Electronics and Communication Engineering Department
Digital Logic Design course (EEC242)
Fall 2022 - 2023

Digital Clock

الاسم	ID
أحمد أسامة محمد عفيفي محمد	20010038
أحمد عبد الحكيم عبد السلام علي	20010124
محمد أشرف السيد محمود	20011488
مصطفي خالد خميس	20011922
مصطفى محمد عبد العظيم حسنين	20011950
زياد عزت سليمان محمد	20010634

Steps of digital clock mechanism

The entity declarations

1. Inputs to the clock is the:

[clock---- initial value of second--- initial value of minutes ------initial value of hours] from the type of integer with initial value=0 and specific range [0:60] or [0 :24]

2. Outputs to the clock is the:

[final value of second--- final value of minutes ------ final value of hours] from the type of integer with initial value=0 and specific range [0:60] or [0 :24]

3. Buffer:

[set option of the clock] from the type standard logic

```
1   library IEEE;
2   use IEEE.STD_LOGIC_1164.ALL;
3
4   entity DigitalClock is
5       port(sec    :  out   integer range 0 to 60      :=  0;
6            min    :  out   integer range 0 to 60      :=  0;
7            hrs    :  out   integer range 0 to 24      :=  0;
8            inSec  :  in    integer range 0 to 60      :=  0;
9            inMin  :  in    integer range 0 to 60      :=  0;
10           inHrs  :  in    integer range 0 to 24      :=  0;
11           clk    :  in    std_logic   := '0';
12           set    :  buffer std_logic  := '0');
13  end DigitalClock;
14
```

The architecture body

Signals

1. Counter seconds: a temporary variable to store the changeable values of the seconds.

2. Counter minutes: a temporary variable to store the changeable values of the minutes

3. Counter hours: a temporary variable to store the changeable values of the hours.

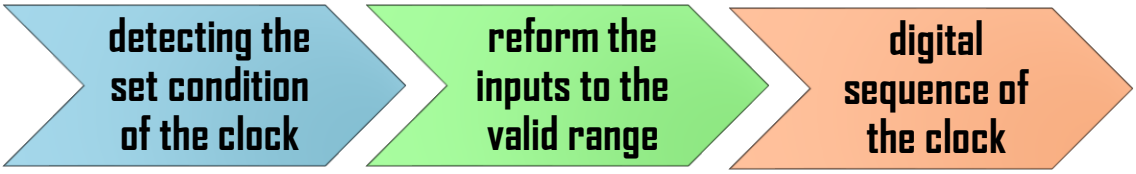
```
architecture Behavioural of DigitalClock is
  signal counterSec : integer range 0 to 60 := 0;
  signal counterMin : integer range 0 to 60 := 0;
  signal counterHrs : integer range 0 to 24 := 0;
```

Process

I. The effecting variables on the process are [clock, set]
Declaring 3 inner variables will be stored in the signals:
1)inner counter sec
2)inner counter min
3)inner counter

```
begin
   process(clk, set)
   variable innercountSec : integer range 0 to 60 := 0;
   variable innercountMin : integer range 0 to 60 := 0;
   variable innercountHrs : integer range 0 to 24 := 0;
```

II. The If conditions



- First stage

➢ 1)If (“set=1”)

innerCounterSec = initial value of seconds

innerCounterMin = initial value of minutes

innerCounterHrs = initial value of hours

Set = ‘0’

```
if (set = '1') then
   innercountSec := inSec;
   innercountMin := inMin;
   innercountHrs := inHrs;
```

- second stage

➢ else if (clock event and clock) =1

1)settling the inputs value to the range

a) If initial value of second>60

initial value of second= initial value of second -60.

Counter seconds = initial value of second

Counter minutes= initial value of minutes+1

```
if inSec >= 60 then
   innercountSec := inSec - 60;
   innercountMin := inMin + 1;
end if;
```

b) If initial value of minutes>60

initial value of minutes= initial value of minutes -60.

