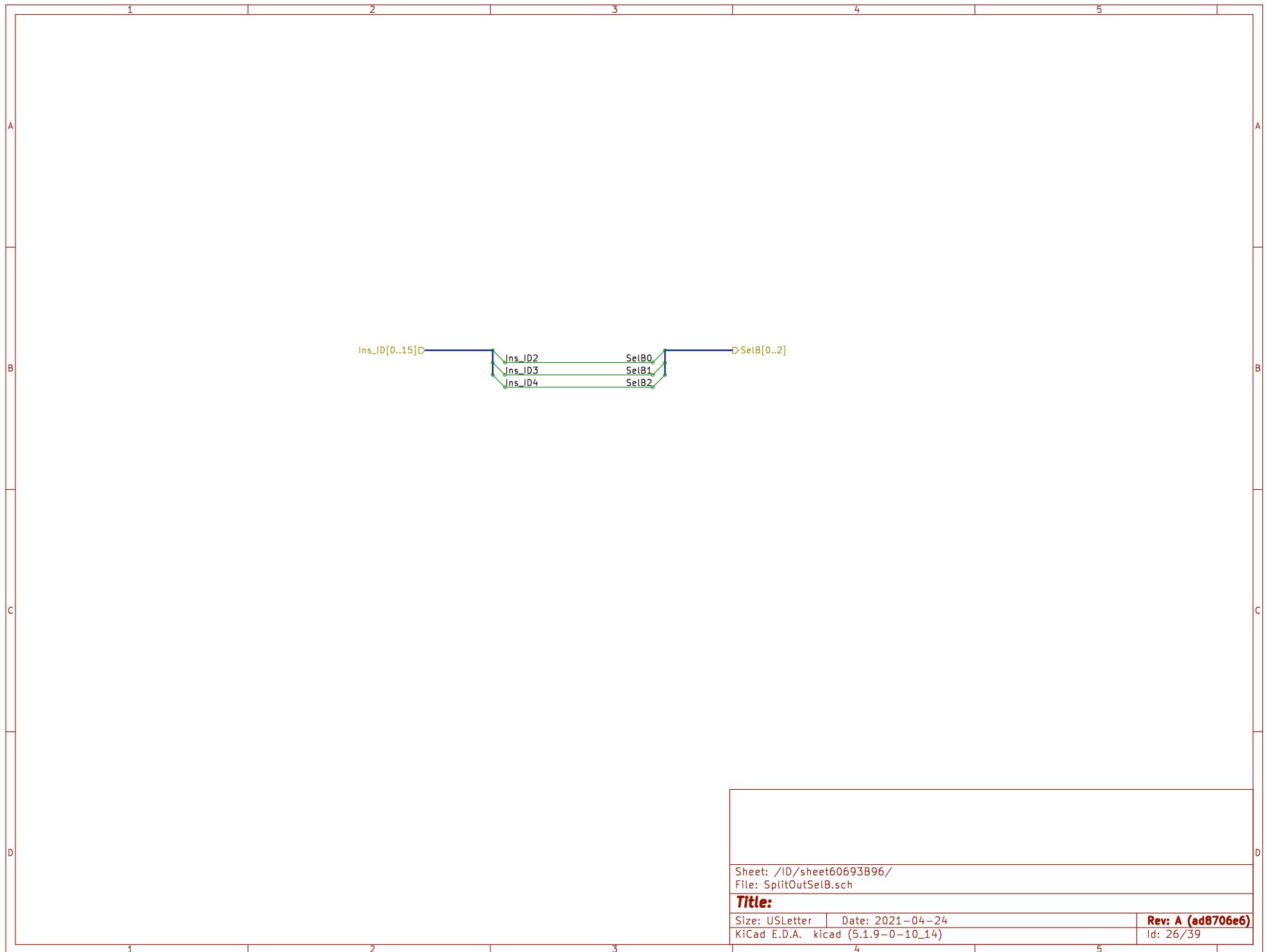
Sheet: /ID/sheet60693B95/
File: SplitOutSelA.sch

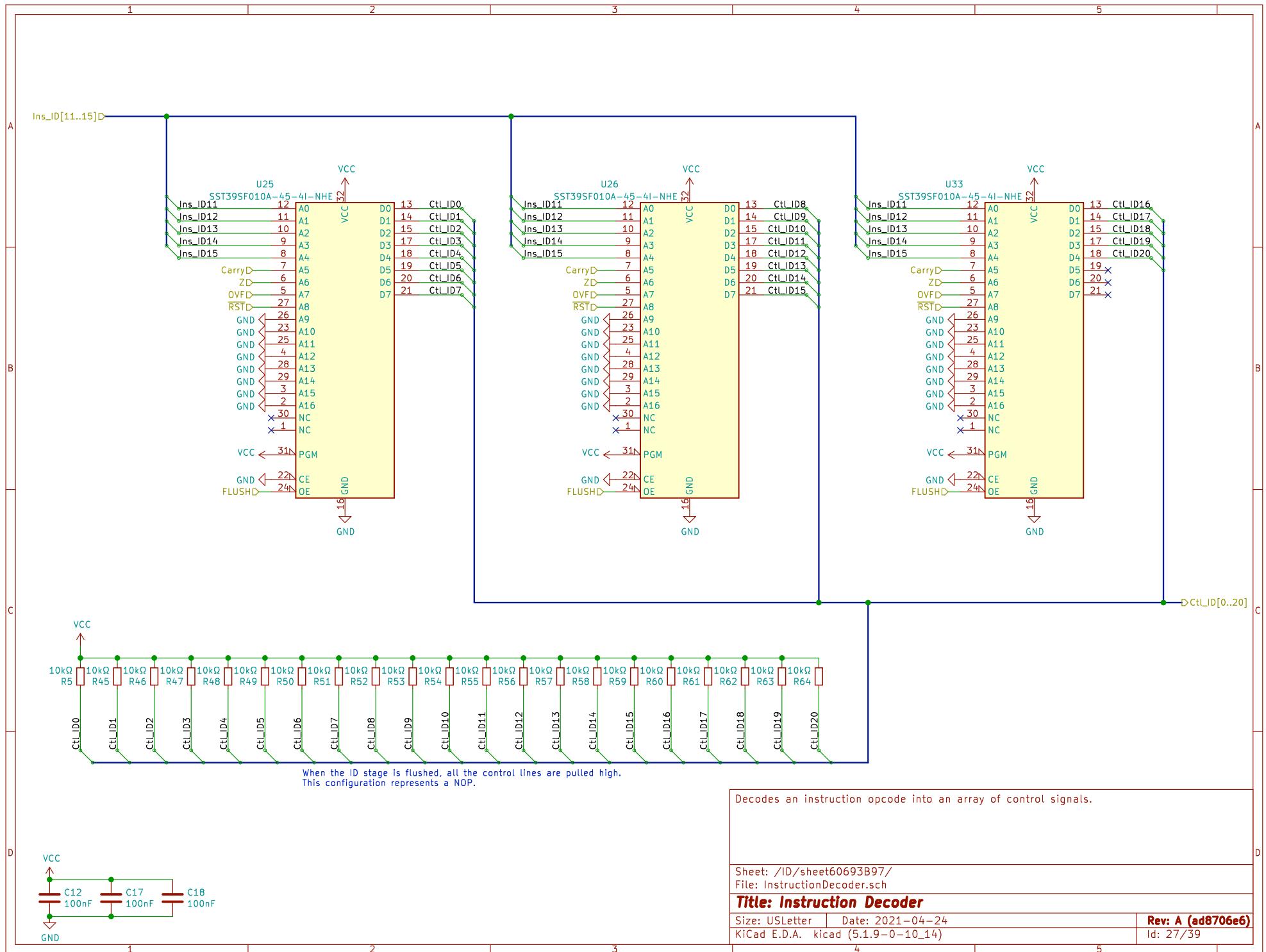
Title:

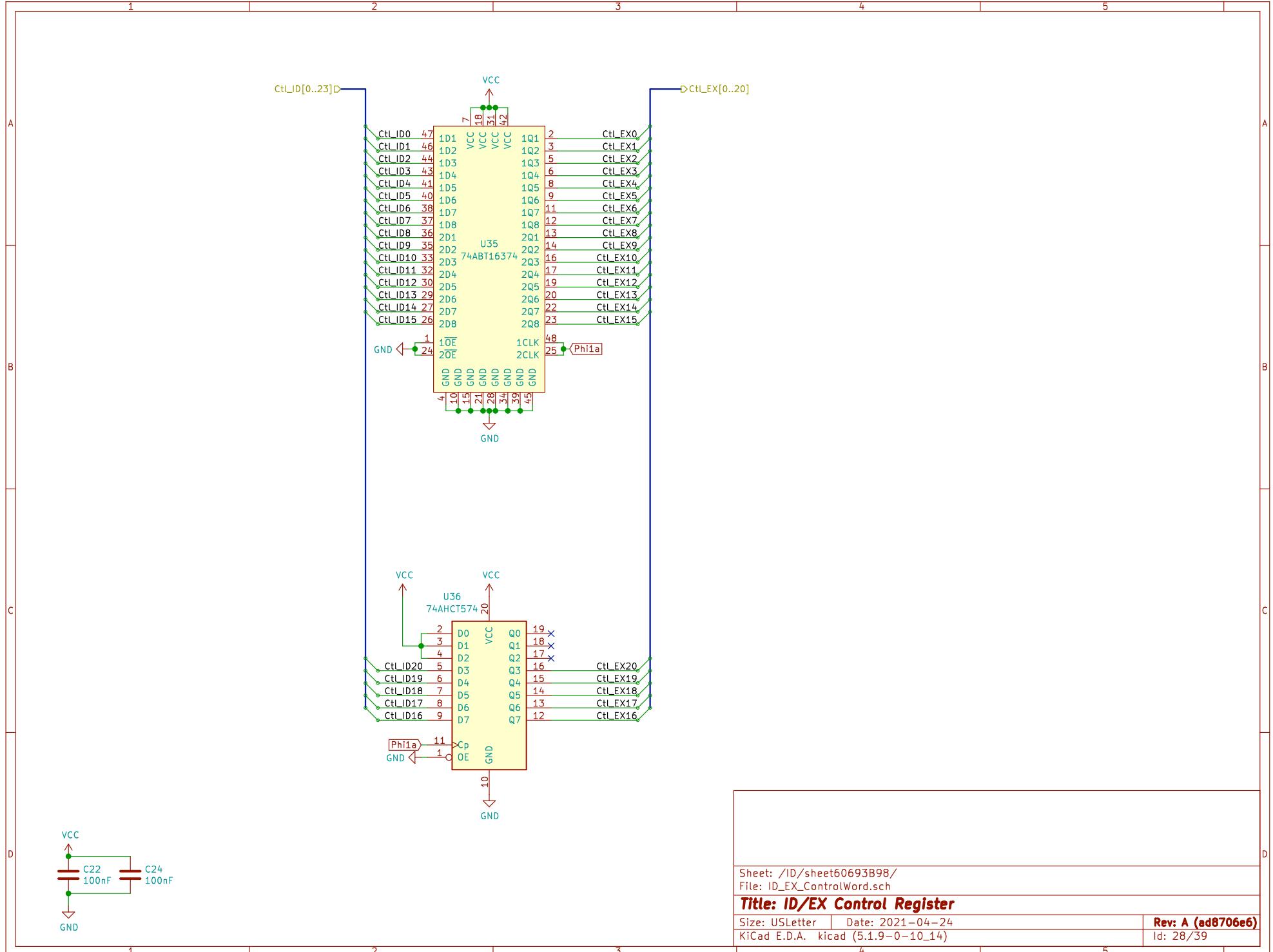
Size: USLetter | Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)

