

Barrel Shifter (4-bit)

- This is a combinational CLK (not sequential)
- It shifts the 4-bit by 0-3 positions (4-bit)
either left or Right based on direction (dir)
- Uses multiplexing logic (case statement) to instantly rotate / shift data - no CLK, no seq. logic
- Code explained :-

→ Module ports

1. data_in [3:0] → 4-bit ilp word

2. shift_amt [1:0] → shift 0, 1, 2, 3

3. dir → shift direct

→ 0 = logical left shift

→ 1 = logical right shift

4. data_out [3:0] → shifted ilp

→ core logic :- (always @(*) pure combinational)

A nested case block selects "shift direct" first
then select the shift amount.

→ left logical shift (dir=0)

shift-amt

expⁿ

ilp pattern

0 no-shift

data-in

1 shift by 1 (left)

{ data_in[2:0], 1'b0 }

2 shift left by 2

{ data_in[1:0], 2'b00 }

3 shift left by 3

{ data_in[0], 3'b000 }

* MSB move towards left, 0 fill at right

→ Right logical shift ($\text{dir} = 1$)

shift-amt

op^n

0/p pattern

0

no shift

data-in

1

shift right by 1

{1'b0, data-in[3:1]}

2

shift right by 2

{2'b00, data-in[3:2]}

3

shift right by 3

{3'b000, data-in[3]}

* 1'st bits moves towards the right, zeros fill on the left

e.g:- input = d[1101]

① shift amt = 2, dir = 0 (left)

$$1101 \xrightarrow{\leftarrow} 1010 \xrightarrow{\leftarrow} [0100] = \underline{\underline{Y}}$$

② shift amt = 3, dir = 1 (right)

$$1101 \xrightarrow{\rightarrow} 0110 \xrightarrow{\rightarrow} 0010 \xrightarrow{\rightarrow} [0001] = \underline{\underline{I}}$$

- Hence CLT implements pure logical shifting
(no rotation, no arithmetic sign extend).

