

```

1  library ieee;
2  use ieee.std_logic_1164.all;
3  use IEEE.numeric_std.ALL;
4  use std.textio.all;
5
6  Entity EN_Shift is
7      Generic ( N : natural := 64 );
8      Port (
9          A : in std_logic_vector( N-1 downto 0 );
10         ShiftCount : in std_logic_vector( 5 downto 0 );
11         Y_LL , Y_RL, Y_RA : out std_logic_vector (N-1 downto 0)
12     );
13 end EN_Shift;
14
15 architecture IEEE_fn of EN_Shift is
16     signal shift_val : integer range 0 to N-1;
17
18     begin
19         -- Convert ShiftCount to integer
20         shift_val <= to_integer(unsigned(ShiftCount));
21
22         -- Left logical shift
23         Y_LL <= std_logic_vector(shift_left(unsigned(A), shift_val));
24
25         -- Right logical shift
26         Y_RL <= std_logic_vector(shift_right(unsigned(A), shift_val));
27
28         -- Right arithmetic shift (Treat A as signed so the MSB is the sign bit and
29         is extended)
30         Y_RA <= std_logic_vector(shift_right(signed(A), shift_val));
31
32 end IEEE_fn;
33
34 architecture barrel of EN_Shift is
35     -- Left logical levels
36     signal ll0, ll1, ll2, ll3, ll4, ll5, ll6 : std_logic_vector(N-1 downto 0);
37     -- Right logical levels
38     signal rl0, rl1, rl2, rl3, rl4, rl5, rl6 : std_logic_vector(N-1 downto 0);
39     -- Right arithmetic levels
40     signal ra0, ra1, ra2, ra3, ra4, ra5, ra6 : std_logic_vector(N-1 downto 0);
41
42     begin
43         -- level 0: input
44         ll0 <= A;
45         rl0 <= A;
46         ra0 <= A;
47
48         -- level 1: shift by 1 if ShiftCount(0) = '1'
49         -- Left logical
50         ll1 <= ll0(N-2 downto 0) & '0'
51             when ShiftCount(0) = '1' else
52             ll0;
53
54         -- Right logical
55         rl1 <= '0' & rl0(N-1 downto 1)
56             when ShiftCount(0) = '1' else
57             rl0;
58
59         -- Right arithmetic (sign bit extended)
60         ra1 <= ra0(N-1) & ra0(N-1 downto 1)
61             when ShiftCount(0) = '1' else
62             ra0;
63
64         -- level 2: shift by 2 if ShiftCount(1) = '1'
65         -- Left logical
66         ll2 <= ll1(N-3 downto 0) & (1 downto 0 => '0')
67             when ShiftCount(1) = '1' else
68             ll1;
69
70         -- Right logical
71         rl2 <= (1 downto 0 => '0') & rl1(N-1 downto 2)
72             when ShiftCount(1) = '1' else

```

```

73         r11;
74
75     -- Right arithmetic
76     ra2 <= (1 downto 0 => ra1(N-1)) & ra1(N-1 downto 2)
77         when ShiftCount(1) = '1' else
78         ra1;
79
80     -- level 3: shift by 4 if ShiftCount(2) = '1'
81     -- Left logical
82     l13 <= l12(N-5 downto 0) & (3 downto 0 => '0')
83         when ShiftCount(2) = '1' else
84         l12;
85
86     -- Right logical
87     r13 <= (3 downto 0 => '0') & r12(N-1 downto 4)
88         when ShiftCount(2) = '1' else
89         r12;
90
91     -- Right arithmetic
92     ra3 <= (3 downto 0 => ra2(N-1)) & ra2(N-1 downto 4)
93         when ShiftCount(2) = '1' else
94         ra2;
95
96     -- level 4: shift by 8 if ShiftCount(3) = '1'
97     -- Left logical
98     l14 <= l13(N-9 downto 0) & (7 downto 0 => '0')
99         when ShiftCount(3) = '1' else
100         l13;
101
102     -- Right logical
103     r14 <= (7 downto 0 => '0') & r13(N-1 downto 8)
104         when ShiftCount(3) = '1' else
105         r13;
106
107     -- Right arithmetic
108     ra4 <= (7 downto 0 => ra3(N-1)) & ra3(N-1 downto 8)
109         when ShiftCount(3) = '1' else
110         ra3;
111
112     -- level 5: shift by 16 if ShiftCount(4) = '1'
113     -- Left logical
114     l15 <= l14(N-17 downto 0) & (15 downto 0 => '0')
115         when ShiftCount(4) = '1' else
116         l14;
117
118     -- Right logical
119     r15 <= (15 downto 0 => '0') & r14(N-1 downto 16)
120         when ShiftCount(4) = '1' else
121         r14;
122
123     -- Right arithmetic
124     ra5 <= (15 downto 0 => ra4(N-1)) & ra4(N-1 downto 16)
125         when ShiftCount(4) = '1' else
126         ra4;
127
128     -- level 6: shift by 32 if ShiftCount(5) = '1'
129     -- Left logical
130     l16 <= l15(N-33 downto 0) & (31 downto 0 => '0')
131         when ShiftCount(5) = '1' else
132         l15;
133
134     -- Right logical
135     r16 <= (31 downto 0 => '0') & r15(N-1 downto 32)
136         when ShiftCount(5) = '1' else
137         r15;
138
139     -- Right arithmetic
140     ra6 <= (31 downto 0 => ra5(N-1)) & ra5(N-1 downto 32)
141         when ShiftCount(5) = '1' else
142         ra5;
143
144     -- Final outputs
145     Y_LL <= l16;

```

```
146      Y_RL <= r16;  
147      Y_RA <= ra6;  
148  
149  
150  end barrel;
```