Rev: A (ad8706e6)
Id: 25/39

1 2 3 4 5





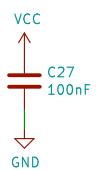
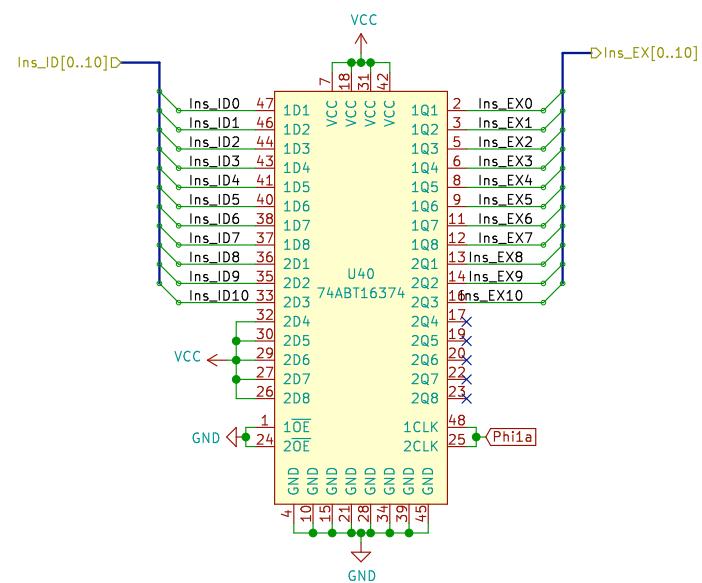


Sheet: /ID/sheet60693B98/
File: ID_EX_ControlWord.sch

Title: ID/EX Control Register

Size: USLetter Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)

Rev: A (ad8706e6)
Id: 28/39

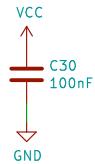
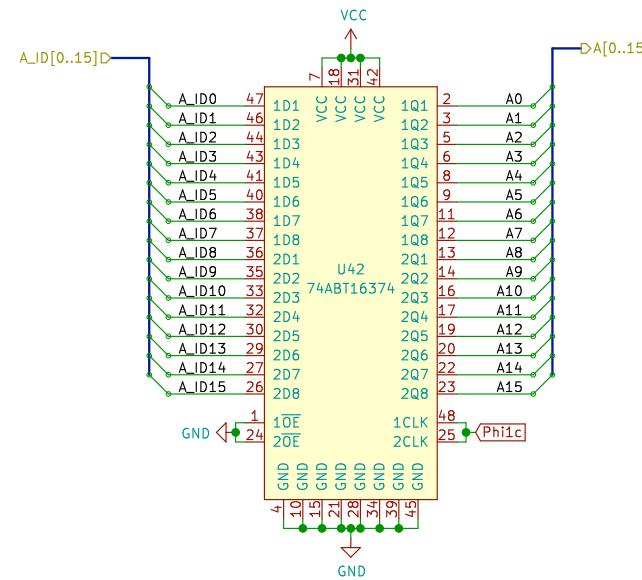


Sheet: /ID/sheet60693B99/
File: ID_EX_InstructionWord.sch

Title:

Size: USLetter | Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)

Rev: A (ad8706e6)
Id: 29/39



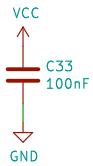
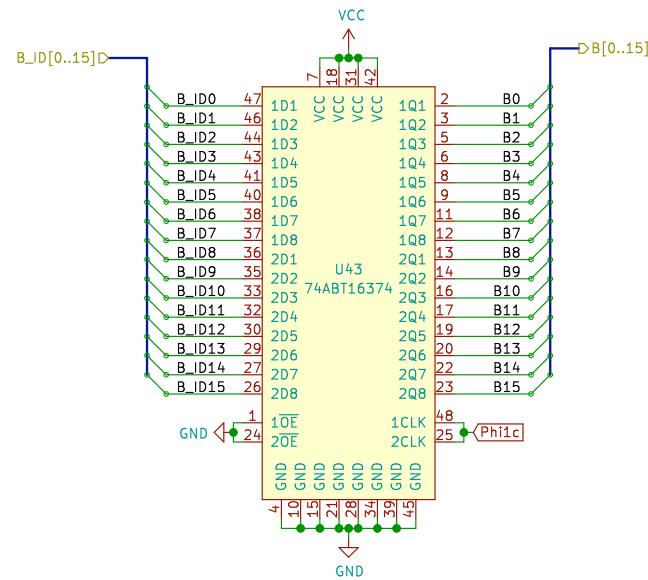
Sheet: /ID/sheet60693B9A/
File: ID_EX_A.sch

Title:

Size: USLetter | Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)

Rev: A (ad8706e6)
Id: 30/39

1 2 3 4 5



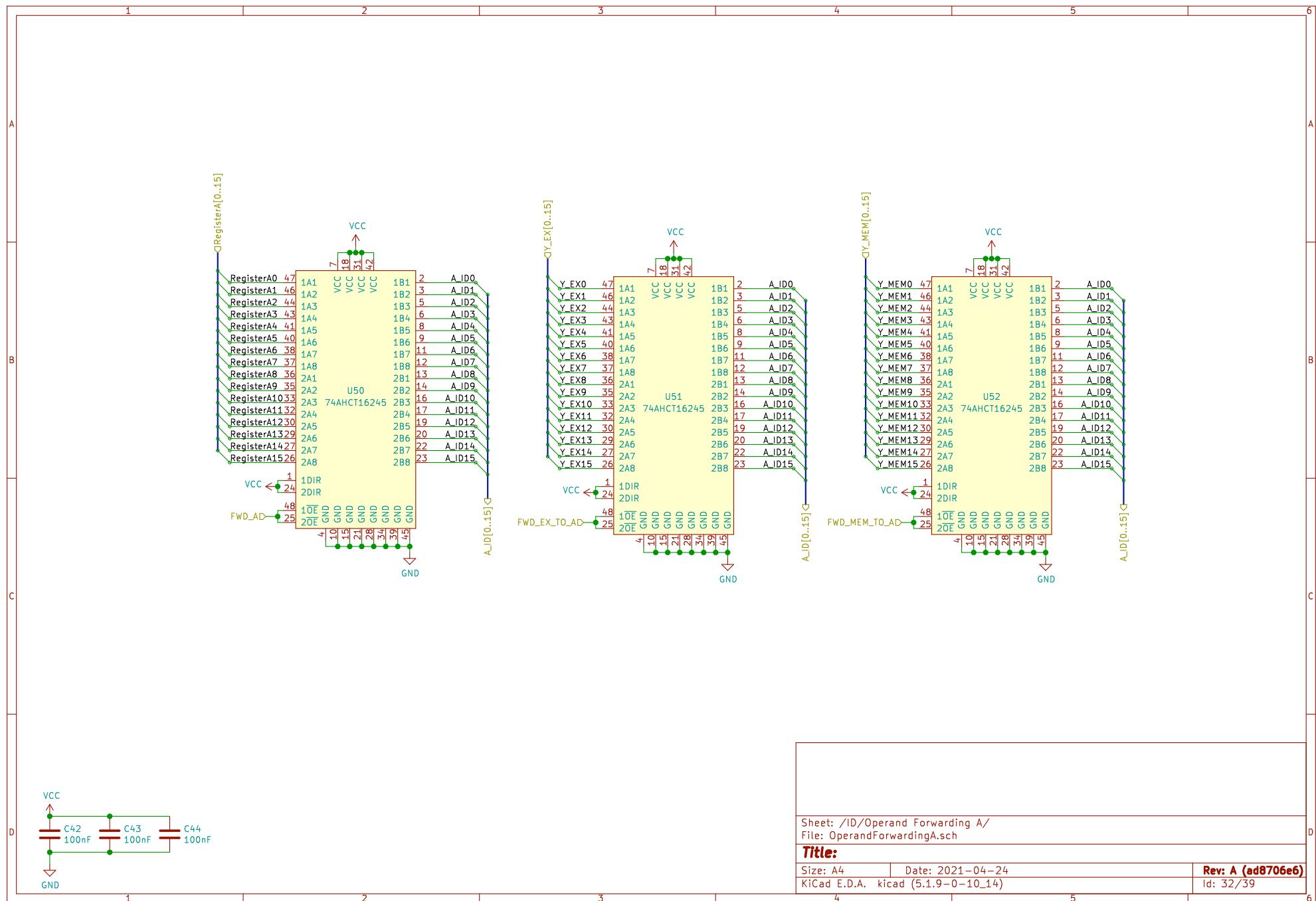
Sheet: /ID/sheet60693B9B/
File: ID_EX_B.sch

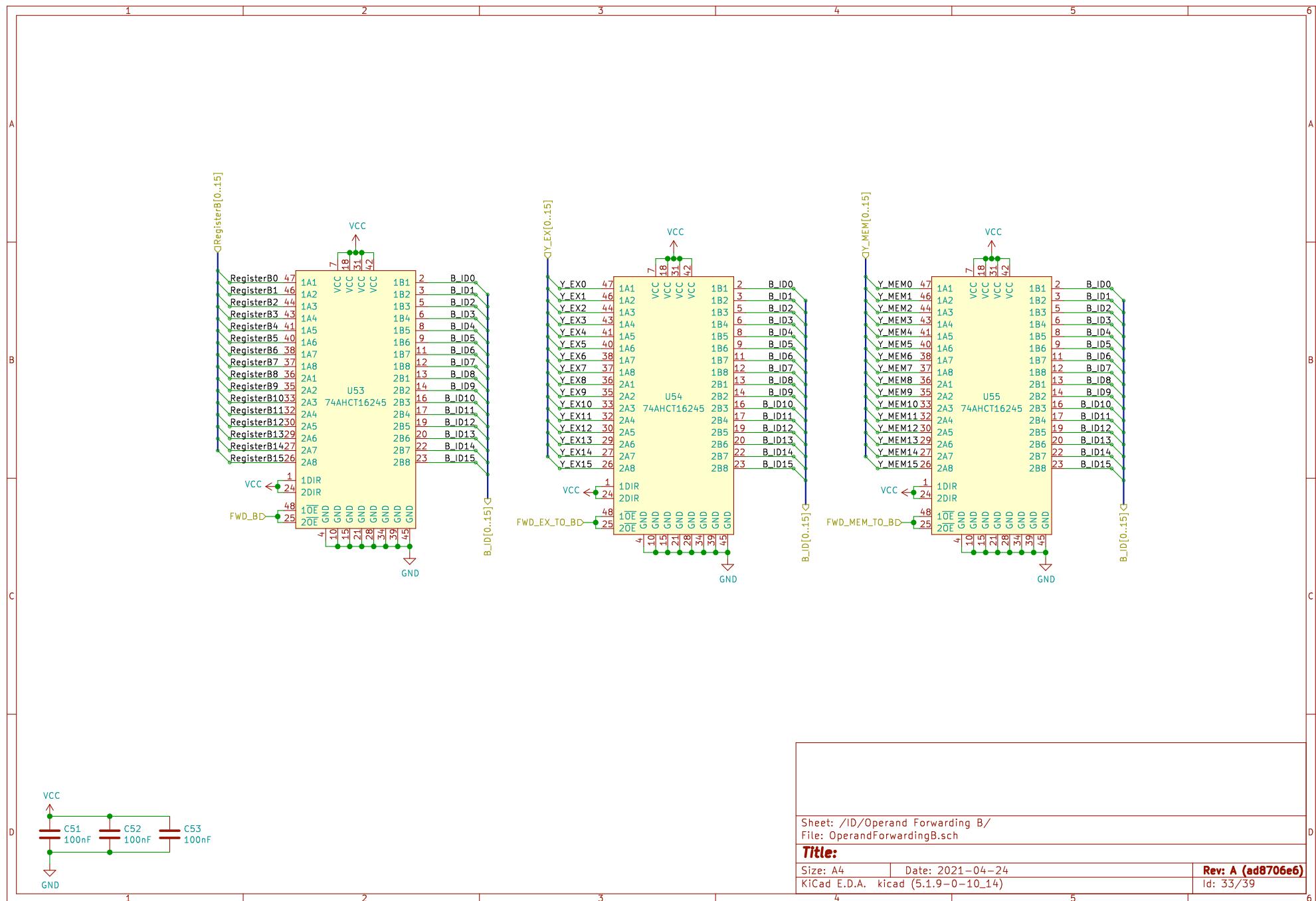
Title:

Size: USLetter | Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)

Rev: A (ad8706e6)
Id: 31/39

1 2 3 4 5





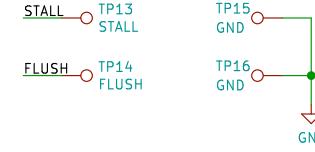
Hazard Control detects several types of hazards and resolves each one either by stalling the pipeline, or by forwarding an operand from a later pipeline stage.

On a jump, the program counter must be allowed to load a new value. At the same time, the ID and IF stages must both be flushed.

If the instruction in EX wants to update the Flags register, and the opcode in ID is determined to be one that wants to make use of the flags, then there is a Flags hazard. In this case, stall the pipeline for one cycle.

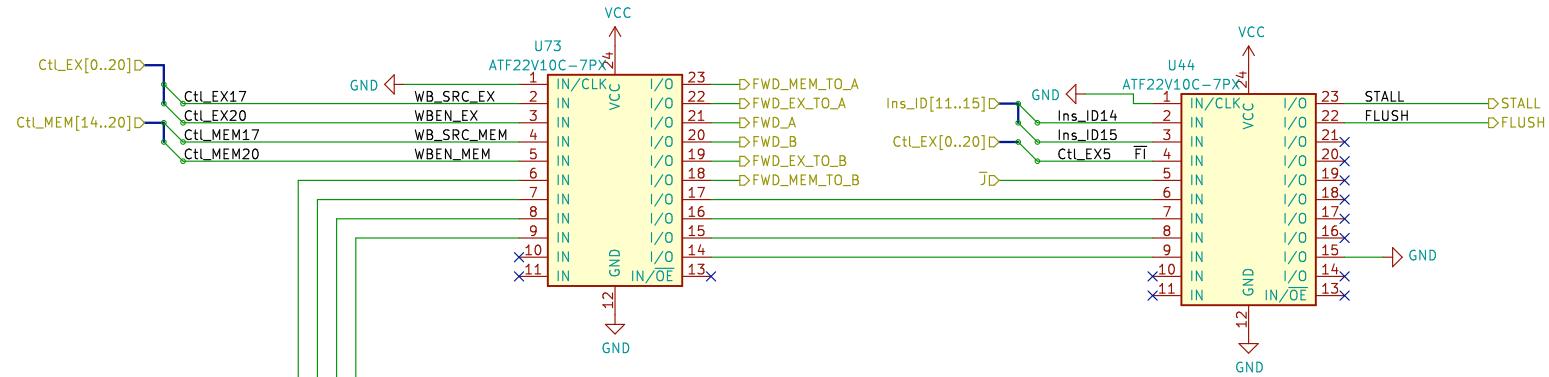
If the Ra or Rb registers refer to the destination register in the EX or MEM stage, and the instructions in the EX or MEM stage want to write back to that register, then there is a RAW error. In this case, forward an operand from a later pipeline stage to supply the EX stage on the next clock cycle.

There's no path to forward the store operand. Cases involving this operand must be resolved by stalling the pipeline.



A

A



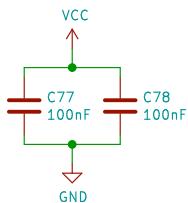
B

B



C

C



D

D

Control logic for dealing with pipeline hazards

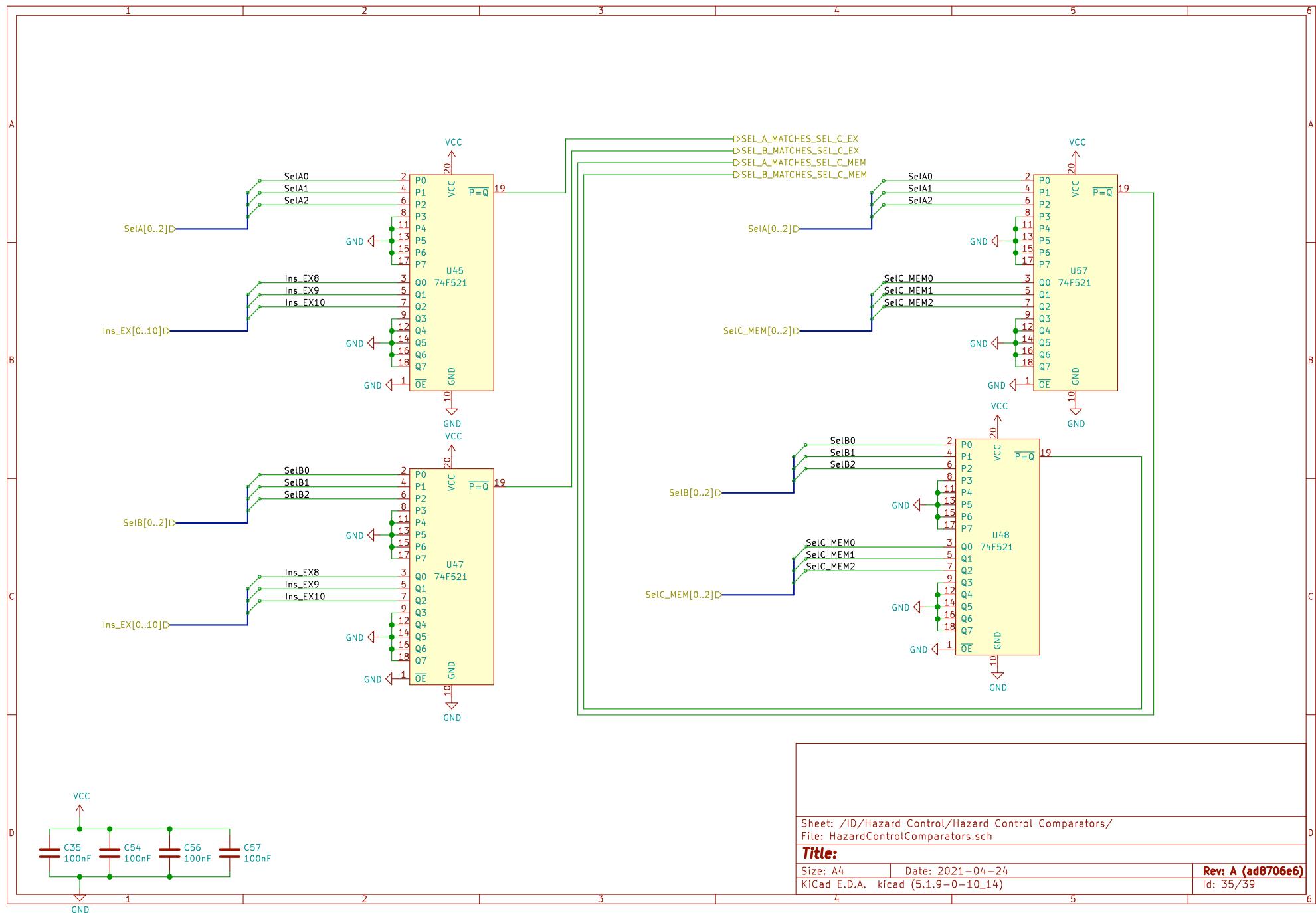
This may stall the pipeline on a hazard. For RAW hazards, it produces signals to control operand forwarding.

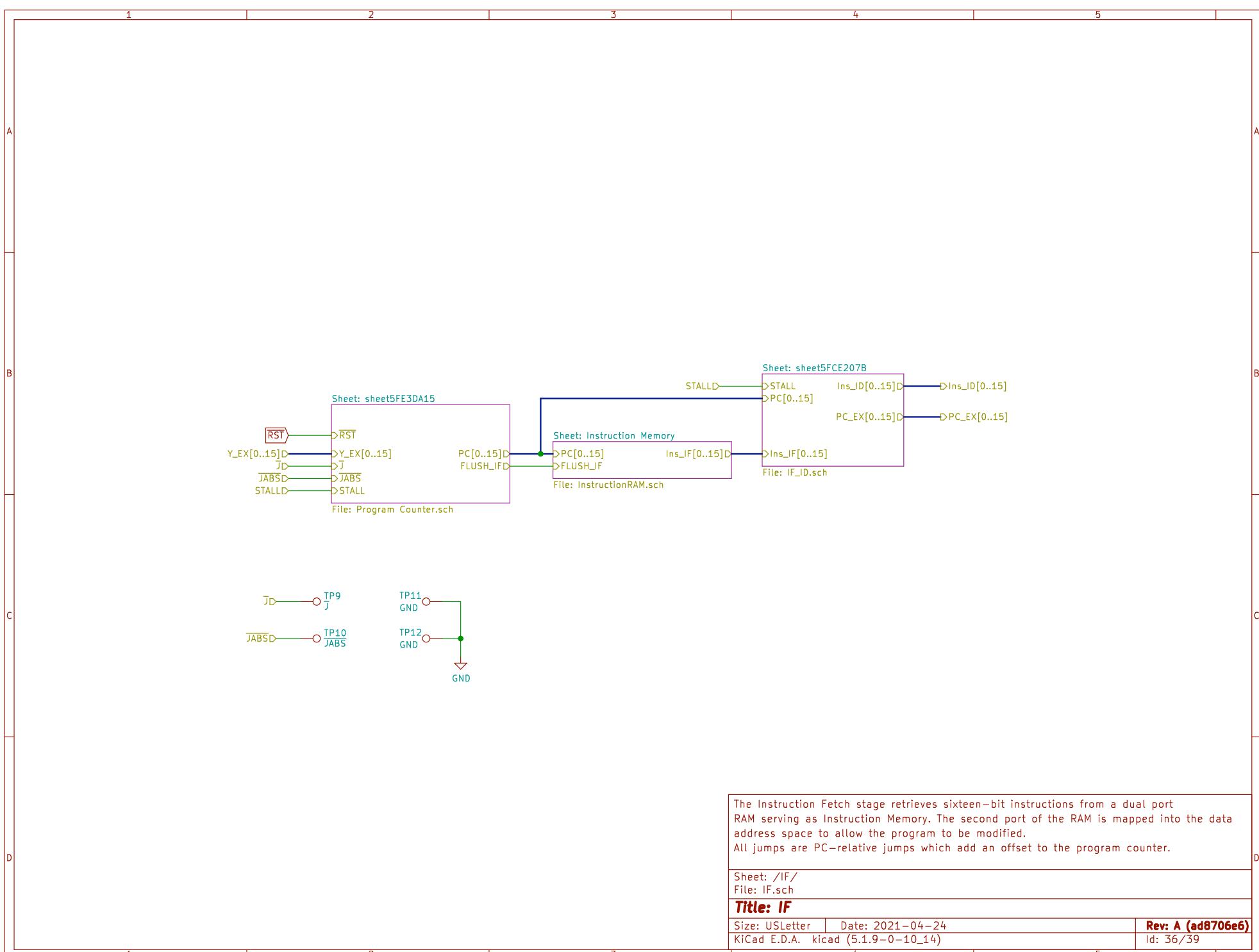
Sheet: /ID/Hazard Control/
File: HazardControl.sch

