

A

A

B

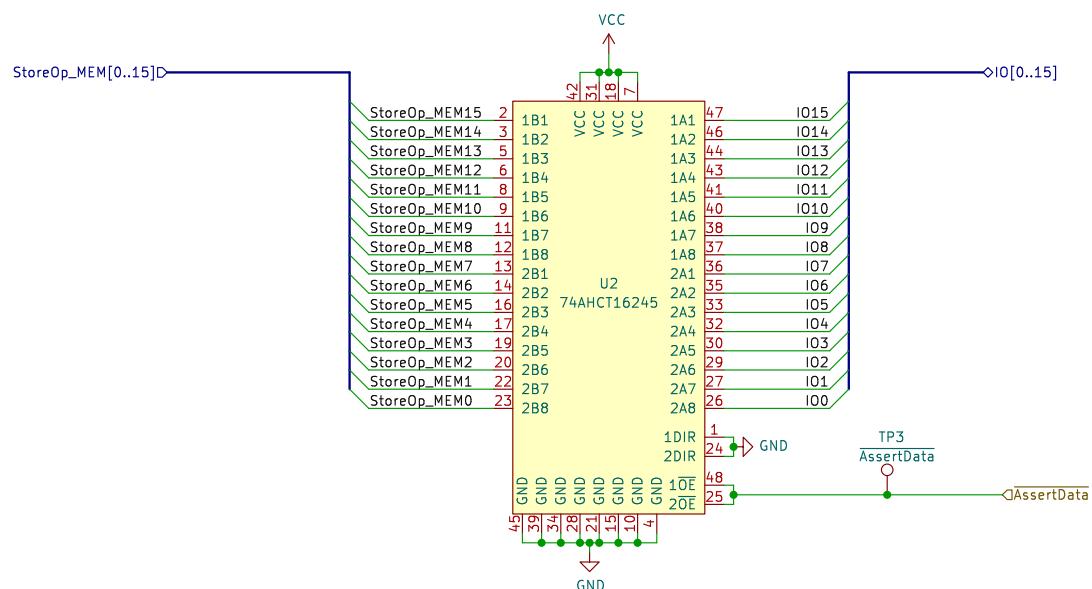
B

C

C

D

D



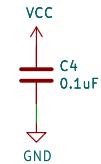
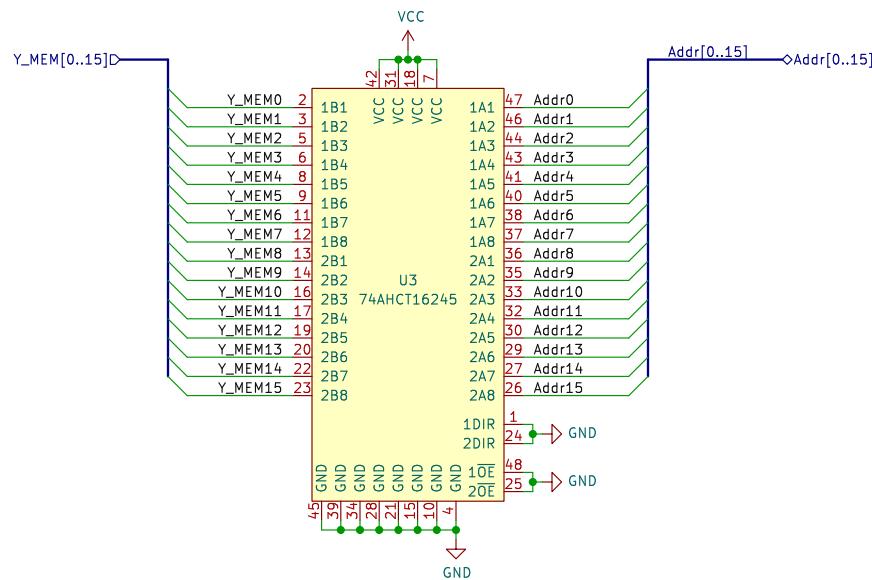
The store operand can be disconnected from the system memory bus via tristate outputs. This allows peripheral devices to assert their own value to the bus during a Load.

Sheet: /MEM/Buffer StoreOp As Bus I/O/
File: BufferStoreOpAsBusIO.kicad_sch

Title: MEM: Buffer StoreOp

Size: USLetter Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

Rev: A
Id: 5/37



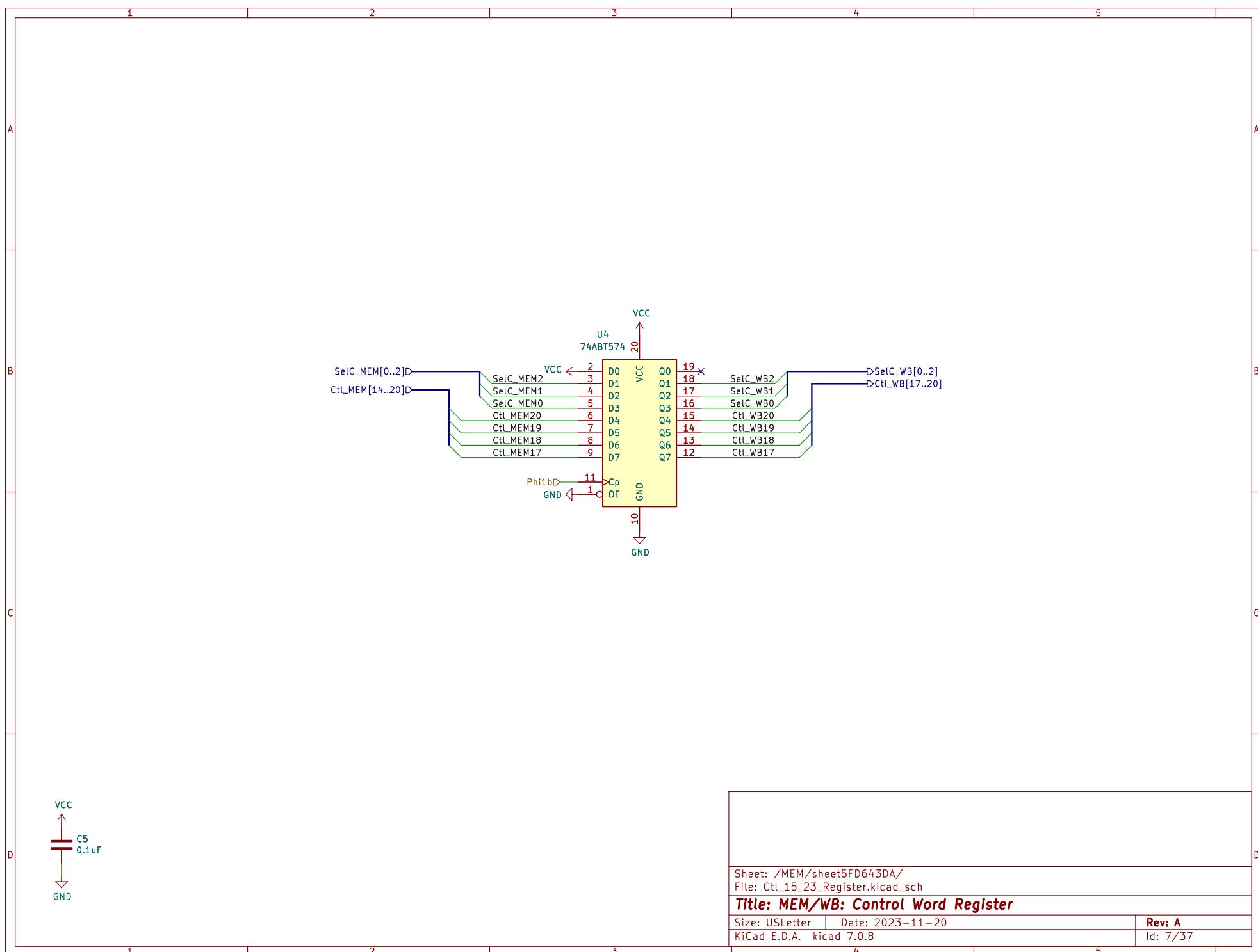
Buffer the effective address before it leaves the main board.

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

Title: MEM: Buffer ALUResult

Size: USLetter | Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

Rev: A
Id: 6/37

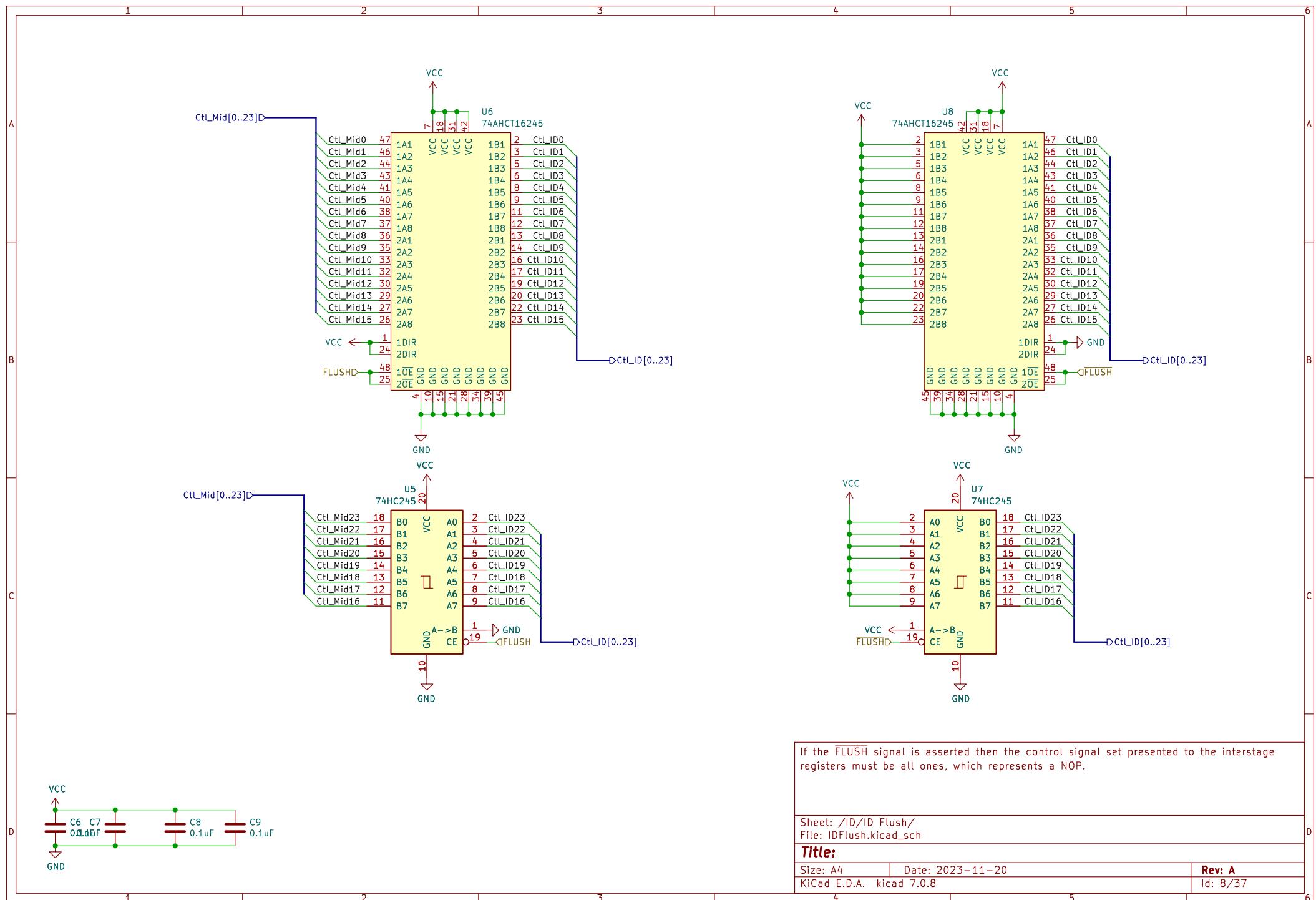


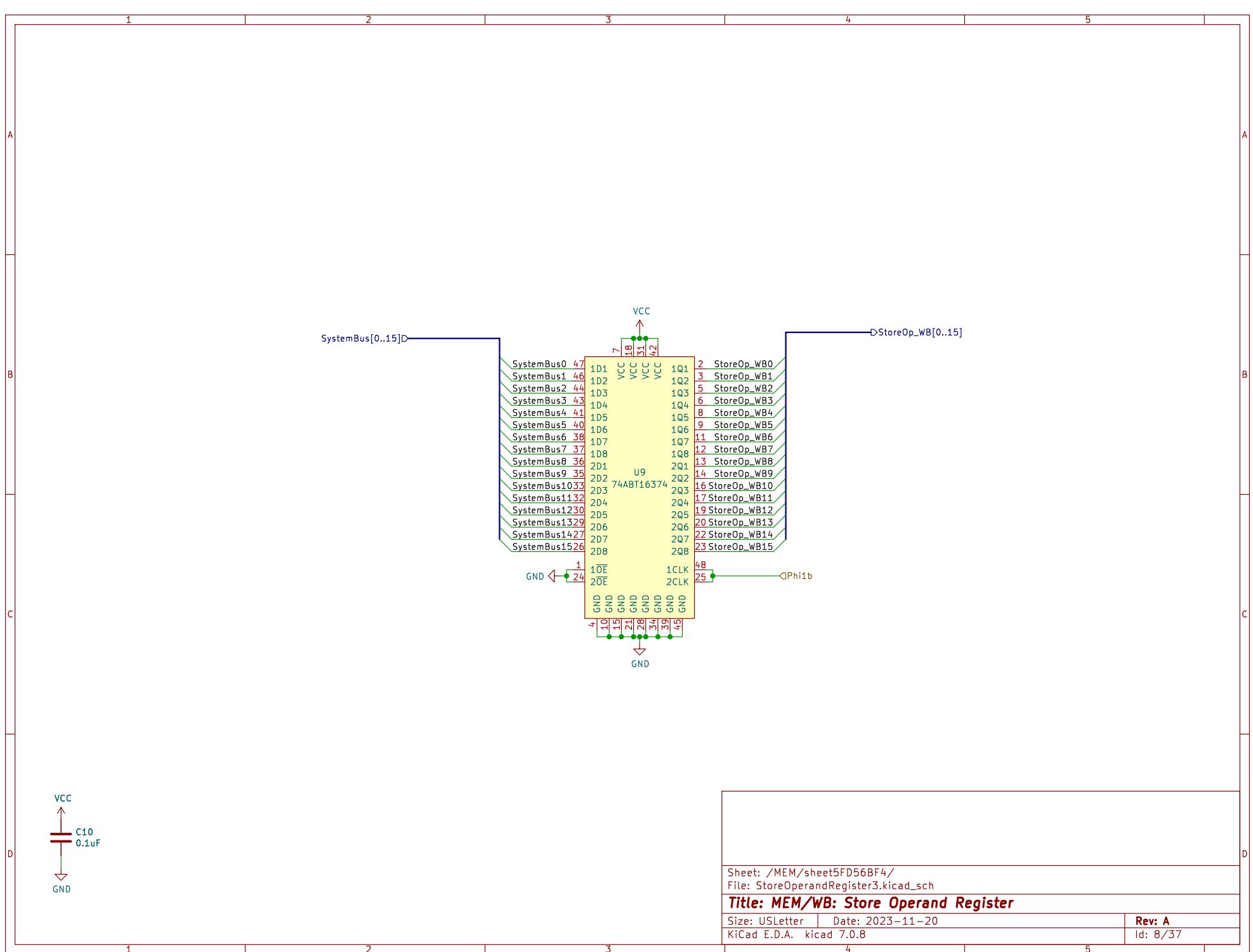
Sheet: /MEM/sheet5FD643DA/
File: Ctl_15_23_Register.kicad_sch