Counter minutes= initial value of minutes -60

Counter hours= initial value of hours+1

```
if innercountMin >= 60 then
   innercountMin := innercountMin - 60;
   innercountHrs := inHrs + 1;
end if;
```

c) If initial value of hours>24

initial value of hours= initial value of hours – 60

```
if innercountHrs >= 24 then
   innercountHrs := innercountHrs - 24;
end if;
```

Then set the counter signals to the values of the variables and reset the set buffer to '0'

```
counterSec <= innercountSec;
counterMin <= innercountMin;
counterHrs <= innercountHrs;
set <= '0';
```

- Third stage

➢ Increment seconds by 1

➢ if Counter seconds ≥ 59

I. Counter minutes = Counter minutes + 1

II. Counter seconds = 0

```
elsif clk 'event and clk = '1' then
   counterSec <= counterSec + 1;
   if counterSec >= 59 then
      counterMin <= counterMin + 1;
      counterSec <= 0;
```

➢ if Counter minutes ≥ 59

I. Counter hours = counter hours + 1

II. Counter minutes = 0

```
if counterMin >= 59 then
   counterHrs <= counterHrs + 1;
   counterMin <= 0;
```

➢ if Counter hours ≥ 23

I. Counter hours=0

II. Counter minutes=0

III. Counter seconds=0

```
if counterHrs >= 23 then
   counterHrs <= 0;
   counterMin <= 0;
   counterSec <= 0;
```

The result

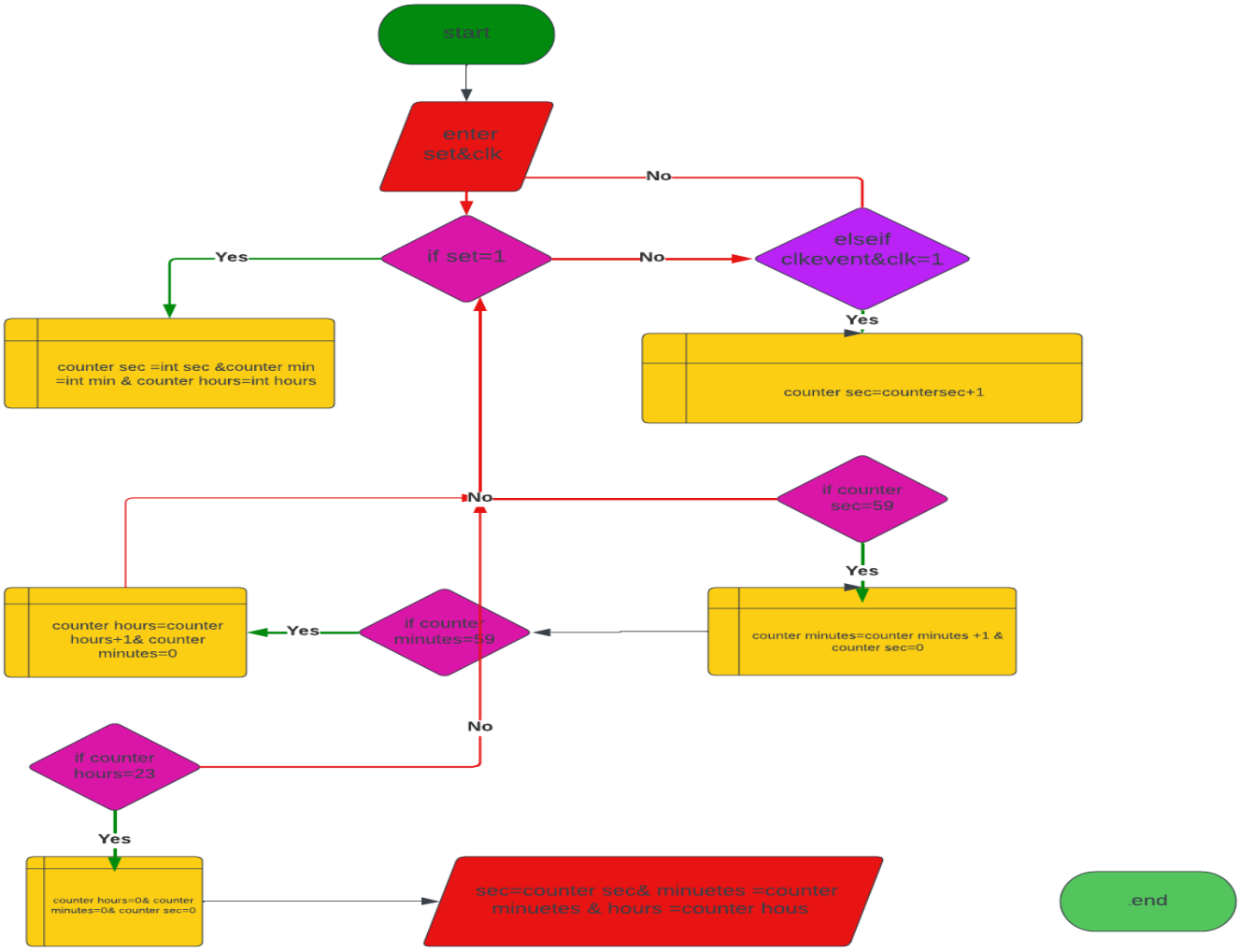
final value of second= Counter seconds

final value of minutes= Counter minutes

final value of hours= Counter hours

```
        sec <= counterSec;
        min <= counterMin;
        hrs <= counterHrs;
end Behavioural;
```

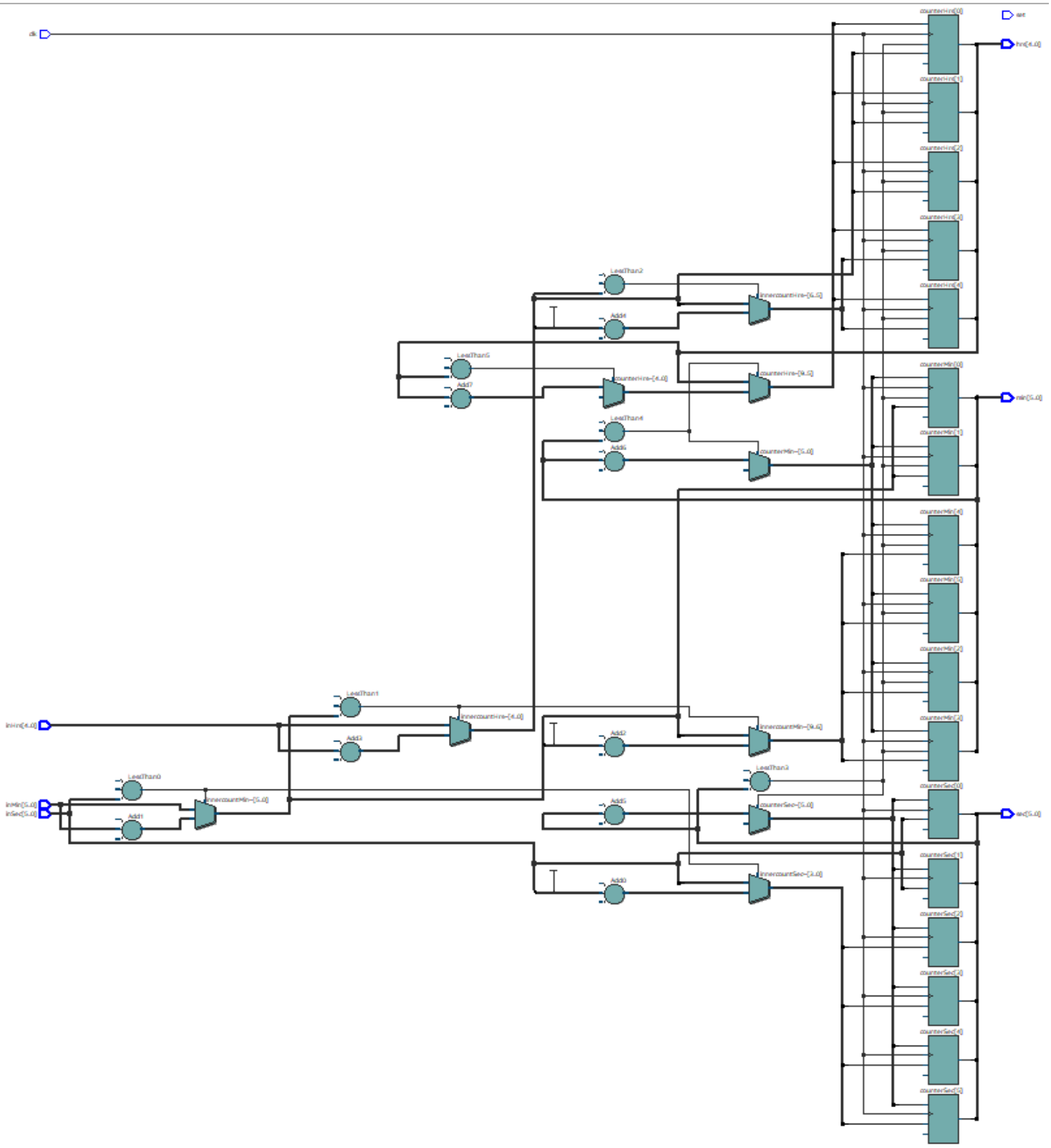
The flowchart



The code

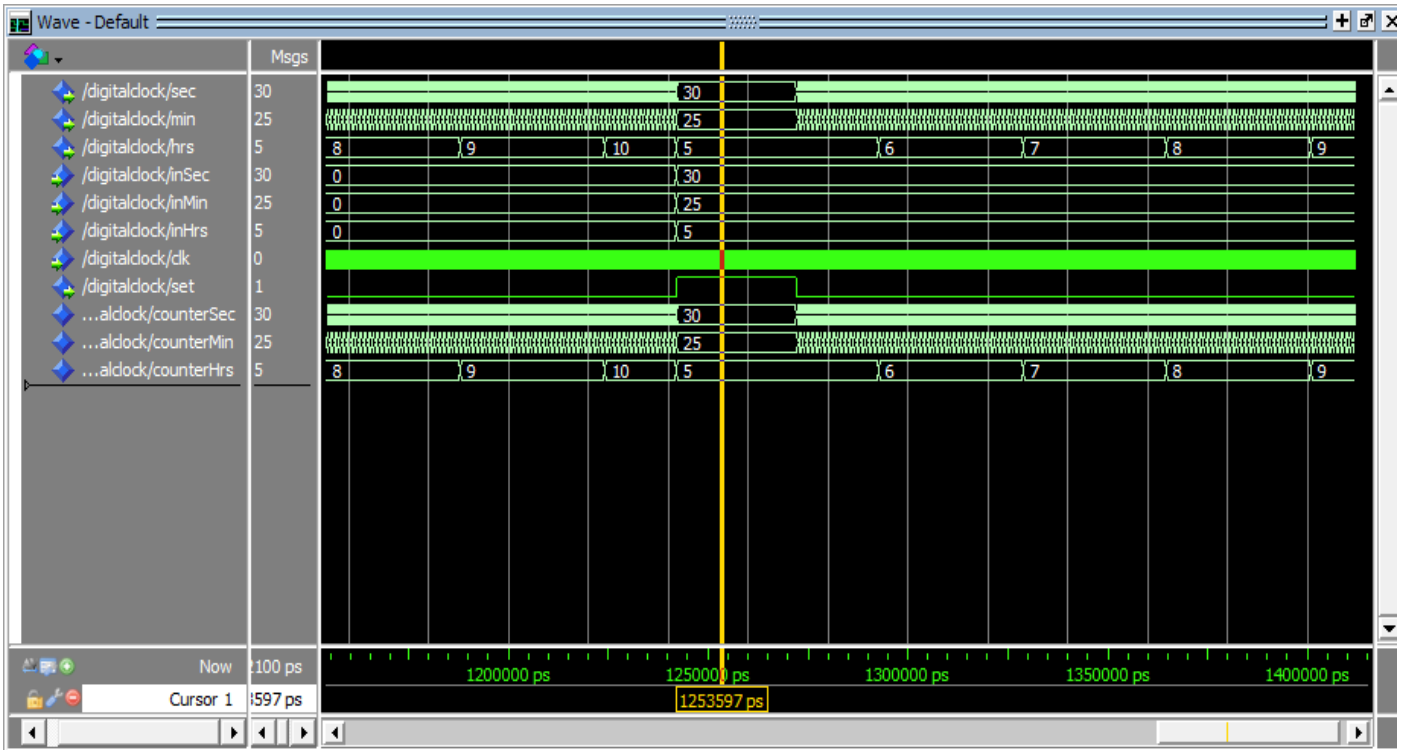
```
1   library IEEE;
2   use IEEE.STD_LOGIC_1164.ALL;
3
4   entity DigitalClock is
5      port(sec    :  out   integer range 0 to 60      :=  0;
6           min    :  out   integer range 0 to 60      :=  0;
7           hrs    :  out   integer range 0 to 24      :=  0;
8           inSec  :  in    integer range 0 to 60      :=  0;
9           inMin  :  in    integer range 0 to 60      :=  0;
10          inHrs  :  in    integer range 0 to 24      :=  0;
11          clk    :  in    std_logic    := '0';
12          set    :  buffer std_logic   := '0');
13  end DigitalClock;
14
15  architecture Behavioural of DigitalClock is
16  signal counterSec : integer range 0 to 60 := 0;
17  signal counterMin : integer range 0 to 60 := 0;
18  signal counterHrs : integer range 0 to 24 := 0;
19  begin
20     process(clk, set)
21     variable innercountSec : integer range 0 to 60 := 0;
22     variable innercountMin : integer range 0 to 60 := 0;
23     variable innercountHrs : integer range 0 to 24 := 0;
24     begin
25        if (set = '1') then
26           innercountSec := inSec;
27           innercountMin := inMin;
28           innercountHrs := inHrs;
29           if inSec >= 60 then
30              innercountSec := inSec - 60;
31              innercountMin := inMin + 1;
32           end if;
33           if innercountMin >= 60 then
34              innercountMin := innercountMin - 60;
35              innercountHrs := inHrs + 1;
36           end if;
37           if innercountHrs >= 24 then
38              innercountHrs := innercountHrs - 24;
39           end if;
40           counterSec <= innercountSec;
41           counterMin <= innercountMin;
42           counterHrs <= innercountHrs;
43           set <= '0';
44        elsif clk 'event and clk = '1' then
45           counterSec <= counterSec + 1;
46           if counterSec >= 59 then
47              counterMin <= counterMin + 1;
48              counterSec <= 0;
49              if counterMin >= 59 then
50                 counterHrs <= counterHrs + 1;
51                 counterMin <= 0;
52                 if counterHrs >= 23 then
53                    counterHrs <= 0;
54                    counterMin <= 0;
55                    counterSec <= 0;
56                 end if;
57              end if;
58           end if;
59        end if;
60     end process;
61     sec <= counterSec;
62     min <= counterMin;
63     hrs <= counterHrs;
64  end Behavioural;
```

The schematic

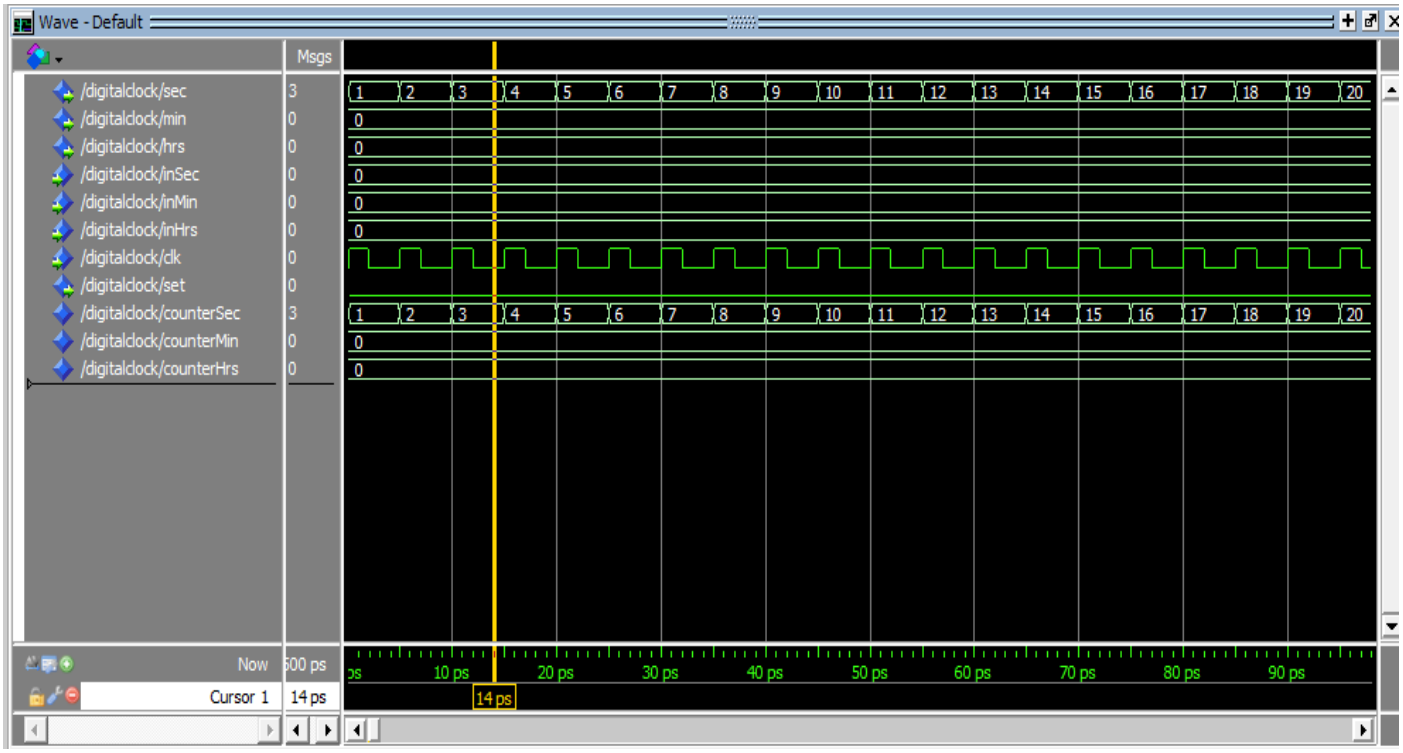


The screens of