BINSHADH BASHEER B210517CS Q1

a. **8 x 2 PROM using F1(X, Y, Z) =∑m (1,4,6,7); F2(X, Y, Z) = ∑m (0,3,4,5,7);**

data flow

```
module prom1(input x,y,z,e, output f1, f2);
wire a0, a1, a2, a3, a4, a5, a6, a7;
decoder f3(x,y,z,e,a0, a1, a2, a3,a4, a5, a6, a7);
assign f1=a1|a4|a6|a7;
assign f2=a0|a3|a4|a5|a7;
endmodule
module decoder(input x,y,z, e, output a0, a1, a2, a3, a4, a5, a6, a7);
assign a0=e&~x&~y&~z;
assign a1=e&~x&~y&z;
assign a2=e&~x&y&~z;
assign a3=e&~x&y&z;
assign a4=e&x&~y&~z;
assign a5=e&x&~y&z;
assign a6=e&x&y&~z;
assign a7=e&x&y&z;
endmodule
```

gate_level

```
module prom1(input x,y,z,e, output f1, f2);
wire a0, a1, a2, a3, a4, a5, a6, a7;
decoder f3(x,y,z,e,a0, a1, a2, a3,a4, a5, a6, a7);
or (f1,a1,a4,a6,a7);
or (f2,a0,a3,a4,a5,a7);
endmodule
module decoder(input x,y,z, e, output a0, a1, a2, a3, a4, a5, a6, a7);
assign a0=e&~x&~y&~z;
assign a1=e&~x&~y&z;
assign a2=e&~x&y&~z;
assign a3=e&~x&y&z;
assign a4=e&x&~y&~z;
assign a5=e&x&~y&z;
assign a6=e&x&y&~z;
assign a7=e&x&y&z;

endmodule
```

behavioral

```
module prom1(input x,y,z,e, output reg f1, f2);
wire a0, a1, a2, a3, a4, a5, a6, a7;
always@(*)
begin
decoder f3 (x,y,z,e,a0, a1, a2, a3,a4, a5, a6, a7);
f1=a1|a4|a6|a7;
 f2=a0|a3|a4|a5|a7;
end
endmodule
module decoder(input x,y,z, e, output a0, a1, a2, a3, a4, a5, a6, a7);
always@(*)
begin
 a0=e&~x&~y&~z;
 a1=e&~x&~y&z;
 a2=e&~x&y&~z;
```

```
 a3=e&~x&y&z;
 a4=e&x&~y&~z;
 a5=e&x&~y&z;
 a6=e&x&y&~z;
 a7=e&x&y&z;
end
endmodule
```

test bench

```
module prom1_tb;
reg x,y,z,e;
wire f1, f2;
integer i;
prom1 f4(x,y,z,e,f1, f2);
initial
begin
e=1'b1;
for(i=0;i<8; i=i+1)
begin
{x,y,z}=i;
 #10;
end
end
endmodule
```

b. PAL(Programmable Array Logic) using F1(A,B,C,D)=∑ m(1,2,3,4,5,6,12,14,15), F2(A,B,C,D)=∑m(1,2,3,4,5,6,7,9,11,15), F3(A,B,C,D)=∑ m(1,3,4,5,9,10,11,13,15)

data flow

```
module q1b_data(
            input a,b,c,d,
            output f1,f2,f3
);
wire a0,b0,c0,d0;
wire [15:0]y;
assign a0 = ~a;
assign b0 = ~b;
assign c0 = ~c;
assign d0 = ~d;
assign y[1] = a0&b0&c0&d;
assign y[2] = a0&b0&c&d0;
assign y[3] = a0&b0&c&d;
assign y[4] = a0&b&c0&d0;
assign y[5] = a0&b&c0&d;
assign y[6] = a0&b&c&d0;
assign y[7] = a0&b&c&d;
assign y[9] = a&b0&c0&d;
assign y[10] = a&b0&c&d0;
assign y[11] = a&b0&c&d;
assign y[12] = a&b&c0&d0;
assign y[13] = a&b&c0&d;
assign y[14] = a&b&c&d0;
assign y[15] = a&b&c&d;
assign f1 =  y[1];
assign f2 =  y[1] | y[2] | y[3] | y[4] | y[5] | y[6] | y[7]  | y[9]  | y[11] | y[15] ;
assign f3 =  y[1] | y[2] | y[3] | y[4] | y[5] | y[6] | y[12] | y[14] | y[15];
endmodule
```

test bench

```
module tb_data;
```

```
reg a,b,c,d;
wire f1,f2,f3;
integer i;
q1b_data h(a,b,c,d,f1,f2,f3);
initial begin
for(i=0;i<16;i=i+1)
        begin
        {a,b,c,d} =i;
        #20
end
end
endmodule
```

gate_level

```
module pal(
output f1,f2,f3,
input a,b,c,d);
wire y1,y2,y3,y4,y5,y6,y7,y8,y9,y10,y11,y12;
and(y1,b,~d);
and(y2,~a,~c,d);
and(y3,~a,~b,c);
and(y4,a,b,c);
and(y5,~a,b);
and(y6,~b,d);
and(y7,c,d);
and(y8,~a,c);
and(y9,~b,d);
and(y10,~a,b,~c);
and(y11,a,~b,c);
and(y12,a,d);
or(f1,y1,y2,y3,y4);
or(f2,y5,y6,y7,y8);
or(f3,y9,y10,y11,y12);
endmodule
```

TEST BENCH

```
module pal_tb;
wire f1,f2,f3;
reg a,b,c,d;
integer i;
pal g(.f1(f1),.f2(f2),.f3(f3),.a(a),.b(b),.c(c),.d(d));
initial
begin
for(i=0;i<16;i=i+1)
begin
{a,b,c,d}=i;
#20;
end
#20 $stop;
end
endmodule
```

behavioral

```
module q1b_data(
        input a,b,c,d,
        output reg  f1,f2,f3
);
reg a0,b0,c0,d0;
reg [15:0]y;
```

```
always@(*)begin
 a0 = ~a;
 b0 = ~b;
 c0 = ~c;
 d0 = ~d;
 y[1] = a0&b0&c0&d;
 y[2] = a0&b0&c&d0;
 y[3] = a0&b0&c&d;
 y[4] = a0&b&c0&d0;
 y[5] = a0&b&c0&d;
 y[6] = a0&b&c&d0;
 y[7] = a0&b&c&d;
 y[9] = a&b0&c0&d;
 y[10] = a&b0&c&d0;
 y[11] = a&b0&c&d;
 y[12] = a&b&c0&d0;
 y[13] = a&b&c0&d;
 y[14] = a&b&c&d0;
 y[15] = a&b&c&d;
 f1 =  y[1];
 f2 =  y[1] | y[2] | y[3] | y[4] | y[5] | y[6] | y[7]  | y[9]  | y[11] | y[15] ;
 f3 =  y[1] | y[2] | y[3] | y[4] | y[5] | y[6] | y[12] | y[14] | y[15];
end
endmodule
```

test bench

```
module tb_data;
reg a,b,c,d;
wire f1,f2,f3;
integer i;
q1b_data h(a,b,c,d,f1,f2,f3);
initial begin
for(i=0;i<16;i=i+1)
        begin
        {a,b,c,d} =i;
        #20
end
end
endmodule
```

c. PLA with minimum additional logic using F1(A, B, C, D) =∑ m (0,3,4,5,6,12,14,15)
F2(A, B, C, D) =∑m (1,3,4,5,6,7,9,11,13,15)

data flow

```
module pla(
output f1,f2,
input a,b,c,d);
wire y1,y2,y3,y4,y5,y6,y7;
assign y1=b&~d,
y2=~a&b&~c,
y3=~a&~c&~d,
y4=a&b&c,
y5=~a&~b&c&d,
y6=d,
y7=~a&b,
f1=y1|y2|y3|y4|y5,
f2=y6|y7;
endmodule
```

gate_level

```
module pla(
output f1,f2,
input a,b,c,d);
wire y1,y2,y3,y4,y5,y7;
and(y1,b,~d);
and(y2,~a,b,~c);
and(y3,~a,~c,~d);
and(y4,a,b,c);
and(y5,~a,~b,c,d);
and(y7,~a,b);
or(f1,y1,y2,y3,y4,y5);
or(f2,d,y7);
endmodule
```

behavioral

```
module pla(
output reg f1,f2,
input a,b,c,d);
reg y1,y2,y3,y4,y5,y6,y7;
always@(*)begin
y1=b&~d,
y2=~a&b&~c,
y3=~a&~c&~d,
y4=a&b&c,
y5=~a&~b&c&d,
y6=d,
y7=~a&b,
f1=y1|y2|y3|y4|y5,
f2=y6|y7;
end
endmodule
```

test bench

```
module pla_tb;
wire f1,f2;
reg a,b,c,d;
integer i;
pal g(.f1(f1),.f2(f2),.a(a),.b(b),.c(c),.d(d));
initial
begin
for(i=0;i<16;i=i+1)
begin
{a,b,c,d}=i;
#20;
end
#20 $stop;
end
endmodule
```