Title: MEM/WB: Control Word Register

Size: USLetter | Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

Rev: A
Id: 7/37





A

A

B

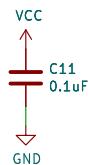
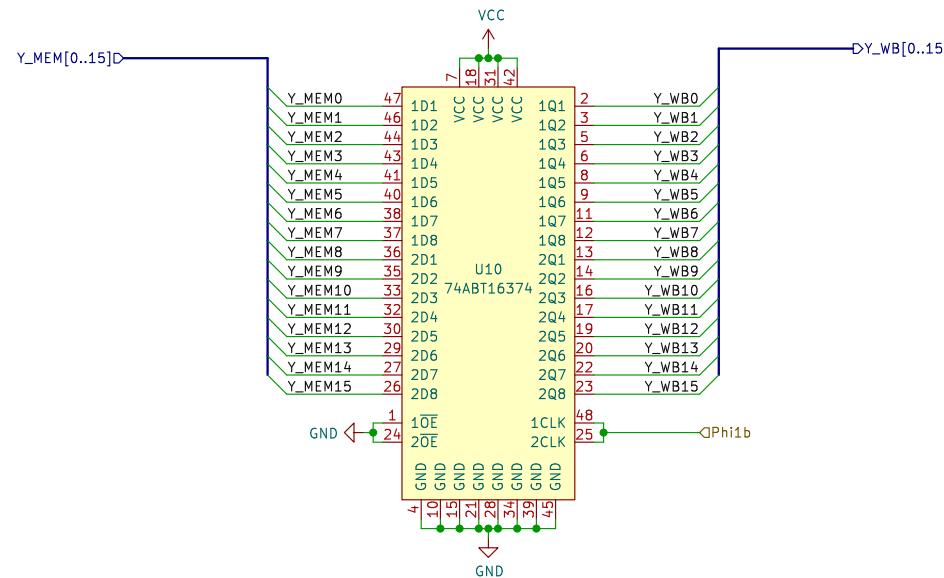
B

C

C

D

D



Sheet: /MEM/Sheet5FD9EFDA/
File: ALUResultRegister_MEM_WB.kicad_sch

Title: ALU Result Register MEM/WB

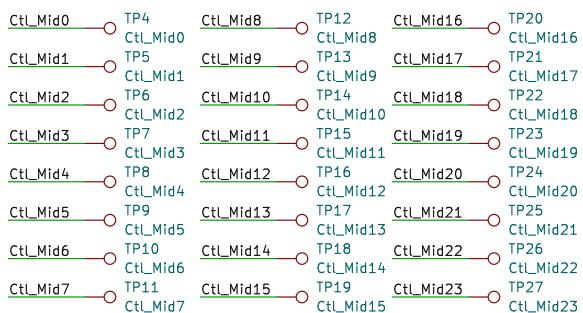
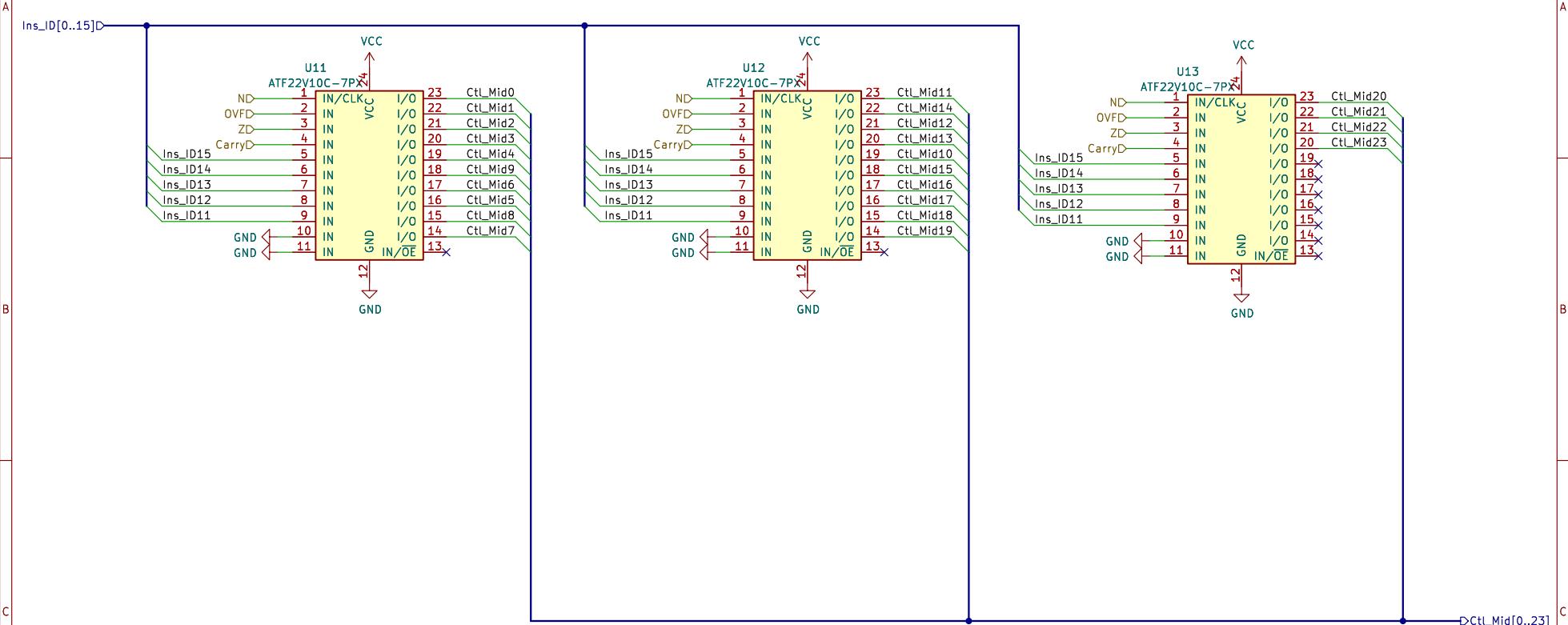
Size: USLetter | Date: 2023-11-20

KiCad E.D.A. kicad 7.0.8

Rev: A

Id: 10/37

1 2 3 4 5



Decodes an instruction opcode into an array of control signals.

Sheet: /ID/sheet60693B97/
File: InstructionDecoder.kicad_sch

Title: Instruction Decoder

Size: USLetter | Date: 2023-11-20

KiCad E.D.A. kicad 7.0.8

Rev: A

Id: 28/37

1 2 3 4 5

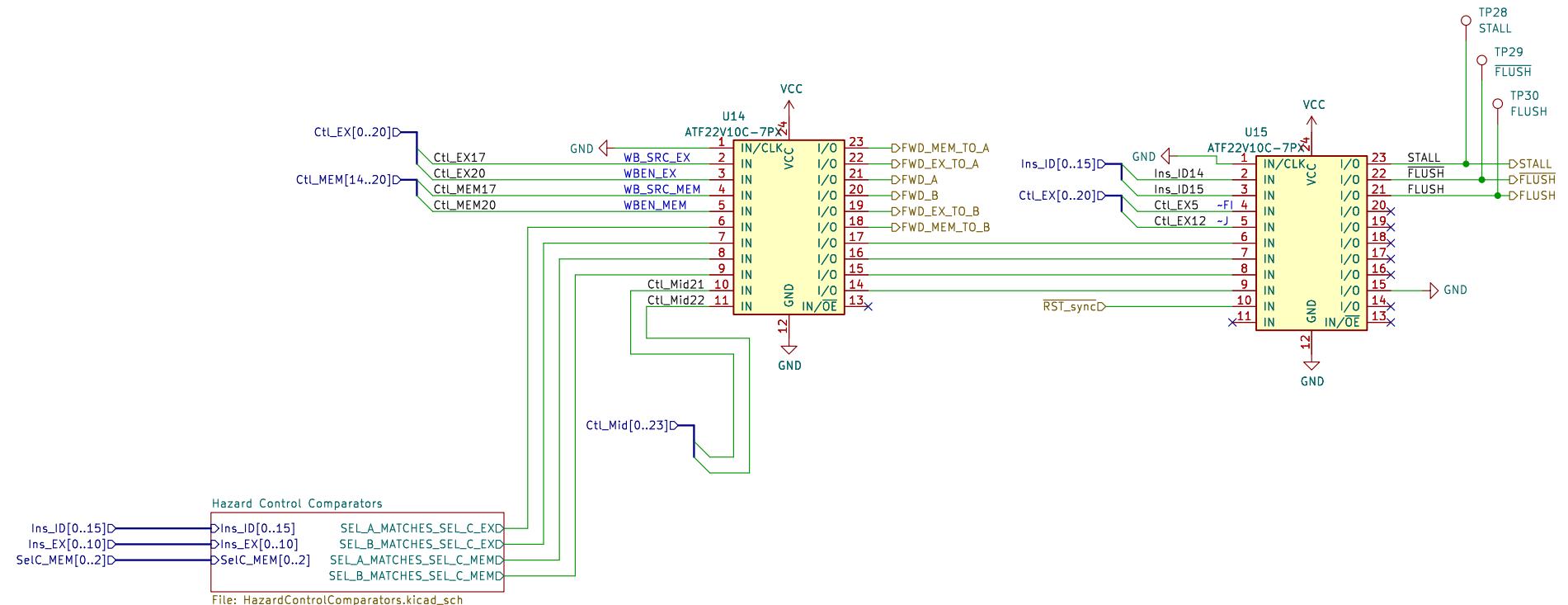
Hazard Control detects several types of hazards and resolves each one either by stalling the pipeline, or by forwarding an operand from a later pipeline stage.

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.



Control logic for dealing with pipeline hazards

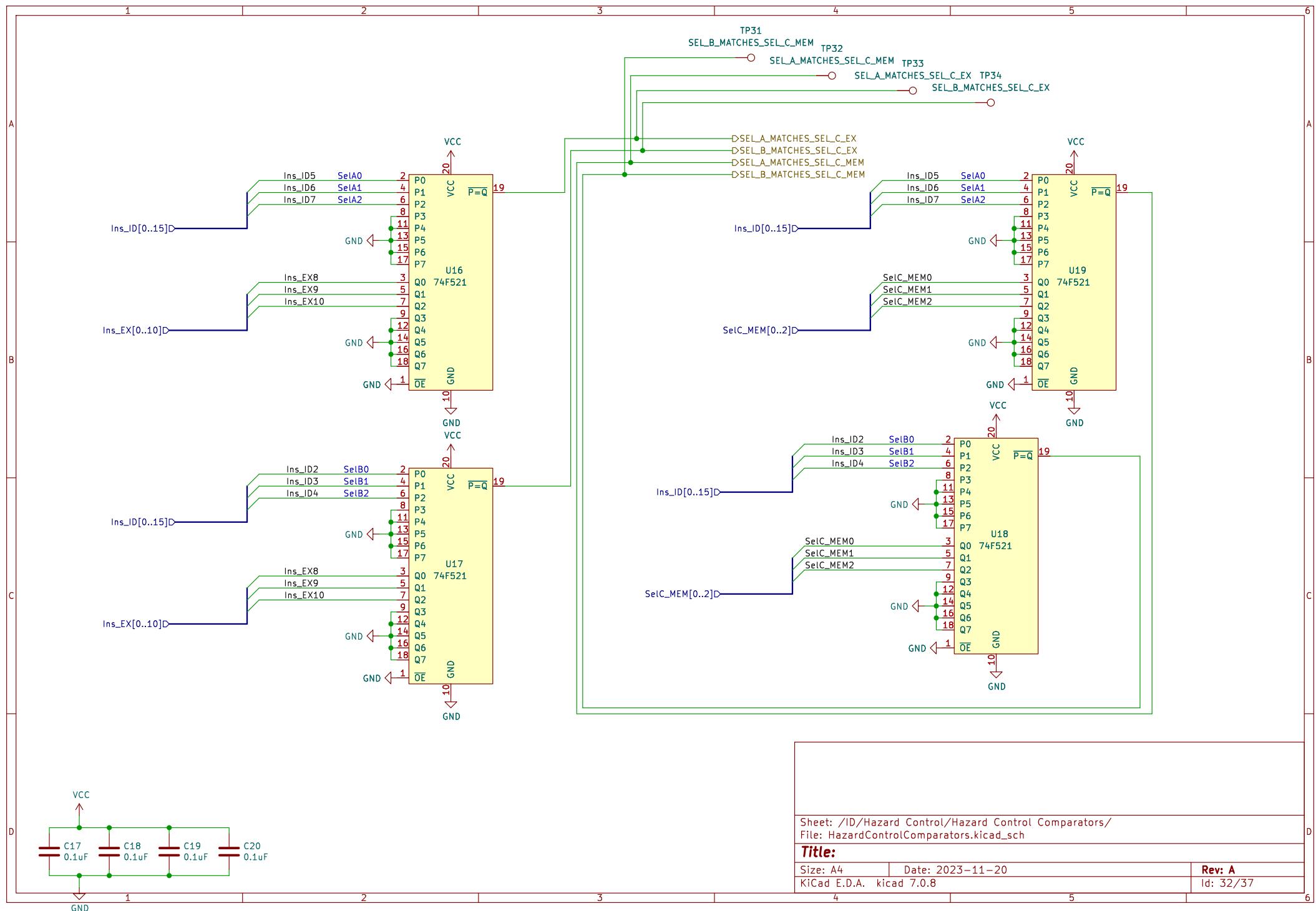
This may stall the pipeline on a hazard. For RAW hazards, it produces signals to control operand forwarding.

