4-bit Microprocessor Design and Implementation

CND 221:ADVANCED FULL CUSTOM VLSI DESIGN

TEAM MEMBERS

Student Name	ID
Muhammed Khaled Omar Orabi	V23010695
Ahmed mohamed mohamed shawky Elattar	V23010568
Mohammed Salah Aboshosha Mohammed	V23010141
Omar Dyaa Mohammed Mohamed	V23010501
Amr mohamed ali hassan	V23010177

PROJECT DESCRIPTION:

- This project Aims to develop and implement 4 bits microprocessor. In this project, students will delve into the intricate aspects of microprocessor layout and circuit design. The project will emphasize hands-on
  experience in creating circuit designs, layout designs, and verifications for critical microprocessor components, such as the ALU, memory, control unit, and other circuitry using custom and standard cell
  libraries.
  This project involves the schematic and layout design of a 4-bit microprocessor data path, including an
  ALU, a barrel shifter, and a register file using CMOS circuitry and Siemens VLSI design tools. The project will incorporate all the skills and knowledge students have gained from the individual lab assignments and provide a team of students with an opportunity to work on a more open-ended digital
  integrated circuits design problem

PROJECT DESCRIPTION:

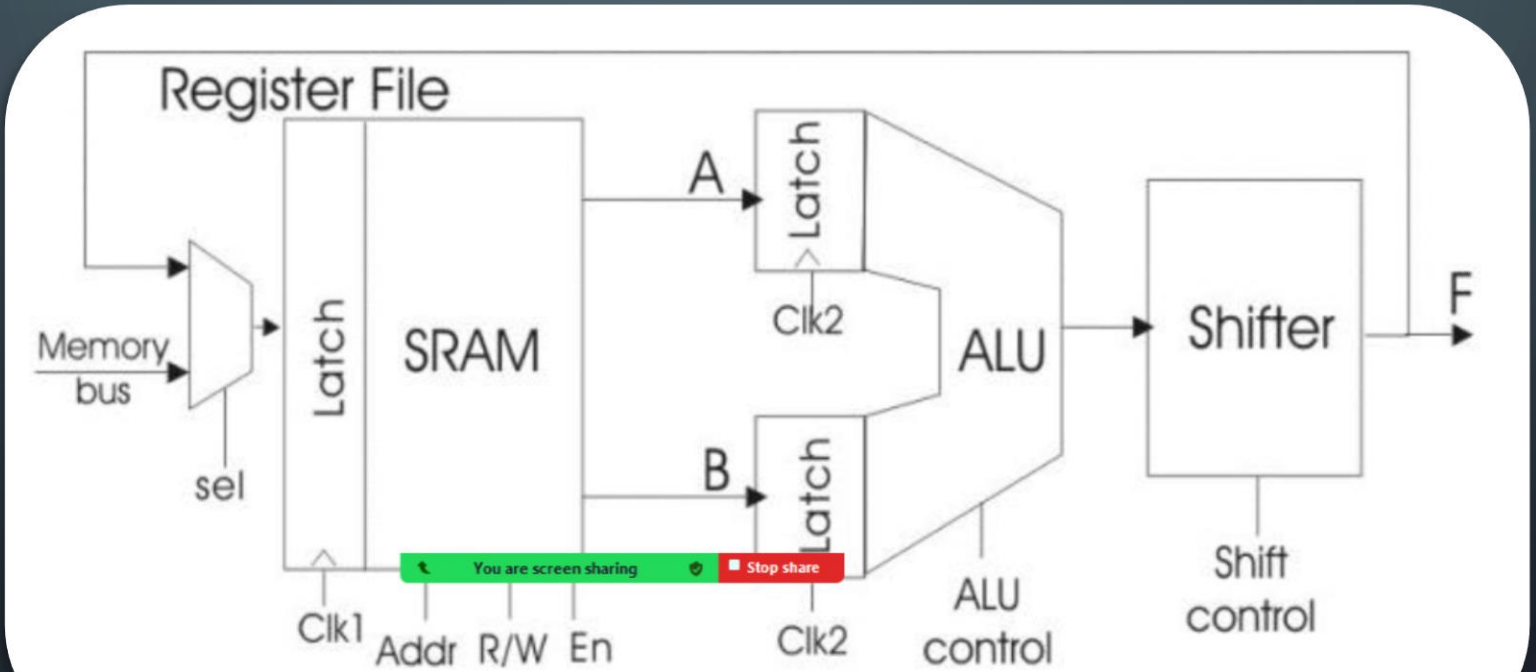


Figure 1. Structure of the microprocessor data path to be designed in this project.

DATAPATH:

- The data path is the core element of a microprocessor that can perform many functions, including
  arithmetic and logic evaluation, data movement, and storage of results in a memory address specified by
  the instruction. In this design project, teams must design a complete 4-bit data path consisting of a register
  file, an ALU (Arithmetic Logic Unit) and a shifter, as shown in Figure 1. The data path will be capable of
  multiple instructions determined by several control signals.

DATAPATH:

- The ALU can perform both arithmetic and bit-wise logical operations. The ALU will have two 4-bit data inputs and several control signals that specify the ALU function to be executed in a given clock cycle.
  The ALU input data is loaded into the latch at the rising edge of the ALU clock (clk2), and the output is generated after a delay due propagation through the combinational logic of the ALU and the shifter.
  The
  central component of an ALU is an adder, and for this project a Carry select adder will be used.
  A shifter provides data manipulation capabilities. A barrel shifter will be used because it has a very efficient layout and can perform n-bit shifts in a single clock cycle.
  The register file is a block of memory that provides the data inputs to and stores outputs from the ALU.
  The register file at read/write address is specified by control bits included in the opcode. In this project, the
  register file and ALU/shifter will be synchronized by two nonoverlapping clock phases that determine when data is latched into the register file (clk1) and into the ALU (clk2).

DATAPATH DESIGN SPECIFICATIONS:

- The following specifications must be achieved by the data path you design:

  a. ***Bit width:*** 4

  b. ***Inputs:*** one 4-bit word (from the memory bus), 13 control/address bits (6 for the ALU/shifter, 7 for the register file), two non-overlapping clocks (clk1, clk2).

  c. ***Outputs:*** 4-bit word (F), 1 carry-out

  d. ***Functions***: must implement ***at least*** the following functions:

ALU FUNCTIONS:

- *A NOR B*
  - *A XOR B*
  - *A NAND B*
  - *NOT A (invert)*
  - *Increment A*
  - *Decrement A*
  - *A + B*
  - *A * B (Bonus Function)*

If you implement only these functions, you must do so with only three ALU control signals. There
is no explicit Cin bit; Cin must be generated from the ALU control bits

SHIFTER

- The shifter must implement:
  - Pass (no shift)
  - Shift left 1.
  - Shift left 2.
  - Shift right 1.
  - Shift right 2.
  - Rotate right 1.
  - Rotate left 1.

EXPANDED FUNCTIONS

- Two function expansion bits can be provided to implement additional ALU and/or shifter functions.

- ***Register file***

  The 4x4 SRAM register file must implement 2-port read_from_memory and 1-port write_to_memory functions. Each clock cycle will be either a read or a write cycle, so two cycles will be needed to complete a function.

  e. ***Power supply***: VDD = 3 volts, referenced to 0-volt ground.

  f. ***Size specifications***: no specific layout size is required, but optimization of physical size is a design priority, and higher scores will be awarded to groups with smaller layouts. The floorplan in Fig. 2 will assist in achieving an optimal layout.

  g. ***Timing specifications***: signal timing requirements are defined below. The worst-case propagation delay for all ALU + shifter functions must be optimized (e.g., less than **50ns**)

CONTROL BIT SPECIFICATIONS:

- **ntrol Bit Specifications:**
  It is recommended to complete this project with only the following 13 control signals: 1-bit register file
  input mux selection, two 2-bit register addresses, 1-bit register file read/write, 1-bit register file enable, 3-
  ALU controls, and 3-bits shift controls. These control signals must be given the signal names defined
  below. Note: this only defines the externally-supplied control signals; you can decode or otherwise
  manipulate them to construct any necessary internal control signals.

DATA PATH LAYOUT

- The 4-bit datapath can be implemented by constructing a 1-bit slice horizontally and stacking 4 one-bit
  slices vertically. This layout floorplan for the data path is shown in Fig. 2. The data flows horizontally from
  the left to the right. The control signals run vertically across each slice. You may implement some
  functional blocks as 4 bits, but make sure you organize it in 1-bit slices that can be pitch-matched to the
  other functional blocks.

DATA PATH LAYOUT

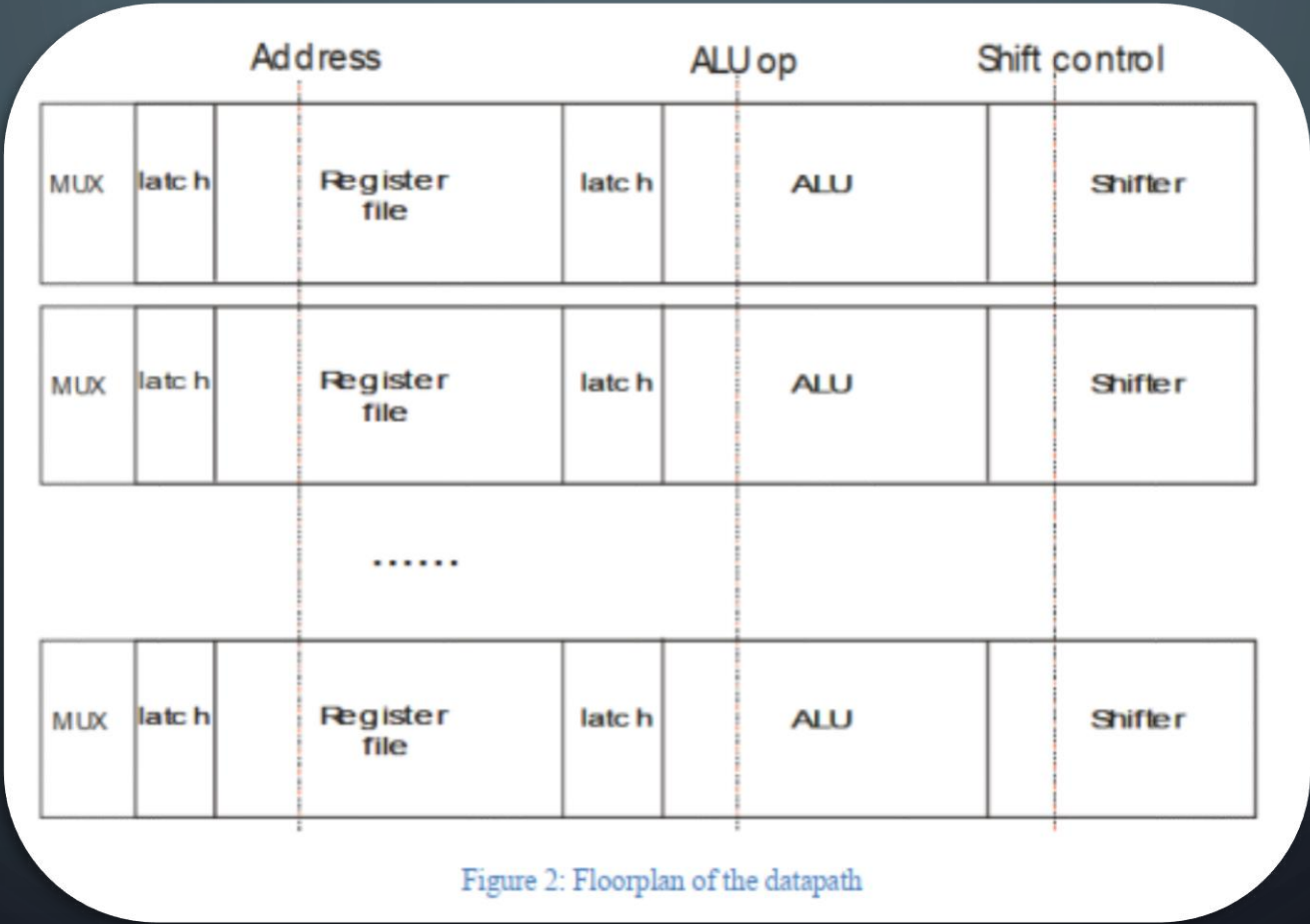


Figure 2: Floorplan of the datapath

REGISTER FILE DESCRIPTION:

- The 4-bit register file should be implemented as a 4x4 SRAM. Since the SRAM is small, you do not have to implement a sense amplifier. The register file should have two 4-bit read (output) ports and one 4-bit write (input) port. Input to the register file can come either from the ALU output for from an external memory bus, as selected by a MUX at the input of the register file. The memory bus should be implemented as a 4-bit input word. A 4-bit latch should be included to store the register file input word, which, during a write cycle, will be saved to the SRAM block.
  The register file uses an enable signal for the general task of enabling inputs to and outputs from the register file at the proper time. The register file must be carefully designed to properly implement the enable function. In the read cycle, register file output is passed to the ALU when enable is high; otherwise, that output should be at high impedance. The output should also be at high impedance during the entire write cycle, when the R/W signal is high. In the write cycle, data is stored in the SRAM only when enable is high; otherwise, the data from the register file latch should not be allowed to affect the SRAM. To accomplish this, the enable signal should act on the SRAM address decoder outputs. The enable function allows the data path to implement functions that do not act on the register file, such as branches or status register checks, although these functions are not implemented in this design project.
  The register file address contains two independent 2-bit addresses. Address “A” should specify the address for data passed to ALU input A, and address “B” should specify ALU input B. Address “A” is also used to specify the address for storing data to the SRAM during a write cycle. The timing of the data path should allow the value of address A to be changed before the write cycle, as shown in Fig. 3. However, for this project you can keep the address constant and store the ALU/shifter data back to the same location where input A was taken from.

REGISTER FILE DESCRIPTION:

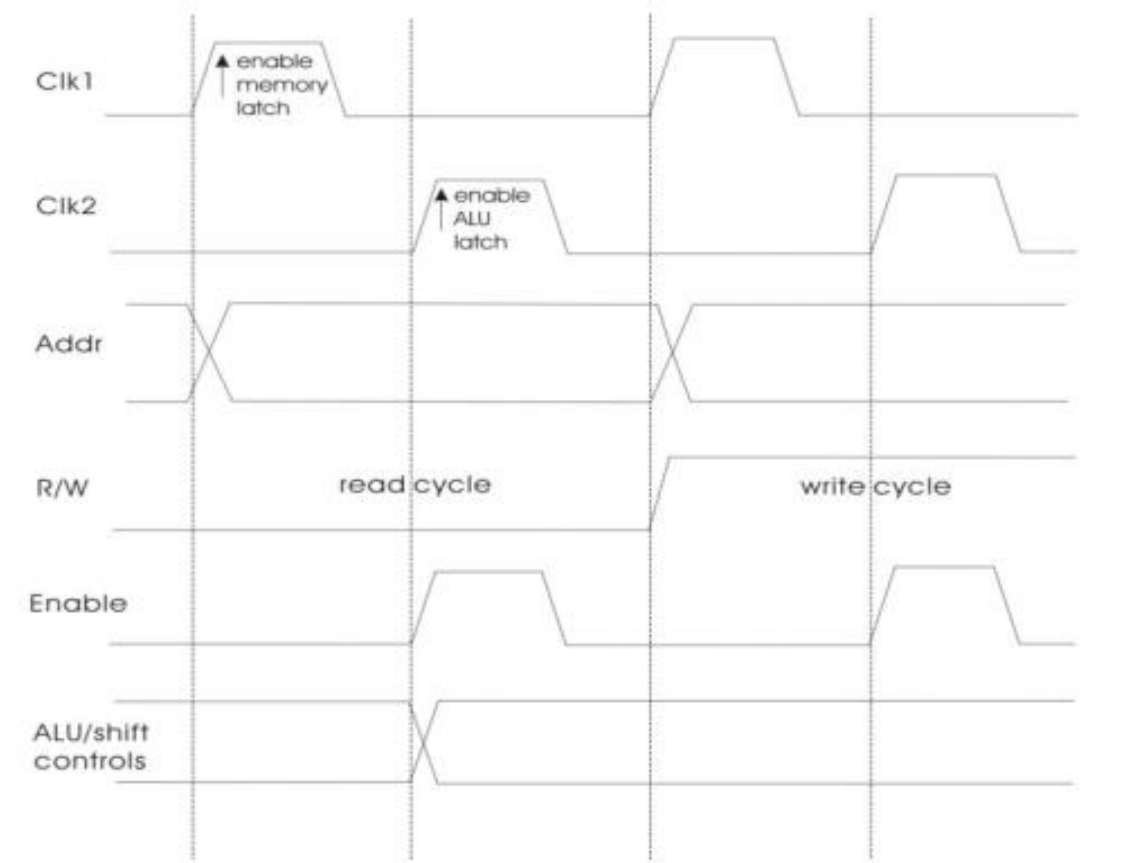


Figure 3: Timing diagram for a complete 2-cycle data path function. The first cycle is a register file.

CLOCK/TIMING DESCRIPTION:

- Two non-overlapping clock phases are required for the data path, as shown in Fig. 3. Data from the
  ALU/shifter or memory is loaded in the register file latch when clk1 is high and held there while clk1 is
  low. Data from the register file is loaded in the ALU latch when clk2 is high and held there when clk2 is
  low. Two cycles of the clocks are needed to complete a data path function; data is passed to and through
  the ALU/shifter in the “read” cycle, and ALU/shifter data is stored in the register file on the “write” cycle.
  Address and control signals for all data path functions must be synchronized to the clock signals
  specifically as shown in Fig. 3.

PROJECT REQUIREMENTS (DELIVERABLES):

- The following cell views for all circuit blocks must be created using the available tools:
  - Schematic, Symbol, and Layout
  - DRC, LVS, and Parasitic extraction.
  - Circuit simulation (functional simulation of all datapath functions)
  - Post layout simulation (functional simulation of all datapath functions after adding PEX file)

INPUT AND OUTPUT NAMING:

- **Inputs**
  - md<0:3>, memory bus data
  - rfs, reg. file mux selection
  - rfen, reg. file enable
  - rw, read/write
  - ada<0:1>, register address A
  - adb<0:1>, register address B
  - f<0:2>, shifter controls
  - f<3:5>, ALU controls
  - clk1, clk2
  - **outputs:**
  - Y<0:3>
  - Cout

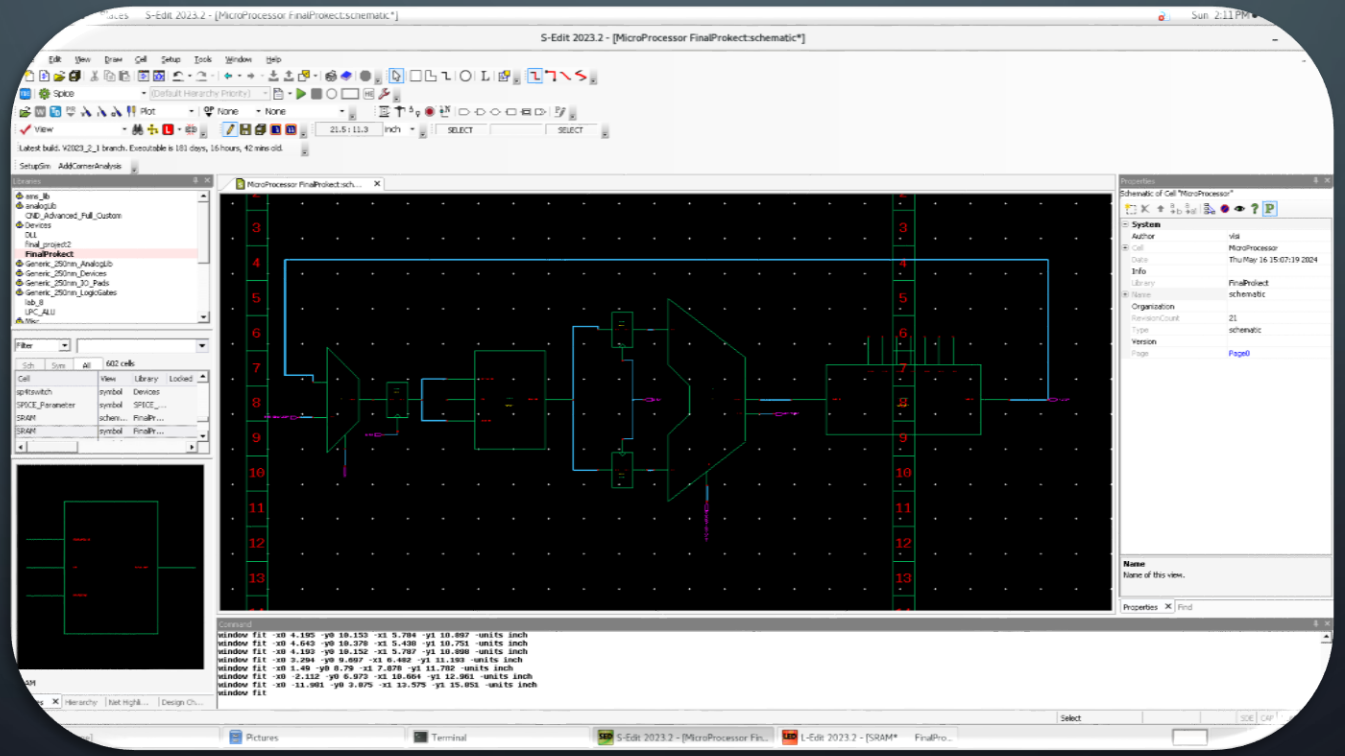
THE FOLLOWING CHARACTERISTICS MUST BE MEASURED AND REPORTED

- o Slowest logic function and propagation delay for that function (ALU)
- o Slowest arithmetic function and propagation delay from that function (ALU)
- o Slowest propagation delay of the data path (ALU + Shifter)
- Hint: Propagation delays are measured from clock edge to last output transition.
- o Physical area of the complete data path layout
- o Total number of transistors in the data path (available from the final LVS output)
- o Total, static, and dynamic power consumption for a full read and write cycle during an ADD and
- Shift_Left_2 operations.

