Binary-to-BCD Converter

Discussion D3.1

Shift and Add-3 Algorithm

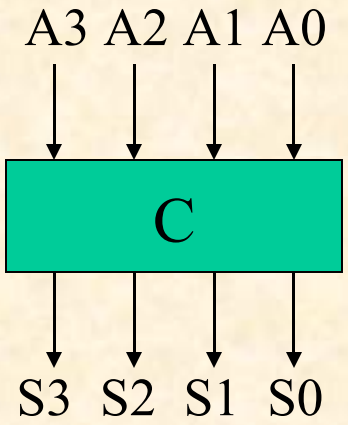
1. Shift the binary number left one bit.
2. If 8 shifts have taken place, the BCD number is in the *Hundreds*, *Tens*, and *Units* column.
3. If the binary value in any of the BCD columns is 5 or greater, add 3 to that value in that BCD column.
4. Go to 1.

Operation	Hundreds	Tens	Units	Binary	
HEX				F	F
Start				1 1 1 1	1 1 1 1

Steps to convert an 8-bit binary number to BCD

Operation	Hundreds	Tens	Units	Binary	
HEX				F	F
Start				1 1 1 1	1 1 1 1
Shift 1			1	1 1 1 1	1 1 1
Shift 2			1 1	1 1 1 1	1 1
Shift 3			1 1 1	1 1 1 1	1
Add 3			1 0 1 0	1 1 1 1	1
Shift 4		1	0 1 0 1	1 1 1 1	
Add 3		1	1 0 0 0	1 1 1 1	
Shift 5		1 1	0 0 0 1	1 1 1	
Shift 6		1 1 0	0 0 1 1	1 1	
Add 3		1 0 0 1	0 0 1 1	1 1	
Shift 7	1	0 0 1 0	0 1 1 1	1	
Add 3	1	0 0 1 0	1 0 1 0	1	
Shift 8	1 0	0 1 0 1	0 1 0 1		
BCD	2	5	5		

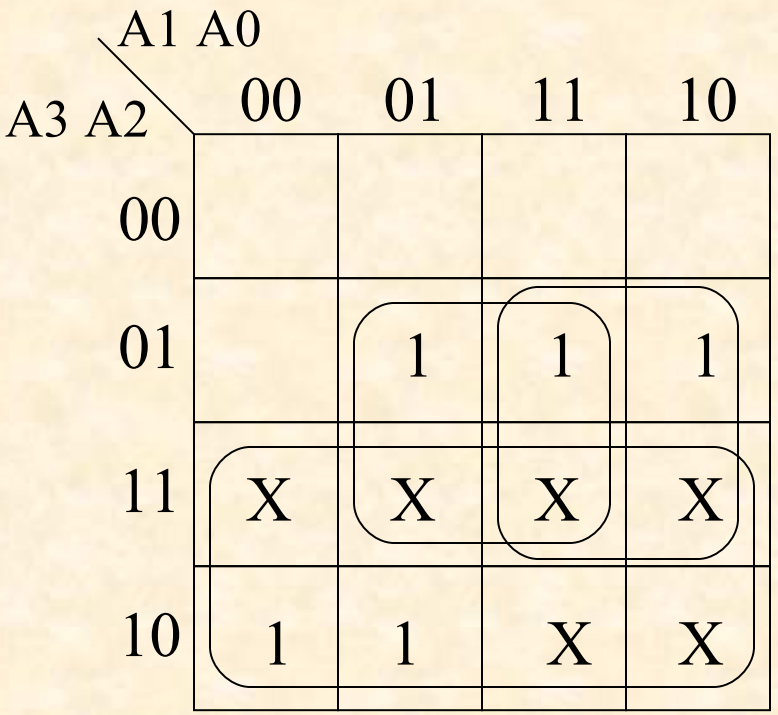
Truth table for Add-3 Module



A3	A2	A1	A0	S3	S2	S1	S0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	0	1	0	0
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	X	X	X	X
1	0	1	1	X	X	X	X
1	1	0	0	X	X	X	X
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X

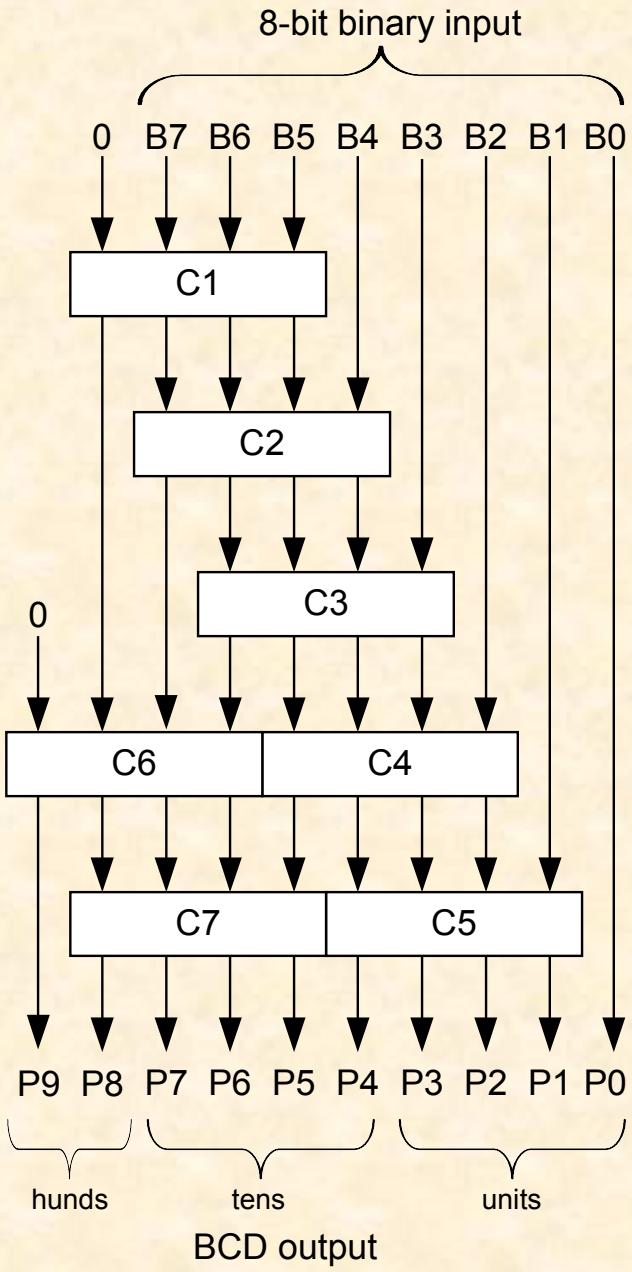
K-Map for S3

A3	A2	A1	A0	S3	S2	S1	S0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	0	1	0	0
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	X	X	X	X
1	0	1	1	X	X	X	X
1	1	0	0	X	X	X	X
1	1	0	1	X	X	X	X
1	1	1	0	X	X	X	X
1	1	1	1	X	X	X	X



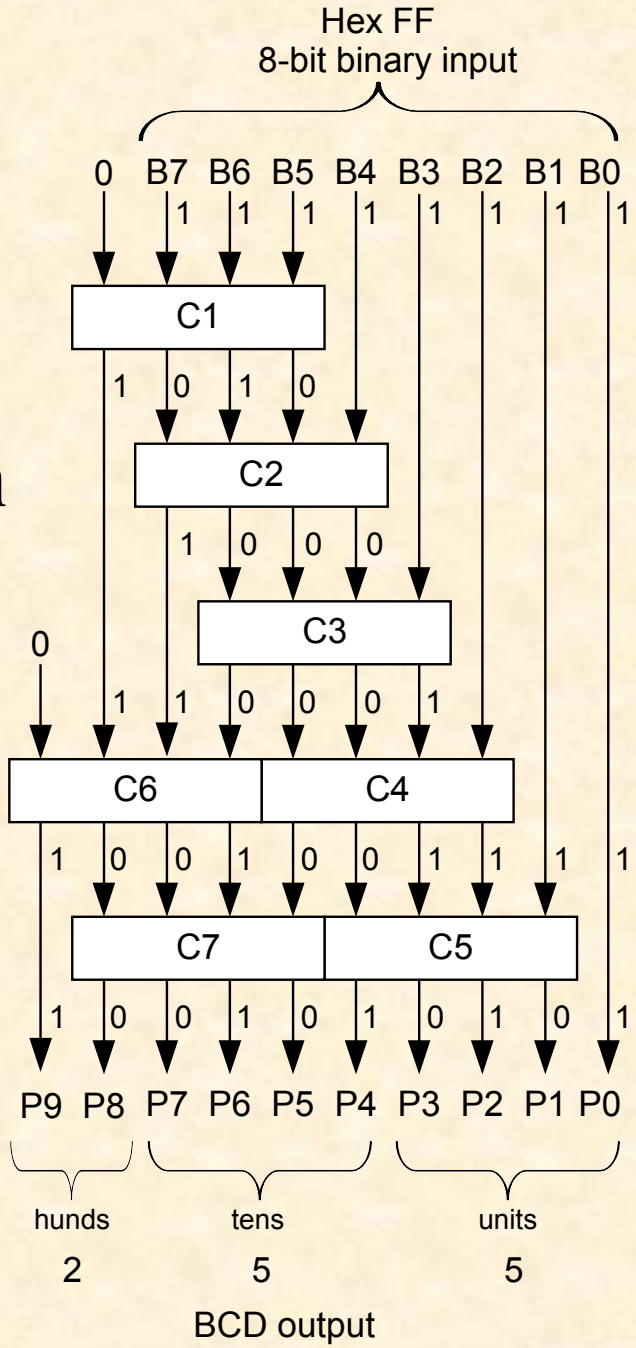
S3 = A3
+ A2 * A0
+ A2 * A1

Binary-to-BCD Converter



Binary-to-BCD Converter

Structural Solution



Steps to convert a 6-bit binary number to BCD

1. Clear all bits of *z* to zero
2. Shift *B* left 3 bits

   $z[8:3] = B[5:0];$
3. Do 3 times

   if *Units* >4 then add 3 to *Units*

   (note: *Units* = $z[9:6]$)

   Shift *z* left 1 bit
4. *Tens* = $P[6:4] = z[12:10]$

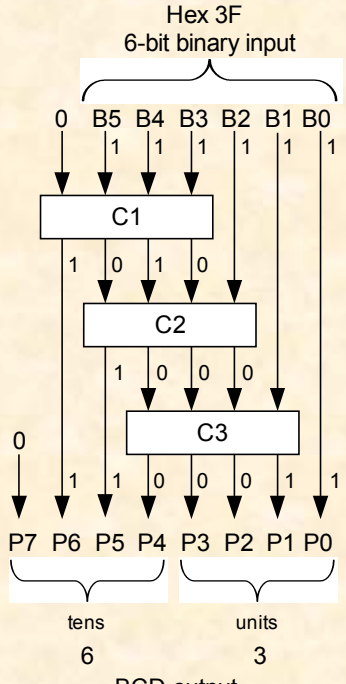
   Units = $P[3:0] = z[9:6]$

Operation	Tens	Units	Binary
B			5 4 3 2 1 0
HEX			3 F
Start			1 1 1 1 1 1
Shift 1		1	1 1 1 1 1
Shift 2		1 1	1 1 1 1
Shift 3		1 1 1	1 1 1
Add 3		1 0 1 0	1 1 1
Shift 4	1	0 1 0 1	1 1
Add 3	1	1 0 0 0	1 1
Shift 5	1 1	0 0 0 1	1
Shift 6	1 1 0	0 0 1 1	
BCD	6	3	
P	7 4	3 0	
z	13 10	9 6	5 0

binbcd6.vhd

Operation	Tens	Units	Binary
B			5 4 3 2 1 0
HEX			3 F
Start			1 1 1 1 1 1
Shift 1		1	1 1 1 1 1
Shift 2		1 1	1 1 1 1
Shift 3		1 1 1	1 1 1
Add 3		1 0 1 0	1 1 1
Shift 4	1	0 1 0 1	1 1
Add 3	1	1 0 0 0	1 1
Shift 5	1 1	0 0 0 1	1
Shift 6	1 1 0	0 0 1 1	
BCD	6	3	
P	7 4	3 0	
z	13 10	9 6	5 0

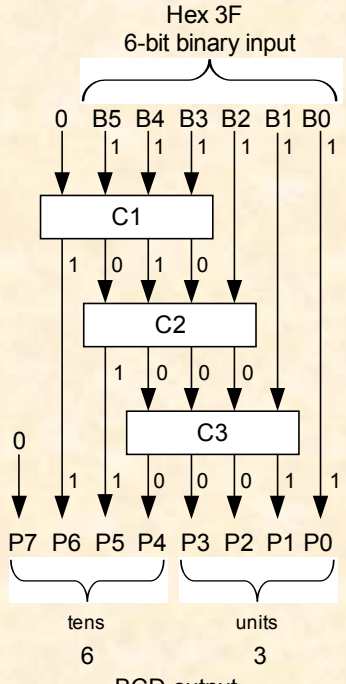
```
-- Title: Binary-to-BCD Converter
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity binbcd6 is
    port (
        B: in STD_LOGIC_VECTOR (5 downto 0);
        P: out STD_LOGIC_VECTOR (6 downto 0)
    );
end binbcd6;


architecture binbcd6_arch of binbcd6 is
begin
  bcd1: process(B)
  variable z: STD_LOGIC_VECTOR (12 downto 0);
```