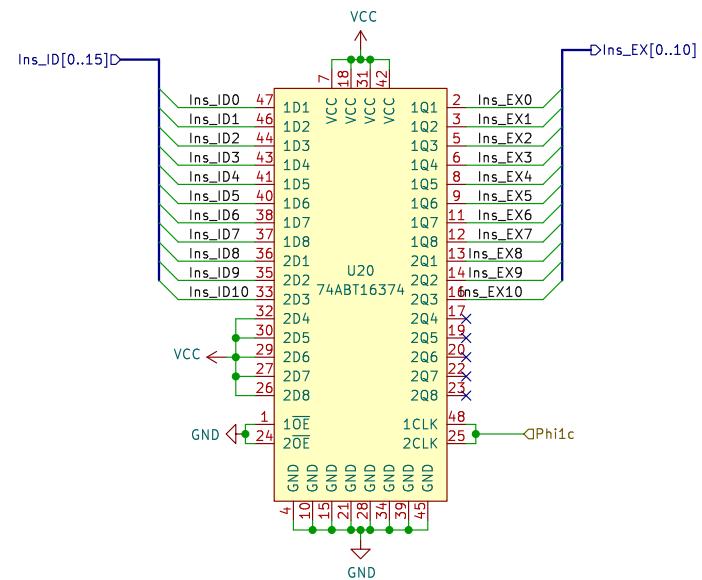
Sheet: /ID/Hazard Control/
File: HazardControl.kicad_sch

Title: Hazard Control

Size: USLetter | Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

Rev: A
Id: 31/37



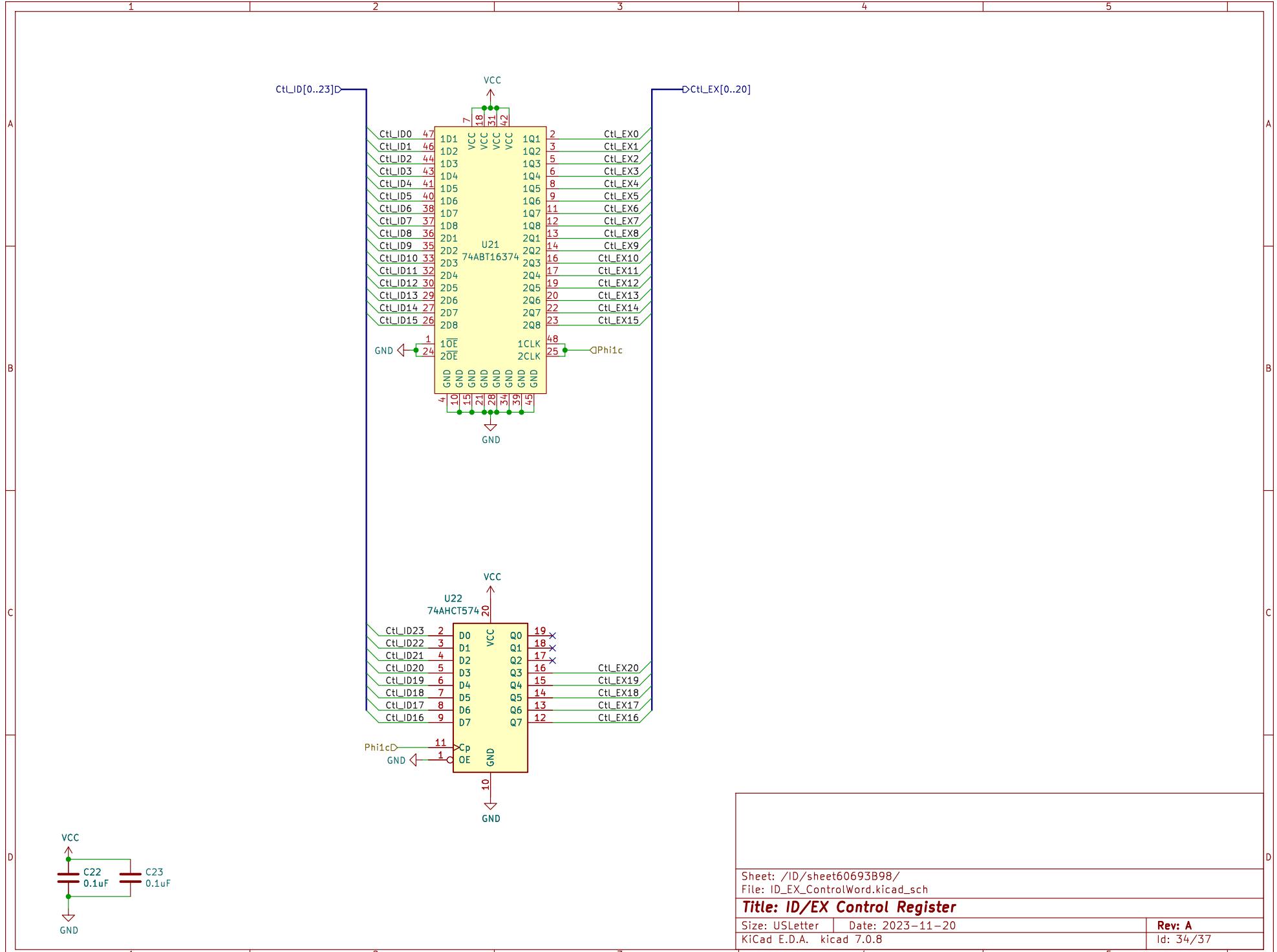


Sheet: /ID/sheet60693B99/
File: ID_EX_InstructionWord.kicad_sch

Title:

Size: USLetter | Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

Rev: A
Id: 33/37



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

Title: ID/EX Control Register

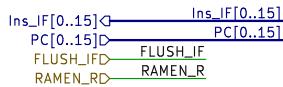
Size: USLetter Date: 2023-11-20

KiCad E.D.A. kicad 7.0.8

Rev: A

Id: 34/37

A



J2 UpperHeader	
1	PC0
2	PC1
3	PC2
4	PC3
5	PC4
6	PC5
7	PC6
8	PC7
9	PC8
10	PC9
11	PC10
12	PC11
13	PC12
14	PC13
15	PC14
16	PC15
17	▶ GND
18	▶ VCC

J3 LowerHeader	
1	Ins_IF0
2	Ins_IF1
3	Ins_IF2
4	Ins_IF3
5	Ins_IF4
6	Ins_IF5
7	Ins_IF6
8	Ins_IF7
9	Ins_IF8
10	Ins_IF9
11	Ins_IF10
12	Ins_IF11
13	Ins_IF12
14	Ins_IF13
15	Ins_IF14
16	Ins_IF15
17	FLUSH_IF
18	RAMEN_R

RAMEN_R enables the ROM.
The right side of the RAM is used for Instruction Fetch. If the IF stage is not being fed by RAM then it is being fed by this ROM.

B

A

C

B

D

C

Sheet: /Memory/ROM/
File: ROM.kicad_sch

Title:

Size: A4 | Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

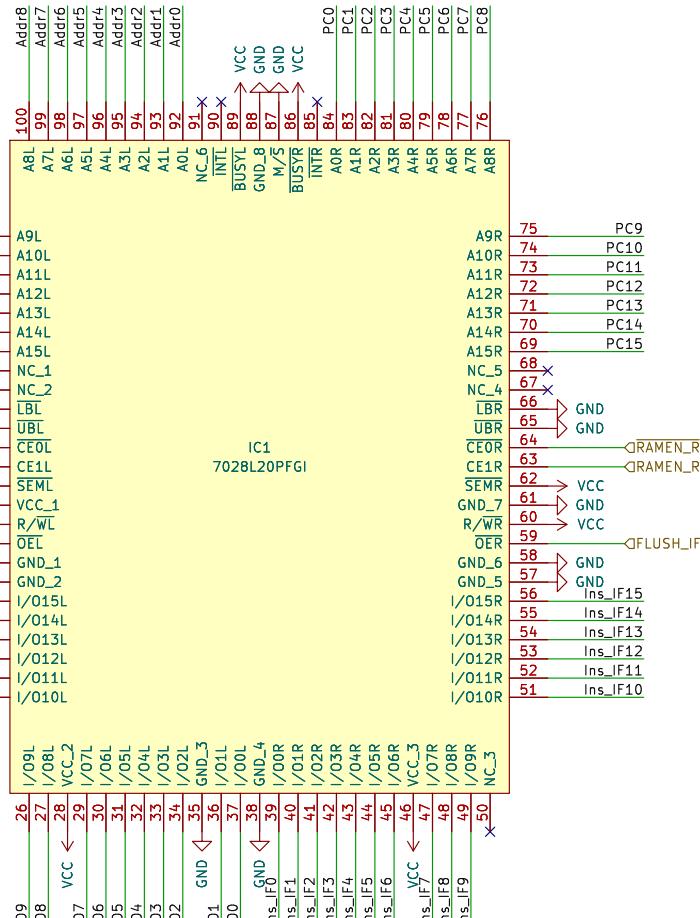
Rev: A
Id: 14/37

The RAM is configured in \bar{S} mode to disable the onboard write arbitration logic.

IO[0..15] ◊ IO[0..15]
 Addr[0..15] ◊ Addr[0..15]
 Ins_IF[0..15] ◊ Ins_IF[0..15]
 PC[0..15] ◊ PC[0..15]

If the IF stage is not being fed by ROM then it is being fed by RAM.

Addr9	1
Addr10	2
Addr11	3
Addr12	4
Addr13	5
Addr14	6
Addr15	7
	8
	9 NC_1
	10 GND
	11 GND
RAMEN_LD	12 LBL
RAMEN_LD	13 UBL
VCC	14 CEOL
VCC	15 SEML
VCC_1	16 VCC_1
MemStoreD	17 R/WL
MemLoadD	18 OEL
GND	19 GND_1
GND	20 GND_2
IO15	21 /015L
IO14	22 /014L
IO13	23 /013L
IO12	24 /012L
IO11	25 /011L
IO10	/010L



RAM is implemented with a 64K x 16 dual port SRAM so that the IF stage can fetch an instruction on the second read port.

Sheet: /Memory/RAM/
File: RAM.kicad_sch

Title: Turtle16: RAM

Size: A4 Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

Rev: A
Id: 15/37

A

A

B

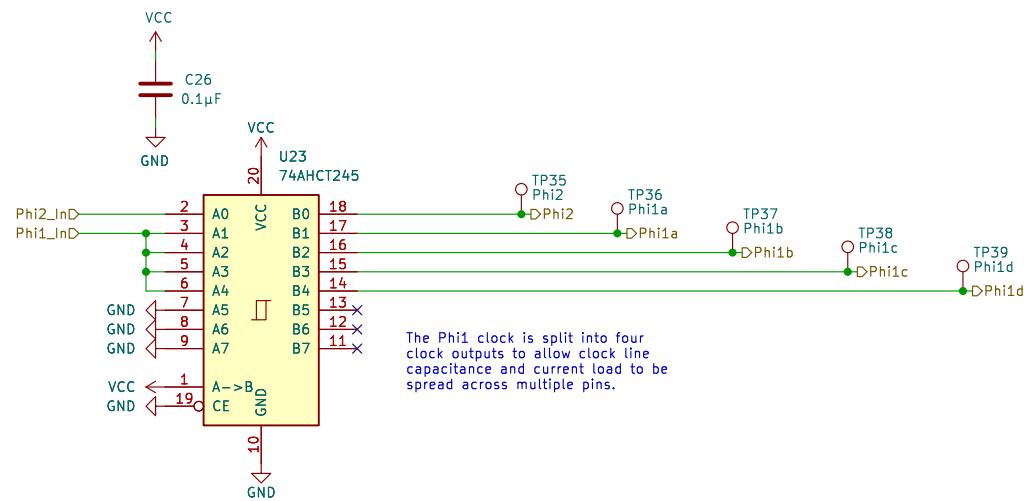
B

C

C

D

D



Sheet: /Clock Distribution/
File: ClockDistribution.kicad_sch

Title: Turtle16: Clock Distribution

Size: USLetter | Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

Rev: A
Id: #/37

A

A

B

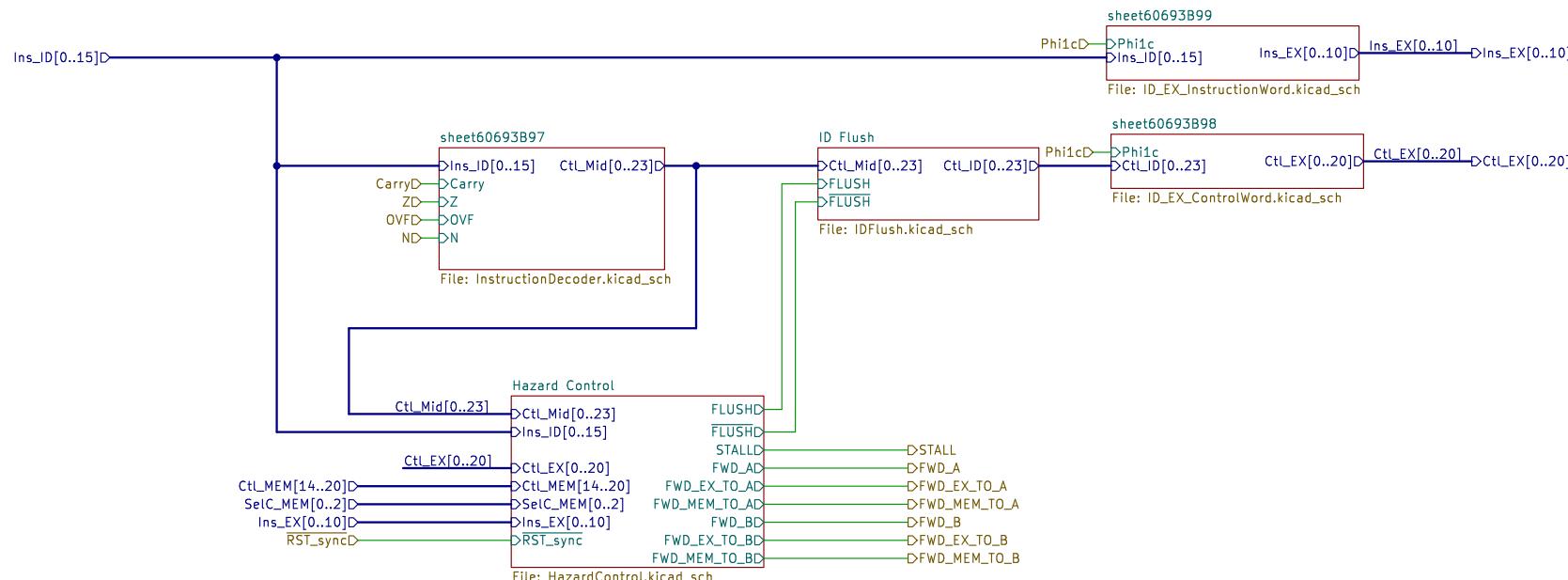
B

C

C

D

D



#	Mnemonic	Description
0	/HLT	Halt Clock
1	SelStoreOpA	Select Store Operand 0
2	SelStoreOpB	Select Store Operand 1
3	SelRightOpA	Select Right Operand 1
4	SelRightOpB	Select Right Operand 2
5	/FI	Flags Register In
6	CarryIn	ALU Carry input
7	IO	ALU IO input
8	I1	ALU I1 input
9	I2	ALU I2 input
10	RS0	ALU RS0 input
11	RS1	ALU RS1 input
12	/J	Jump
13	/JABS	Absolute Jump
14	/MemLoad	Memory Load
15	/MemStore	Memory Store
16	/AssertStoreOp	Drive the Store operand onto the bus I/O lines
17	WriteBackSrc	Select write back source
18	/WRL	Write back low byte
19	/WRH	Write back high byte
20	/WBEN	Enable write back to register file
21	LUNUSED	Left operand is unused
22	RUNUSED	Right operand is unused