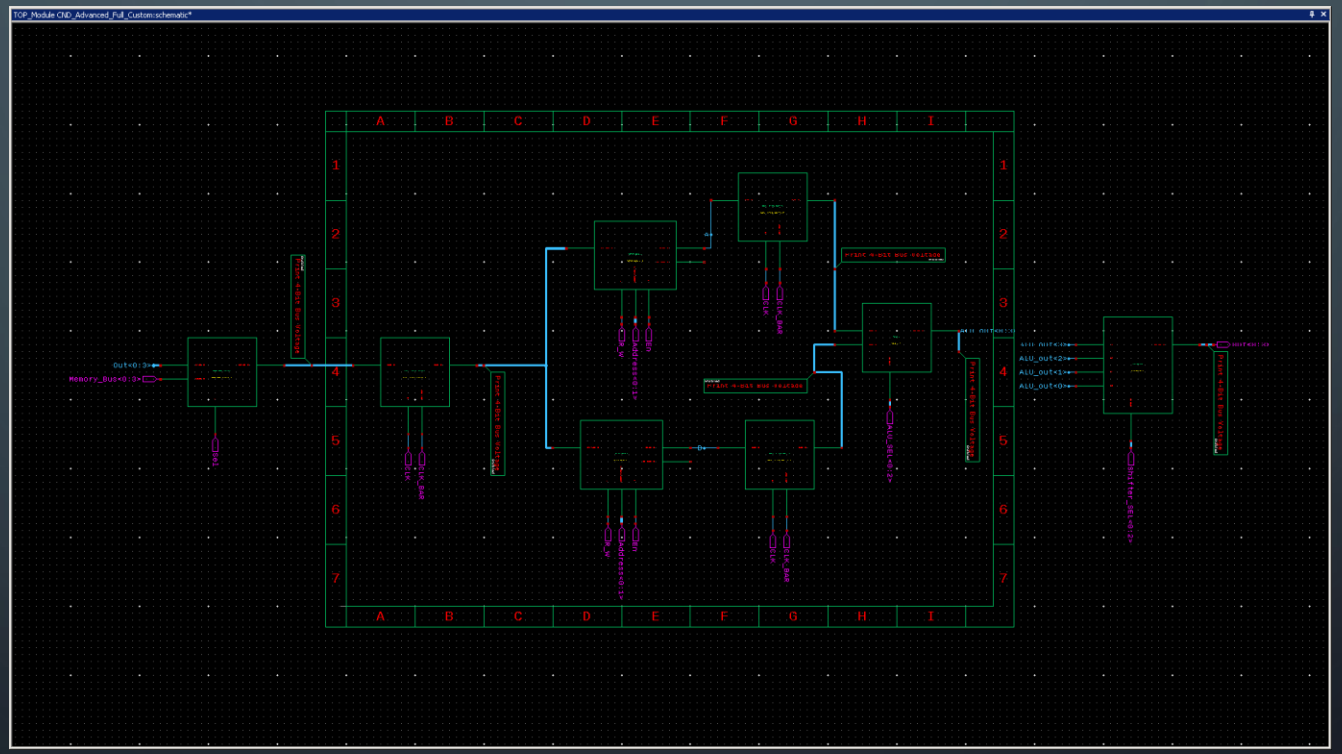
CIRCUIT REQUIREMENTS

- o A Carry select adder must be implemented.
  o A barrel shifter must be implemented.
  o The register file memory must be implemented with a multi-port 4x4 SRAM.
  o Minimum sized transistors can be used, but are not required.
  o All required functions must be implemented using only the specified control/select inputs.
  Page 6 of 6 CND 221 – Final Project
  o You must employ hierarchical design where you instantiate lower-level cells into higher level circuit block, both in schematic and layout design
  o Buffers should be designed and included wherever fan-out is larger than 5 gates.
  Hint:
  It is always best to pass DRC and LVS on smaller cells BEFORE using them to construct larger cells. Final LVS can be very time consuming for many groups

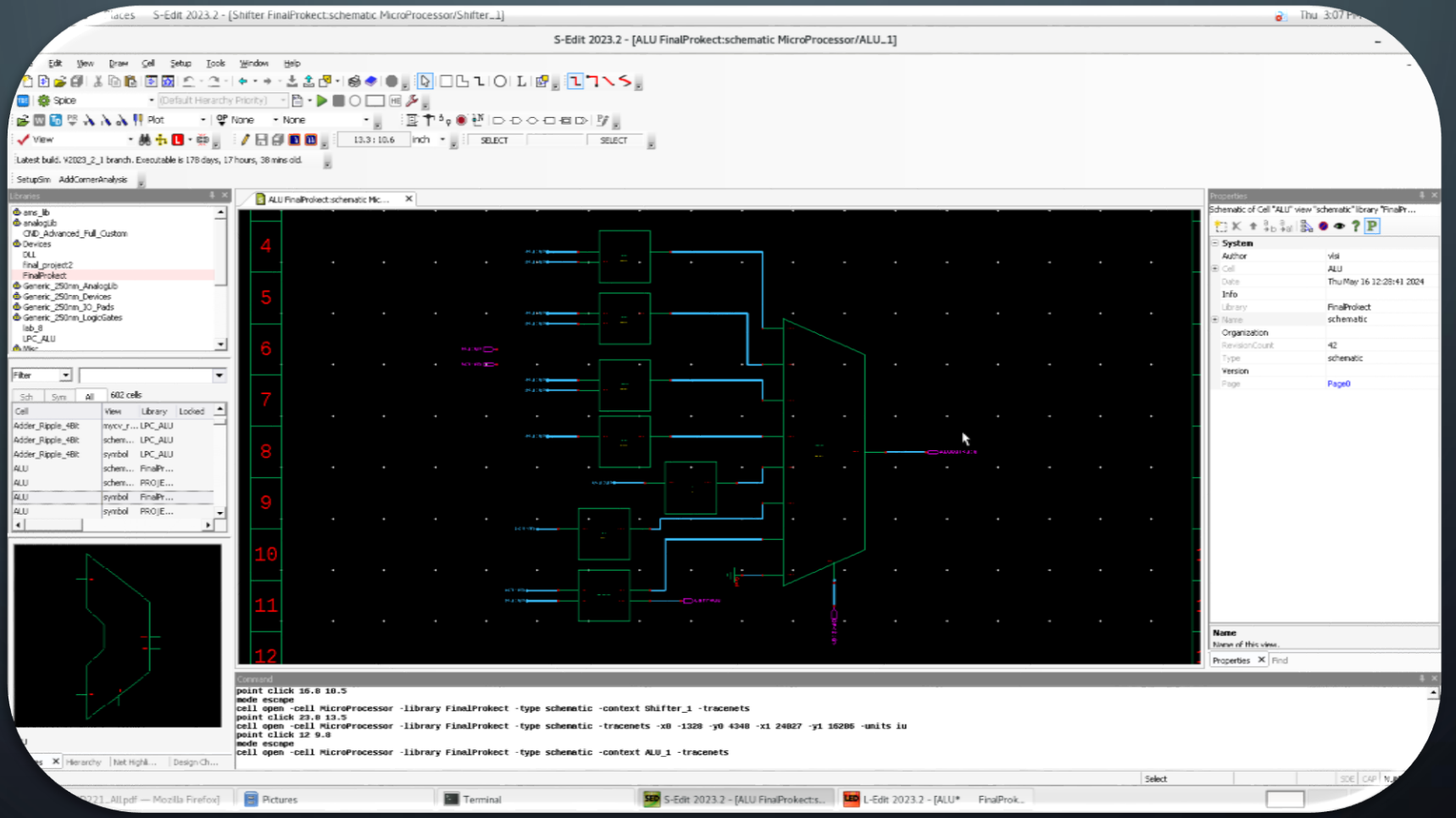
FINAL PROJECT



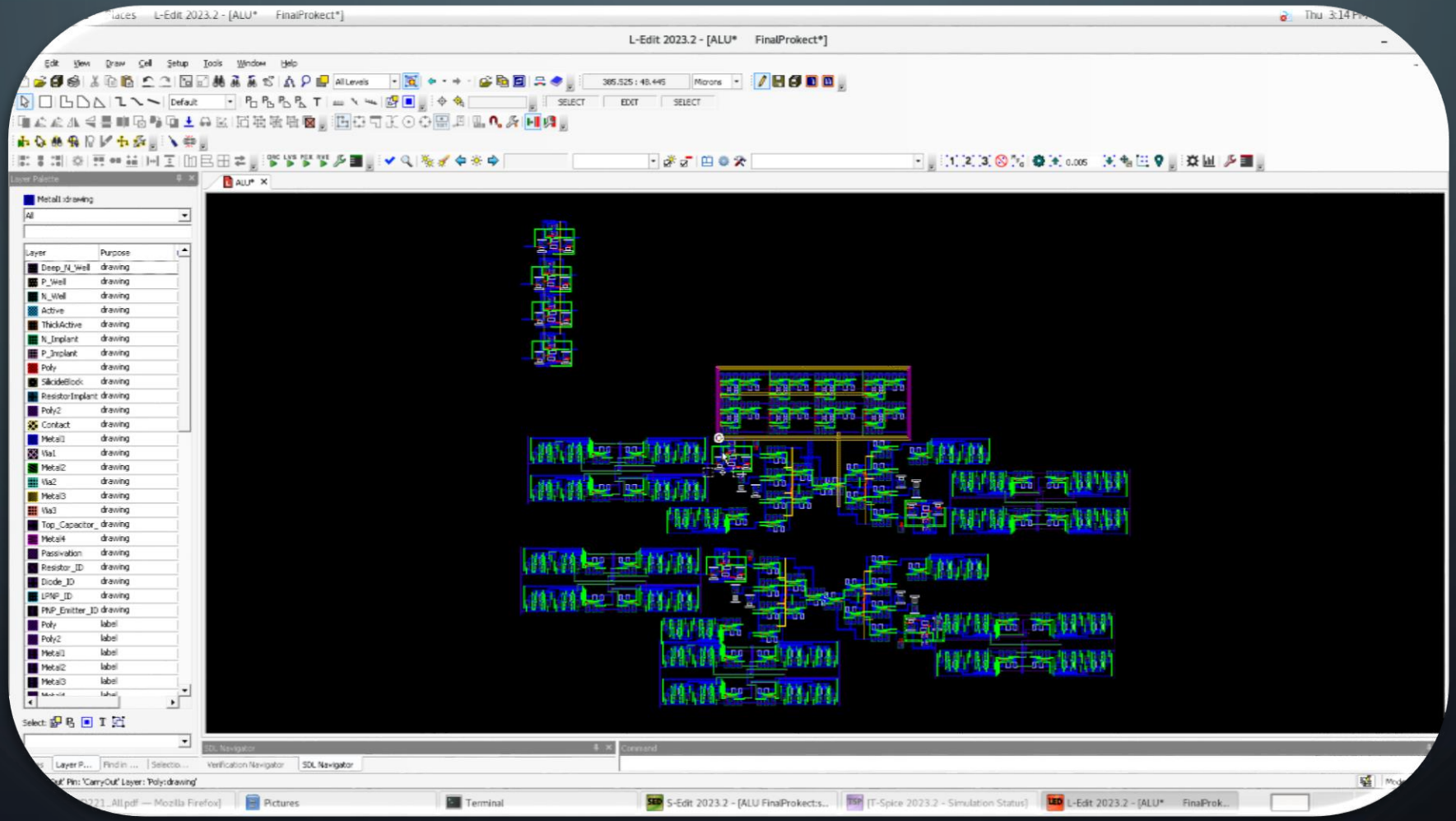
TOP MODULE


TOP_Module CND_Advanced_Full_Custom:schematic*
A B C D E F G H I
1 2 3 4 5 6 7
Out<0:3>
Memory_Bus<0:3>
ALU_out<3>
ALU_out<2>
ALU_out<1>
ALU_out<0>

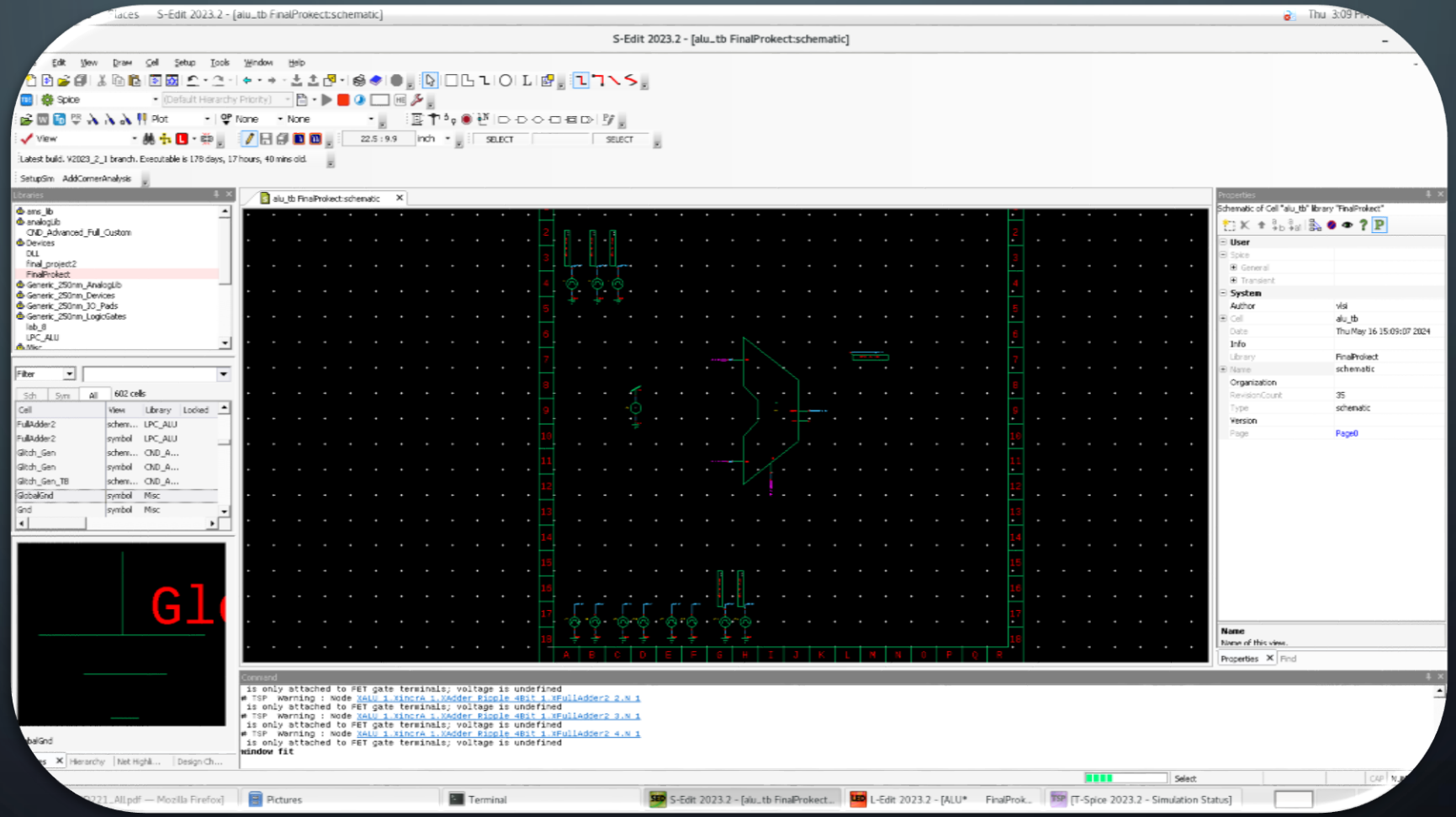
ALU


S-Edit 2023.2 - [ALU FinalProkect:schematic MicroProcessor/ALU_1]
point click 16.8 10.5
mode escape
cell open -cell MicroProcessor -library FinalProkect -type schematic -context Shifter_1 -tracenets
point click 23.8 13.5
cell open -cell MicroProcessor -library FinalProkect -type schematic -tracenets -x0 -1328 -y0 4348 -x1 24827 -y1 16286 -units iu
point click 12 9.8
mode escape
cell open -cell MicroProcessor -library FinalProkect -type schematic -context ALU_1 -tracenets

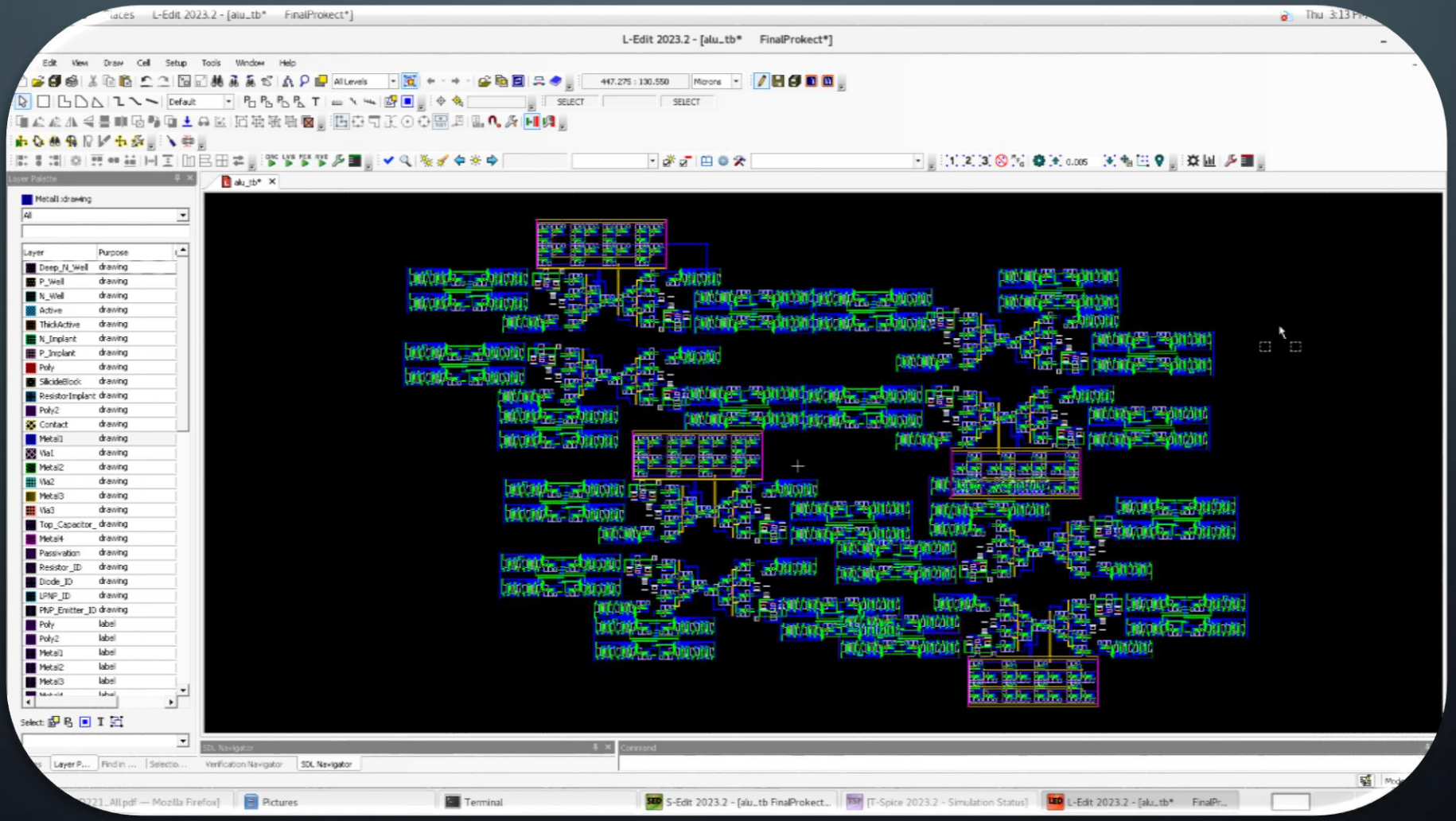
ALU


L-Edit 2023.2 - [ALU* FinalProkect*]
Edit View Draw Cell Setup Tools Window Help
Layer Palette
Metal1 drawing
Layer Purpose
Deep_N_Well drawing
P_Well drawing
N_Well drawing
Active drawing
ThickActive drawing
N_Implant drawing
P_Implant drawing
Poly drawing
SilicideBlock drawing
ResistorImplant drawing
Poly2 drawing
Contact drawing
Metal1 drawing
Via1 drawing
Metal2 drawing
Via2 drawing
Metal3 drawing
Via3 drawing
Top_Capacitor_ drawing
Metal4 drawing
Passivation drawing
Resistor_ID drawing
Diode_ID drawing
LPNP_ID drawing
PNP_Emitter_ID drawing
Poly label
Poly2 label
Metal1 label
Metal2 label
Metal3 label
ALU*
SDL Navigator
Command
Pictures
Terminal

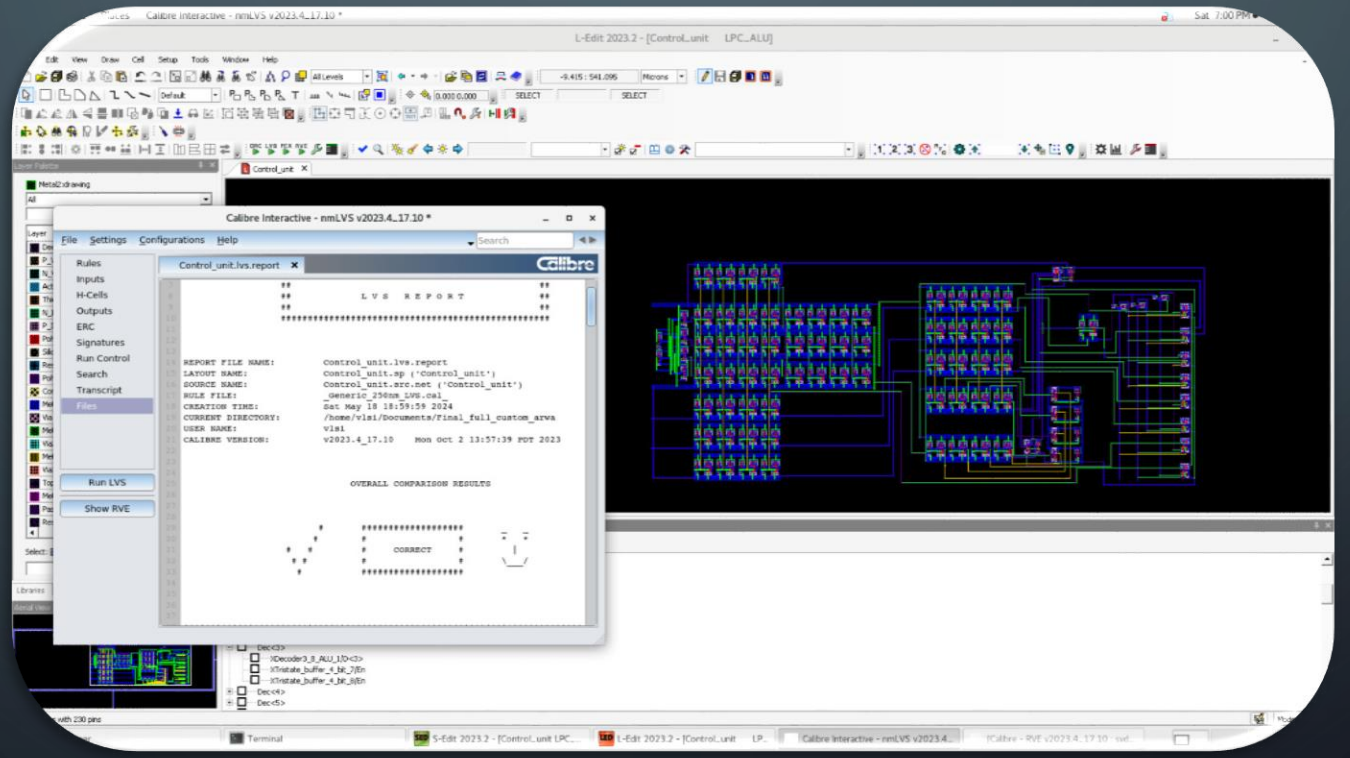
ALU TB


S-Edit 2023.2 - [alu_tb FinalProkect:schematic]
alu_tb FinalProkect:schematic
Command
is only attached to FET gate terminals; voltage is undefined
TSP warning : Node XALU_1.XincrA_1.XAdder_Ripple_4Bit_1.XFullAdder2_2.N_1
is only attached to FET gate terminals; voltage is undefined
TSP warning : Node XALU_1.XincrA_1.XAdder_Ripple_4Bit_1.XFullAdder2_3.N_1
is only attached to FET gate terminals; voltage is undefined
TSP warning : Node XALU_1.XincrA_1.XAdder_Ripple_4Bit_1.XFullAdder2_4.N_1
is only attached to FET gate terminals; voltage is undefined
window fit

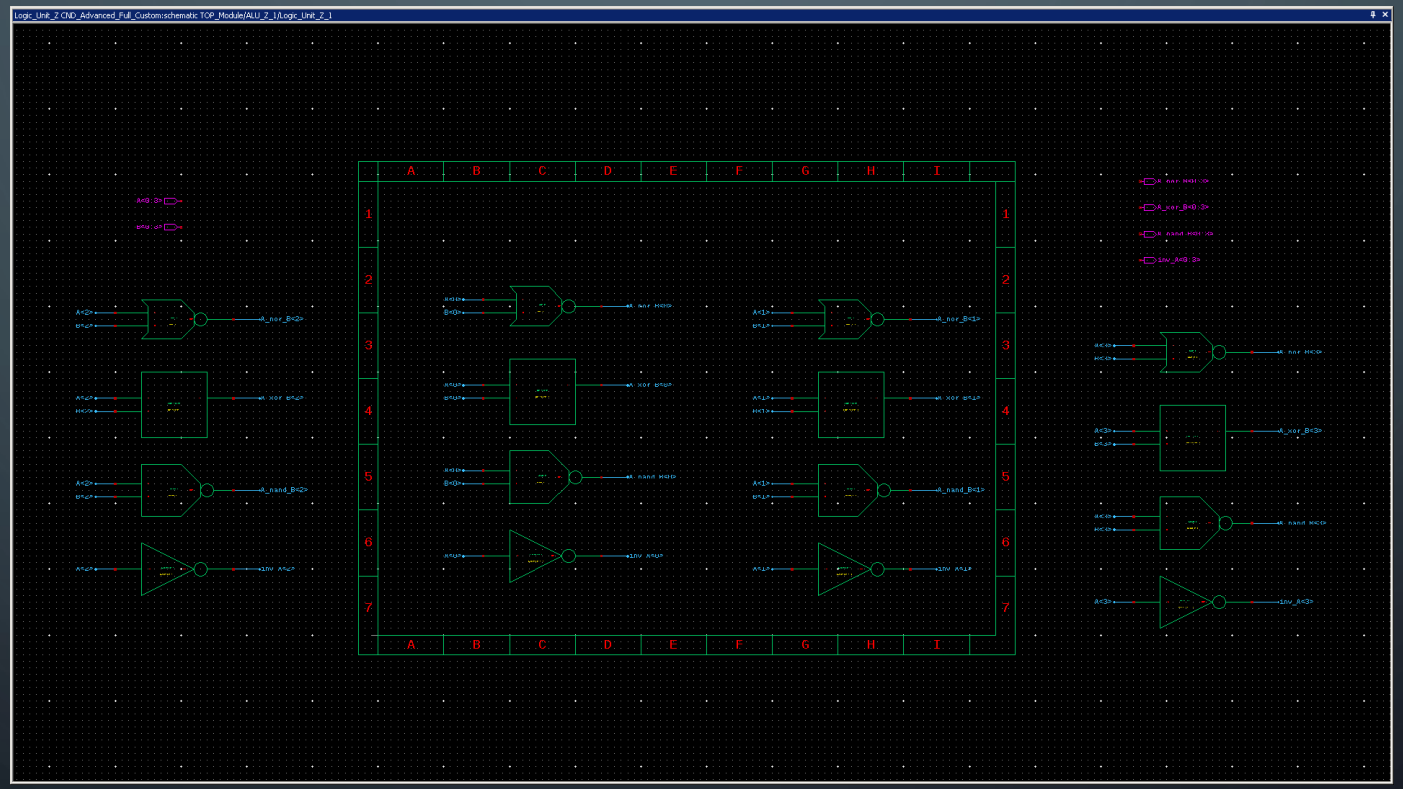
ALU TB


L-Edit 2023.2 - [alu_tb* FinalProkect*]
Edit View Draw Cell Setup Tools Window Help
Layer Palette
Layer Purpose
Deep_N_Well drawing
P_Well drawing
N_Well drawing
Active drawing
ThickActive drawing
N_Implant drawing
P_Implant drawing
Poly drawing
SilicideBlock drawing
ResistorImplant drawing
Poly2 drawing
Contact drawing
Metal1 drawing
Via1 drawing
Metal2 drawing
Via2 drawing
Metal3 drawing
Via3 drawing
Top_Capacitor drawing
Metal4 drawing
Passivation drawing
Resistor_ID drawing
Diode_ID drawing
LPNP_ID drawing
PNP_Emitter_ID drawing
Poly label
Poly2 label
Metal1 label
Metal2 label
Metal3 label
Verification Navigator
SDL Navigator
Command
Pictures
Terminal

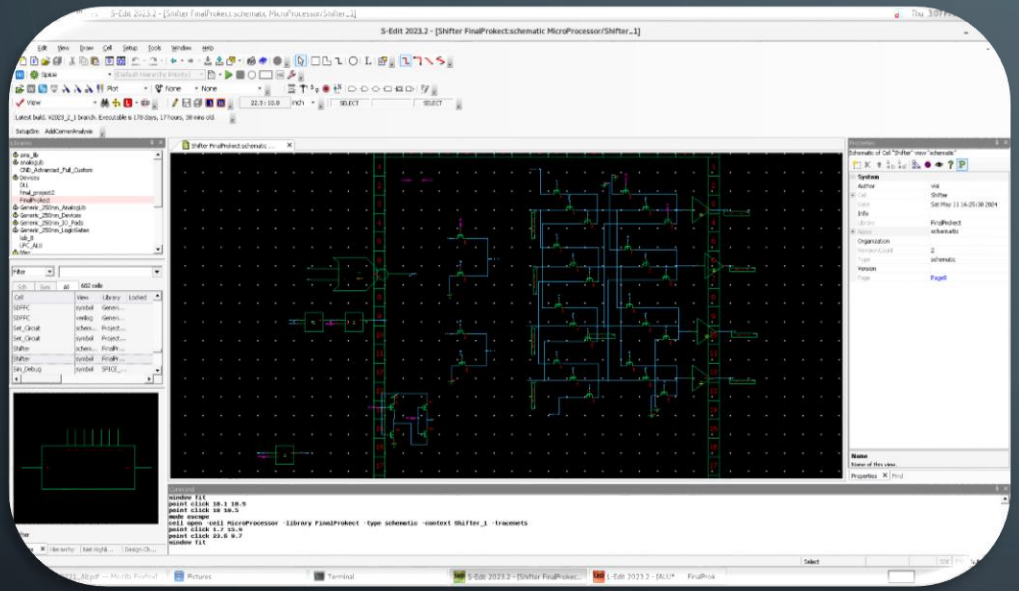
CONTROL UNIT

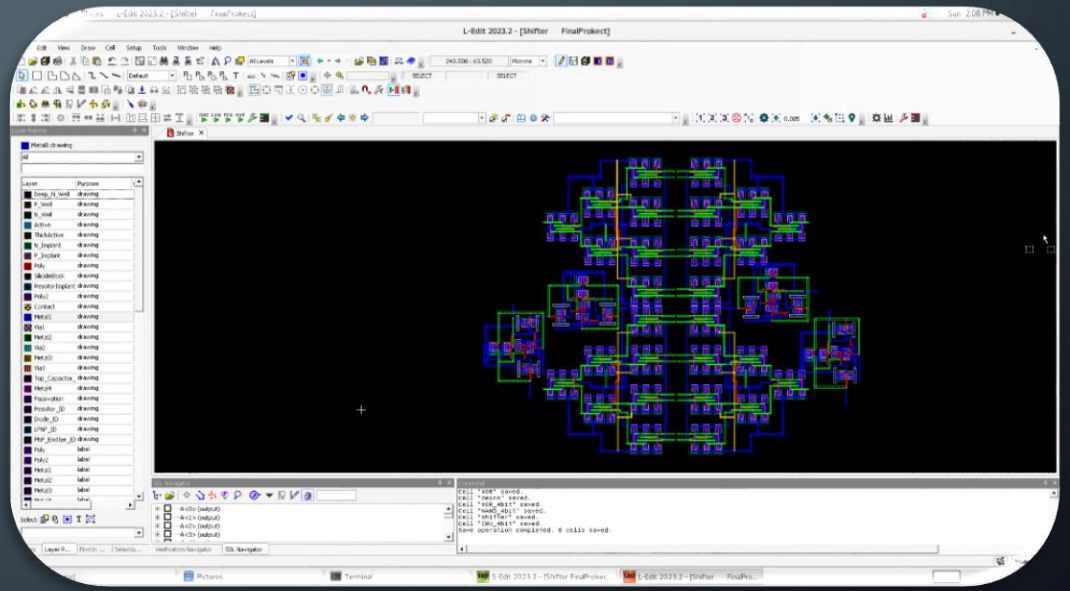


LOGIC UNIT

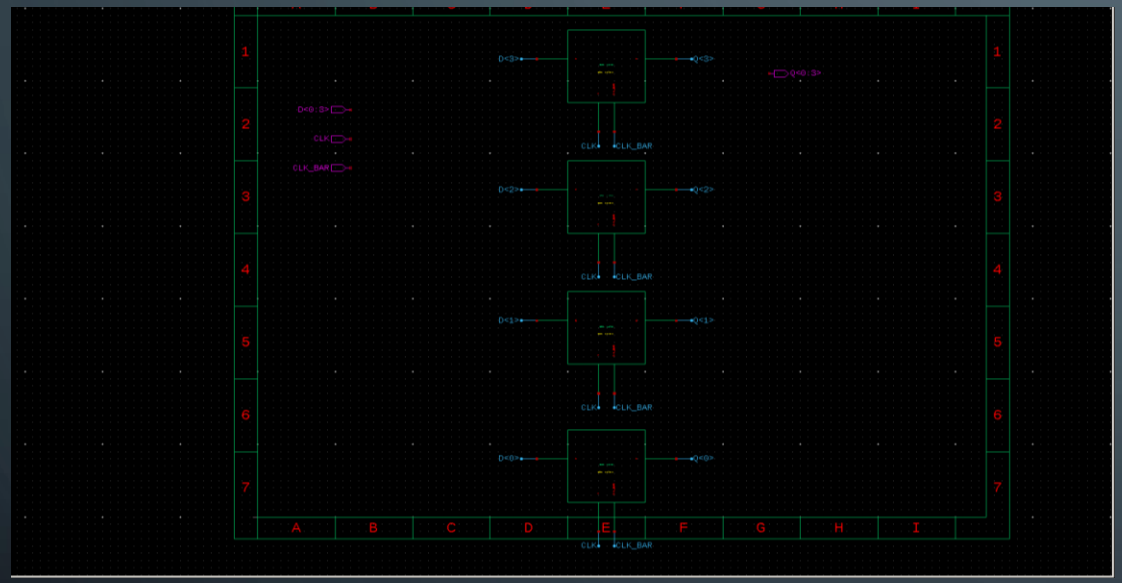

Logic_Unit_2_CND_Advanced_Full_Custom:schematic TOP_Module/ALU_2_1/Logic_Unit_2_1
A
B
C
D
E
F
G
H
I
1
2
3
4
5
6
7

