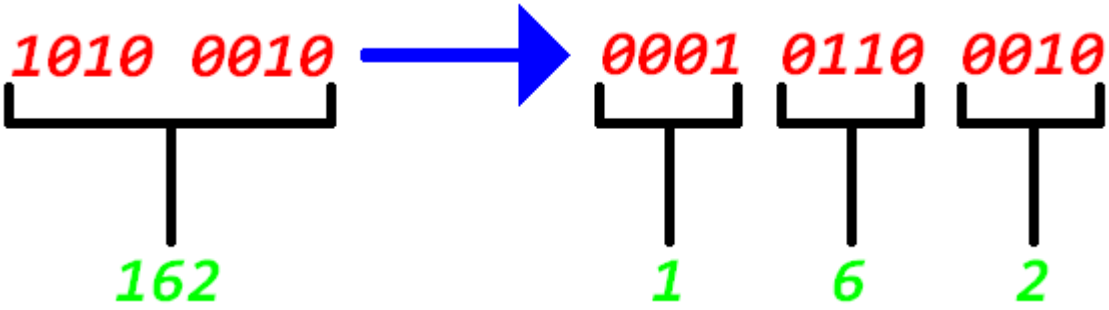
Home

Binary to BCD Conversion Algorithm



Purpose:

Conversion of a binary number into separate binary numbers representing digits of the decimal number.
(this example is for 8-bits, other sizes follow the same pattern)

Algorithm:

1. If any column (100's, 10's, 1's, etc.) is 5 or greater, add 3 to that column.
2. Shift all #'s to the left 1 position.
3. If 8 shifts have been performed, it's done! Evaluate each column for the BCD values.
4. Go to step 1.

Psuedo-Code:

```
for(i=0; i<8; i++) {
        //check all columns for >= 5
        for each column {
                if (column >= 5)
                        column += 3;

        //shift all binary digits left 1
        Hundreds <<= 1;
        Hundreds[0] = Tens[3];
        Tens << = 1;
        Tens[0] = Ones[3];
        Ones << = 1;
        Ones[0] = Binary[7];
        Binary <<= 1;
}
```

Algorithm In Action:

100's	10's	1's	Binary	Operation
			1010 0010	
		1	010 0010	<< #1
		10	10 0010	<< #2
		101	0 0010	<< #3
		1000		add 3
	1	0000	0010	<< #4
	10	0000	010	<< #5
	100	0000	10	<< #6
	1000	0001	0	<< #7
	1011			add 3
1	0110	0010		<< #8

162

1 6 2

BCD Conversion in Hardware:

Verilog:

```
module BCD(
    input [7:0] binary,
    output reg [3:0] Hundreds,
    output reg [3:0] Tens,
    output reg [3:0] Ones
    );



   integer i;
   always @(binary)
   begin
      //set 100's, 10's, and 1's to 0
      Hundreds = 4'd0;
      Tens = 4'd0;
      Ones = 4'd0;

      for (i=7; i>=0; i=i-1)
      begin
         //add 3 to columns >= 5
         if (Hundreds >= 5)
            Hundreds = Hundreds + 3;
         if (Tens >= 5)
            Tens = Tens + 3;
         if (Ones >= 5)
            Ones = Ones + 3;

         //shift left one
         Hundreds = Hundreds << 1;
         Hundreds[0] = Tens[3];
         Tens = Tens << 1;
         Tens[0] = Ones[3];
         Ones = Ones << 1;
         Ones[0] = binary[i];
      end
   end

endmodule
```