CSCI 150
Introduction to Digital and Computer System Design
Lecture 3: Combinational Logic Design IV

Jetic Gū

Overview

- Focus: Logic Functions
- Architecture: Combinatory Logical Circuits
- Textbook v4: Ch3 3.6, 3.7; v5: Ch3 3.6, 3.7
- Core Ideas:
    1. Encoder
    2. Multiplexer

P0
Review

Systematic Design Procedures

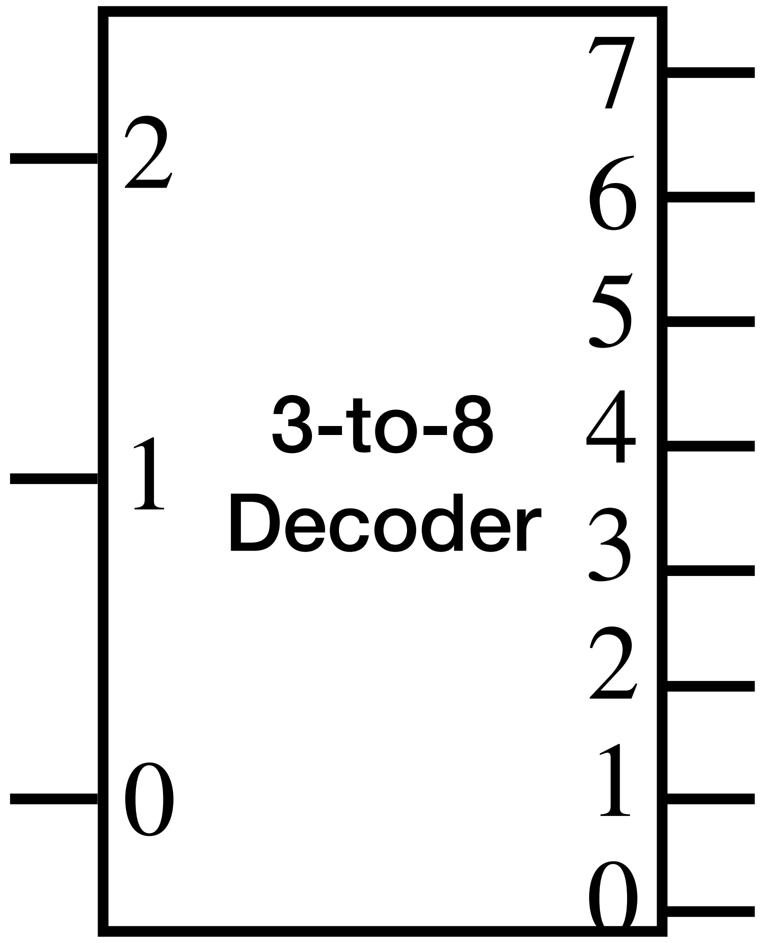
1. **Specification**: Write a specification for the circuit
2. **Formulation**: Derive relationship between inputs and outputs of the system e.g. using truth table or Boolean expressions
3. **Optimisation**: Apply optimisation, minimise the number of logic gates and literals required
4. **Technology Mapping**: Transform design to new diagram using available implementation technology
5. **Verification**: Verify the correctness of the final design in meeting the specifications

Review

P0
Review

Functional Components

- Value-Fixing, Transferring, Inverting, Enabler
- Decoder
  - Input: $A_0A_1\ldots A_{n-1}$
  - Output: $D_0D_1\ldots D_{2^n-1}, D_i = m_i$



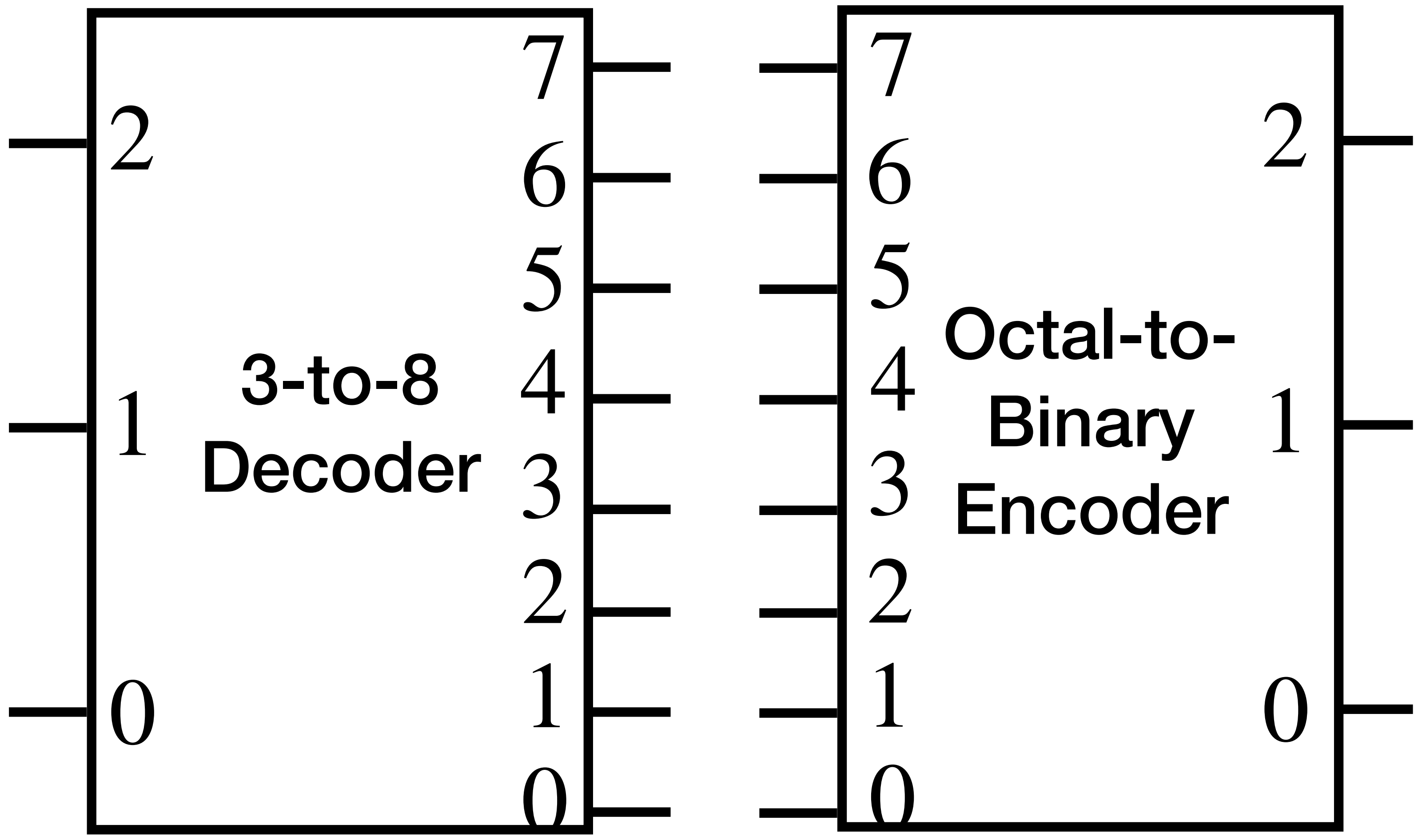
Review

Encoder

Wait, didn’t we just covered this?
Oh, that’s decoder

Encoder

- Inverse operation of a decoder
- 2^n inputs, only one is giving positive input[1]
- n outputs

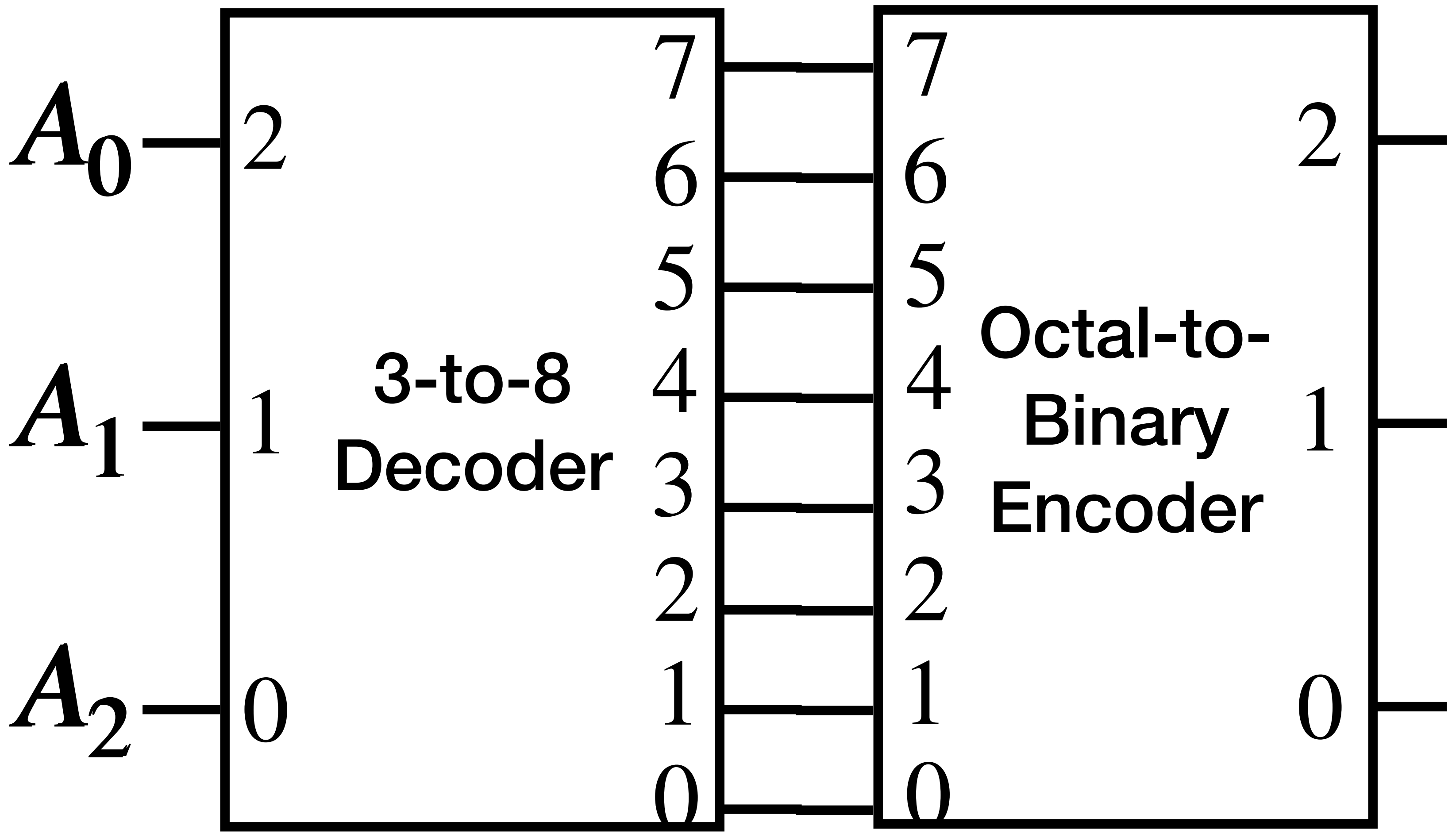


[1] In reality, could be less

P1
Encoder

Encoder

- Inverse operation of a decoder
- 2^n inputs, only one is giving positive input[1]
- n outputs



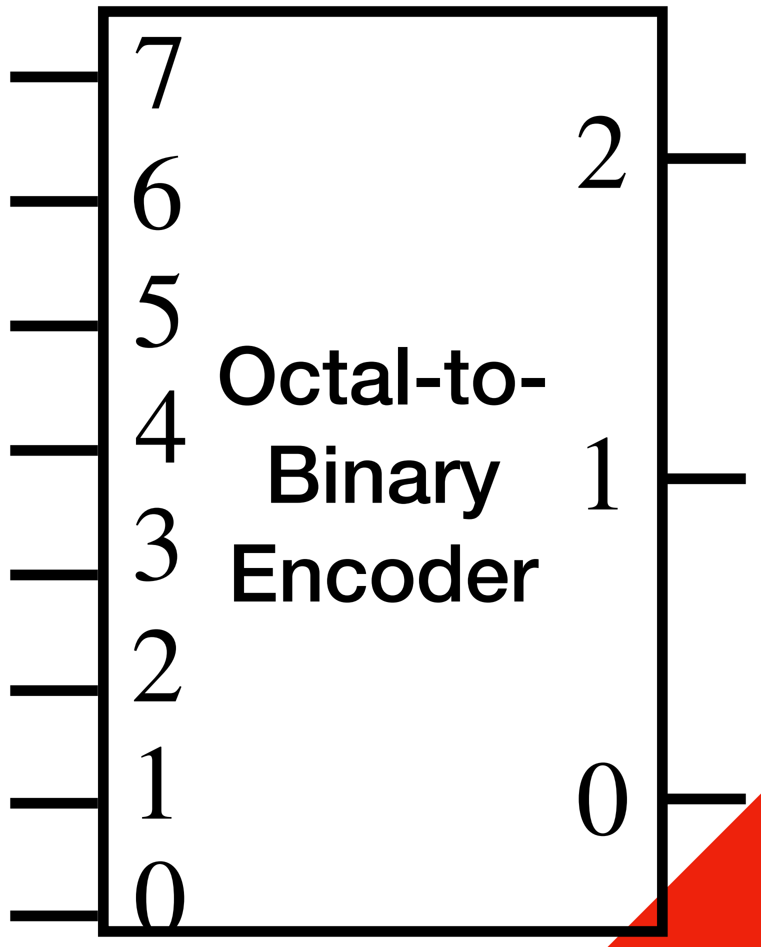
Concept

[1] In reality, could be less

Encoder

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	A_2	A_1	A_0
							1	0	0	0
						1		0	0	1
					1			0	1	0
				1				0	1	1
			1					1	0	0
		1						1	0	1
	1							1	1	0
1								1	1	1

$$A_0 = D_1 + D_3 + D_5 + D_7$$
$$A_1 = D_2 + D_3 + D_6 + D_7$$
$$A_2 = D_4 + D_5 + D_6 + D_7$$



Concept

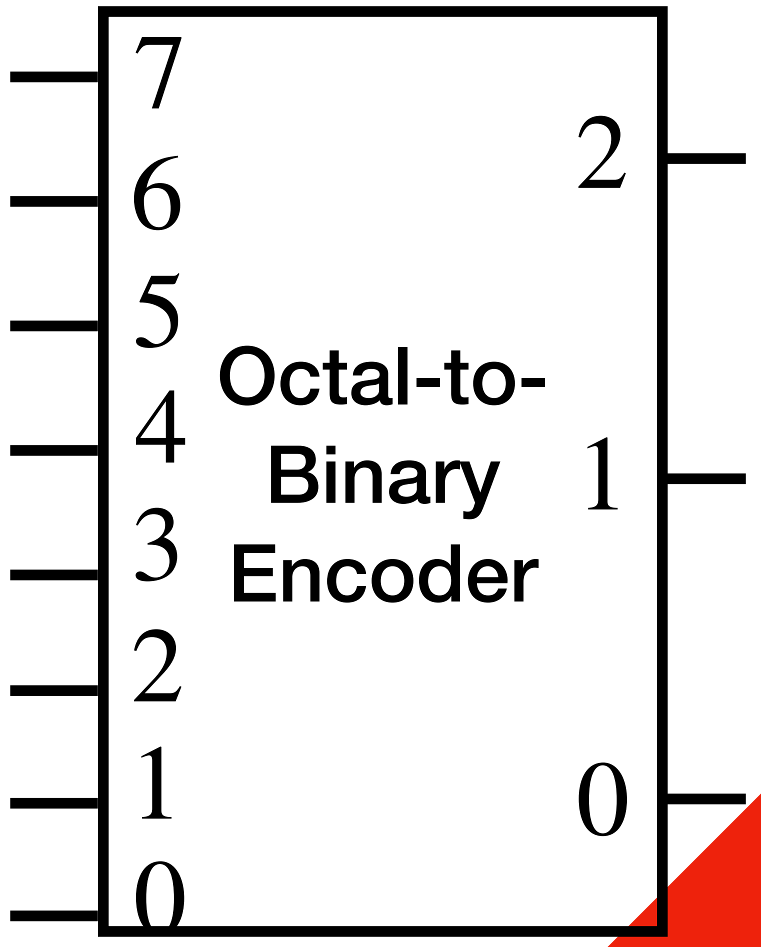
Encoder

$$A_0 = D_1 + D_3 + D_5 + D_7$$

$$A_1 = D_2 + D_3 + D_6 + D_7$$

$$A_2 = D_4 + D_5 + D_6 + D_7$$

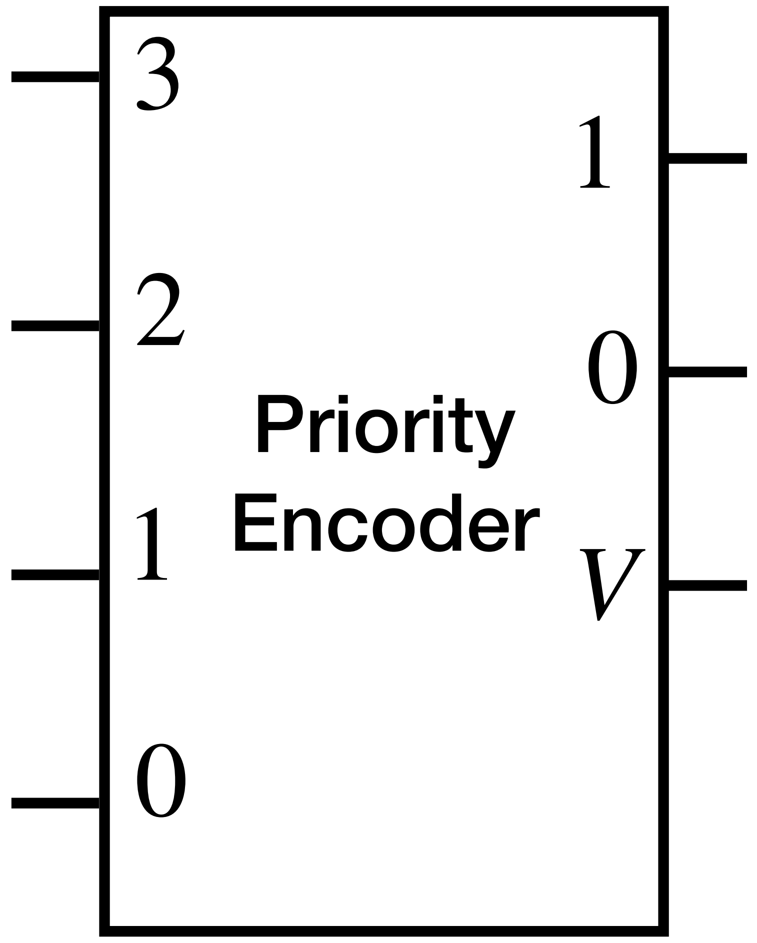
- What happens if the inputs are all 0s?
- What happens if the inputs include multiple 1s?



