Midterm Project

Design of 4 Bit ALU

Fall 2001

By
K.Narayanan
George Washington University
E.C.E Department

Abstract

An attempt to Design digital circuits like the ones used in the ubiquitous computers, microprocessors is the aim of this project. In the Design and simulation of the circuits, CMOS technology was adopted. CMOS technology is built on Field effect transistors which have the following advantages:

- At steady state the power consumption is negligible, facilitating a greater level of integration.
- The design of CMOS circuits is simpler.
- The Input current drawn by a MOSFET is negligible and hence does not load the preceding stage.

Tanner software, Sedit, Ledit was used in simulation of the digital circuits. The project aims at demonstrating the use of fundamental building blocks of Combinational and Sequential Circuits like adder, multiplexer, decoder, flip-flop and register.

1.2 Specification:

The Specification (Refer Annex 1) Identifies the operations of a 4-Bit ALU.The ALU is required to provide support for 4bit Addition, Subtraction and other related operations.

Based on the specification several modules were individually designed and tested before integration.

1.3 Design Objectives:

The Design should conform to the following objectives:

- Correctness of Logic, i.e Addition 2 Numbers say 1+2 should result in 3.
- Aim at Minimising the active components i.e the gates used and hence the overall layout area.
- Must take into consideration the finite time available for any process with regards to the associated gate delays in the chosen path.
- Must be innovative in implementing new means to achieving the end.

1.4 Testing:

The Design must be thoroughly tested to ratify whether the implementation meets the above-mentioned objectives.

Testing was done both at modular level with individual test-vectors, and at the Integrated level when all the modules interact with one another.

The Multi-level tests ensure the robustness of the design and unearths minor defects in the design that was corrected iteratively.

Acknowledgements

Teaching profession is sometimes regarded as an ungrateful, unreciprocated occupation. A teacher shares his or her wisdom with batches of students who might not acknowledge the effort that goes in the process of pedagogy.

On the other hand an enthused, inspired and awed student is what the unrequited teacher is looking for.
I am greatly indebted to the Faculty of ECE122- ***Professor Can. E. Korman***, Teaching assistants **Faisal Mohd Yasin , Ritabrata Roy** and Lab assistants who have enabled the realization of the learning process.

1.0 Project Requirements

2.0 The ALU must be able to Latch in the inputs A, B with the rising edge and perform the operations and output the result with the falling edge of the 33ns clock.

3.0 The ALU should perform operations on the operands based on the input selection bits s0-3 bits, 4 bits corresponding to 2^4 or 16 different operations as listed in the specifications document in Annex 1.

4.0 All operations of the ALU are UNSIGNED 4 bit operations.

5.0 A Carry Bit should be set if the ALU operation results in a Number, 1 more than 15 which is the Maximum unsigned number that can be represented by 4 bits.

6.0 An Overflow bit should be set if the result of an ALU operation does not correspond to the abovementioned number system, i.e. in the Unsigned number system a multiplication of 2 Numbers leading to a result which cannot be represented by 4 bits or a negative result in subtraction of 2 numbers should result in the overflow bits to be set.

7.0 All operations must be repeatable and be consistent with regards to the correctness of the Logic.

2.0 Design procedure

One cannot drill a hole with a screwdriver or loosen a screw with a can opener. One has to have the right tools to do a job correctly. The end is not the justification to the means employed, i.e. one has to use the right tools, building blocks and proper methods to achieve the end and there may be many feasible solutions to the same problem.

The ALU as the name suggests provides for arithmetic and logical operations on the operands. The Basic Arithmetic of addition, subtraction, multiplication et.al and Logical functions like AND, OR, XOR are some of the functions of the ALU.
In the above problem statement of design of an ALU, the 4 bit unsigned addition, subtraction using 2's complement method are the fundamental arithmetic operations using which one can do most of the arithmetic functions.

The problem at hand is to design a 4 Bit ALU. One that performs simple addition subtraction and shifting functions using Logic gates and sequential circuits. Since the 16 different operations have been identified one could begin by designing the individual operations. Once the specifics of the individual operations are done the task at hand is to integrate them into the end objective of having a fully functional 4-bit Unsigned ALU.

2.1 Bottom – Up Approach

In the Bottom Up approach of solving a problem, one identifies the least significant units (i.e. functions which cannot be broken down further functionally) and uses them as building blocks for the complex system.

In the case of ALU these fundamental functional units are ***Adders***, combinational circuits used for addition, subtraction, multiplication and division. ***Registers***, sequential circuits used to store a stream of bits. ***Multiplexers*** and ***Demultiplexers,*** combinational control elements that are used to control the flow of data in the circuit.

2.2 Design Tips:

1. It is good practice not to interconnect wires to form junctions even if we are sure that only one lead into the junction or node drives the node at any given time. Use logic gates for such inter-connects.
2. Use De-Morgans theorem to reduce NOR operations to their equivalent NAND operations for PMOS transistors in series in NOR make the gate area larger than the NMOS transistors in series in NAND.
3. Try simplifying circuits using K-Maps for simplified circuits save a lot of active components and leads to efficient chip realestating.

3. BINARY UNSIGNED 4-Bit Arithmetic:

The Nine Arithmetic operations involving addition and subtraction indicated by the specifications document (Annex 1) use the following inputs of the 4- Bit adder which is implemented using four Full-Adders in tandem with the carry-out of first stage feeding the carry-in of the next stage.

Table 1

Sl no	Opera-tion	Input a0-3	Input b0-3	carry -in	carry-out
1	A + B	A0-3	B0-3	0	**If 1** represents Resulting Numbers lying in 16 to 31 range **If 0** represents Resulting Numbers lying in 0 to 15 range
2	A – B	A0-3	1's complement of B	1	**If 1**, the Resulting Number is +ve and the bits represent the actual Number. **If 0,** the Resulting Number is -ve and hence in the case of *Unsigned* ALU the **Overflow (V)** bit is set. The bits is a 2's complement representation.
3	A++	A0-3	B0-3 = 0000	1	As in Case 1.
4	B++	A0-3= 0000	B0-3	1	As in Case 1.
5	A--	A0-3	B0-3 = 1110** 1's complement of 1	1	As in Case 2.
6	B--	A0-3= 1110** 1's complement of 1	B0-3	1	As in Case 2.
7	Neg A	A0-3= ** 1's complement of a0-3	B0-3 = 0000	1	As in Case 2.
8	Neg B	a0-3 = 0000	b0-3= ** 1's complement of b0-3	1	As in Case 2.
9	Max (a,b)	As in Case 2	As in Case 2	1	As in Case 2 * Maximum Number is determined based on whether Carry Bit **C** is set or reset.

4 Modules and Input Output Relationships

Modules and their relationship may be classified into **Fundamental** and **Derived Modules** to develop a hierarchical order to study and evaluate them.

5.1 Fundamental Modules:

The Basic Modules to Be Used for the ALU are:

1. Half Adder:

The Symbol and Schematic representation of the Half adder are given in Fig 1 and Fig 2. The half adder is used as the Basic unit to create the 1 Bit Full Adder.

> **NOTE:** *All the Modules individual and integrated test cases are included as transient result charts in* **Annex 2**.

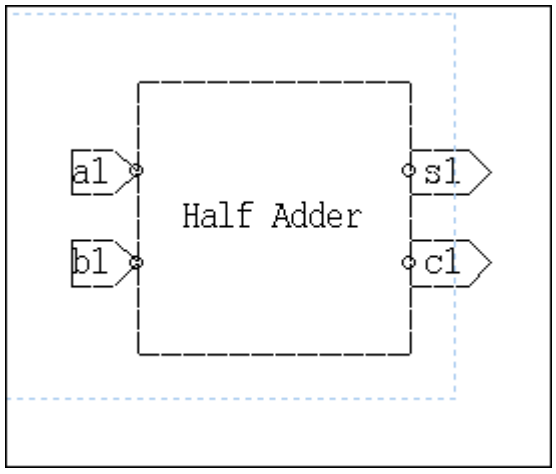


Figure 1 - Half Adder Symbol

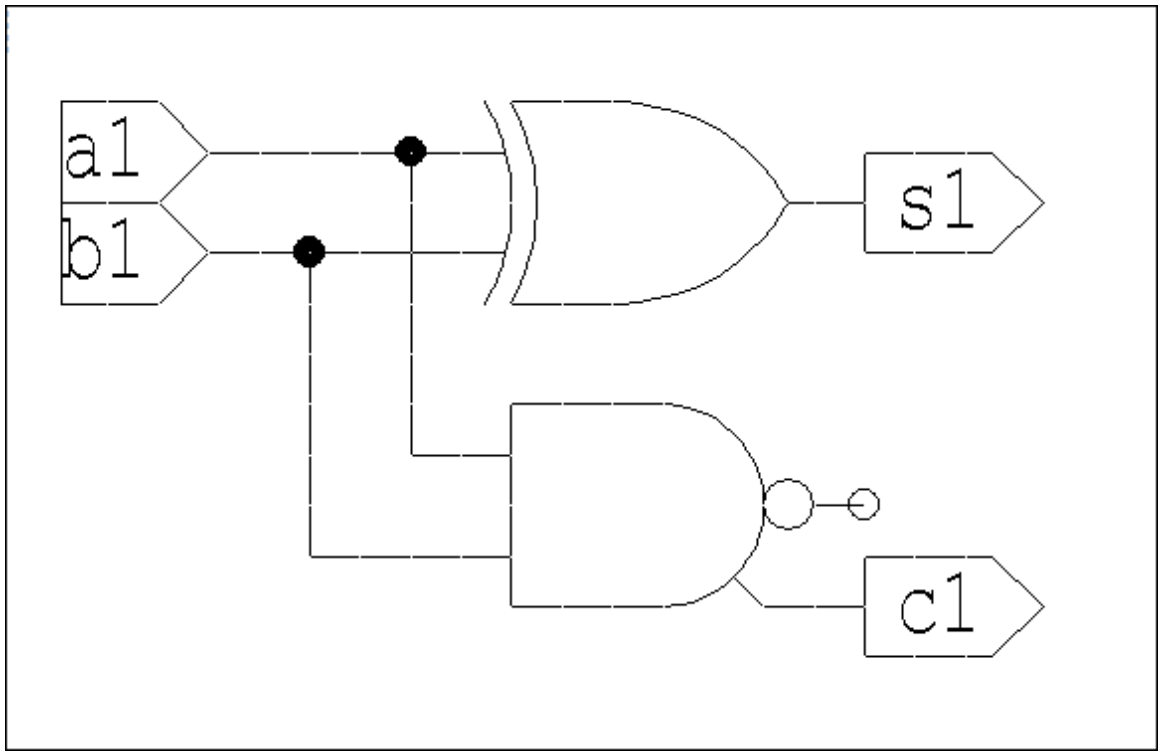


Figure 2 - Half Adder Schematic

2 Full Adder:

The Full Adder represents the addition process in whole with the presence of the 3rd input the Carry In from the previous addition. The symbol and schematic of a Full adder is illustrated in Fig 3 and Fig 4.

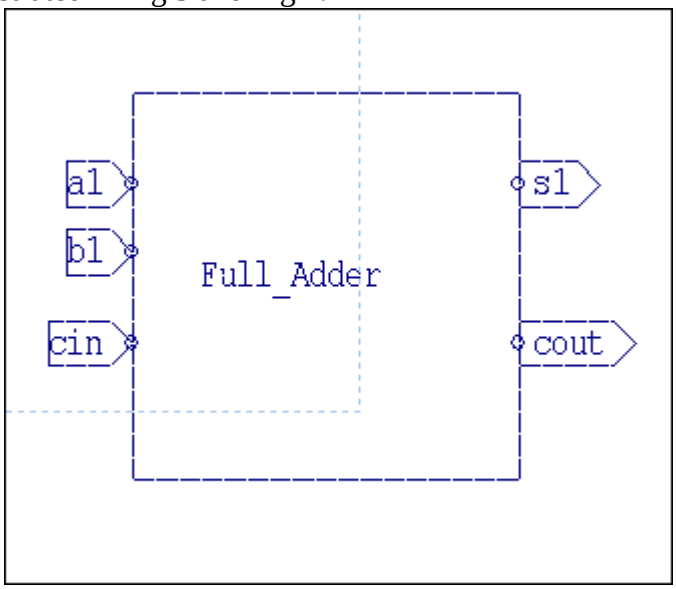


Figure 3 - Full Adder Symbol

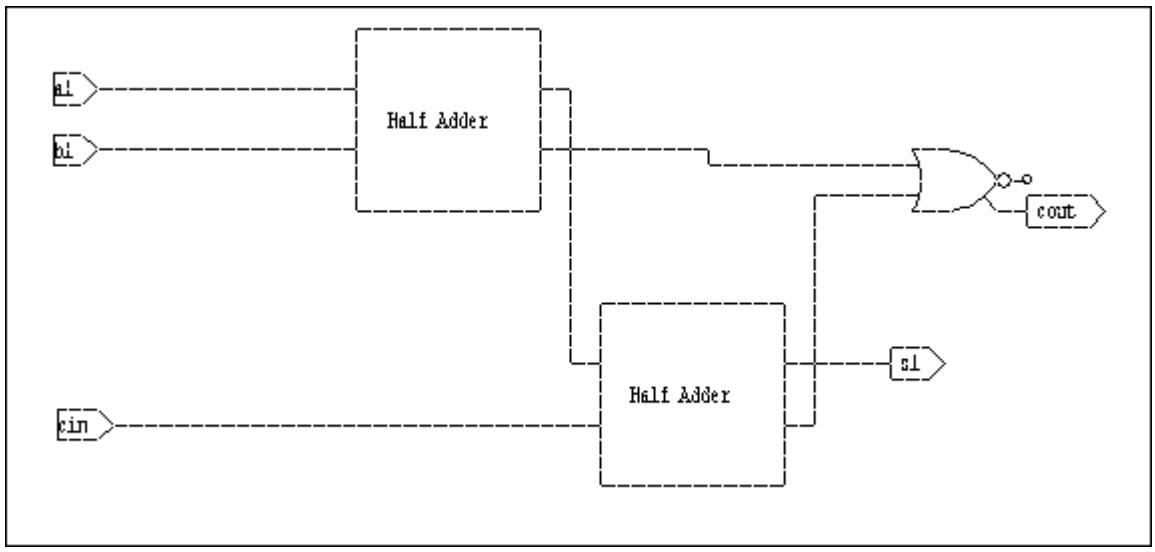


Figure 4 – Full Adder Schematic

3 Multiplexer:

The Multiplexer is used to choose one of the many given inputs at a time for an operation. Depending on the number of inputs to choose from we can implement a 2x1,4x1,8x1,16x1 multiplexer that is used to select amongst 2,4,8,16 inputs respectively. A 2^n input Mux would require a n bit select logic. Figures 5 thro 8 Indicate the methods of creating 4 and 8 input based Mux from the fundamental 2x1 Mux available in the Tanner Library.

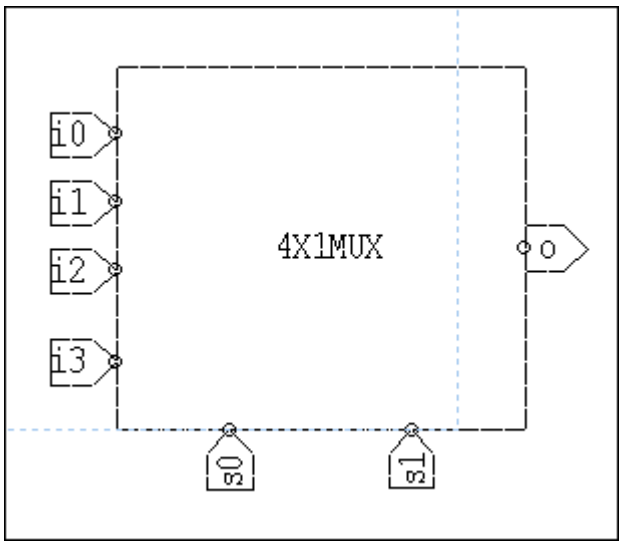


Figure 5 – 4x1 Multiplexer Symbol

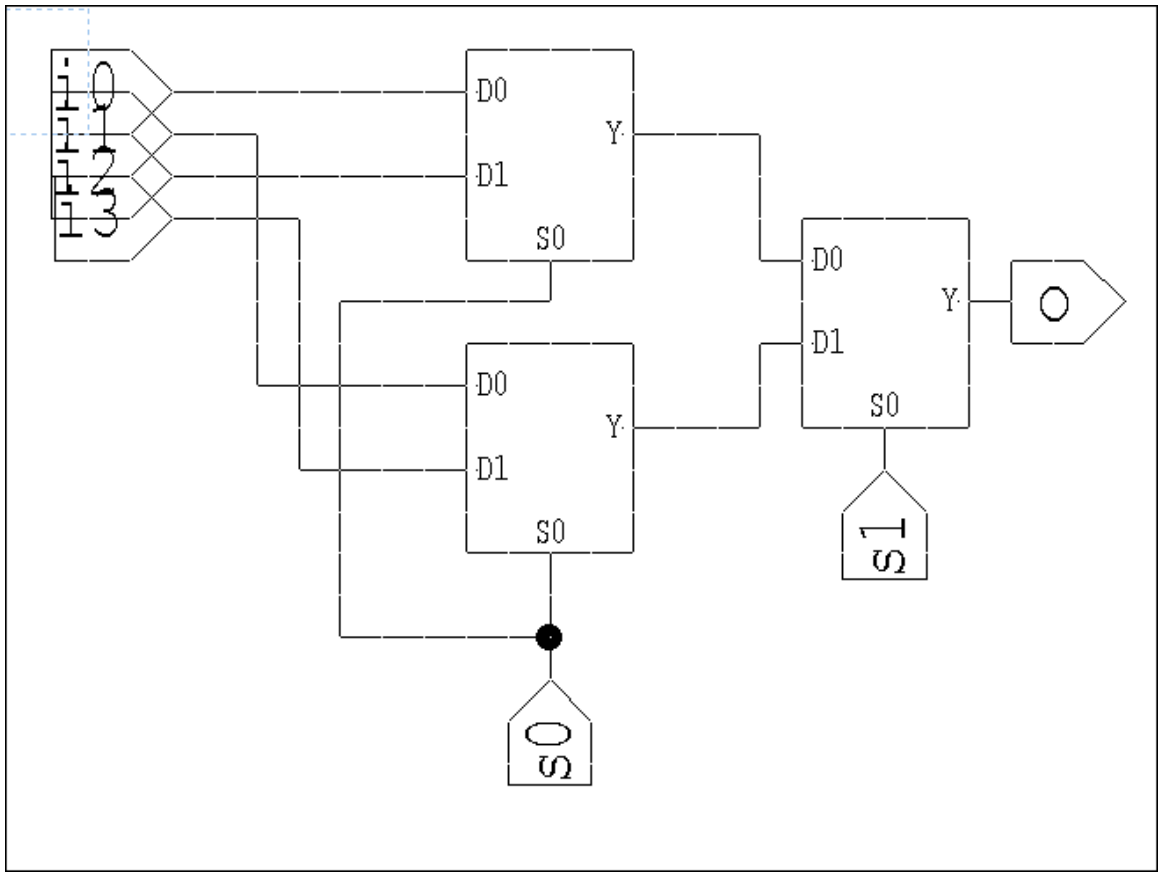


Figure 6 - 4x1 Multiplexer Schematic

NOTE: *The ordering of the input selection based on the s0, s1 bits in the 4x1 Mux by combining two 2x1 Mux's. The inputs 1,3,2,4 are selected based on (s0,s1) values corresponding to (0,0), (0,1), (1,0), (1,1).*

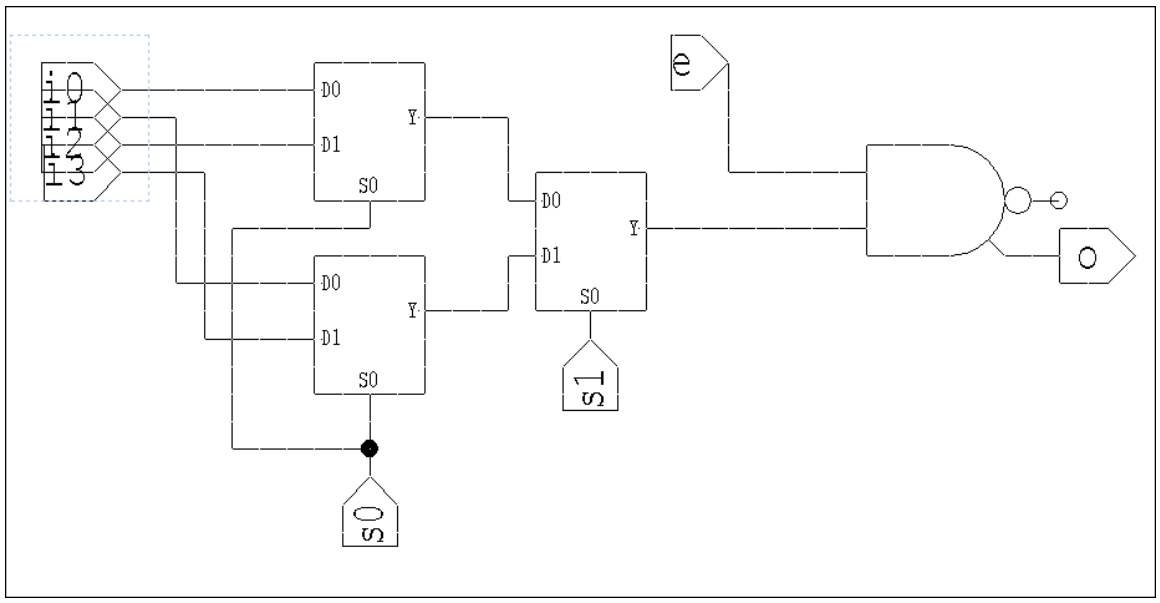


Figure 7- 4x1 Mux With Chip Select

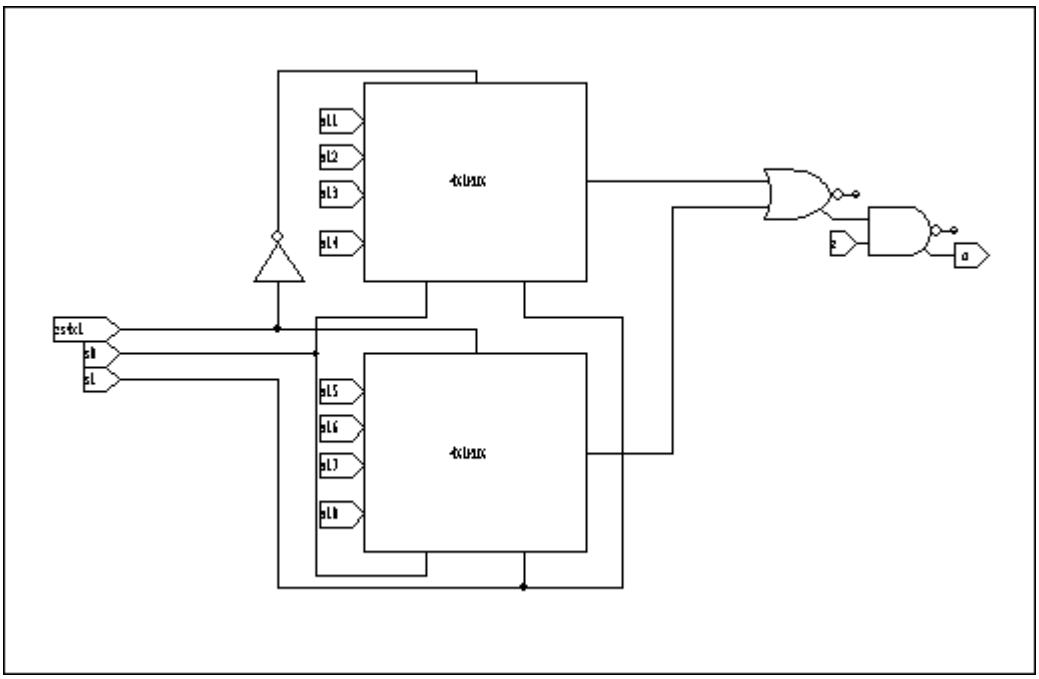


Figure 8 – 8x1 Mux Schematic

NOTE: *The Use of Chip select logic in creation of an 8x1 from two 4x1 and 16x1 from two 8x1 Mux's. The Chip is selected with a High logic and hence an inverted chip-select is used to select the first bank while the high CS selects the second bank thus keeping the select logic contiguous.*

4 Decoder or Demultiplexer:

Is a Natural complement of the Multiplexer. Here 2^n Outputs are created from an n input Address. Decoders are used commonly to generate the Chip Select or Enable signal for a module based on the n inputs. Every address bit can take a binary value of 0 or 1 thus enabling us to address 2^n output ports to which the input may be selectively directed. Fig 9 and 10 gives a simple 1x2 decoder while Fig 15 and 16 represents a 4x16-modified decoder that is used in the project to generate the Chip Selects.

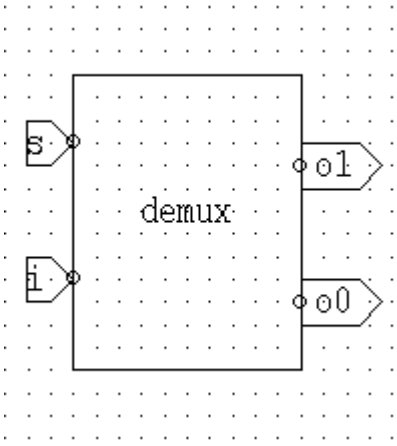


Figure 9 - 1x2 Demux Symbol

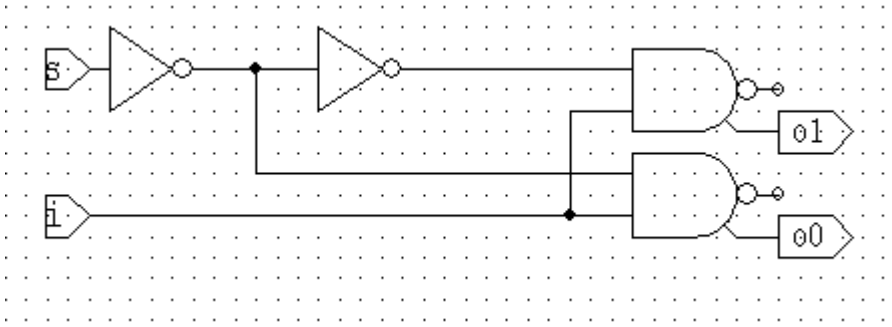


Figure 10 – 1x2 Demux Schematic

5 Logical Operation Modules:

The Logical Operations involves the Bit-wise AND, OR and XOR operations. These Logical operations are implemented by using the available logic gates to perform the bit wise operation. Each Module has its own chip select logic that isolates it from the circuit when not enabled.

