Report

Team 82

Ahmed Hatem 49-17964 T17
ahmed.semeda@student.guc.edu.eg

Kareem Eid 49-19194 T25
Kareem.eid@student.guc.edu.eg

Abdelrahman Hussein 49-17771 T08
Abdelrahman.mahmoud@student.guc.edu.eg

Adhm Mohamed 49-19384 T17
Adhm.ahmed@student.guc.edu.eg

Mahmoud Hossameldin 49-14403 T22
mahmoud.hossameldin@student.guc.edu.eg

Idea and parts:

The idea of the project is to design a smart train barrier which closes when a train approach it and open afterwards when the train leaves. We used 2 IR-censors: one to notify us if the train is approaching and another one to let us know if the train has left, a push button to force opening or closing the barrier in case of emergency (if there is any problem with the censors), a servo motor to move the barrier, a buzzer to notify people crossing the railway, a ldr sensor to measure whether it's dark or bright outside and a set of leds to make the barrier visible in darkness.

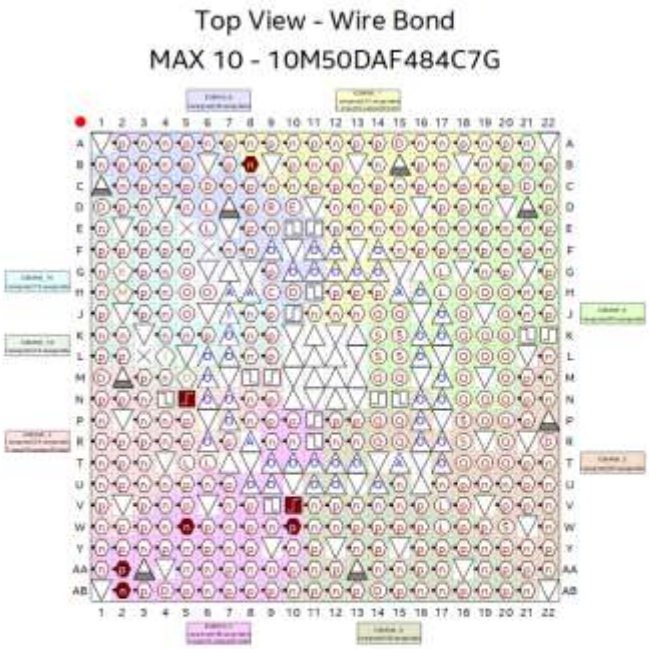
Implementation:

- The IR-censors and the button are connected directly to the inputs of the fpga.
- Arduino is used to read the value of the ldr censor (analog) and illuminate the leds according to the enable signal coming from the fpga(logic high when the train is coming).
- The fpga is connected to 2 outputs (servo motor and buzzer).

Results:

- The first IR censor sends a signal to the fpga to make it close the barrier and turn on the buzzer.
- The second IR censor sends a signal to the fpga to make it open the barrier and turn off the buzzer.

Pin planner:



Named: * Edit: Input

Node Name	Direction	Location	I/O Bank	VREF Group	Itter Locatior	I/O Standard	Reserved	Jrrent Streng	Slew Rate	ifferential Pai	ict Preservati
ardsignal	Output	PIN_W5	3	B3_N0	PIN_W5	2.5 V		12mA ..ault)	2 (default)		
btn	Input	PIN_B8	7	B7_N0	PIN_B8	2.5 V		12mA ..ault)			
buzz	Output	PIN_AA2	3	B3_N0	PIN_AA2	2.5 V		12mA ..ault)	2 (default)		
clk	Input	PIN_N5	2	B2_N0	PIN_N5	2.5 V		12mA ..ault)			
first	Input	PIN_W10	3	B3_N0	PIN_W10	2.5 V		12mA ..ault)			
pwmout	Output	PIN_AB2	3	B3_N0	PIN_AB2	2.5 V		12mA ..ault)	2 (default)		
second	Input	PIN_V10	3	B3_N0	PIN_V10	2.5 V		12mA ..ault)			
<<new node>>											

Main code:

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.std_logic_unsigned.all;
use work.servo_package.all;


entity final is
port(clk, btn, first, second: in STD_logic;
                pwmout, buzz, ardsignal: buffer std_logic);
end final;
```

```
Architecture arch of final is
signal position : integer range 0 to 999;
begin
process(first, second, btn)
begin
if(btn'event and btn = '1') then
        if(buzz = '0') then
                position <= 860;
                buzz <= '1';
        else
                position <= 500;
                buzz <= '0';
        end if;
end if;
if(first = '0') then
        position <= 860;
        buzz <= '1';
end if;
if (second = '0') then
        position <= 500;
        buzz <= '0';
end if;
end process;
ardsignal <= buzz;
s1 : servo port map(clk,'0', position, pwmout);
end arch;
```

Servo motor code:

```
library ieee;
```

```
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.math_real.round;

entity servo is
  port (
    clk : in std_logic;
    rst : in std_logic;
    position : in integer range 0 to 999;
    pwm : out std_logic
  );
end servo;

architecture rtl of servo is

  constant max_count : integer := 30000;
  constant cycles_per_step : positive := 30;

  constant counter_max : integer := 200000;

  signal counter : integer range 0 to counter_max;

  signal want_cycle : integer range 0 to max_count;

begin

  COUNTER_PROC : process(clk)
  begin
    if rising_edge(clk) then
      if rst = '1' then
```

```
      counter <= 0;

    else
      if counter < counter_max then
        counter <= counter + 1;
      else
        counter <= 0;
      end if;

    end if;
  end if;
end process;

PWM_p : process(clk)
begin
  if rising_edge(clk) then
    if rst = '1' then
      pwm <= '0';

    else
      pwm <= '0';

      if counter < want_cycle then
        pwm <= '1';
      end if;

    end if;
  end if;
end process;
```

```
  DUTY_CYCLE_PROC : process(clk)
  begin
    if rising_edge(clk) then
      if rst = '1' then
        want_cycle <= 0;

      else
        want_cycle <= position * cycles_per_step;

      end if;
    end if;
  end process;

end architecture;

library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.math_real.round;

package servo_package is
component servo
  port (
    clk : in std_logic;
    rst : in std_logic;
    position : in integer range 0 to 999;
    pwm : out std_logic
  );
  end component;
end servo_package;
```

Arduino code:

```
const int ledPin = 8;
const int ldrPin = A0;
const int ardsignal = 7;
void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
  pinMode(ldrPin, INPUT);
  pinMode(ardsignal, INPUT);
}

void loop() {
  int ldrStatus = analogRead(ldrPin);
  bool train = digitalRead(ardsignal);
  if(train){
  if (ldrStatus <= 80)
  {
    digitalWrite(ledPin, HIGH);
    Serial.print("Its Dark, Turn on the LED:");
    Serial.println(ldrStatus);

  }
  else
  {
    digitalWrite(ledPin, LOW);
    Serial.print("Its Bright, Turn off the LED:");
    Serial.println(ldrStatus);
  }
  }
```

```
  else{
    digitalWrite(ledPin, LOW);
  }

}
```

Circuits: