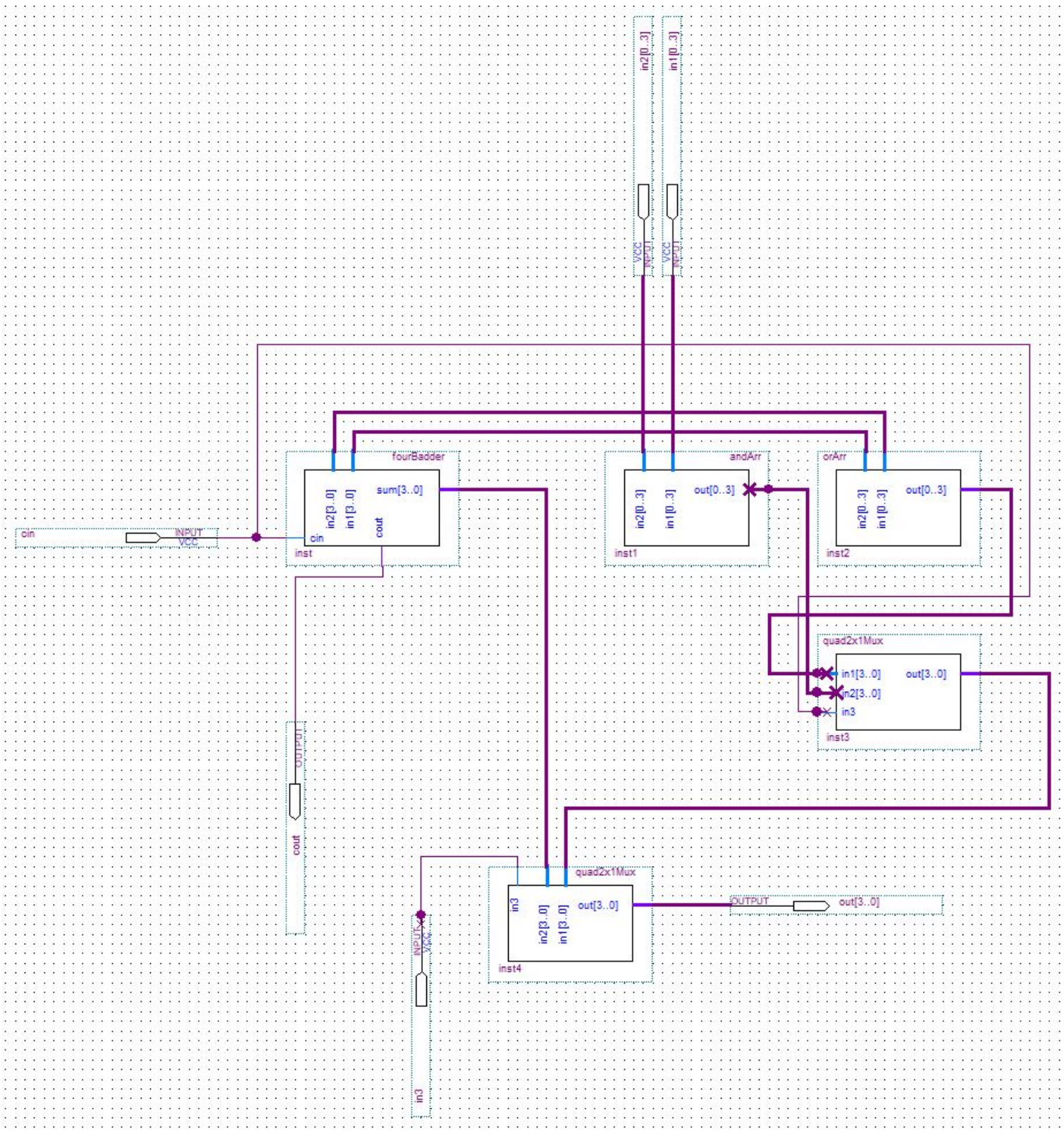
ENCS234
HDL - Assignment

- **Instructor : Dr. Abdallatif Abuissa**
- **Student's Name : Bara Adnan**
- **Student's ID : 1161357**
- **Date : 01/02/2018**

1) System design:



2) Code:

A) 1 bit adder:

```
1  module oneBadder ( in1, in2, cin, cout, sum );
2
3    input in1, in2, cin;
4    output cout, sum;
5
6    assign sum = in1 ^ in2 ^ cin;
7    assign cout = ( in1 && in2 ) || ( in1 && cin ) || ( in2 && cin );
8
9    endmodule
```