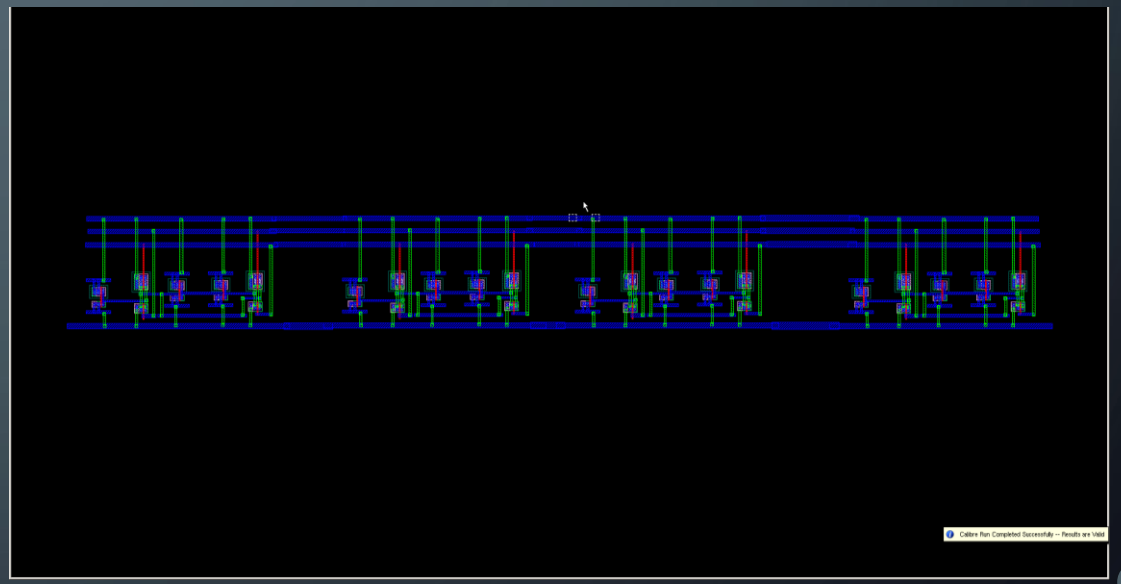
SHIFTER



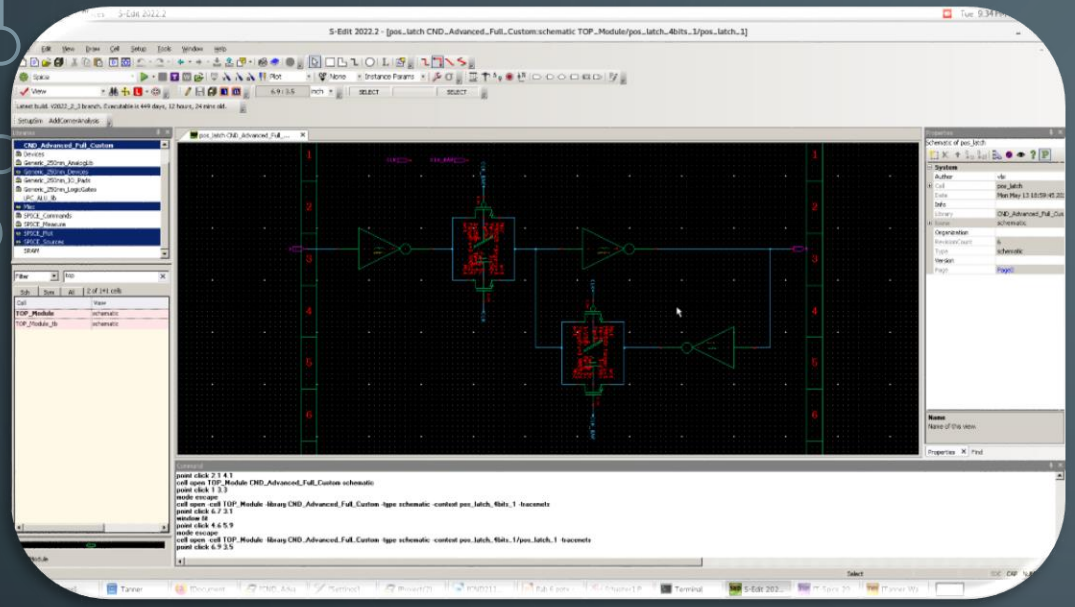


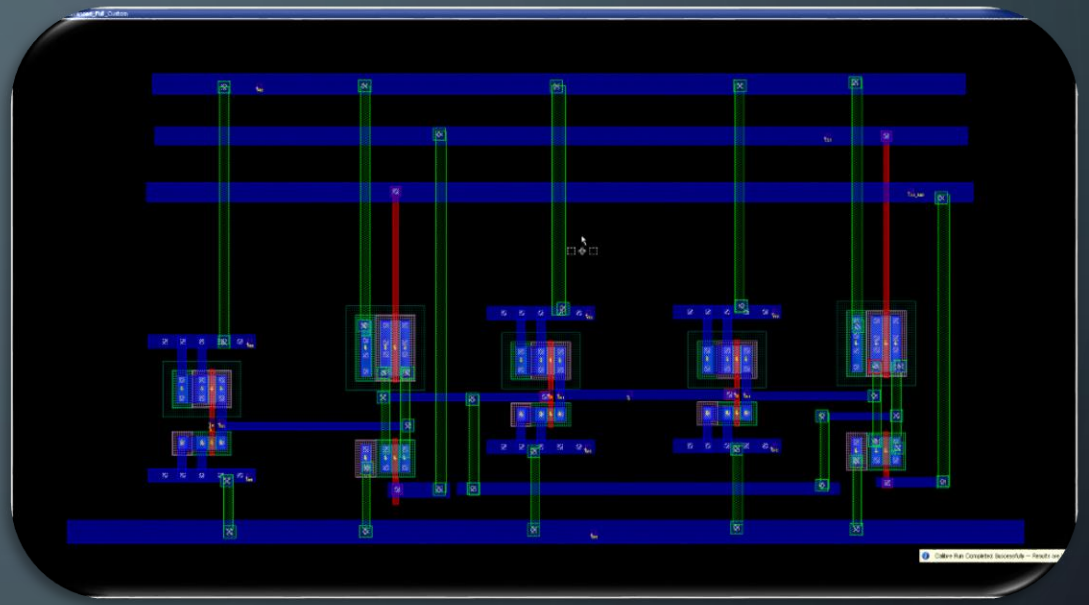
4BIT LATCH

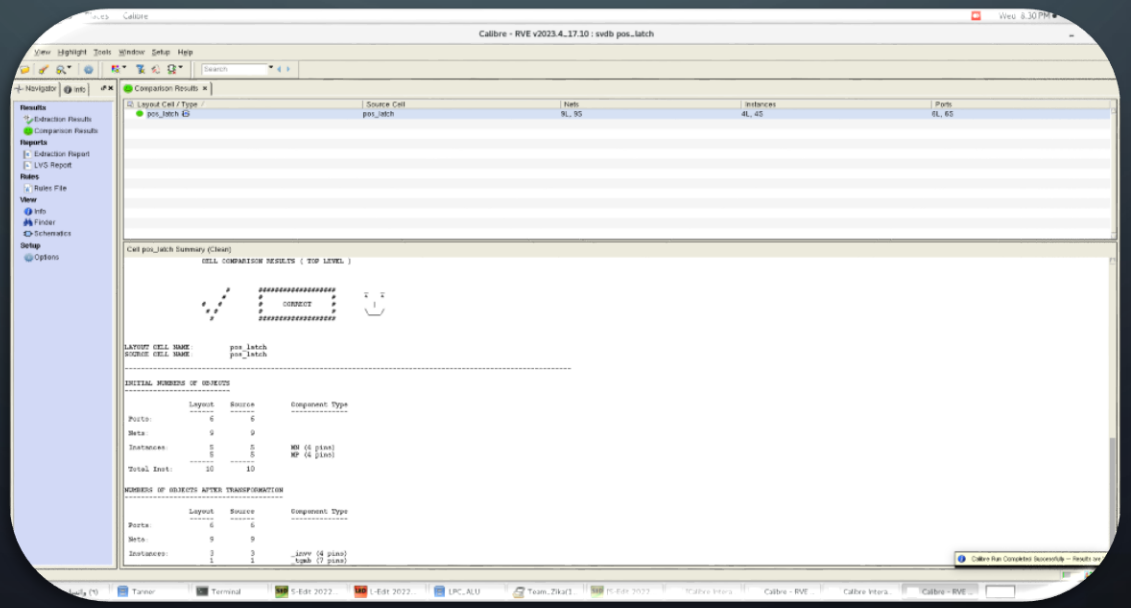

D<3>
Q<3>
Q<0:3>
D<0:3>
CLK
CLK_BAR
D<2>
Q<2>
D<1>
Q<1>
D<0>
Q<0>
1
2
3
4
5
6
7
A
B
C
D
E
F
G
H
I


Calibre Run Completed Successfully -- Results are Valid

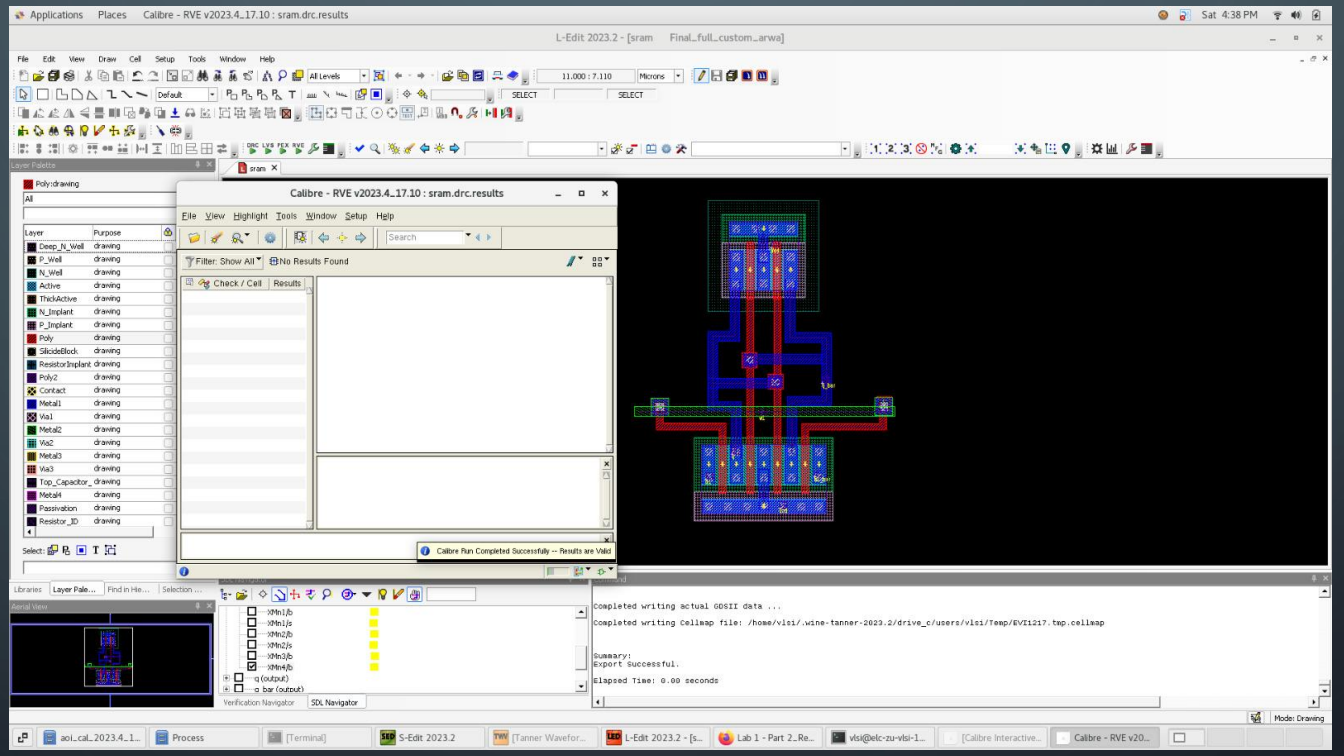
POS LATCH



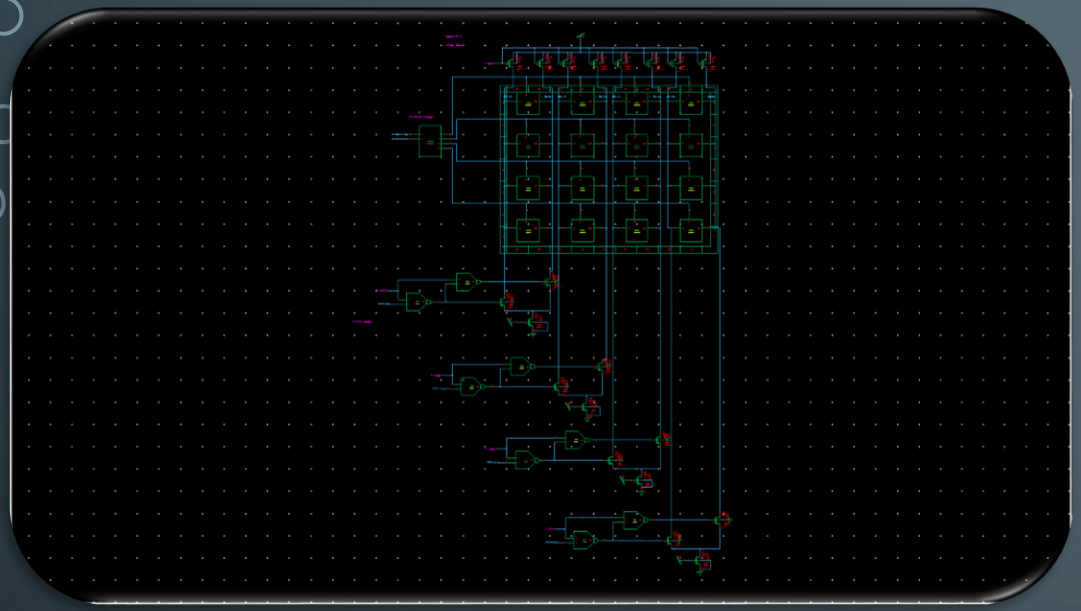


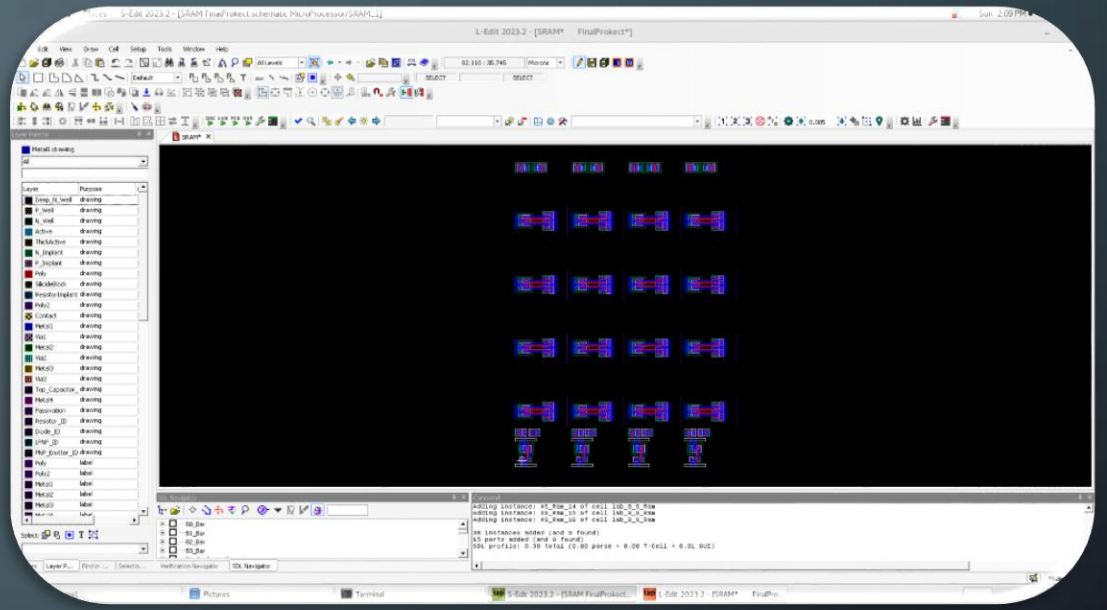


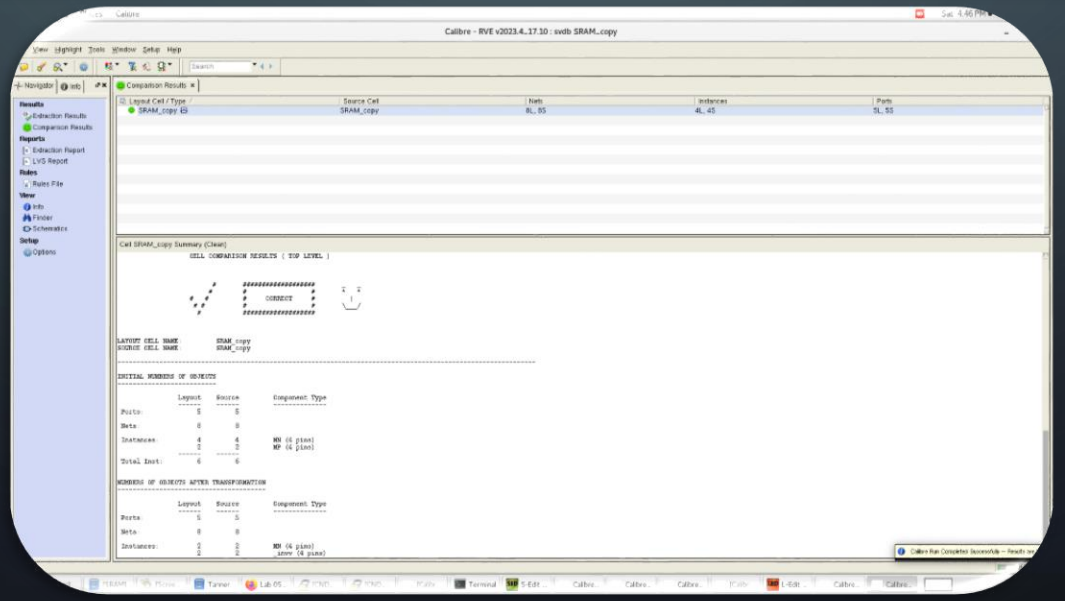
S RAM


Calibre - RVE v2023.4_17.10 : sram.drc.results
L-Edit 2023.2 - [sram Final_full_custom_arwa]
Calibre Run Completed Successfully -- Results are Valid
Completed writing actual GDSII data ...
Completed writing Cellmap file: /home/vlsi/.wine-tanner-2023.2/drive_c/users/vlsi/Temp/EVI1217.tmp.cellmap
Summary:
Export Successful.
Elapsed Time: 0.00 seconds