the results

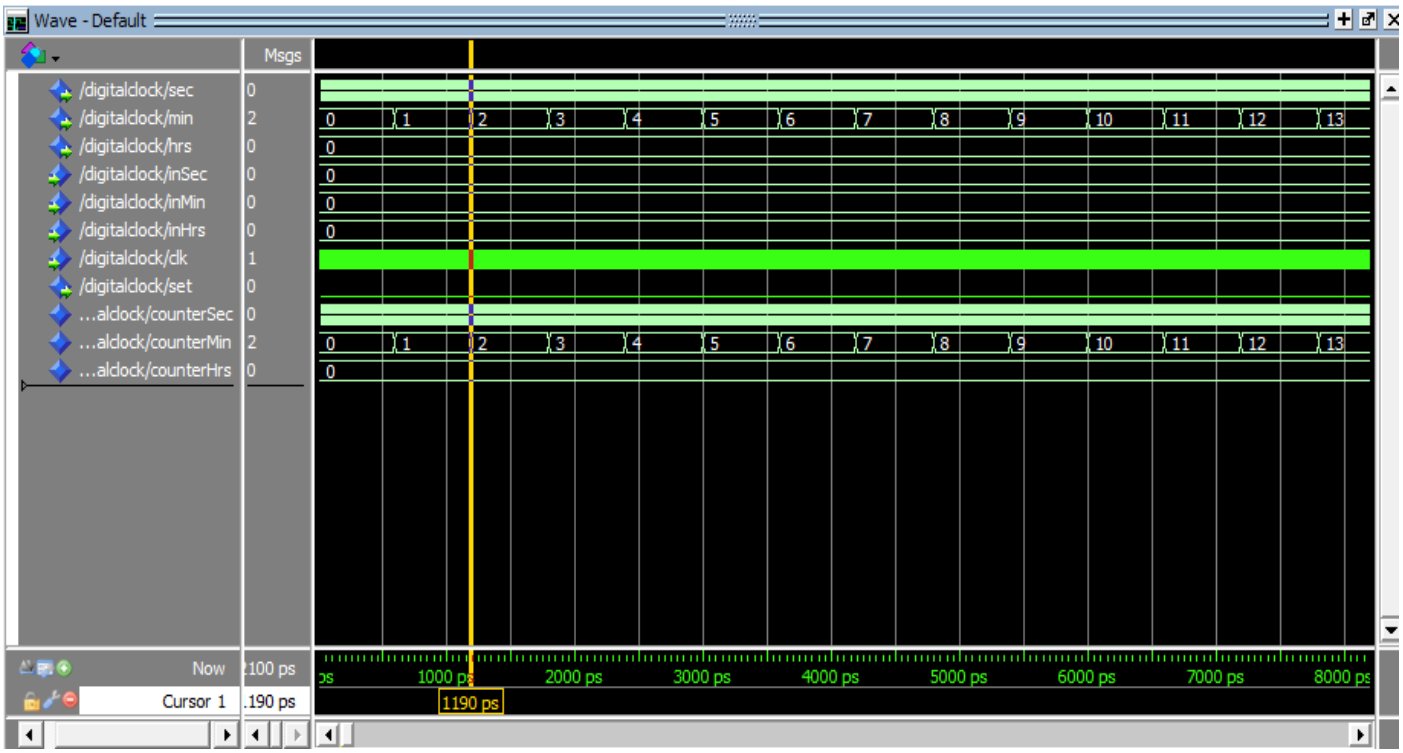
- The set ='1'



