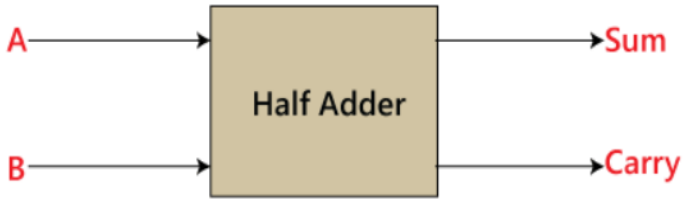
HALF ADDER:

The Half-Adder is a basic building block of adding two numbers as two inputs and produce out two outputs. The adder is used to perform OR operation of two single bit binary numbers. The **A** and **B** bits are two input states, and **'carry**' and **'sum** 'are two output states of the half adder.

Block diagram



Truth Table

Inputs		Outputs	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

EXPLANATION:

A half adder is used to add two single-digit binary numbers and results into a two-digit output. It is named as such because putting two half adders together with the use of an OR gate results in a full adder. In other words, it only does. half the work of a full adler. The adder works by combining the operations of basic logic gates, with the simplest form using only a XOR and an AND gate. This can also be converted into a circuit that only has AND, OR and NOT gates. This is especially useful since these three simpler logic gate ICs (integrated circuits) are more common. and available than the XOR IC, though this might result in a bigger circuit since three different chips are used instead of just one

ADVANTAGES:

- By using an inveter, it can be converted to the half subtractor
- Simple in design compared to full adder

DISADVANTAGES:

- Half adders have no scope of adding the carry bit resulting from the addition of previous bits.
- The real-time scenarios involve adding the multiple numbers of bits which cannot be accomplished using half adder.
- It is not suitable for cascading for multi-bit additions.

RTL CODE:

DATA FLOW REPRESENTATION:

```
module half_adder(sum,carry,a,b);
  output sum,carry;
  assign sum=a^b;
  assign carry=a&b;
endmodule
```

STRUCTURAL REPRESENTATION:

```
module half_adder(sum,carry,a,b);
 output sum,carry;
 input a,b;
  xor gate1(sum,a,b);
  and g2(carry,a,b);
endmodule
```

BEHAVIORAL REPRESENTATION:

```
module half_adder(sum,carry,a,b);
 output sum,carry;
 input a,b;
  always @ (*)
    begin
     sum=a^b;
    carry=a&b;
    end
 endmodule
```

TEST BENCH:

```
module test_best;
  reg a,b;
  wire sum,carry;
  // instantating design module

  half_adder a1(sum,carry,a,b);
  initial
    begin
      $dumpfile("dump.vcd");
      $dumpvars(1);
    end
  initial
    begin
      a=0;b=0;
```

```
    end
  initial
    begin
      #10 a=0;b=1;
      #10 a=1;b=0;
      #10 a=1;b=1;
    end
  initial
    begin
      $monitor($time,"a=%b,b=%b,carry=%b,sum=%d",a,b,sum,carry);
      #60 $finish();
    end
endmodule
```