B) 4 bit adder:

```
1  module fourBadder ( in1, in2, cin, sum, cout );
2
3    input cin;
4    input [ 3:0 ] in1, in2;
5    output cout;
6    output [ 3:0 ] sum;
7    wire w1, w2, w3, w4;
8    wire ww1, ww2, ww3, ww4;
9
10   assign ww1 = in2[ 0 ] ^ cin;
11   assign ww2 = in2[ 1 ] ^ cin;
12   assign ww3 = in2[ 2 ] ^ cin;
13   assign ww4 = in2[ 3 ] ^ cin;
14
15   oneBadder ( in1[ 0 ], ww1, cin, w1, sum[ 0 ] );
16   oneBadder ( in1[ 1 ], ww2, w1, w2, sum[ 1 ] );
17   oneBadder ( in1[ 2 ], ww3, w2, w3, sum[ 2 ] );
18   oneBadder ( in1[ 3 ], ww4, w3, w4, sum[ 3 ] );
19
20   assign cout = w4 ^ w3;
21
22   endmodule
```

C) And array:

```
1  module andArr ( in1, in2, out );
2
3   input [ 0:3 ] in1, in2;
4   output [ 0:3 ] out;
5
6   assign out[ 0 ] = in1[ 0 ] && in2[ 0 ];
7   assign out[ 1 ] = in1[ 1 ] && in2[ 1 ];
8   assign out[ 2 ] = in1[ 2 ] && in2[ 2 ];
9   assign out[ 3 ] = in1[ 3 ] && in2[ 3 ];
10
11  endmodule
```

D) Or array:

```
1  module orArr ( in1, in2, out );
2
3   input [ 0:3 ] in1, in2;
4   output [ 0:3 ] out;
5
6   assign out[ 0 ] = in1[ 0 ] || in2[ 0 ];
7   assign out[ 1 ] = in1[ 1 ] || in2[ 1 ];
8   assign out[ 2 ] = in1[ 2 ] || in2[ 2 ];
9   assign out[ 3 ] = in1[ 3 ] || in2[ 3 ];
10
11  endmodule
```

E) 2x1 Mux:

```
1  module mux2x1 ( in1, in2, in3, out );
2
3   input in1, in2, in3;
4   output out;
5   wire w1, w2;
6
7   assign w1 = ( in1 && !in3 ), w2 = ( in2 && in3 ), out = w1 || w2;
8
9   endmodule
```

F) Quad Mux:

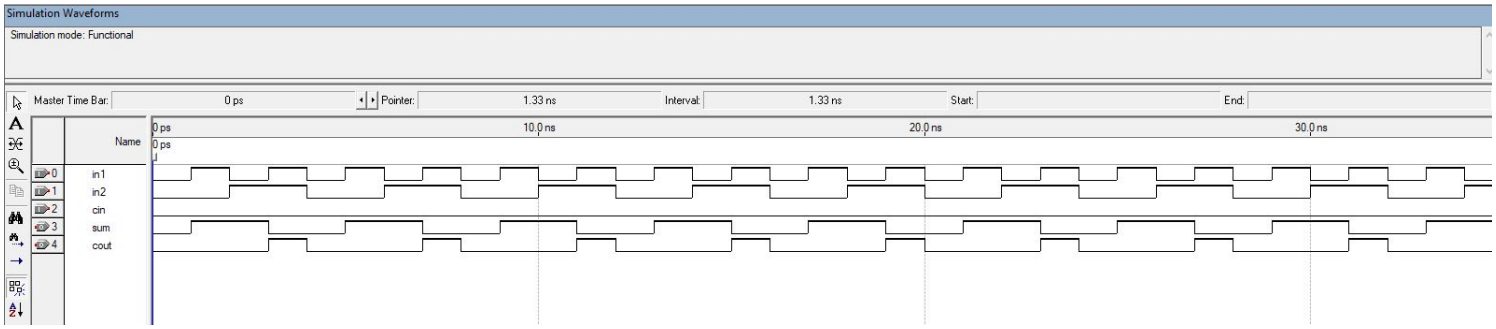
```
1  module quad2x1Mux ( in1, in2, in3, out );
2
3   input [ 3:0 ] in1, in2;
4   input in3;
5   output [ 3:0 ] out;
6
7   mux2x1 ( in1[ 0 ], in2[ 0 ], in3, out[ 0 ] );
8   mux2x1 ( in1[ 1 ], in2[ 1 ], in3, out[ 1 ] );
9   mux2x1 ( in1[ 2 ], in2[ 2 ], in3, out[ 2 ] );
10  mux2x1 ( in1[ 3 ], in2[ 3 ], in3, out[ 3 ] );
11
12  endmodule
```