Concept

Priority Encoder

- Additional Validity Output V
  - Indicating whether the input is valid (contains 1)
- Priority
  - Ignores $D_{<i}$ if $D_i = 1$

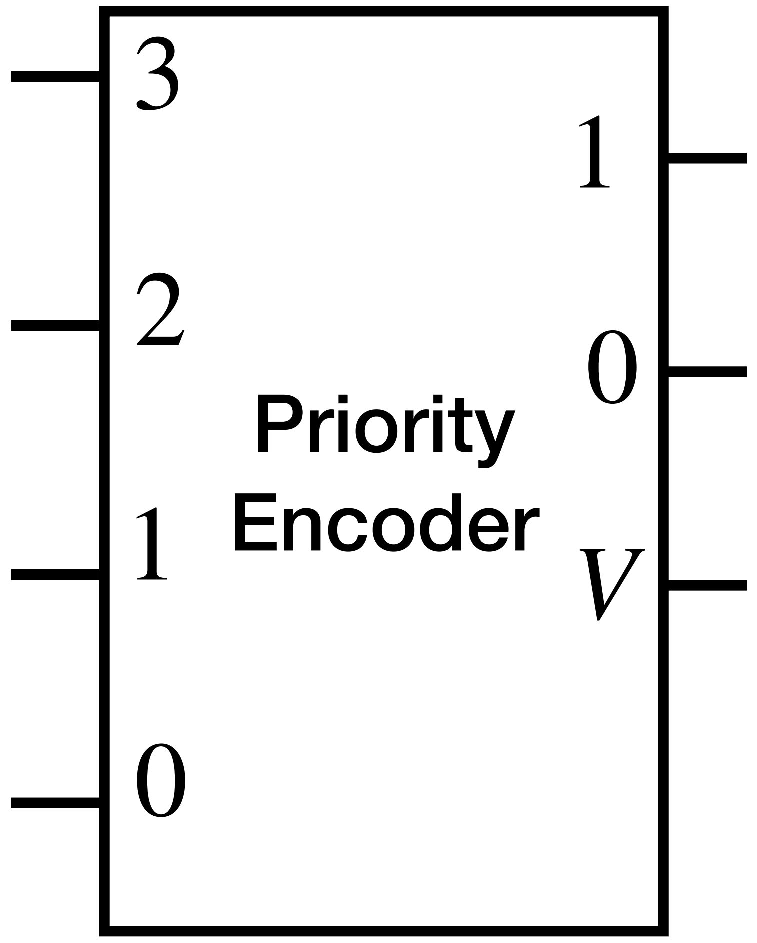


Concept

P1
Encoder

Priority Encoder

D_3	D_2	D_1	D_0	A_1	A_0	V
0	0	0	0	X	X	0
0	0	0	1	0	0	1
0	0	1	X	0	1	1
0	1	X	X	1	0	1
1	X	X	X	1	1	1



Concept

Priority Encoder

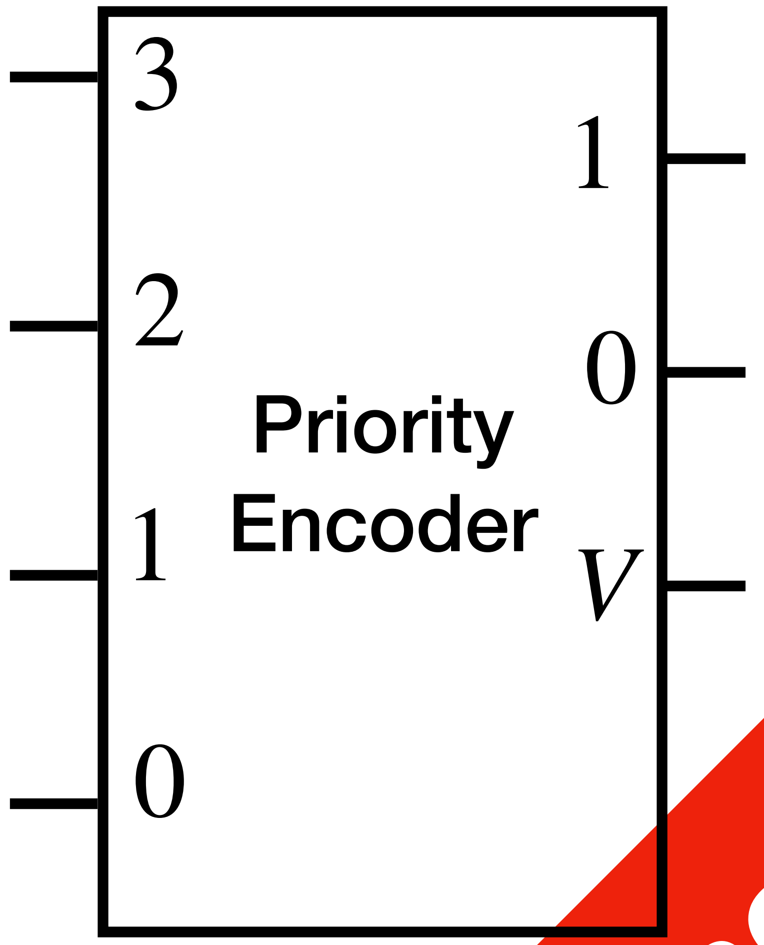
D_3	D_2	D_1	D_0	A_1	A_0	V
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	X	0	1	1
0	1	X	X	1	0	1
1	X	X	X	1	1	1

$$V = D_3 + D_2 + D_1 + D_0$$

$$A_1 = D_3 + \overline{D_3}D_2 = D_2 + D_3$$

$$A_0 = \overline{D_3}\,\overline{D_2}D_1 + D_3$$

$$= \overline{D_2}D_1 + D_3$$



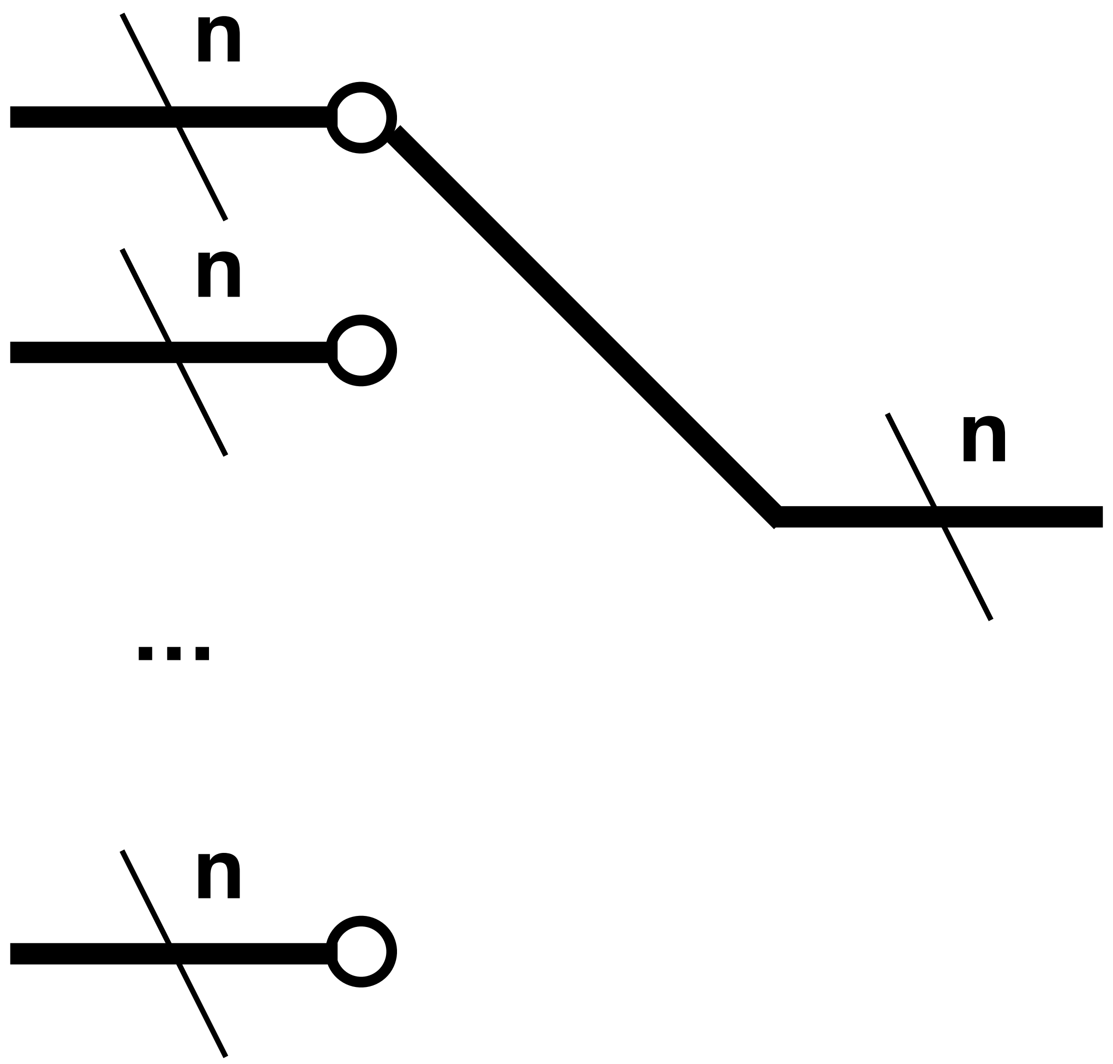
Multiplexer

Switch Modes

Summary

Multiplexer

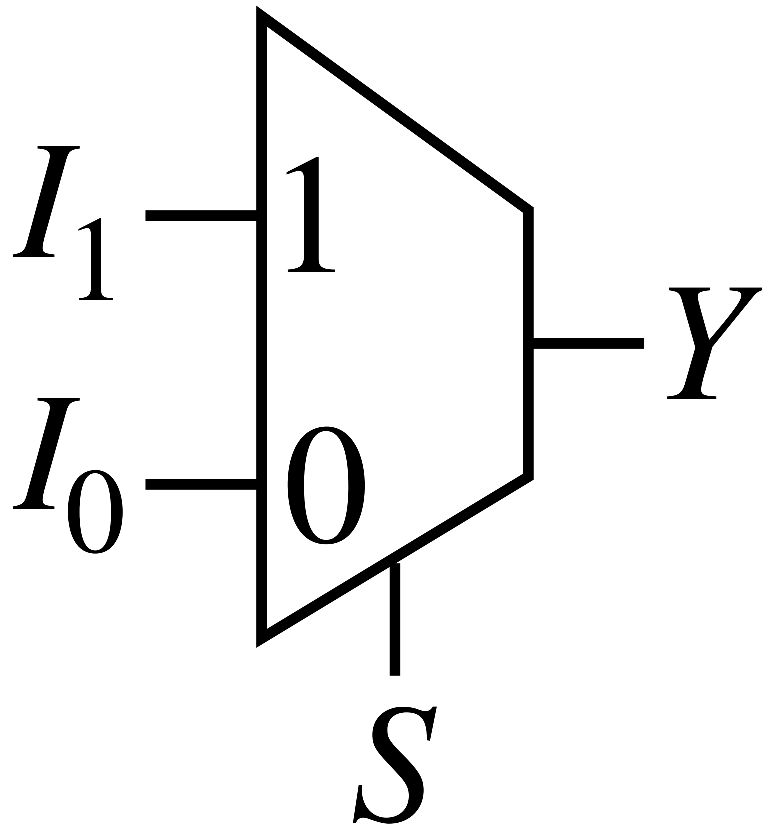
- Multiple n-variable input vectors
- Single n-variable output vector
- Switches: which input vectors to output



Concept

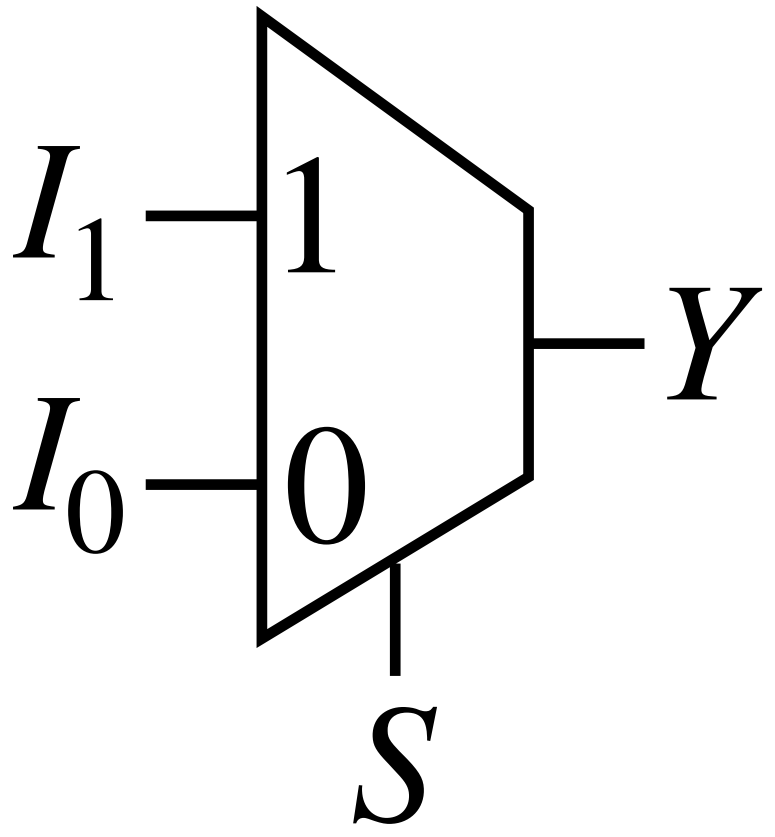
Single-Bit 2-to-1 Multiplexer

- 2 single-bit inputs
- 1 single-bit output
- 1-bit switch



Single-Bit 2-to-1 Multiplexer

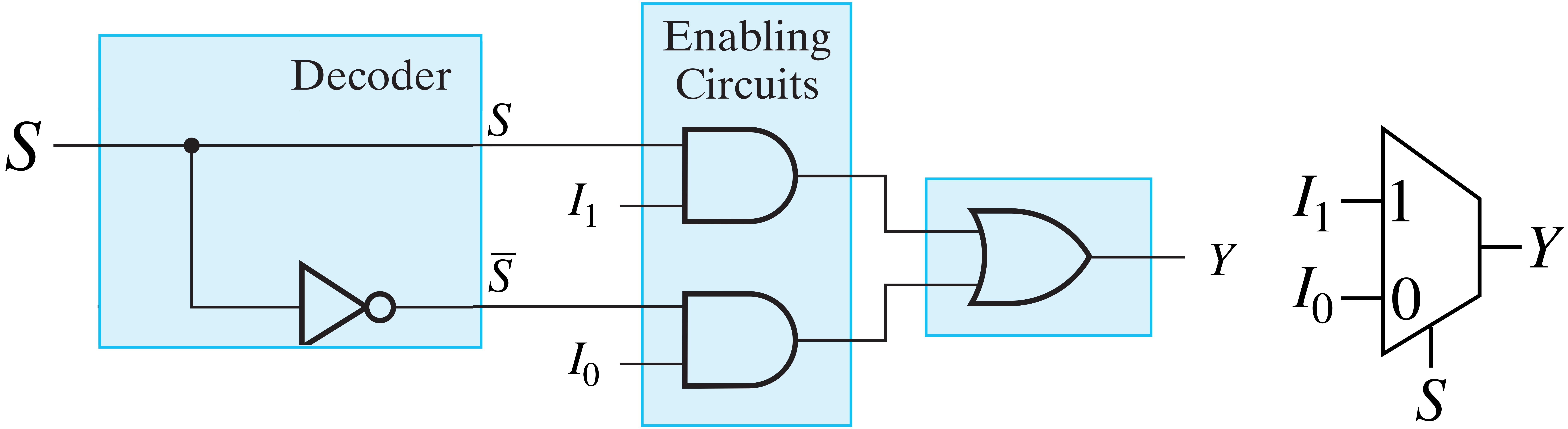
S	I_0	I_1	Y
0	0	X	0
0	1	X	1
1	X	0	0
1	X	1	1



