

**The University of Azad Jammu and Kashmir,**

**Muzaffarabad**

|  |  |
| --- | --- |
| Name | Kamal Ali Akmal |
| Course Name | **Computer Architecture and Logic Design** |
| Submitted to | **Engr. Sidra Rafique** |
| Semester | **2nd** |
| Session | **2024-2028** |
| Roll No | **2024-SE-38** |
| Lab | **Open Ended Lab** |
| Submission date | **05 September 2025** |

**Task 1: Parity Generator and Checker**

**1. 4-bit Parity Generator Circuit**

* Inputs: A, B, C, D (4-bit data word).
* Output: Parity Bit (P).
* **Even parity rule:** The parity bit is chosen such that the total number of 1’s (data bits + parity) is even.

***Boolean Equation:***

P=A⊕B⊕C⊕DP

**2. Implementation**

* Use XOR gates in PLA or a digital logic simulator like Electronic Work Bench..
* Connect inputs A, B, C, D → XOR chain → Output P.

**3. Testing the Generator**

* Try all 16 combinations of A, B, C, D.
* Verify that (A ⊕ B ⊕ C ⊕ D) + P always gives **even number of 1s**.

**4. Parity Checker Circuit**

* Inputs: A, B, C, D, and received Parity Bit (P).
* Output: Error signal (E).

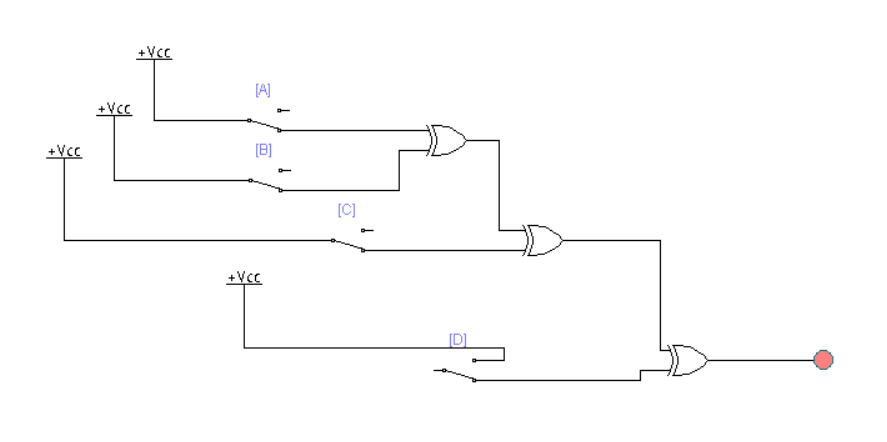
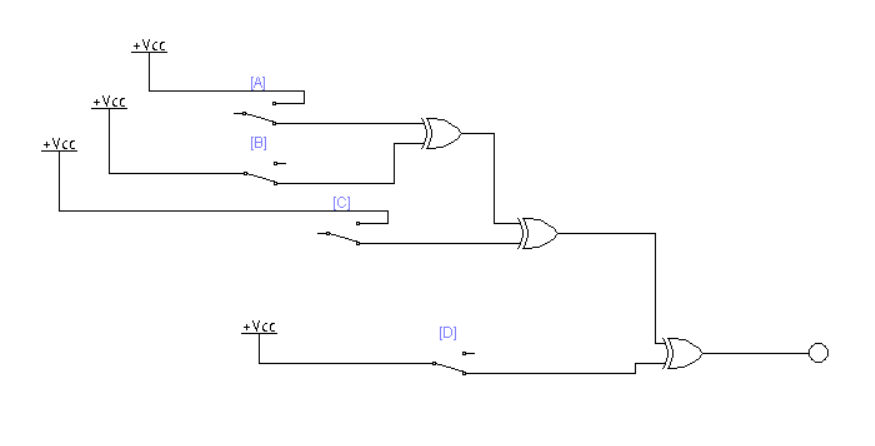
***Boolean Equation:***

E=A⊕B⊕C⊕D⊕P

* If **E = 0** → No error.
* If **E = 1** → Error detected.

**5. Implementation and Testing**

* Connect inputs to XOR chain.
* Apply different data with parity bit.
* Check if circuit correctly detects error when a bit flips.