The instruction decoder turns a 5-bit opcode into an array of control signals. The decoder takes the condition code from the flags register into account to effect conditional instructions. Simultaneously, read the register file using indices extracted from the instruction word.

Sheet: /ID/
File: ID.kicad_sch

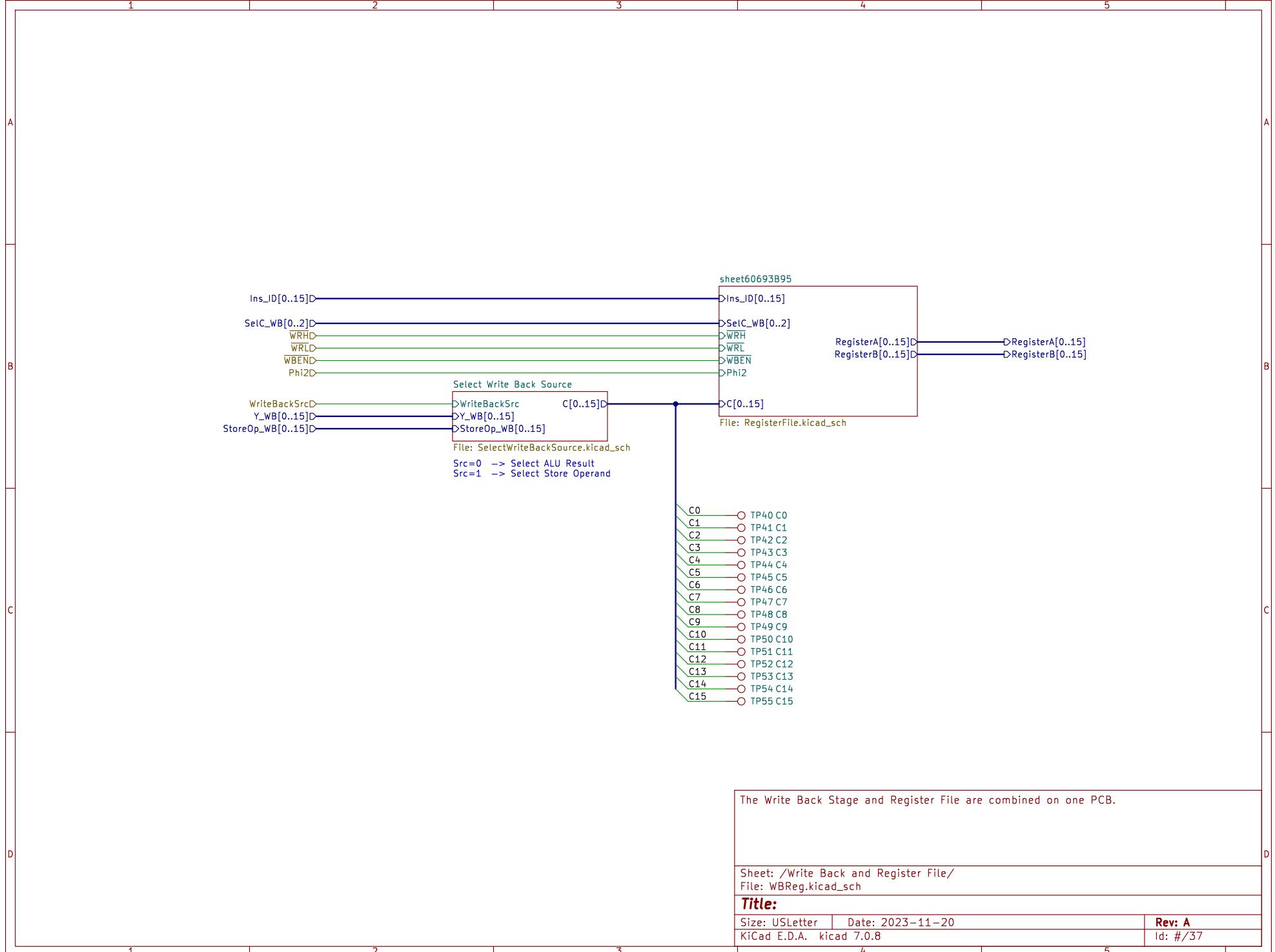
Title: ID

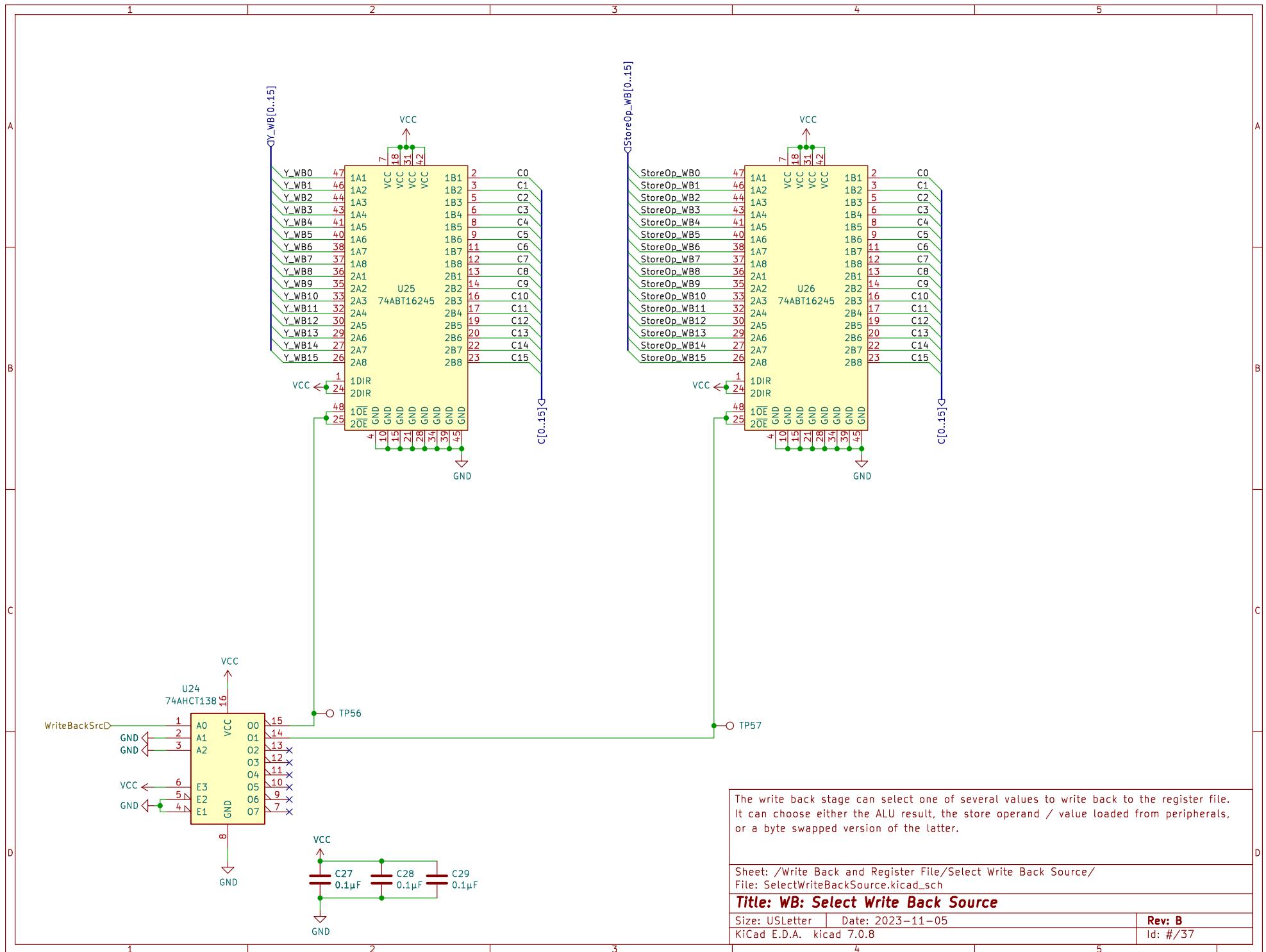
Size: USLetter Date: 2023-11-20

KiCad E.D.A. kicad 7.0.8

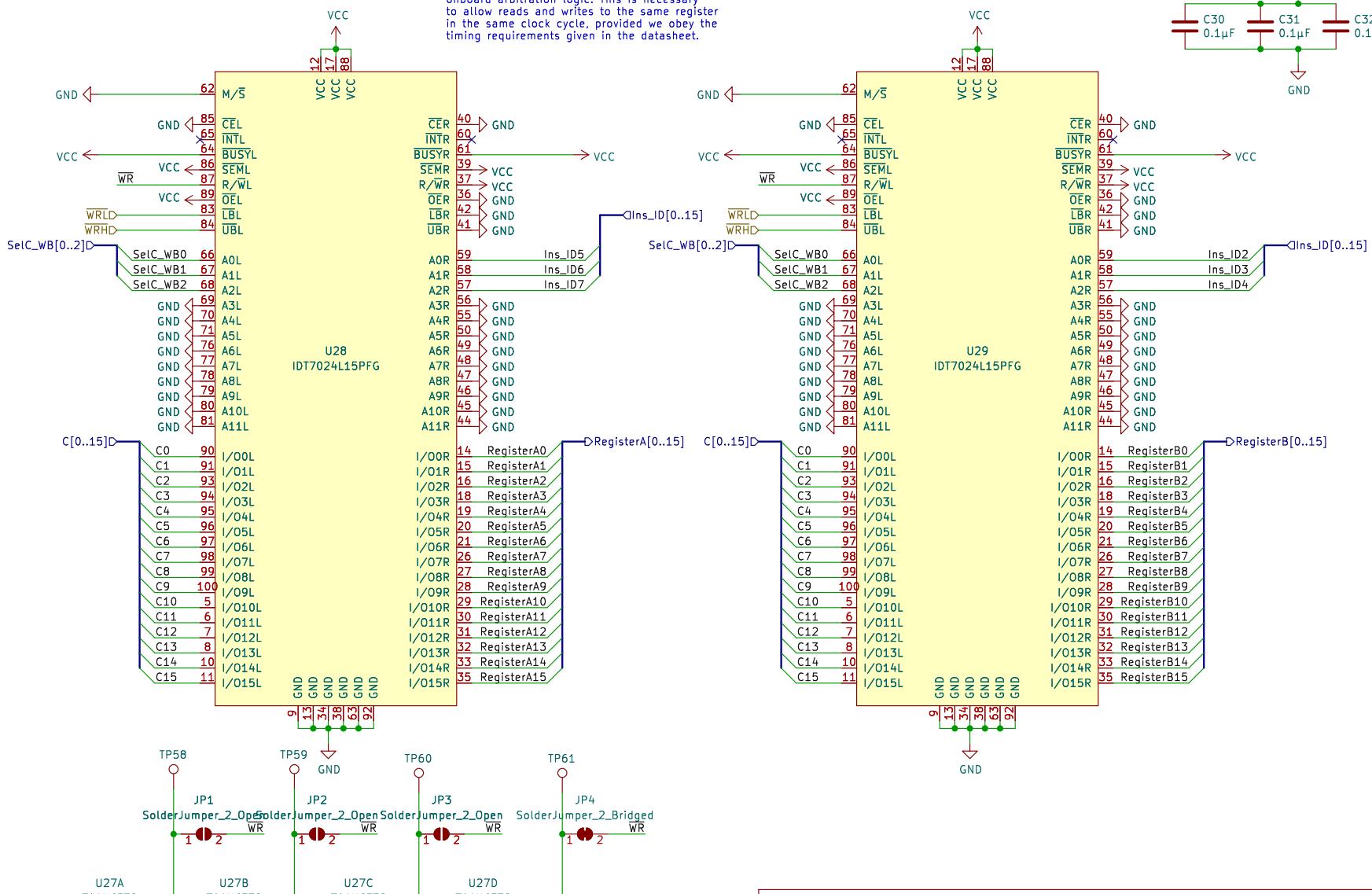
Rev: A

Id: #/37





Per AN-91, the dual-port SRAMs must be configured in \overline{S} mode to disable the onboard arbitration logic. This is necessary to allow reads and writes to the same register in the same clock cycle, provided we obey the timing requirements given in the datasheet.



Delay the \overline{WR} signal by up to 8ns per gate. The three OR-gates cause the WR signal to appear on the dual-port SRAM about 24ns after the other inputs. This is necessary to ensure correct values are written to the SRAM.

The data sheet notes that any of various control signals, including WR must remain high while address bits are changing.

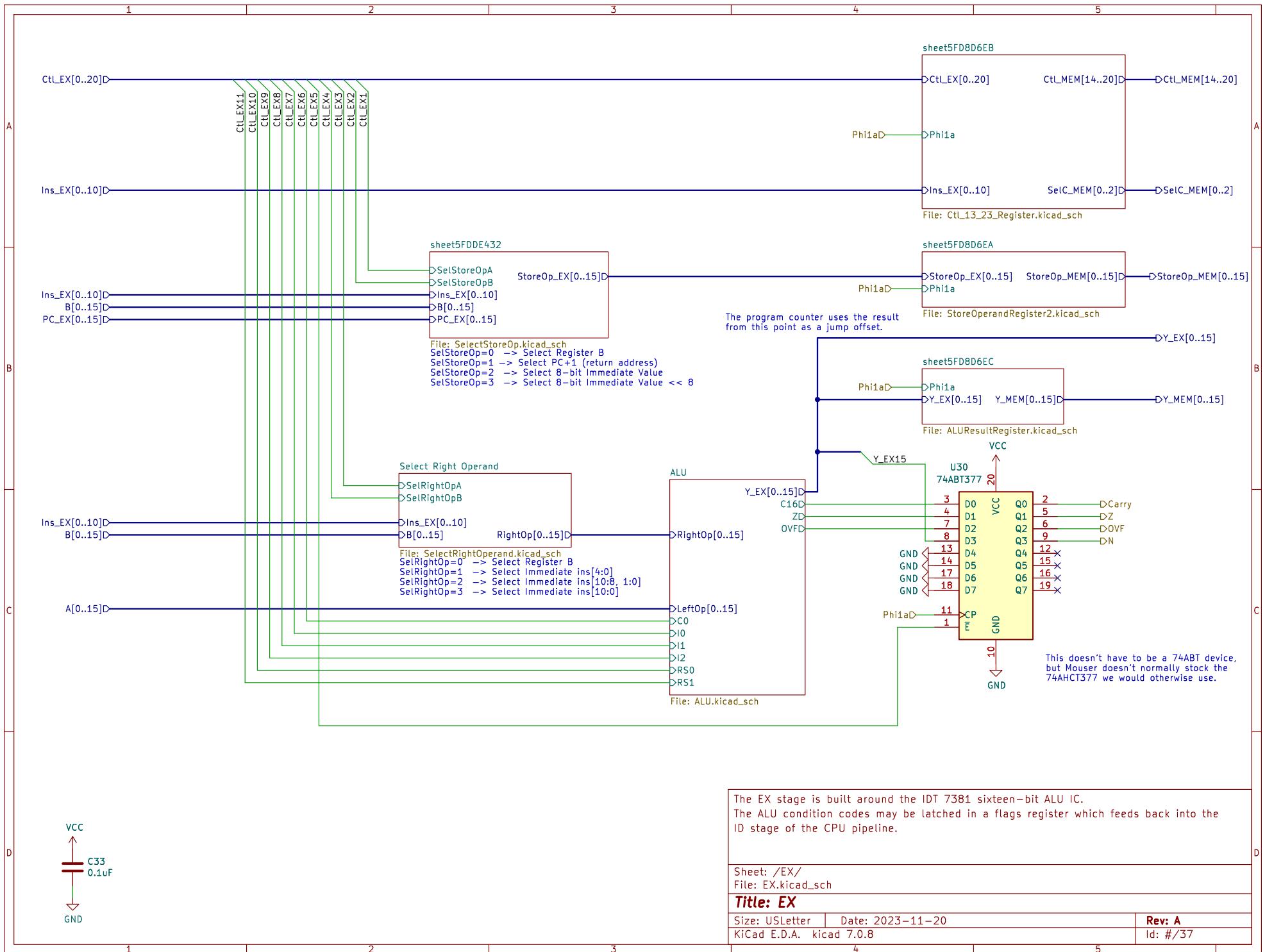
Triple Port Register File build from dual port SRAM

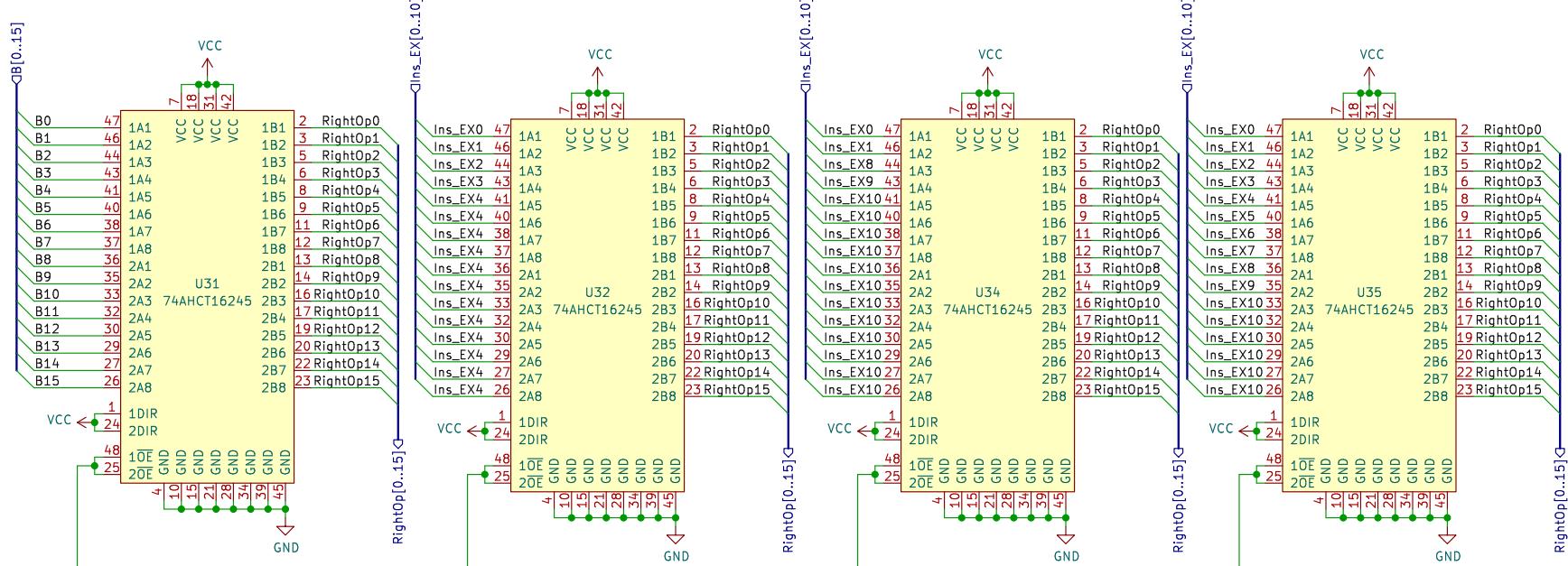
Sheet: /Write Back and Register File/sheet60693B95/
File: RegisterFile.kicad_sch

Title: Register File

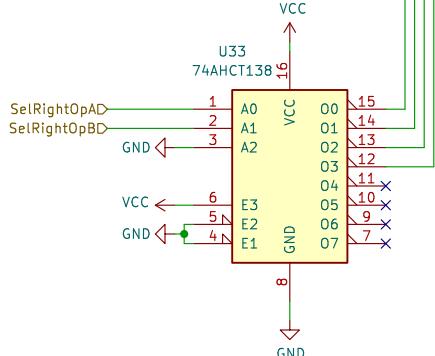
Size: USLetter | Date: 2023-11-05
KiCad E.D.A. kicad 7.0.8

Rev: B
Id: #/37





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.kicad_sch

Title: Select Right Operand

Size: USLetter Date: 2023-11-20

KiCad E.D.A. kicad 7.0.8

Rev: A

Id: #/37

A

A

B

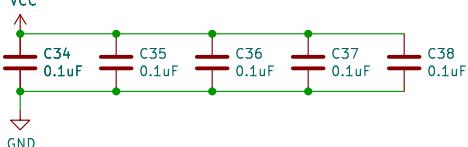
B

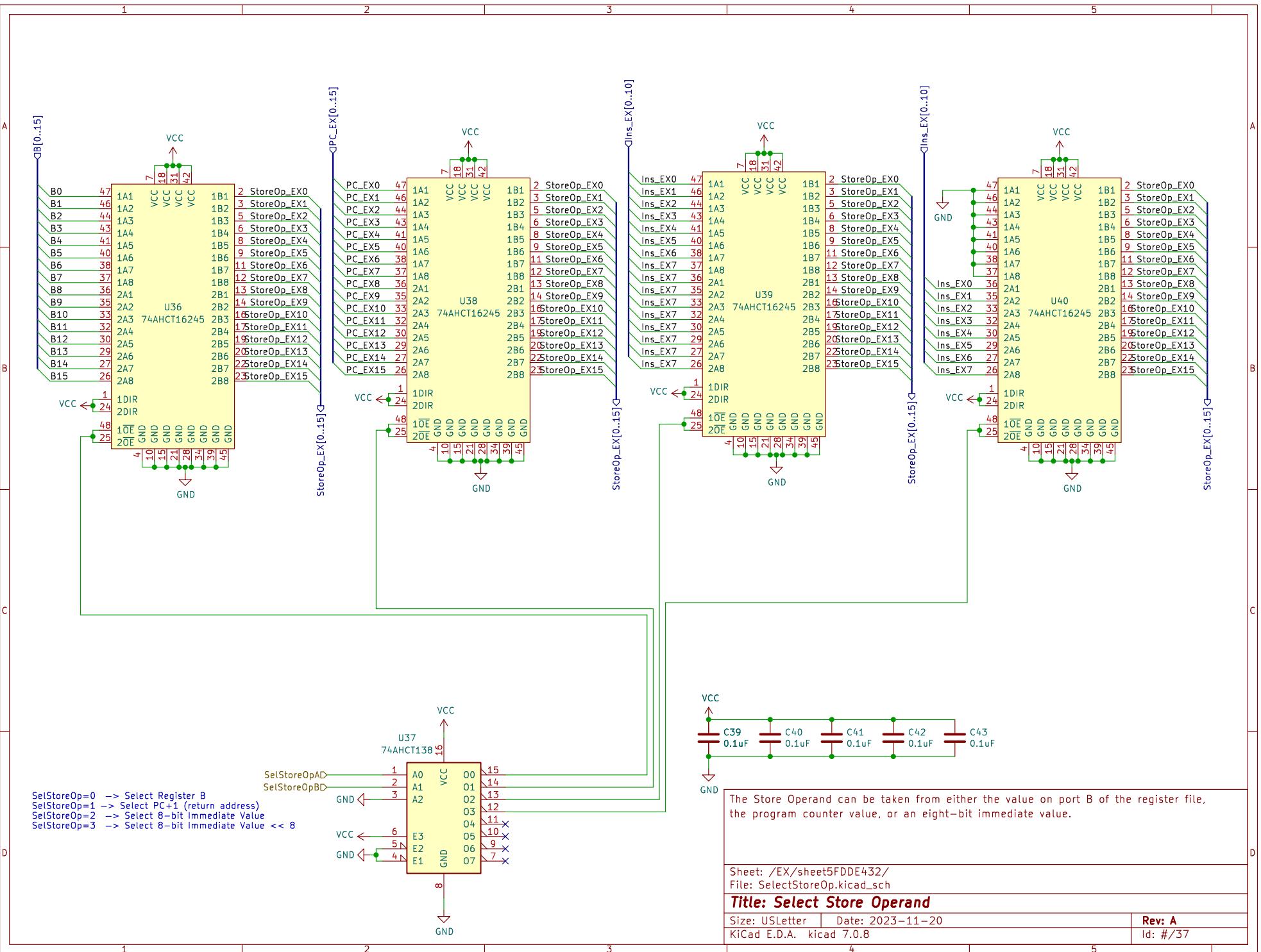
C

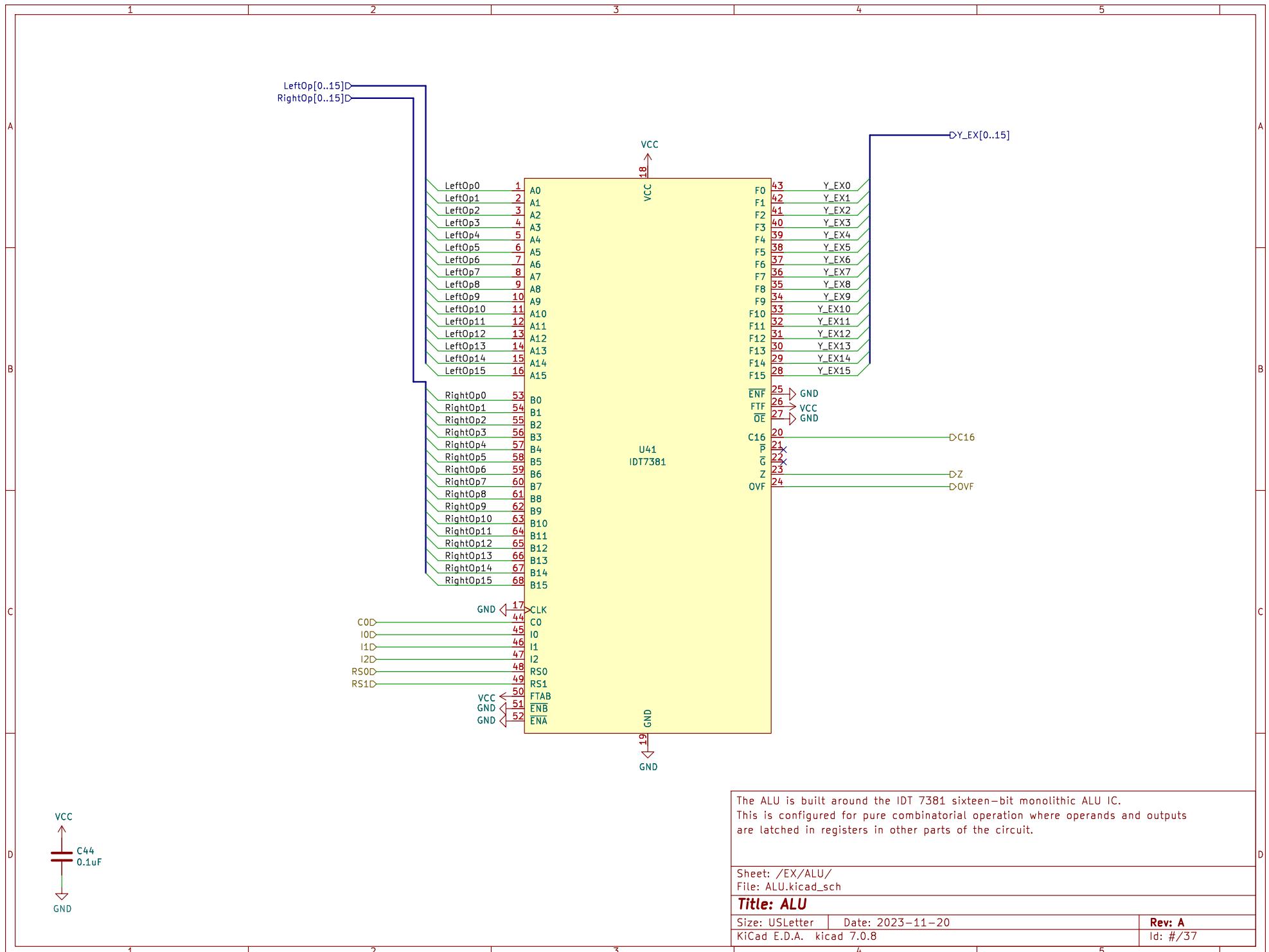
C

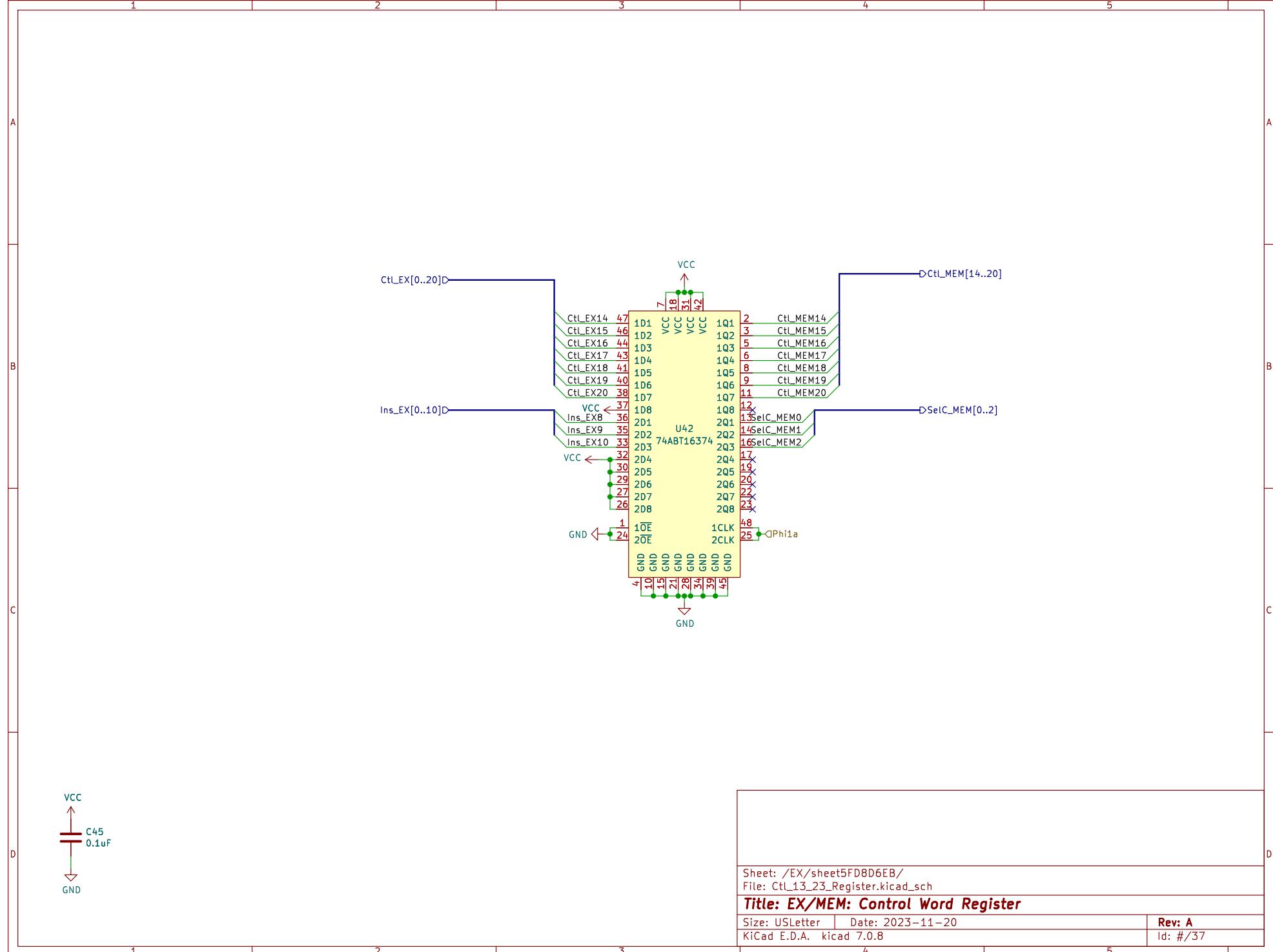
D

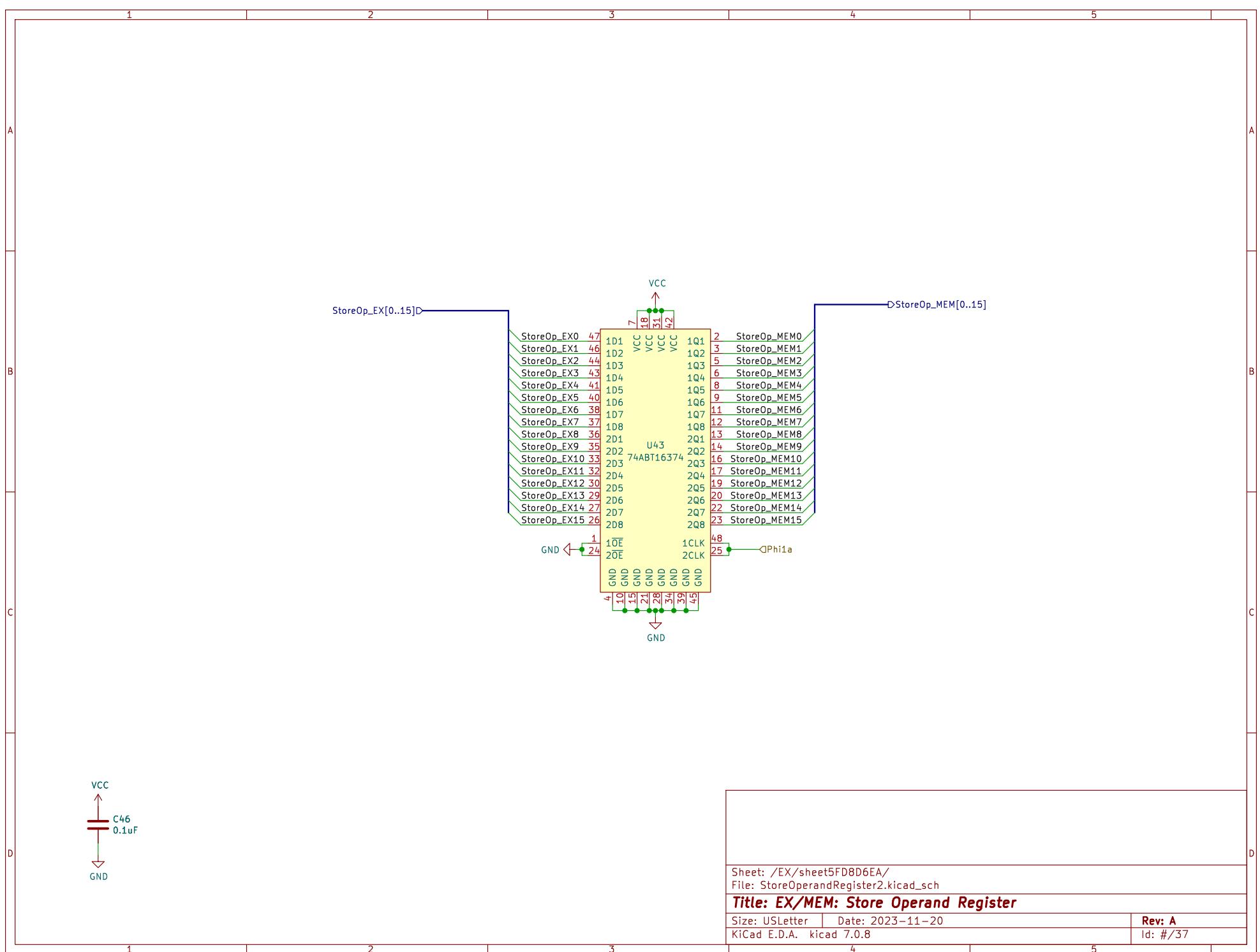
D

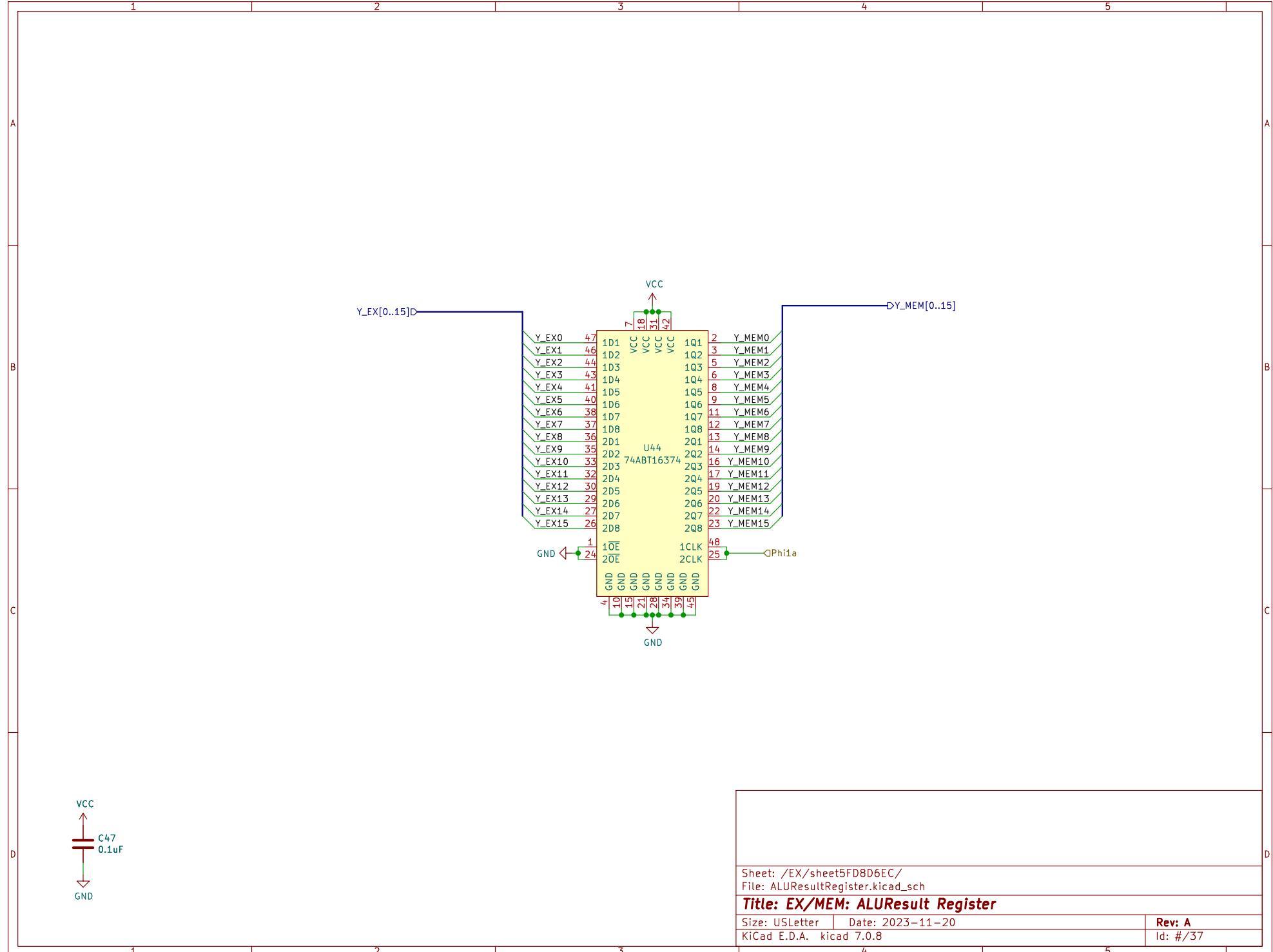












A

B

C

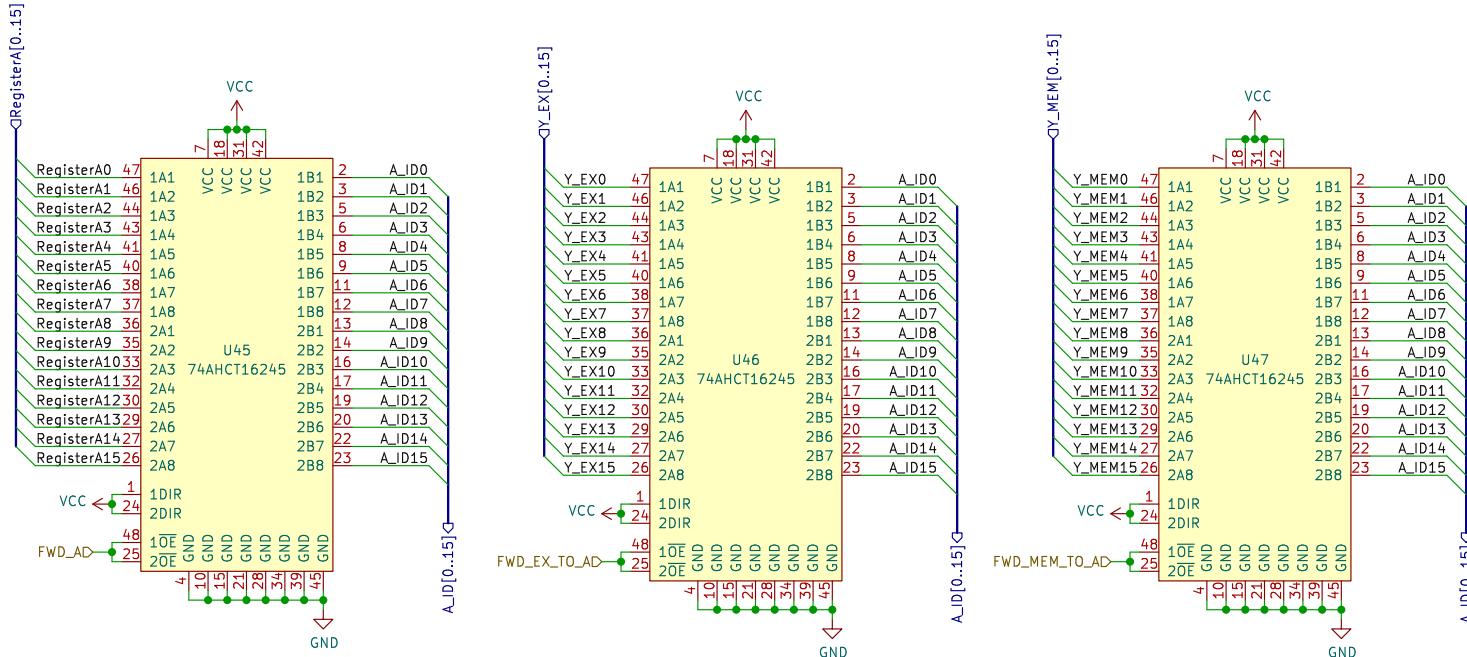
D

A

B

C

D

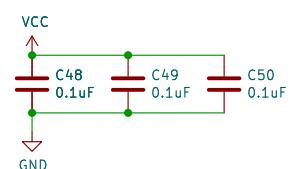


Sheet: /Operand Forwarding A/
File: OperandForwardingA.kicad_sch

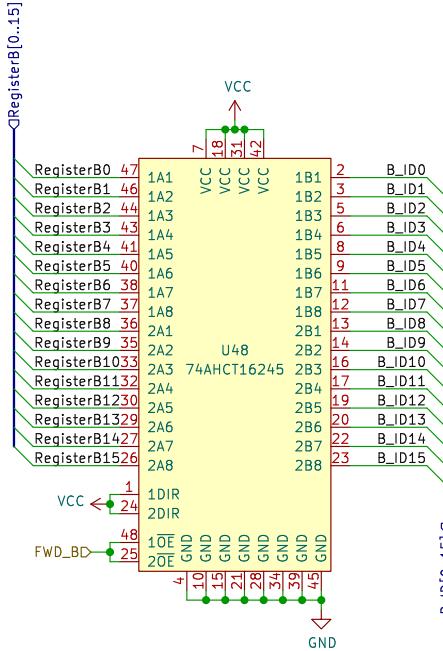
Title:

Size: A4 | Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

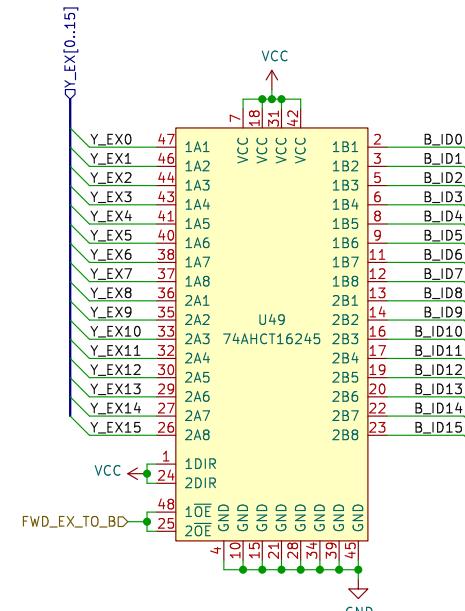
Rev: A
Id: #/37



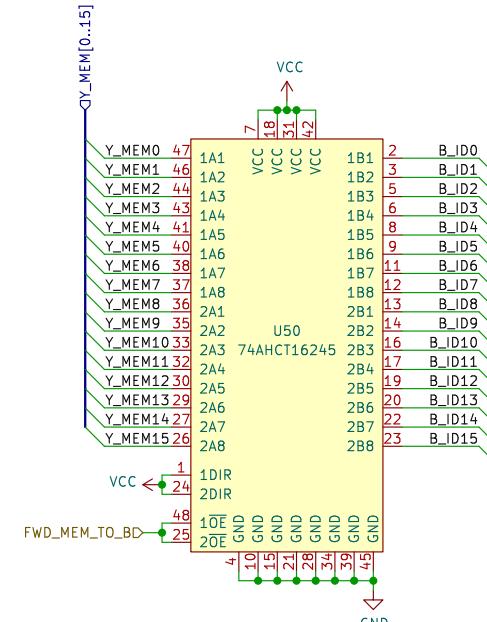
A



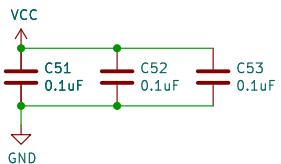
B



C



D

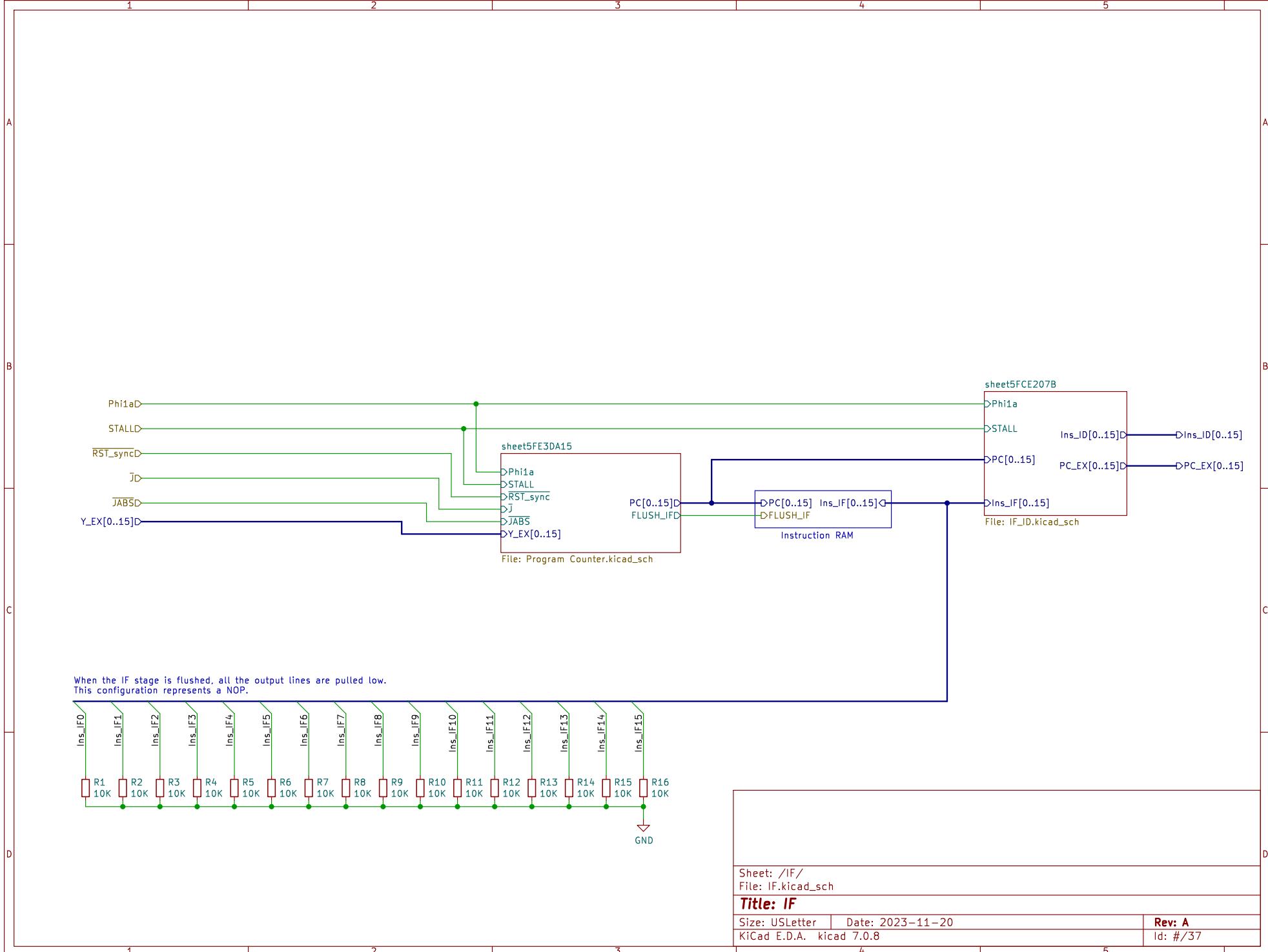


Sheet: /Operand Forwarding B/
File: OperandForwardingB.kicad_sch