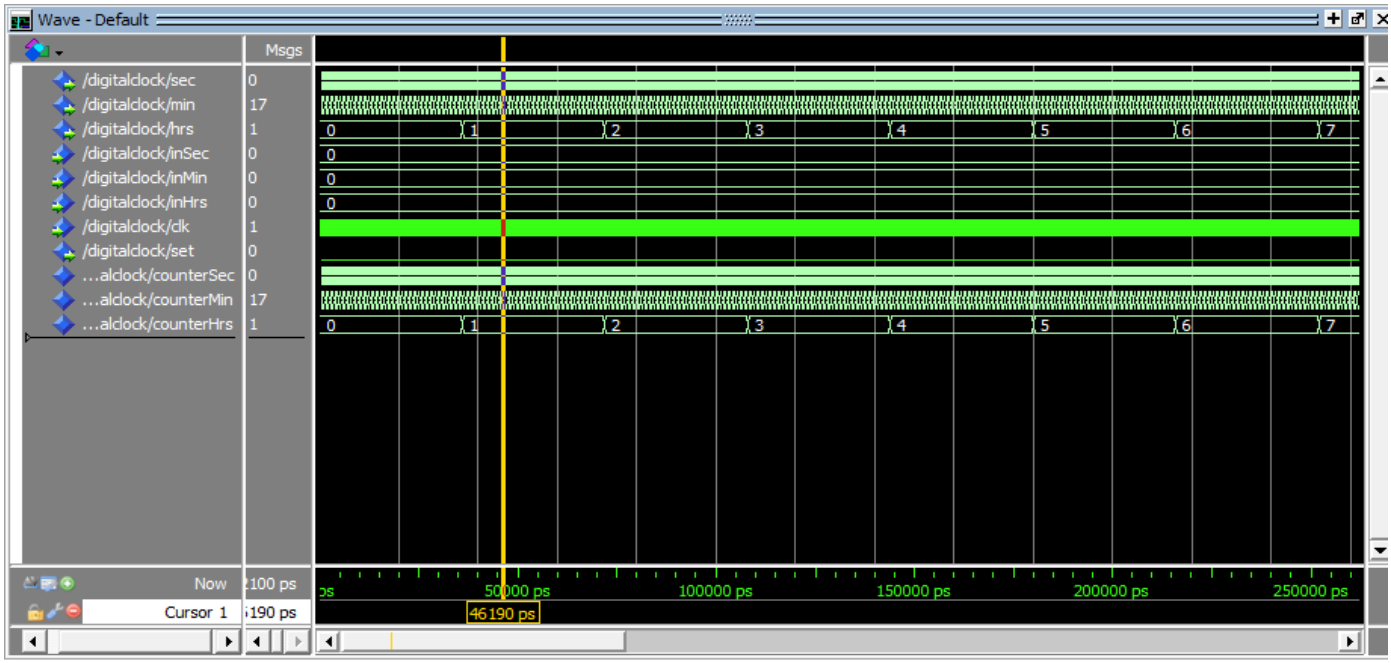
- The seconds count



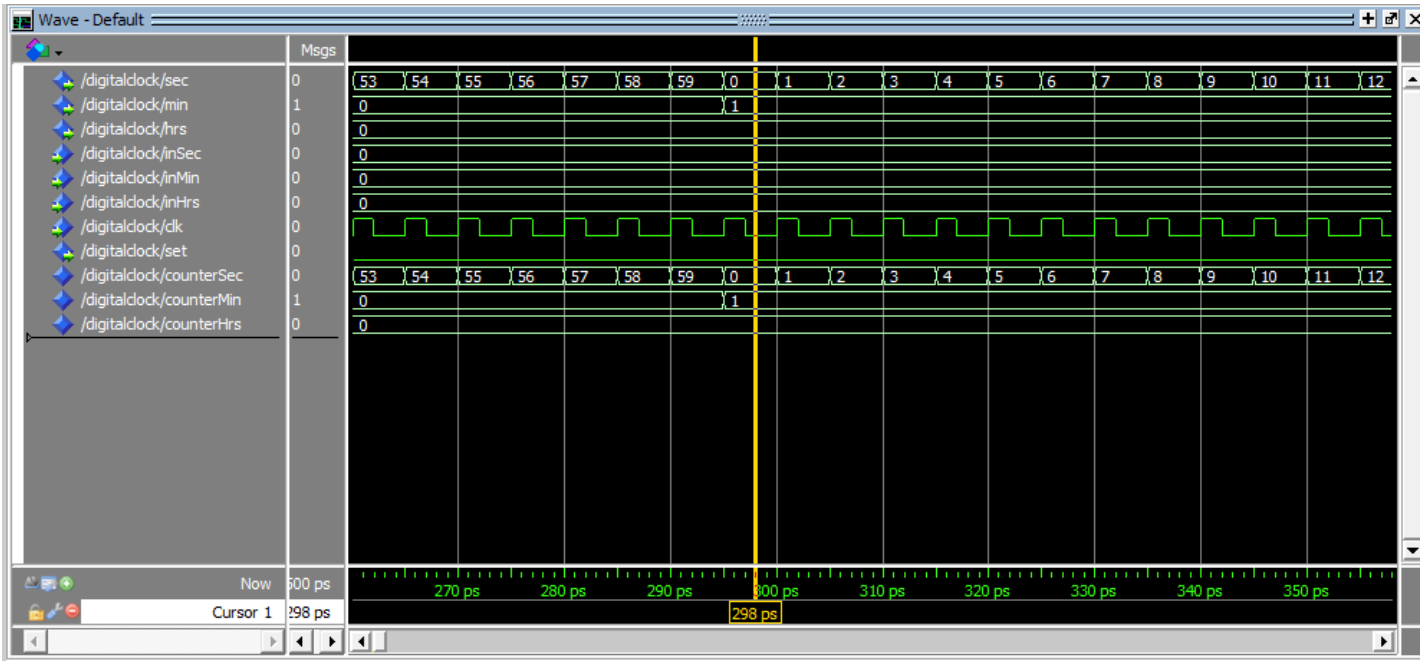
- The minutes count



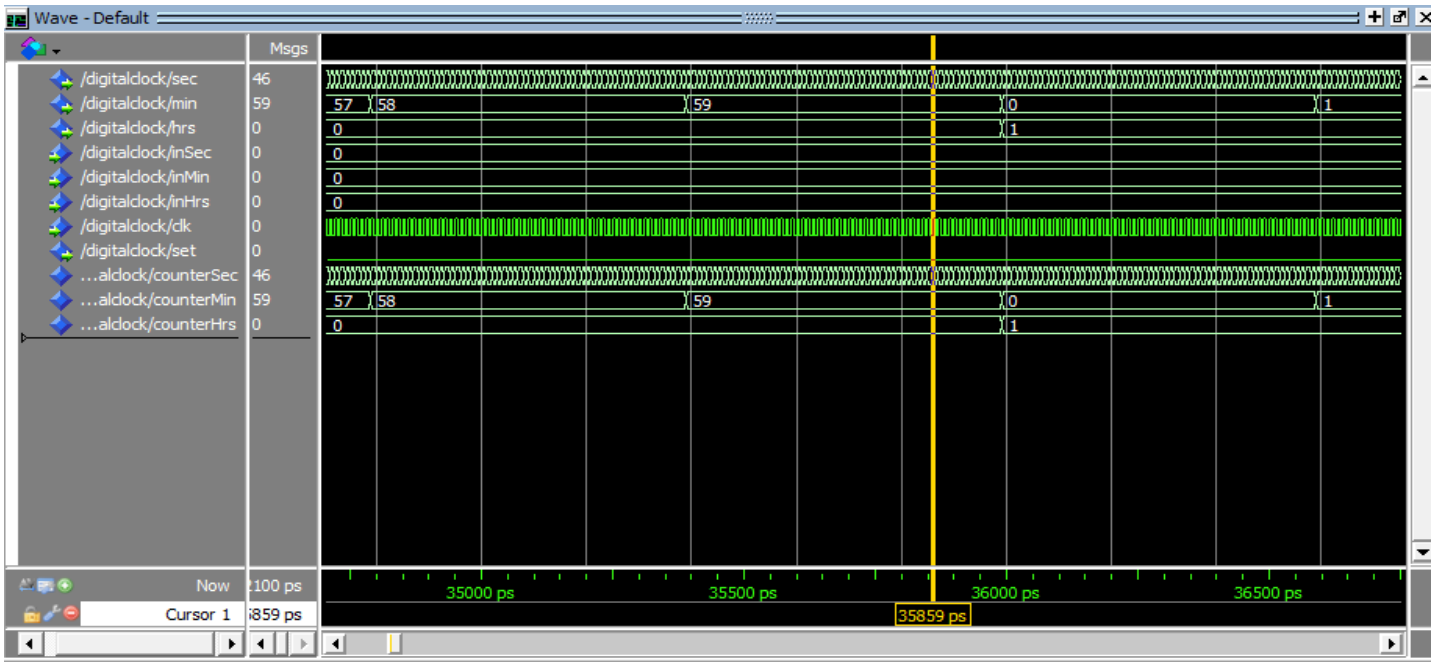
- The hours count



- The transition from seconds to minutes



- The transition from minutes to hours



- Resetting the clock after hour=24