PRESSURE CONTROLLER PROJECT REPORT

Name:Hadyabdelhady

Assignment:1

Topic:FirstTerm Project1

CONTENTS:

- Case study
- Methodology
- Requirement Diagram
- Space Exploration (HW/SW Partitioning)
- System Analysis: Use Case Diagram
- System Analysis: Activity Diagram
- System Analysis: Sequence Diagram
- System Design
- System Design Simulation
- A brief look at source files
- Sections and symbols for each object file
- Sections and symbols for final executable file
- Finding the entry point using readelf utility
- A brief look at the map file and symbols
- Software Usage and Hardware Simulation

Case Study:

A pressure controller informs the crew of a cabin with an alarm when

the pressure exceeds 20 bars in the cabin.

The alarm duration equals 60 seconds.

Keep track of the measured values.

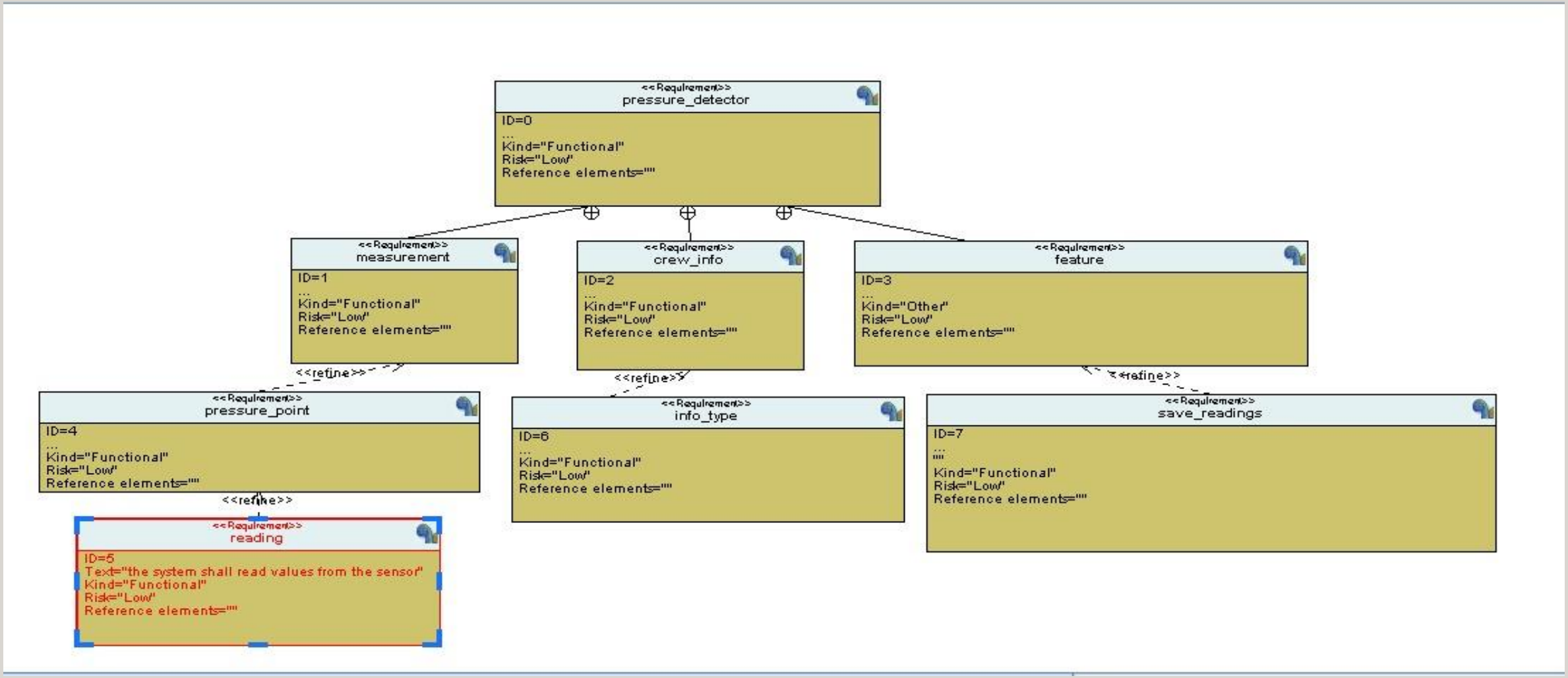
Assumptions:

- The system setup and shutdown procedures are not modeled.
- The system maintenance is not modeled.
- The pressure sensor never fails.
- The alarm never fails.
- The system never faces power cut.

Methodology:

- Since the system has multiple modules that are no easy to integrate,
- the system will use a testing-based model like v-model. Every phase in
- this project will be tested and especially the implementation phase.
- Each software module will be implemented and unit-tested separately
- then integrated and integration testing will be performed.