Single-Bit 2-to-1 Multiplexer

Technology

- 1 x 1-to-2 Decoder
- 2 x 1-bit Enabler
- 1 x 2-input OR Gate

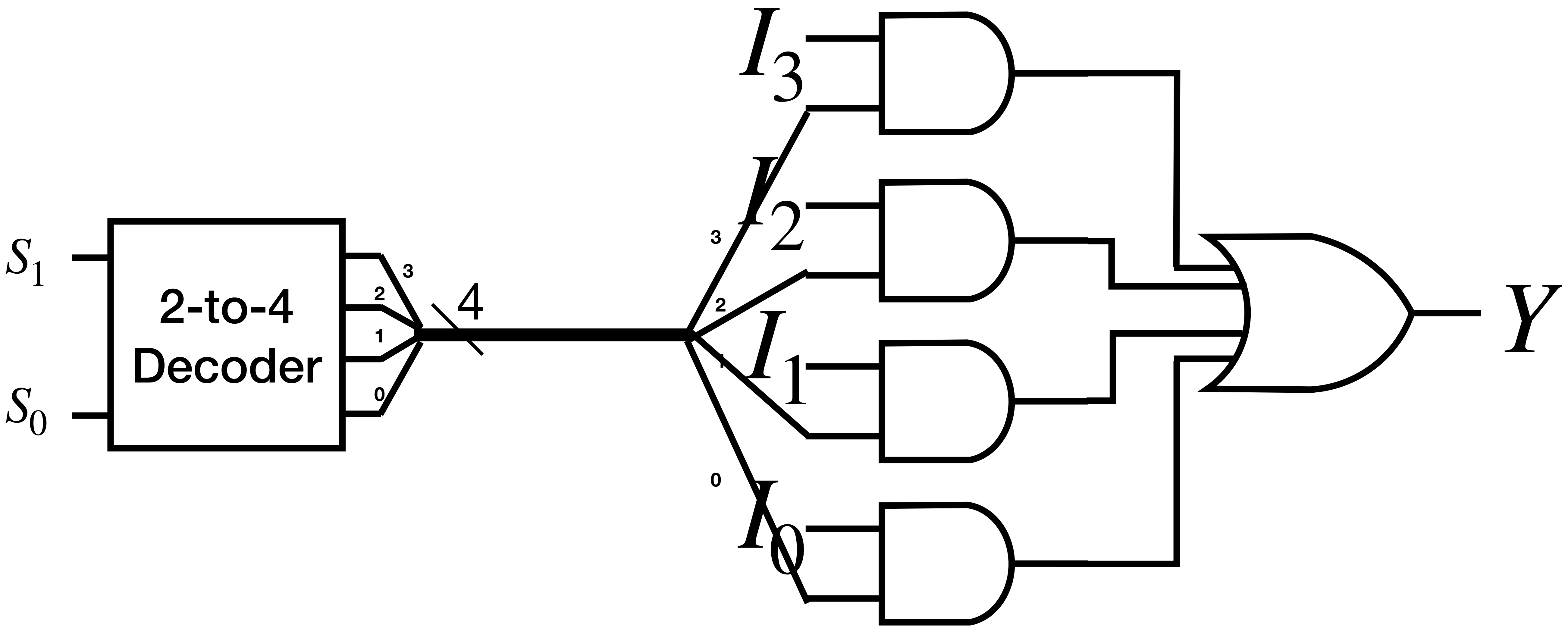


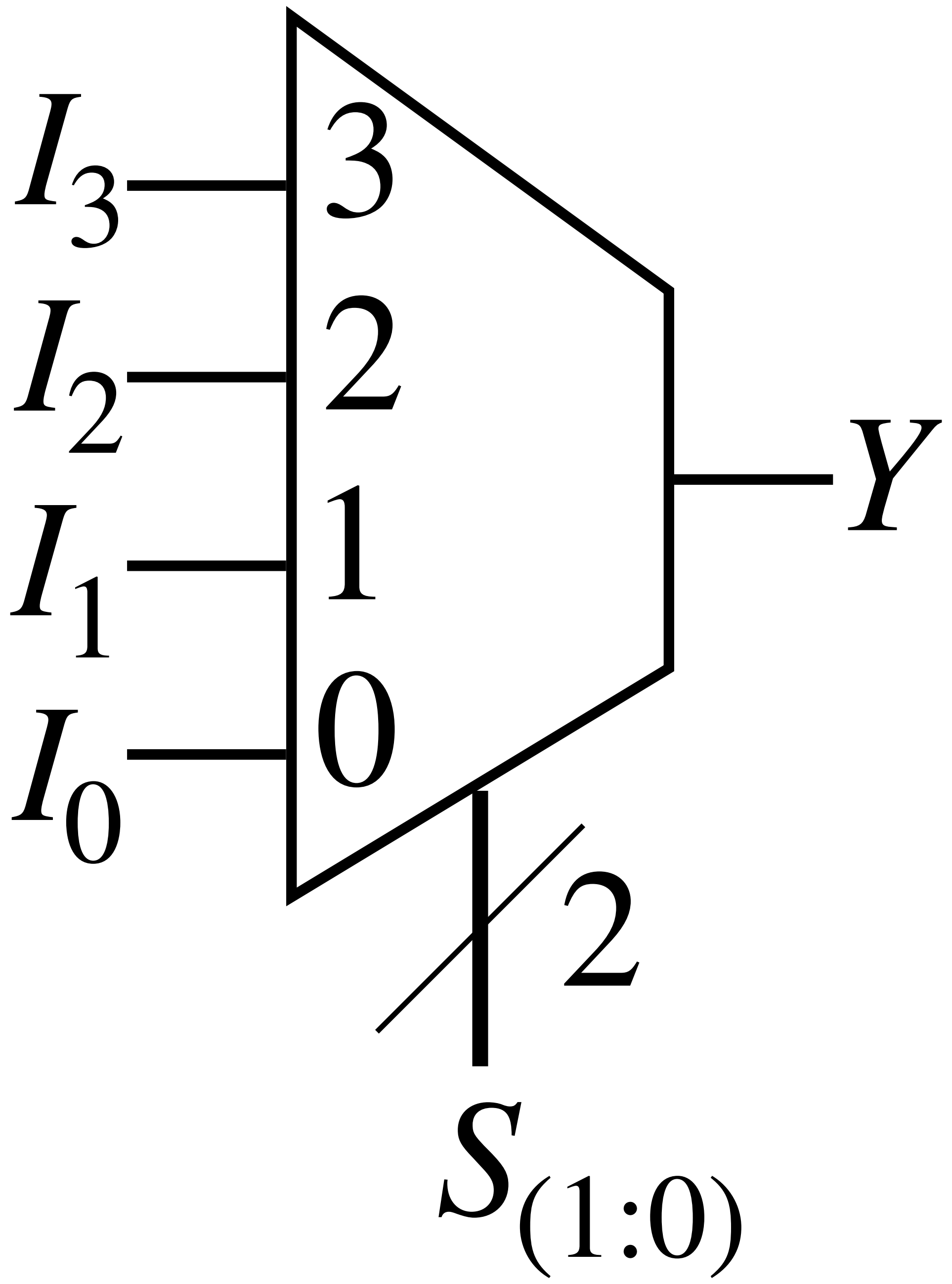
P2
Multiplexer

Single-Bit 4-to-1 Multiplexer

Technology

- 1 x 2-to-4 Decoder
- 4 x 1-bit Enabler
- 1 x 4-input OR Gate





Concept

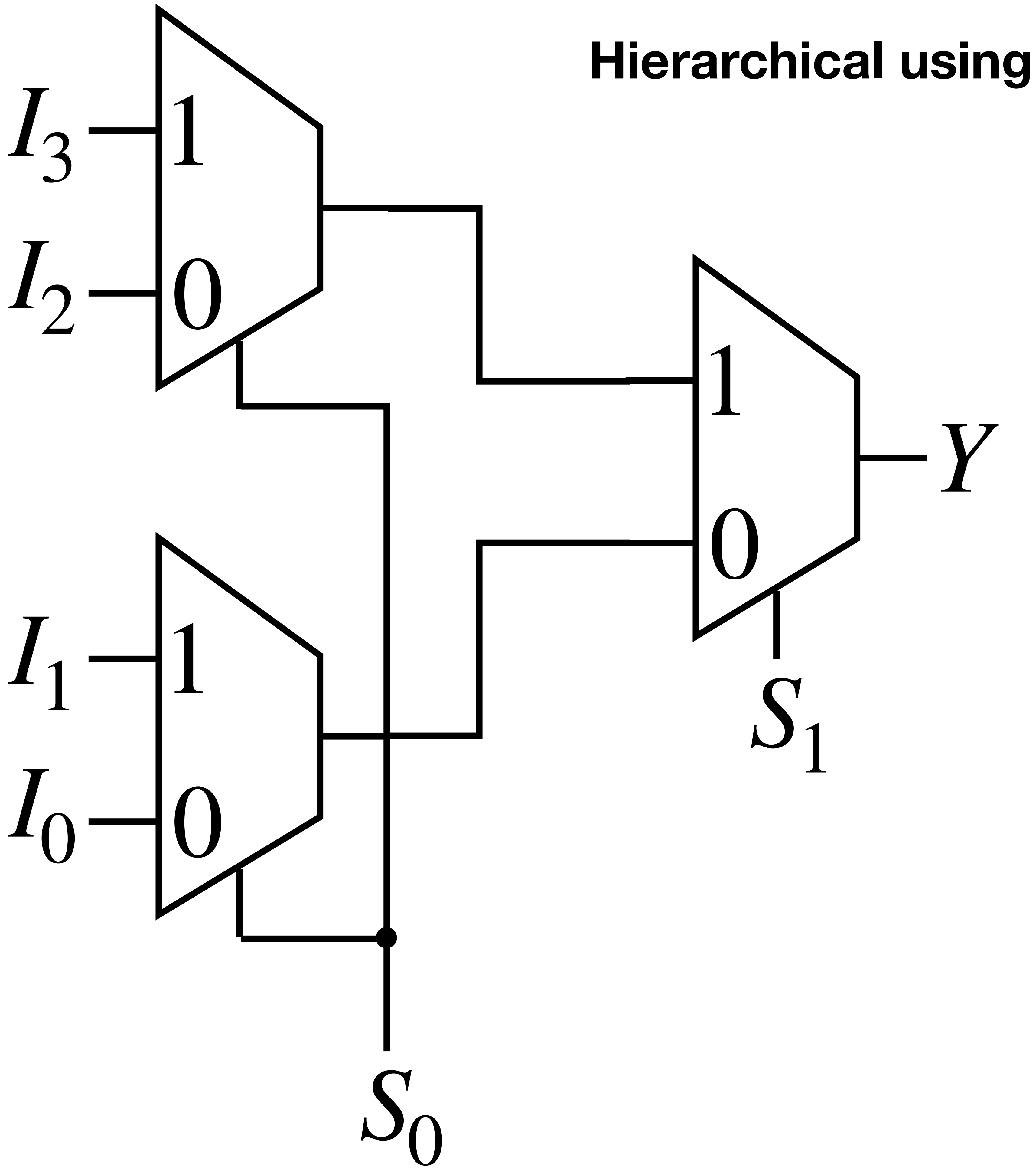
P2
Multiplexer

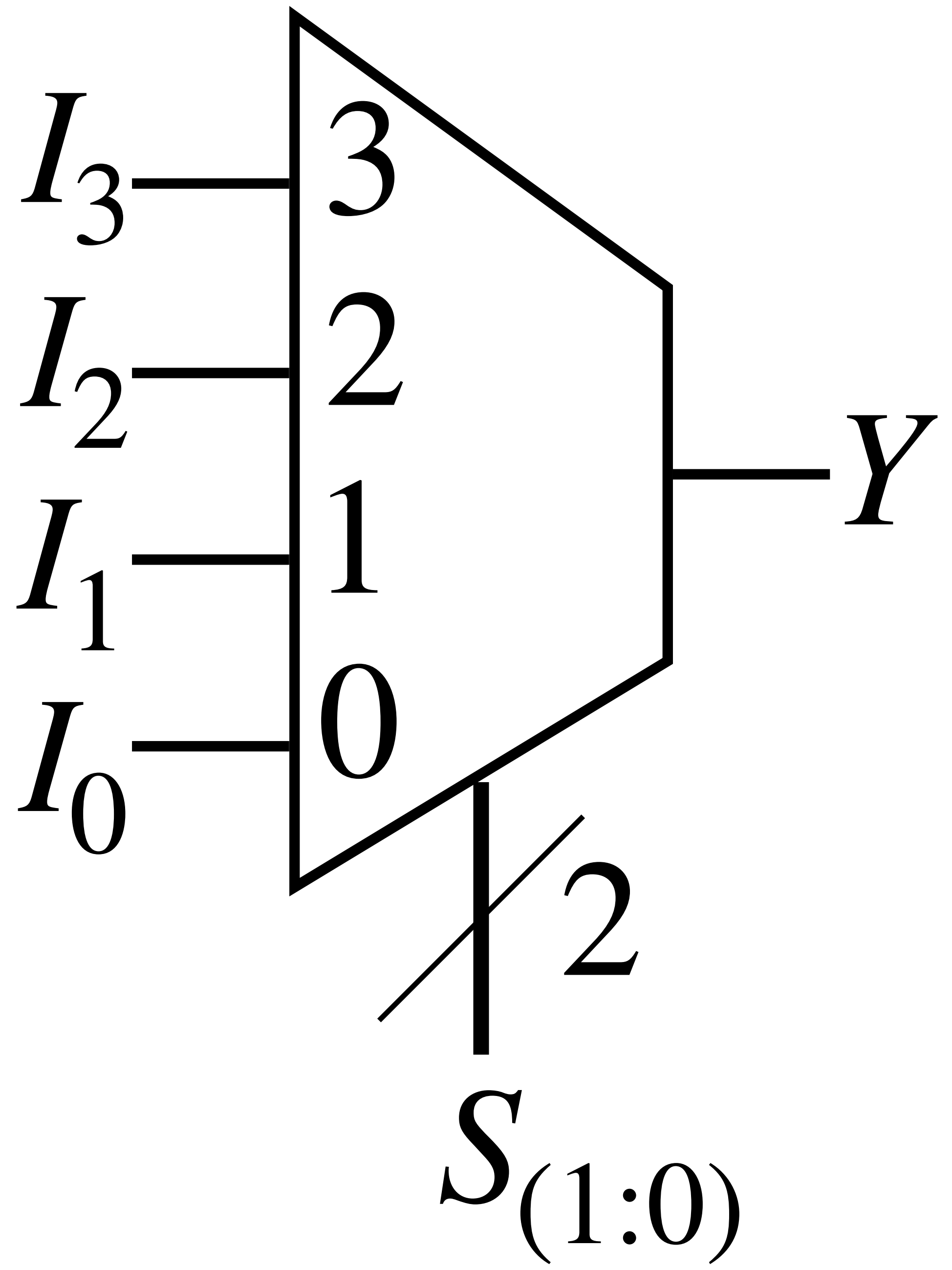
Single-Bit 4-to-1 Multiplexer

Technology

- 3 x 1bit 2-to-1 MUX

Hierarchical using MUX





Concept

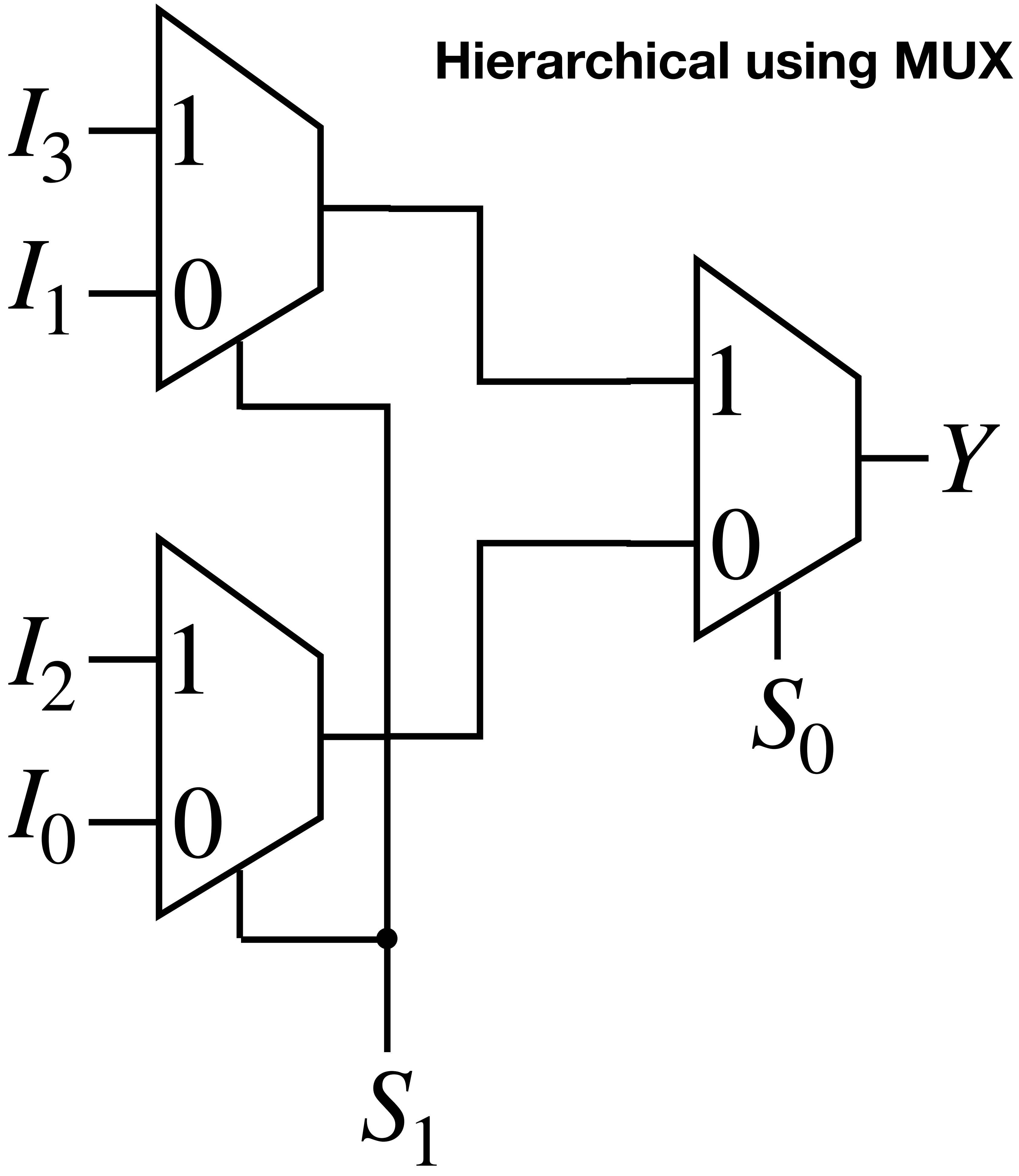
P2
Multiplexer

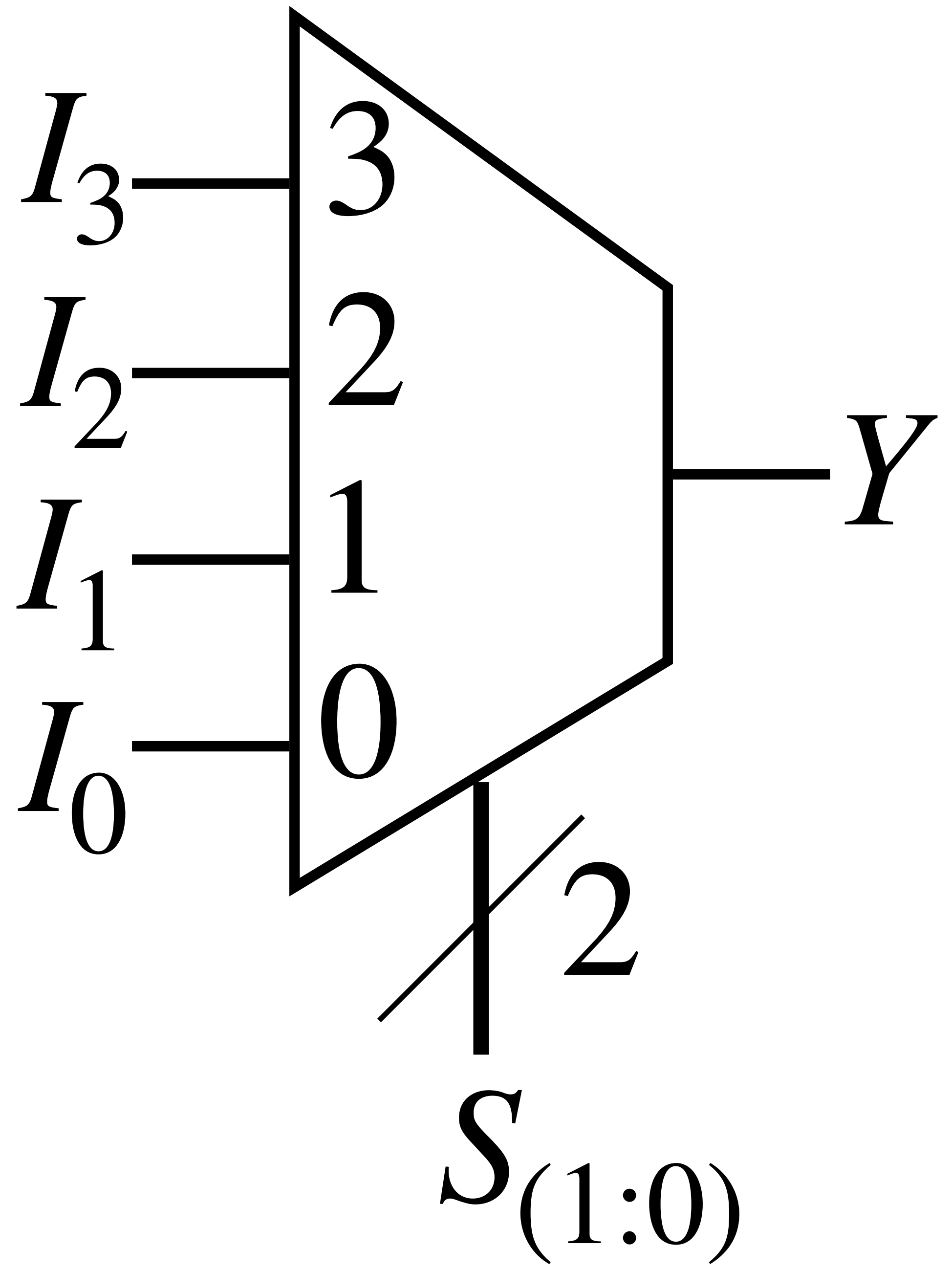
Single-Bit 4-to-1 Multiplexer

Technology

- 3 x 1bit 2-to-1 MUX

Hierarchical using MUX (Alternative)





Concept

P3
LogicWorks

Circuit Drawing Time!

For Lab 2

Summary

P3
LogicWorks

Last Circuit Drawing Practice

1. Sub-circuit