**4-bit Parity Generator Circuit**

**Figure 1: When two Switch is OFF and two ON means inputs (A, B, C, D) = (0 ,1, 0,1) then output is 0.**

**Figure 2: When three Switches ON and one OFF means all inputs (A, B, C, D) = (1,1,1,0), then output is 1.**

**4-bit Parity Generator (Even Parity): -**

Formula: P=A⊕B⊕C⊕D

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A |  |  |  |  |  |  |  | B |  |  |  |  |  |  |  | C |  |  |  |  |  |  |  |  | D |  |  |  |  |  |  |  | P (Parity) |
| 0 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 0 |
| 0 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 1 |
| 0 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 1 |
| 0 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 0 |
| 0 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 1 |
| 0 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 0 |
| 0 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 0 |
| 0 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 1 |
| 1 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 1 |
| 1 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 0 |
| 1 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 0 |
| 1 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 1 |
| 1 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 0 |
| 1 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 1 |
| 1 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  |  | 1 |
| 1 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  |  | 1 |  |  |  |  |  |  |  | 0 |

A computer screen shot of a computer

AI-generated content may be incorrect.

**4-bit Parity Checker (Error Detection)**

* Inputs: A, B, C, D, and received Parity Bit (P).
* Output: Error signal (E).

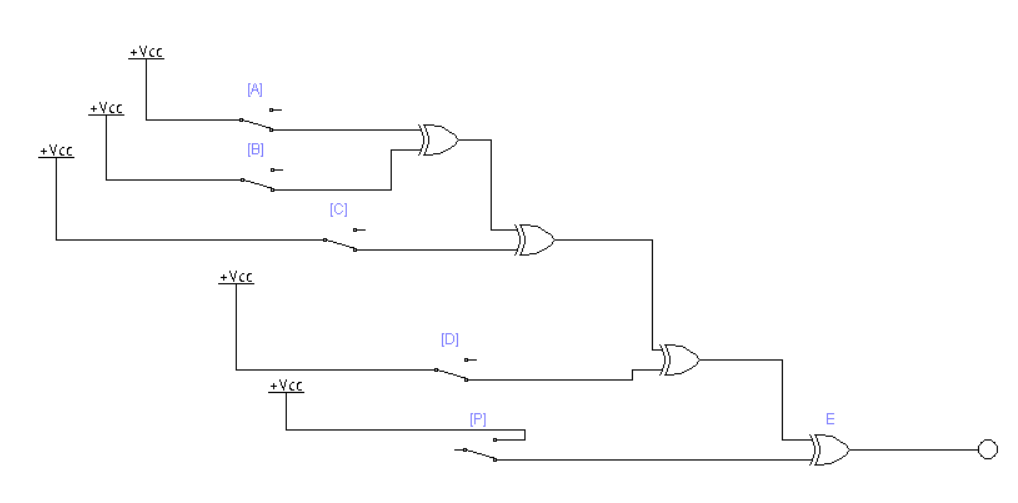
***Boolean Equation:***

E=A⊕B⊕C⊕D⊕P

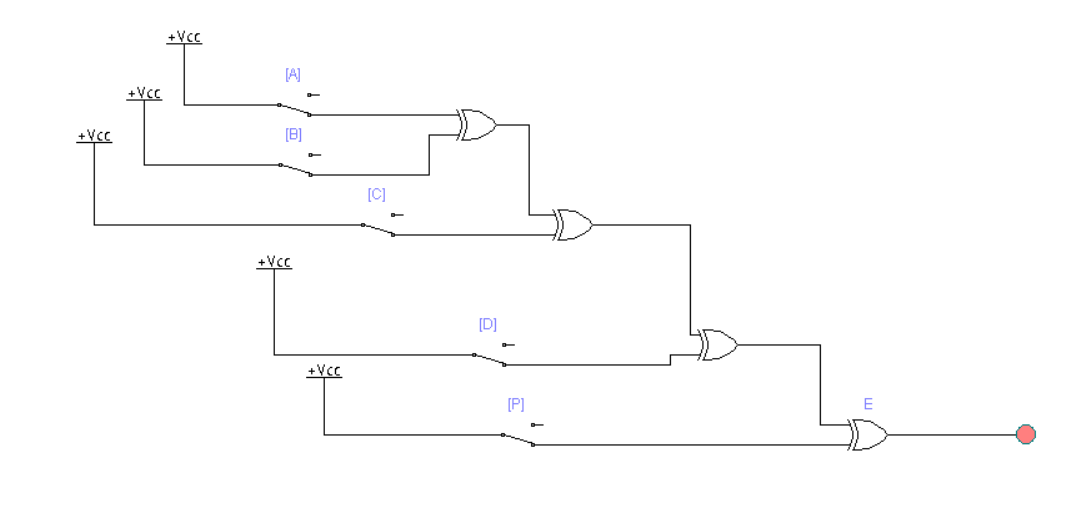
* If **E = 0** → No error.
* If **E = 1** → Error detected.

**Implementation and Testing**

* Connect inputs to XOR chain.
* Apply different data with parity bit.
* Check if circuit correctly detects error when a bit flips.

**4-bit Parity Checker (Error Detection)**

**Figure 1 : When four Switches ON and one OFF means all inputs (A, B, C, D, P) = (1,1,1,1,0), then output is 0.**



**Figure 2: When all Switches ON means all inputs (A, B, C, D, P) = (1,1,1,1,1), then output is 1**

**Parity Checker Truth Table**

***Boolean Equation:***

E=A⊕B⊕C⊕D⊕PE

Inputs: **A, B, C, D, P** → Output: **E (Error)**

| **A** |  |  |  |  |  | **B** |  |  |  |  |  | **C** |  |  |  |  |  |  | **D** |  |  |  |  |  |  | **P (Expected)** |  |  |  |  | **E (Output)** |  |  |  |  |  | **Meaning** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 |  |  |  |  |  | 0 |  |  |  |  |  | 0 |  |  |  |  |  |  | 0 |  |  |  |  |  |  | 0 |  |  |  |  | 0 |  |  |  |  |  | ✅ No error |
| 0 |  |  |  |  |  | 0 |  |  |  |  |  | 0 |  |  |  |  |  |  | 0 |  |  |  |  |  |  | 1 |  |  |  |  | 1 |  |  |  |  |  | ❌ Error |
| 1 |  |  |  |  |  | 0 |  |  |  |  |  | 1 |  |  |  |  |  |  | 0 |  |  |  |  |  |  | 0 |  |  |  |  | 0 |  |  |  |  |  | ✅ No error |
| 1 |  |  |  |  |  | 0 |  |  |  |  |  | 1 |  |  |  |  |  |  | 0 |  |  |  |  |  |  | 1 |  |  |  |  | 1 |  |  |  |  |  | ❌ Error |
| 1 |  |  |  |  |  | 1 |  |  |  |  |  | 1 |  |  |  |  |  |  | 1 |  |  |  |  |  |  | 0 |  |  |  |  | 0 |  |  |  |  |  | ✅ No error |
| 1 |  |  |  |  |  | 1 |  |  |  |  |  | 1 |  |  |  |  |  |  | 1 |  |  |  |  |  |  | 1 |  |  |  |  | 1 |  |  |  |  |  | ❌ Error |

**Task 2: Low-Level Program for Shift Instructions**

1. **SHL (Shift Left Logical)** – Multiplies number by 2.
2. **SHR (Shift Right Logical)** – Divides unsigned number by 2.
3. **SAR (Shift Arithmetic Right)** – Divides signed number by 2, keeps sign.
4. **SAL (Shift Arithmetic Left)** – Similar to SHL.

MOV AL, 00001101b ; Load data = 13 (binary)

SHL AL, 1 ; Shift Left → result = 26

SHR AL, 1 ; Shift Right → result = 13

SAR AL, 1 ; Arithmetic Right → sign preserved

SAL AL, 1 ; Arithmetic Left → same as SHL

**By using emulator,**

**Step #01:**

A screenshot of a computer

AI-generated content may be incorrect.

**Result:**

A screenshot of a computer

AI-generated content may be incorrect.

**Deliverables**

1. **Working Circuits**
   * 4-bit parity generator and checker implemented in simulator.
2. **Truth Table & Boolean Equations**

| **A** |  |  |  |  |  |  |  | **B** |  |  |  |  |  |  | **C** |  |  |  |  |  |  | **D** |  |  |  |  |  |  | **Parity P = A⊕B⊕C⊕D** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  | 0 |  |  |  |  |  |  | 0 |  |  |  |  |  |  | 0 |
| 0 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  | 0 |  |  |  |  |  |  | 1 |  |  |  |  |  |  | 1 |
| 0 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  | 1 |  |  |  |  |  |  | 0 |  |  |  |  |  |  | 1 |
| 0 |  |  |  |  |  |  |  | 0 |  |  |  |  |  |  | 1 |  |  |  |  |  |  | 1 |  |  |  |  |  |  | 0 |
| … |  |  |  |  |  |  |  | … |  |  |  |  |  |  | … |  |  |  |  |  |  | … |  |  |  |  |  |  | … |

***Boolean Equation:***

E=A⊕B⊕C⊕D⊕P

1. **Summary**
   * Designed a 4-bit parity generator using XOR gates.
   * Verified functionality with truth table and simulation.
   * Designed a parity checker for error detection.
   * Implemented and tested both circuits in simulator.
   * Wrote Assembly program for shift operations SHL, SHR, SAR, SAL.