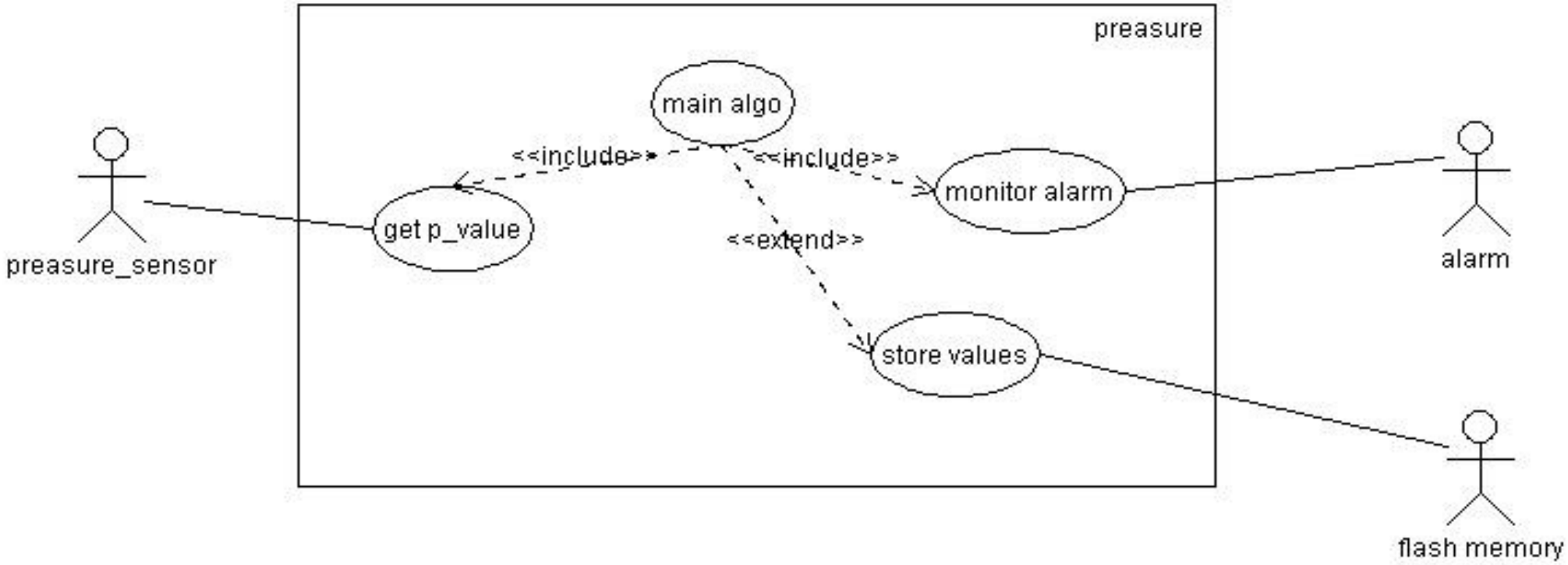
REQUIREMENT DIAGRAM:



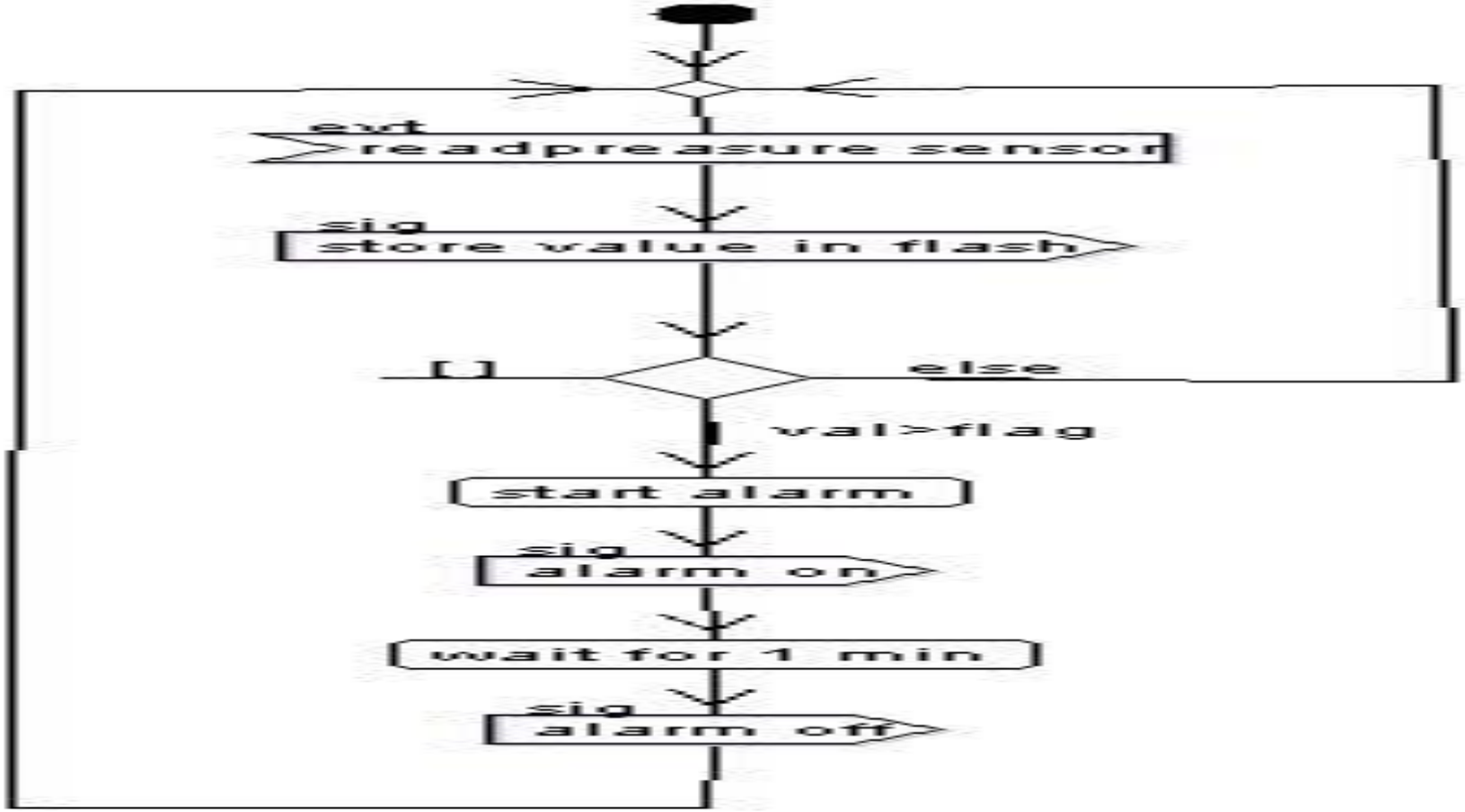
Space Exploration (HW/SW Partitioning):

