**DESIGN OF ALU USING m-GDI TECHNIQUE**

**WITH 2T XOR GATE AND DECODER**

A mini project report submitted in partial fulfillment of the requirements for the degree of Bachelor of technology

In

Electronics and Communication Engineering

By

T.Ramya S190078

M.Sirisha S191059

B. Krishna Veni S191112

B.Akhila S190419

SK.Basheer Begum S190198

G.Sai Ram S190986

Under the esteemed guidance of

Dr.H.Srinivasa Varaprasad M.Tech, PhD

Assistant professor

Department of Electronics and Communication Engineering

Rajiv Gandhi University of Knowledge Technologies

SRIKAKULAM-532410,NOVEMBER- 2024

**CERTIFICATE**

This is to certify that the work entitled “**Design of alu using m-gdi technique with 2T xor gate and decoder”** is a bonafide record of authentic work carried out by S190078, S191059, S190419, S190198, S190986 under my supervision and guidance for the partial fulfillment of the requirement of the award of the degree of Bachelor of Technology in the Department of Electronics and Communication Engineering at RGUKT- SRIKAKULAM The results embodied in this work have not been submitted to any other university or institute for the award of any degree or diploma. This thesis, in our opinion, is worthy of consideration for the award of the degree of Bachelor of Technology in accordance with the regulations of the institute

Dr H Srinivasa Varaprasad, MTech, Ph.D., Miss. Sudha Rani , MTech

ECE Assistant Professor, Head of the Department,

Department of ECE, Department of ECE,

RGUKT-Srikakulam RGUKT-Srikakulam

**Acknowledgement**

I would like to express my special thanks of gratitude to my project guide **Dr H.Srinivasa Varaprasad**, Assistant Professor, Department of ECE, who gave me the golden opportunity to do this wonderful project on the topic **“** **Design of ALU using m-GDI Technique with 2T XOR gate and Decoder** **”**, which also helped me in doing a lot of research and I came to know about so many new things I am really thankful to teaching and non teaching staff .

We extend our gratitude for giving this opportunity to RGUKT-Srikakulam.

**Electronics and Communication Engineering**

**IIIT-SRIKAKULAM**

Date:12/11/2024

Place:Nuzvid

**TABLE OF CONTENTS**

| **s.no** | **Table of Contents** | **Page Numbers** |
| --- | --- | --- |
| 1 | TITLE | 1 |
| 2. | CERTIFICATE | 2 |
| 3. | ACKNOWLEDGEMENT | 3 |
| 4. | ABSTRACT | 6 |
| 5. | CHAPTER 1  INTRODUCTION | 7 |
| 6. | CHAPTER 2  LITERATURE REVIEW | 8 |
| 7. | CHAPTER 3  BLOCK DIAGRAM OF ALU | 9 |
| 8. | CHAPTER 4  EXISTING SYSTEM  GDI | 10 |
| 9. | CHAPTER 4  PROPOSED SYSTEM  M-GDI | 11-13 |
| 10. | CHAPTER 6  KEY COMPONENTS OF ALU  DIFFERENCE BETWEEN GDI AND M-GDI | 14-21 |
| 11. | CHAPTER 7  WORKING  APPLICATIONS | 22-24 |
| 12. | CHAPTER 8  CODE  OUTPUT WAVEFORMS | 25-30 |
| 13. | CONCLUSION | 31 |
| 14. | FUTURE SCOPE | 32 |
| 15. | REFERENCES | 32 |
| 16. | ANNEXURE | 32 |

**ABSTRACT**

**Problem Statement :**

Design an Arithmetic Logic Unit (ALU) leveraging a 2-Transistor (2T) XOR gate and a decoder to achieve optimized performance, reduced power consumption, and minimal transistor count, addressing the limitations of conventional ALU designs in resource-constrained environments.

**Existing Methods and Techniques:**

Over the last few decades a lot of work has been taken to do in conventional CMOS based circuits and more impact power efficiently. Various techniques, such as Transmission Logic Gate, Domino Logic, Pass transistor Logic, Double pass transistor Logic and others have been implemented in order to improve the performance of CMOS based circuits.

The Gate Diffusion Input (GDI) technique is a novel approach for drastically reducing power requirements . Furthermore, GDI reduces the number of transistors, resulting in a smaller chip size giving the designs an advantage over traditional methods.

**Drawbacks of Existing Methods:**

On the basis of GDI techniques, there are two types of transistors: PMOS and NMOS. Each of these has four subtypes: N,D,P,G .G,N,P serves as inputs. However GDI techniques have a big issue with gate diffusion input. Since NMOS generates Logic 1 and PMOS generates Logic 0.The existing GDI Technique is the major disadvantage.

**What We Are Doing in This Project:**

The aim of the project is to design an ALU that uses a 2-transistor XOR gate and a Modified Gate Diffusion Input (m-GDI) technique, a more compact and power-efficient design with modern technology. If the design is an adder using GDI technique it requires 28 transistors. But in m-GDI the adder requires only 8 transistors. This is the major advantage of the project.

