به نام خدا

این کد شامل دو ماژول است: یک فیلیپ فلاپ lk و یک شمارنده دودویی 4 بیتی با بار شدن موازی و پاک شدن همزمان.

ماژول " jk_ff "(فیلیپ فلاپ jk)

در ابتدا یک ماژول برای مدار فیلیپ فلاپ jk به نام ماژول " jk_ff " که دارای 4 ورودی و 1 خروجی می باشد به صورت زیر تعریف کردیم :

```
module jk_ff (
    input j,
    input k,
    input clk,
    input clr,
    output reg q
    );
```

`Input j` = ورودی j

`Input k` = ورودی k

`Input $clk$` = ورودی سیگنال ساعت

`Input $clr$` = ورودی سیگنال پاک سازی (clear)

`Output reg q` = خروجی q که به صورت یک ریجستر تعریف شده است.

در ادامه با استفاده از لبه های کلاک و پاک شونده زمانی که لبه بالارونده سیگنال ساعت با لبه پایین رونده سیگنال پاکسازی رخ میدهد اجرا می شود.

```
always @(posedge clk or negedge clr)
  begin
    if (clr==0)
      q = 0;
    else
      begin
      if (j==0 && k==0)
        q = q;
    else
      if(j==0 && k==1)
        q = 0;
    else
      if(j==1 && k==0)
        q = 1;
    else
      if(j==1 && k==1)
        q = ~q;
      end
  end

endmodule
```

اگر `clr` برابر 0 باشد، `q` به 0 تنظیم می شود (پاکسازی).

در غیر اینصورت،براساس مقادیر 'j' و `k` ، خروجی `q` به صورت زیر تنظیم میشود:

اگر `j==0` و `k==0` باشد `q` تغییر نمیکند.

اگر `j==0` و `k==1` باشد `q` به 0 تنظیم میشود.

اگر `j==1` و `k==0` باشد `q` به 1 تنظیم میشود.

اگر `j==1` و `k==1` باشد `q` معکوس میشود (toggle).

در ادامه کد یک ماژول کلی برای مدار شمارنده دودویی 4 بیتی با بار شدن موازی و پاک شدن همزمان به نام ماژول "binary_counter " که دارای 5 ورودی و 2 خروجی است تعریف میکنیم :

```
module binary_counter (
    input [3:0] I,
    input ld,
    input clr,
    input clk,
    input count,
    output carry,
    output [3:0] A
);
```

`input [3:0] $I$` = ورودی 4 بیتی برای بارگذاری موازی.

`input $ld$` = ورودی سیگنال بارگذاری.

`input $clr$` = ورودی سیگنال پاکسازی.

`input $clk$` = ورودی سیگنال ساعت.

`input count`= ورودی سیگنال شمارش.

`output carry`= خروجی سیگنال حمل (carry).

`output [3:0] A`= خروجی 4 بیتی شمارنده.

تعریف سیم ها برای سیگنال های `j` و `k` برای هر فیلیپ فلاپ jk و سیگنال `carry`:

```
wire j0, k0, j1, k1, j2, k2, j3, k3,carry;
```

در ادامه کد هر وایر را با توجه به گیت مربوطه assign کردیم :

```
/*****************************  GATES  ******************************/

assign j0 = (~clr & ld &count) | ((~clr & ld) & I[0]);
assign k0 = clr | (~clr & ~ld & count) | ((~clr & ld) & ~I[0]);
assign j1 = (A[0] & (~clr & ~ld & count)) | ((~clr & ld) & I[1]);
assign k1 = clr | (A[0] & (~clr & ~ld & count)) | ((~clr & ld) & ~I[1]);
assign j2 = (A[1] & ( A[0] & (~clr & ~ld & count)) | ((~clr & ld) & I[2]));
assign k2 = clr | (A[1] & ( A[0] & (~clr & ~ld & count)) | ((~clr & ld) & ~I[2]));
assign j3 = (A[2] & (A[1] & ( A[0] & (~clr & ~ld & count)))) | ((~clr & ld) & I[3]);
assign k3 = clr | (A[2] & (A[1] & ( A[0] & (~clr & ~ld & count)))) | ((~clr & ld) & ~I[3]);
assign carry = A[3] & (A[2] & (A[1] & ( A[0] & (~clr & ~ld & count))));
```

تخصیصات منطقی برای سیگنال های `j` و `k` هر فلیپ فلاپ jk و

سیگنال `carry`:

`j0` و `k0` برای فلیپ فلاپ با کمترین بیت.

`j1` و `k1` برای فلیپ فلاپ با بیت دوم.

`j2` و `k2` برای فلیپ فلاپ با بیت سوم.

`j3` و `k3` برای فلیپ فلاپ با بیت بالاترین.

`carry` زمانی فعال می شود که تمام بیت های `A` برابر 1 باشد.

سپس به دلیل وجود 4 عدد فیلیپ فلاپ در مدار 4 بار ماژول مربوط به

فیلیپ فلاپ صدا زده میشه:

```
/*****************  CALL THE MODUIE FLIP_FLOP_JK  ********************/

jk_ff ff0 (
  .j(j0),
  .k(k0),
  .clk(clk),
  .q(A[0])
);

jk_ff ff1 (
  .j(j1),
  .k(k1),
  .clk(clk),
  .q(A[1])
);

jk_ff ff2 (
  .j(j2),
  .k(k2),
  .clk(clk),
  .q(A[2])
);

jk_ff ff3 (
  .j(j3),
  .k(k3),
  .clk(clk),
  .q(A[3])
);
endmodule
```

فراخوانی ماژول `jk_ff` برای هر بیت از شمارنده :

`ff0` برای بیت 0.

`ff1` برای بیت 1.

`ff2` برای بیت 2.

`ff3` برای بیت 3.