2. Implementing 2-to-4 Decoder using drawing tools
3. Implementing 3-to-8 Decoder using 2-to-4 Decoders
4. Implementing 4-to-16 Decoder using 2-to-4 Decoders or 3-to-8 Decoders
5. Implementing Octal-to-3 Priority Encoder
6. Implementing 1bit 4-to-1 Multiplexer using 2-to-4 Decoder
7. Implementing 4bit 4-to-1 Multiplexer using 1bit 4-to-1 Multiplexers

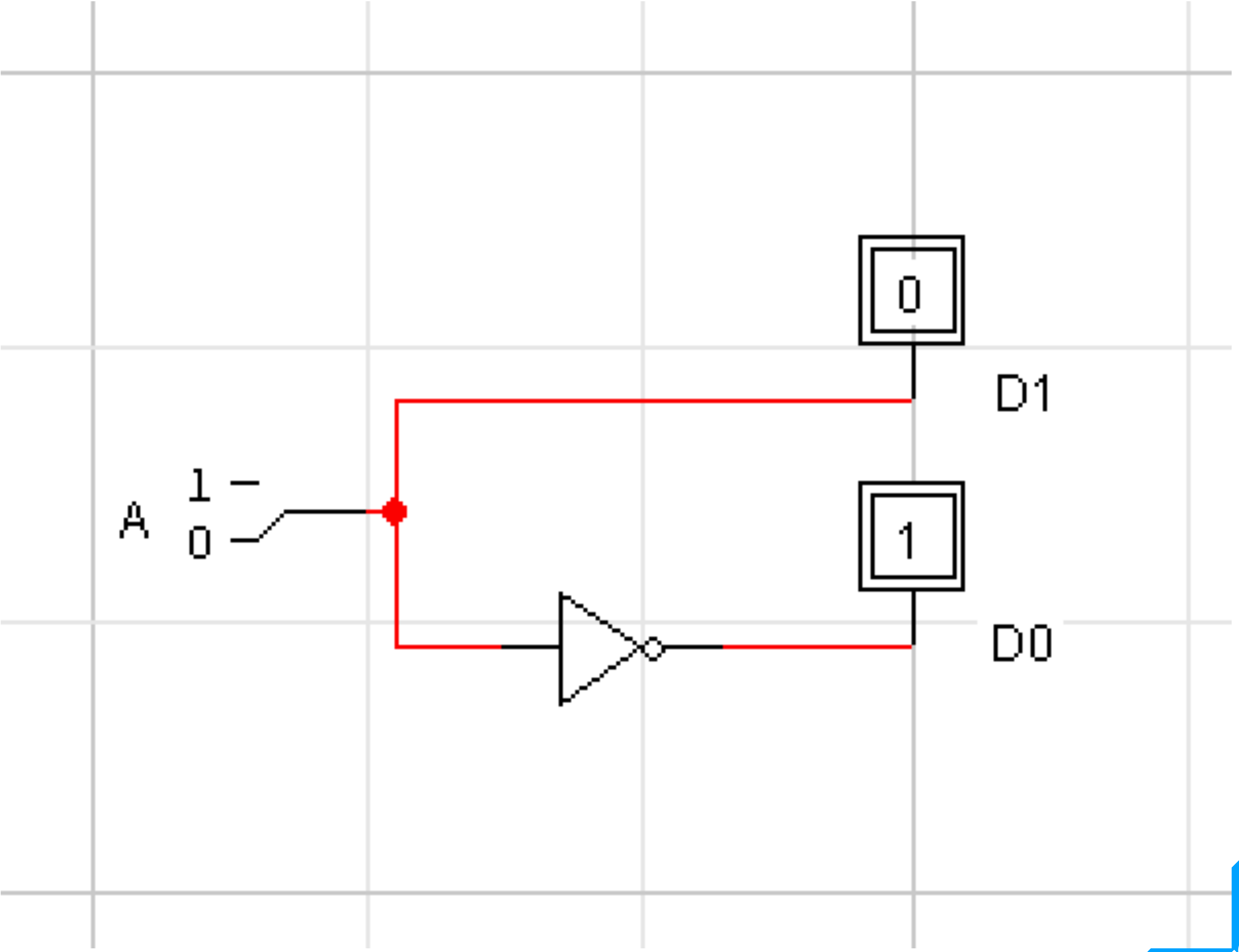
[1] You will be reusing these designs in later lectures and assignments

Concept

P3
LogicWorks

Sub-circuit

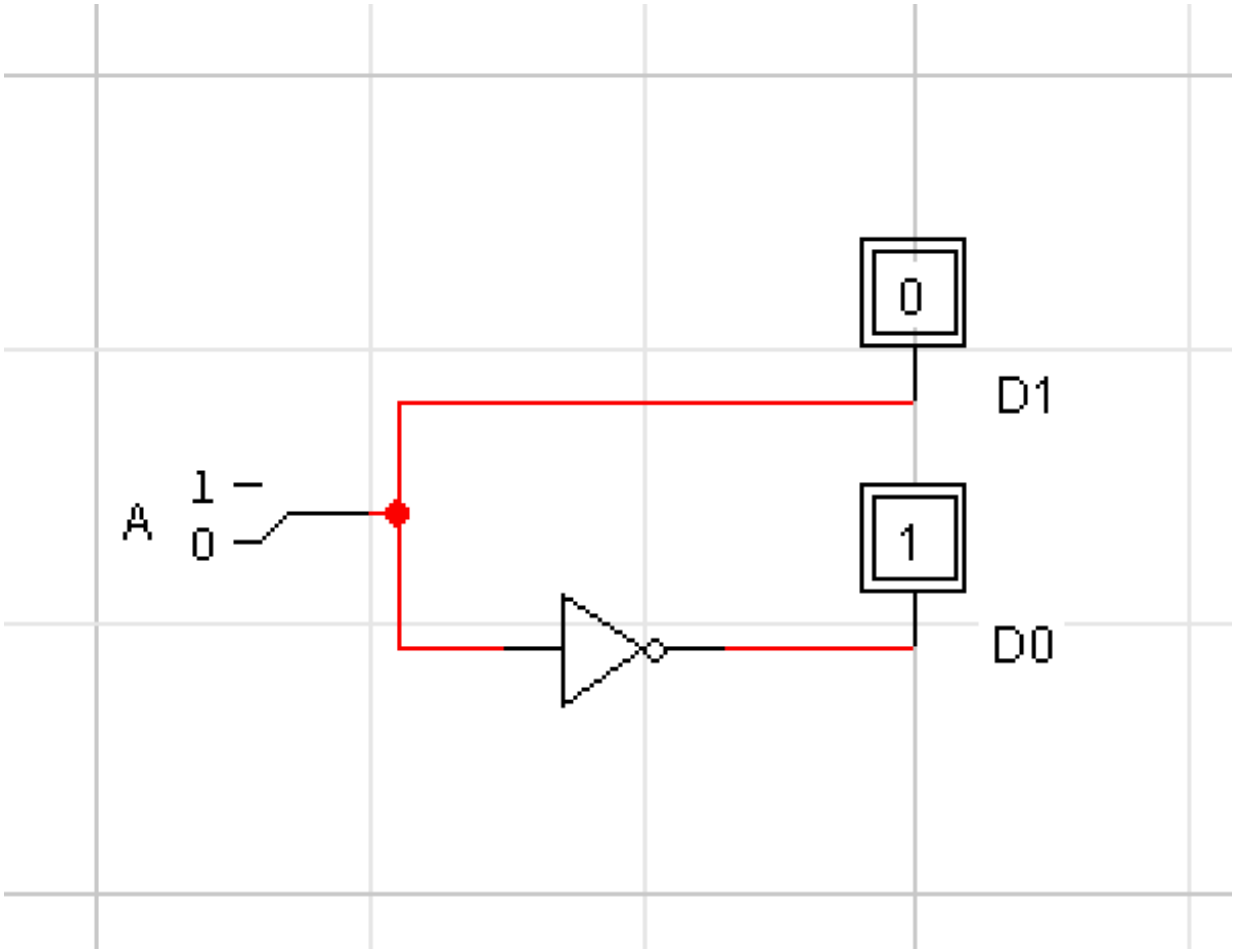
1. Implement a 1-to-2 Decoder
2. Correctly label ALL I/O devices
3. Test the circuit, make sure it's actually working
4. Only when the above are done, should you proceed

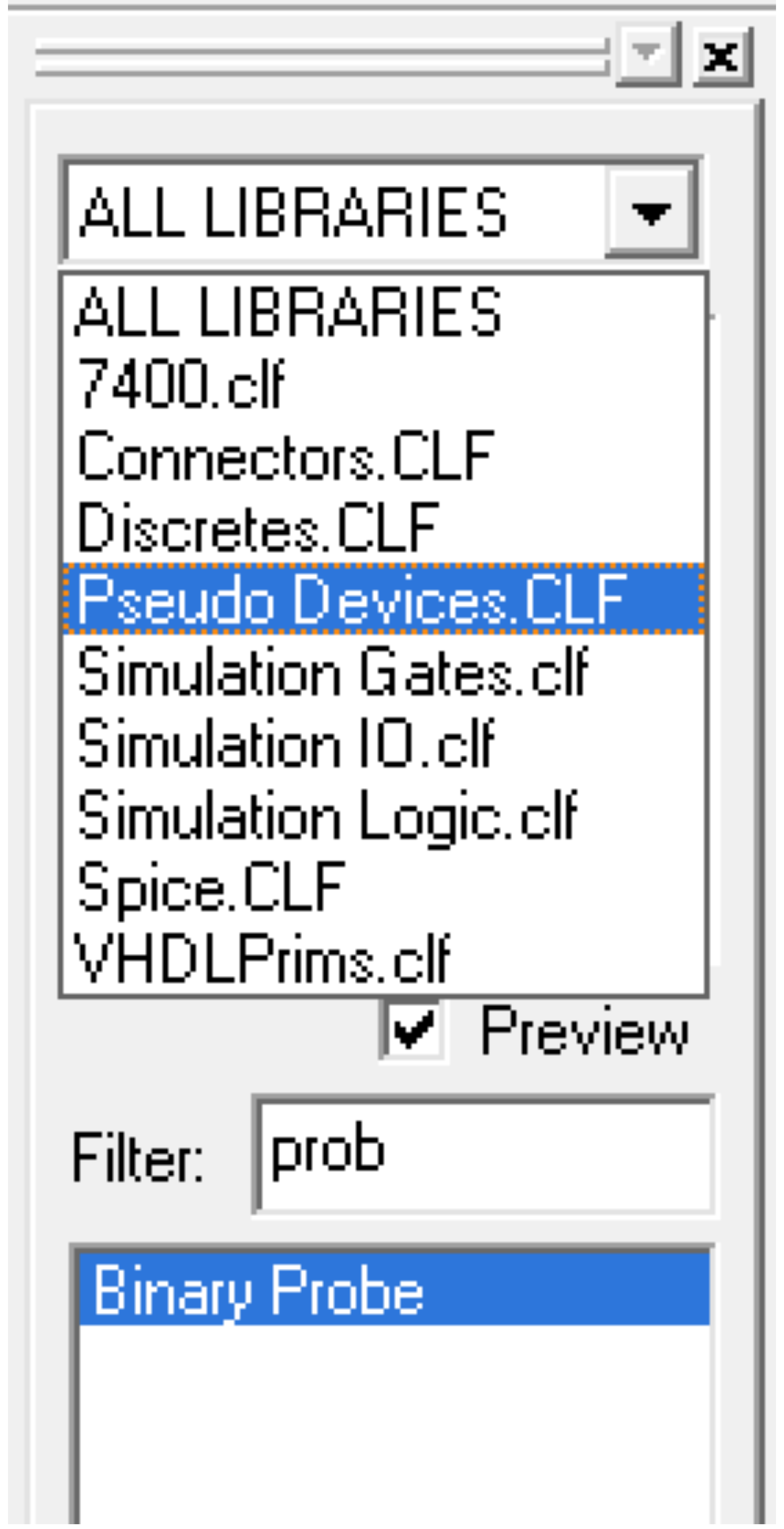


Technical

Before we get started, make sure you have implemented a 1-to-2 Decoder using switches and probes correctly

Sub-circuit




ALL LIBRARIES

ALL LIBRARIES
7400.clf
Connectors.CLF
Discretes.CLF
Pseudo Devices.CLF
Simulation Gates.clf
Simulation IO.clf
Simulation Logic.clf
Spice.CLF
VHDLPrims.clf

☑ Preview

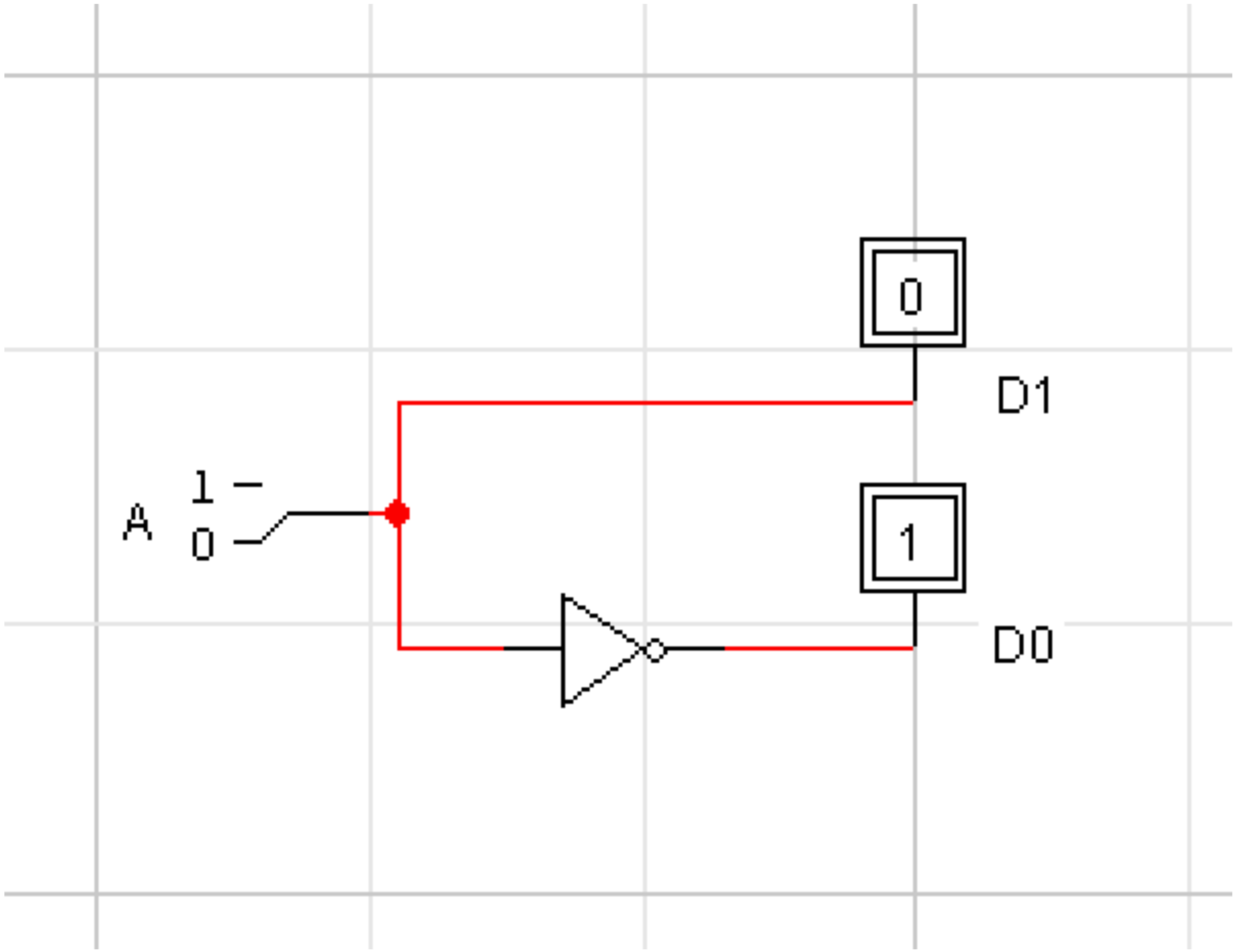
Filter: prob

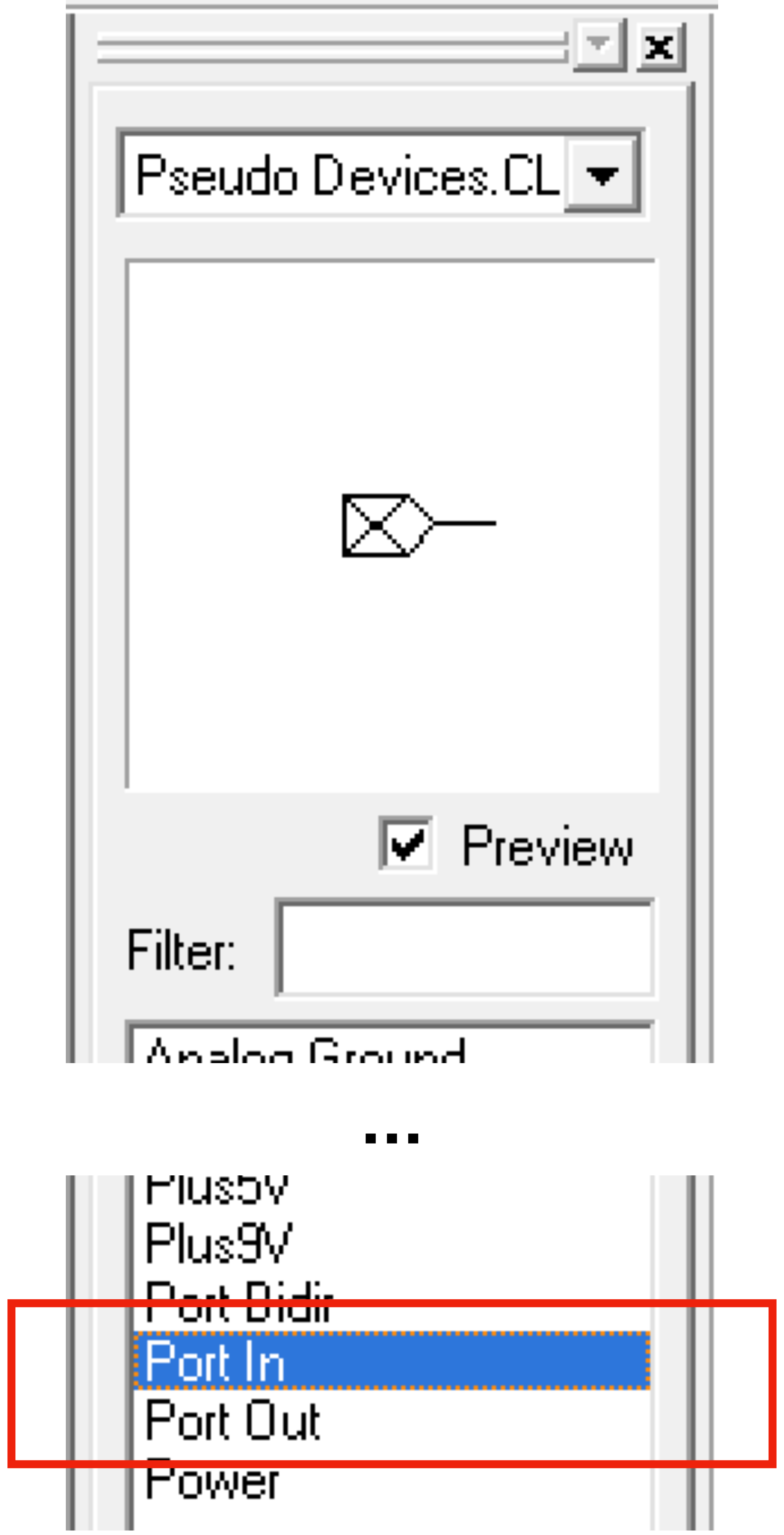
Binary Probe

1. Select the `Pseudo Devices` library

P3
LogicWorks

Sub-circuit



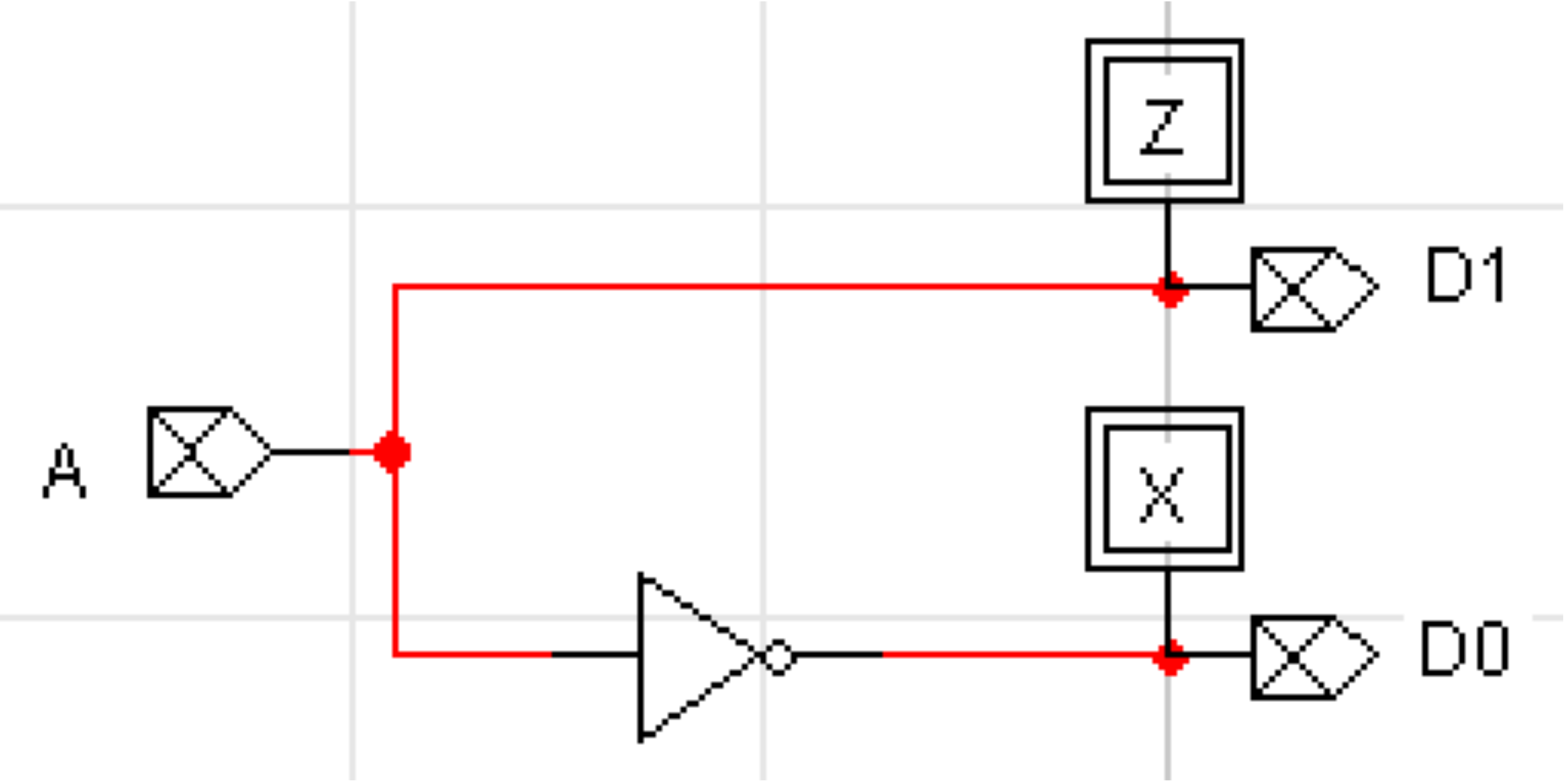


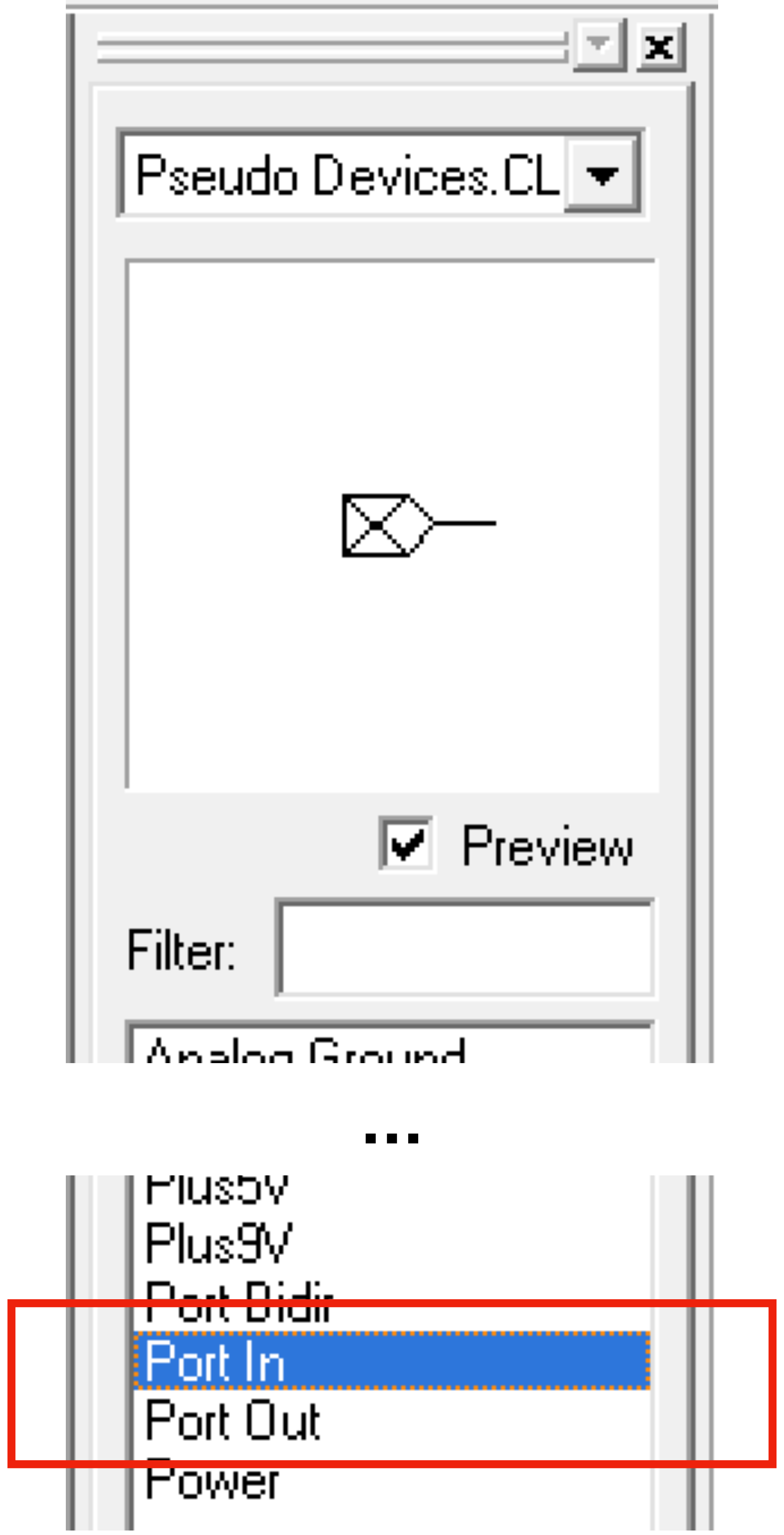
2. Replace the switch with `Port In`, replace the probs with `Port Out`, give them names