**Expected Result:**

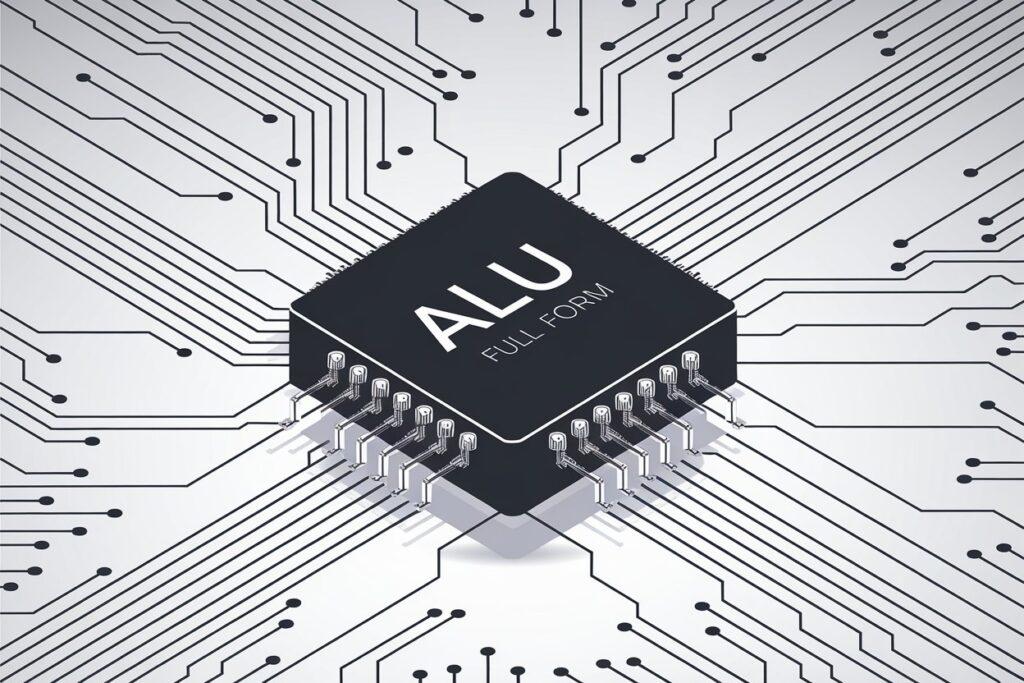
Compared to GDI Technique m-GDI Technique requires less number of transistors. When the count of transistors reduces automatically the chip size and power consumption will be reduced.

**Keywords:**

Arithmetic and Logic Unit (ALU) , VLSI Design , 2T XOR ,Modified Gate Diffusion Input (GDI) , Verilog

**CHAPTER 1**

**INTRODUCTION**

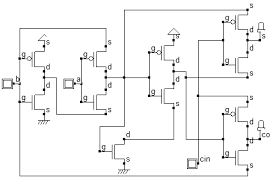
The Arithmetic Logic Unit's main component is the Central Processing Unit (CPU). Arithmetic operations such as Addition, Subtraction, Multiplication, and Division are included in the CPU. AND gate, OR gate, NAND gate, XOR gate, multiplexer. All of these operations are examples of logical operations. ALU operations necessitate a much larger number of capable computers, as well as application-specific circuits and VLSI chips. 

The main functions of designing larger are Low-power implementation and compact implementation Dissipation and these circuits are difficult to understand and they have no exception in ALU. Over the last few decades a lot of work has been taken to do in conventional CMOS based circuits and more impact power efficiently . Various techniques, such as Transmission Logic gate, Domino Logic, Pass Transistor logic, Double pass transistor logic and others have been implemented in order to improve the performance of CMOS - based circuits.

​The Gate Diffusion Input (GDI) technique is a novel approach for drastically reducing power requirements. Furthermore, GDI reduces the number of transistors, resulting in a smaller chip size, giving the designs an advantage over traditional methods .On the basis of GDI techniques, there are two types of transistors: PMOS and NMOS. Each of these has four subtypes: N.D.P and G . G,N,P serve as inputs.However,GDI techniques have a big issue with gate diffusion input.since NMOS generates logic 1 and PMOS generates logic 0.The existing GDI Technique is the major disadvantage.

The following segments were introduced in the modified Gate Diffusion Input are: **​**

* Arithmetic Unit: This unit is responsible for performing simple arithmetic operations such as addition and subtraction. ​
* Logical Unit: This component performs logical operations like OR, AND, XOR, NOT, and NAND gates are discussed in this article.​
* To pick the desired input line, a buffer unit was used. ​



**fig1:** 8-Bit ALU design using m-GDI technique

**CHAPTER 2**

**LITERATURE REVIEW**

| s.no | Title | Journal | Authors | Outcomes |
| --- | --- | --- | --- | --- |
| 1. | Modified Gate Diffusion Input Technique: A new  technique for enhancing performance in full adder circuits. | IEEE (2012) | Uma, r.,& Dhavachelvan,P.. | Pseudo-NMOS is simple and fast but reduces noise margins and increases power consumption. |
| 2. | 9T and 8T full adder subtractor design using modified gate diffusion input and 3T XOR technique. | IEEE(2019) | Sarkar, S..Sarkar.  S..Atta, A..Pahari. T..Majumdar. N..& Mondal, S. | Full Subtractor is implemented by using 3T XOR gate in GDI(Gate Diffusion Input) |
| 3. | 8-Bit Arithmetic Logic Unit Design Using Modified Gate Diffusion Input(m-GDI) Technique. | IJAEM(2021) | Anil Nageswar Rangapure, Dr. Kiran v | ALU design using Modified Gate Diffusion Technique |
| 4. | Design of ALU Using 2T XOR GAte and Decoder. | IJCI (2021) | Ayyappa Surendra Babu, N.Ashok Kumar. | ALU design using Modified Gate Diffusion Technique with less chip count |

**CHAPTER 3**

**BLOCK DIAGRAM OF ALU**

A diagram of a multiplier

Description automatically generated

**fig2:** Block Diagram of ALU

Inputs (A, B):

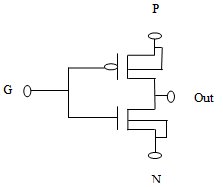
* The ALU takes two inputs, typically labeled A and B, which are the operands for the operation. These inputs are generally binary values.

Arithmetic and Logic Unit (ALU):

* The Arithmetic and Logic Unit performs both arithmetic (e.g., addition, subtraction, multiplication) and logical operations (e.g., AND, OR, NOT, XOR).
* The operations performed are based on the control signals received from the control unit.
* The ALU contains internal circuits for each type of operation:
* Arithmetic operations: Adders, subtractors, multipliers (basic arithmetic)**.**
* Logical operations: AND gates, OR gates, XOR gates (logical operations).

**CHAPTER 4**

**EXISTING SYSTEM**

**GATE DIFFUSION INPUT (GDI):**

The Gate Diffusion Input (GDI) technique is a low-power digital logic design method that aims to minimize the power consumption and area requirements of digital circuits. It is particularly useful in low-voltage, low-power applications. The GDI technique reduces the number of transistors needed for logic gates while maintaining the functionality of conventional logic designs. It is based on an innovative approach to implementing logic gates using only two transistors in a unique configuration.

