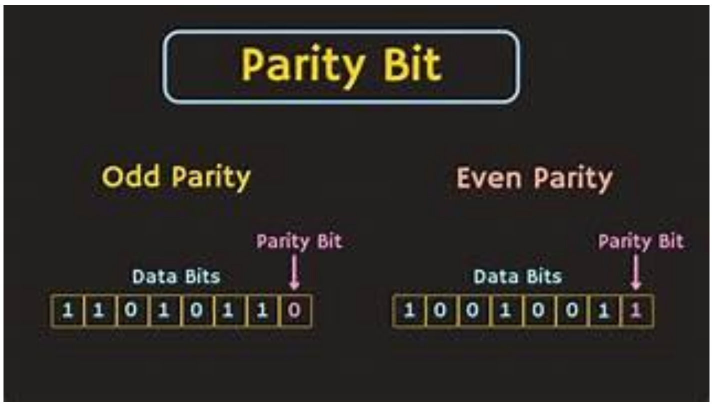
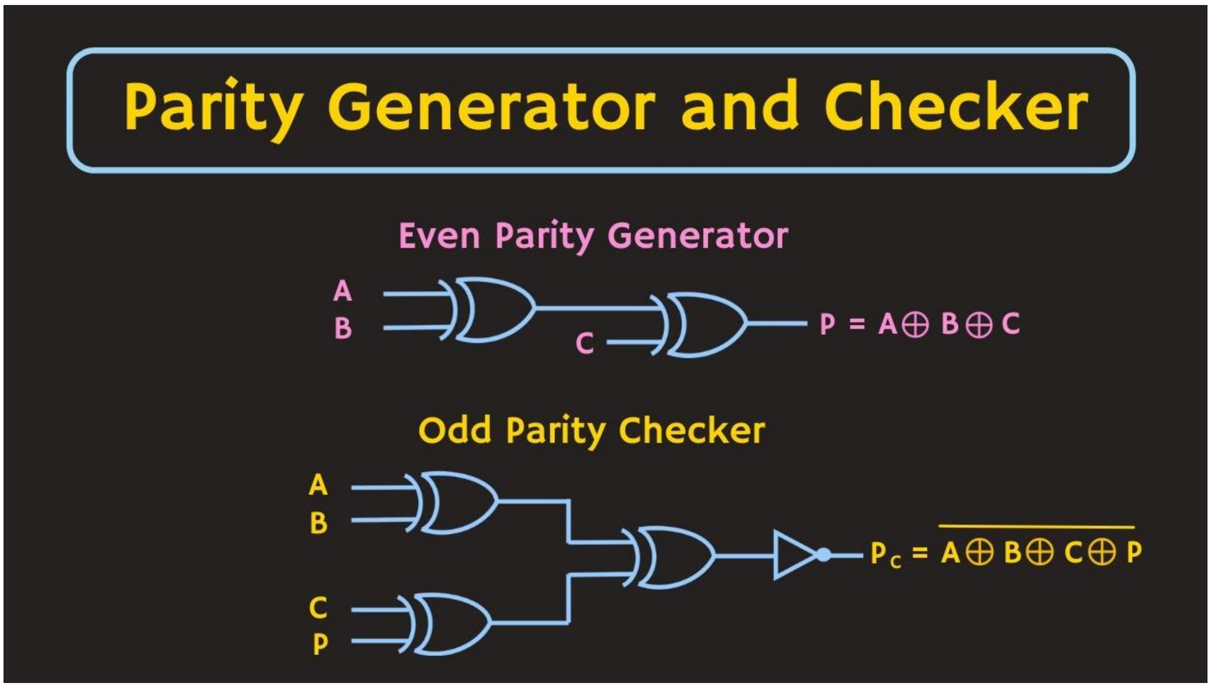
PARITY CHECKER (EVEN OR ODD)

Now imagine a scenario. You want to send a stream of digital bits. Let’s say that this stream has n bits. You are slightly concerned with errors entering your message. So you say, “Hey! I need to use some kind of error detection mechanism”. So you decide to use ‘parity bits.’ Now you have two choices. You can either use the ***even parity mechanism***. Or you can use the ***odd parity mechanism***.



Even parity mechanism: The target is to make the total number of 1s even. For example, if you have a message signal “010”, you can clearly see that it has just one 1. So we add a parity bit to make it two 1s. Now the number of 1s is even.

Odd parity mechanism: Here, the target is the make the total number of 1s odd. For example, consider the same message signal from above. “010”. The parity bit here will be….complete the sentence. ***0!*** That's right. There's already an odd number of 1s in the message signal.

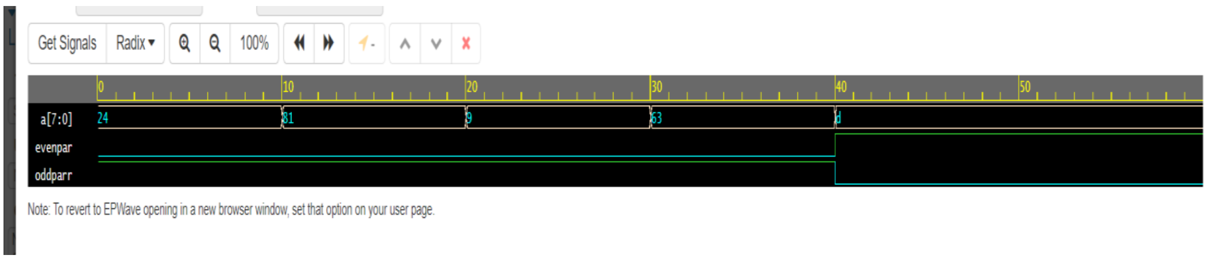

Parity Generator and Checker
Even Parity Generator
A
B
C
P = A⊕ B⊕ C
Odd Parity Checker
A
B
C
P
$P_C = \overline{A \oplus B \oplus C \oplus P}$

RTL CODE:

```
module vl(input[7:0]a,output evenpar,output oddparr);
  assign oddparr=~^(a);
  assign evenpar=^(a);
endmodule
```

TESTBENCH:

```
module test;
  reg [7:0] a;
  wire evenpar,oddparr;
  vl a1(a,evenpar,oddparr);
  initial begin
    $dumpfile("dump.vcd");
    $dumpvars(1);
  end
  initial begin
    repeat(5) begin
      a=$random;
      #10;
    end
  end
  initial begin
    #60 $finish();
  end
endmodule
```



Note: To revert to EPWave opening in a new browser window, set that option on your user page.