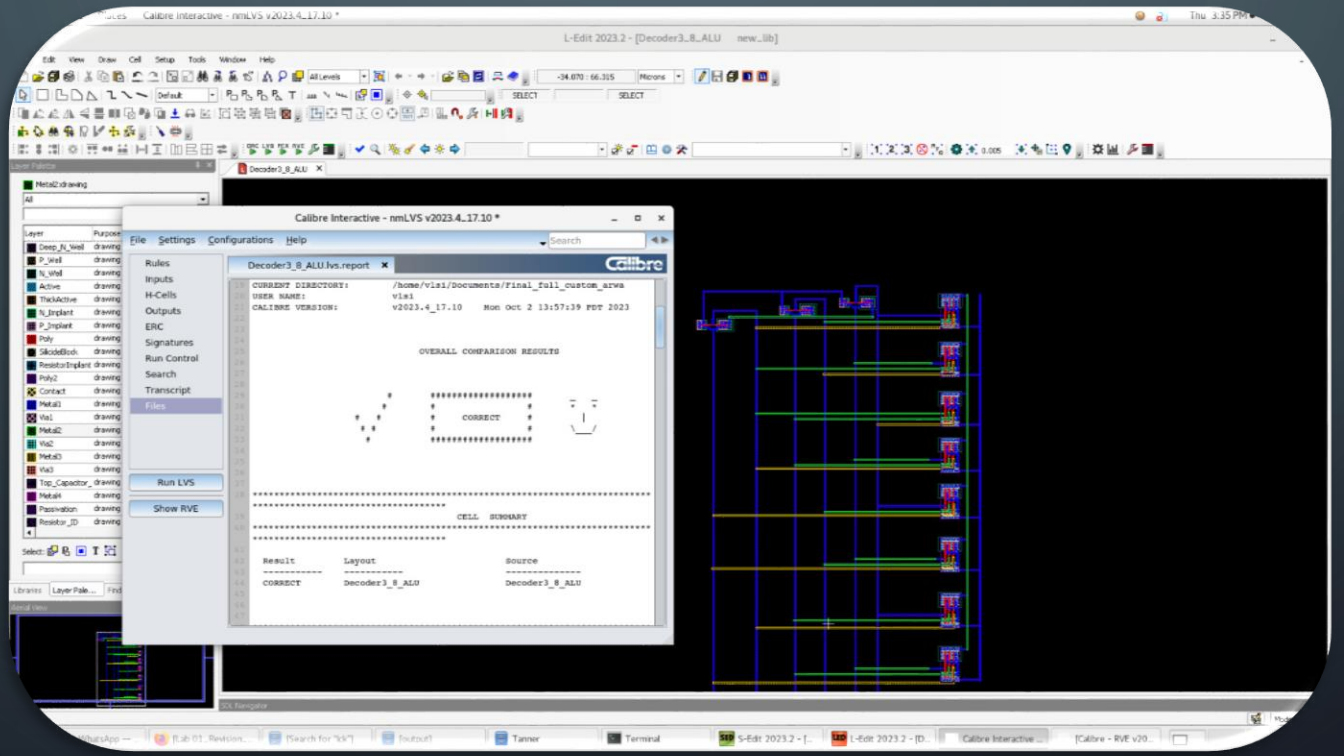
S RAM 4*4



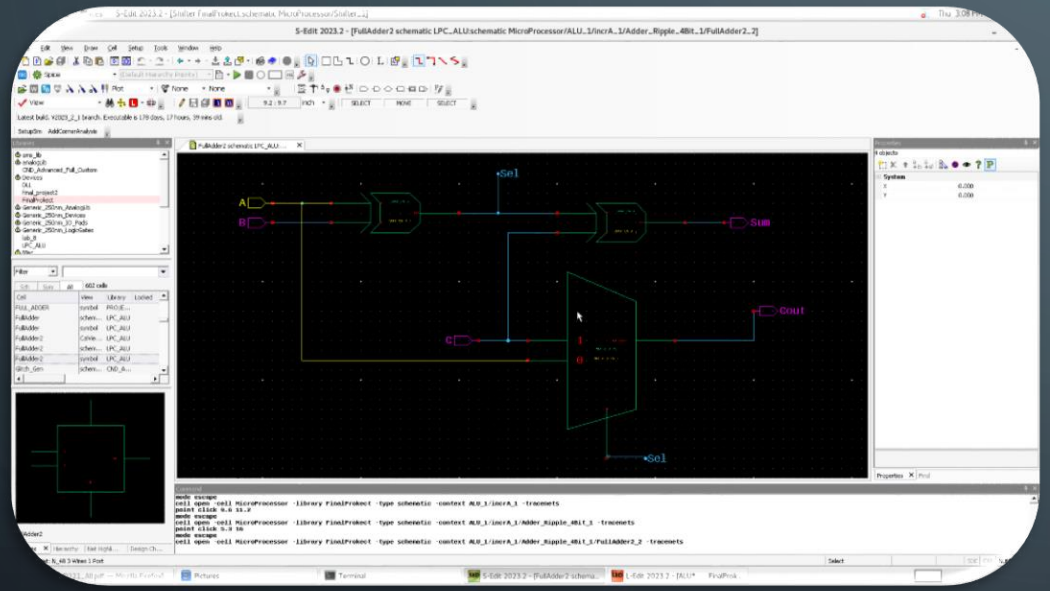


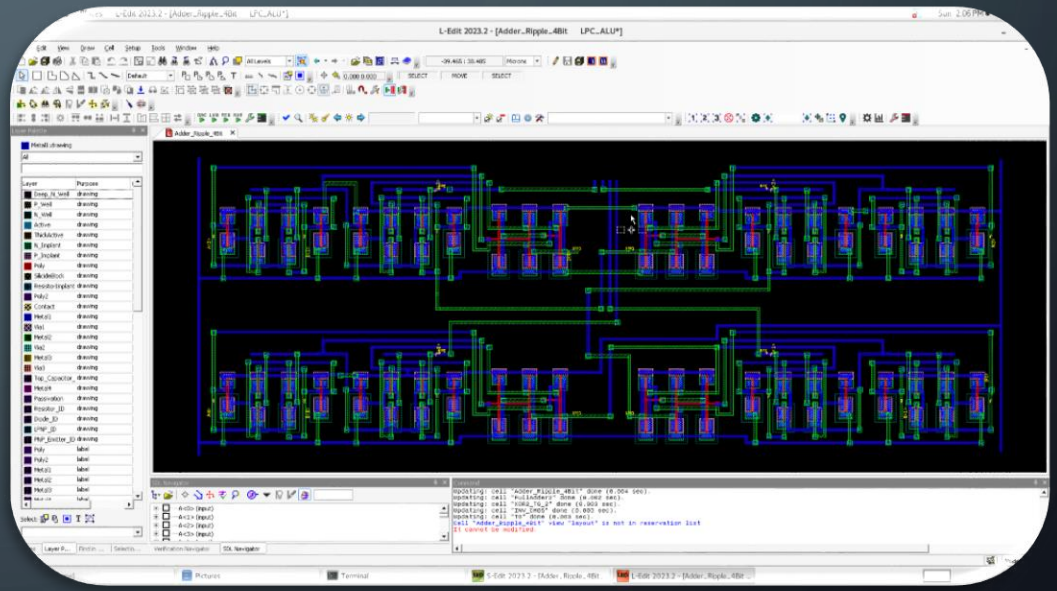


S RAM DECODER

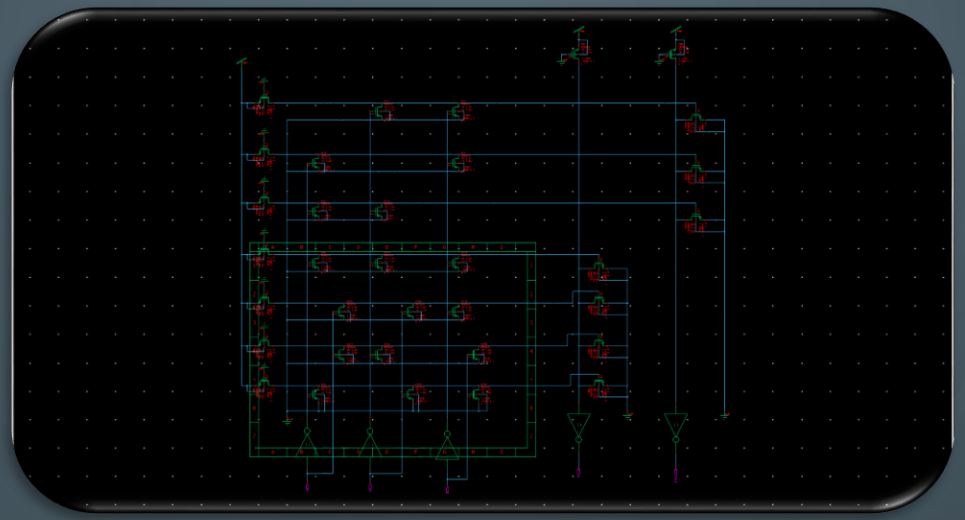


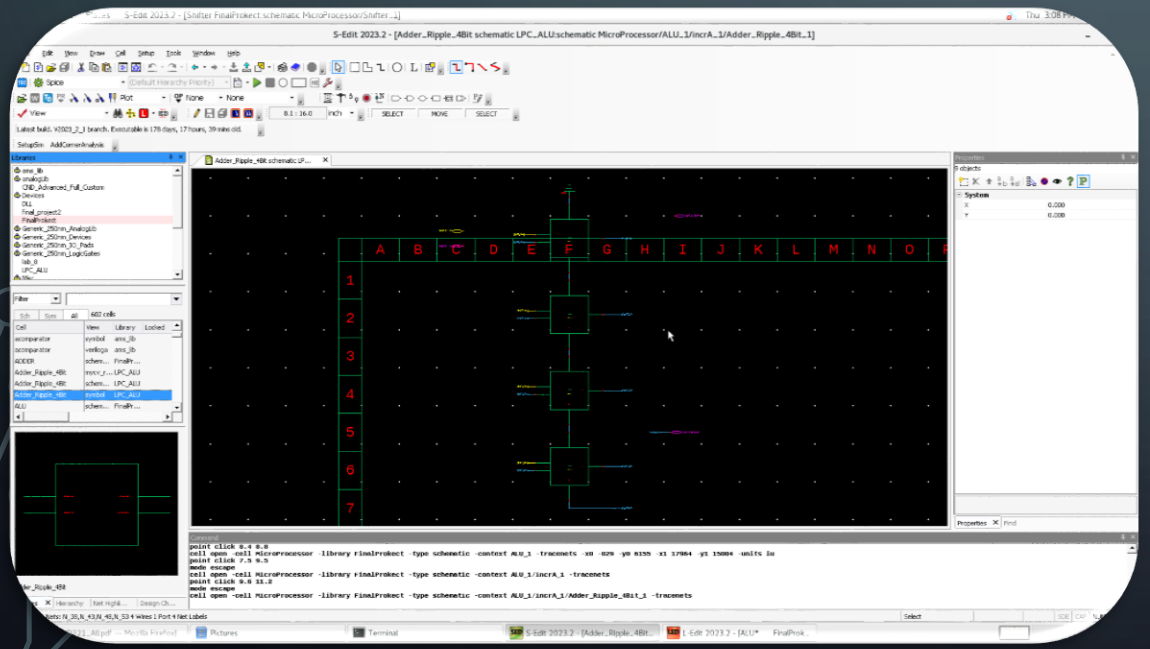
FULL ADDER

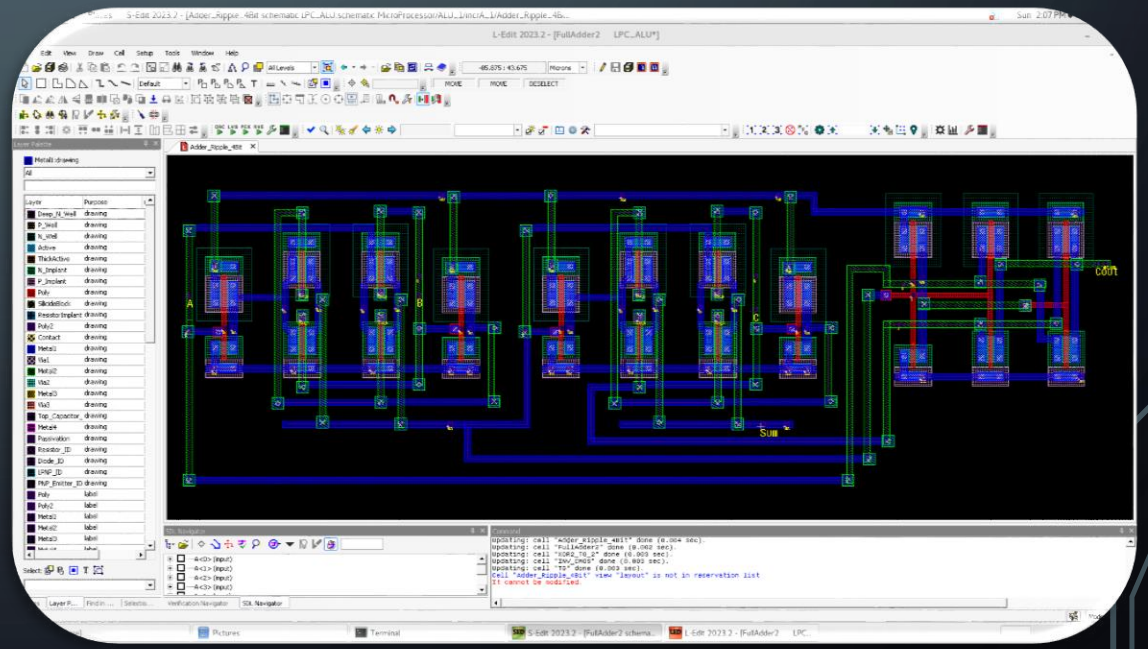




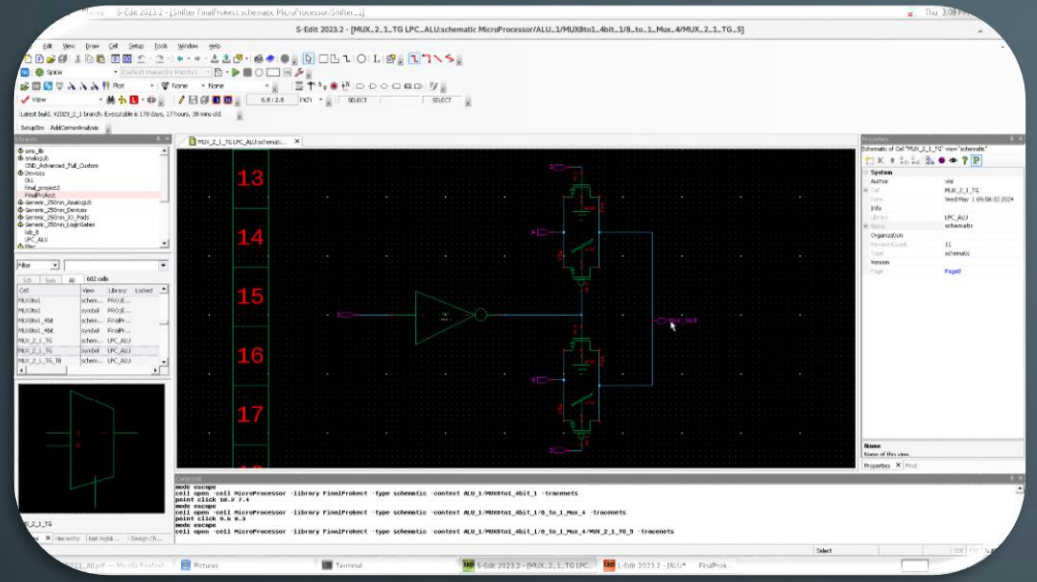
ADDER RIBBLE

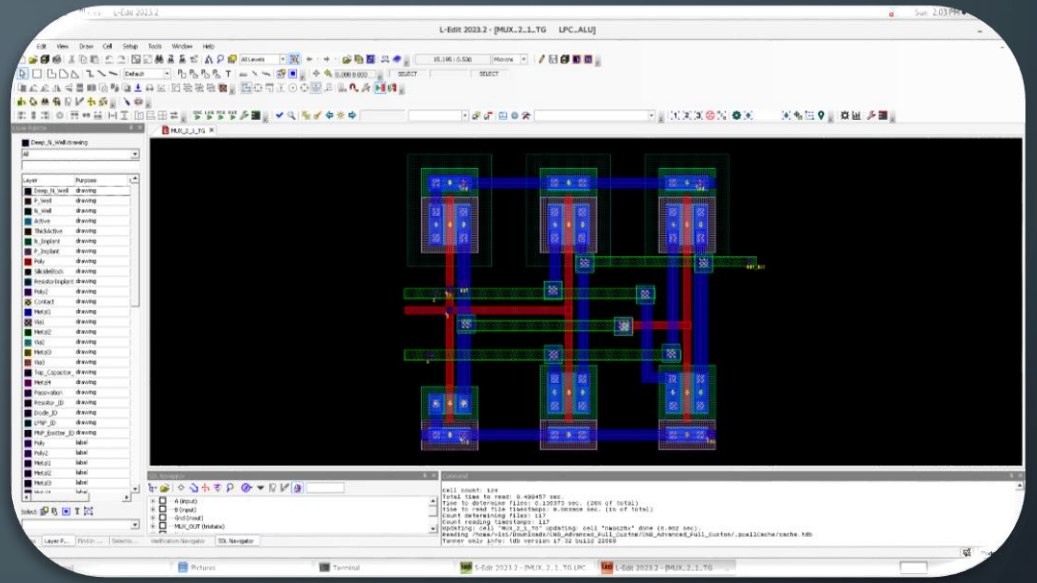


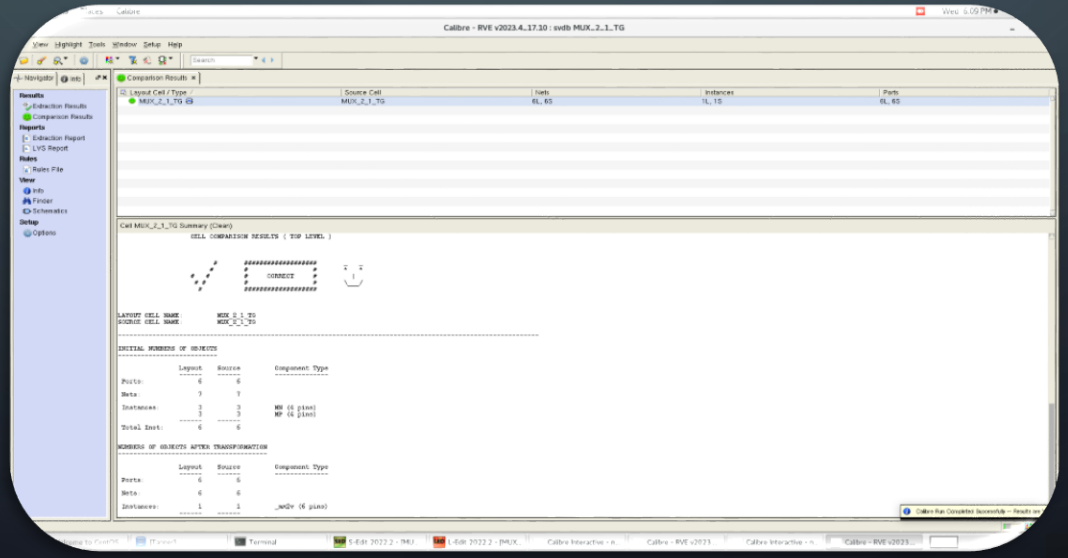




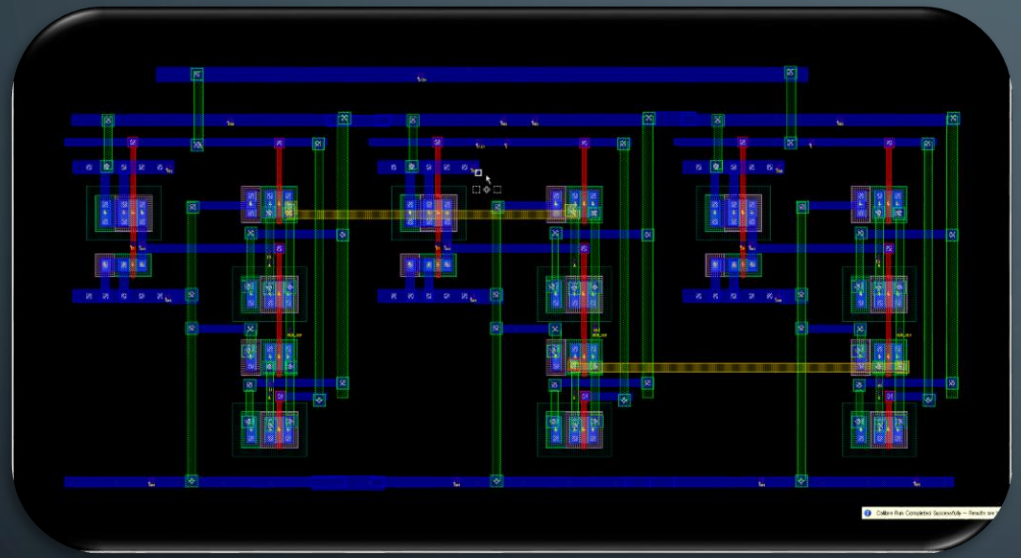
MUX 2 TO 1

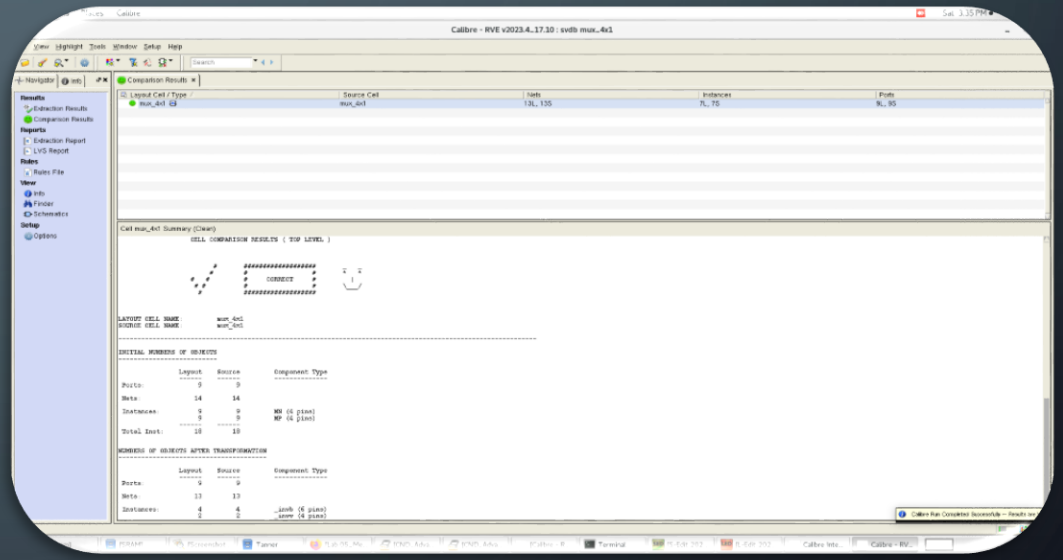




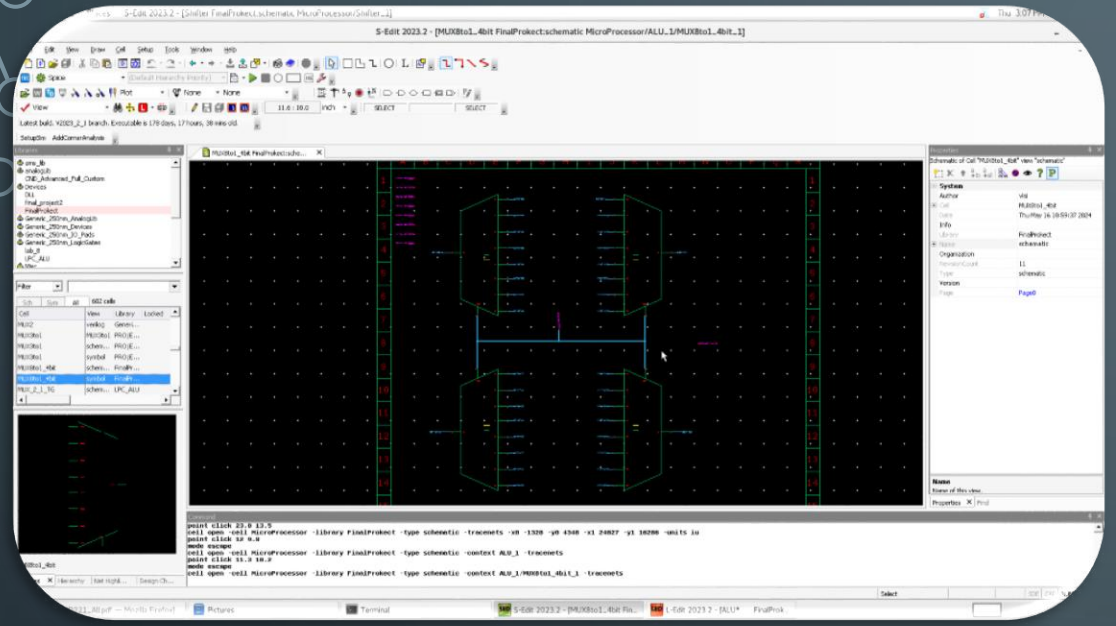


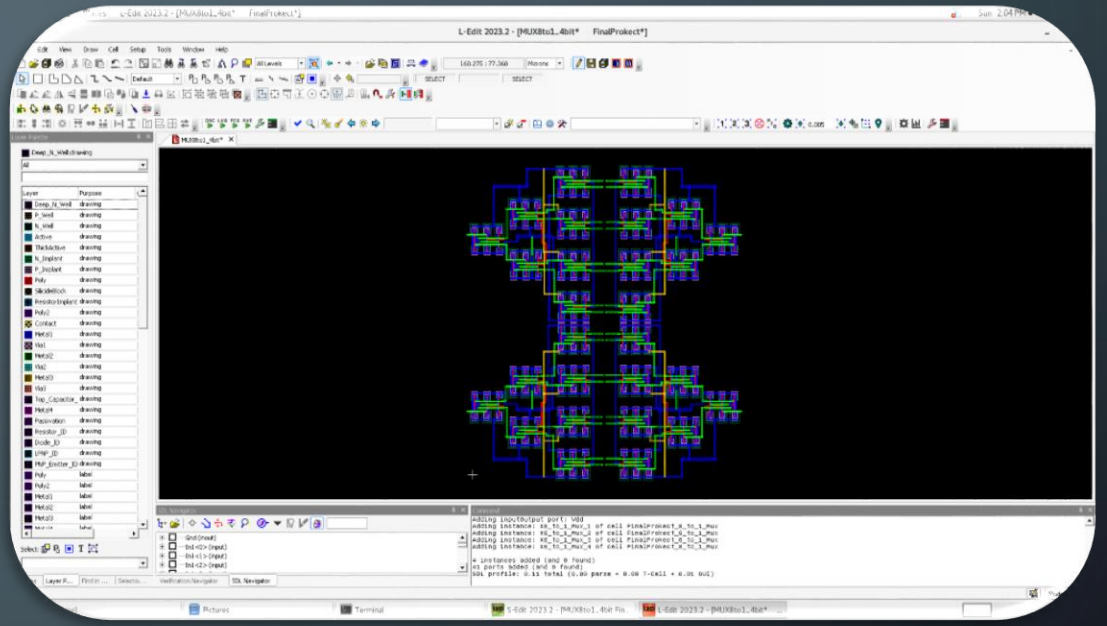
MUX 4*1

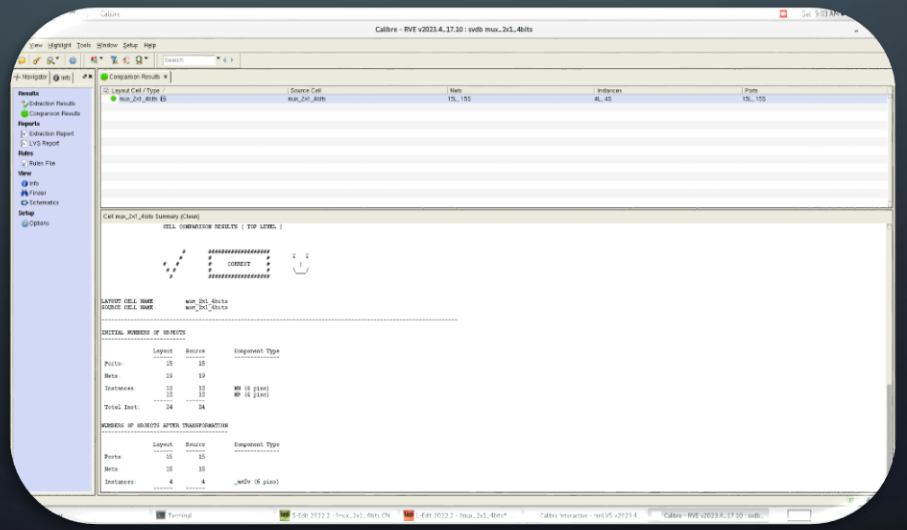




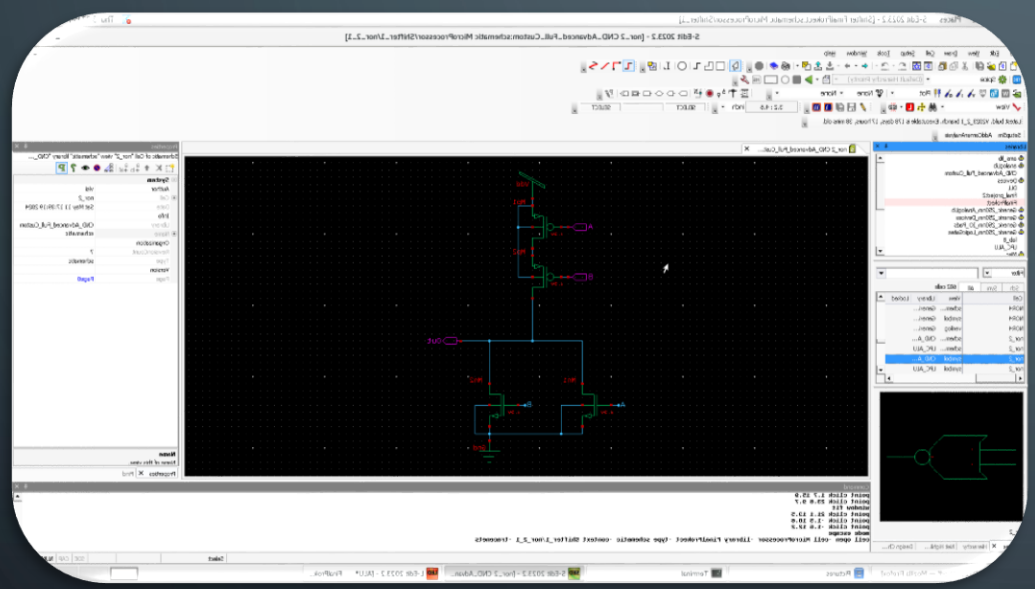
MUX8TO1 4BIT

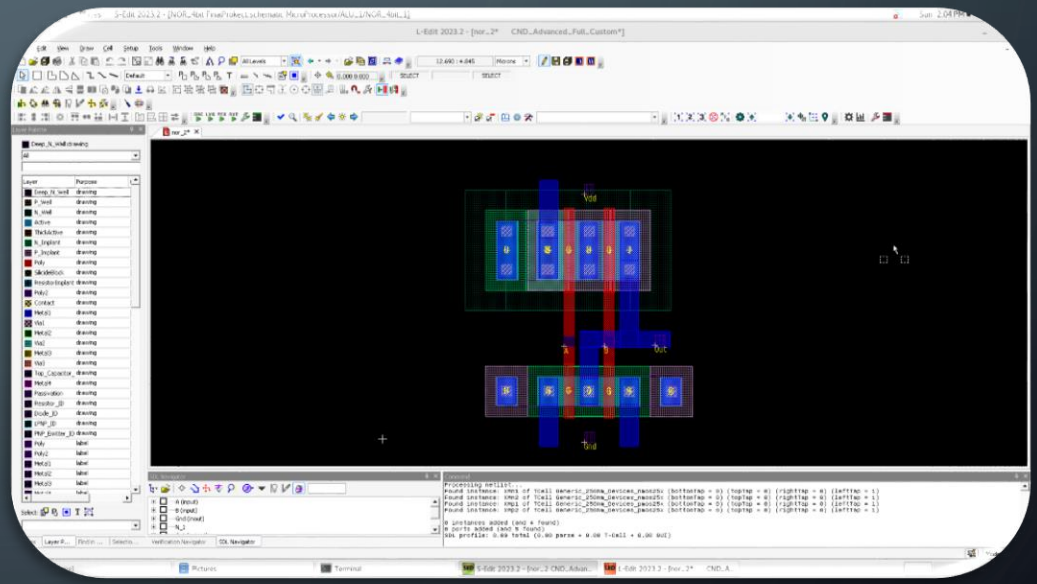




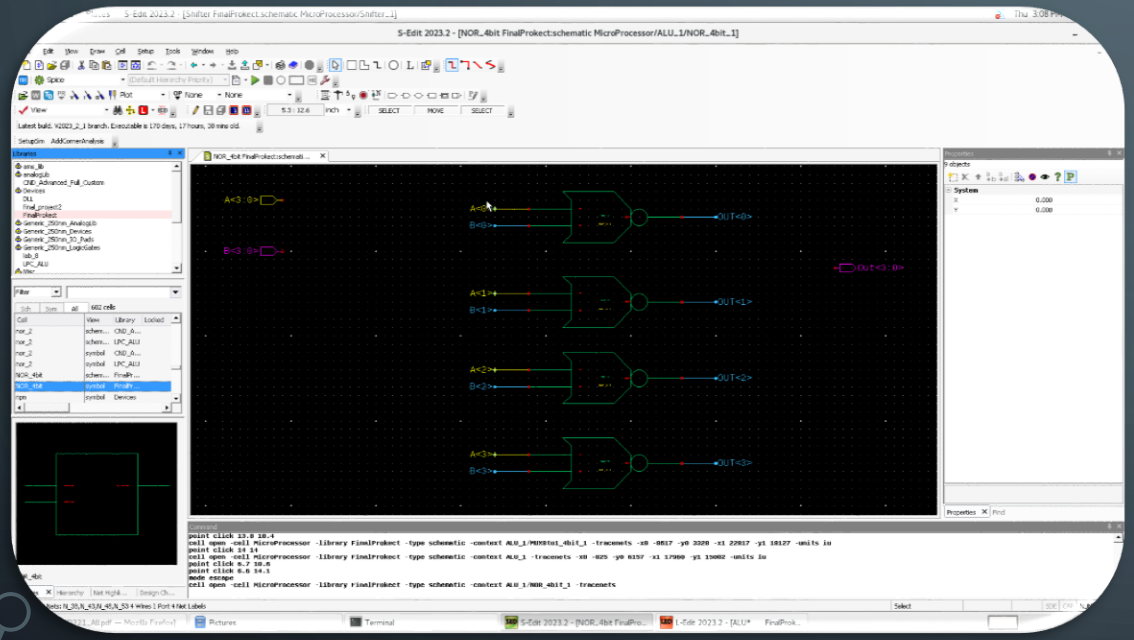


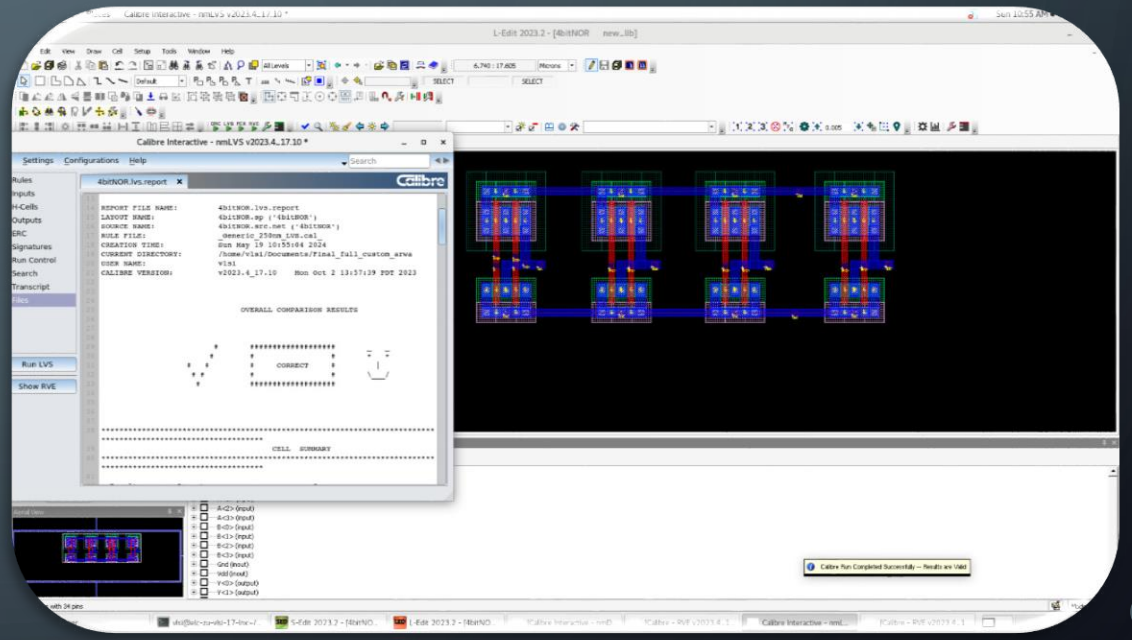
NOR



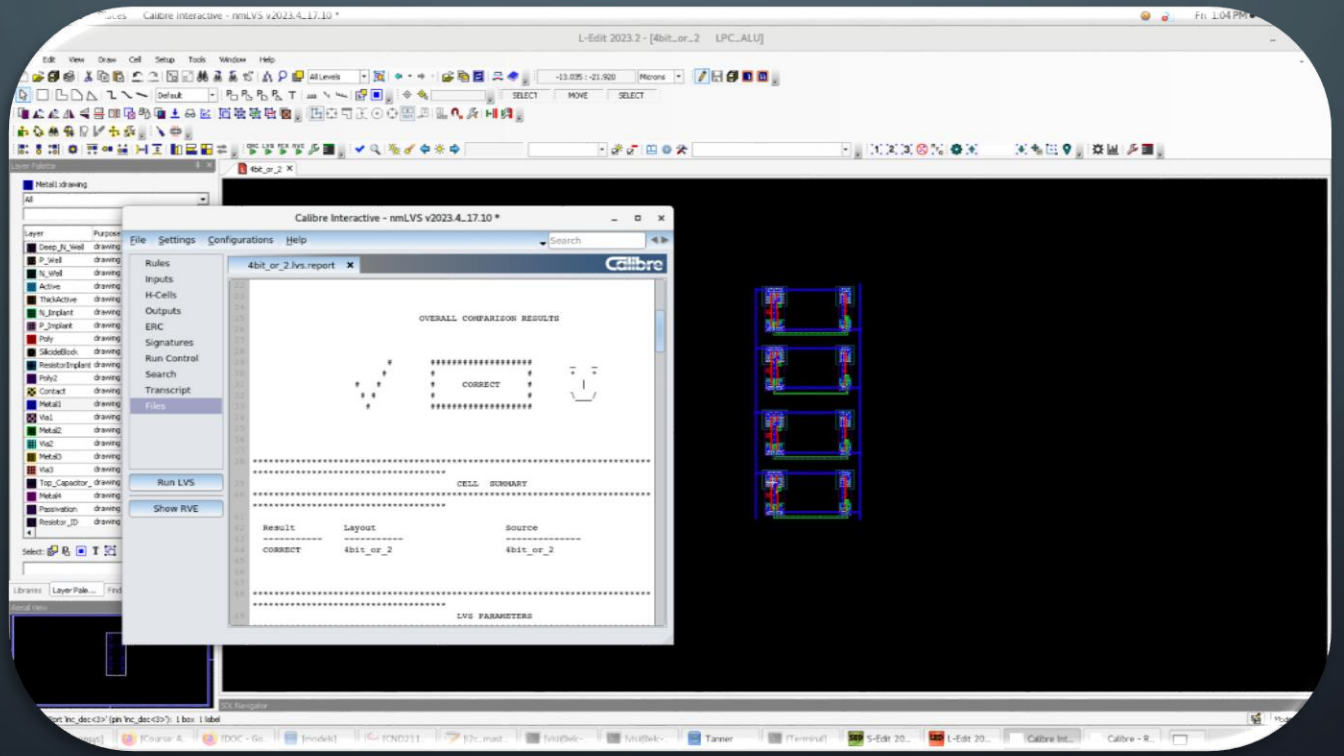


NOR 4 BIT

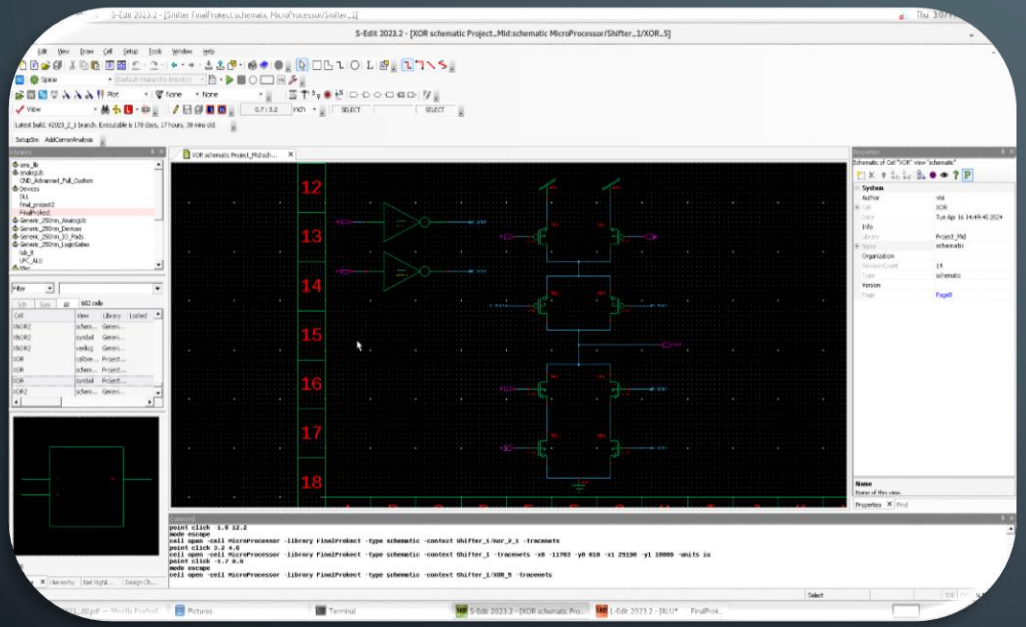

5-Edit 2023.2 - [NOR_4bit FinalProkect:schematic MicroProcessor/ALU_1/NOR_4bit_1]
A<3:0>
B<3:0>
OUT<0>
OUT<1>
OUT<2>
OUT<3>
Out<3:0>

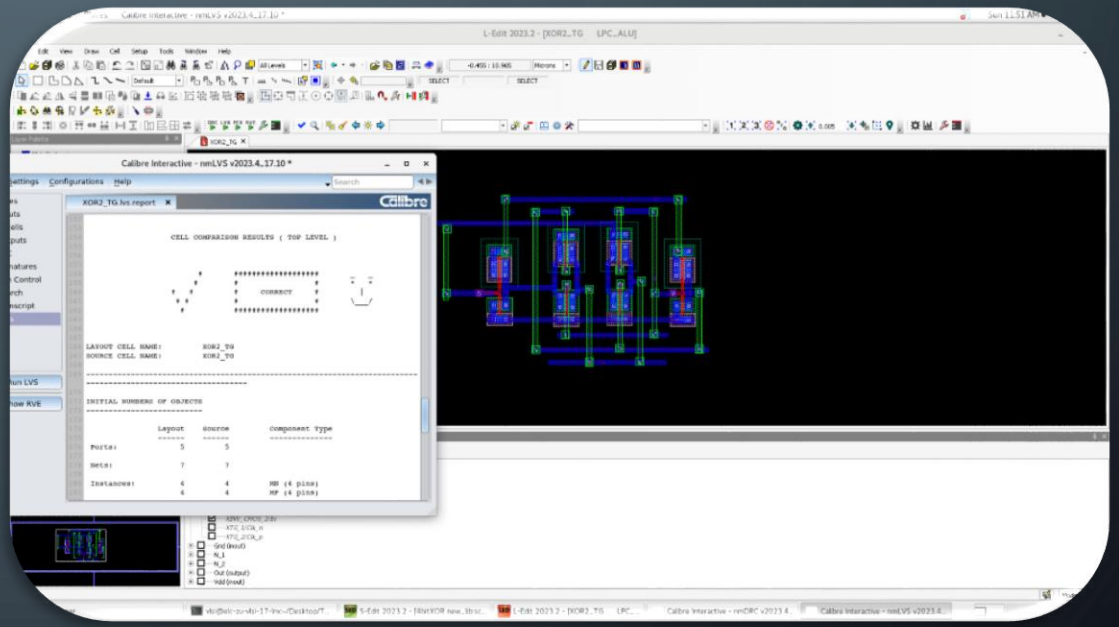

Calibre Interactive - nmLVS v2023.4_17.10 *
4bitNOR.lvs.report
OVERALL COMPARISON RESULTS
CORRECT
CELL SUMMARY

4 BIT OR

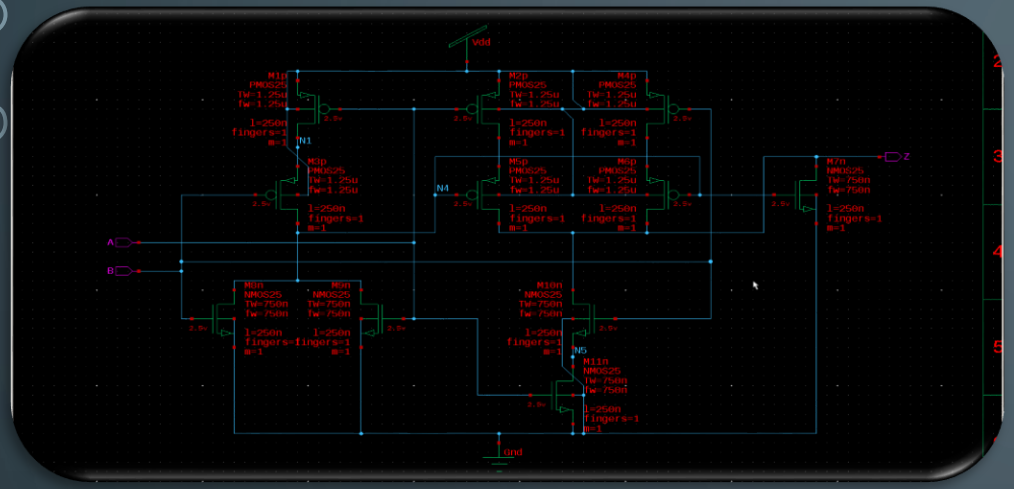

Calibre Interactive - nmLVS v2023.4_17.10 *
File Settings Configurations Help
Rules
Inputs
H-Cells
Outputs
ERC
Signatures
Run Control
Search
Transcript
Files
Run LVS
Show RVE
4bit_or_2.lvs.report
OVERALL COMPARISON RESULTS
CORRECT
CELL SUMMARY
Result Layout Source
CORRECT 4bit_or_2 4bit_or_2
LVS PARAMETERS

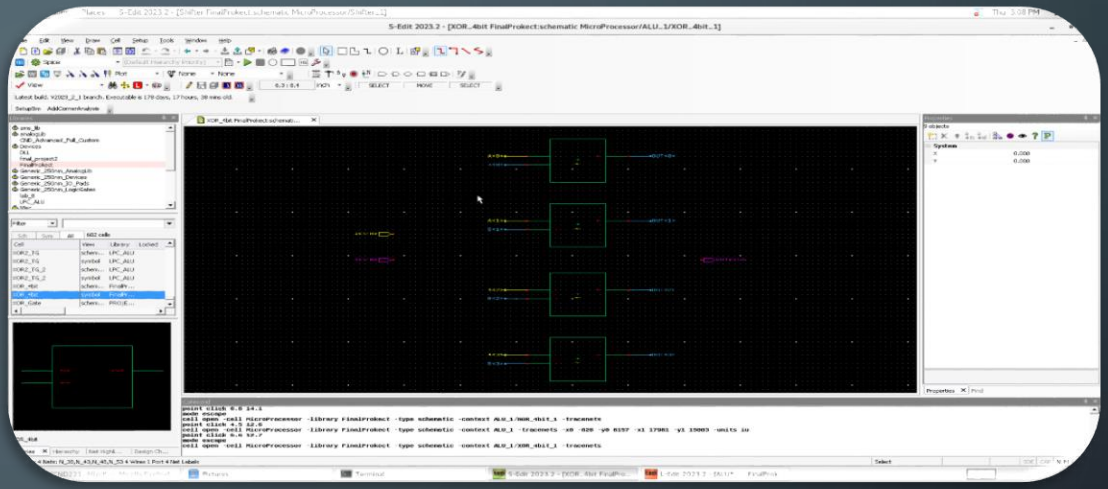
XOR

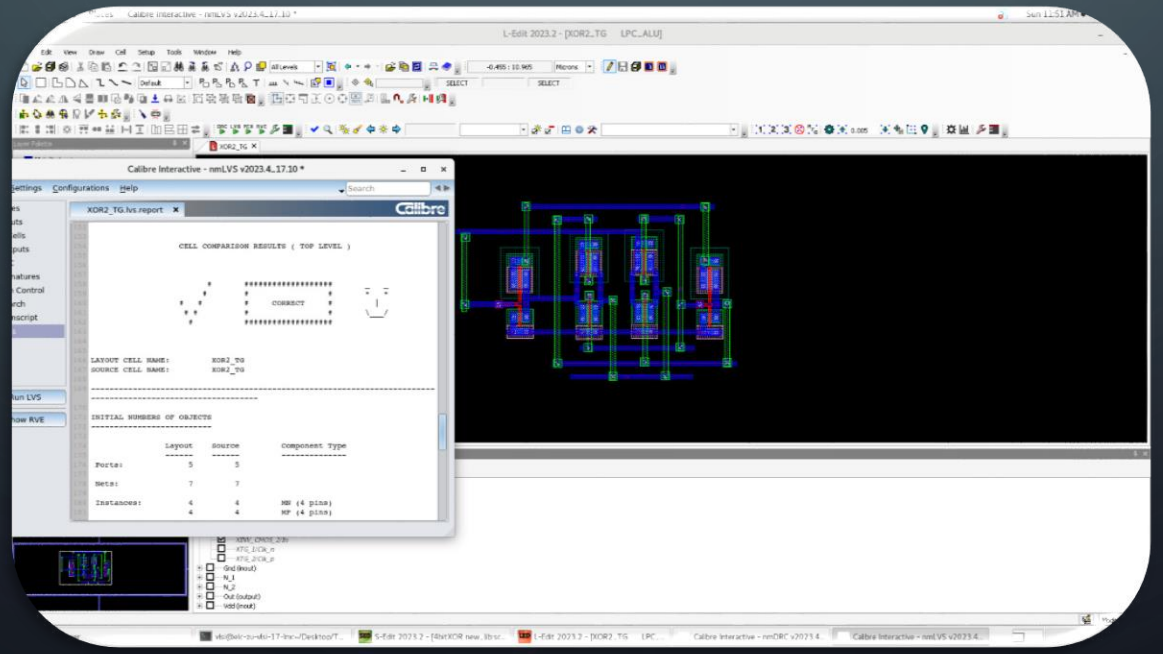

S-Edit 2023.2 - [XOR schematic Project_Mid:schematic MicroProcessor/Shifter_1/XOR_5]
12
13
14
15
16
17
18


Calibre Interactive - nmLVS v2023.4_17.10 *
CELL COMPARISON RESULTS (TOP LEVEL)
CORRECT
LAYOUT CELL NAME: XOR2_TG
SOURCE CELL NAME: XOR2_TG
INITIAL NUMBERS OF OBJECTS
Layout
Source
Component Type
Ports: 5 5
Nets: 7 7
Instances: 4 4 NM (4 pins)
4 4 MP (4 pins)

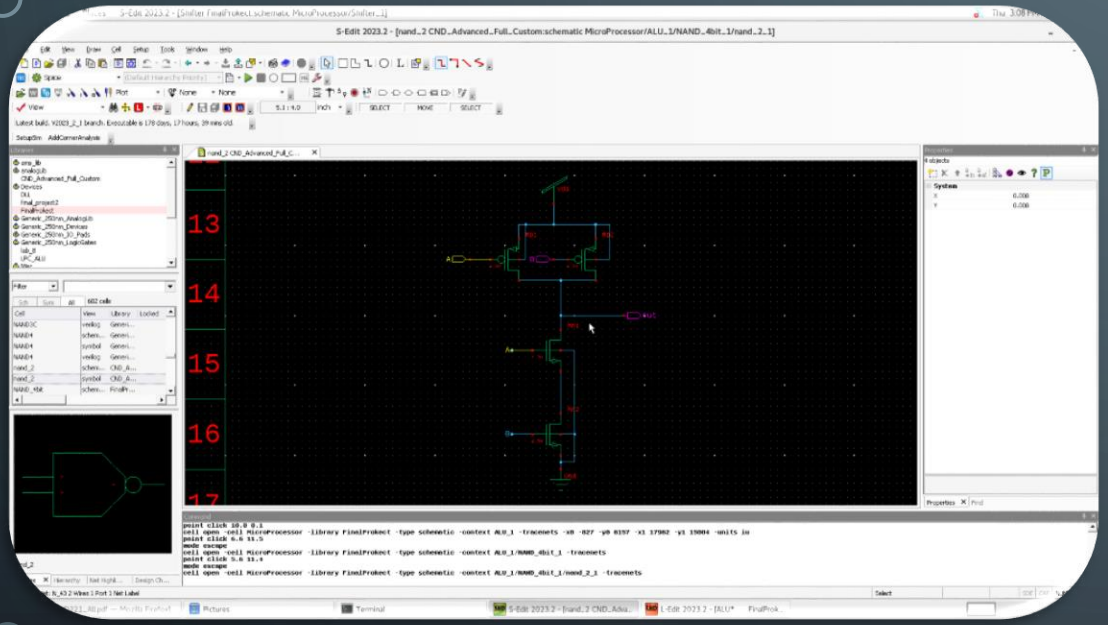
XOR 4 BIT

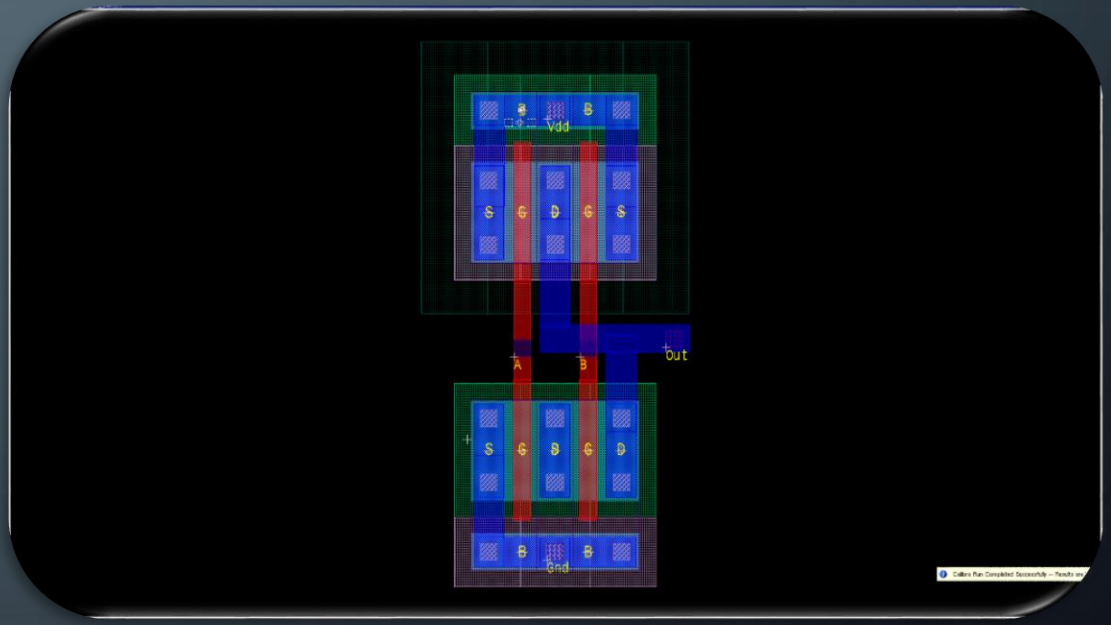

Vdd
Gnd
M1p
M2p
M4p
M3p
M5p
M6p
M7n
M8n
M9n
M10n
M11n
N1
N4
N5
A
B
Z



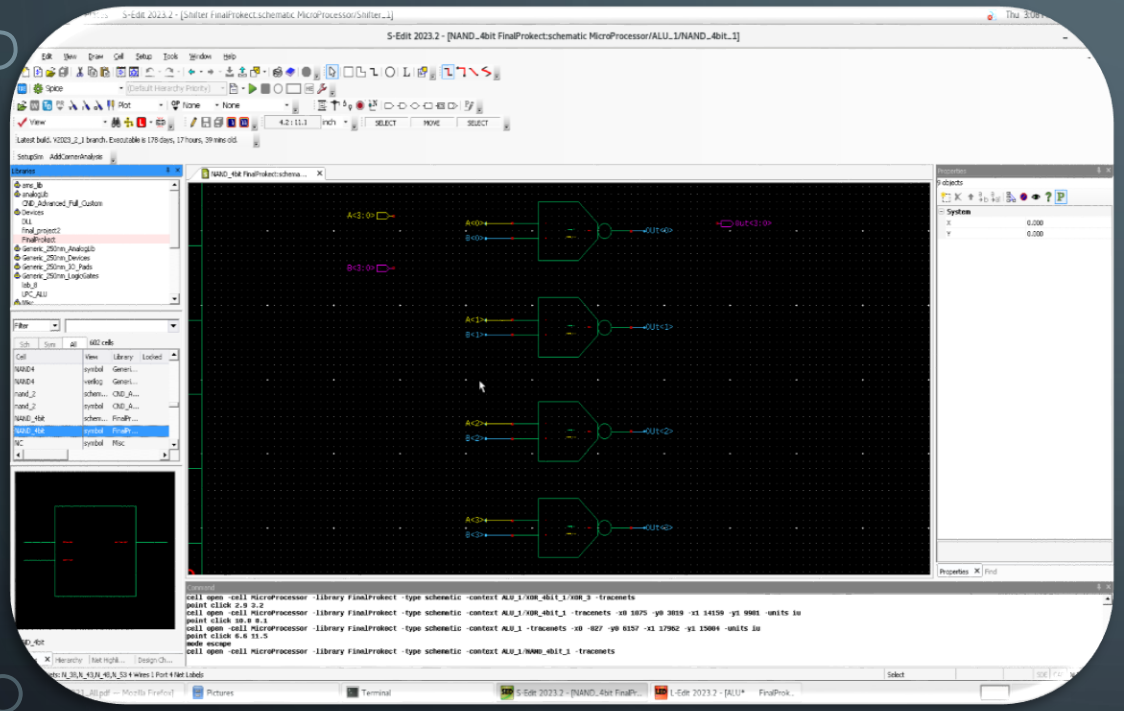

Calibre Interactive - nmLVS v2023.4_17.10 *
XOR2_TG.lvs.report
CELL COMPARISON RESULTS (TOP LEVEL)
CORRECT
LAYOUT CELL NAME: XOR2_TG
SOURCE CELL NAME: XOR2_TG
INITIAL NUMBERS OF OBJECTS
Layout Source Component Type
Ports: 5 5
Nets: 7 7
Instances: 4 4 NM (4 pins)
4 4 MP (4 pins)

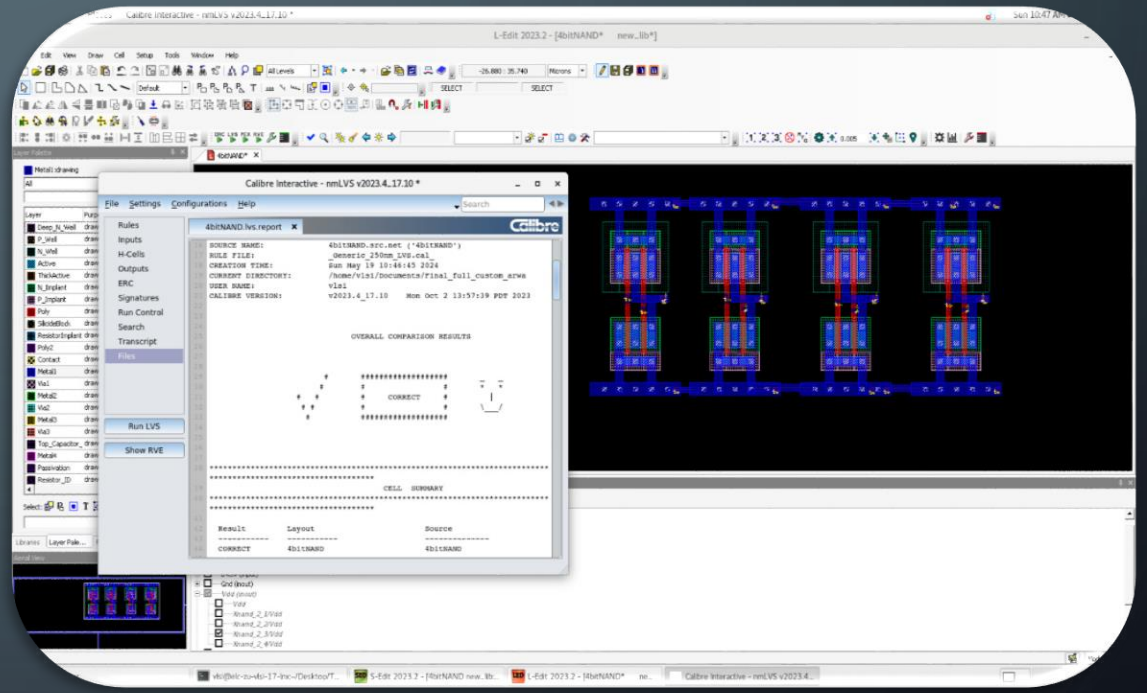
NAND


S-Edit 2023.2 - [nand..2 CND..Advanced..Full..Custom:schematic MicroProcessor/ALU..1/NAND..4bit..1/nand..2..1]
13
14
15
16

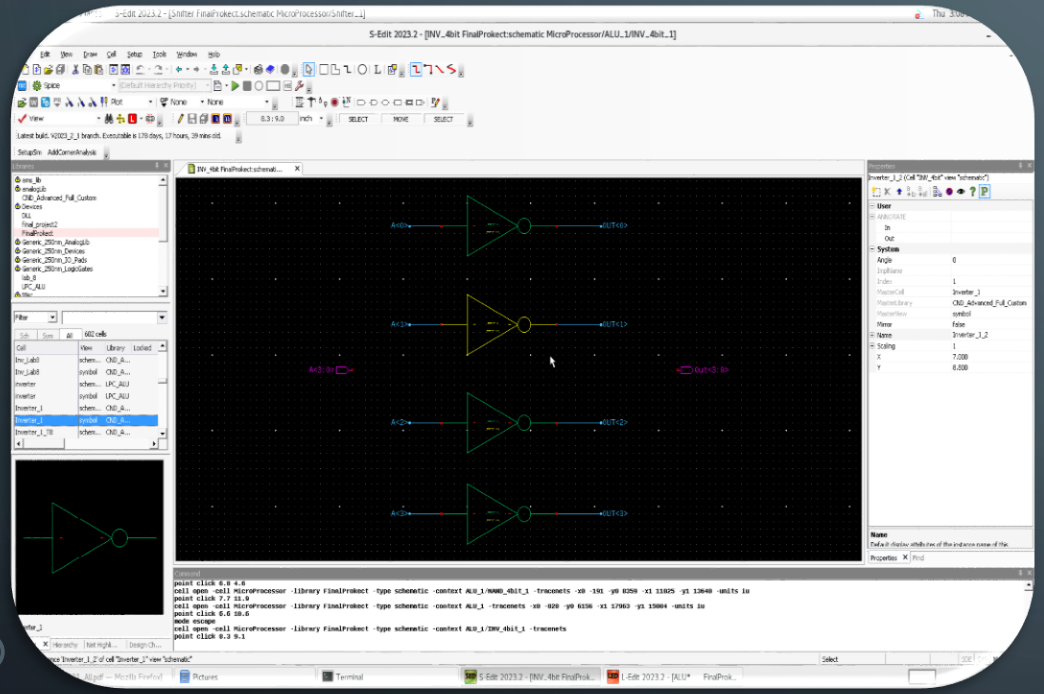

Vdd
A
B
Out
Gnd

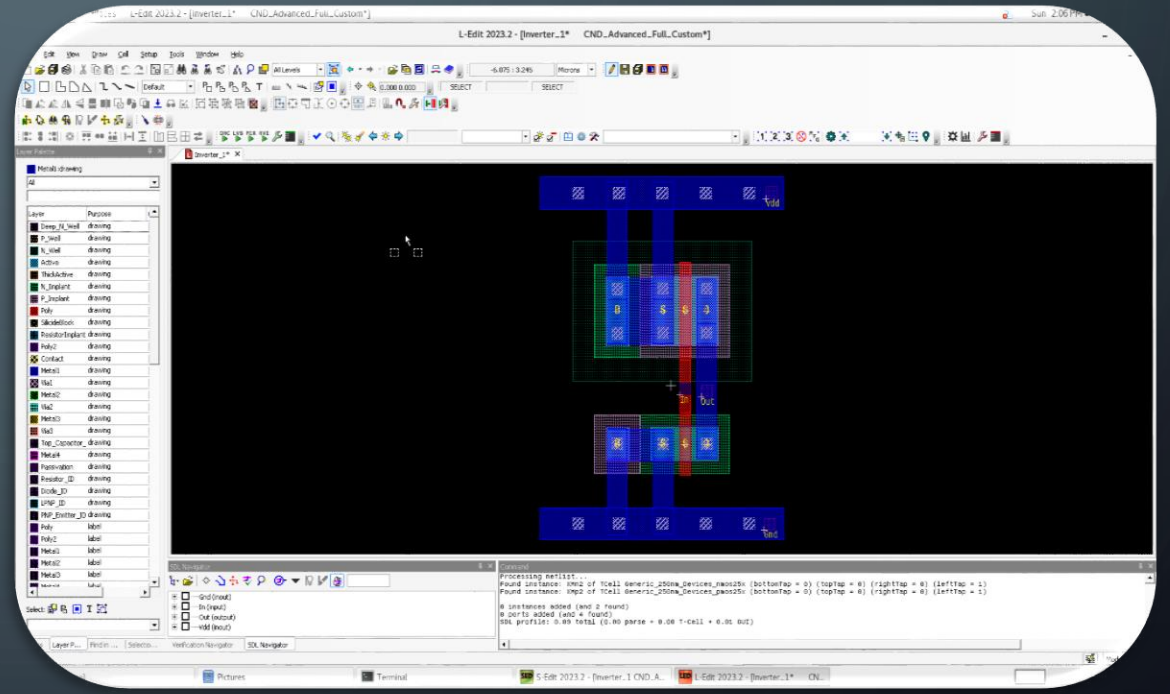
NAND 4 BIT


S-Edit 2023.2 - [NAND_4bit FinalProkect:schematic MicroProcessor/ALU_1/NAND_4bit_1]

