Mastering Embedded Systems Online Diploma

First Term Final Project 1

Pressure detection

Name: Mohamed Ahmed Samir

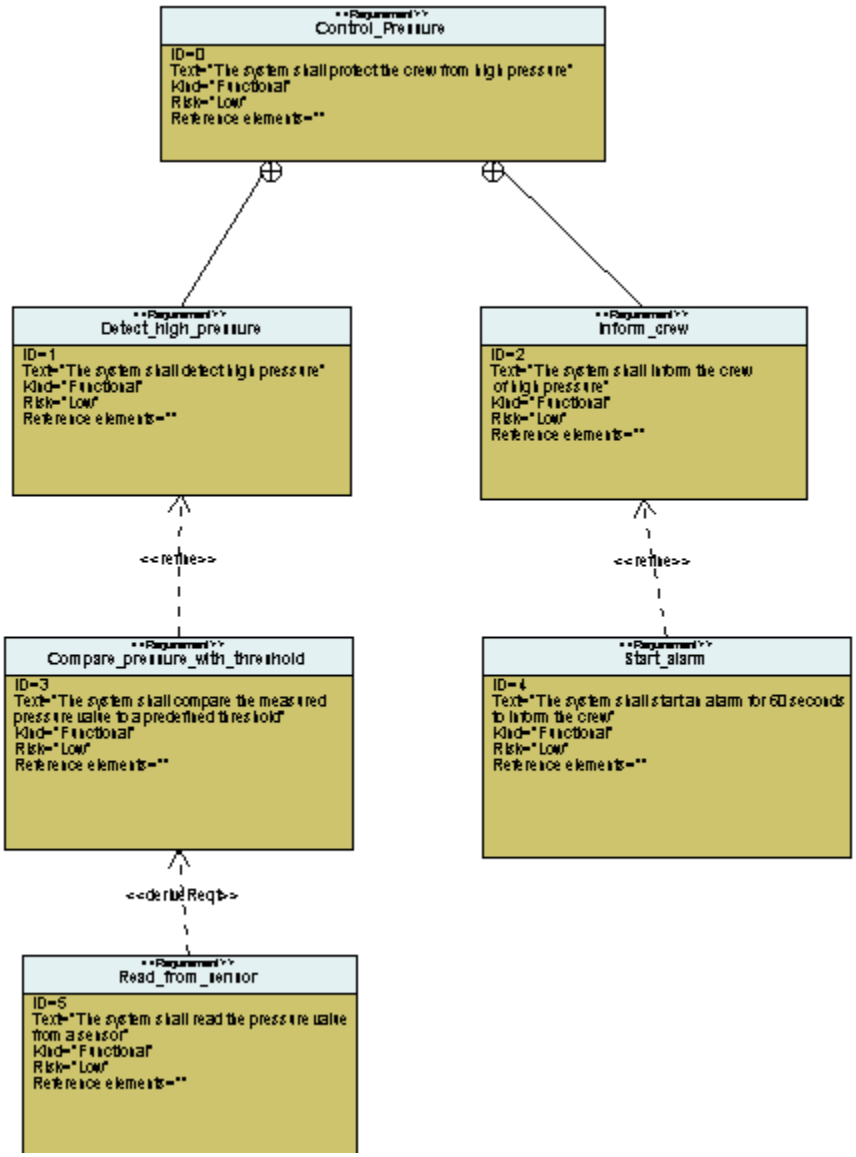
Email: mm.ahmedd504@gmail.com

Problem description

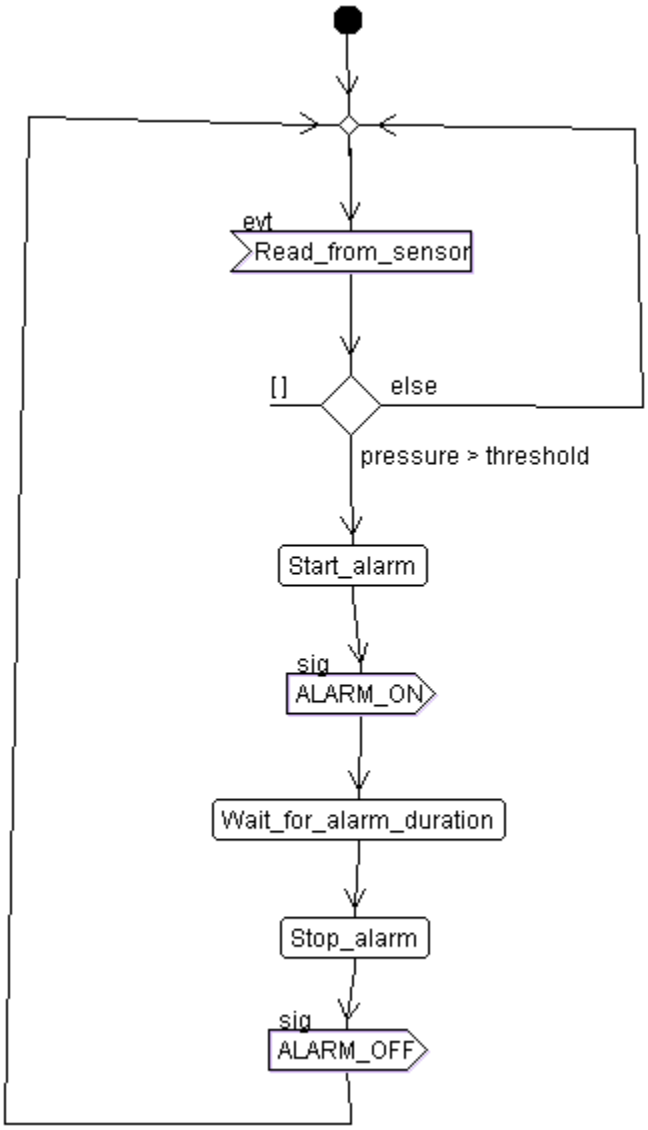
A client expects you to deliver the software of the following system specifications (from the client):

1. A pressure controller informs the crew of a cabin with an alarm when the pressure exceeds 20 bars in the cabin.
2. The alarm duration equals 60 seconds.

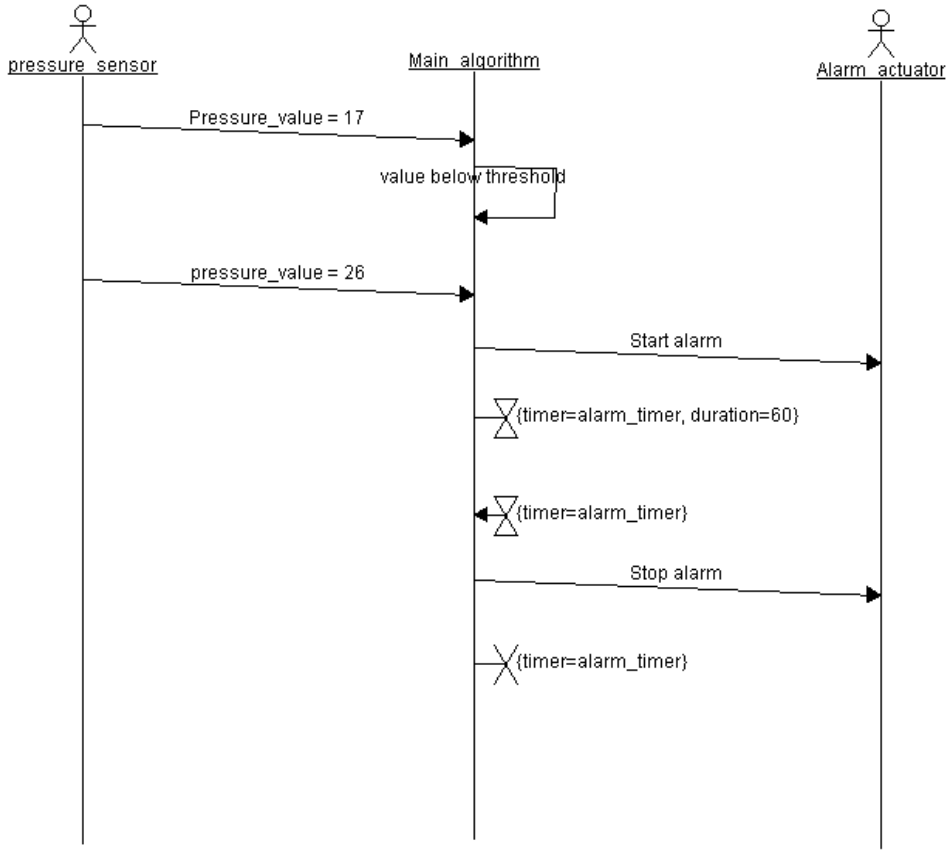
Requirements Diagram



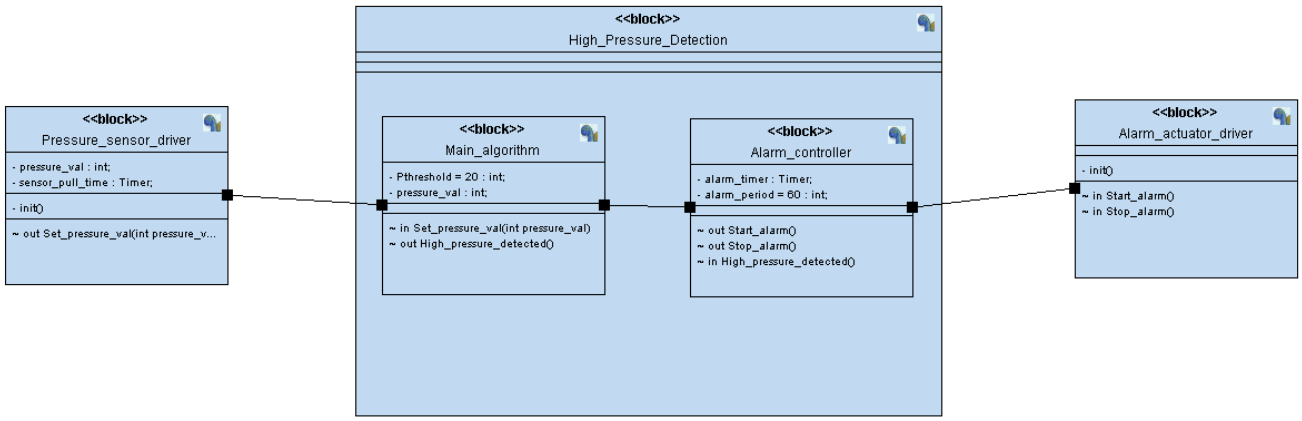
Activity Diagram



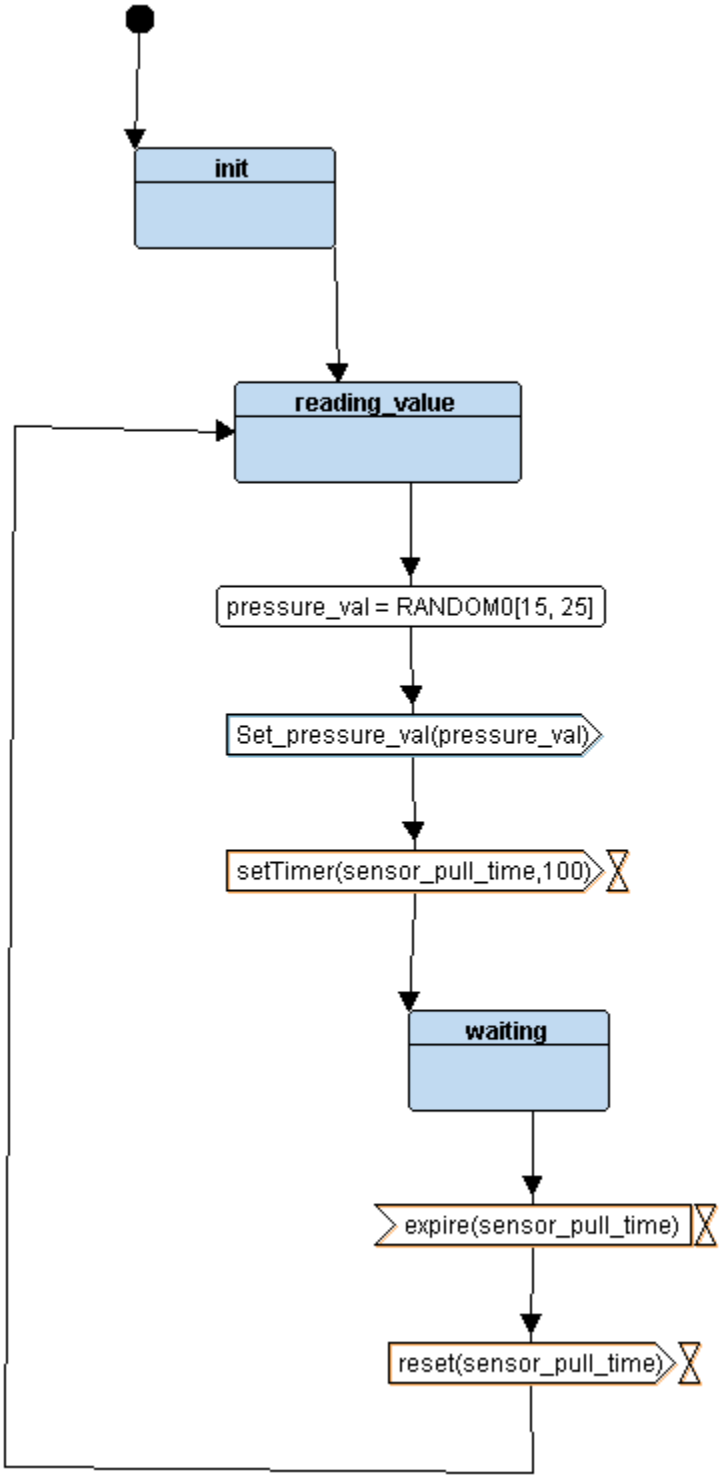
Sequence Diagram



Block Diagram

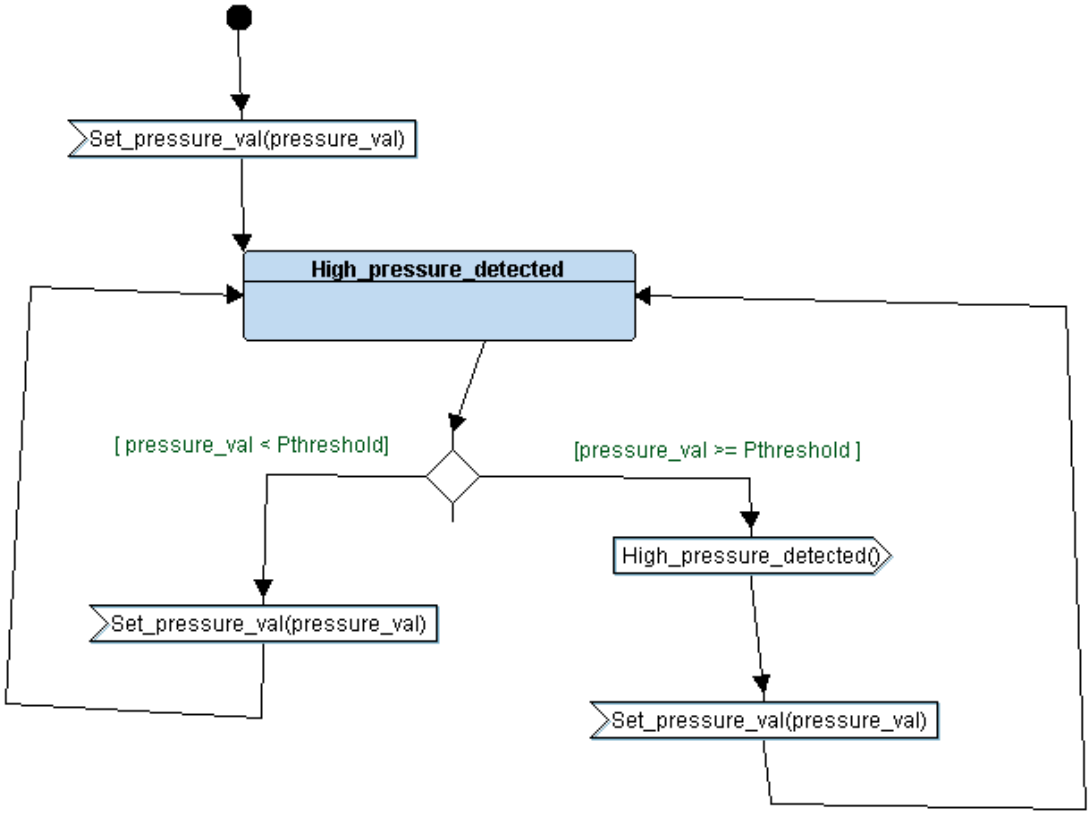


Pressure sensor state Diagram



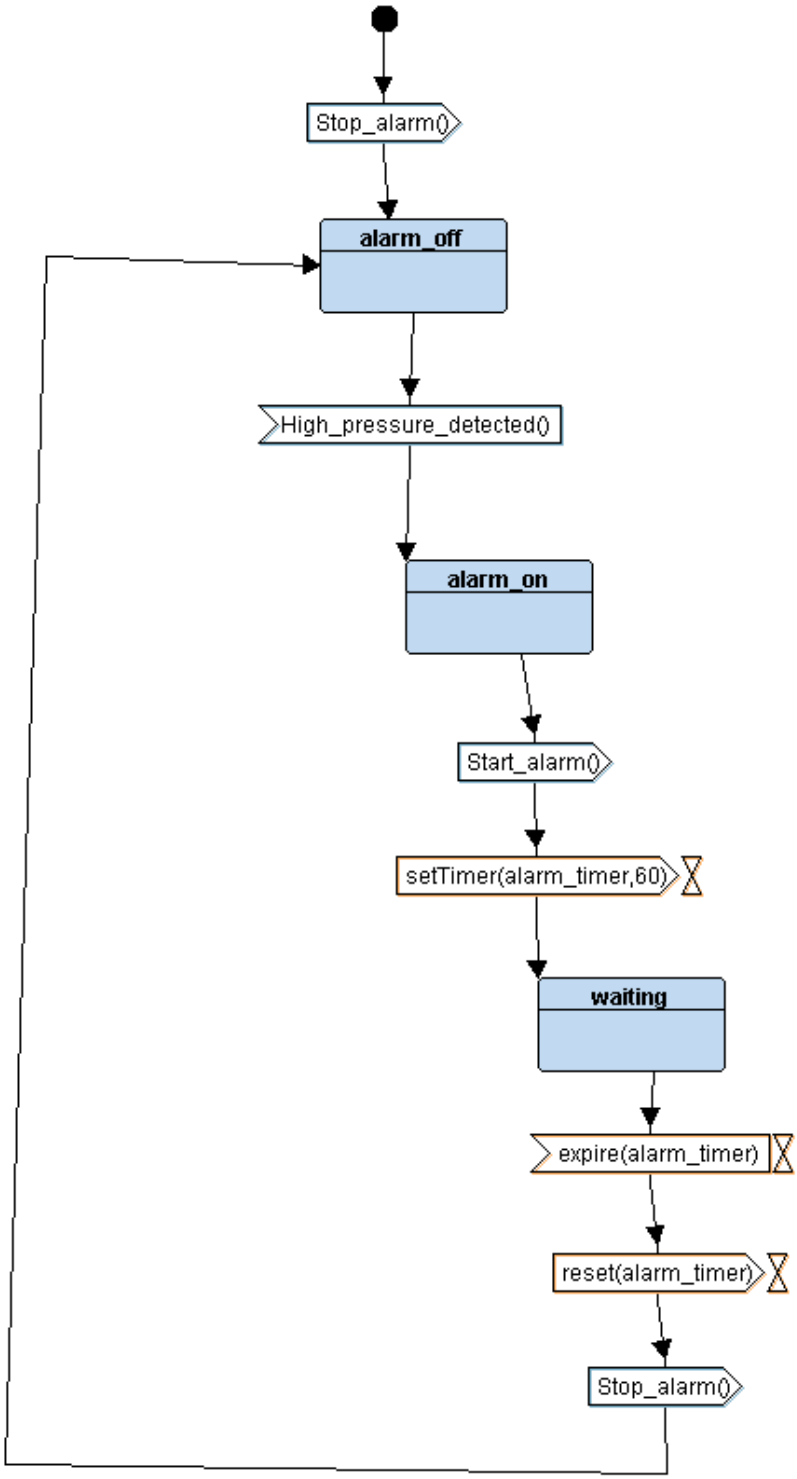
```c
#include "pressure_sensor.h"
#include "driver.h"
#include "stdlib.h"
#define Sensor_pull_time 100000

void (*PS_state)();
int pval = 0;

STATE_Define(PS_reading){
    Pressure_Sensor_State_Id = PS_reading;
    pval = getPressureVal();
    set_pressure_val(pval);
    PS_state = STATE(PS_waiting);
}

STATE_Define(PS_waiting){
    Pressure_Sensor_State_Id = PS_waiting;
    Delay(Sensor_pull_time);
    PS_state = STATE(PS_reading);
}


void pressure_sensor_init(){
    PS_state = STATE(PS_reading);
}


```

Main algorithm state Diagram


Set_pressure_val(pressure_val)
High_pressure_detected
[pressure_val < Pthreshold]
[pressure_val >= Pthreshold]
High_pressure_detected()
Set_pressure_val(pressure_val)
Set_pressure_val(pressure_val)

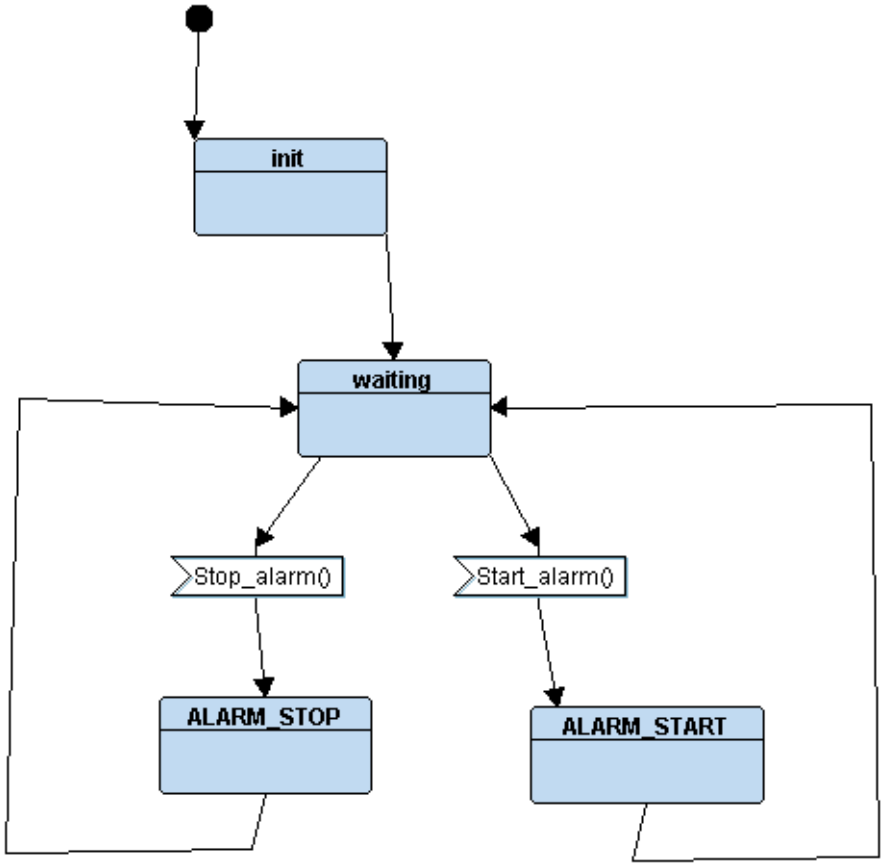
```c
#include "detection.h"

void (*detection_state)();

int current_pressure_value=0;
int pressure_threshold = 20;


void set_pressure_val(int pval){
    current_pressure_value = pval;
}


STATE_Define(pressure_detected){

    if(current_pressure_value > pressure_threshold){
        high_pressure();

    }

    detection_state = STATE(pressure_detected);
}

```

Alarm controller state Diagram



```c
#include "Alarm_controller.h"
#include "driver.h"


void (*alarm_controller_state)();

void high_pressure(){
    alarm_controller_state = STATE(alarm_on);
}


STATE_Define(alarm_on){
    Alarm_Controller_State_Id = alarm_on;
    start_alarm();
    alarm_controller_state = STATE(alarm_off);
}

STATE_Define(alarm_off){
    Alarm_Controller_State_Id = alarm_off;
    stop_alarm();
    alarm_controller_state = STATE(waiting);
}

STATE_Define(waiting){
    Alarm_Controller_State_Id = waiting;

}

```

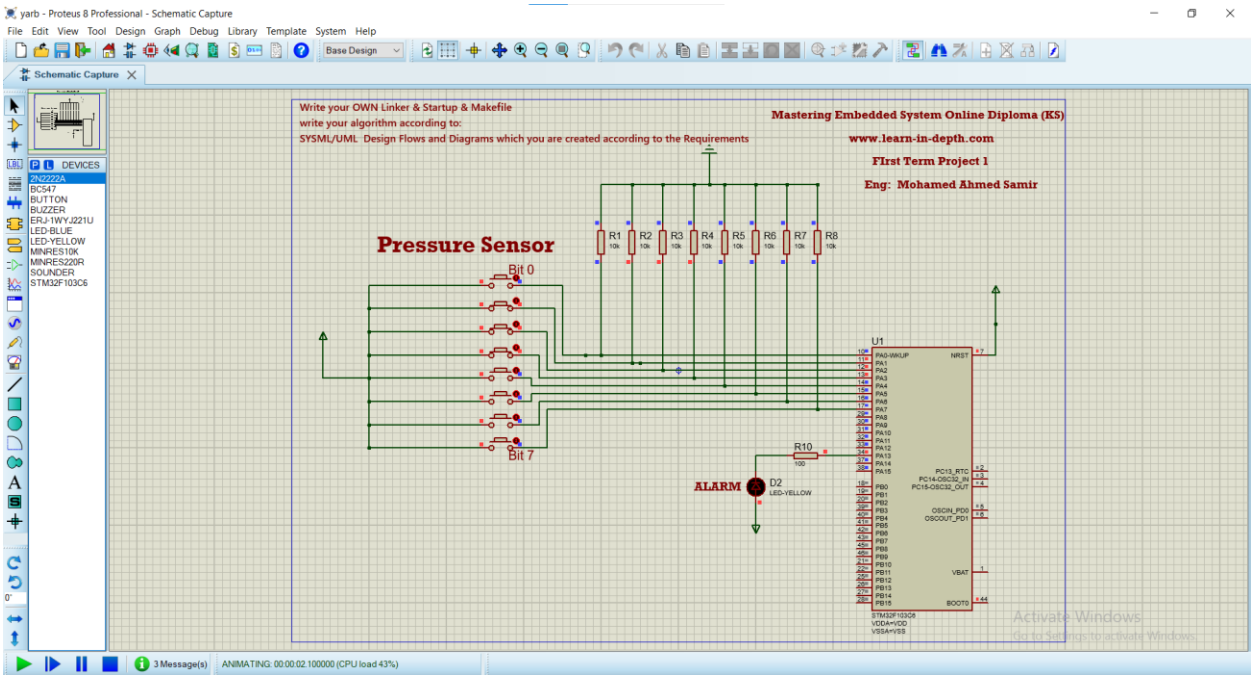
Alarm actuator state Diagram


init
waiting
Stop_alarm()
Start_alarm()
ALARM_STOP
ALARM_START

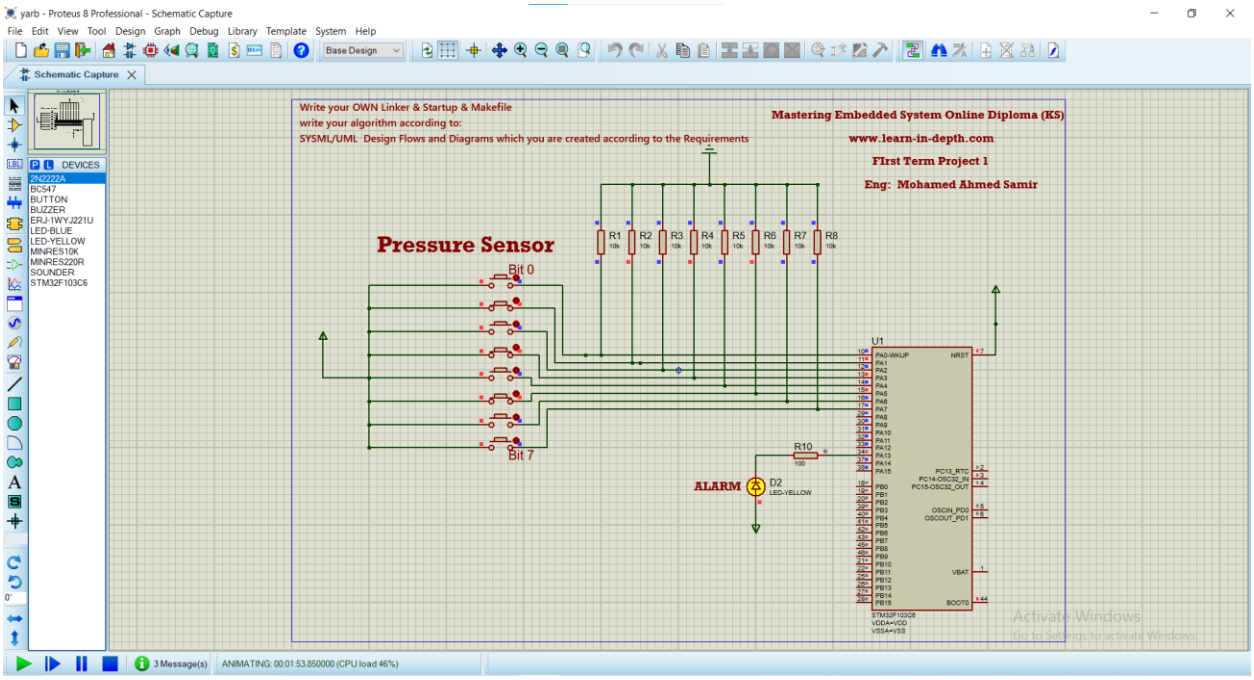
```c
#include "Alarm_actuator.h"
#include "driver.h"

#define alarm_duration 2000000

void (*alarm_actuator_state)();

void start_alarm(){
    alarm_actuator_state = STATE(alarm_start);
}

void stop_alarm(){
    alarm_actuator_state = STATE(alarm_stop);
}

STATE_Define(alarm_start){
    Alarm_Actuator_State_Id = alarm_start;
    Set_Alarm_actuator(0);
    Delay(alarm_duration);
    alarm_actuator_state = STATE(alarm_waiting);
}

STATE_Define(alarm_stop){
    Alarm_Actuator_State_Id = alarm_stop;
    Set_Alarm_actuator(1);
    alarm_actuator_state = STATE(alarm_waiting);
}

STATE_Define(alarm_waiting){
    Alarm_Actuator_State_Id = alarm_waiting;



}


void alarm_actuator_init(){
    alarm_actuator_state = STATE(alarm_waiting);
}
```

Simulation results

Pressure = $(00001110)_2 = 14$ which is less than the threshold 20



Pressure = $(00101010)_2 = 42$ which is higher than the threshold 20



The alarm stays on for 1 minute then turns off if the pressure becomes less than 20.