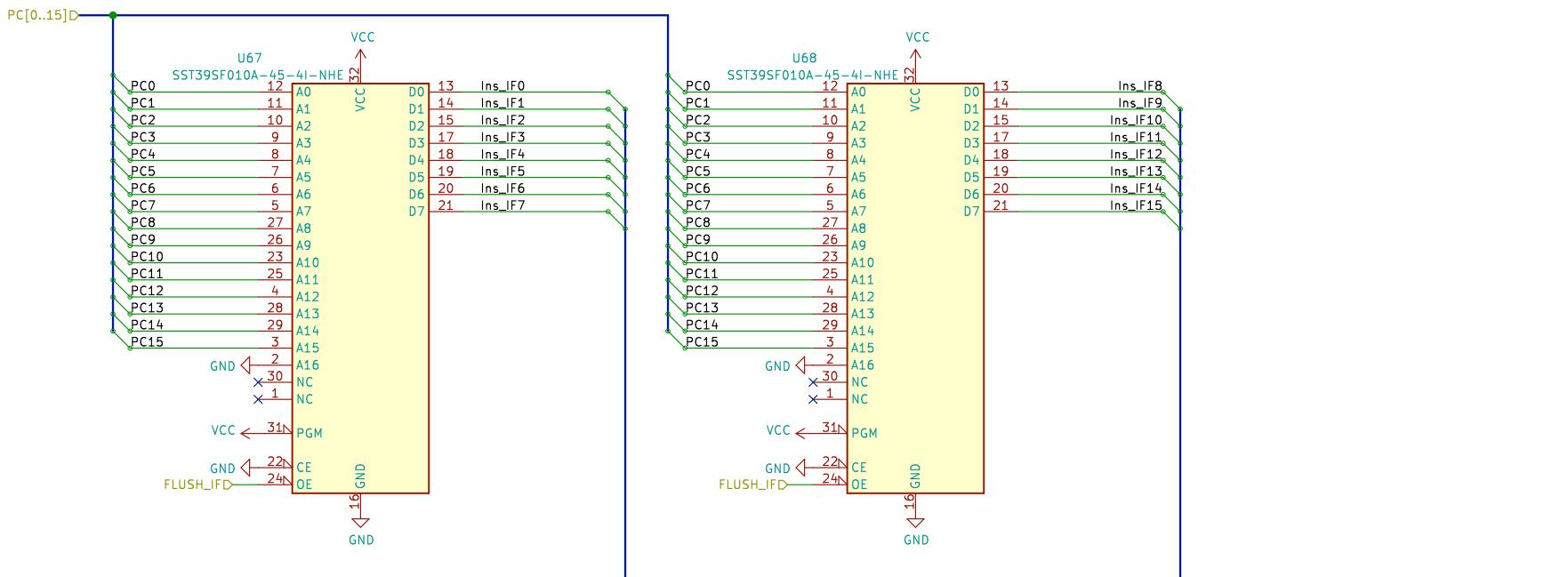
Title: Hazard Control

Size: USLetter | Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)

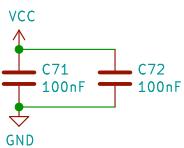
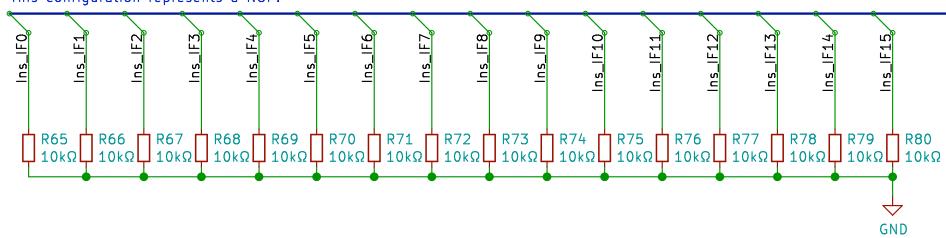
Rev: A (ad8706e6)
Id: 34/39







When the IF stage is flushed, all the output lines are pulled low.
This configuration represents a NOP.