Key Features of GDI Technique:

1. Low Power Consumption: GDI logic gates consume less power than conventional CMOS gates, making them ideal for low-power applications.  **fig3:** Basic GDI Unit Cell
2. Low Area: The number of transistors used is reduced compared to traditional CMOS designs, which saves silicon area.
3. High Speed: The technique can provide high-speed operation because of reduced parasitic capacitances and shorter signal paths.
4. Versatility: A variety of logic functions (AND, OR, XOR, etc.) can be implemented using this technique

Basic Working of the GDI Technique:

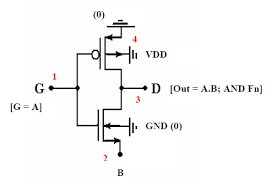
GDI logic gates are implemented using two n-type transistors (NMOS) and two p-type transistors (PMOS), with additional inputs for controlling the gate operations. The key element in the GDI approach is the Gate Diffusion concept, where the inputs are used directly at the gate terminals of the transistors.

### 

**CHAPTER 5**

**PROPOSED SYSTEM**

**Modified Gate Diffusion Input (MGDI)**

Modified Gate Diffusion Input (MGDI) is a variant of the Gate Diffusion Input (GDI) logic, designed to overcome some of the limitations in conventional CMOS and GDI logic styles, such as area, power consumption, and delay. MGDI modifies the basic GDI structure to improve performance in terms of speed and power efficiency, making it suitable for low-power applications.

**1.**Gate Diffusion Input (GDI) Basics

GDI is a logic style that provides a way to implement complex logic functions with minimal transistor count compared to CMOS. A basic GDI cell consists of three inputs:

* **G (Gate):** Common gate of both NMOS and PMOS transistors.
* **P (Source of PMOS):** Typically connected to a high voltage level (VDD).
* **N (Source of NMOS):** Typically connected to a low voltage level (GND).

**fig4:** Basic m-GDI Unit Cell

By manipulating the connections of G, P, and N, various logic functions can be implemented. MGDI was introduced to address these issues by modifying the basic GDI cell structure. The MGDI cell introduces additional control transistors and modifications in connection configurations. This helps:

* Achieve full swing in output, improving noise margins.
* Simplify the design for single power supply operation.
* Reduce power consumption by limiting unnecessary switching and leakage paths.

### 2. Working Principles of MGDI

### In MGDI, additional control transistors are added to the GDI structure, allowing it to support full-swing operations even with single power supply designs. These modifications ensure that the output does not suffer from threshold voltage drops, leading to improved logic levels. MGDI circuits typically use fewer transistors than equivalent CMOS designs but retain better noise margins compared to standard GDI.

**DECODER FOR ALU CONTROL**

MGDI logic offers a compact way to implement logic gates with fewer transistors, making it a great fit for low-power and high-speed decoders. In MGDI, logic functions are implemented by adjusting the connections to the G (gate), P (source of PMOS), and N (source of NMOS) terminals.A diagram of a circuit

Description automatically generated

An MGDI-based decoder can be built using a combination of MGDI logic gates (like AND and OR) to decode the control inputs into outputs. Here’s a step-by-step outline:

* **Input Lines:** The decoder takes binary inputs (opcode) that represent the operation code.
* **Output Lines:** Each output corresponds to a specific ALU function (e.g., ADD, SUBTRACT, AND).
* **Logic Gates:** The MGDI decoder primarily uses AND gates and inverters to decode the input combinations.

Each output line is generated by combining MGDI logic gates to activate only for specific binary input patterns, like 00, 01, 10, etc. **fig5:** Decoder 1-8 Basic Circuit

**2T XOR GATE**

The XOR gate is fundamental in ALU designs, especially for functions like addition, where XOR is used in full adder circuits. The 2T XOR gate in MGDI is implemented using only two transistors, making it highly efficient. Here’s how the 2T XOR gate is constructed in MGDI:

**Inputs (A and B):** These are the two binary inputs for the XOR gate.

* **Transistor Arrangement:** The gate uses only two transistors (one NMOS and one PMOS) with inputs configured to produce an XOR output.
* **Working:** The output will be high (1) when the inputs are opposite (one high, one low), which is the XOR condition.

A diagram of a circuit

Description automatically generated

**fig6:** m-GDI is used in the 2T XOR gate

**Implementing an ALU with MGDI and 2T XOR Gate**

In an ALU using MGDI, different functions are implemented in functional blocks, which are activated based on the decoder’s output. Here’s how the design could be organized:

* **Function Blocks:** Each operation (like addition, subtraction, etc.) is implemented in separate blocks.
* **XOR Gates in Full Adders:** The 2T XOR gate is used within the full adder circuit for addition. With MGDI, only two transistors are needed per XOR gate, greatly reducing transistor count.
* **Control Signals:** The decoder output activates the correct operation based on the opcode.
* **Single Power Supply:** Unlike conventional GDI, MGDI can operate with a single power supply, improving compatibility and stability.

**CHAPTER 6**

**KEY COMPONENTS OF ALU**A diagram of a circuit

Description automatically generated

**fig7:** Basic logic gates using m-GDI technique

The Modified Gate Diffusion Input (MGDI) technique allows the implementation of basic logic gates with fewer transistors than traditional CMOS designs, making them highly suitable for low-power and area-efficient circuits. Here’s how basic logic gates (like Inverter, AND, OR, NAND, NOR, XOR, and XNOR) are implemented using MGDI.