1 Four Bit OR Module

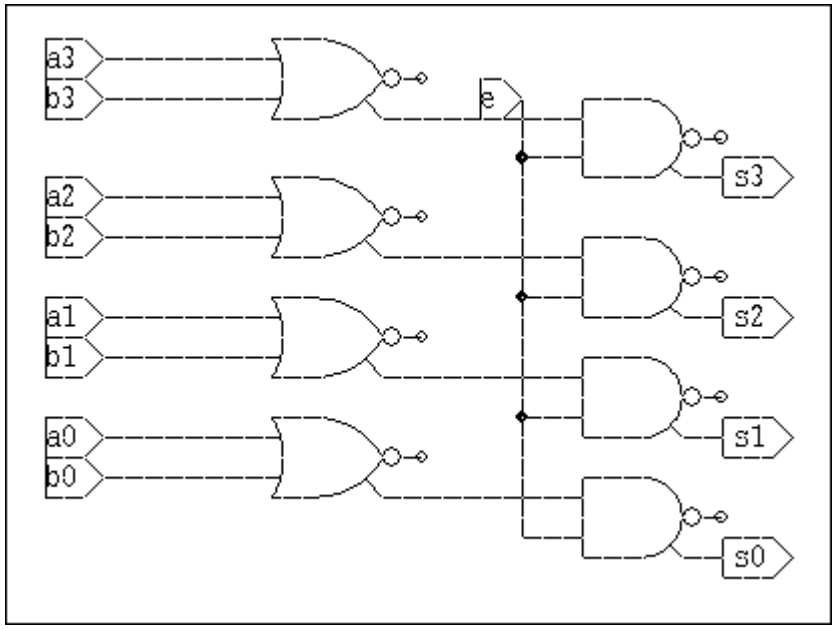


Figure 11 - 4 Bit OR Module Schematic

2 Four Bit AND Module

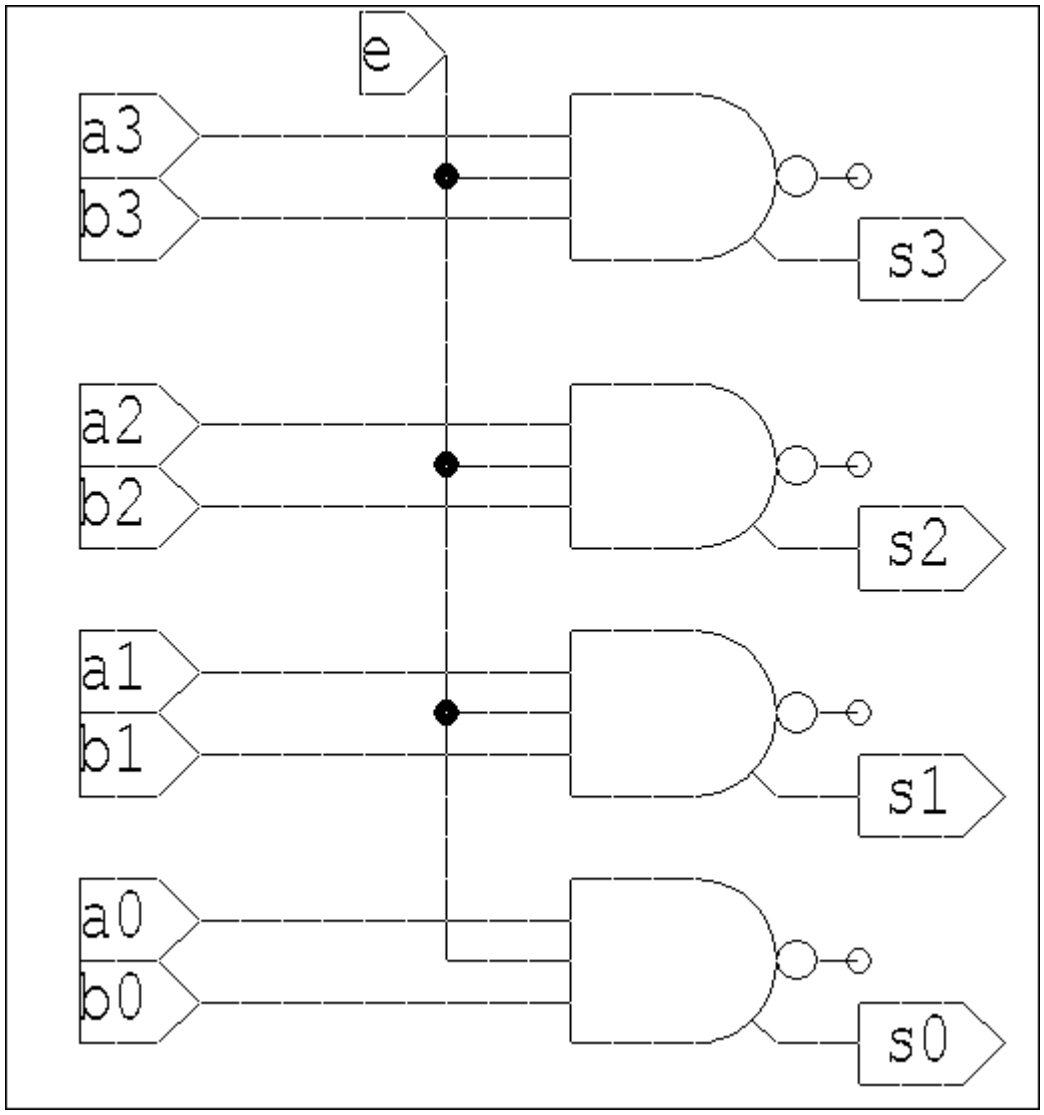


Figure 12 - 4 Bit AND Module Schematic

3 Four Bit XOR Module

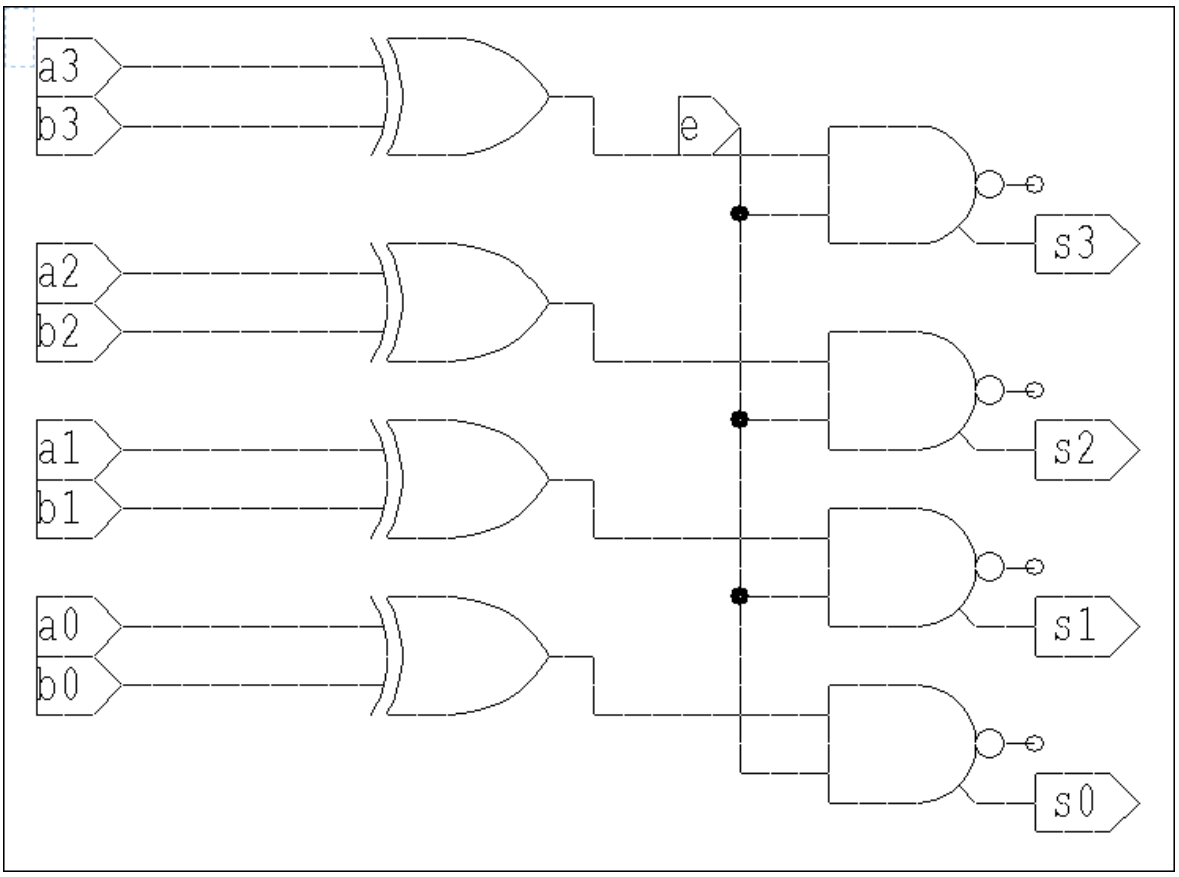


Figure 13 - 4 Bit XOR Module Schematic

5.2 Derived Modules:

1 Four Bit Full Adder:

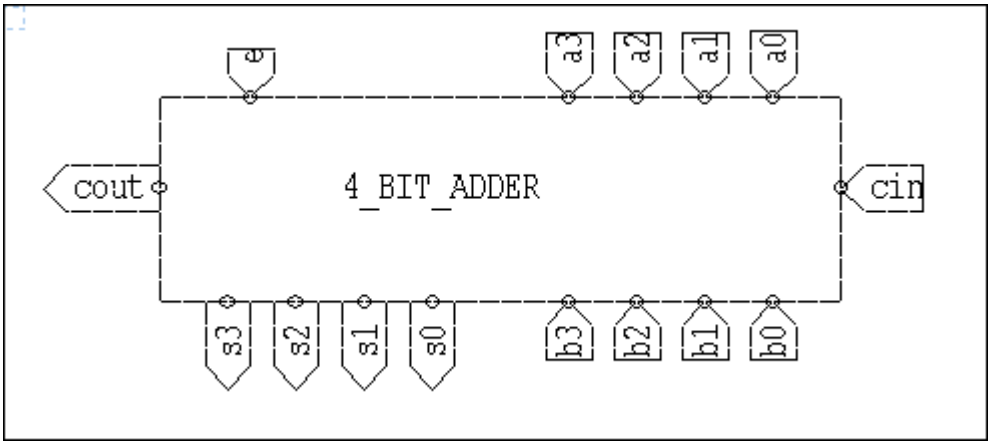


Figure 14 – Symbol of 4 Bit Adder - Subtractor

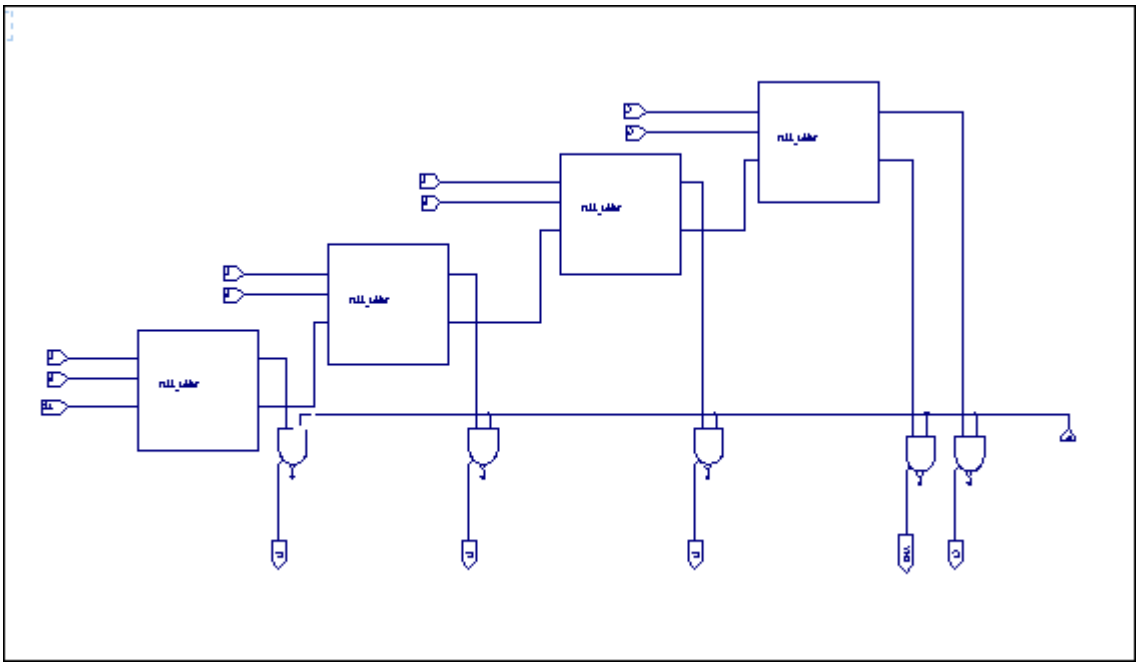


Figure 15 – Schematic of 4 Bit Adder-Subtractor

2 16x1 Mux:

The 16x1 Mux is built on the same lines as a 8x1 Mux using the Chip Select as the additional input in combining the Mux's Input. The 16 different operations requires 8 such modules to specify the 16 combinations of (a0-3, b0-3) pairs. Table 1. Lists the different inputs for different combinations of A and B for specific ALU operations.