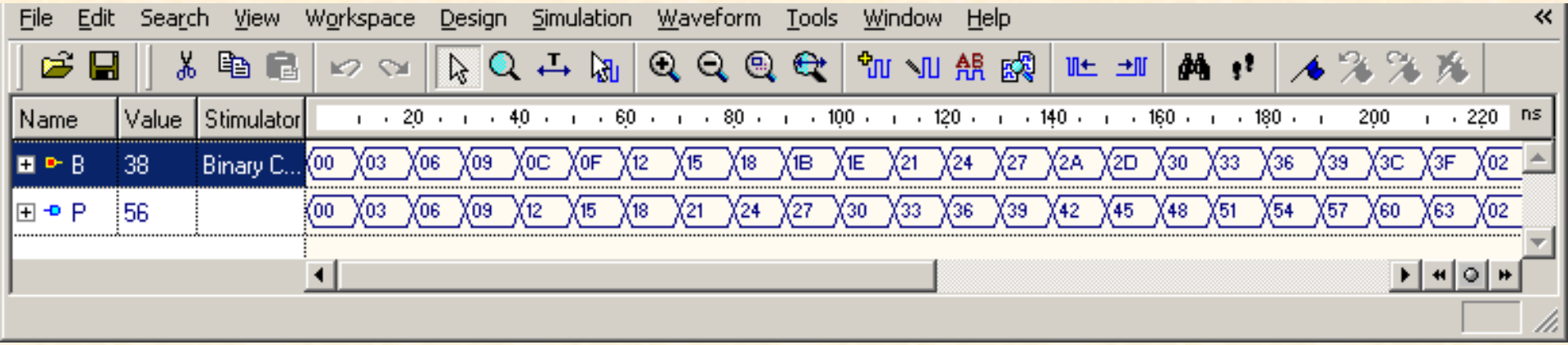


binbcd6.vhd (cont.)

```
  begin
       for i in 0 to 12 loop
          z(i) := '0';
       end loop;
       z(8 downto 3) := B;

       for i in 0 to 2 loop
          if z(9 downto 6) > 4 then
             z(9 downto 6) := z(9 downto 6) + 3;
          end if;
          z(12 downto 1) := z(11 downto 0);
       end loop;

       P <= z(12 downto 6);
  end process bcd1;
end binbcd6_arch;
```

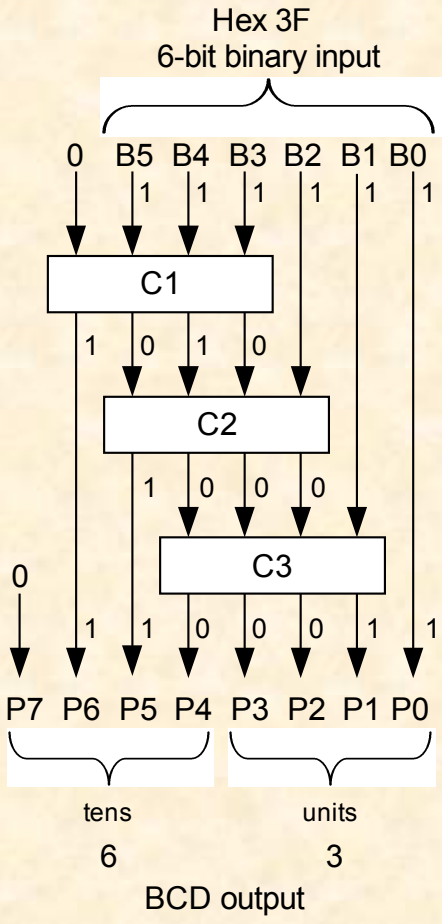
Operation	Tens	Units	Binary
B			5 4 3 2 1 0
HEX			3 F
Start			1 1 1 1 1 1
Shift 1		1	1 1 1 1 1
Shift 2		1 1	1 1 1 1
Shift 3		1 1 1	1 1 1
Add 3		1 0 1 0	1 1 1
Shift 4	1	0 1 0 1	1 1
Add 3	1	1 0 0 0	1 1
Shift 5	1 1	0 0 0 1	1
Shift 6	1 1 0	0 0 1 1	
BCD	6	3	
P	7 4	3 0	
z	13 10	9 6	5 0





