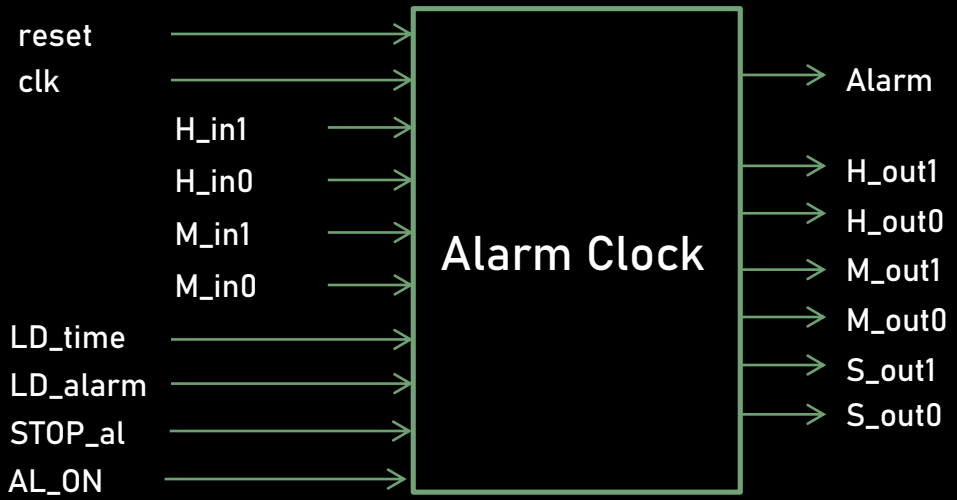
CLOCK GENERATOR WITH ALARM

Khushi Baurasi
ECE – 211114213
NIT - BHOPAL

FEATURES

- ✓ Clock generation.
- ✓ Initializing clock time to a particular value.
- ✓ Setting time for alarm.
- ✓ Enabling and disabling alarm.
- ✓ Stopping alarm.

BLOCK DIAGRAM



FLOW CHART


RESET
YES
Set Alarm to zero.
Set time to the
given input time.
NO
Clk_1s
YES
LD_alarm
NO
LD_time
YES
Set time to the
given input time.
YES
Set Alarm clock
to the given input
time
NO
Clock operates
normally by
adding seconds.
Alarm_time==
real time
YES
AL_ON
YES
Raise alarm
Output.
YES
STOP_al
YES
Lower alarm
Output.
NO

VERILOG CODE

```
module Aclock(                output [3:0]  M_out1,
input reset,                  output [3:0]  M_out0,
input clk,                    output [3:0]  S_out1,
input [1:0] H_in1,            output [3:0]  S_out0);
input [3:0] H_in0,
input [3:0] M_in1,            reg clk_1s;
input [3:0] M_in0,            reg [3:0] tmp_1s;
input LD_time,                reg [5:0] tmp_hour, tmp_minute, tmp_second;
input   LD_alarm,             reg [1:0] c_hour1,a_hour1;
input   STOP_al,              reg [3:0] c_hour0,a_hour0;
input   AL_ON,                reg [3:0] c_min1,a_min1;
output reg Alarm,             reg [3:0] c_min0,a_min0;
output [1:0]  H_out1,         reg [3:0] c_sec1,a_sec1;
output [3:0]  H_out0,         reg [3:0] c_sec0,a_sec0;
```

Initialization

```
function [3:0] mod_10;
 input [5:0] number;
 begin
 mod_10 = (number >=50) ? 5 : ((number >= 40)? 4 :((number >= 30)? 3 :((number >= 20)? 2 :((number >= 10)? 1 :0))));
 end
endfunction
```

MOD 10 function

```
always @(posedge clk_1s or posedge
reset )
 begin
 if(reset) begin
 a_hour1 <= 2'b00;
 a_hour0 <= 4'b0000;
 a_min1 <= 4'b0000;
 a_min0 <= 4'b0000;
 a_sec1 <= 4'b0000;
 a_sec0 <= 4'b0000;
 tmp_hour <= H_in1*10 + H_in0;
 tmp_minute <= M_in1*10 + M_in0;
 tmp_second <= 0;
 end
 else begin
 if(LD_alarm) begin
 a_hour1 <= H_in1;
 a_hour0 <= H_in0;
 a_min1 <= M_in1;
 a_min0 <= M_in0;
 a_sec1 <= 4'b0000;
 a_sec0 <= 4'b0000;
 end
```

```
if(LD_time) begin
 tmp_hour <= H_in1*10 + H_in0;
 tmp_minute <= M_in1*10 + M_in0;
 tmp_second <= 0;
 end
 else begin
 tmp_second <= tmp_second + 1;
 if(tmp_second >=59) begin
 tmp_minute <= tmp_minute + 1;
 tmp_second <= 0;
 if(tmp_minute >=59) begin
 tmp_minute <= 0;
 tmp_hour <= tmp_hour + 1;
 if(tmp_hour >= 24) begin
 tmp_hour <= 0;
 end
 end
 end
 end
 end
 end
```

