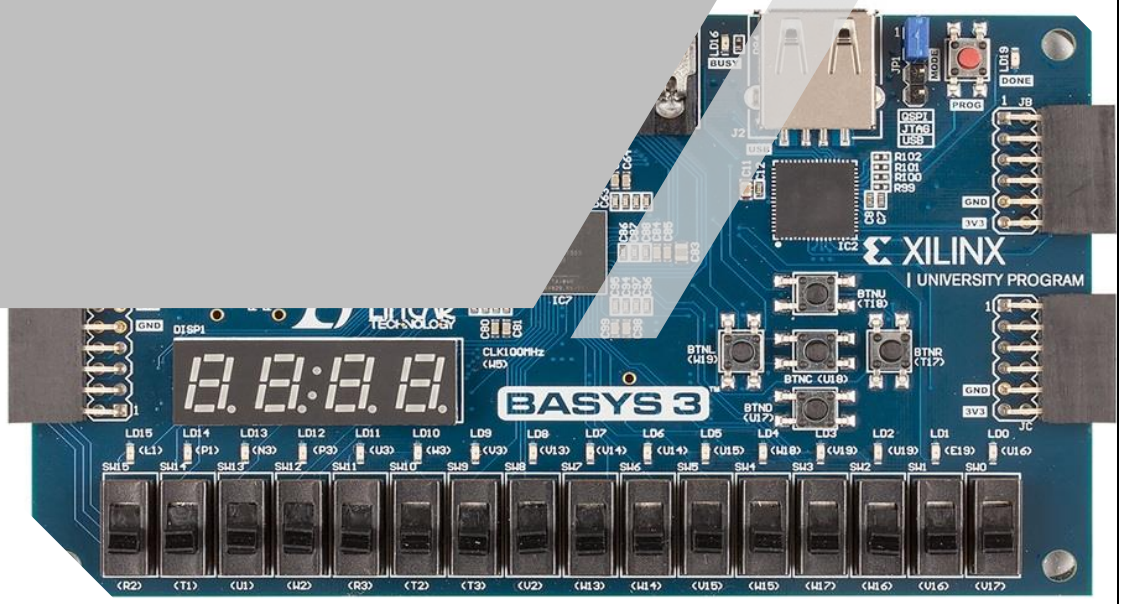
AUC SCHOOL OF SCIENCES AND ENGINEERING

# CSCE2301 – Digital Design I

## Project 2 Report

# Digital Alarm Clock



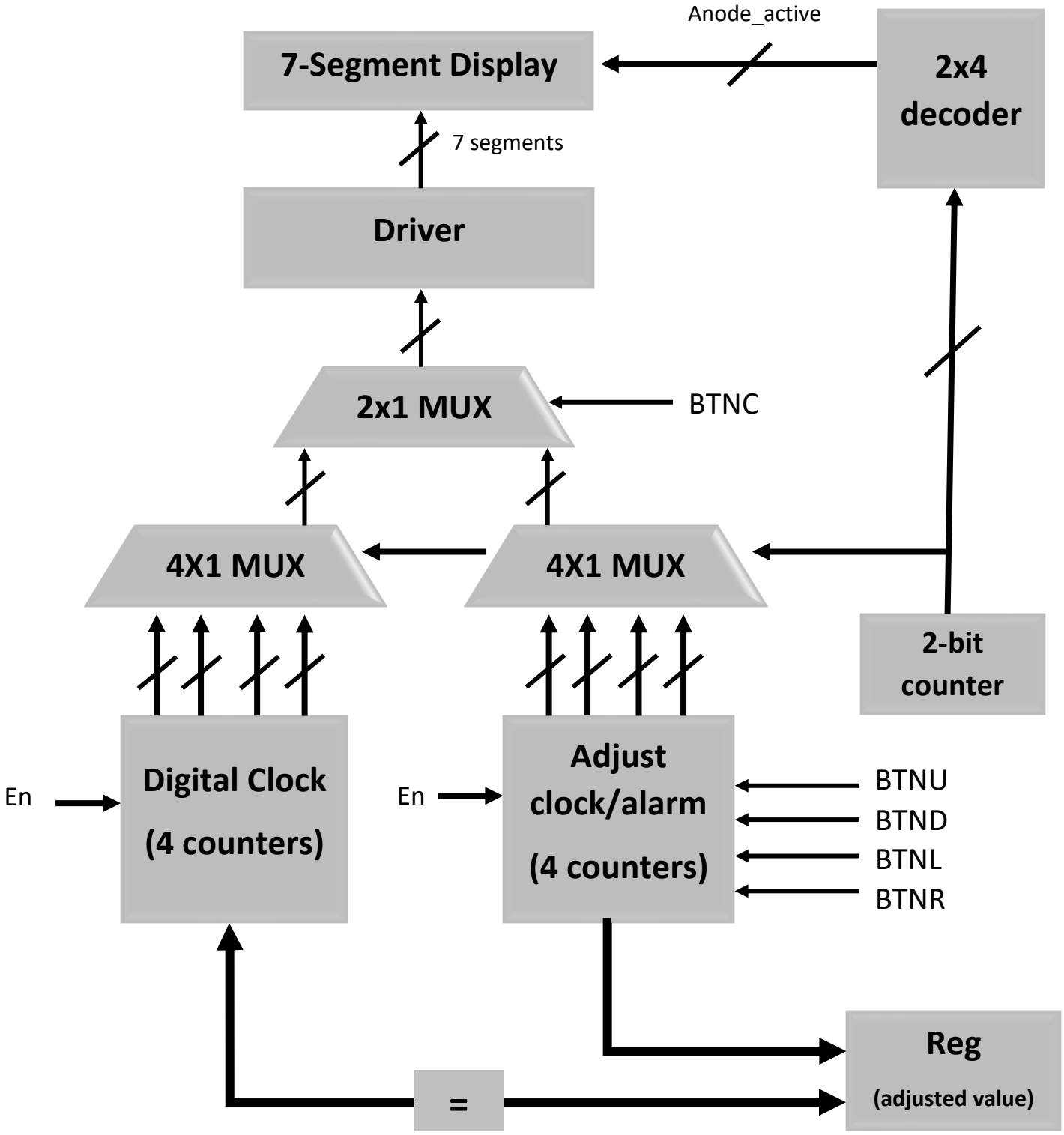

**Group members:**

| | |
|---|---|
| Mohammed Abuelwafa | 900172603 |
| Marwah Sulaiman | 900172284 |