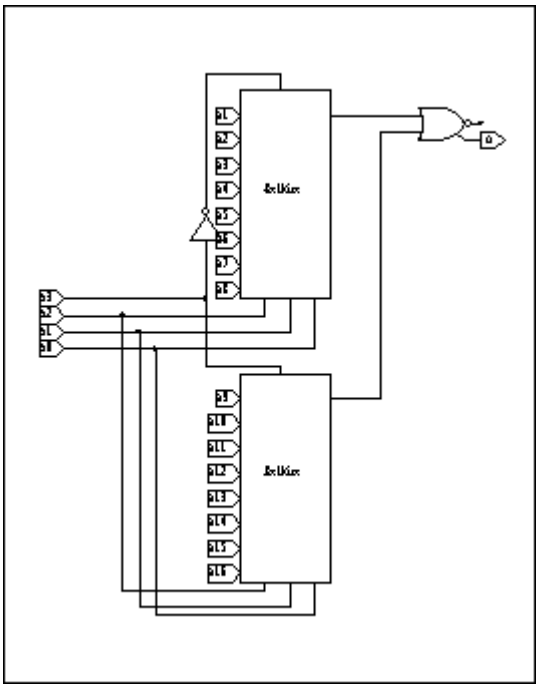


Figure 16 – Symbol of 16x1 Mux

3 16 Input Selector:

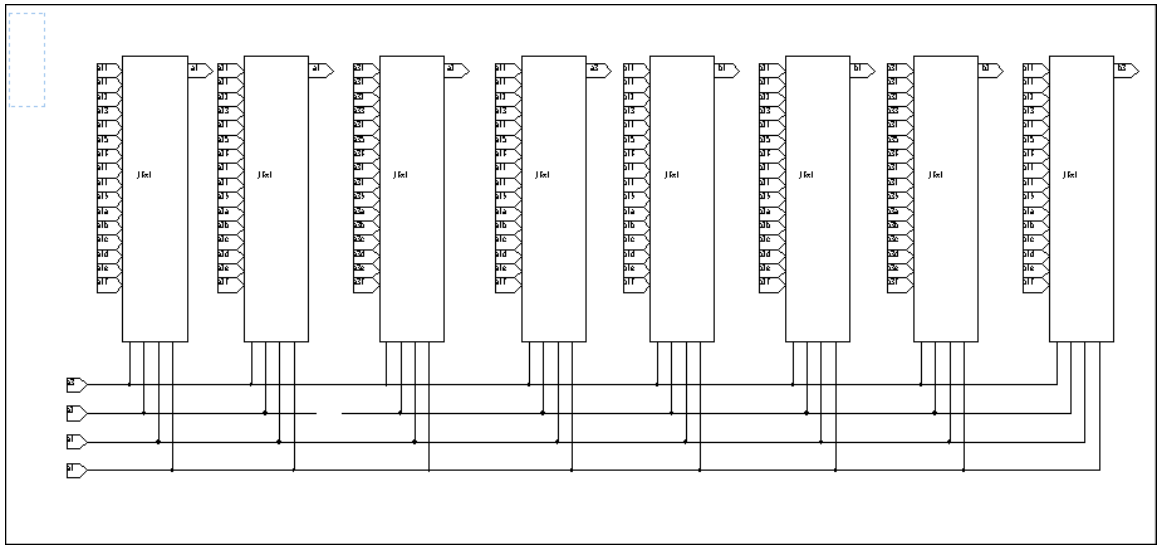


Figure 17 – Symbol of the 16 Input Selector

*** 16 different combinations of (a0-3,b0-3) chosen using the above Input selector Module (Refer Table 1).**

4 4x16 Decoder Chip Select Module

The decoder is used to generate the Chip Selects for the different operations.

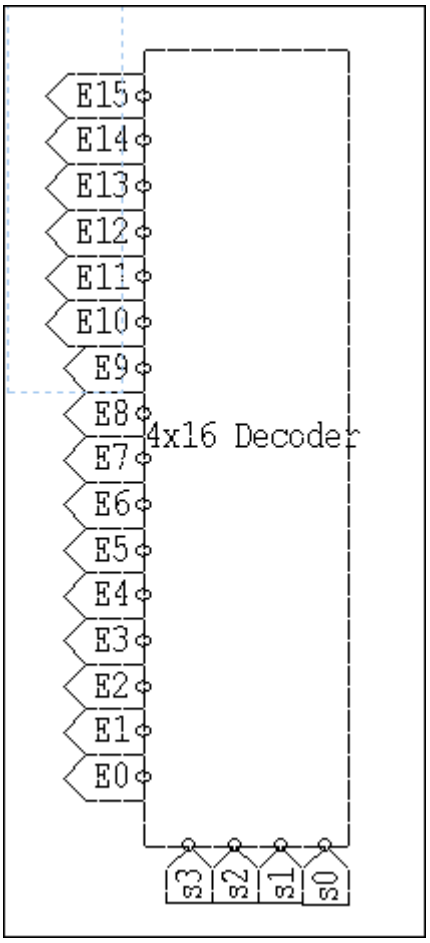


Figure 18 – Symbol of 4x16 Decoder

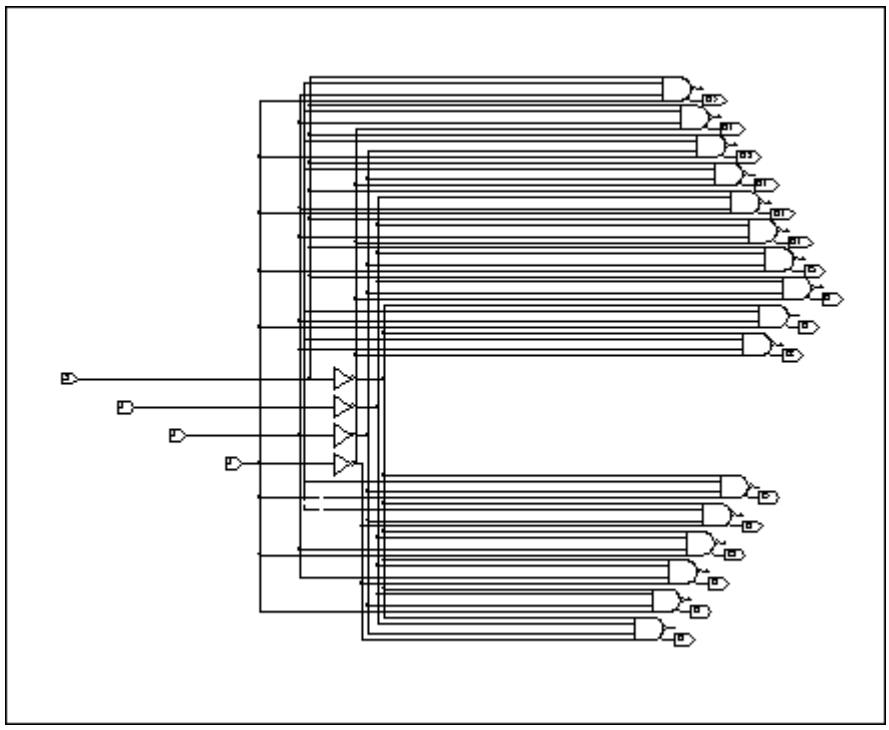


Figure 19 – Schematic of 4x16 Decoder

5 ALU Add On Modules:

The ALU add on modules implements the conversion of the available inputs to a form suitable for a particular operation.
NOTE: All modules have an enable input which has to be asserted for the module to function. Only those modules with correct Chip select generated by the decoder based on the s0-3 bits would then drive the **Result Bus**, which may be shared by all the modules.

1 Incr A add on Module

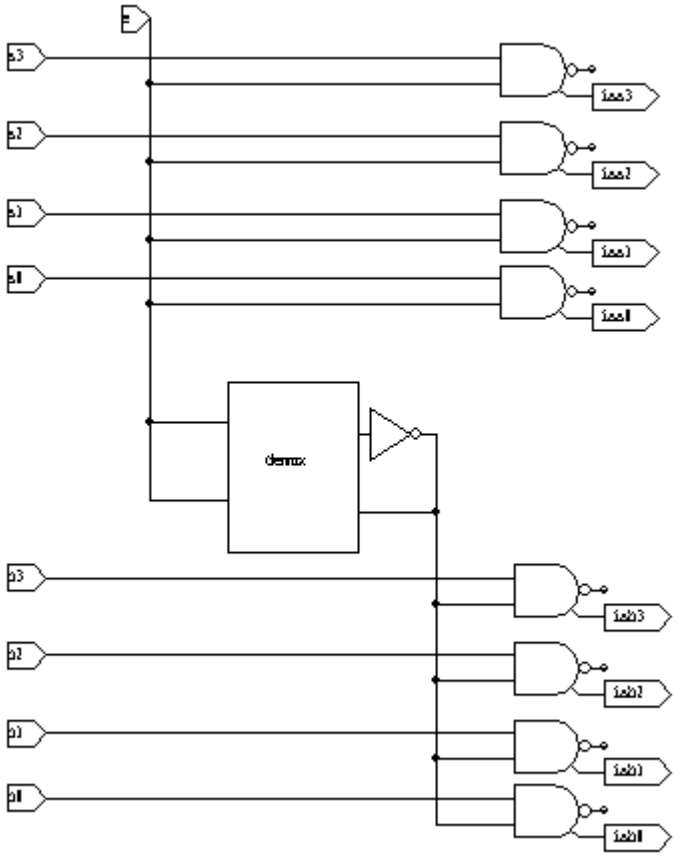


Figure 20 – INCR A Add-on Module Schematic

2 Decr A Add On Module:

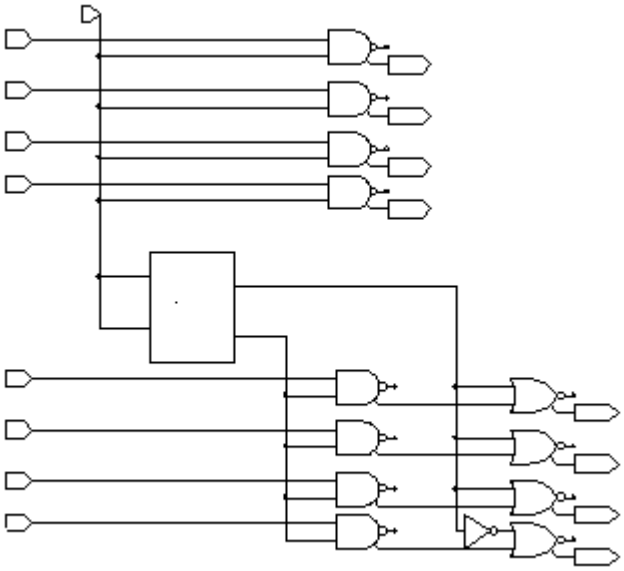


Figure 21- DECR A Add-on Module Schematic

3 Neg A add On Module:

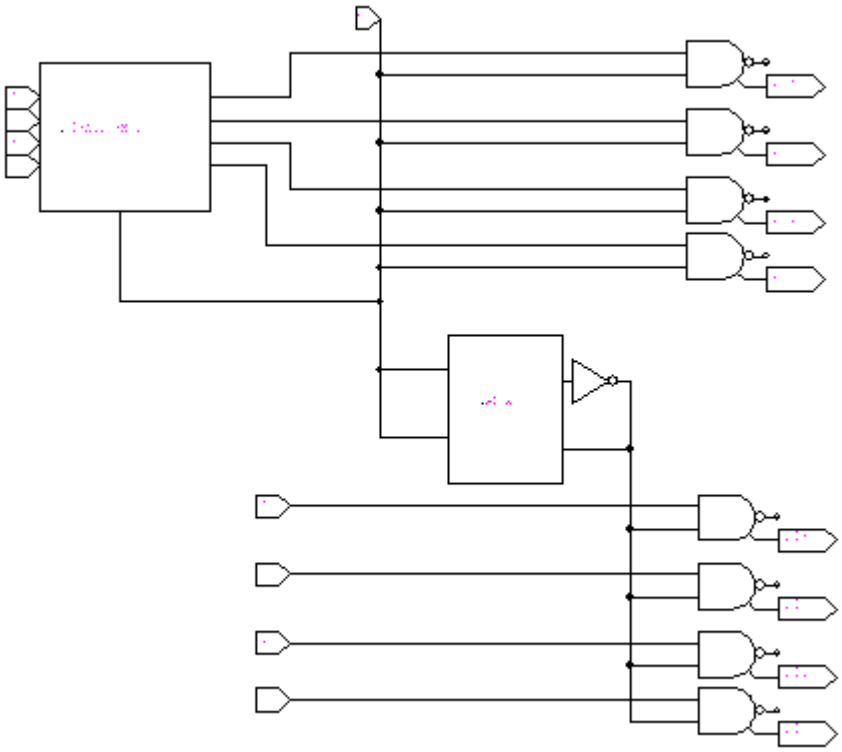


Figure 22 - NEG A Add-on Module Schematic

4 Complementer Module:

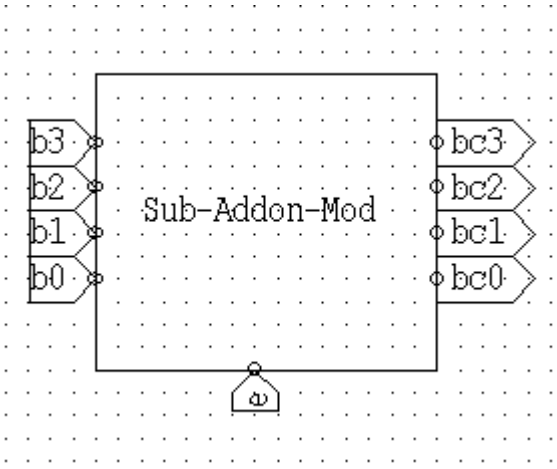


Figure 23 - Subtraction Add-on Module Symbol

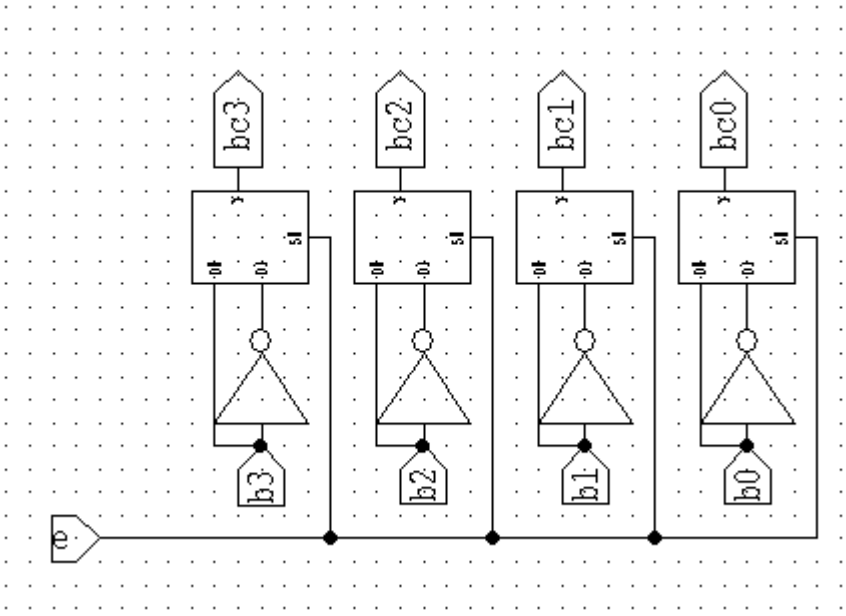


Figure 24 - Subtraction Add-on Module Schematic

6 Shifter Register – Bidirectional shifter with Parallel Load

Shifter register is constructed out of D-Flip Flops. The Data in the D input line is sampled either using the clock or a Positive Gate Trigger (PGT) or a Negative Gate Trigger (NGT). The trigger circuits are built to exploit the inherent gate delays during the High to Low transition or the Low to high transitions to generate the trigger pulse used by the flip-flop to sample the Data Line.

Thus by using Triggering circuits one can do more than one operation in a clock cycle like a read and shift operation in just one cycle.

The basics concept behind a Right Circular Shift and a Left Circular Shift lies simply in the way the Data line of one flip-flop is connected to the Output line of another flip-flop. The simplest Right and Left circular shifters are shown in Fig 22 and Fig 23.

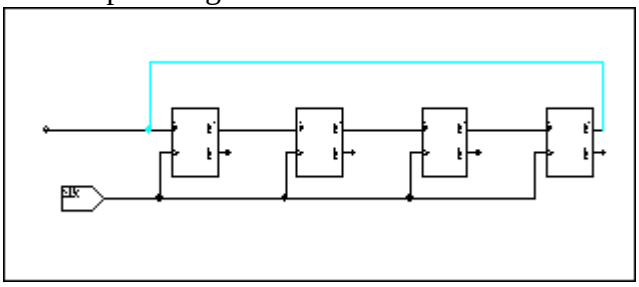


Figure 25 - Right Circular Shift Interconnection

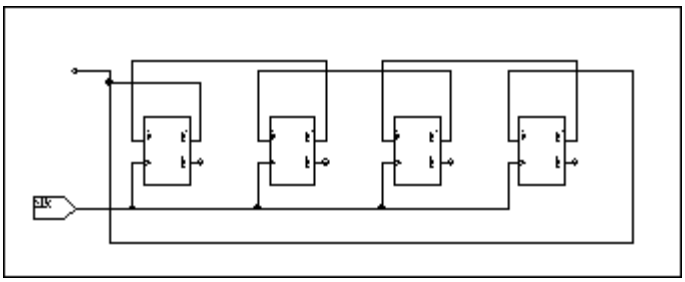


Figure 26 - Left Circular Shift Interconnection

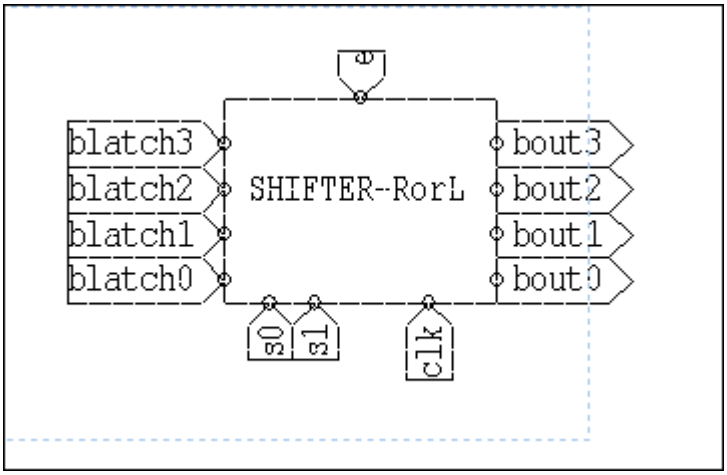


Figure 27 - Bi-Directional Shifter with Parallel Load -Symbol

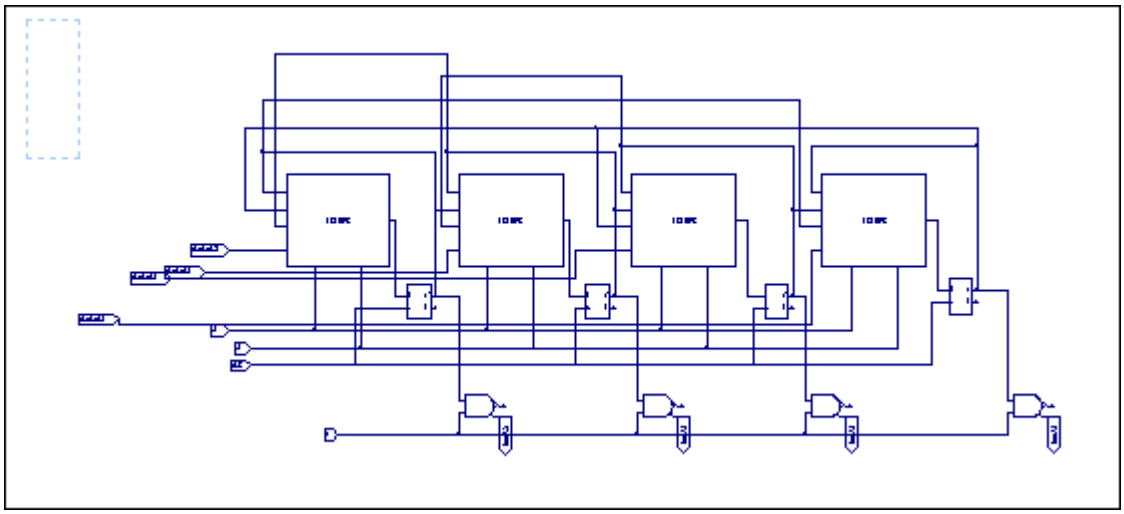


Figure 28 - Bi-Directional Shifter with Parallel Load -Schematic

Is a Bidirectional Shifter that functions as per the interconnections made by choosing the appropriate values to the selector of the 4 x1 Mux if the D and Q are connected as shown in Fig 25 then the above circuit acts like a Right shifter and when connected as shown in Fig26 the circuit acts like a left shifter.

Table 2 indicates the selector values and the behavior of the bi-directional shifter.

Table 2

S0	S1	
0	**0**	**No Change for Every Trigger**
0	**1**	**Right-Circular Shifter**
1	**0**	**Left-Circular Shifter**
1	**1**	**Load To Register**

7 Integrated Circuit

The Individual Modules after being tested were made into symbols to be instantiated into the overall circuit that combines all the operations of the ALU

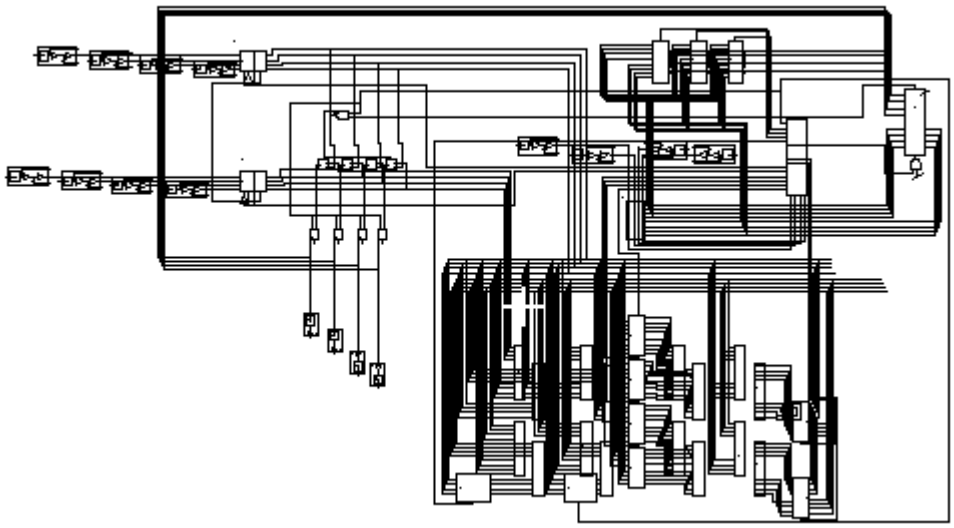


Figure 29 - Integration of All Modules

6 Limitations,Enhancements and Conclusions

- The Circuit Can be simplified to reduce the number of active devices, the current simulation uses around 3200 MOSFETs .

- The Reduction of the Number of Components would also decrease the gate delays involved and bring down the time of simulation. Currently every simulation takes around 100-250 seconds.

- The Layout could not be generated for it was always reporting an application error. The circuit with PadIn and PadOuts is also enclosed.

- The Triggering Circuit Could be incorporated to bring in a synchronization in the circuit. The Computations then would have to be over between successive triggers.

- The simulation using SEdit and Tanner has given an insight to the basic functioning of the building blocks of Combinational and Sequential circuits. The transfer characteristics, gate-delays and functionality of the basic digital elements were understood and an attempt was made to design a 4 bit ALU .

Annex.1

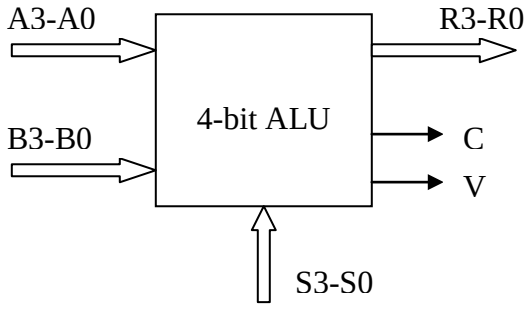
THE GEORGE WASHINGTON UNIVERSITY
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
ECE 122-FALL 2001
MIDTERM PROJECT

Due Date: October 26, 2001

You are a starting engineer in a new firm called ***RYK*Tech Corporation***. The firm is in the business of designing integrated circuits (IC) for various consumer products such as, low power hand held devices, PDA's, etc. Your first task is to design a 4-bit arithmetic logic unit (ALU) core which will be part of the IC to control a new color touch pad display. In order to make the hand held device compete head-on with larger competitors, such as, Palm and HP, the management is under heavy pressure to add as much functionality as possible to the display control chip. Therefore, the primary emphasis at this stage will be on a working ALU design with minimum core layout area. If successful, your core ALU layout design will be incorporated into other designs of the firm, as well. So it is also important to document your design in a professional manner.

4-bit ALU Specifications:

Using Tanner tools, design, simulate, layout and verify the operation of an ALU integrated circuit shown in the following block diagram:



This circuit accepts two 4-bit inputs, A3 to A0 and B3 to B0, and generates a 4-bit output R3 to R0. The relationship between the input variables and the output variables is defined by the 4-bit control input S3 to S0. The network also generates two status signals to indicate an arithmetic carry or borrow from the most significant bit C and arithmetic overflow V. The table below defines the behavior of this network based on the positive-logic assumption.

S3	S2	S1	S0	R	Comments	C	V
0	0	0	0	A		0	0
0	0	0	1	B		0	0
0	0	1	0	A plus B	Add A and B	0/1*	0/1
0	0	1	1	A minus B	Subtract B from A**	0/1	0/1
0	1	0	0	Right A	Right circular shift of A by one bit	0	0
0	1	0	1	Left B	Left circular shift of B by one bit	0	0
0	1	1	0	0 minus A	Negate A**	0	0
0	1	1	1	0 minus B	Negate B**	0	0
1	0	0	0	A plus 1	Increment A	0/1	0/1
1	0	0	1	B plus 1	Increment B	0/1	0/1
1	0	1	0	A minus 1	Decrement A	0/1	0/1
1	0	1	1	B minus 1	Decrement B	0/1	0/1
1	1	0	0	A AND B	AND A with B ***	0	0
1	1	0	1	A OR B	OR A with B ***	0	0
1	1	1	0	A XOR B	XOR A with B ***	0	0
1	1	1	1	Max A, B	Select maximum	0	0

* 0/1 means the value depends on the result.
** Two's complement
*** Bit by bit

Other Design Rules:

1. The design must be able to operate at a 30 MHz clock and 5V-power source. At this stage you are not responsible for the design of the clock.

2. The design documentation is an essential part of the project. No design is acceptable unless accompanied by a detailed professional engineering documentation.

3. Each engineer will make a brief 15 minute professional presentation to highlight the important features of their design.

4. The code S3-S0 will be assigned to each engineer individually. (The code given in the above table is just an example.)

5. All designs will employ the SCMOS logic gate library that is in TannerLB.

6. All layouts will employ the Hewlett Packard 0.5μm n-well technology.

Design Notes:
There are 16 rows corresponding to 16 combinations of the 4-bit control input. Using the hexadecimal representation for the control variable S, we can describe the behavior of the arithmetic logic unit in words. For S=0, the output R is set equal to A, and the carry and overflow signals will be set to 0. For S=1, the output R is set equal to B; again C and V are set to 0. Arithmetic addition and subtraction operations are defined by S=2 and S=3, respectively. The output R represents the sum or difference of the two 4-bit input signals, and C and V are set to 0 or 1, depending on the outcome of the operation.
The next four operations, corresponding to S=4 to S=7, produce an output that is either the shift or negative (two's complement representation) of either 4-bit input signal. For these four operations the arithmetic overflow and carry signals C and V are set to 0. With the next four operations, corresponding to S8 to S11, the output R is generated by incrementing or decrementing one of the 4-bit input signals. Since these represent arithmetic operations, the status signals C and V are set based on the result. Operations defined by S=12, 13 and 14 are logic operations, and the output R is formed bit by bit from the two 4-bit input signals. Status signals C and V are set to 0. For S=15, the largest of the two 4-bit inputs A and B is connected to the output R with C and V set to 0.

Annex 2

The Testing Circuits and the Simulation Graphs are enclosed hereon:

1. FourBit Adder Testing:

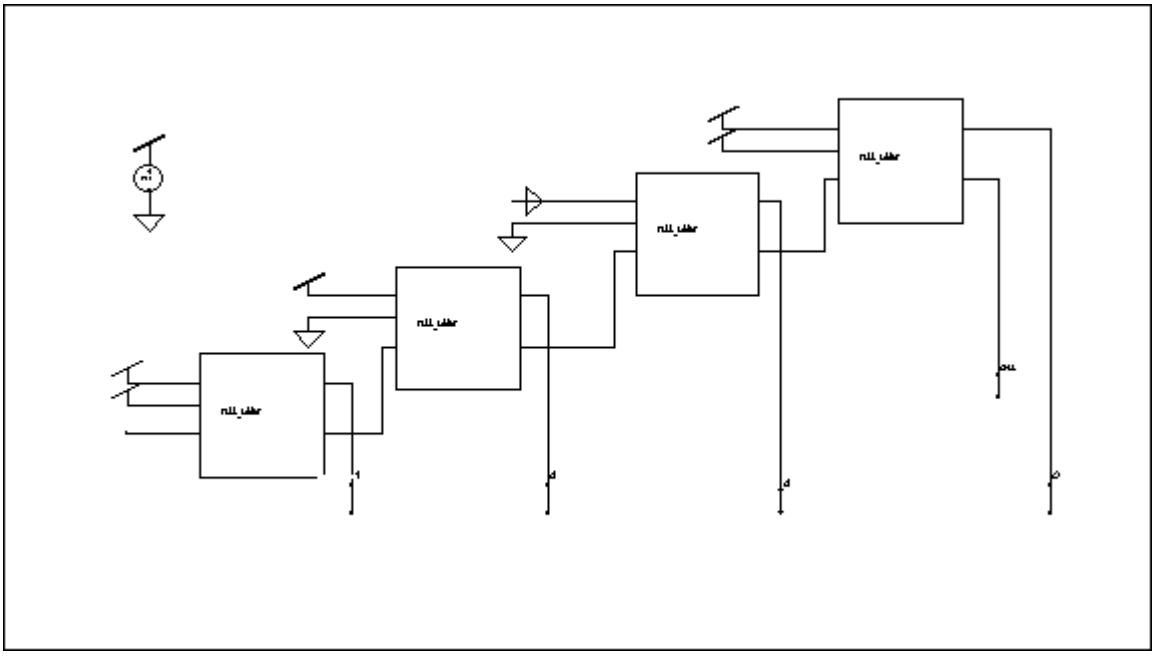


Figure 30 – Testing of 4 Bit Adder

2. 4x1 Mux Testing :

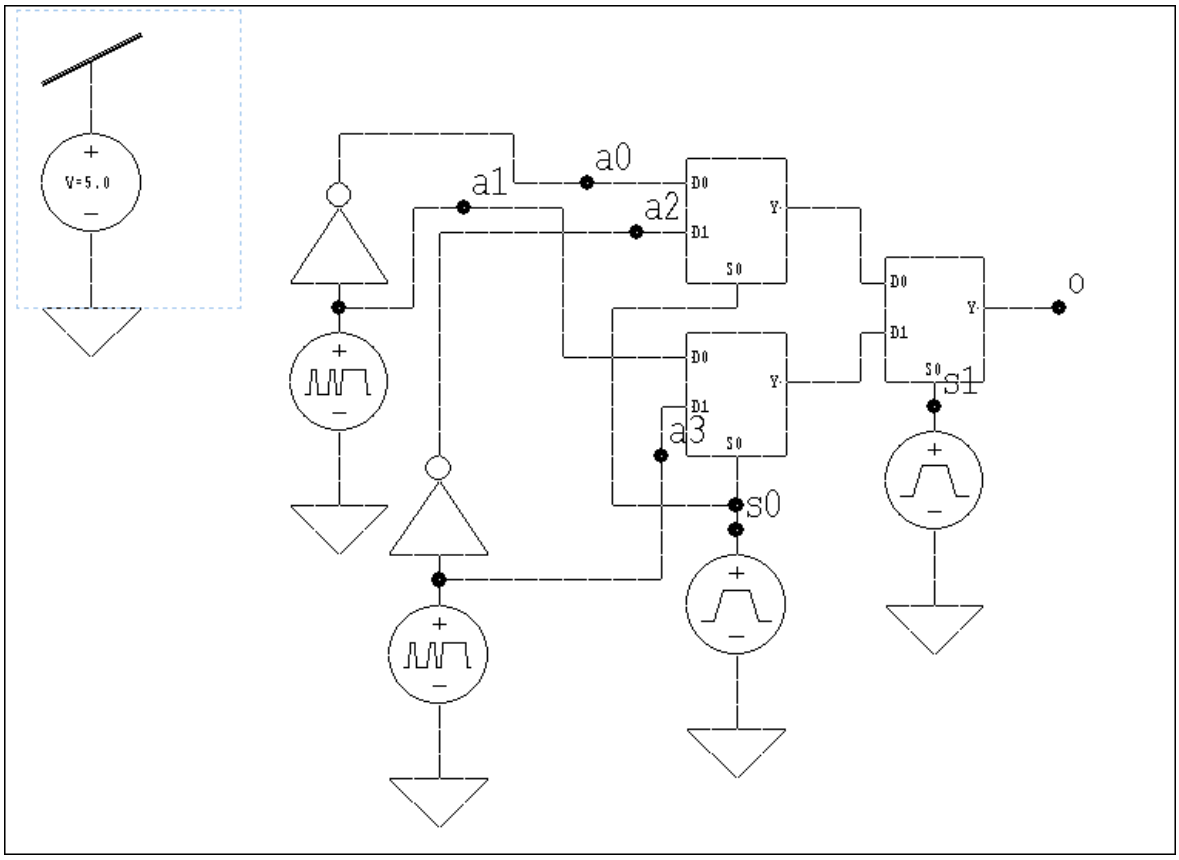


Figure 31 – Testing the 4x1 Mux

3. 8x1 Mux Testing.

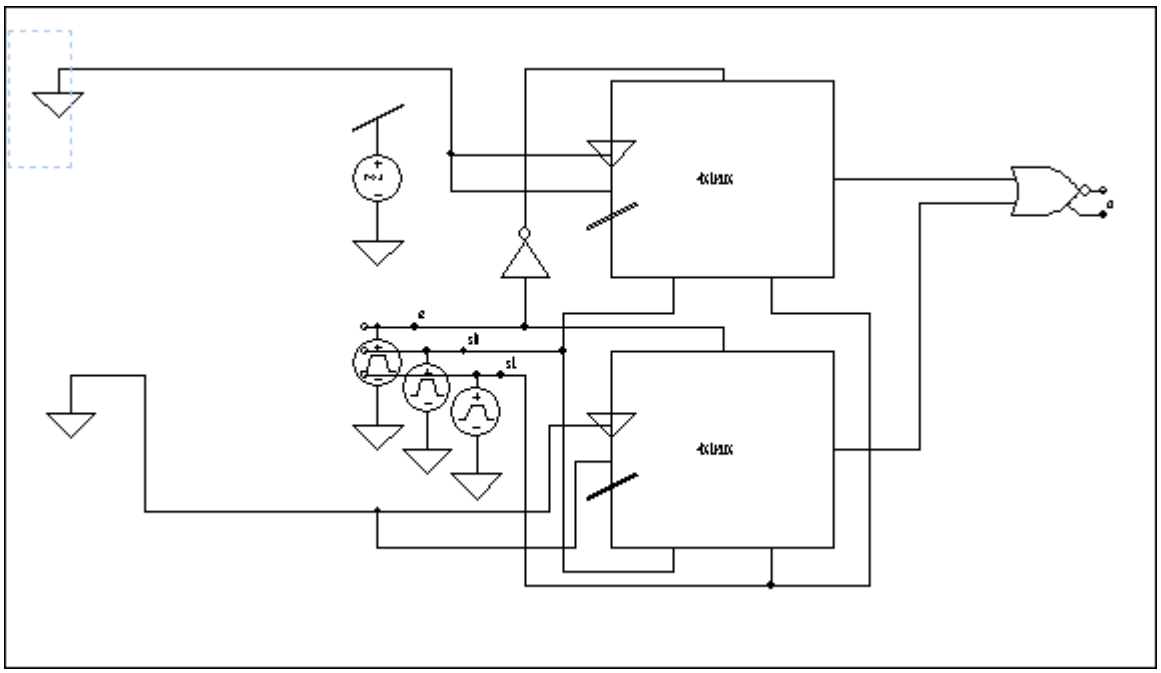


Figure 32 – Testing the 8x1 Mux

4. Shift Register Testing

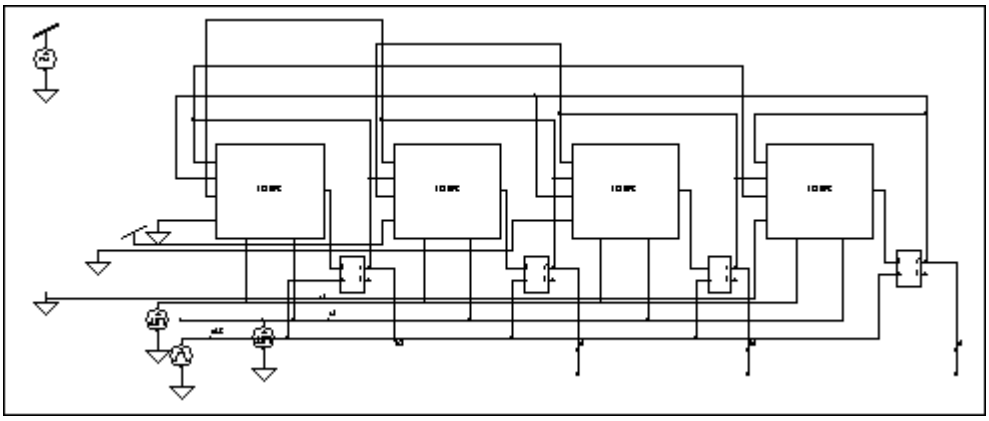


Figure 33 - Shifter Testing for Parallel Load and Right and Left Circular shifts

5. Add Sub Incr Decr Neg Max Testing

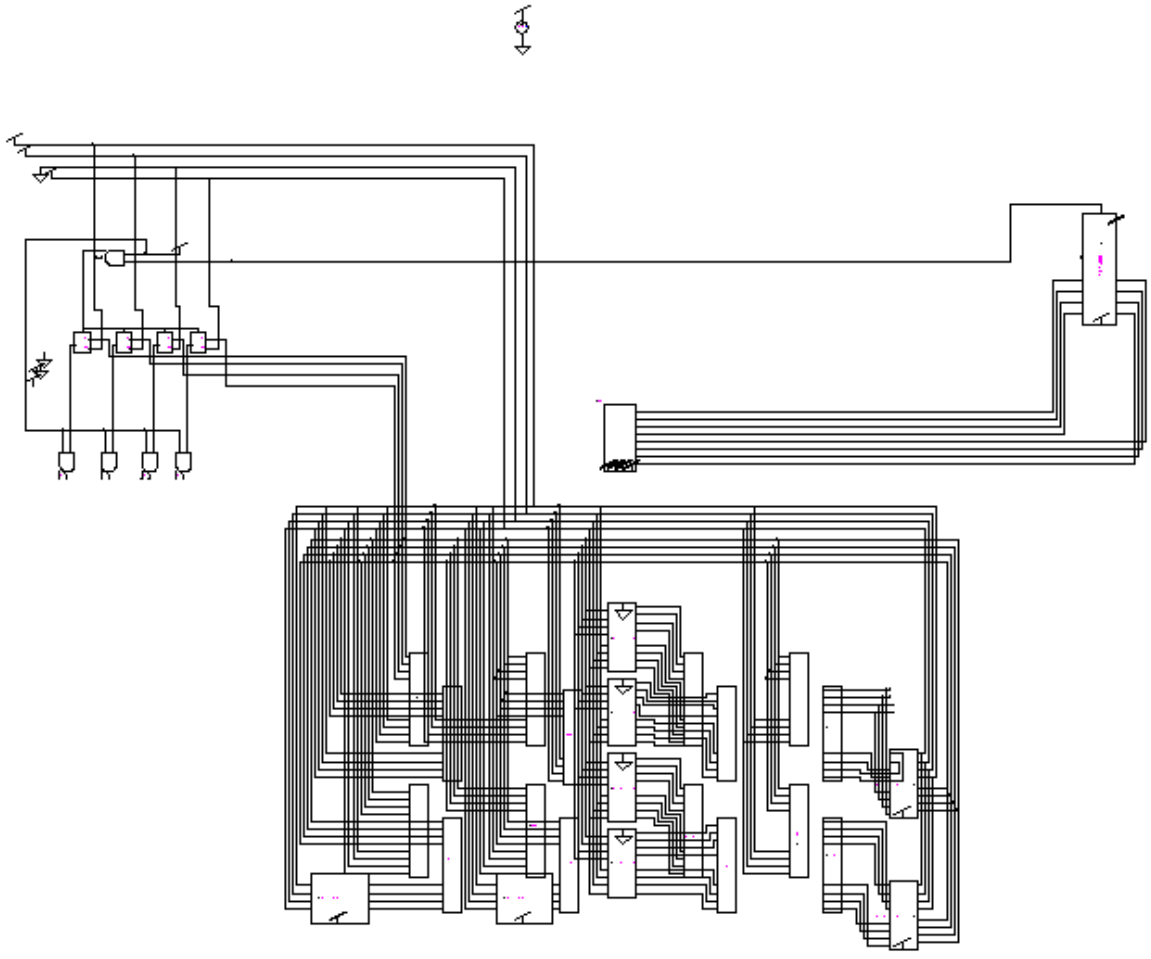


Figure 34 - Testing Add,Sub,Incr,Decr,Negate and Max Operations