### 1. MGDI Inverter

* Inputs: Connect the G (gate) terminal to the input signal.
* Configuration:
  + P input is connected to VDD.
  + N input is connected to GND.
* Operation: The output is the inverted version of the input, acting as an inverter with only two transistors (one PMOS and one NMOS).
* Transistor Count: 2

### 2. MGDI AND Gate

* Inputs:
  + G is connected to input A.
  + P is connected to input B.
  + N is connected to GND.
* Operation: The output is high only when both A and B are high, making it an AND gate.
* Transistor Count: 2

### 

### 3. MGDI OR Gate

* Inputs:
  + G is connected to input A.
  + P is connected to VDD.
  + N is connected to input B.
* Operation: The output is high if either A or B is high, making it an OR gate.
* Transistor Count: 2

### 4. MGDI NAND Gate

* Inputs:
  + G is connected to input A.
  + P is connected to input B.
  + N is connected to GND.
* Operation: The output is low only when both A and B are high, providing the NAND operation.
* Transistor Count: 2

### 5. MGDI NOR Gate

* Inputs:
  + G is connected to input A.
  + P is connected to VDD.
  + N is connected to input B.
* Operation: The output is low if either A or B is high, making it a NOR gate.
* Transistor Count: 2

### 6. MGDI XOR Gate (2T XOR)

* Inputs:
  + G is connected to input A.
  + P is connected to input B.
  + N is connected to GND.
* Operation: The output is high when A and B are different, implementing the XOR function with only two transistors.
* Transistor Count: 2

### 7. MGDI XNOR Gate

* Inputs:
  + G is connected to input A.
  + P is connected to GND.
  + N is connected to input B.
* Operation: The output is high when both A and B are the same, giving an XNOR function.
* Transistor Count: 2

**FULL ADDER USING M-GDI TECHNIQUE**

A diagram of a machine

Description automatically generated

**fig8:** Full adder circuit using m-GDI

A full adder using the Modified Gate Diffusion Input (MGDI) technique can be implemented with minimal transistors, leveraging the simplicity and efficiency of MGDI logic. This low-transistor-count approach is especially useful in applications where power efficiency and area are important, such as embedded systems and energy-efficient processors.

The full adder circuit performs the addition of two binary digits, A and B, along with an input carry (Cin), producing a sum (S) and an output carry (Cout). The equations for a full adder are:

* **Sum (S) = A ⊕ B ⊕ Cin**
* **Carry-out (Cout) = (A • B) + (Cin • (A ⊕ B))**

### 1. MGDI Full Adder Design Overview

* Using MGDI logic, the full adder circuit can be broken down into two primary components:
  + **Sum Calculation**: Uses XOR gates.
  + **Carry-Out Calculation**: Uses AND and OR gates.
* The key advantage of MGDI is the use of a 2-transistor XOR gate, which greatly reduces the transistor count in the sum calculation.

### 2. Sum Calculation Using 2T XOR Gates

* **Step 1**: Calculate the intermediate XOR of A and B.
  + This produces A⊕B, which is then used in both the Sum and Carry-out calculations.
* **Step 2**: XOR the intermediate result A⊕B with Cin to get the Sum.
  + Sum (S) = (A⊕B)⊕Cin.
* Each XOR gate is implemented with two transistors (one PMOS and one NMOS), thanks to M-GDI logic, making the sum calculation compact and efficient.

### 3. Carry-Out Calculation Using MGDI AND and OR Gates

* **AND Gate for A and B**:
  + MGDI implements the AND function with only two transistors. This generates the term A•B.
* **AND Gate for Cin and (A ⊕ B)**:
  + Similarly, another 2-transistor MGDI AND gate is used to calculate Cin•(A⊕B).
* **OR Gate for Carry-Out**:
  + Finally, an MGDI OR gate combines the outputs of the two AND gates to produce Cout, calculated as:
    - Cout=(A•B)+(Cin•(A⊕B))

### 4. MGDI Full Adder Transistor Count

* **2 XOR gates** for Sum calculation (2 transistors each) = 4 transistors.
* **2 AND gates** for Carry-out calculation (2 transistors each) = 4 transistors.
* **1 OR gate** for Carry-out calculation (2 transistors) = 2 transistors.

**Total Transistor Count**: 10 transistors for a full adder, which is significantly less than conventional CMOS designs.

**FULL SUBTRACTOR USING M-GDI TECHNIQUE**A diagram of a circuit

Description automatically generated

**fig9:** Full Subtractor circuit using m-GDI

A full subtractor is a combinational circuit that performs subtraction on three inputs: the minuend (A), subtrahend (B), and borrow-in (Bin). The outputs are the difference (D) and borrow-out (Bout). Implementing a full subtractor using the Modified Gate Diffusion Input (MGDI) technique allows us to achieve a compact, low-power, and high-speed design.

### 1. Full Subtractor Logic

The full subtractor has the following logic equations for its two outputs:

* **Difference (D)**: D=A⊕B⊕Bin
* **Borrow-out (Bout)**: Bout=~A⋅Bin + ~A⋅B + B⋅Bin

### 2. MGDI-Based XOR Gate (for Difference Output)

The difference output in a full subtractor is achieved by a 3-input XOR gate, as shown by the equation for DDD. In MGDI, a 2T XOR gate is commonly used, and for a 3-input XOR, it’s built as a combination of two XOR operations:

1. **Step 1:** First XOR gate for A⊕B..
   * **Inputs:** G connected to A, P to B, and N to GND.
   * **Output:** Intermediate XOR result between A and B.
2. **Step 2:** Second XOR gate for (A⊕B)⊕Bin.
   * **Inputs:** G connected to A⊕B, P to Bin, and N to GND.
   * **Output:** Final difference output, D=A⊕B⊕Bin.

**Transistor Count:** 4 (2 transistors per XOR gate).

### 3. MGDI-Based Logic for Borrow-Out (Bout)

The borrow-out output uses a combination of AND and OR operations based on the equation:

Bout=~A⋅Bin + ~A⋅B + B⋅Bin

This expression can be implemented with the following steps in MGDI:

* **Step 1:** Implement ~A.Bin using an MGDI AND gate.
  + **Inputs:** G connected to A, P connected to GND (inverting A), and N connected to Bin.
* **Step 2:** Implement ~A.B using another MGDI AND gate.
  + **Inputs:** G connected to A, P connected to GND, and N connected to B.
* **Step 3:** Implement B⋅BinB using a standard MGDI AND gate.
  + **Inputs:** G connected to B, P connected to Bin, and N connected to GND.
* **Step 4:** Use an MGDI OR gate to combine the three terms.
  + **Inputs:** The OR gate combines the outputs of steps 1, 2, and 3.