Loading and incrementing time

```
always @(posedge clk or posedge
reset)
 begin
 if(reset)
 begin
 tmp_1s <= 0;
 clk_1s <= 0;
 end
 else begin
 tmp_1s <= tmp_1s + 1;
 if(tmp_1s <= 5)
 clk_1s <= 0;
 else if (tmp_1s >= 10) begin
 clk_1s <= 1;
 tmp_1s <= 1;
 end
 else
 clk_1s <= 1;
 end
 end
```

Make 1s clock

```
always @(*) begin

if(tmp_hour>=20) begin
c_hour1 = 2;
end
else begin
if(tmp_hour >=10)
c_hour1  = 1;
else
c_hour1 = 0;
end
c_hour0 = tmp_hour - c_hour1*10;
c_min1 = mod_10(tmp_minute);
c_min0 = tmp_minute - c_min1*10;
c_sec1 = mod_10(tmp_second);
c_sec0 = tmp_second - c_sec1*10;
end

assign H_out1 = c_hour1;
assign H_out0 = c_hour0;
assign M_out1 = c_min1;
assign M_out0 = c_min0;
assign S_out1 = c_sec1;
assign S_out0 = c_sec0;
```

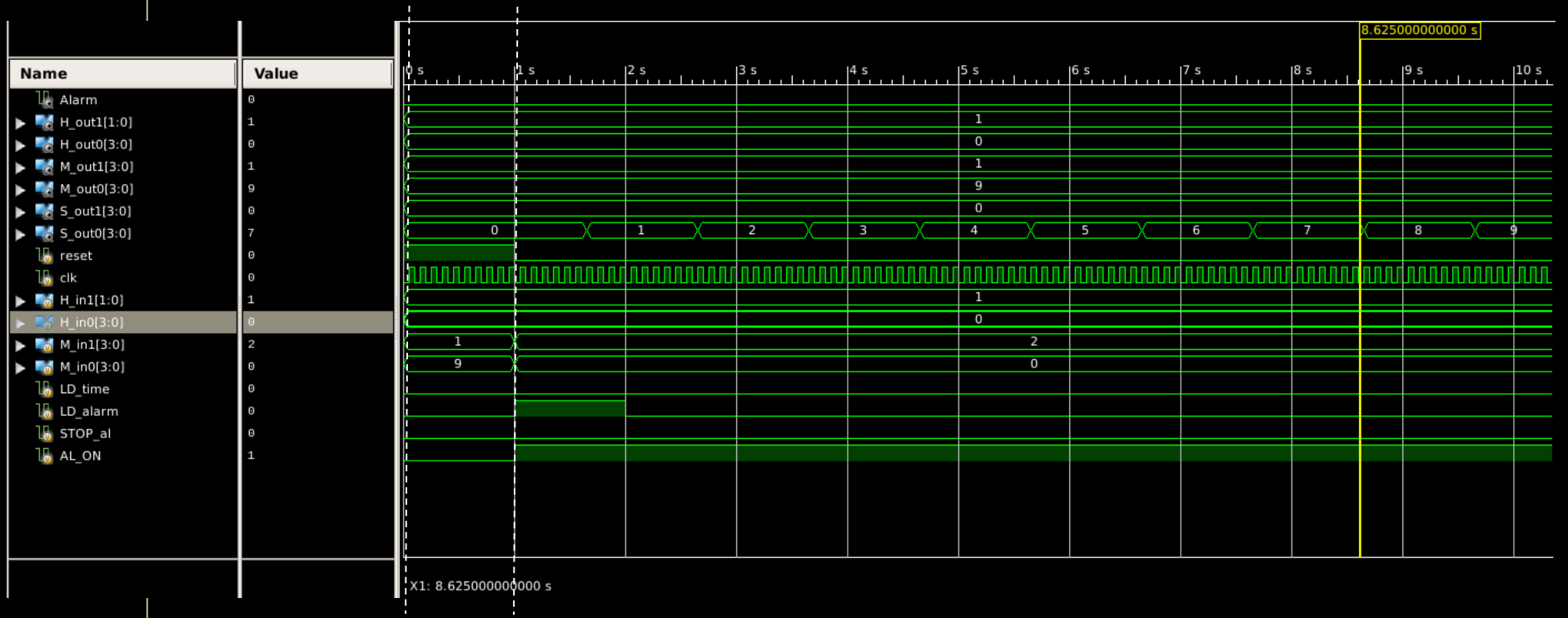
Output time

```
always @(posedge clk_1s or
posedge reset)
begin
 if(reset)
 Alarm <=0;
 else begin

if({a_hour1,a_hour0,a_min1,a_min0,a_
sec1,a_sec0}=={c_hour1,c_hour0,c_mi
n1,c_min0,c_sec1,c_sec0})
 begin / if(AL_ON) Alarm <= 1;
 end
 if(STOP_al)
 Alarm <=0;
 end
 end
```

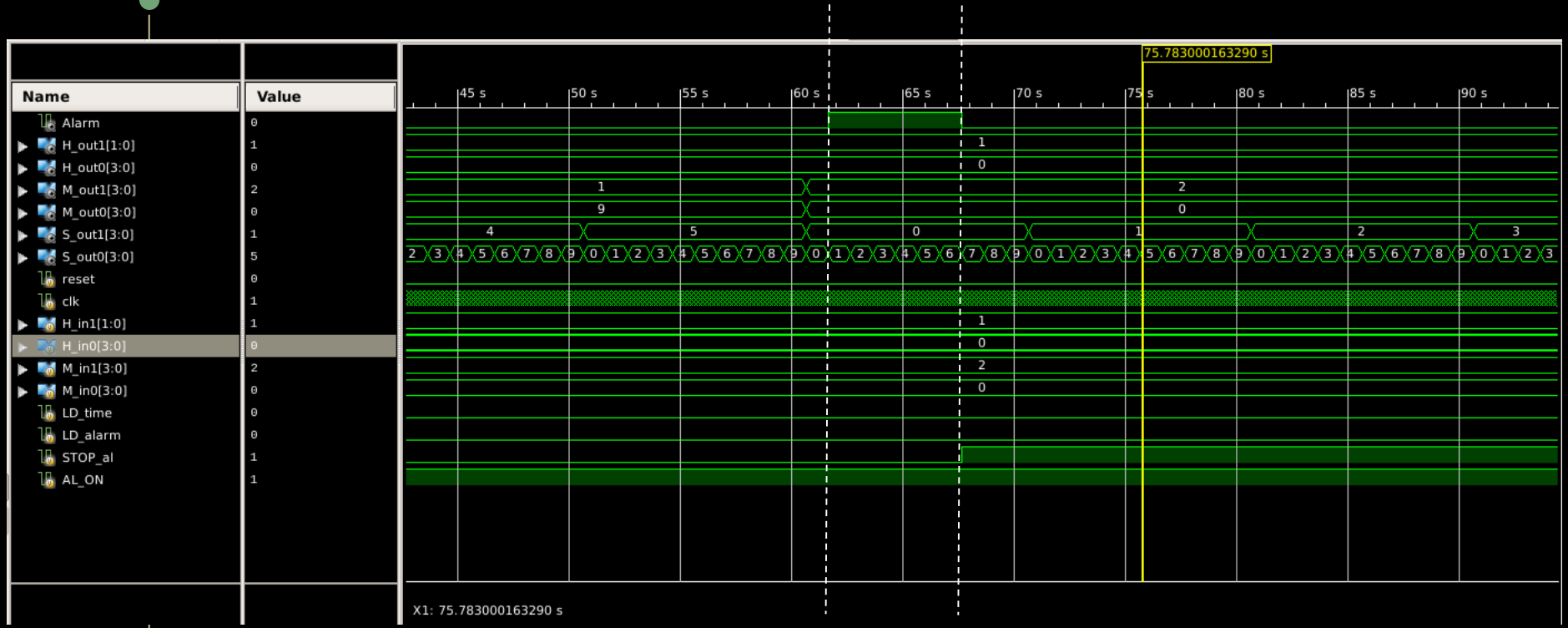
Setting and disabling alarm

SIMULATION OUTPUT



reset=1

LD_alarm=1



Alarm = 1

STOP_al = 1

THANK YOU!