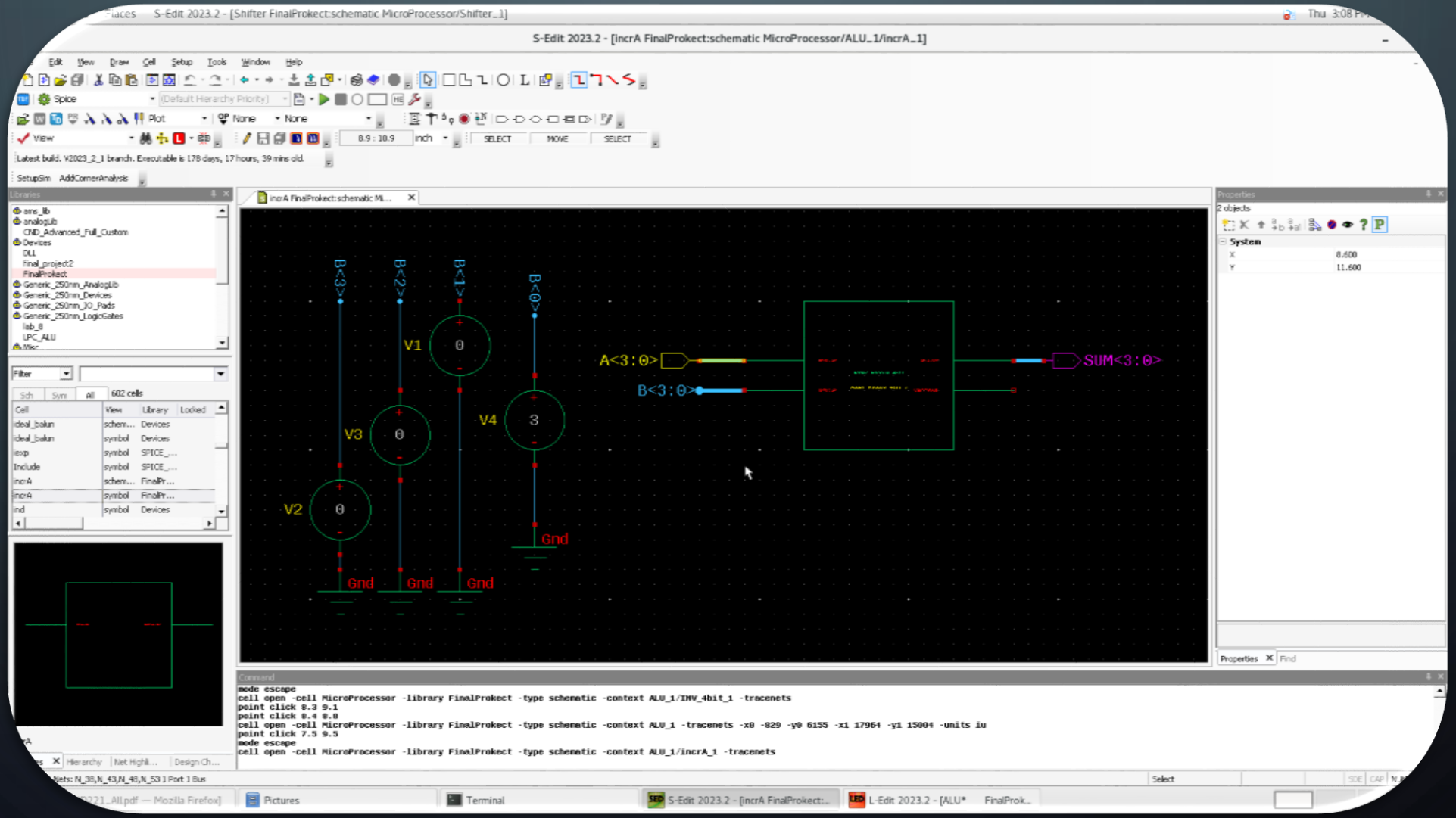

Calibre Interactive - nmLVS v2023.4_17.10 *
File Settings Configurations Help
Rules
Inputs
H-Cells
Outputs
ERC
Signatures
Run Control
Search
Transcript
Run LVS
Show RVE
4bitNAND.lvs.report
OVERALL COMPARISON RESULTS
CORRECT
CELL SUMMARY

INVERTER 4 BIT

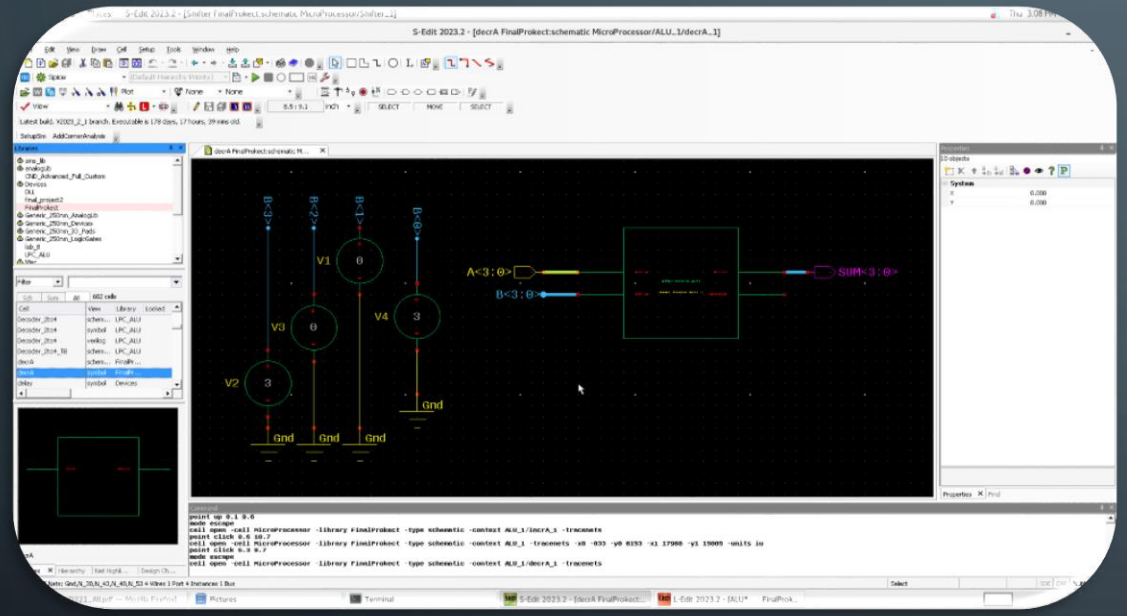


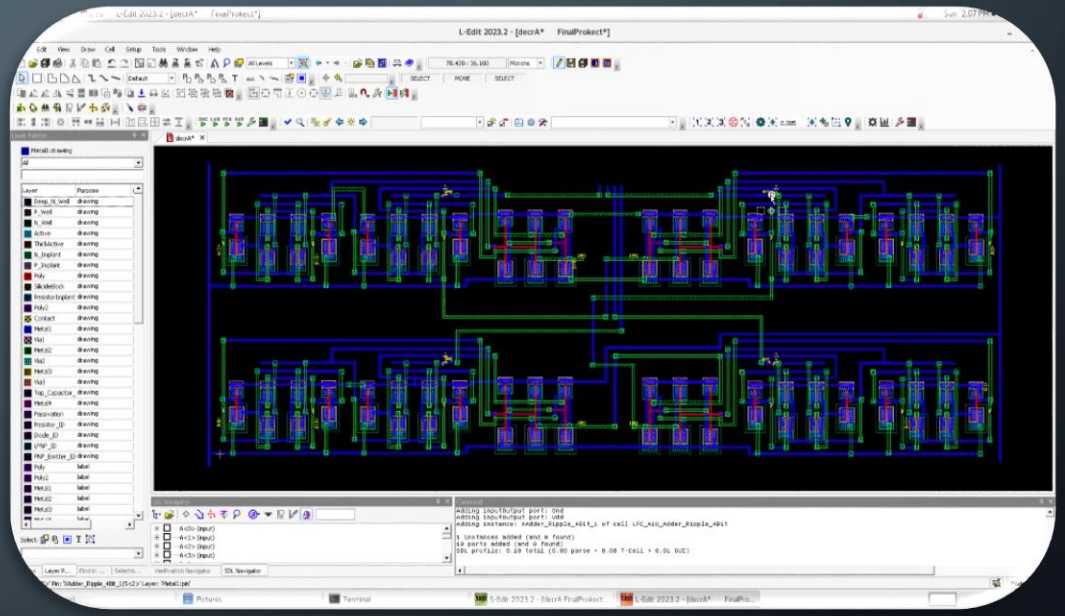


INCR A

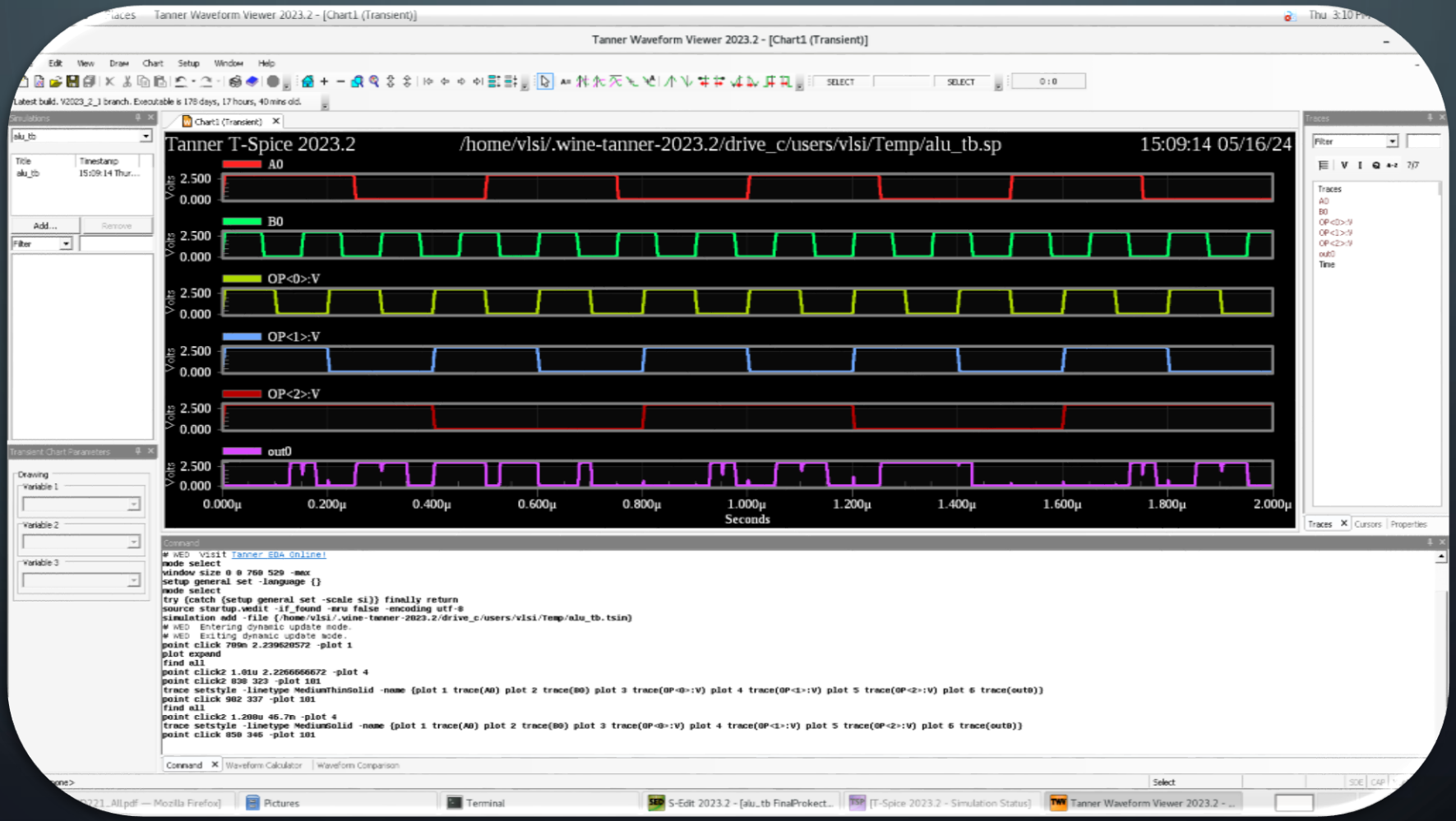

S-Edit 2023.2 - [incrA FinalProkect:schematic MicroProcessor/ALU_1/incrA_1]
A<3:0>
B<3:0>
SUM<3:0>
V1
V2
V3
V4
Gnd
mode escape
cell open -cell MicroProcessor -library FinalProkect -type schematic -context ALU_1/INV_4bit_1 -tracenets
point click 8.3 9.1
point click 8.4 9.8
cell open -cell MicroProcessor -library FinalProkect -type schematic -context ALU_1 -tracenets -x0 -829 -y0 6155 -x1 17964 -y1 15004 -units iu
point click 7.5 9.5
mode escape
cell open -cell MicroProcessor -library FinalProkect -type schematic -context ALU_1/incrA_1 -tracenets

DECR A

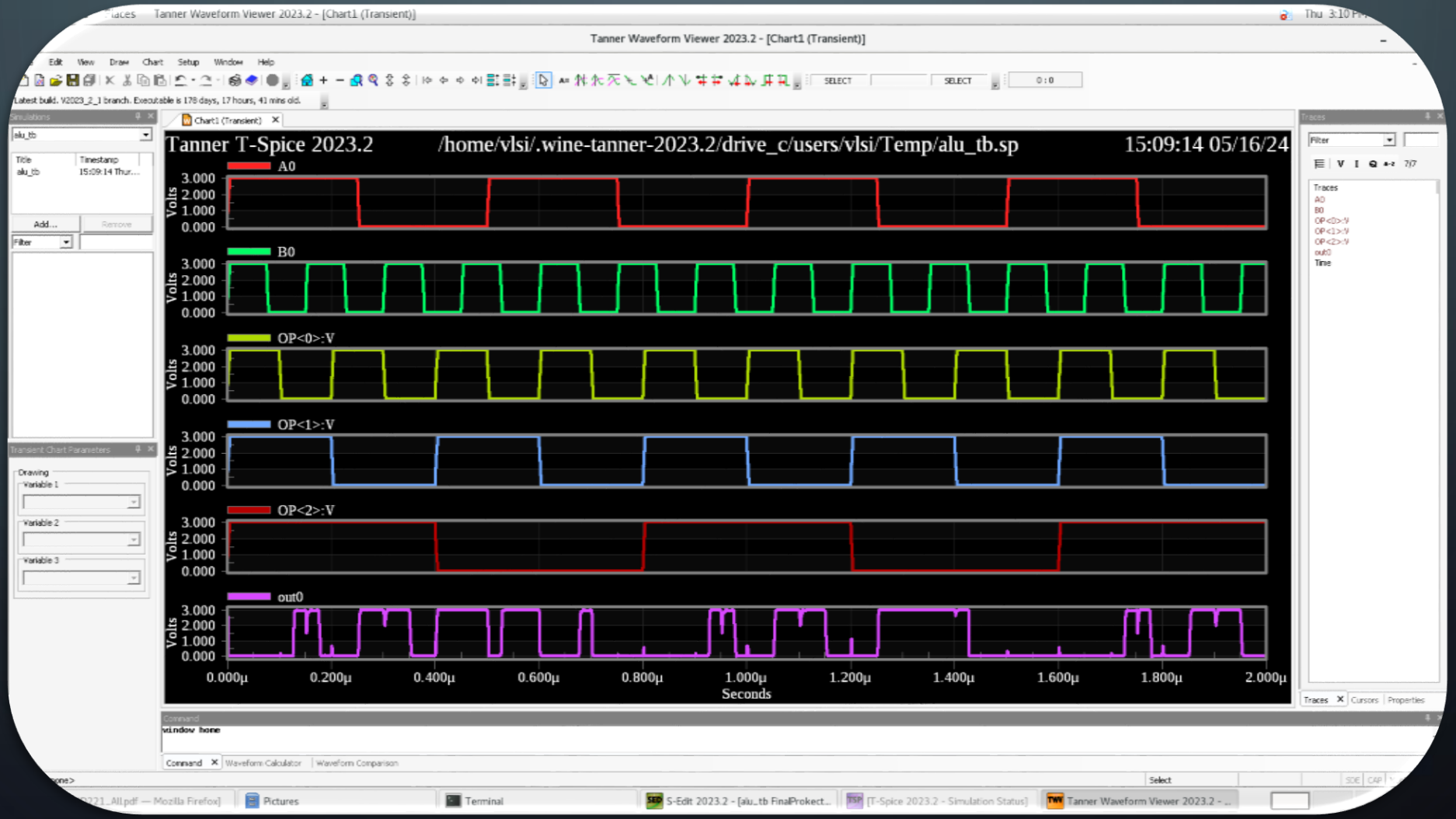

S-Edit 2023.2 - [decrA FinalProkect:schematic MicroProcessor/ALU_1/decrA_1]
V1
V2
V3
V4
Gnd
Gnd
Gnd
Gnd
A<3:0>
B<3:0>
SUM<3:0>


L-Edit 2023.2 - [decrA* FinalProkect*]

CHAR 1


Tanner Waveform Viewer 2023.2 - [Chart1 (Transient)]
Tanner T-Spice 2023.2
/home/vlsi/.wine-tanner-2023.2/drive_c/users/vlsi/Temp/alu_tb.sp
15:09:14 05/16/24
A0
B0
OP<0>:V
OP<1>:V
OP<2>:V
out0
2.500
0.000
Volts
0.000µ
0.200µ
0.400µ
0.600µ
0.800µ
1.000µ
1.200µ
1.400µ
1.600µ
1.800µ
2.000µ
Seconds

CHAR


Tanner Waveform Viewer 2023.2 - [Chart1 (Transient)]
Tanner T-Spice 2023.2
/home/vlsi/.wine-tanner-2023.2/drive_c/users/vlsi/Temp/alu_tb.sp
15:09:14 05/16/24
A0
B0
OP<0>:V
OP<1>:V
OP<2>:V
out0
Volts
3.000
2.000
1.000
0.000
0.000µ
0.200µ
0.400µ
0.600µ
0.800µ
1.000µ
1.200µ
1.400µ
1.600µ
1.800µ
2.000µ
Seconds

FOR MORE INFORMATION, CLICK ON THE GETHUB LINK OR SCAN THE QR

https://github.com/tito642/4-bit-Microprocessor-Design-and-Implementation