# EAST WEST UNIVERSITY

## Project Report

Course Title: Digital Logic Design

Course Code: CSE 345

**Topic:** A 4-bit code converter for even parity converts the 4-bit input to 5-bit output so that even parity is ensured.

Submitted To:

**Musharrat Khan**
Senior Lecturer
Department of Computer Science and Engineering

Submitted By

**Sharmin Akther Rima**

ID: **2018-2-60-112**

Department of CSE

Section: 03

Submitted Date**: September 12, 2021**

# Problem Statement

A 4-bit code converter for even parity converts the 4-bit input to 5-bit output so that even parity is ensured.

# Design Details

## Truth Table:

| Input Variable | | | | Parity Bit | | | | |
|---|---|---|---|---|---|---|---|---|
| **A** | **B** | **C** | **D** | **P** | **E** | **F** | **G** | **H** |
| 0 | 0 | 0 | 0 | **0** | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | **1** | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | **1** | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 | **0** | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | **1** | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 | **0** | 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | **0** | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 | **1** | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | **1** | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | **0** | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | **0** | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | **1** | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | **0** | 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | **1** | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | **1** | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | **0** | 1 | 1 | 1 | 1 |

## K-Map and Function for Output P:

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | | 1 | | 1 |
| 01 | 1 | | 1 | |
| 11 | | 1 | | 1 |
| 10 | 1 | | 1 | |

P= A’B’C’D+A’B’CD’+A’BC’D’+A’BCD+ABC’D+ABCD’+AB’C’D’+AB’CD

=A’B’(C’D+CD’) +A’B(C’D’+CD) +AB(C’D+CD’) +AB’(C’D’+CD)

=A’B’(C xor D)+A’B(C xnor D)+ AB(C xor D)+ AB’ (C xnor D)

=(C xor D) (A’B’+ AB) + (C xnor D) (A’B+ AB’)

=(C xor D)(A xnor B)+(C xnor D)(A xor B)

=(C xor D) (A xor B)’ + (C xor D)’ (A xor B)

=(C xor D) xor (A xor B)

=(A xor B xor C xor D)

## K-Map and Function for Output E:

**K-Map:**

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | | | | |
| 01 | | | | |
| 11 | 1 | 1 | 1 | 1 |
| 10 | 1 | 1 | 1 | 1 |

E=A

## K-Map and Function for Output F:

**K-Map:**

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | | | | |
| 01 | 1 | 1 | 1 | 1 |
| 11 | 1 | 1 | 1 | 1 |
| 10 | | | | |

F=B

## K-Map and Function for Output G:

**K-Map:**

| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | | | 1 | 1 |
| 01 | | | 1 | 1 |
| 11 | | | 1 | 1 |
| 10 | | | 1 | 1 |

G=C

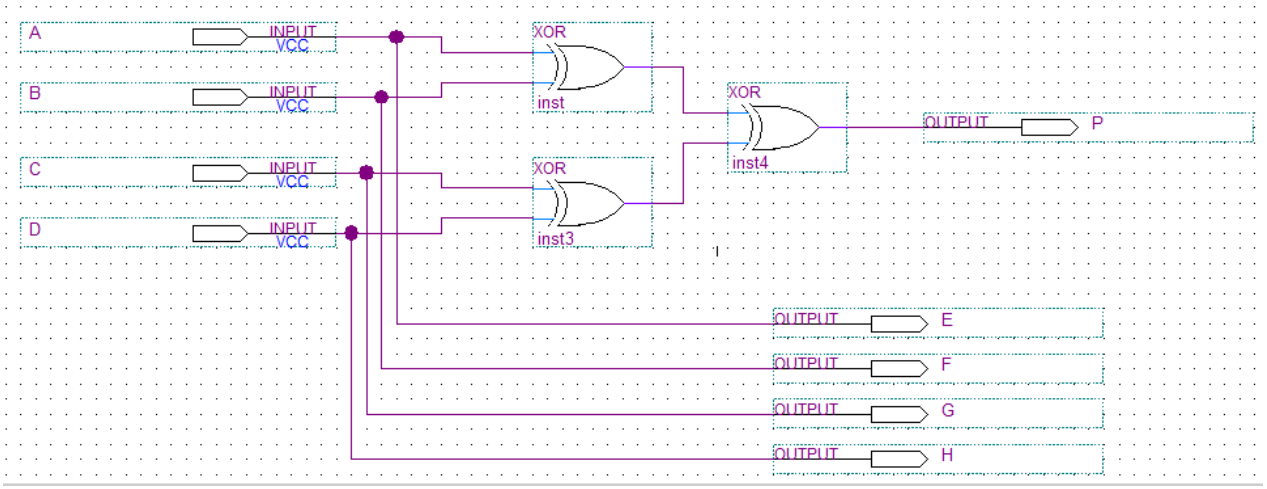
## K-Map and Function for Output H:

**K-Map:**

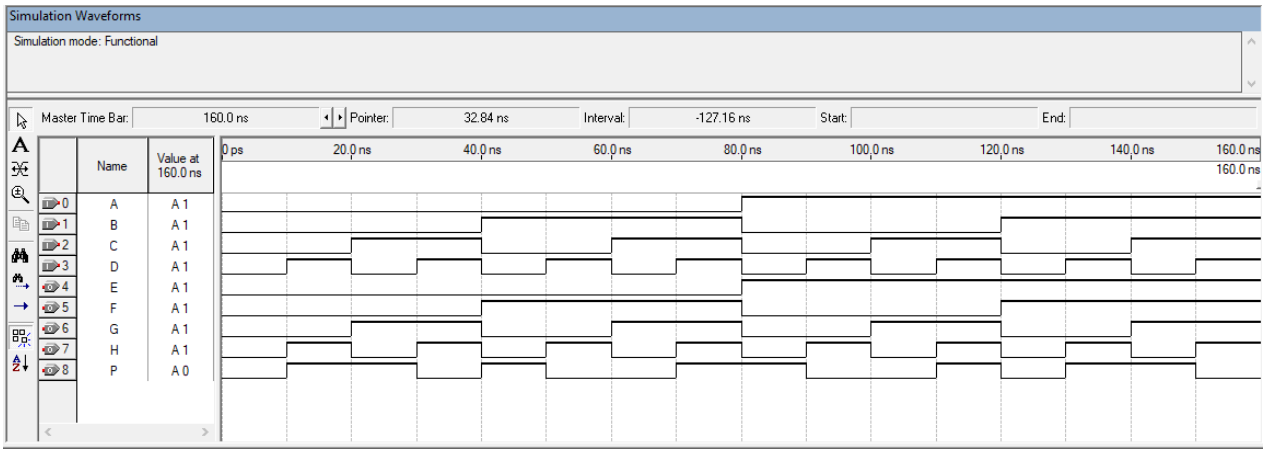
| AB \ CD | 00 | 01 | 11 | 10 |
|---|---|---|---|---|
| 00 | | 1 | 1 | |
| 01 | | 1 | 1 | |
| 11 | | 1 | 1 | |
| 10 | | 1 | 1 | |