Technical

P3
LogicWorks

Sub-circuit



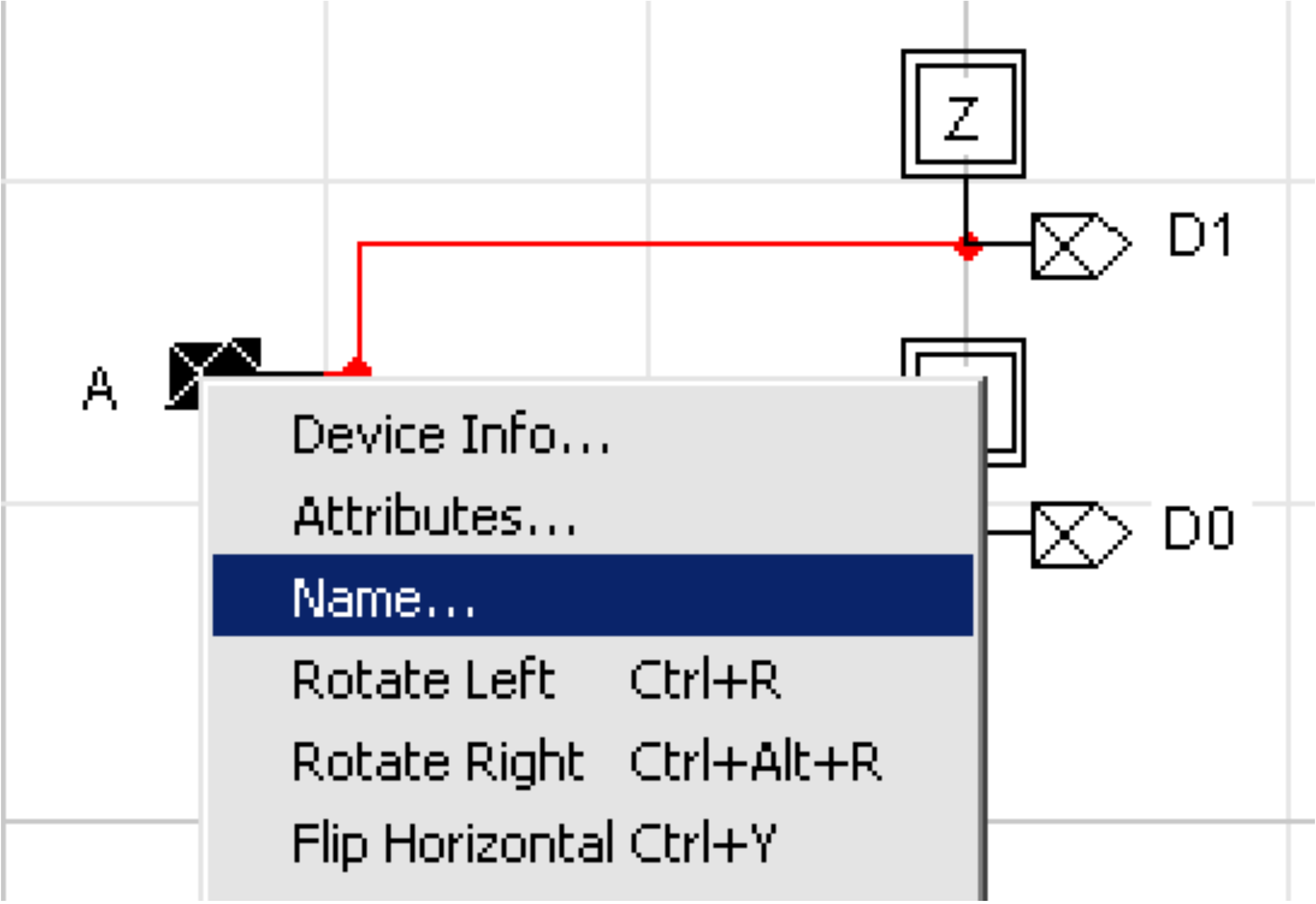


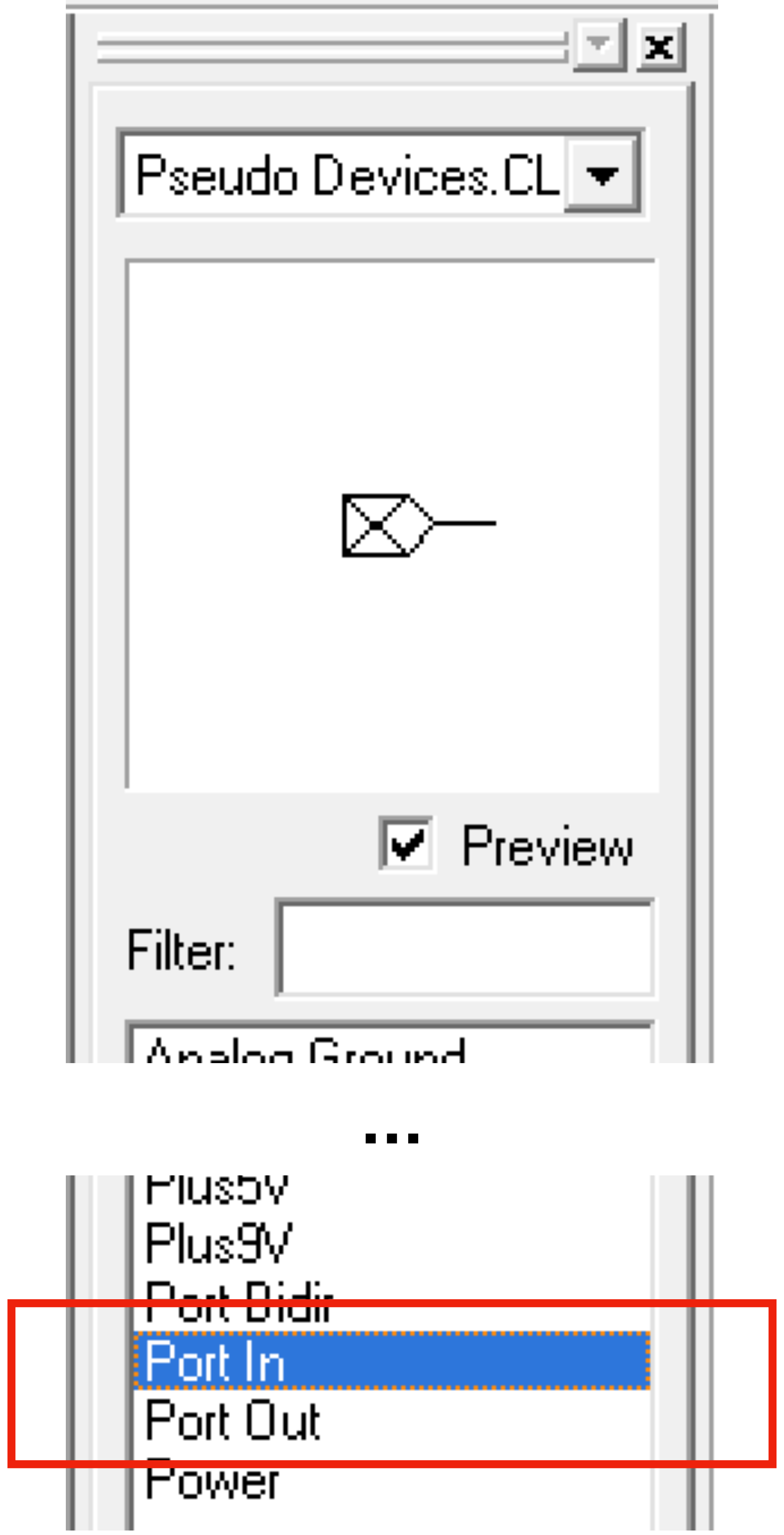
Technical

2. Replace the switch with `Port In`, replace the probs with `Port Out`, give them names

P3
LogicWorks

Sub-circuit



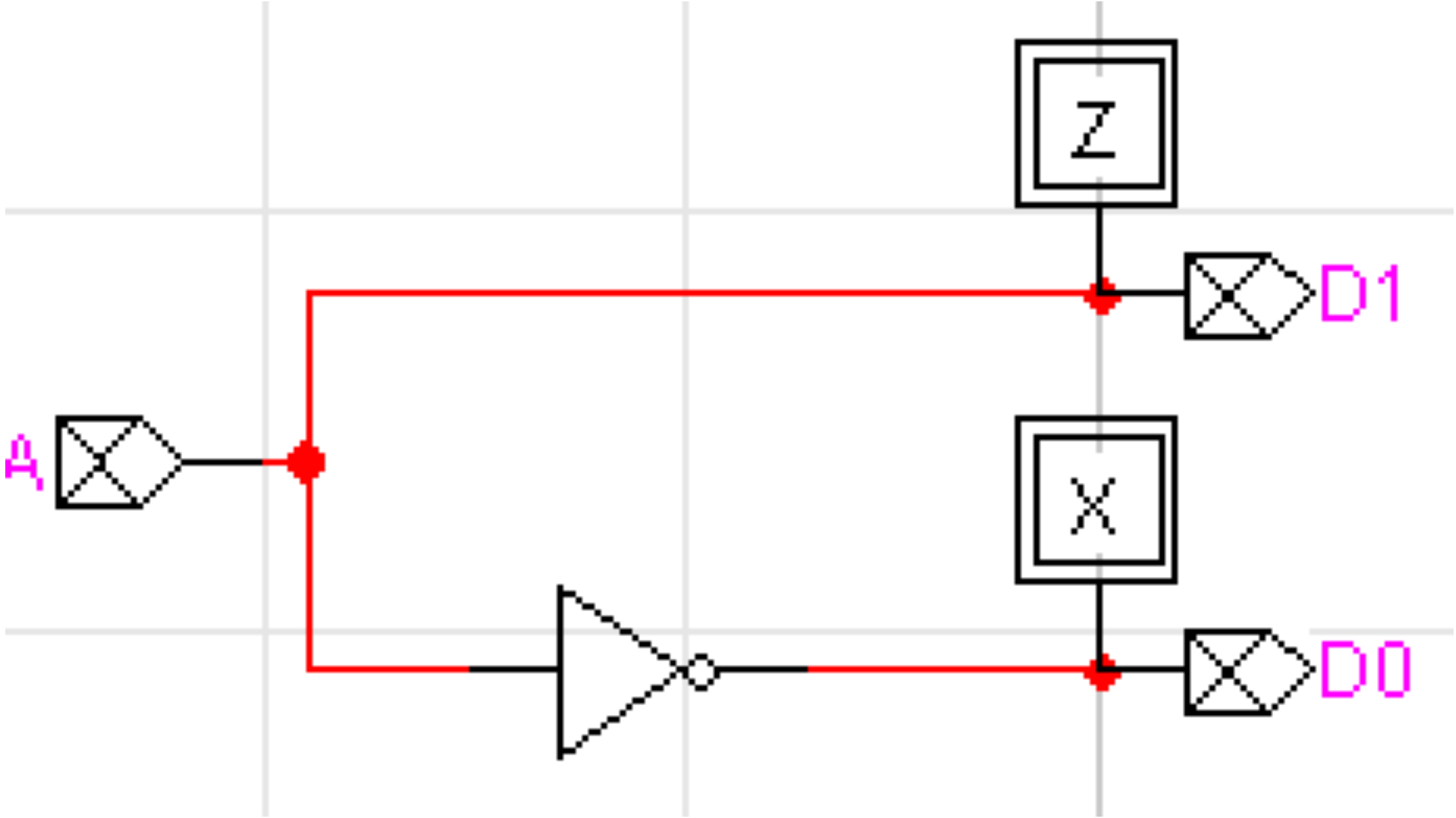


Technical

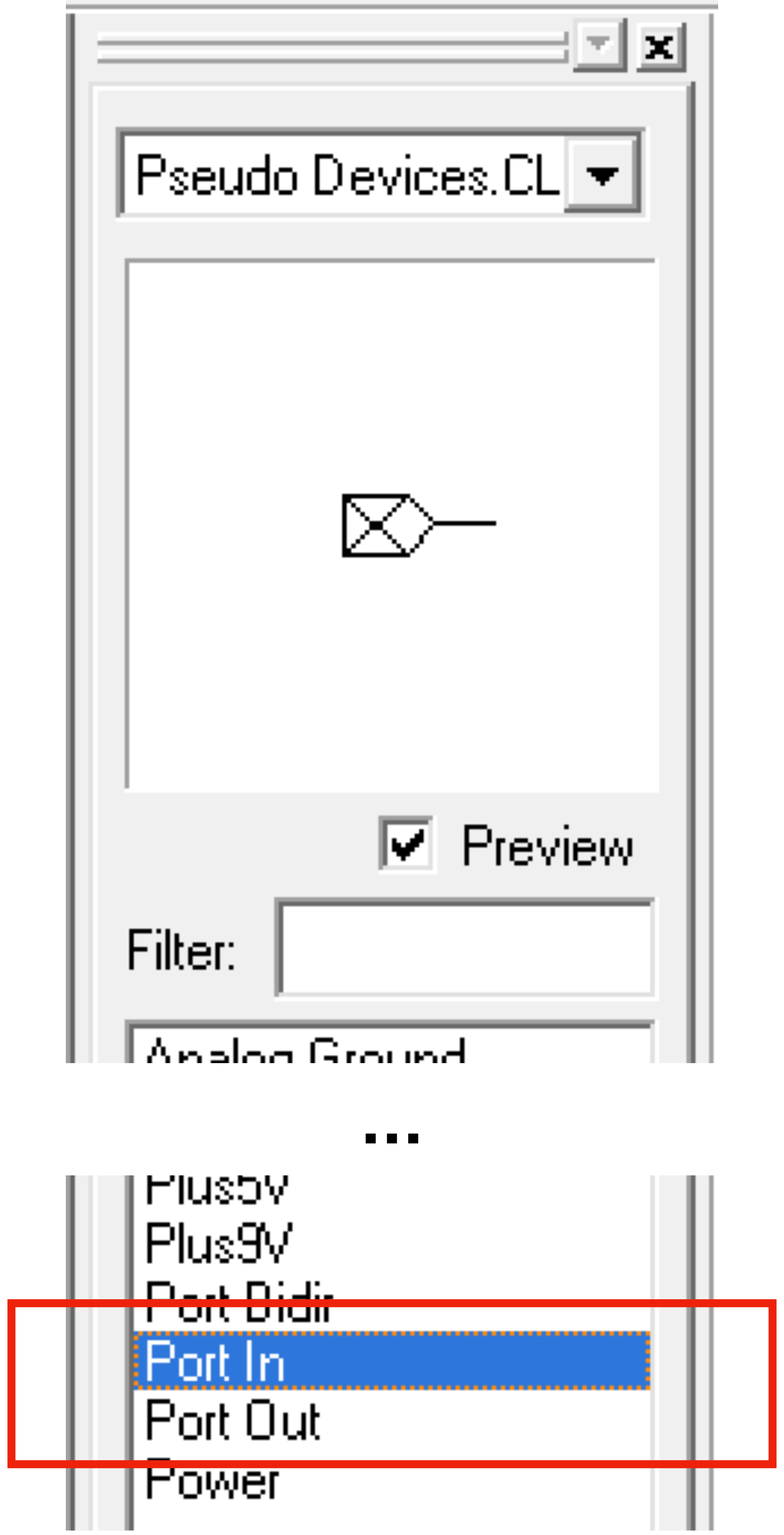
2. Right click on each port to give them names. Textbox will not be good enough

P3
LogicWorks

Sub-circuit



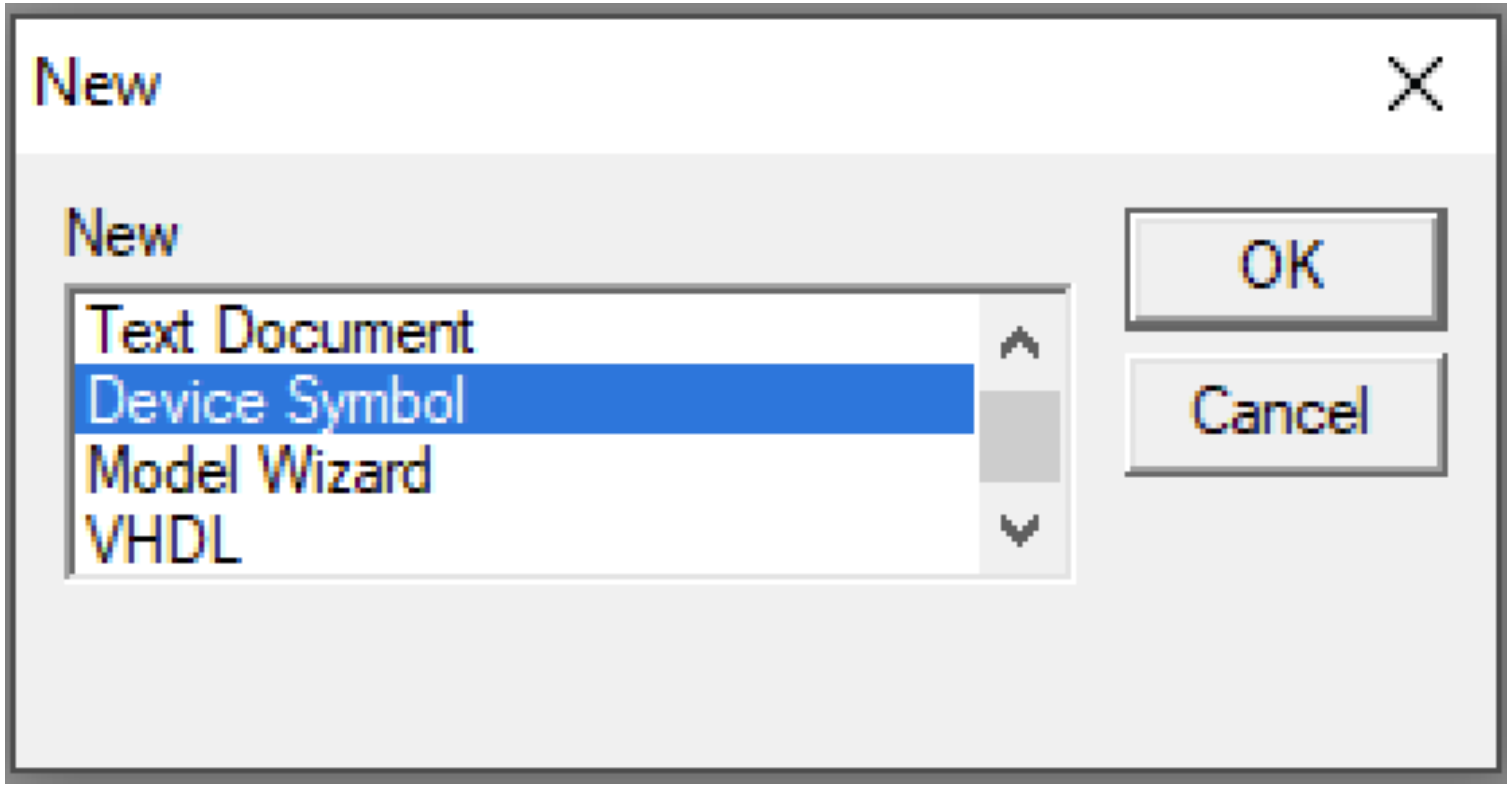
Once correctly done, the names will automatically be in purple



Technical

2. Right click on each port to give them names. Textbox will not be good enough

Sub-circuit



New

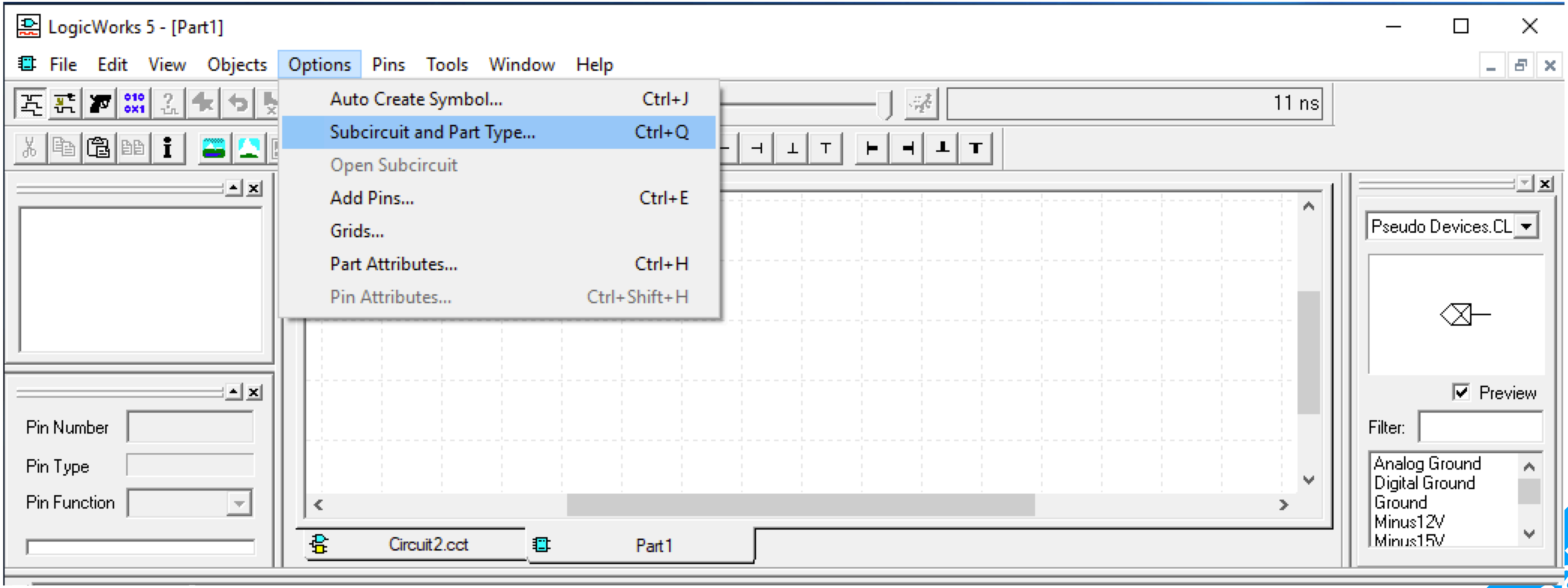
New

Text Document
Device Symbol
Model Wizard
VHDL

3. `File-New-Device Symbol`

P3
LogicWorks

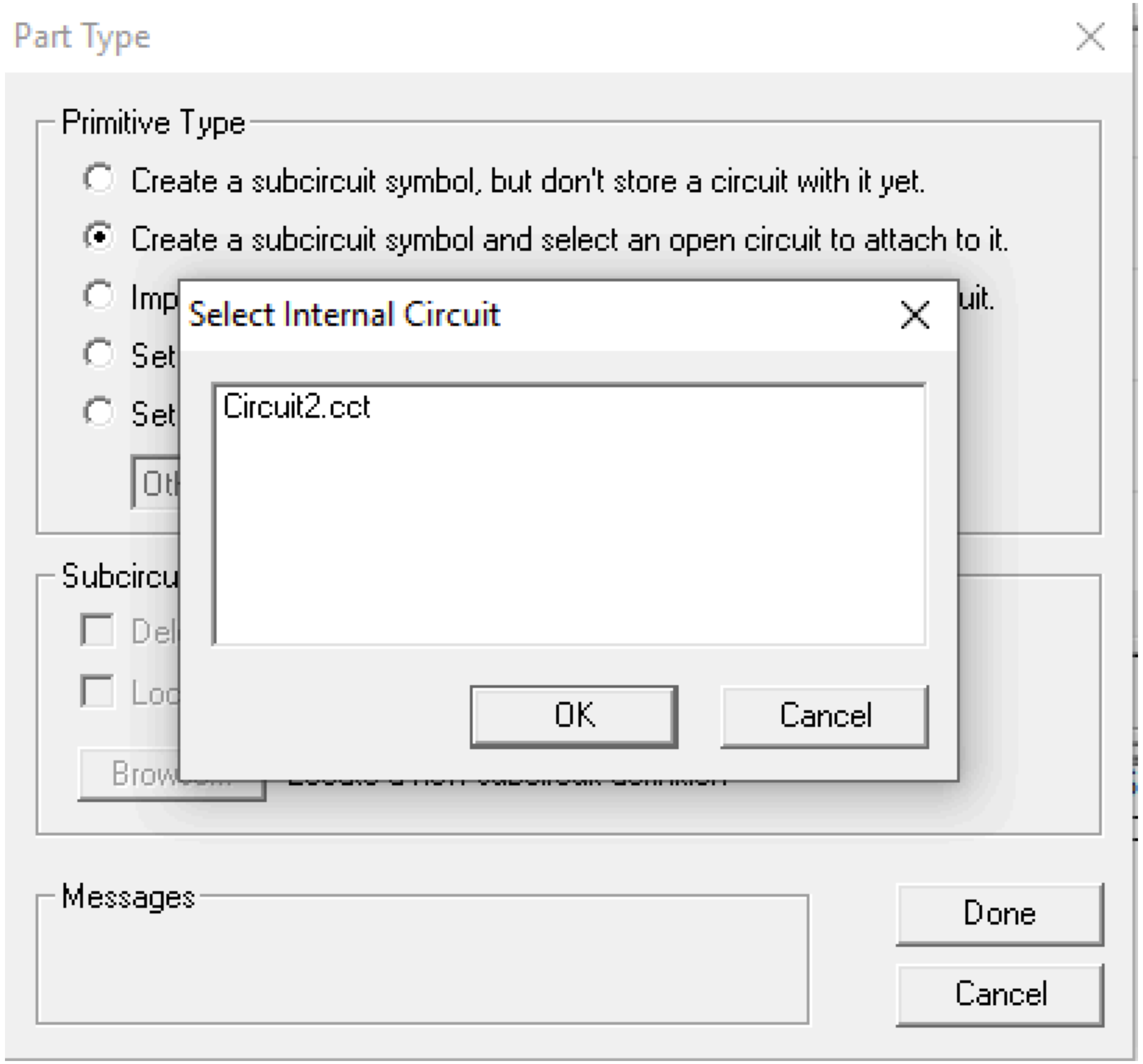
Sub-circuit



4. With the `Circuit.cct` open and `Part1` open, go to `Options — Subcircuit and Part Type`

Technical

Sub-circuit



Part Type

Primitive Type

○ Create a subcircuit symbol, but don't store a circuit with it yet.

● Create a subcircuit symbol and select an open circuit to attach to it.

○ Imp … uit.

○ Set

○ Set

Ot

Subcircu

□ Del

□ Loc

Brow

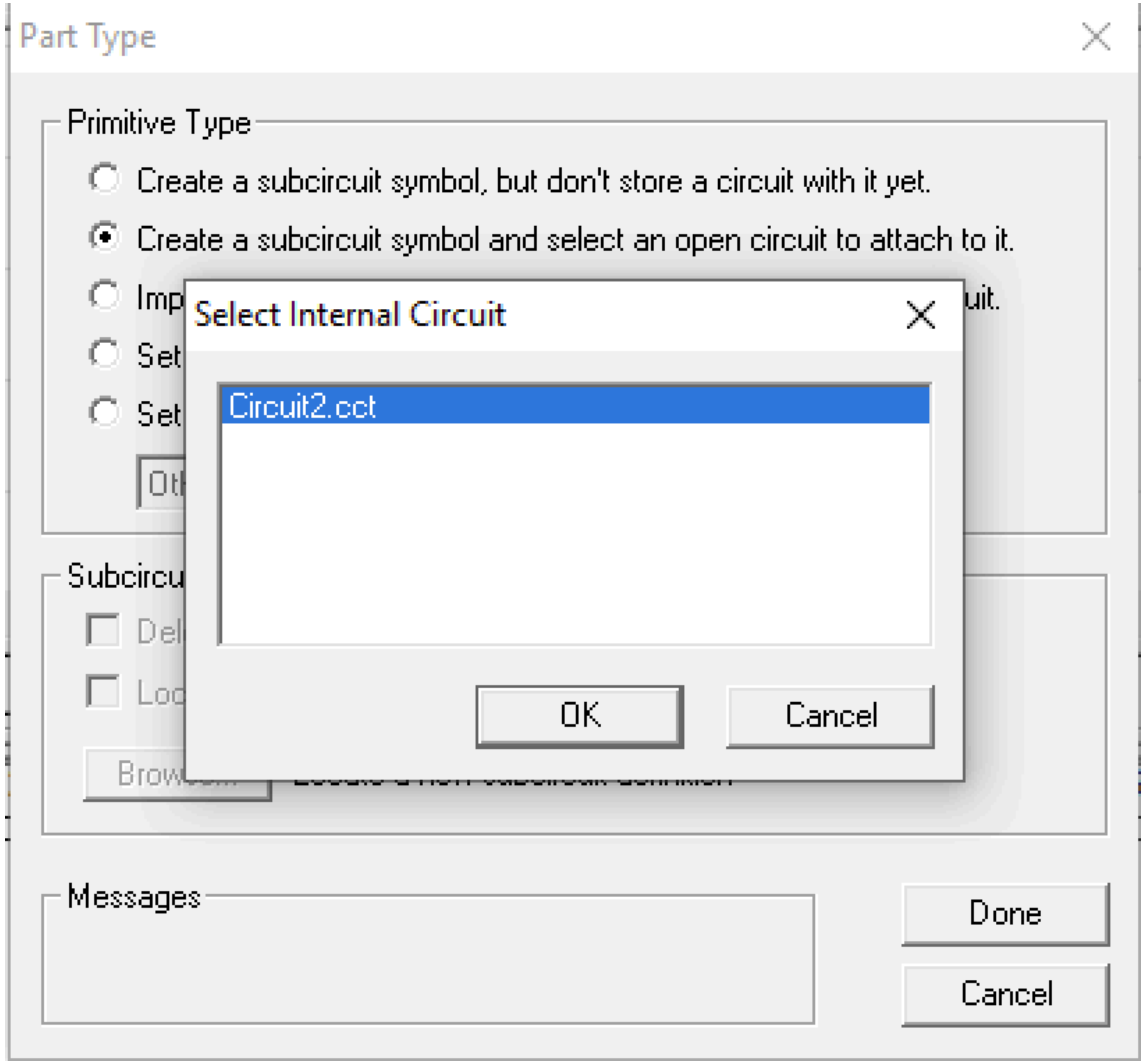
Select Internal Circuit

Circuit2.cct

5. **Select** `Create a subcircuit symbol and select an open circuit to attach to it`

Technical

Sub-circuit



Part Type

Primitive Type

○ Create a subcircuit symbol, but don't store a circuit with it yet.

● Create a subcircuit symbol and select an open circuit to attach to it.

○ Imp uit.

○ Set

○ Set

Oth

Subcircu

□ Del

□ Loc

Select Internal Circuit

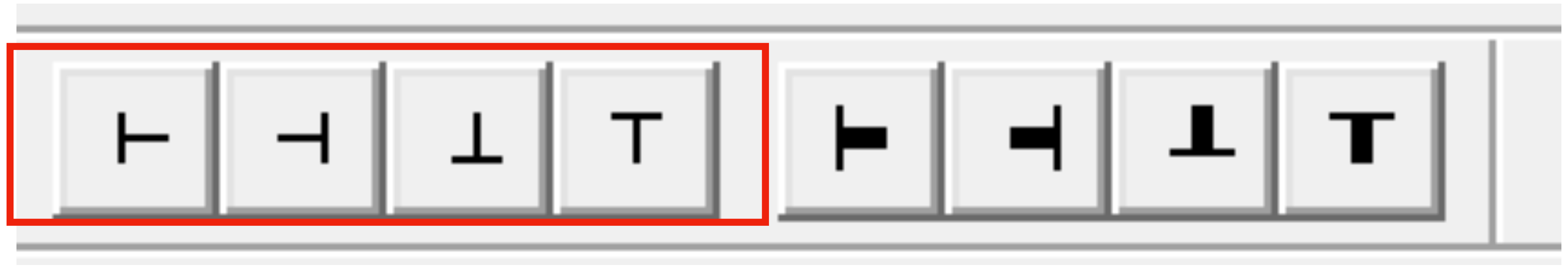
Circuit2.cct

Messages

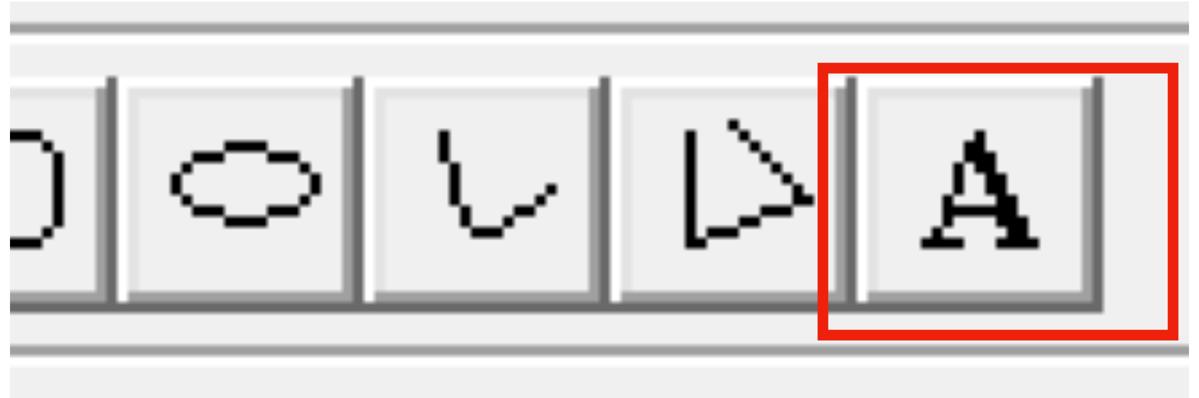
6. **Select** `Circuit.cct`

Technical

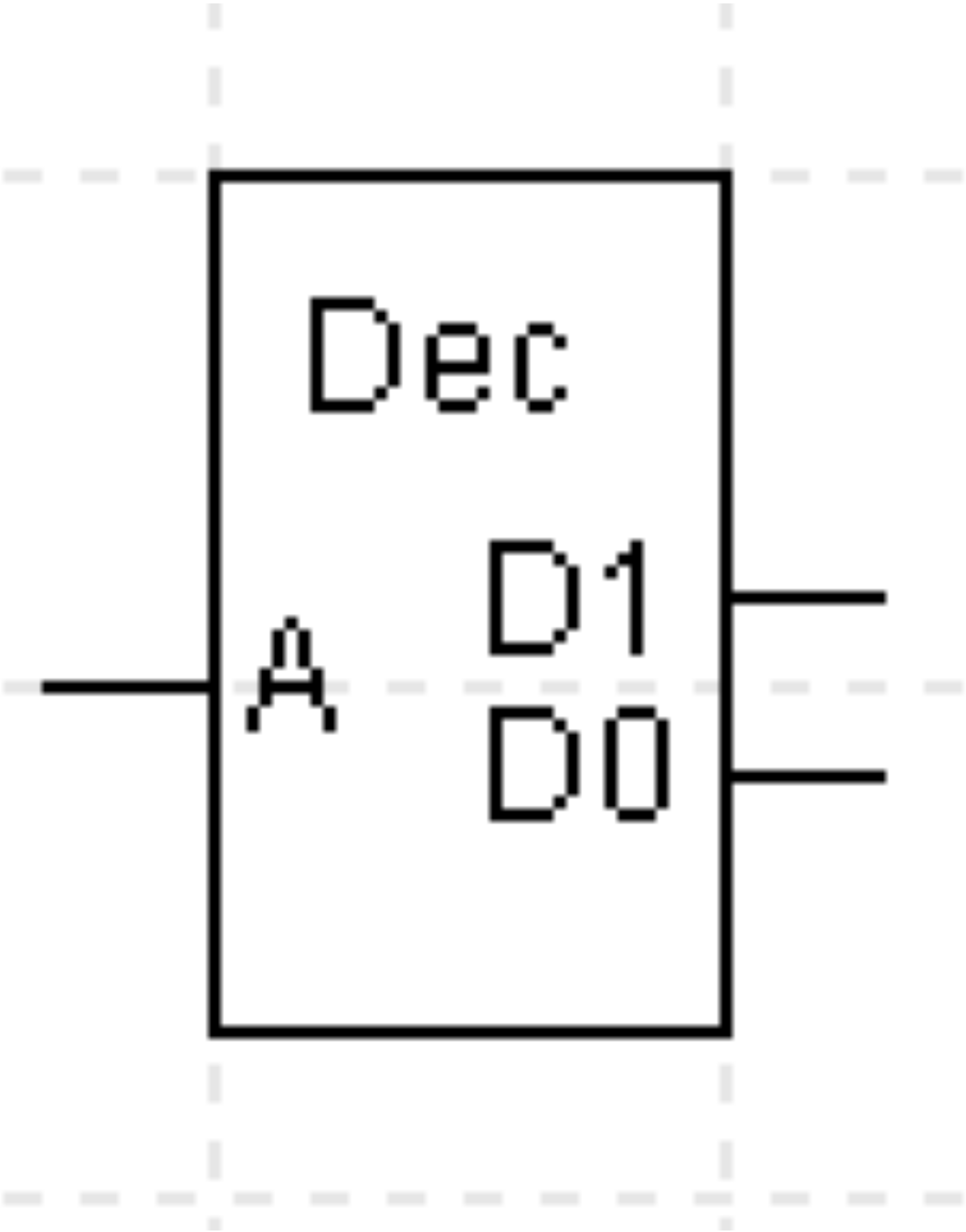
Sub-circuit



Use these pins only!