G) Full system:

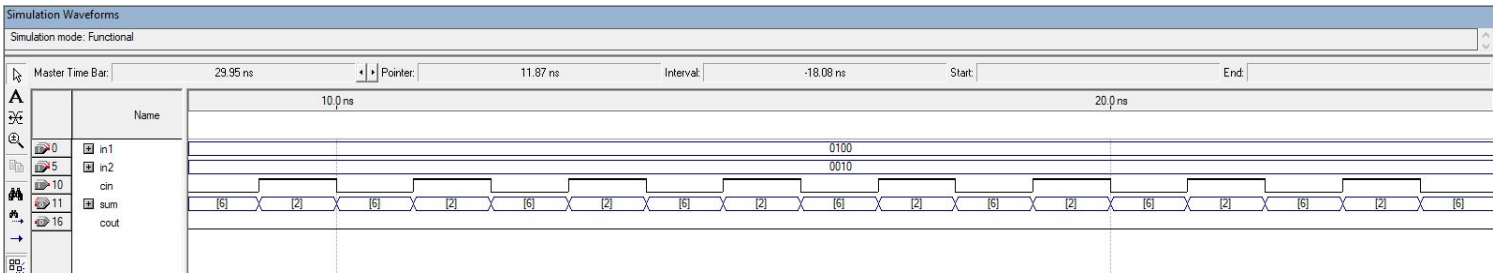
```
1  module kernel ( in1, in2, BoF, flow, s0, s1 );
2
3   input [ 3:0 ] in1, in2;
4   input s0, s1;
5   output [ 3:0 ] flow;
6   output BoF;
7   wire [ 3:0 ] w1, w2, w3, w4;
8
9   orArr ( in1, in2, w1 );
10  andArr ( in1, in2, w2 );
11  fourBadder ( a, in2, s0, w, overflow );
12  quad2x1Mux ( in1, in2, s0, w4 );
13  quad2x1Mux ( w3, w4, s1, flow );
14
15  endmodule
16
```

3) Simulations:

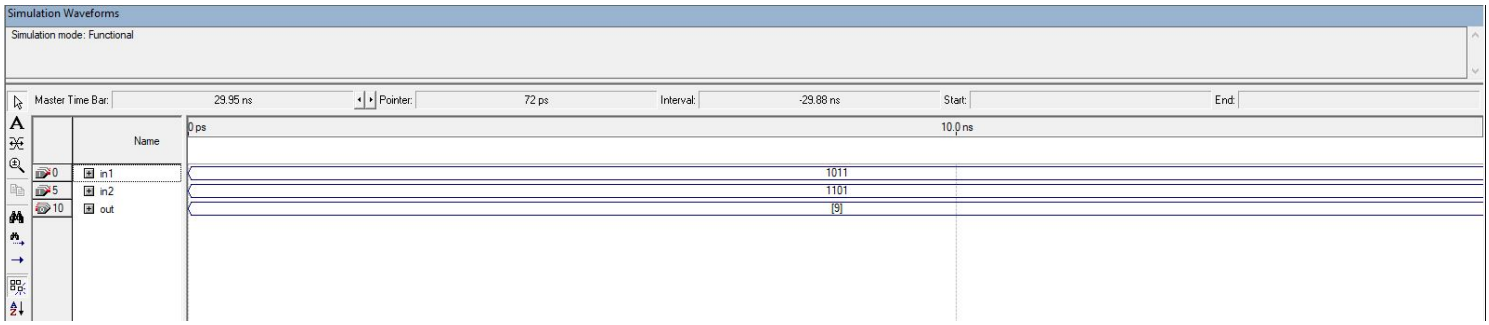
A) 1 bit adder:



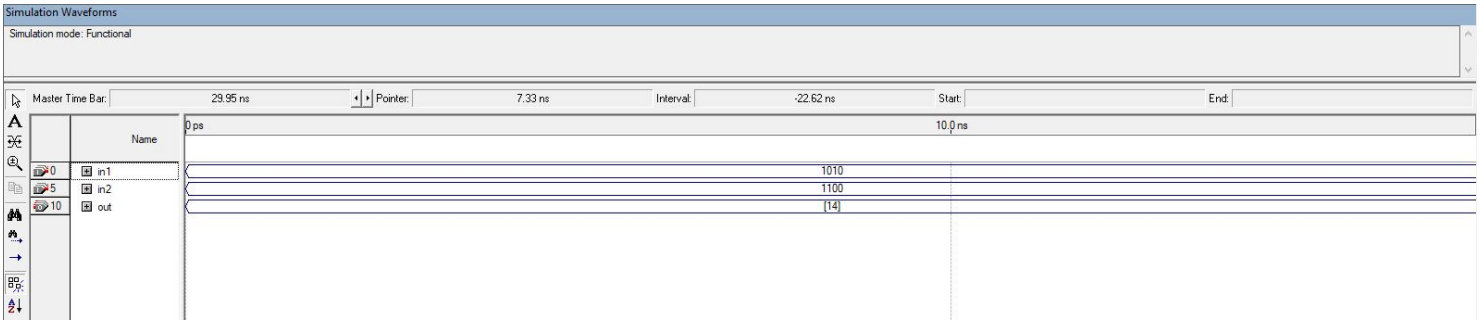
B) 4 bit adder:



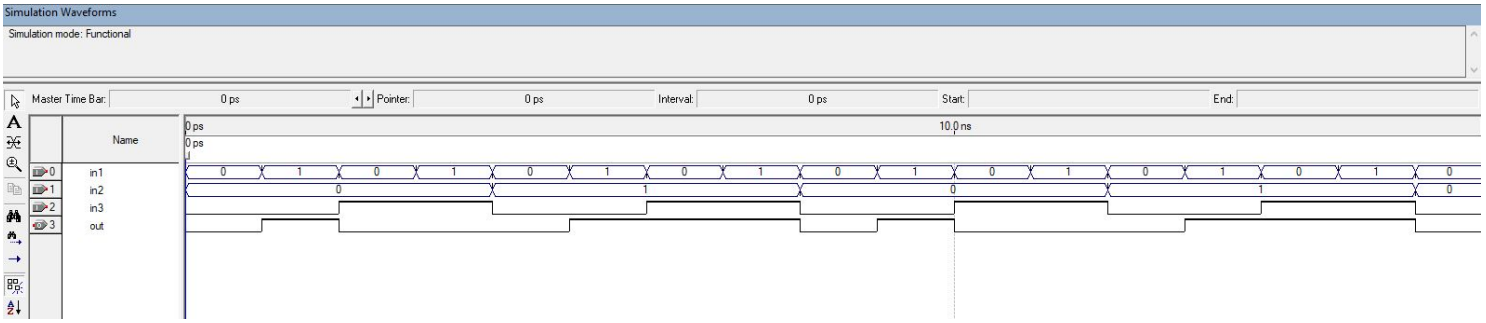
C) And array:



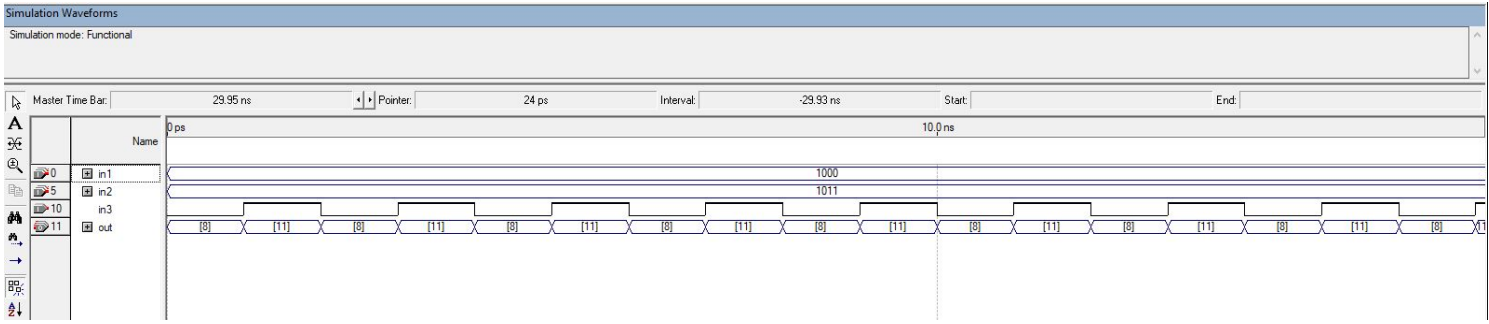
D) Or array:



E) 2x1 Mux:



F) Quad Mux:



G) Full system: