EE2026
Lab 3 pg 12

case hint example explanation

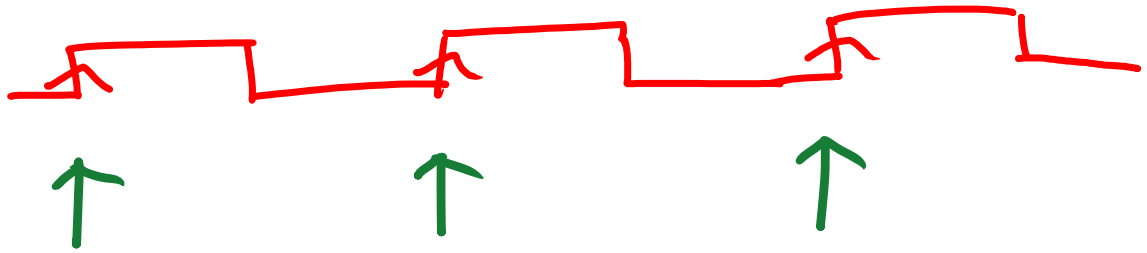
```
always @ (posedge clk_25_mhz)
begin
    case (counter_value)
       2'd0:
          begin
                my_value_a <= 20;
                my_value_b <= 40;
          end
       2'd1:
          begin
                my_value_a <= 100;
                my_value_b <= 200;
          end
       2'd2: my_value_c <= 5;
       default: my_value_d <= 9;
    endcase
end
```

This `begin` and `end` defines the start and end of the always block.

always block

```
always @ (posedge clk_25_mhz)
begin
    case (counter_value)
       2'd0:
           begin
                  my_value_a <= 20;
                  my_value_b <= 40;
           end
       2'd1:
           begin
                  my_value_a <= 100;
                  my_value_b <= 200;
           end
       2'd2: my_value_c <= 5;
       default: my_value_d <= 9;
    endcase
end
```

clk_25_mhz



At each rising edge of this signal (clk_25_mhz), this block of code is executed once.

```
always @ (posedge clk_25_mhz)
begin
    case (counter_value)
        2'd0:
            begin
                    my_value_a <= 20;
                    my_value_b <= 40;
            end
        2'd1:
            begin
                    my_value_a <= 100;
                    my_value_b <= 200;
            end
        2'd2: my_value_c <= 5;
        default: my_value_d <= 9;
    endcase
end
```

This is the case statement. It starts with `case` and ends with `endcase`.

```
always @ (posedge clk_25_mhz)
begin
    case (counter_value)
        2'd0:
            begin
                my_value_a <= 20;
                my_value_b <= 40;
            end
        2'd1:
            begin
                my_value_a <= 100;
                my_value_b <= 200;
            end
        2'd2: my_value_c <= 5;
        default: my_value_d <= 9;
    endcase
end
```

decimal

In this example, the case statement checks for `counter_value`, which is a 2-bit value.

`counter_value` is a 2-bit register with 4 possible decimal values 0, 1, 2, 3.

```
always @ (posedge clk_25_mhz)
begin
    case (counter_value)
        2'd0:
            begin
                    my_value_a <= 20;
                    my_value_b <= 40;
            end
        2'd1:
            begin
                    my_value_a <= 100;
                    my_value_b <= 200;
            end
        2'd2: my_value_c <= 5;
        default: my_value_d <= 9;
    endcase
end
```

①

In this example, the case statement checks for `counter_value`, which is a 2-bit value.

If `counter_value` is 0, these two lines of code in ① will run.

After the codes are executed, we will skip the rest of the lines of code and go to `endcase`.

```
always @ (posedge clk_25_mhz)
begin
    case (counter_value)
        2'd0:
            begin
                    my_value_a <= 20;
                    my_value_b <= 40;
            end
        2'd1:
            begin
                    my_value_a <= 100;
                    my_value_b <= 200;
            end
        2'd2: my_value_c <= 5;
        default: my_value_d <= 9;
    endcase
end
```

① ②

In this example, the case statement checks for `counter_value`, which is a 2-bit value.

If `counter_value` is 1, the two lines of code in ② will run.

The rest of the code is skipped.

```
always @ (posedge clk_25_mhz)
begin
    case (counter_value)
        2'd0:
            begin
                    my_value_a <= 20;
                    my_value_b <= 40;
            end
        2'd1:
            begin
                    my_value_a <= 100;
                    my_value_b <= 200;
            end
        2'd2: my_value_c <= 5;
        default: my_value_d <= 9;
    endcase
end
```

① ② ③

In this example, the case statement checks for `counter_value`, which is a 2-bit value.

If `counter_value` is 1, the two lines of code in ② will run.

The rest of the code is skipped.

If `counter_value` is 2, ③ will run.

```
always @ (posedge clk_25_mhz)
begin
    case (counter_value)
        2'd0:
            begin
                    my_value_a <= 20;
                    my_value_b <= 40;
            end
        2'd1:
            begin
                    my_value_a <= 100;
                    my_value_b <= 200;
            end
        2'd2: my_value_c <= 5;
        default: my_value_d <= 9;
    endcase
end
```

① ② ③ ④

In this example, the case statement checks for `counter_value`, which is a 2-bit value.

If `counter_value` is 3, as it is not specified as one of the cases, the `default` case is executed and ④ is run.

The `default` case catches all undefined values. (eg. don't cares, X, and Z)