H=D

# Circuit Diagram



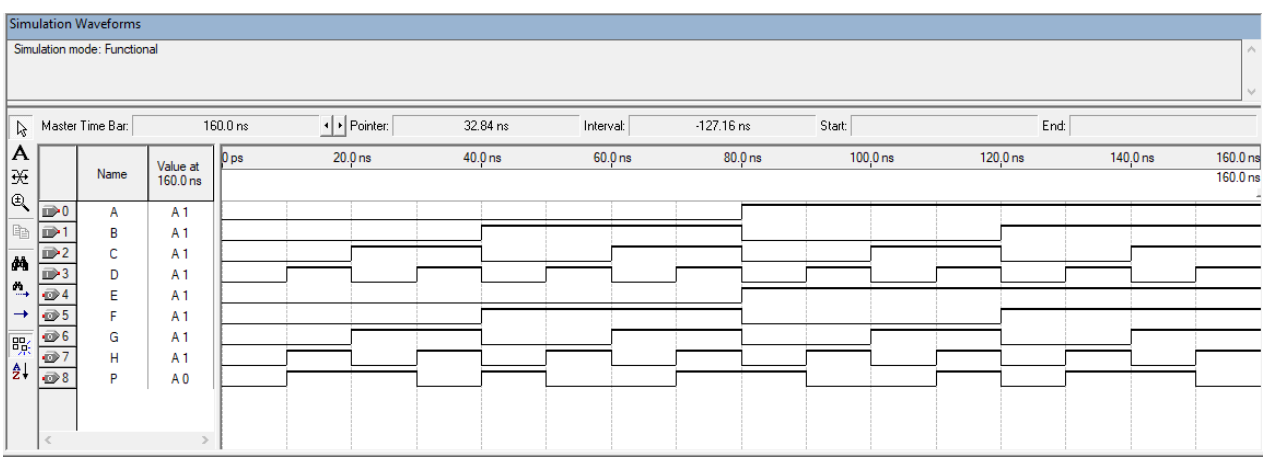

# Simulation



# Structural Verilog Code

```
module ex1 (input A,B,C,D, output P,E,F,G,H);
  wire w1,w2;
  xor g1(W1,A,B),
      g2(W2,C,D),
      g3(P,W1,W2);
  and g4(E,A),
      g5(F,B),
      g6(G,C),
      g7(H,D);
endmodule
```

# Simulation



# **Behavioral Verilog Code** 1.

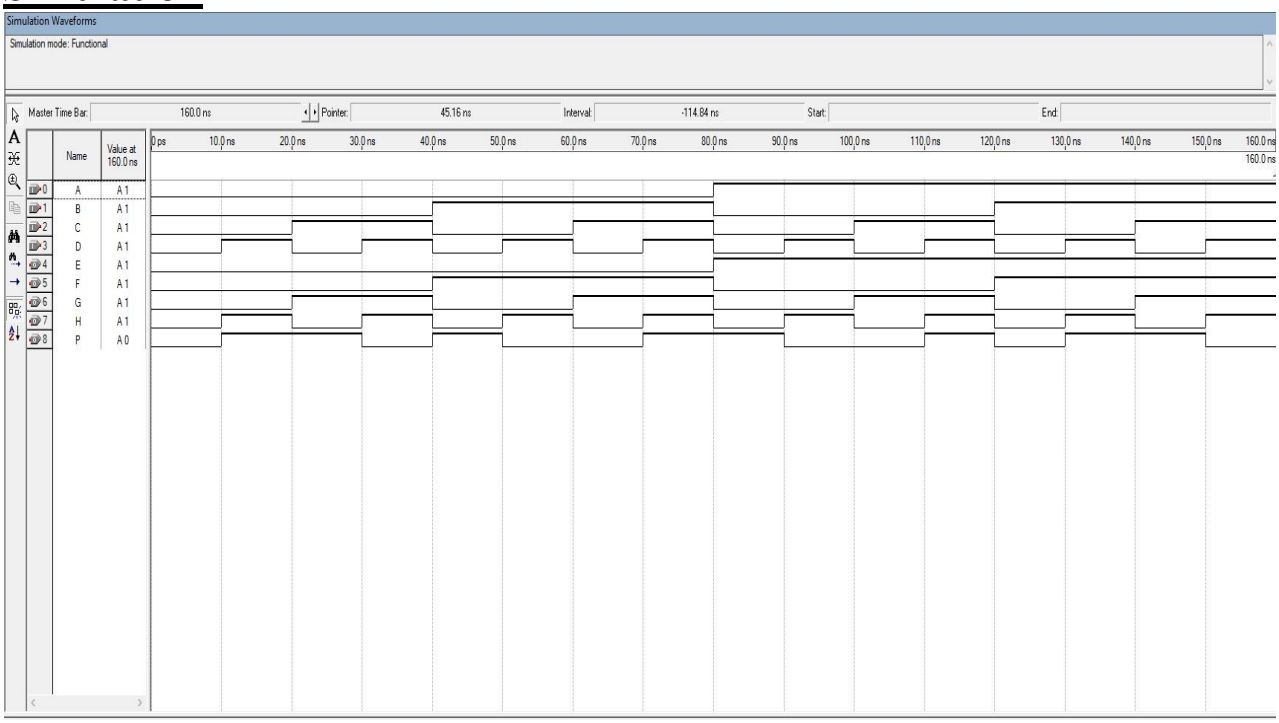
Procedural Code:

expt2.v | Compilation Report - Flow Summary | expt2.vw

```
module expt2 (input A,B,C,D,output reg P,E,F,G,H);
always@(A,B,C,D) begin
 P=0;
 if(~A&~B&~C&D)P=1;
 if(~A&~B&C&~D)P=1;
 if(~A&B&~C&~D)P=1;
 if(~A&B&C&D)P=1;
 if(A&~B&~C&~D)P=1;
 if(A&~B&C&D)P=1;
 if(A&B&~C&D)P=1;
 if(A&B&C&~D)P=1;
 if (A==0)
 E=0;
 else E=1;
 if (B==0)
 F=0;
 else F=1;
 if (C==0)
 G=0;
 else G=1;
 if (D==0)
 H=0;
 else H=1;

 end
 endmodule
```

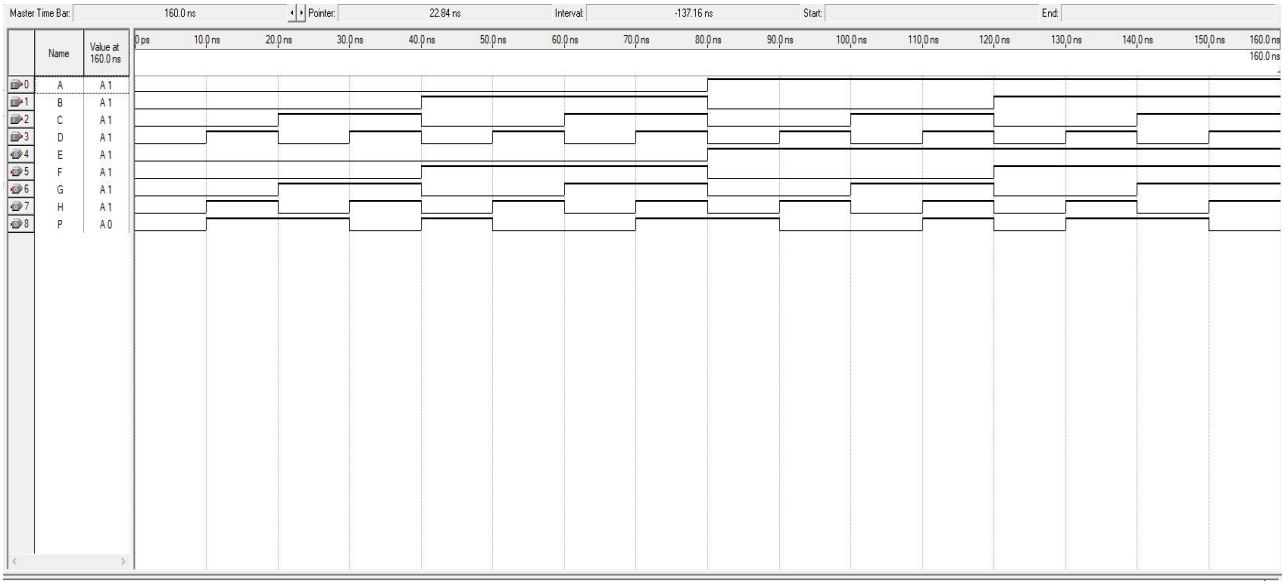
# Simulation




2. Continuous Code:

```
module ex1 (input A,B,C,D, output P,E,F,G,H);
assign P=(~A&~B&~C&D)|(~A&~B&C&~D)|(~A&B&~C&~D)|(~A&B&C&D)|(A&~B&~C&~D)|(A&~B&C&D)|(A&B&~C&D)|(A&B&C&~D);
assign E=A;
assign F=B;
assign G=C;
assign H=D;
endmodule
```

# Simulation


Master Time Bar: 160.0 ns
Pointer: 22.84 ns
Interval: -137.16 ns
Start:
End:
Name
Value at 160.0 ns
0 ps
10.0 ns
20.0 ns
30.0 ns
40.0 ns
50.0 ns
60.0 ns
70.0 ns
80.0 ns
90.0 ns
100.0 ns
110.0 ns
120.0 ns
130.0 ns
140.0 ns
150.0 ns
160.0 ns
0 A A 1
1 B A 1
2 C A 1
3 D A 1
4 E A 1
5 F A 1
6 G A 1
7 H A 1
8 P A 0