

Behavior Modeling

Verilog

If Statements

Syntax

```

if (expression)
begin
...statements...
end

else if (expression)
begin
...statements...
end
...more else if blocks

else
begin
...statements...
end
    
```

```

if (alu_func == 2'b00)
aluout = a + b;
else if (alu_func == 2'b01)
aluout = a - b;
else if (alu_func == 2'b10)
aluout = a & b;
else // alu_func == 2'b11
aluout = a | b;
    
```

Nov 16, 2006 Abdul-Rahman Elshafiei 2

Case Statements

Syntax

```

case (expression)
case_choice1:
begin
...statements...
end

case_choice2:
begin
...statements...
end

...more case choices blocks...

default:
begin
...statements...
end
endcase
    
```

```

case (alu_ctr)
2'b00: aluout = a + b;
2'b01: aluout = a - b;
2'b10: aluout = a & b;
default: aluout = 1'bx; // Treated as don't cares for
endcase // minimum logic generation.
    
```

Nov 16, 2006 Abdul-Rahman Elshafiei 3

For loops

Syntax

```

for (count= value1;
count<=</>/>=
value2;
count=count+/- step)
begin
...statements...
end
    
```

```

integer j;
for(j=0;j<=7;j=j+1)
begin
c[j] = a[j] + b[j];
end
    
```

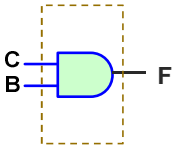
Abdul-Rahman Elshafiei Nov 16, 2006 4

Component Inference

Always

```

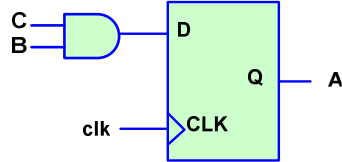
module qualquer (C, B, F)
input C,B;
output F;
reg F;
always @(C,B)
begin
F <= C & B;
end
end module
    
```



The diagram shows a green AND gate with two inputs labeled 'C' and 'B' on the left, and one output labeled 'F' on the right. The gate is enclosed in a dashed rectangular box.

Flip-Flops

```
always@(posedge clk)
begin
a<=b&c;
end
```



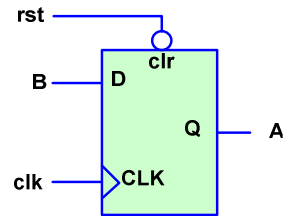
Nov 16, 2006

Abdul-Rahman Elshafei

7

D Flip-Flop with Asynchronous Reset

```
always@(posedge clk or
negedge rst)
begin
if (!rst) a<=0;
else a<=b;
end
```



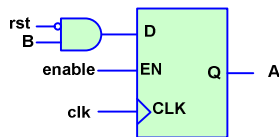
Nov 16, 2006

Abdul-Rahman Elshafei

8

D Flip-flop with Synchronous reset and Enable

```
always@(posedge clk)
begin
if (rst) a<=0;
else if (enable) a<=b;
end
```



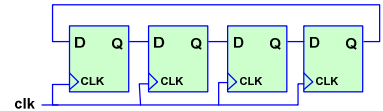
Nov 16, 2006

Abdul-Rahman Elshafei

9

Shift Registers

```
reg[3:0] Q;
always@(posedge clk or
posedge rst)
begin
if (rst) Q<=0;
else begin
Q<=Q<<1;
Q[0]<=Q[3];
end
```



Nov 16, 2006

Abdul-Rahman Elshafei

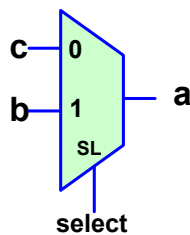
10

Multiplexers

```
Method 1
assign a = (select ? b : c);

Method 2
always@(select or b or c) begin
if(select) a=b;
else a=c;
end

Method 2b
case(select)
1'b1: a=b;
1'b0: a=c;
endcase
```



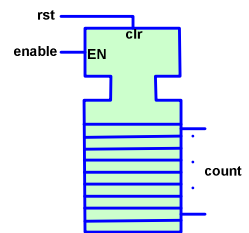
Nov 16, 2006

Abdul-Rahman Elshafei

11

Counters

```
reg [7:0] count;
wire enable;
always@(posedge clk or
negedge rst)
begin
if (rst) count<=0;
else if (enable)
count<=count+1;
end
```



Nov 16, 2006

Abdul-Rahman Elshafei

12