binbcd6.vhd

Operation	Tens	Units	Binary
B			5 4 3 2 1 0
HEX			3 F
Start			1 1 1 1 1 1
Shift 1		1	1 1 1 1 1
Shift 2		1 1	1 1 1 1
Shift 3		1 1 1	1 1 1
Add 3		1 0 1 0	1 1 1
Shift 4	1	0 1 0 1	1 1
Add 3	1	1 0 0 0	1 1
Shift 5	1 1	0 0 0 1	1
Shift 6	1 1 0	0 0 1 1	
BCD	6	3	
P	7 4	3 0	
z	13 10	9 6	5 0



8-Bit Binary-to-BCD Converter

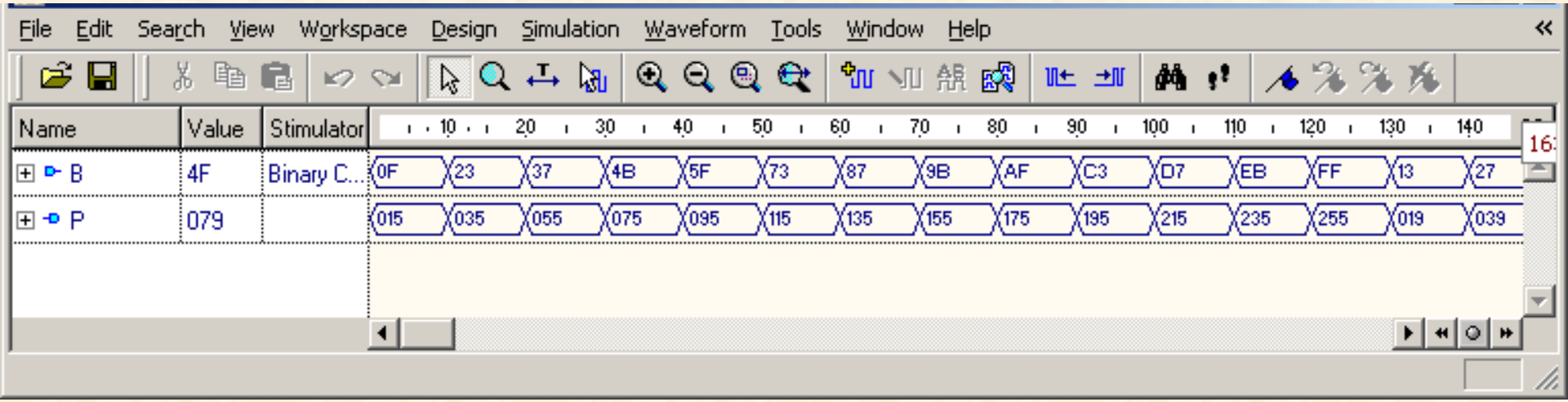
binbcd8.vhd

```
-- Title: Binary-to-BCD Converter
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity binbcd is
    port (
        B: in STD_LOGIC_VECTOR (7 downto 0);
        P: out STD_LOGIC_VECTOR (9 downto 0)
    );
end binbcd;
```

binbcd8.vhd (cont.)

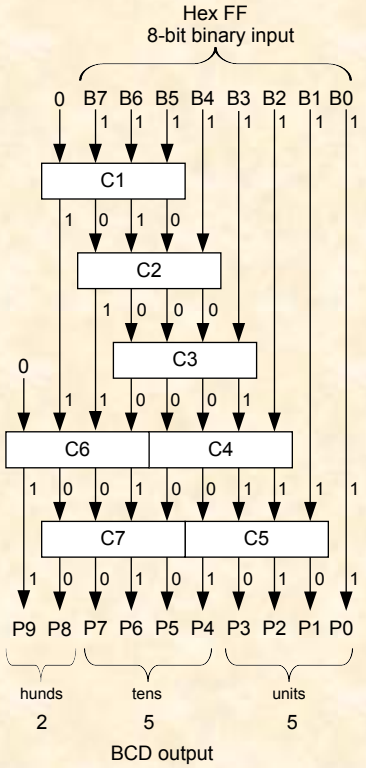
```
architecture binbcd_arch of binbcd is
begin
  bcd1: process(B)

  variable z: STD_LOGIC_VECTOR (17 downto 0);

  begin
       for i in 0 to 17 loop
            z(i) := '0';
       end loop;
            z(10 downto 3) := B;

       for i in 0 to 4 loop
            if z(11 downto 8) > 4 then
                 z(11 downto 8) := z(11 downto 8) + 3;
            end if;
            if z(15 downto 12) > 4 then
                 z(15 downto 12) := z(15 downto 12) + 3;
            end if;
            z(17 downto 1) := z(16 downto 0);
       end loop;
       P <= z(17 downto 8);
  end process bcd1;
end binbcd_arch;
```

Operation	Hundreds	Tens	Units	Binary	
B				7 4	3 0
HEX				F	F
Start				1 1 1 1	1 1 1 1
Shift 1			1	1 1 1 1	1 1 1
Shift 2			1 1	1 1 1 1	1 1
Shift 3			1 1 1	1 1 1 1	1
Add 3			1 0 1 0	1 1 1 1	1
Shift 4		1	0 1 0 1	1 1 1 1	
Add 3		1	1 0 0 0	1 1 1 1	
Shift 5		1 1	0 0 0 1	1 1 1	
Shift 6		1 1 0	0 0 1 1	1 1	
Add 3		1 0 0 1	0 0 1 1	1 1	
Shift 7	1	0 0 1 0	0 1 1 1	1	
Add 3	1	0 0 1 0	1 0 1 0	1	
Shift 8	1 0	0 1 0 1	0 1 0 1		
BCD	2	5	5		
P	9 8	7 4	3 0		
z	17 16	15 12	11 8	7 4	3 0

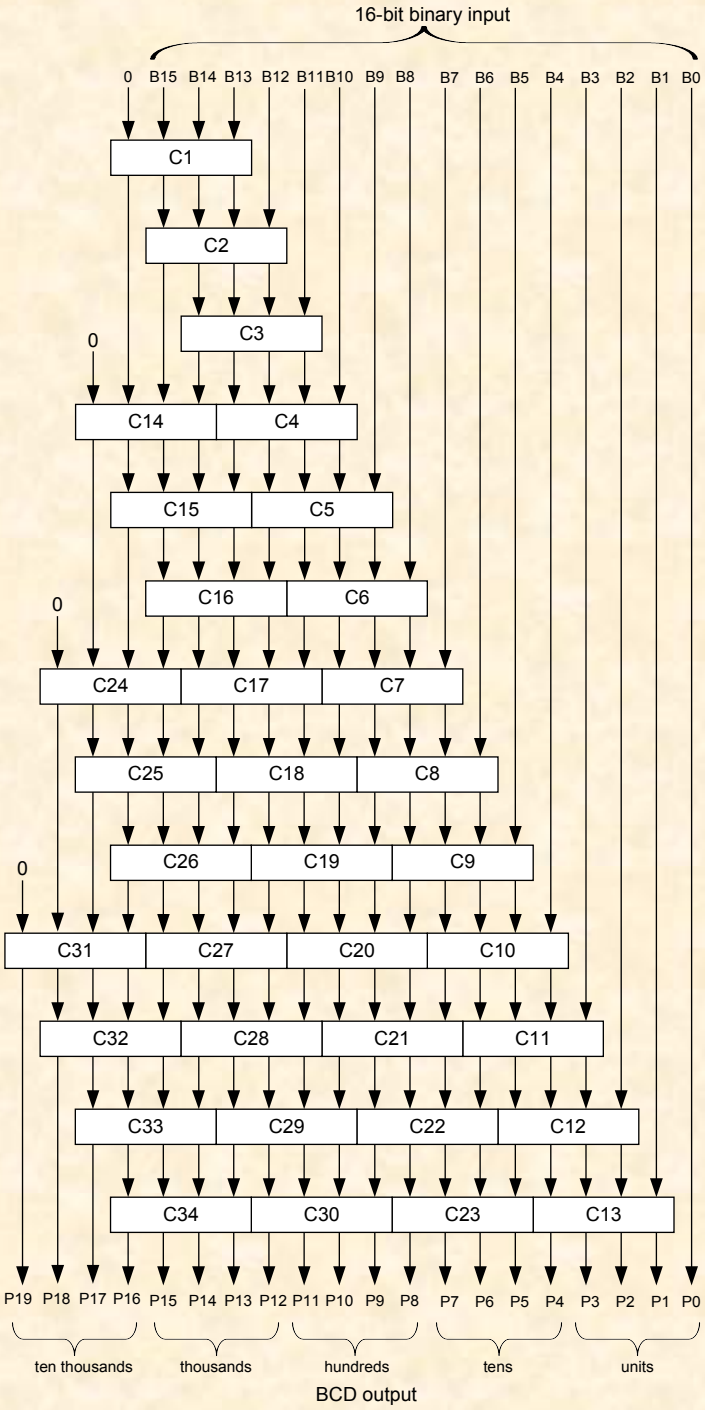


binbcd8.vhd

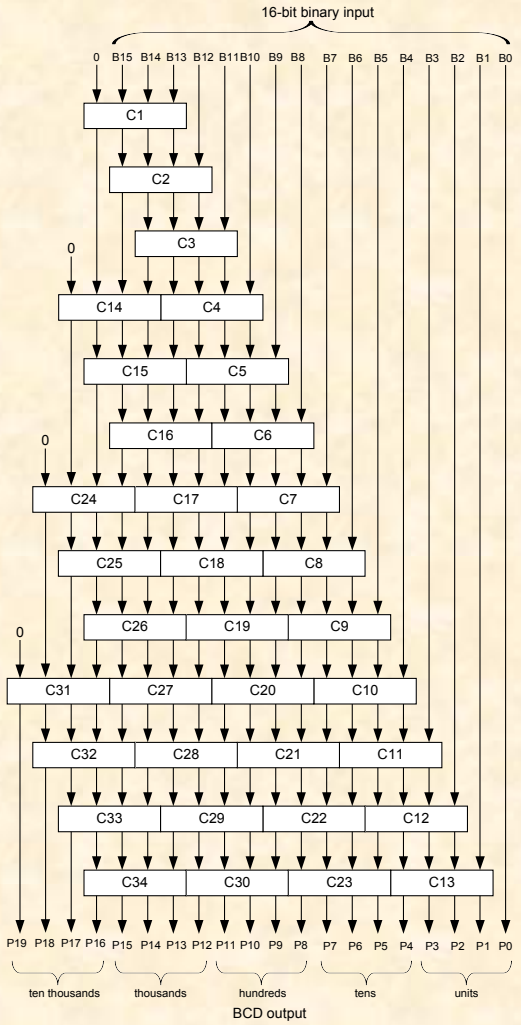
Operation	Hundreds	Tens	Units	Binary
B				7 4 3 0
HEX				F F
Start				1 1 1 1 1 1 1 1
Shift 1			1	1 1 1 1 1 1 1
Shift 2			1 1	1 1 1 1 1 1
Shift 3			1 1 1	1 1 1 1 1
Add 3			1 0 1 0	1 1 1 1 1
Shift 4		1	0 1 0 1	1 1 1 1
Add 3		1	1 0 0 0	1 1 1 1
Shift 5		1 1	0 0 0 1	1 1 1
Shift 6		1 1 0	0 0 1 1	1 1
Add 3		1 0 0 1	0 0 1 1	1 1
Shift 7	1	0 0 1 0	0 1 1 1	1
Add 3	1	0 0 1 0	1 0 1 0	1
Shift 8	1 0	0 1 0 1	0 1 0 1	
BCD	2	5	5	
P	9 8	7 4	3 0	
z	17 16	15 12	11 8	7 4 3 0



16-bit Binary-to-BCD Converter



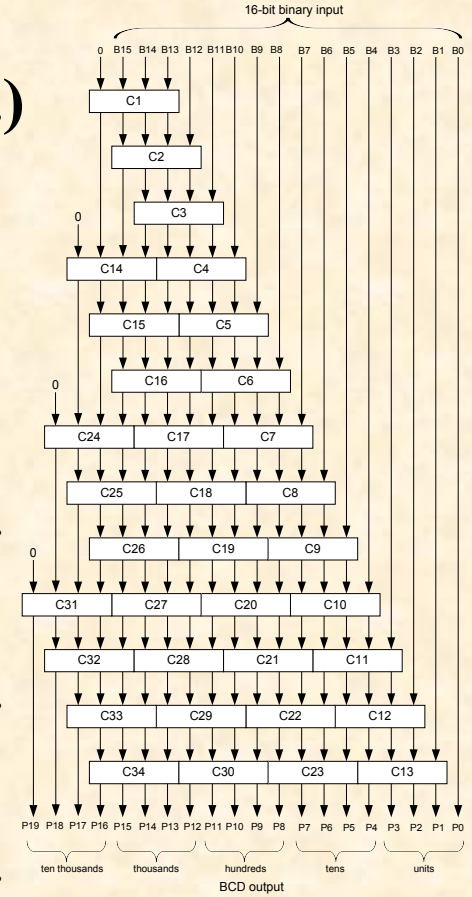
binbcd16.vhd

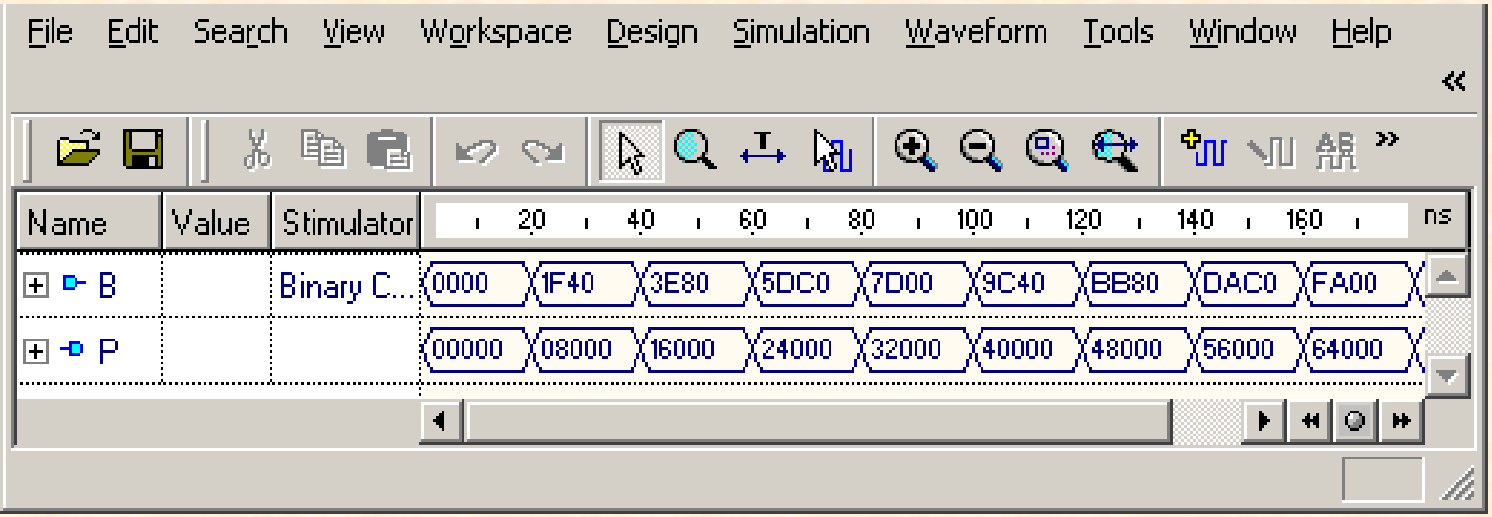



```
-- Title: Binary-to-BCD Converter
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity binbcd16 is
    port (
        B: in STD_LOGIC_VECTOR (15 downto 0);
        P: out STD_LOGIC_VECTOR (18 downto 0)
    );
end binbcd16;

architecture binbcd16_arch of binbcd16 is
begin
  bcd1: process(B)
      variable z: STD_LOGIC_VECTOR (34 downto 0);
```

binbcd16.vhd (cont.)

```
  begin
    for i in 0 to 34 loop
       z(i) := '0';
    end loop;
    z(18 downto 3) := B;

    for i in 0 to 12 loop
       if z(19 downto 16) > 4 then
          z(19 downto 16) := z(19 downto 16) + 3;
       end if;
       if z(23 downto 20) > 4 then
          z(23 downto 20) := z(23 downto 20) + 3;
       end if;
       if z(27 downto 24) > 4 then
          z(27 downto 24) := z(27 downto 24) + 3;
       end if;
       if z(31 downto 28) > 4 then
          z(31 downto 28) :=  z(31 downto 28) + 3;
       end if;
       z(34 downto 1) := z(33 downto 0);
    end loop;
    P <= z(34 downto 16);
  end process bcd1;
end binbcd16_arch;
```





binbcd16.vhd