Title:

Size: A4 | Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

Rev: A
Id: #/37



1 2 3 4 5

A 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.

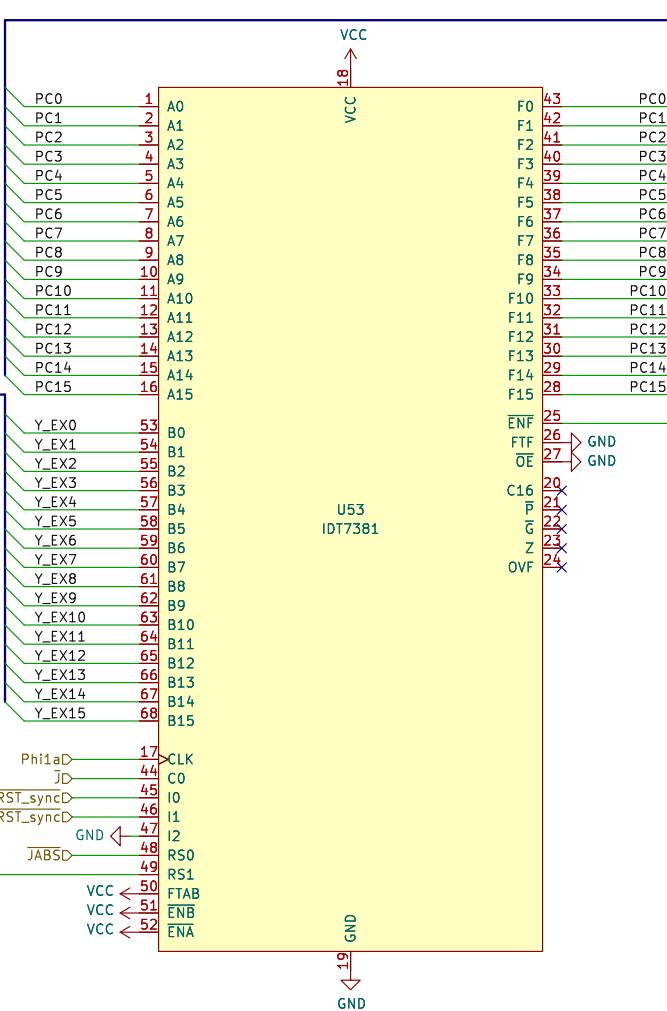
During reset, set the ALU to I2=0, I1=0, IO=0. This causes the ALU to compute a zero and latch it in A on the rising edge of the clock regardless of the value of the A and B inputs. This resets the program counter to zero.

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

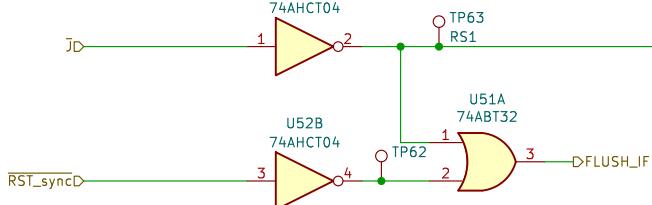
When performing a relative jump, set the ALU to RS1=1, RS0=1, I2=0, I1=1, IO=1, CO=0. The ALU computes $F = A + B$. Since the B port gets its value from the Y result of the EX stage, this computes $PC = PC + offset$.

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

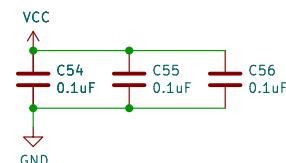
B Y_EX[0..15]D



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



Flush the IF stage on Jump and on Reset.
The flush during Reset ensures the pipeline is filled with NOPs.



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.kicad_sch

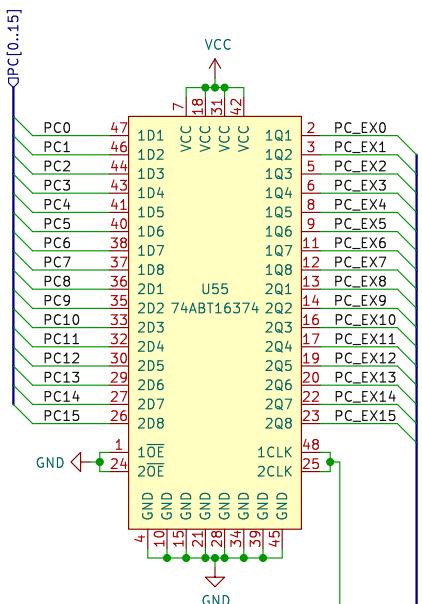
Title: Program Counter

Size: USLetter Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

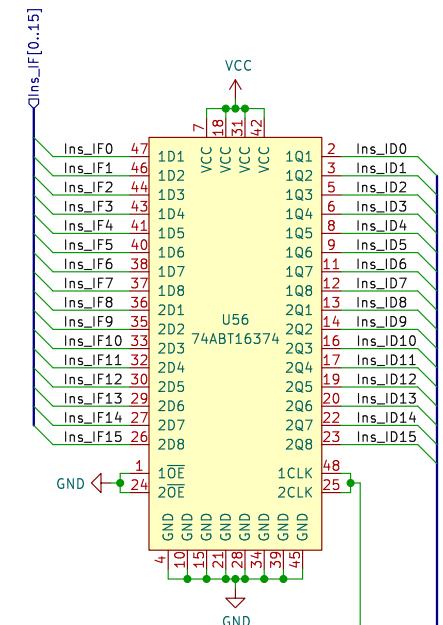
Rev: A
Id: #/37

1 2 3 4 5

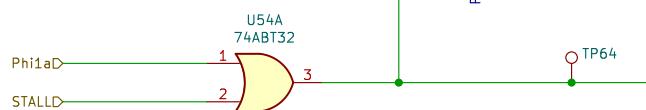
A



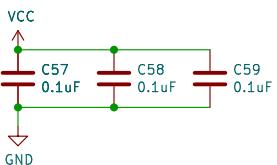
B



C



D



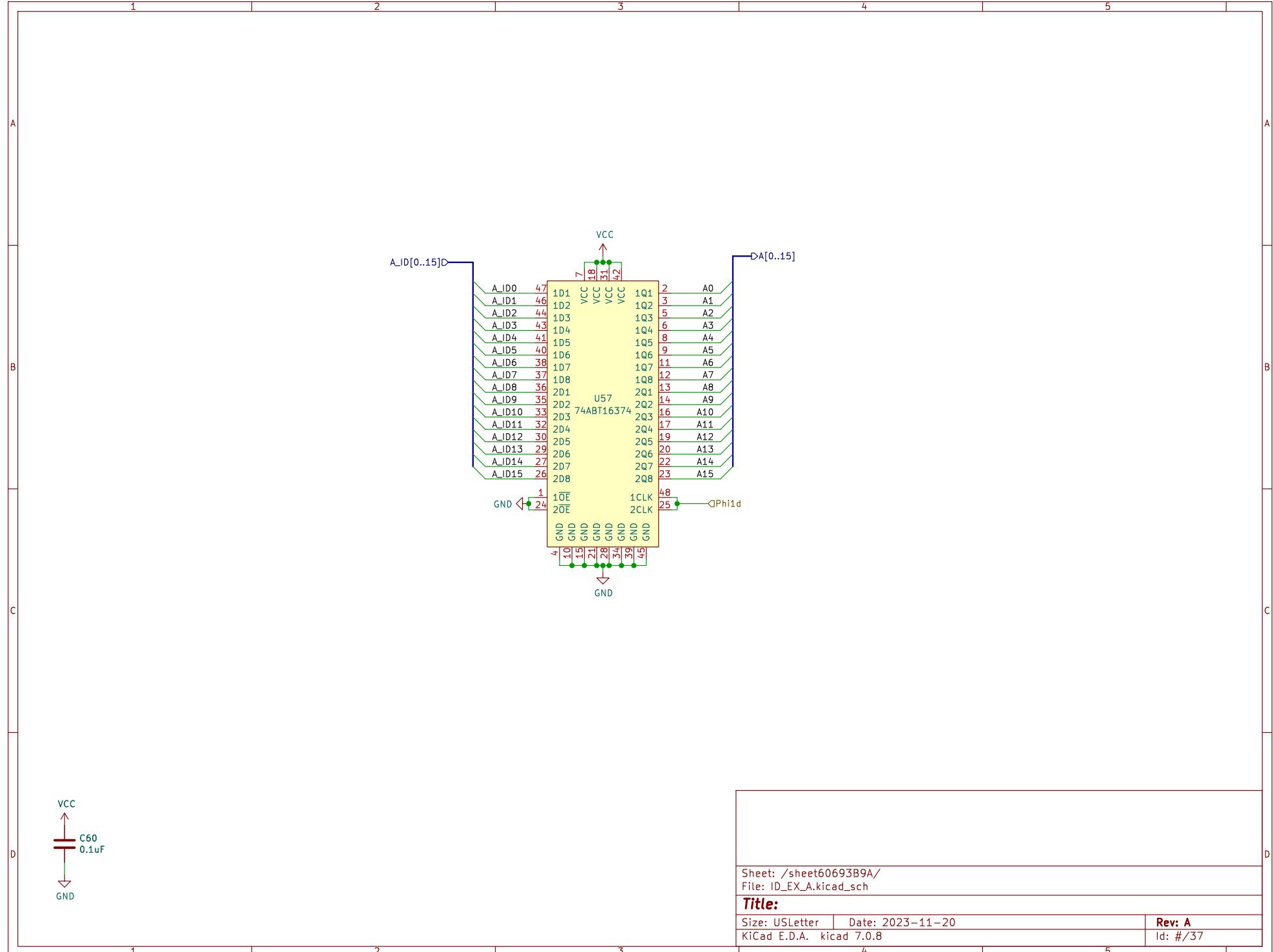
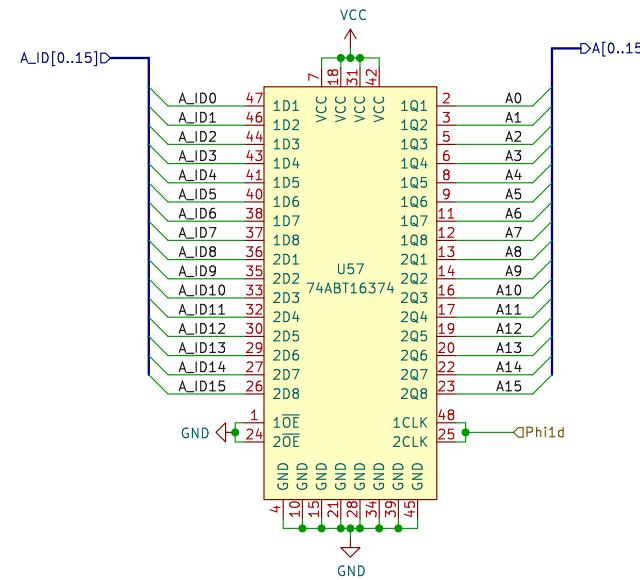
Interstage pipeline registers between IF and ID

Sheet: /IF/sheet5FCE207B/
File: IF_ID.kicad_sch

Title: IF/ID

Size: USLetter Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

Rev: A
Id: #/37

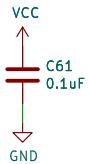
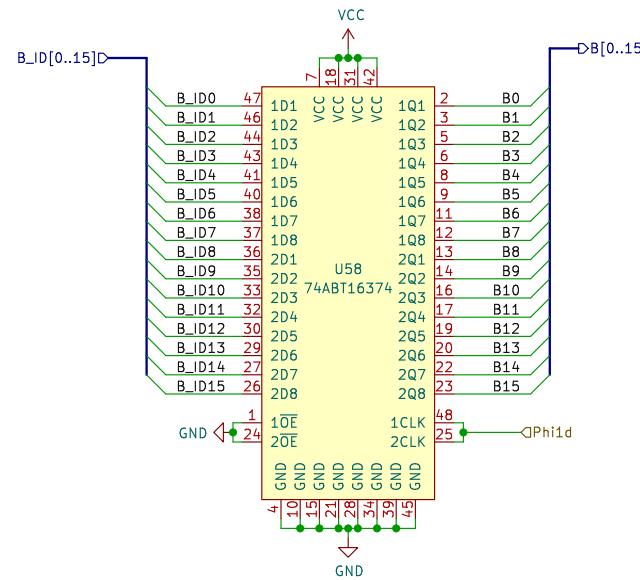


Sheet: /sheet60693B9A/
File: ID_EX_A.kicad_sch

Title:

Size: USLetter | Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

Rev: A
Id: #/37

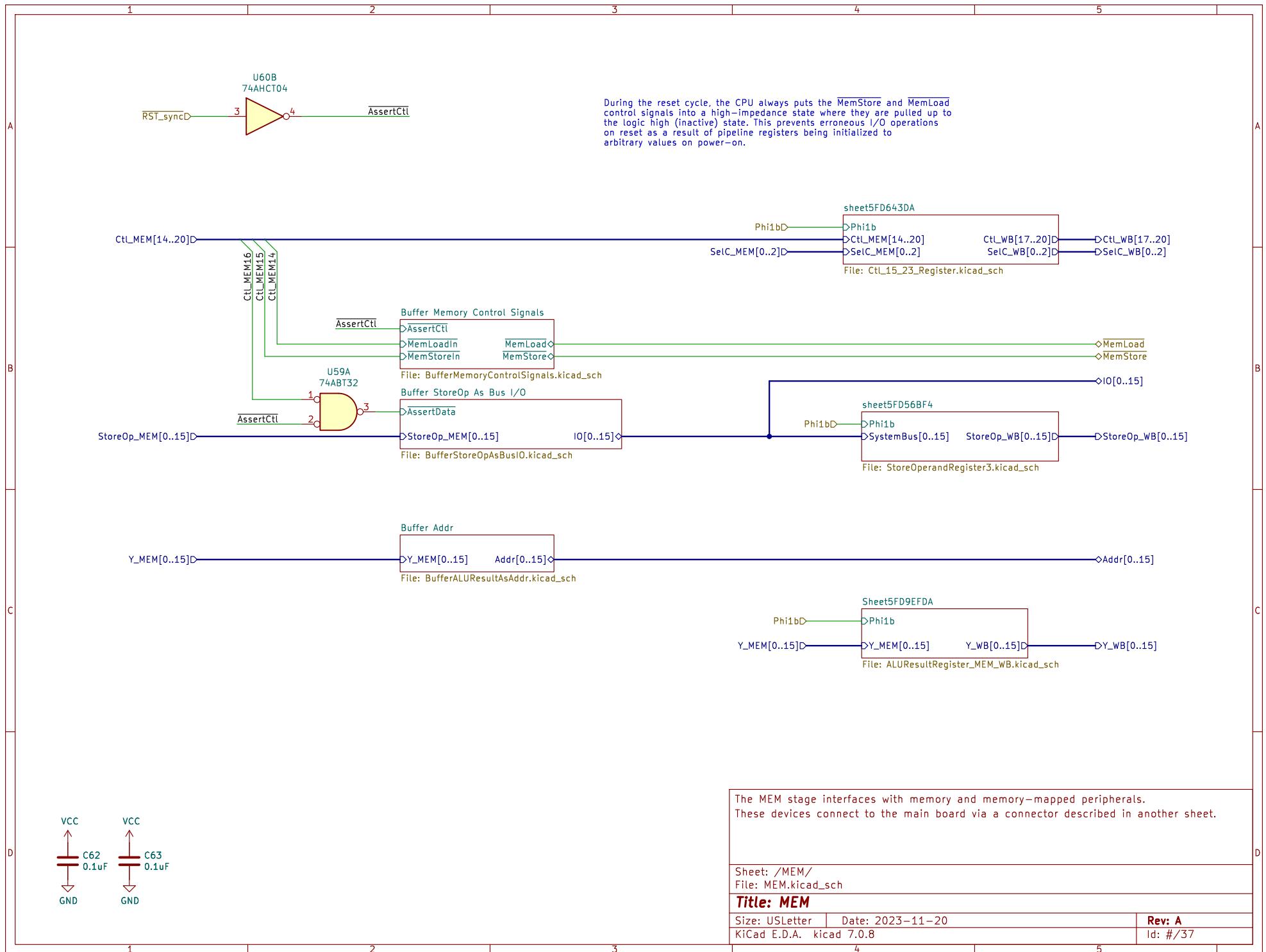


Sheet: /sheet60693B9B/
File: ID_EX_B.kicad_sch

Title:

Size: USLetter | Date: 2023-11-20
KiCad E.D.A. kicad 7.0.8

Rev: A
Id: #/37



1 2 3 4 5 6



A

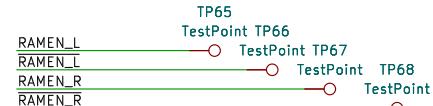
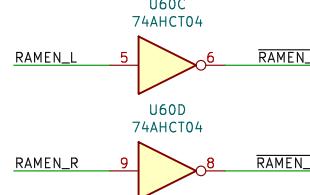
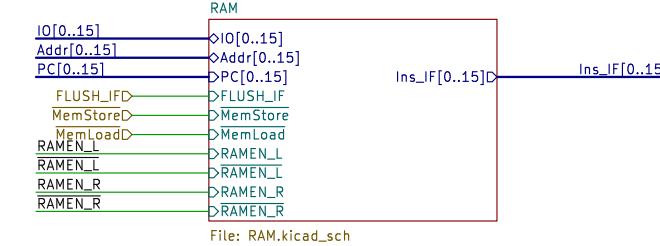
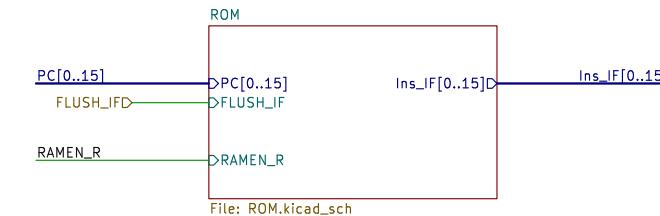
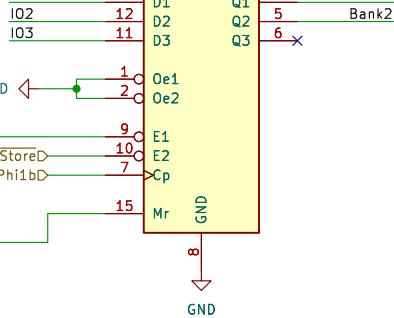
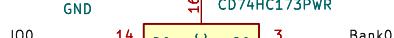
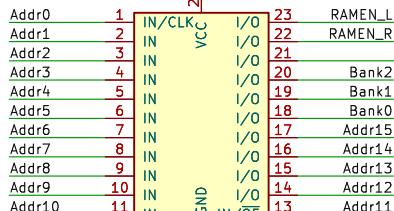
The Bank register resets to zero when the computer resets. Set the bank register by writing a word to address 0xffff.

When the bank register is zero, the instruction fetch unit is fed by ROM.

When the bank register is not zero, the instruction fetch unit is fed by The right side of RAM. This allows the computer to be reprogrammed at run time.

When the bank register is zero or one then the left side of RAM is enabled on the system bus.

The PLD allows some other combinations of memory address and bank to be mapped to RAM during load/store over the memory bus. This mapping may be reprogrammed later to accomodate new devices added to the bus. If there are no other devices on the bus then a good default mapping would be to map all banks and memory ranges to RAM.



Sheet: /Memory/
 File: Memory.kicad_sch

Title:

Size: A4 | Date: 2023-11-20
 KiCad E.D.A. kicad 7.0.8

Rev: A
 Id: #/37

1 2 3 4 5 6