Instructions are stored in a pair of EEPROMs in ZIF sockets.

Sheet: /IF/Instruction Memory/
File: InstructionRAM.sch

Title: Instruction ROM

Size: USLetter | Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)

Rev: A (ad8706e6)
Id: 37/39

Configure the ALU for FTAB=1 and FTF=0. This causes the A and B registers to be bypassed entirely. The F output updates on the next rising edge of the clock.

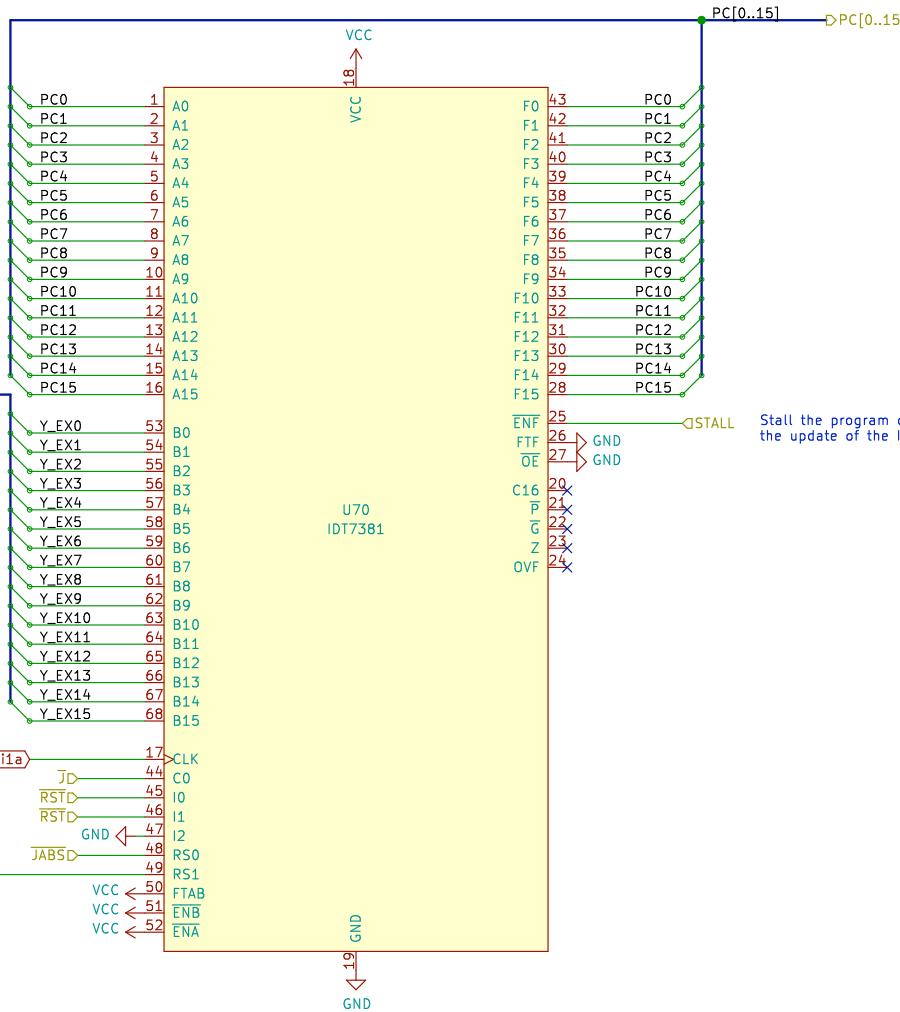
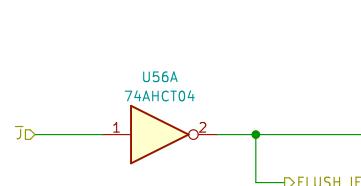
During reset, set the ALU to I2=0, I1=0, I0=0. This causes the ALU to compute a zero and latch it in A on the rising edge of the clock regardless of the value of the A and B inputs. This resets the program counter to zero.

When incrementing, set the ALU to RS1=0, RS0=1, I2=0, I1=1, I0=1, C0=1. The ALU computes $F = A + 0 + C_0$. Since output is wired to feedback to input port A, this computes $PC = PC + 1$.

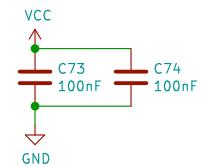
When performing a relative jump, set the ALU to RS1=1, RS0=1, I2=0, I1=1, I0=1, C0=0. The ALU computes $F = A + B$. Since the B port gets its value from the Y result of the EX stage, this computes $PC = PC + \text{offset}$.

When performing an absolute jump, set the ALU to RS1=1, RS0=0, I2=0, I1=1, I0=1, C0=0. The ALU computes $F = 0 + B$. Since the B port gets its value from the Y result of the EX stage, this computes $PC = \text{target}$.

Y_EX[0..15]



Stall the program counter by disabling the update of the IDT7381's F register.



Sixteen-bit program counter will either increment on the clock, add a specified sixteen-bit offset, or else reset to zero.

Sheet: /IF/sheet5FE3DA15/
File: Program Counter.sch

Title: Program Counter

Size: USLetter | Date: 2021-04-24
KiCad E.D.A. kicad (5.1.9-0-10_14)

Rev: A (ad8706e6)
Id: 38/39