Add a text description

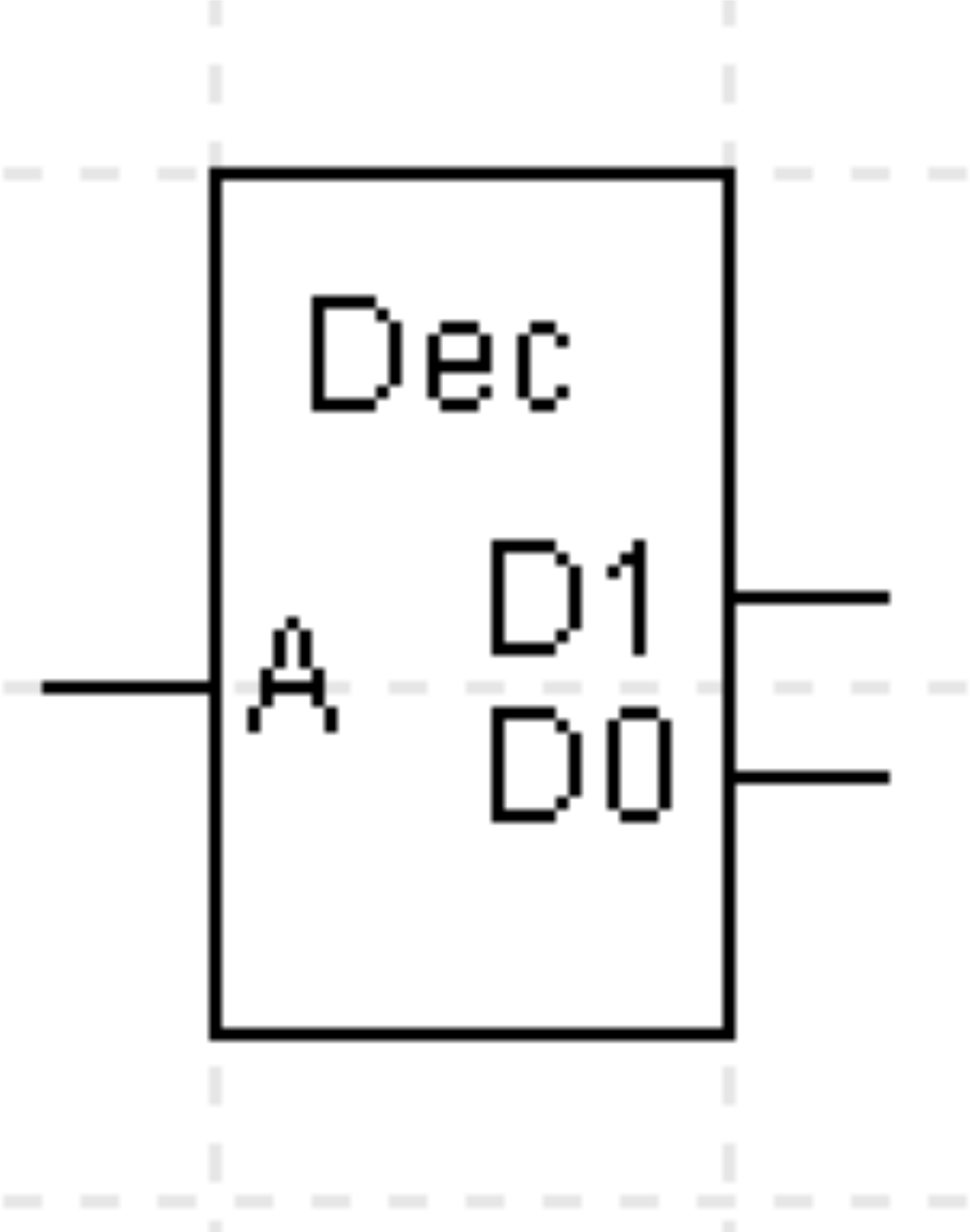


7. Draw a rectangle and add the THIN pins

Technical

Sub-circuit

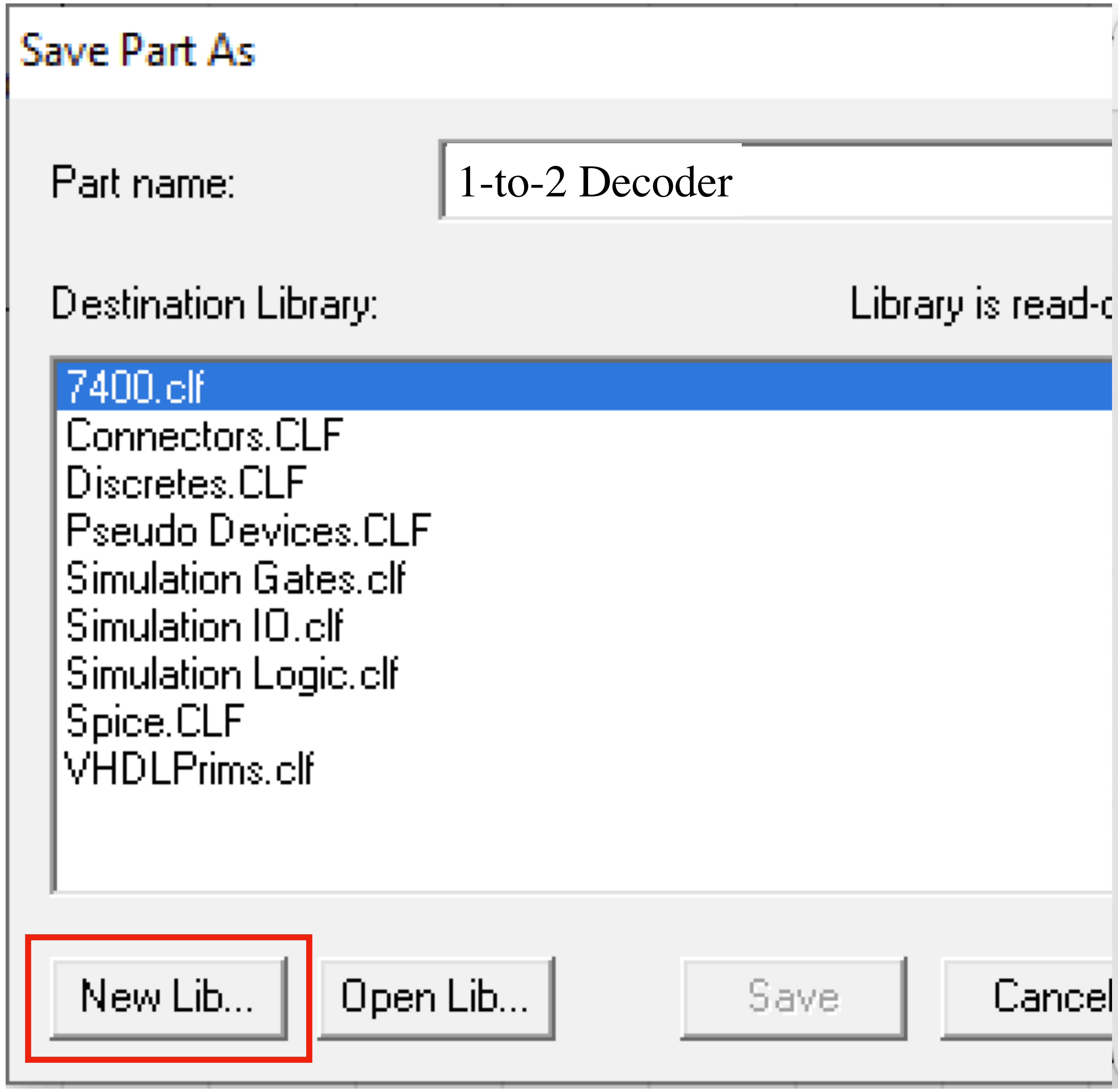
1. Ensure inputs are on the left, outputs on the right
2. Ensure that there are no excessive pins (pins that do not exist in the original circuit diagram)
3. Ensure the more significant digit is on top (D1 above D0)
4. Ensure the pins on the same side are equally spaced with distance of 2 (e.g. between D1 and D0)



7. Draw a rectangle and add the THIN pins

P3
LogicWorks

Sub-circuit

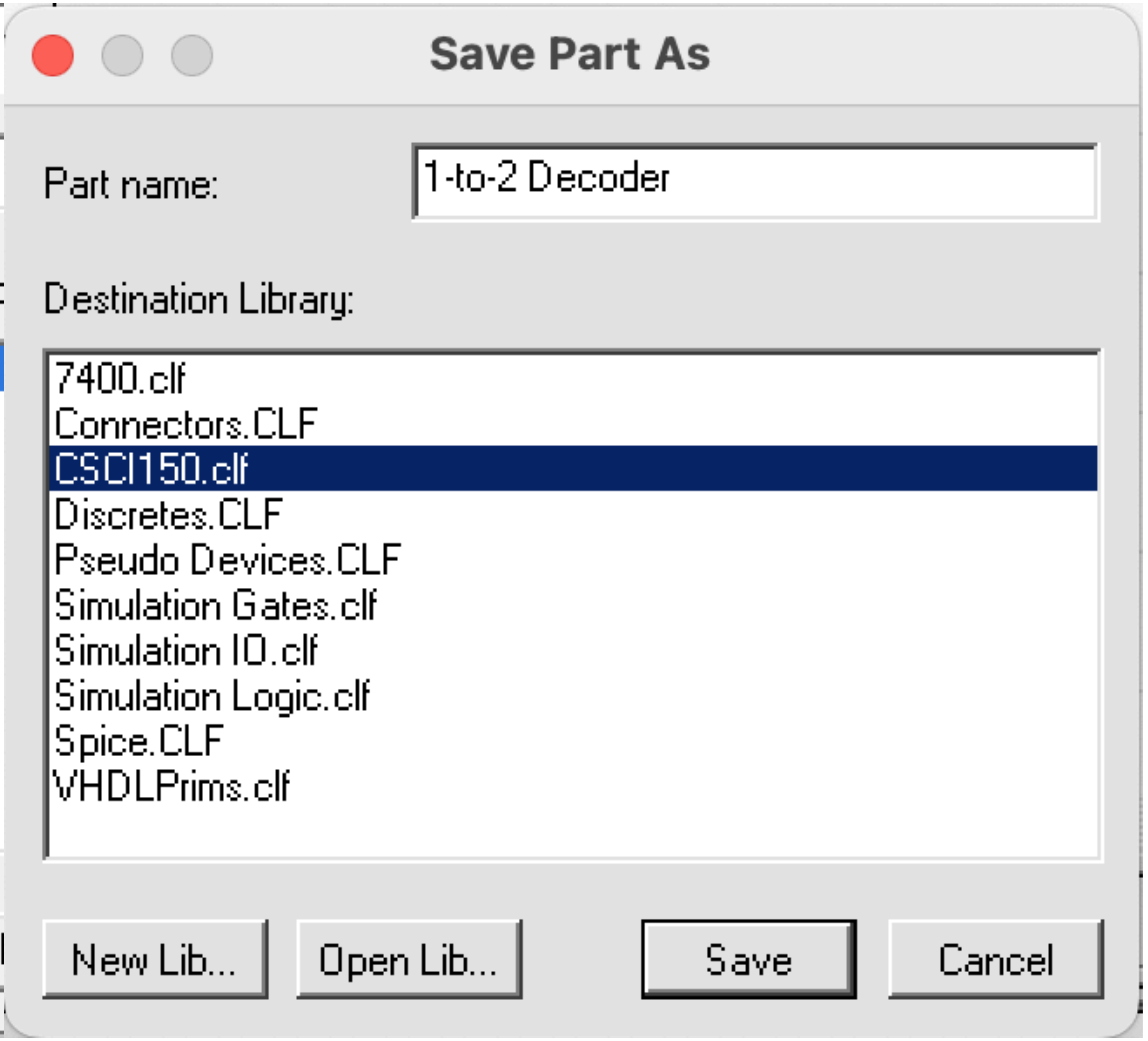


Save Part As

Part name: 1-to-2 Decoder

Destination Library: Library is read-o

7400.clf
Connectors.CLF
Discretes.CLF
Pseudo Devices.CLF
Simulation Gates.clf
Simulation IO.clf
Simulation Logic.clf
Spice.CLF
VHDLPrims.clf



Save Part As

Part name: 1-to-2 Decoder

Destination Library:

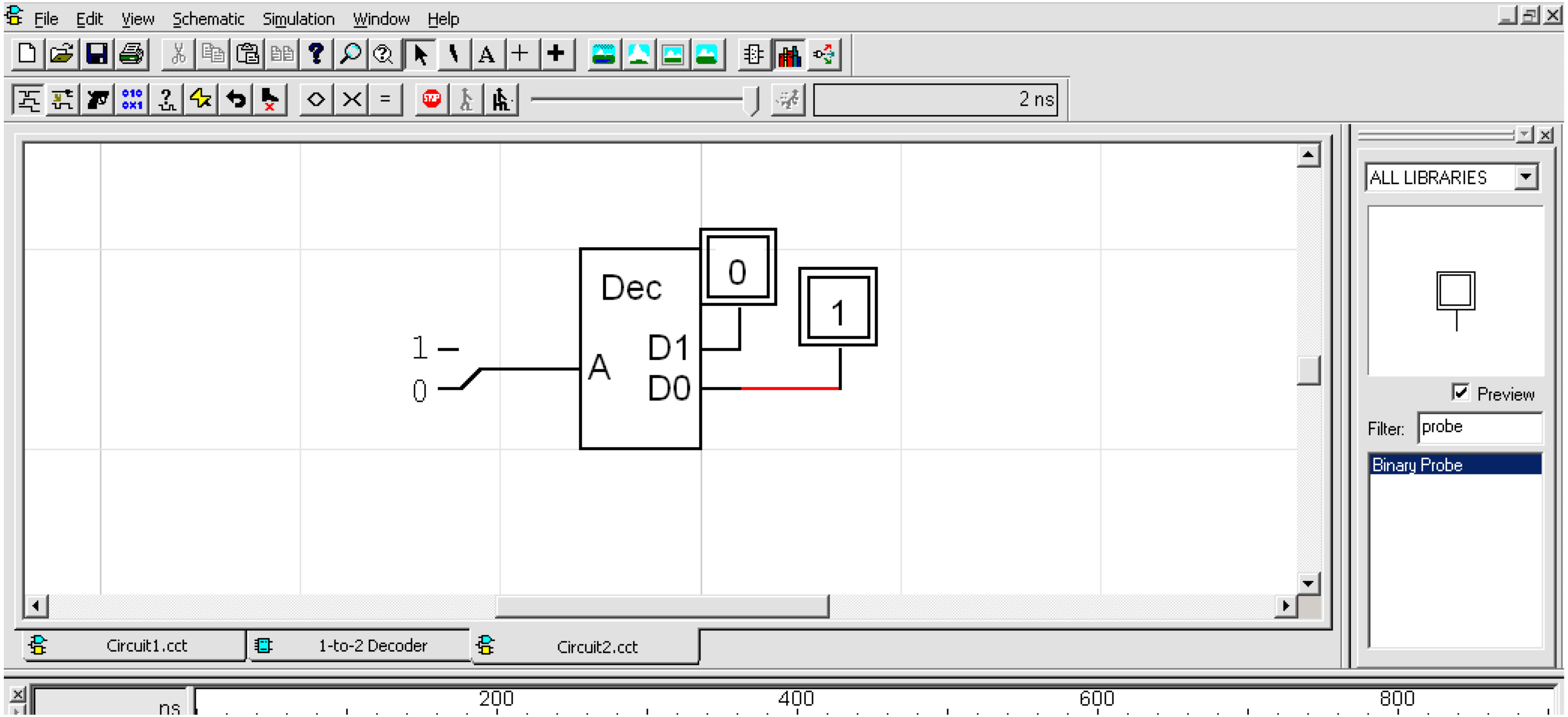
7400.clf
Connectors.CLF
CSCI150.clf
Discretes.CLF
Pseudo Devices.CLF
Simulation Gates.clf
Simulation IO.clf
Simulation Logic.clf
Spice.CLF
VHDLPrims.clf

8. Save the part in a new library as 1-to-2 Decoder, call your library CSCI150

Technical

P3
LogicWorks

Sub-circuit



9. Create a new circuit, test your new design before calling it a day

Technical

P3
LogicWorks

Circuit Drawing Practice

1. Sub-circuit
2. Implementing 2-to-4 Decoder using drawing tools
3. Implementing 3-to-8 Decoder using 2-to-4 Decoders
4. Implementing 4-to-16 Decoder using 2-to-4 Decoders or 3-to-8 Decoders
5. Implementing Octal-to-3 Priority Encoder
6. Implementing 1bit 4-to-1 Multiplexer using 2-to-4 Decoder
7. Implementing 4bit 4-to-1 Multiplexer using 1bit 4-to-1 Multiplexers [1]

[1] You will be reusing these designs in later lectures and assignments

Concept