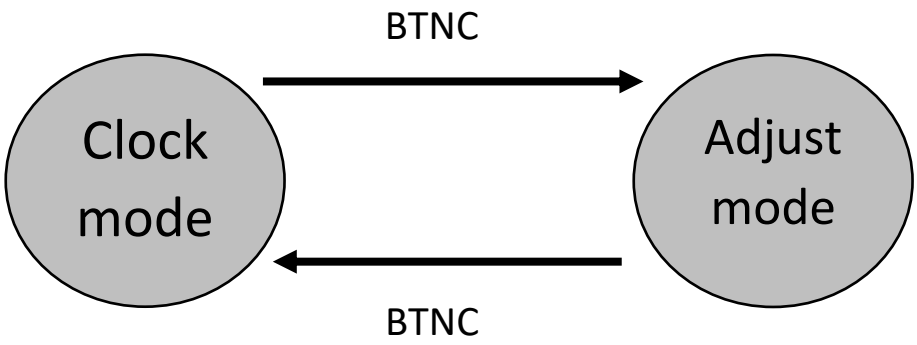
**Supervised by:** Dr. Mohamed Shalan

## 1. Design

## Block diagram:



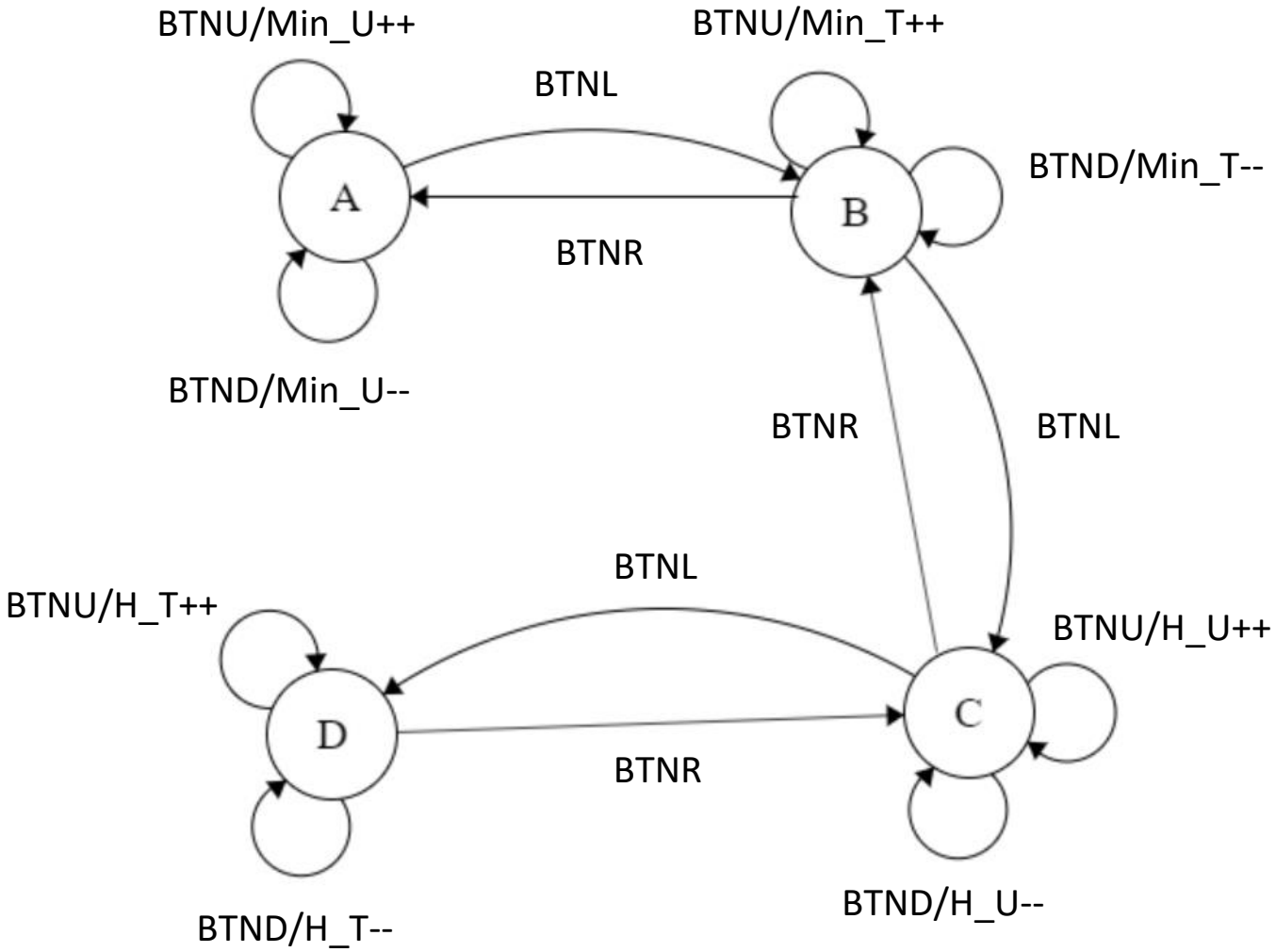
## FSM (general):




- LD0 is OFF
- The 2nd decimal point from the left shall blink with a frequency of 2Hz.
- To adjust the clock press BTNC for at least 2 seconds to enter the “adjust” mode.
- When the current time matches the set alarm time the LED blinks; hitting any button stops the blinking.

- LD0 is ON.
- The 2nd decimal point from the left does not blink.
- Pressing BTNC exits the “adjust” mode to the “clock/alarm” mode.
- Pressing BTNL or BTNR selects what to adjust: “time hour”, “time minute”, “alarm hour”, and “alarm minute” (in sequence). LD12, LD13, LD14 and LD15 states reflect the parameter to be adjusted. For example, LD13 is ON while adjusting “alarm hour”; pressing BTNL change the parameter to be adjusted to “time minute”, turns LD14 on and turns LD12 off.
- Pressing BTNU increment the selected parameter (e.g., “time hour”) o Pressing BTND decrement the selected parameter (e.g., “time hour”)

## FSM (adjust mode)



# Main modules :

Main:

The main module drives all the other modules. It contains a mod 2 binary counter, a digital clock module, an alarm module, display driver module, two 4x1 multiplexers and one 2x1 multiplexer and four registers. The flow of the design goes as the following. The main module has a binary counter whose count is initially set to zero which refers to the state of the digital clock, and this counter is enabled by BTNC which switches the state between the digital clock and the adjust mode. The outputs of the digital clock or the adjust module are forwarded to a 2x1 mux that chooses which one is going to get displayed according to the select value which is an output of the mod2 binary counter. In order to enable the leds , four enables are used to differentiate between the values alarm_hours, alarm_minutes , clock_hours , clock_minutes. The four registers are used to store the outputs of the adjust mode, and if they are alarm values then they are compared constantly with the values of the clock in order to blink LD0 and if they are clock values then they are loaded to the clock registers.

Digital Clock:

This module has six counters; two for seconds, two for minutes and two for hours. The enable of each counter is set according to the value of the previous counter in order to accommodate for change in all counters in the right time. Moreover, the digital clock module counters count according to a clock that comes out of a clock divider that has a frequency of 1 Hz. Furthermore, the digital clock module only outputs the outputs of four counters [Hour_tens, Hour_units, Min_tens,Min_units] of the six and an output *dot* which is responsible for blinking the decimal point. *Dot* is a time varying signal of a frequency of 2 Hz which is sent to the anode active in order to blink.

Adjust:

this module is composed of four counters, each is responsible for adjusting one of the following: Hours_tens, Hours_units, Min_Tens, Min_units. The driving modules of this module are binary counters that has two additional inputs which are BTNU and BTND in order to increment of decrement the values of the Counter.

Adjust_pos

This module is a counter that is considered with adjusting the position of the leds according to the values of BTNR, BTNL.

Debouncer_2s

This module is used to delay the detection of BTNC for 2 seconds by using 20 flip flops with a clock cycle of 0.1 s.

Push_det

This module is composed of three driving modules which are the synchronizer, the debouncer and the Rising_edge_detector

Seven_seg_proc

- This module is the display driver and it is structured as the following:
- The value of the seven segments as the following:

```
case (x)
4'b0000: segments = 7'b0000001 ;
4'b0001: segments = 7'b1001111 ;
4'b0010:segments = 7'b0010010 ;
4'b0011:segments = 7'b0000110 ;
4'b0100:segments = 7'b1001100 ;
4'b0101:segments = 7'b0100100 ;
4'b0110:segments = 7'b0100000 ;
4'b0111:segments = 7'b0001111 ;
4'b1000:segments = 7'b0000000 ;
4'b1001:segments = 7'b0000100 ;

Endcase
```

- The values of the anode_active in order to display the decimal point

- This Module is driven by a 2-bit binary counter in the main that has a clock of 100hz in order to display the numbers with high frequency in order not to be noticed by the user.

## Implementation:

- The Control unit that produces all the control signals was implemented using counters that switches between states for simplicity

- The Push buttons produce a one-time rising input, so in order to stop the blinking led with one press an FSM was designed in order to drive this part as per the following:

```
assign aa =( ((U|D|R|L)  && sel_clk ) );
always @ (posedge clk or posedge rst )begin
if(rst )
state<= 0;
else
state<= nextstate;
end

always @ ( aa or state ) begin
case (state)
0:
case(aa)
0: nextstate =0;
1: nextstate =1;
```

```
endcase
1:
case(aa)
0: nextstate = 1;
1: nextstate = 1;
endcase

endcase

case(enalarm_H | enalarm_M )
0:nextstate =0;
endcase

end

assign bb = (state ==1);
```

# Implementation Issues:

- the user must adjust the clock firstly before adjusting the alarm in order to be able to stop the alarm when it rings
- BTNC doesn't stop the alarm (BTNU,BTND,BTNR,BTNL do )
- LD0 is not On in the adjust mode

# Validation activities:

1- Observing the clock count and making sure it stops at 23:59 and restart with 00:00

2- Comparing the clock count to a real clock to validate our used frequency/clock divider

3- Switching between the two modes using BTNC

4- Setting different values to adjust the clock

5- Setting different values to adjust the alarm

6- Observing the LED responsible for the alarm notification and stopping it using any button from BTNU,BTND,BTNR,BTNL.

7- Testing the reset input