**Transistor Count:** 8 transistors (6 for ANDs + 2 for OR).

**DIFFERENCE BETWEEN GDI,CMOS AND M-GDI**

| logic gates | **GDI** | **CMOS** | **M-GDI** |
| --- | --- | --- | --- |
| **XOR** | A collage of a diagram  Description automatically generated | A collage of a diagram  Description automatically generated | A diagram of a circuit  Description automatically generated |
| **AND** | A collage of a diagram  Description automatically generated | A collage of a diagram  Description automatically generated | A diagram of a circuit  Description automatically generated |
| **OR** |  |  |  |

**Here are the basic difference between GDI, CMOS and m-GDI:**A collage of a diagram

Description automatically generatedA collage of a diagram

Description automatically generatedA diagram of a circuit

Description automatically generated

**1.Number of Transistors**

CMOS : Requires a large number of transistors to implement basic logic functions (typically twice the number of transistors, as it uses both NMOS and PMOS transistors).

GDI: Reduce the number of transistors by using a more flexible configuration of inputs(Gate, Source, Drain and Bulk) to create logic gates.

m-GDI: Similar to GDI but ensures proper bulk connections (PMOS bulk toVDD and NMOS bulk to GND) for stable operation, still using fewer transistors compared to CMOS.

**2.Power Consumption:**

CMOS: Consumes more due to the higher number of transistors and switching activities.

GDI: Consumes less power because it uses fewer transistors and can avoid static power dissipation in some configurations.

m-GDI: Maintains the low power consumption of GDI but improves stability by avoiding issues like body effect and latch-up.

**3.Bulk Connection:**

CMOS: Bulk connections are fixed (NMOS bulk to GND, PMOS bulk to VDD) and do not change with the inputs.

GDI: Bulk connections can vary and be connected to different input signals, but this can introduce instability or require complex fabrication.

m-GDI: Simplifies the GDI method by fixing the bulk connections(PMOS bulk to VDD, NMOS bulk to GND), ensuring better stability and easier fabrication,while still reducing the transistor count.

**CHAPTER 7**

**WORKING**

Designing an Arithmetic Logic Unit (ALU) with the Modified Gate Diffusion Input (MGDI) technique involves structuring the ALU to handle basic arithmetic and logic operations with an emphasis on minimal power consumption and transistor count. This project overview covers the structure, components, and working operation of an MGDI-based ALU with a focus on using a 2-transistor XOR gate and an MGDI decoder.

This ALU can perform the following operations with low-power MGDI logic:

1. Addition
2. Subtraction
3. Bitwise AND
4. Bitwise OR

These operations are selected via control signals decoded by the ALU’s control decoder.

### Key Components

1. MGDI Decoder: Interprets the operation code (opcode) to generate control signals, enabling the appropriate function in the ALU.
2. 2T XOR Gate: Implements XOR logic with just two transistors, which is particularly useful in addition and subtraction circuits.
3. MGDI Logic Gates: Efficiently implement other logic functions like AND, OR, and INVERTER with minimal transistors.

### Working Operation

Here’s a step-by-step breakdown of how the MGDI ALU operates:

#### 1. Input and Control Signals

* Inputs: The ALU receives two binary numbers (let's say A and B, each 2 bits for simplicity in this design).
* Control Signals: These control signals come from the opcode and are used by the MGDI decoder to enable specific functions. For a 2-bit control input (opcode), we can have four operations:
  + 00: Addition
  + 01: Subtraction
  + 10: Bitwise AND
  + 11: Bitwise OR

#### 2. MGDI Decoder

* The decoder interprets the opcode (control input) and activates one of the four possible operations.
* This decoder is built using MGDI logic gates (primarily AND and OR), configured to output a high signal on one line depending on the opcode.

#### 3. ALU Function Blocks

Each function (ADD, SUBTRACT, AND, OR) is built as a separate block using MGDI logic.

* Addition (Full Adder):
  + A full adder circuit is implemented for addition using the 2T XOR gate, along with MGDI-based AND and OR gates.
  + The sum output of the full adder is calculated using XOR, and carry-out is generated through MGDI AND and OR gates.
* Subtraction:
  + Subtraction is achieved by using the same full adder with the B input inverted (for two's complement subtraction).
  + An MGDI inverter is used to invert the bits of B before feeding them into the adder.
* Bitwise AND and OR:
  + The bitwise AND and OR functions are implemented using MGDI-based AND and OR gates for direct logical operations between A and B.

#### 4. Selecting the Output

* Each function block outputs a result based on the selected operation.
* The outputs from these blocks are then fed into a multiplexer controlled by the decoder output, allowing only the selected function's output to appear at the final ALU output.

#### 5. Example Operation

Suppose we want to perform an addition of A = 01 and B = 10 with opcode 00 (ADD):

* The ALU control decoder interprets 00 and activates the addition block.
* The 2-bit numbers A and B are processed by the full adder using the 2T XOR gate for sum calculation.
* The sum and carry outputs are generated, producing the final result on the ALU output.

### Advantages of Using MGDI with 2T XOR Gate

* Reduced Transistor Count: The 2T XOR gate, coupled with other MGDI gates, drastically reduces the transistor count compared to CMOS designs.
* Power Efficiency: Fewer transistors and lower switching activity lead to reduced power consumption, ideal for low-power applications.
* Improved Speed: The simplicity of MGDI circuits enables faster signal propagation, enhancing ALU speed.

**APPLICATIONS**

**Application of MGDI-Based ALU in Projects**

This compact and efficient ALU design finds a wide range of applications, particularly in areas that require low power and limited space. Here are some key applications:

* Embedded Systems and IoT Devices : In embedded systems, where power and area constraints are critical, an MGDI-based ALU is ideal for performing arithmetic and logical operations with minimal energy consumption. This design is suitable for devices like sensors, wearable technology, and battery-powered IoT devices.
* Portable Medical Devices : Many medical devices, such as portable ECG monitors and wearable health trackers, require power-efficient ALUs to process real-time data. The MGDI-based ALU can handle data processing tasks while consuming very little power, extending battery life.
* Low-Power Signal Processing : Applications like digital filters and fast Fourier transforms (FFTs) in signal processing can benefit from a low-power ALU. An MGDI-based ALU is particularly useful for DSP operations in audio processing, image processing, and communication systems where power and speed are both essential.
* Energy-Efficient Processors : In processors designed for mobile and wearable devices, an MGDI-based ALU reduces power consumption, extending the battery life of the device. These processors may use MGDI ALUs as core components in their arithmetic logic circuits.
* Data Encryption and Compression : Operations within encryption algorithms and data compression techniques often rely on basic arithmetic and logical operations. A low-power ALU using MGDI can help achieve secure data processing while keeping energy requirement

**CHAPTER 8**

**CODE**

**Source Code:**

// XOR Gate - Verilog Representation of the 2T XOR Gate

module xor\_gate (

input a,

input b,

output y);

assign y = a ^ b; // XOR operation

endmodule

// 2:4 Decoder - Used to control which operation is selected in the ALU

module decoder\_2x4 (

input [1:0] select,

output reg [3:0] out);

always @(\*) begin

case (select)

2'b00: out = 4'b0001; // Select Operation 0 (Addition)

2'b01: out = 4'b0010; // Select Operation 1 (Subtraction)

2'b10: out = 4'b0100; // Select Operation 2 (AND)

2'b11: out = 4'b1000; // Select Operation 3 (XOR)

default: out = 4'b0000; // Default case

endcase

end

endmodule

// 8-bit ALU Module

module alu\_8bit (

input [7:0] A, // 8-bit Input A

input [7:0] B, // 8-bit Input B

input [1:0] select, // 2-bit selection from decoder to choose operation

output reg [7:0] result, // 8-bit ALU result

output carry\_out // Carry output (for addition and subtraction)

);

wire [7:0] sum, diff, and\_op, xor\_op;

wire [3:0] control; // 4-bit control signal from decoder

// Instantiate the 2T XOR gate module for each bit

xor\_gate xor0(.a(A[0]), .b(B[0]), .y(xor\_op[0]));

xor\_gate xor1(.a(A[1]), .b(B[1]), .y(xor\_op[1]));

xor\_gate xor2(.a(A[2]), .b(B[2]), .y(xor\_op[2]));

xor\_gate xor3(.a(A[3]), .b(B[3]), .y(xor\_op[3]));

xor\_gate xor4(.a(A[4]), .b(B[4]), .y(xor\_op[4]));

xor\_gate xor5(.a(A[5]), .b(B[5]), .y(xor\_op[5]));

xor\_gate xor6(.a(A[6]), .b(B[6]), .y(xor\_op[6]));

xor\_gate xor7(.a(A[7]), .b(B[7]), .y(xor\_op[7]));

// Perform Arithmetic Operations

assign {carry\_out, sum} = A + B; // Addition

assign diff = A - B; // Subtraction

// Perform Logical Operations

assign and\_op = A & B; // AND Operation

// Instantiate the 2:4 decoder

decoder\_2x4 dec(.select(select), .out(control));

// ALU Result Selection Based on Decoder Output

always @(\*) begin

case (control)

4'b0001: result = sum; // Addition

4'b0010: result = diff; // Subtraction

4'b0100: result = and\_op; // AND

4'b1000: result = xor\_op; // XOR

default: result = 8'b00000000; // Default: No operation

endcase

end

endmodule

**Test Bench Code:**

`timescale 1ns / 1pc

module tb\_alu\_8bit; // Testbench for 8-bit ALU

// Inputs

reg [7:0] A;

reg [7:0] B;

reg [1:0] select;

// Outputs

wire [7:0] result;

wire carry\_out;

// Instantiate the ALU module

alu\_8bit uut (

.A(A),

.B(B),

.select(select),

.result(result),

.carry\_out(carry\_out) );

// Test vectors and stimulus

initial begin

// Monitor outputs

$monitor("Time: %0d A=%b B=%b select=%b -> result=%b carry\_out=%b", $time, A, B, select, result, carry\_out);

// Initialize Inputs and apply test cases

// Test Case 1: Addition (A + B)

A = 8'b00001111; // 15 in decimal

B = 8'b00000001; // 1 in decimal

select = 2'b00; // Select addition operation

#10; // Wait 10 time unit

// Test Case 2: Subtraction (A - B)

A = 8'b00010010; // 18 in decimal

B = 8'b00000011; // 3 in decimal

select = 2'b01; // Select subtraction operation

#10; // Wait 10 time unit

// Test Case 3: AND Operation (A & B)

A = 8'b11001100; // A = 204

B = 8'b10101010; // B = 170

select = 2'b10; // Select AND operation

#10; // Wait 10 time units

// Test Case 4: XOR Operation (A ^ B)

A = 8'b11110000; // A = 240

B = 8'b00001111; // B = 15

select = 2'b11; // Select XOR operation

#10; // Wait 10 time unit

// Test overflow in addition

A = 8'b11111111; // A = 255

B = 8'b00000001; // B = 1

select = 2'b00; // Select addition operation

#10; // Wait 10 time units

// Test Case 5: Subtraction with negative result

A = 8'b00000001; // A = 1

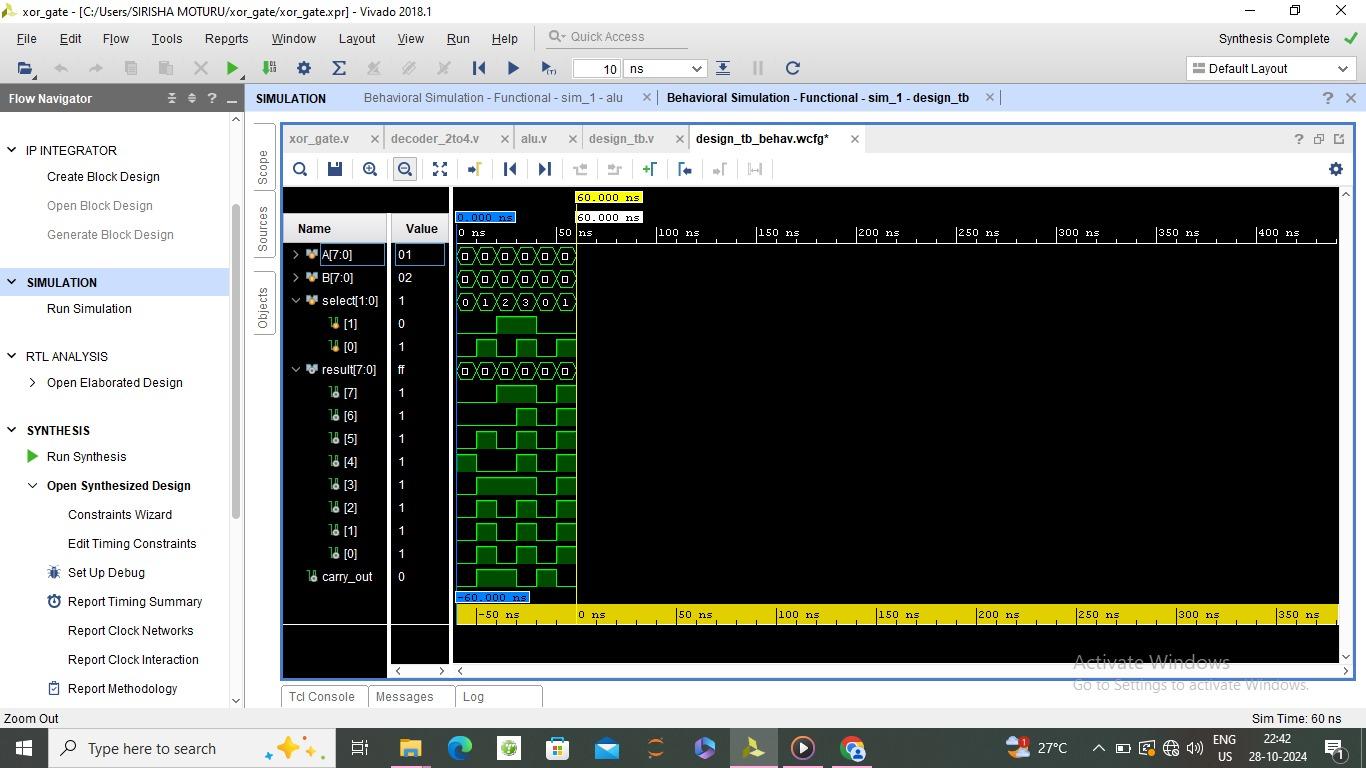
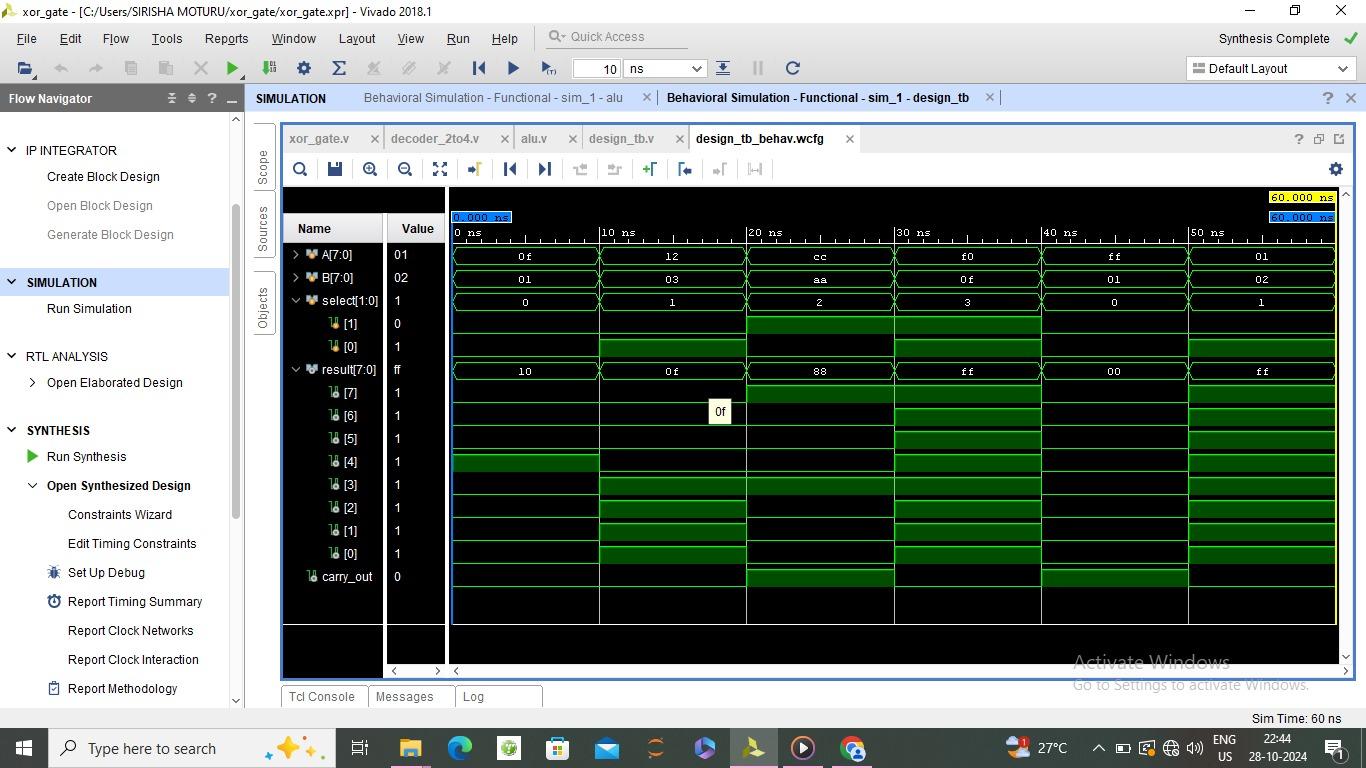
B = 8'b00000010; // B = 2

select = 2'b01; // Select subtraction operation

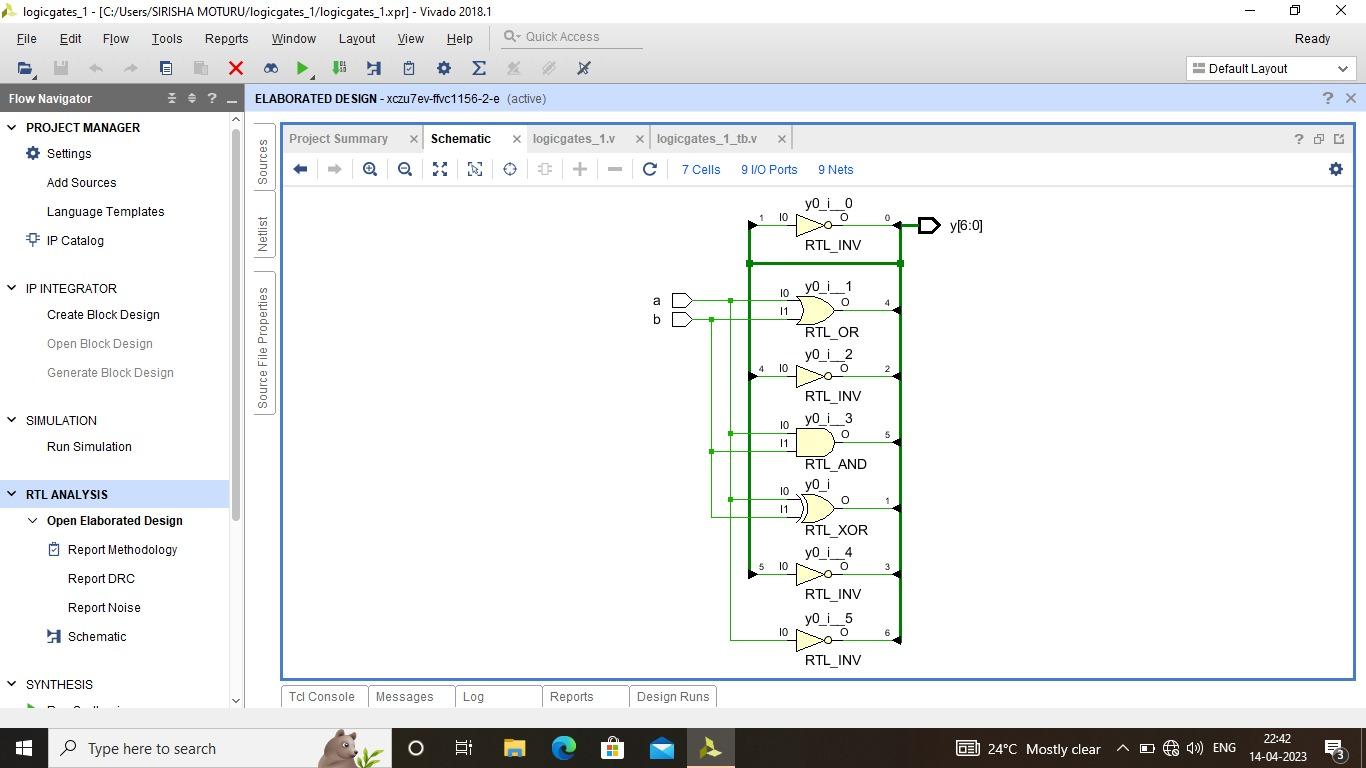
#10; // Wait 10 time units

$finish; // End the simulation end endmodule

**Output Waveforms**

**Simulation Results:** 

**SCHEMATIC DIAGRAM**



**CONCLUSION**

The m-GDI technique, combined with the 2T XOR gate design, significantly reduces the transistor count compared to traditional CMOS-based designs. This reduction in transistor usage leads to a smaller chip area, which is advantageous for compact designs and portable applications.By using the m-GDI technique, the ALU design achieves lower power consumption due to fewer switching activities and reduced parasitic capacitance. This makes it particularly suitable for energy-constrained applications like IoT devices and wearable technology.

The reduced transistor count and simpler gate design help minimize signal propagation delay, thus enhancing the overall speed of the ALU. This is beneficial for applications requiring fast computational capabilities, such as digital signal processing and real-time systems.While the m-GDI technique and the 2T XOR gate improve efficiency, they can add design complexity in the integration of the ALU with other digital components. However, the reduced complexity of individual gates supports scalability, making it feasible to extend the ALU design for more complex operations without drastically increasing power or area requirements.

A challenge in using m-GDI is its reduced noise margin compared to standard CMOS, which may affect reliability in noisy environments. Careful layout and design techniques can mitigate this issue, though further optimizations may be needed to maintain signal integrity in larger, more complex systems.

In conclusion, using the m-GDI technique with a 2T XOR gate in ALU design proves to be an effective approach for low-power, high-speed, and area-efficient digital circuits. This technique is well-suited for applications where efficiency is crucial, although designers must carefully address noise margins and integration complexity for robust performance.

**FEATURE SCOPE**

The future scope for ALU design using the m-GDI technique with a 2T XOR gate and decoder is promising, especially in the context of low-power, high- performance computing. This approach can be further optimized for use in advanced microprocessors, embedded systems, and portable devices, where energy efficiency and space are critical. Additionally, as technology nodes continue to shrink, m-GDI based designs can help mitigate power leakage and heat dissipation challenges. The technique can also be explored for use in applications like AI accelerators, Internet of Things (IOT) devices, and wearable electronics, where efficient power management is crucial.

**REFERENCES**

* “Modified Gate Diffusion Input Technique: A New Technique for Enhancing Performance in Full Adder Circuits”, Procedia Technology 6 ( 2012 ) 74 –81 by P.Dhavachelvan and R.Uma.
* “Minimizing power consumption in digital CMOS circuits,” Proc. IEEE, vol. 83, pp. 498–523, Apr.1995, by R.W.Broderson and A. P. Chandrakasan.
* “Gate-Diffusion Input (GDI): A Power-Efficient Method for Digital Combinatorial Circuits”, IEEE Transactions On Very Large Scale Integration (VLSI) Systems, Vol. 10, No. 5, October2002 by Israel A.Wagner ,Alexander Fish and Arkadiy Morgenstein
* “4-Bit Fast Adder Design: Topology and Layout with Self- Resetting Logic for Low Power VLSI Circuits” International Journal of Advanced Engineering Sciences and Technology, Vol No. 7, Issue No. 2, pp 197- 205,2011. by R.UMA.

**ANNEXURE**

https://ijcionline.com/paper/10/10321ijci06.pdf