- For the hardware, we have STM32 microcontroller with a cortex-m3
- processor that will be more than enough for this application.

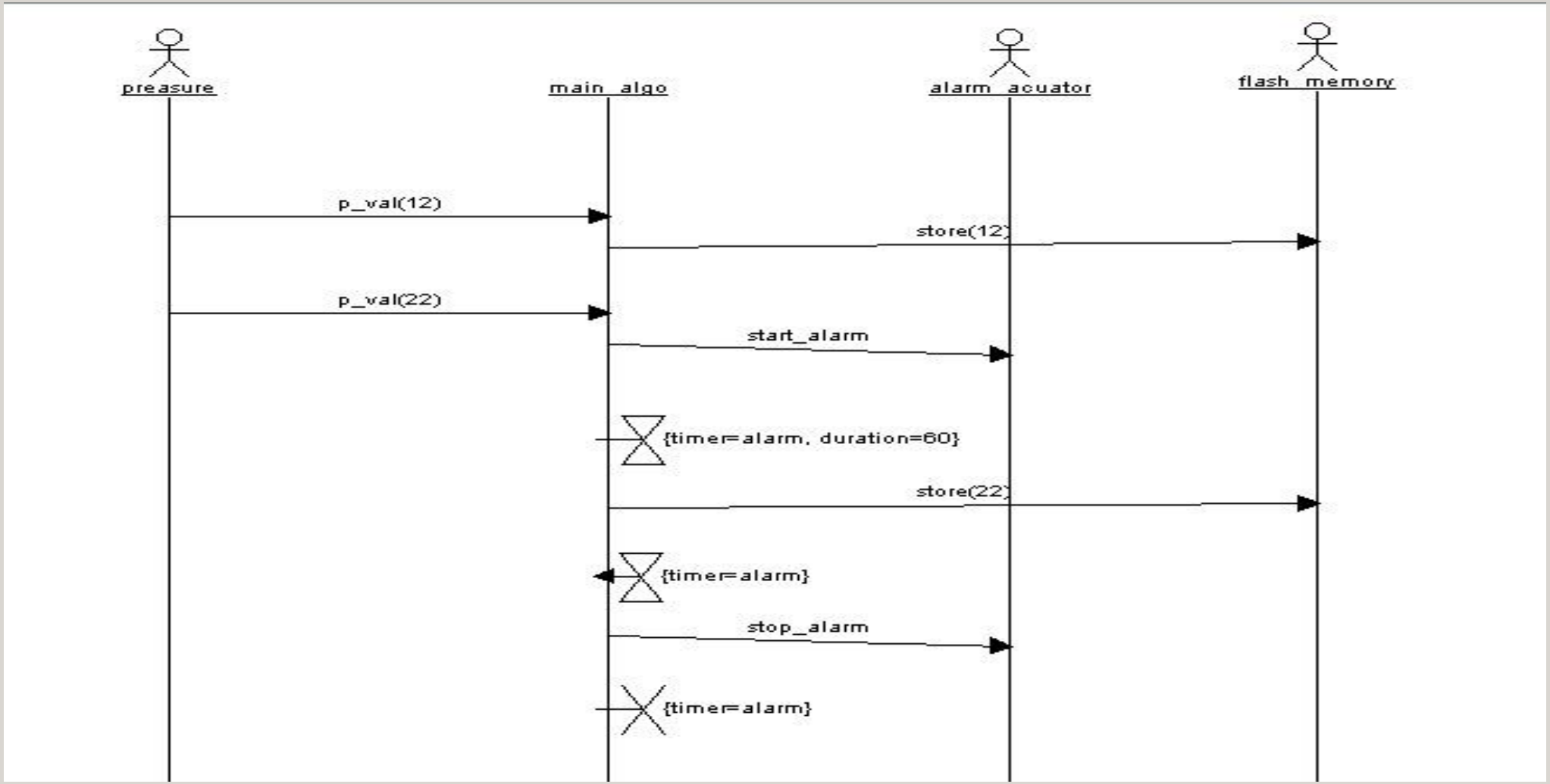
SYSTEM ANALYSIS: USE CASE DIAGRAM:



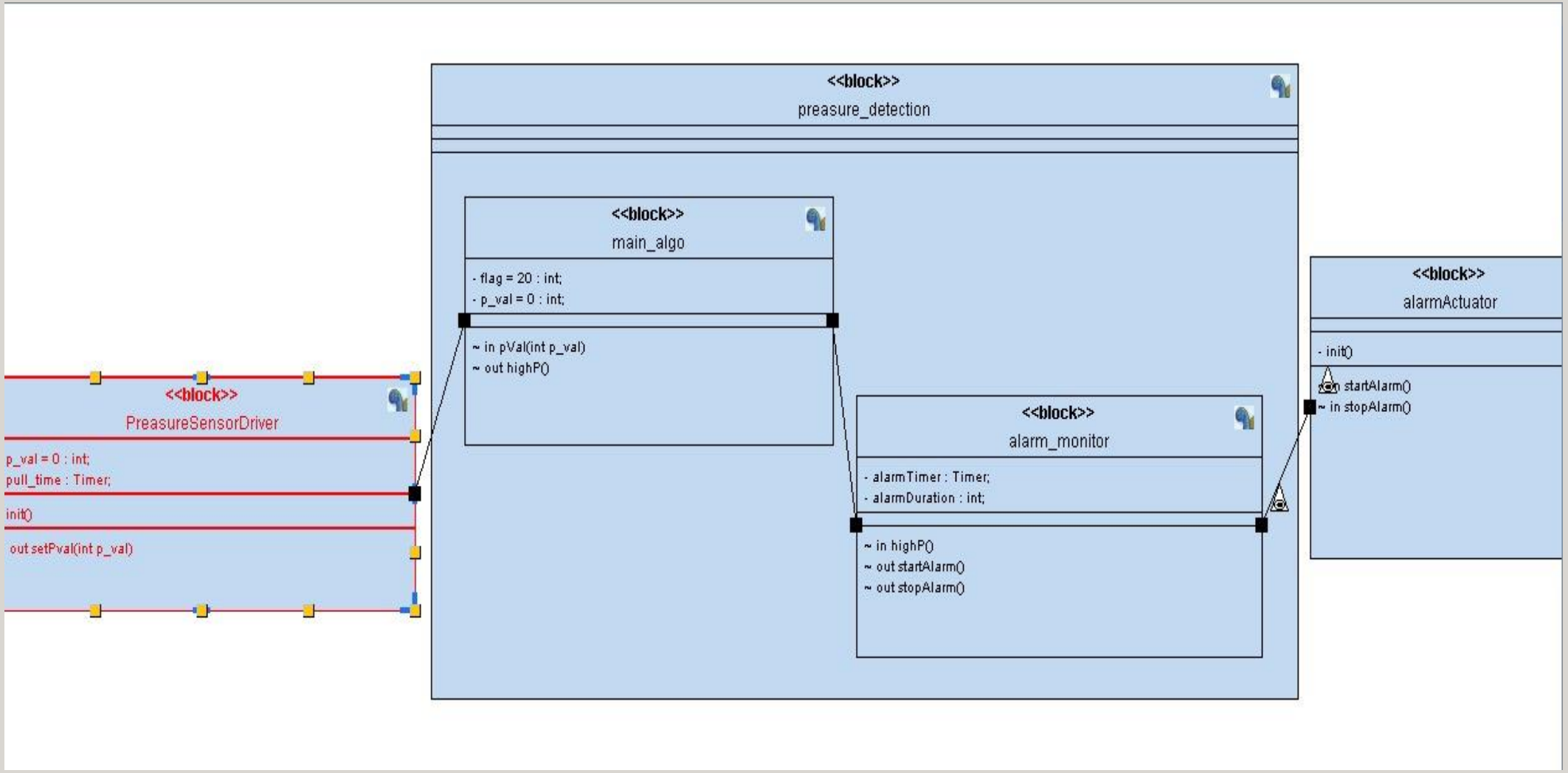
SYSTEM ANALYSIS: ACTIVITY DIAGRAM:



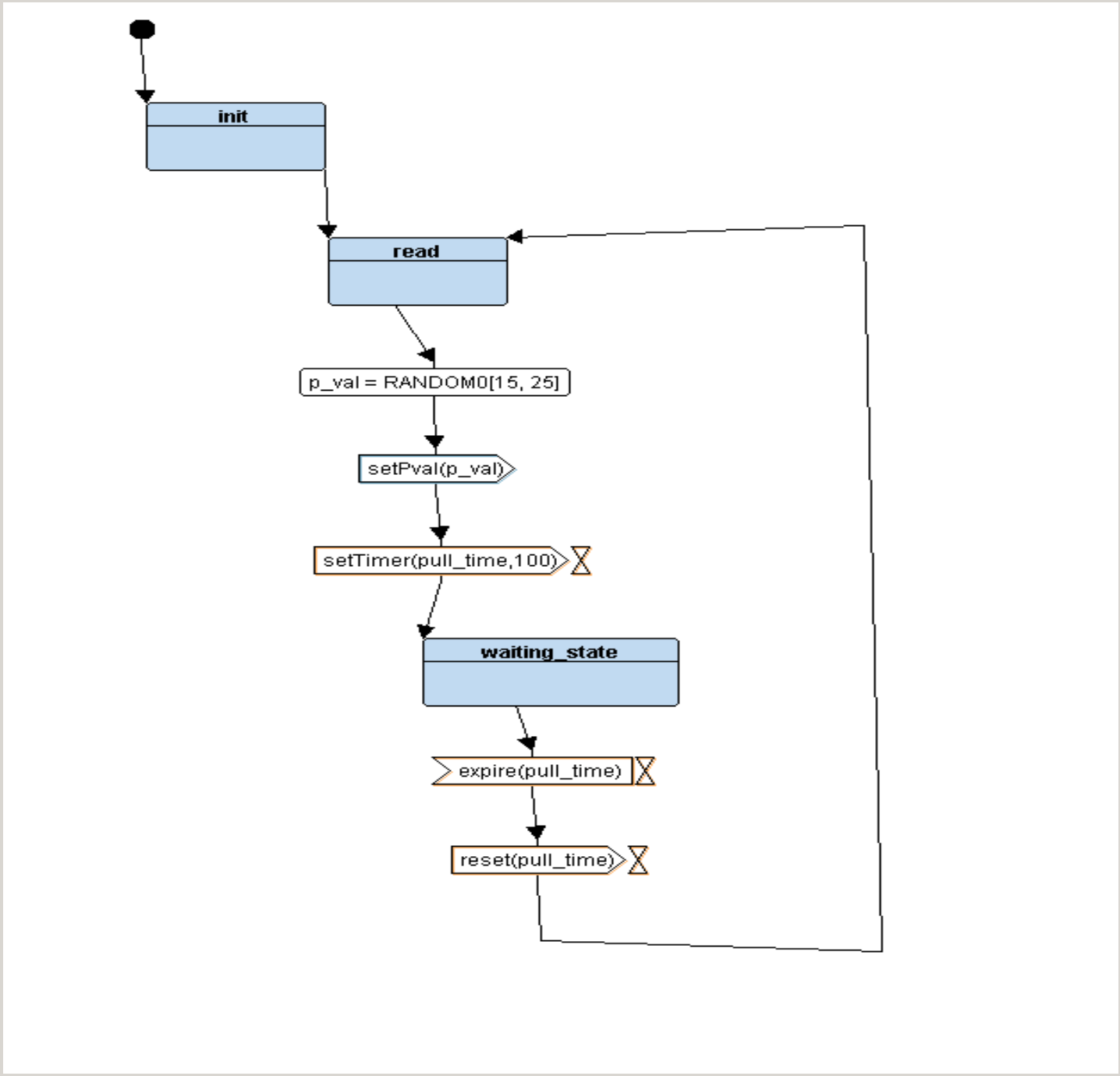
SYSTEM ANALYSIS: SEQUENCE DIAGRAM:



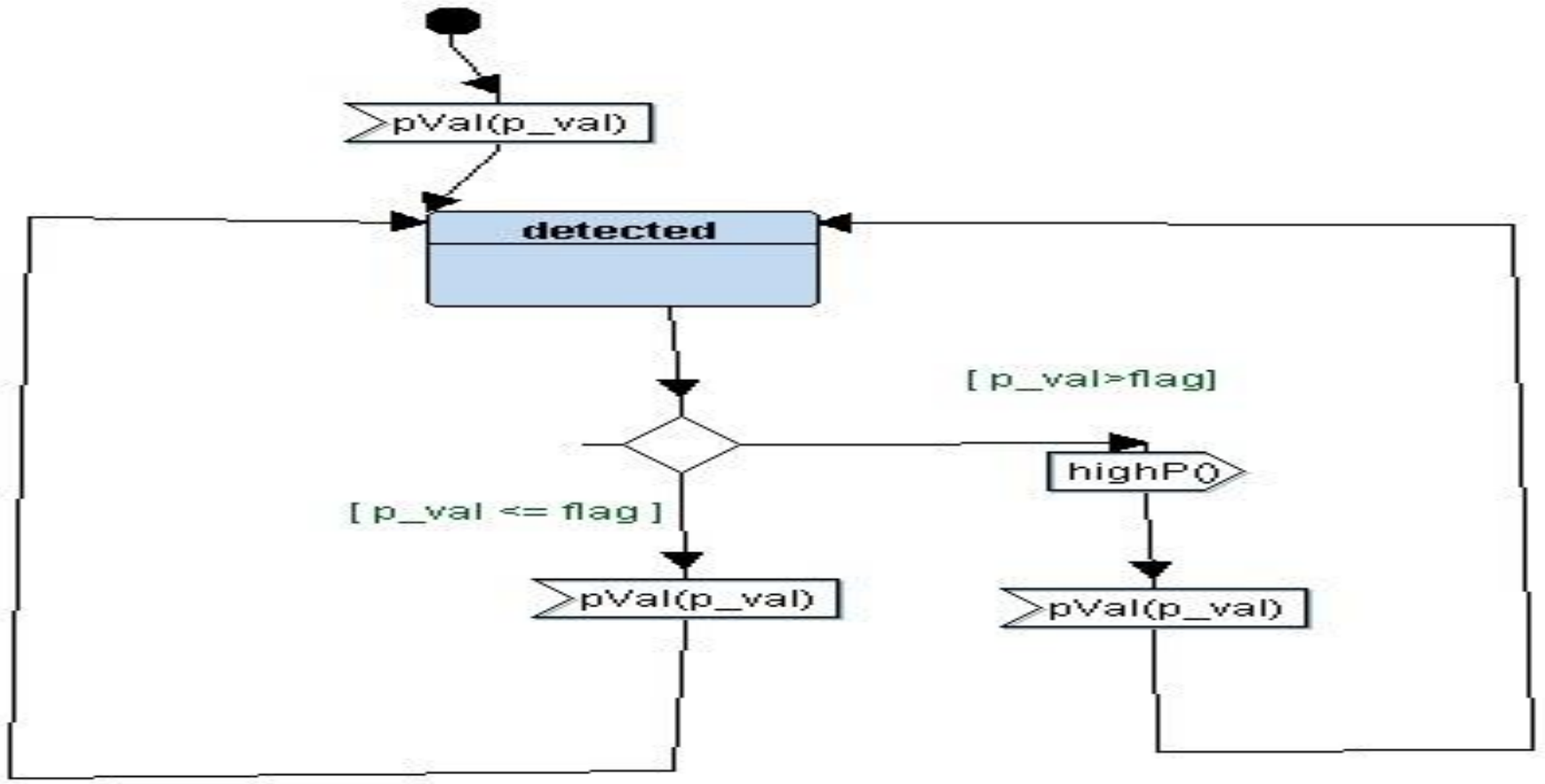
SYSTEM DESIGN:



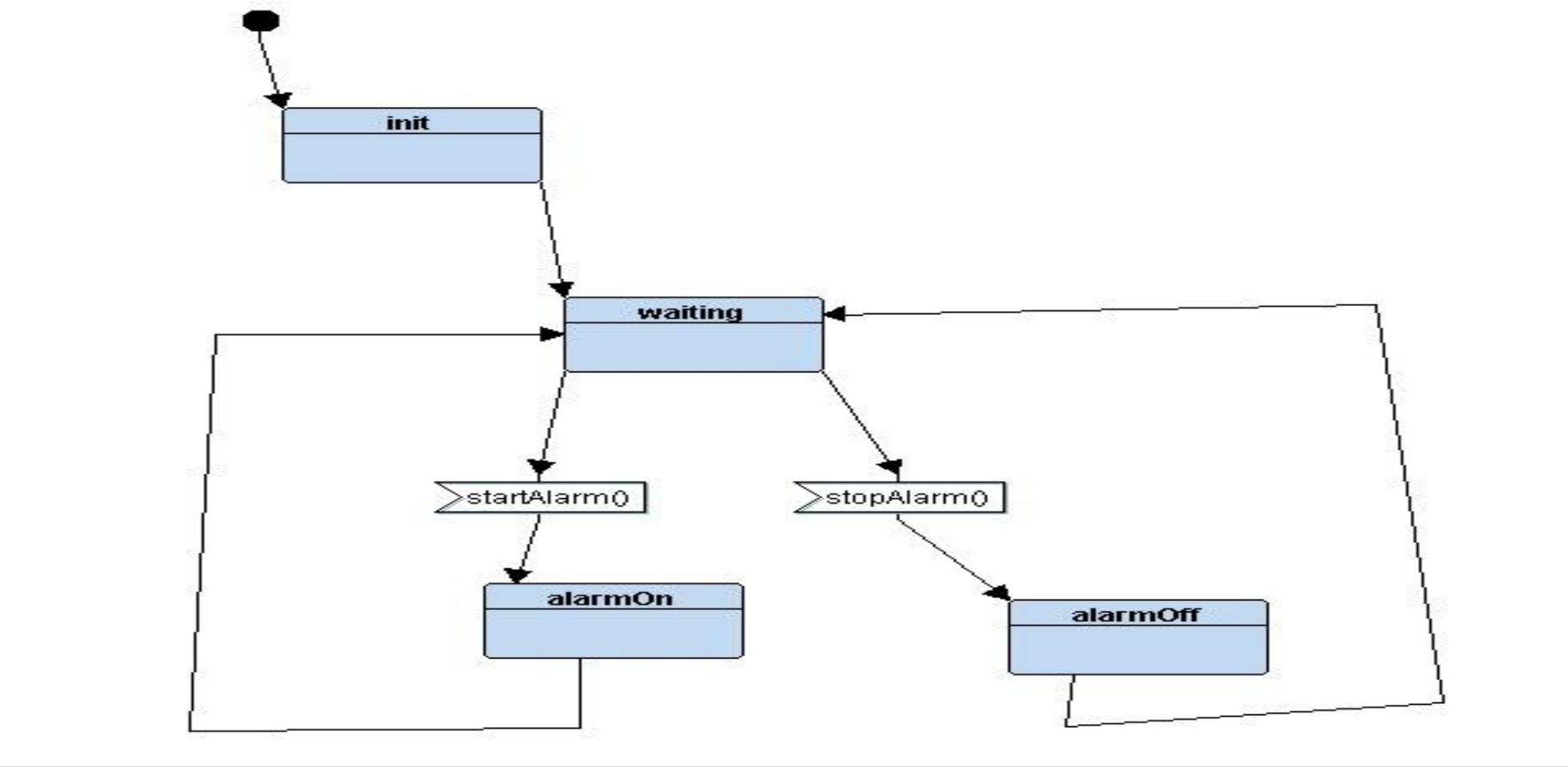
I-PRESSURE SENSOR STATE DIAGRAM



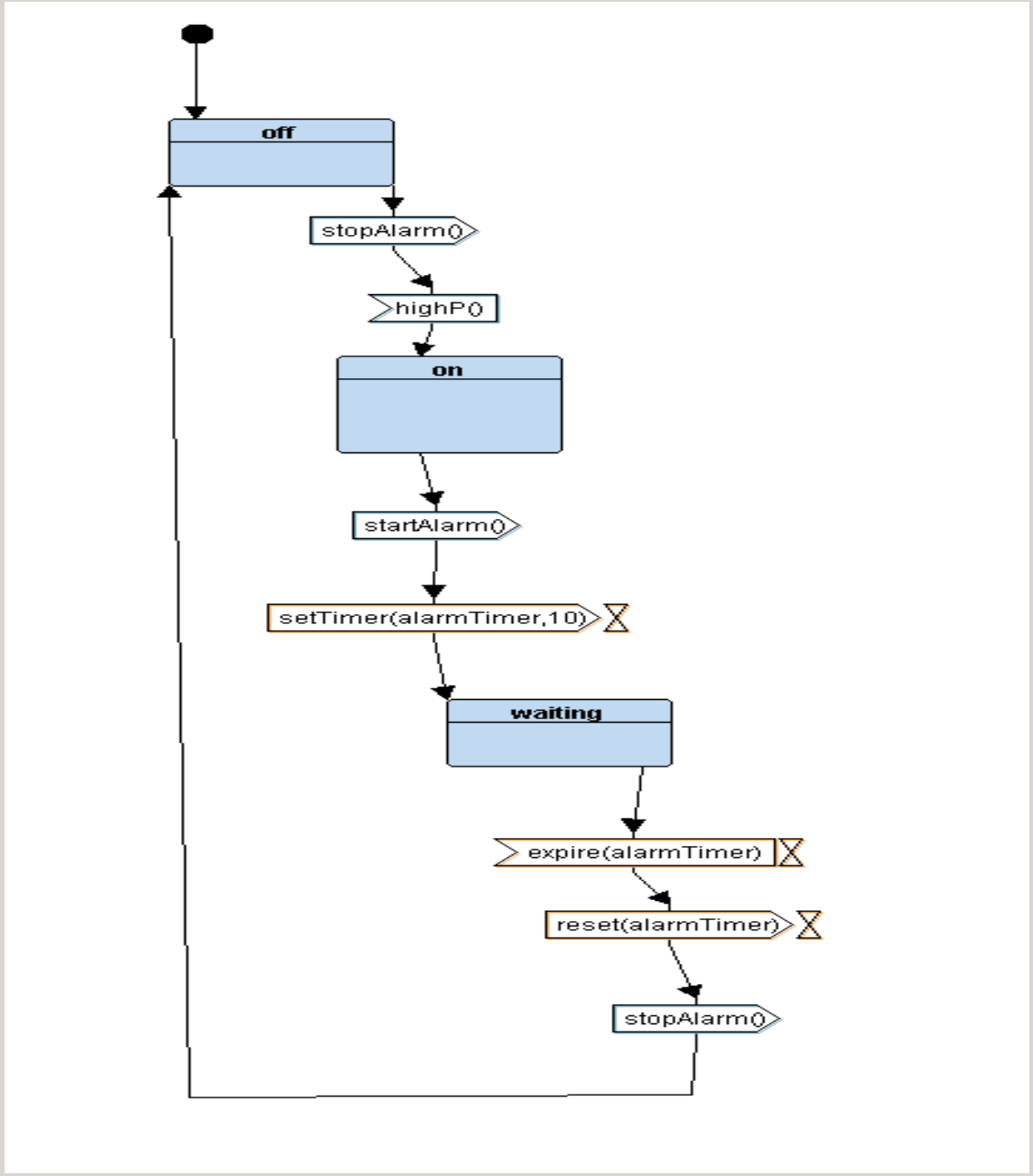
2-MAIN PROGRAM STATE DIAGRAM:



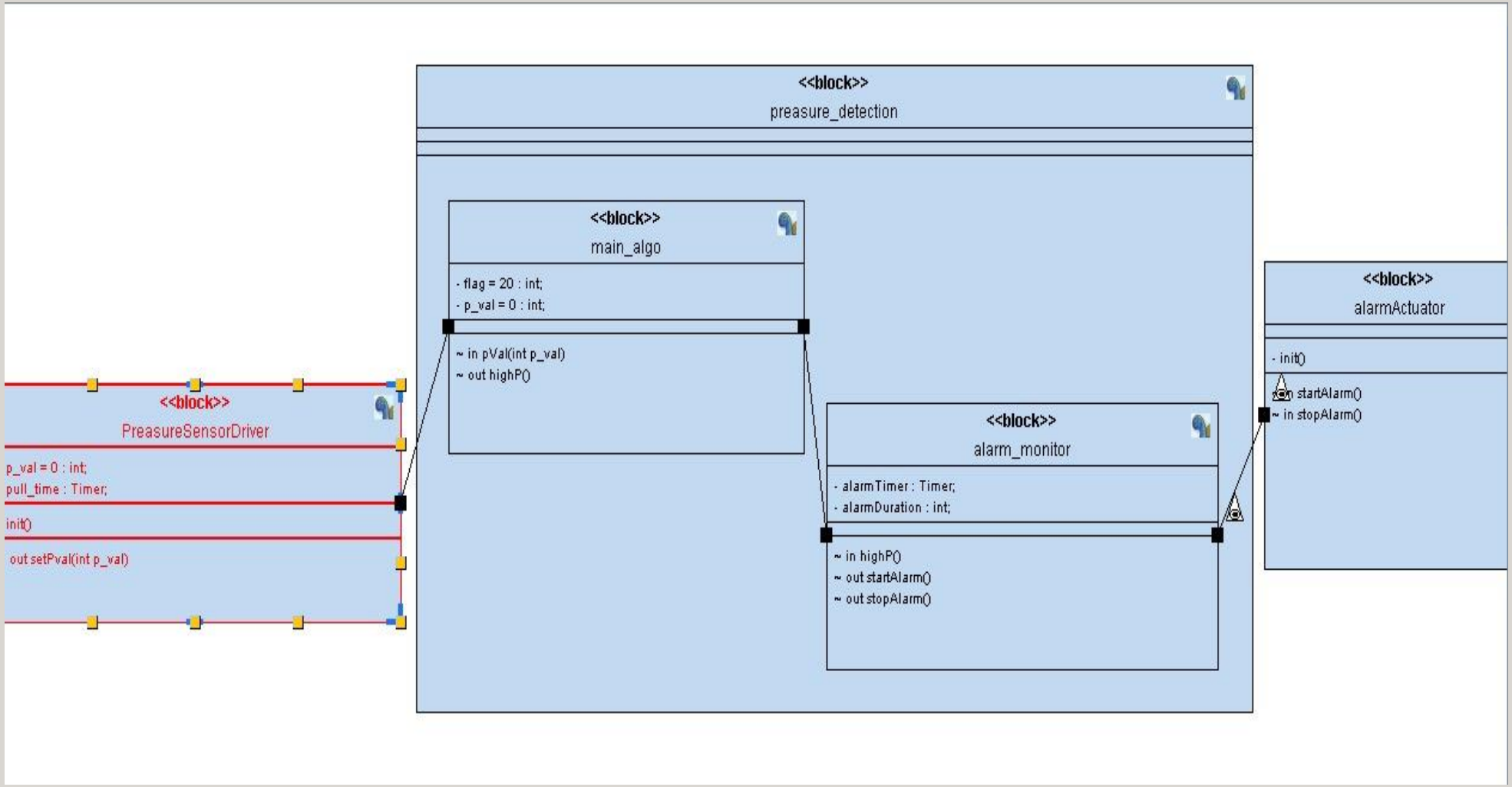
3-ALARM ACTUATOR STATE DIAGRAM:



4-ALARM MONITOR STATE DIAGRAM:



MAIN.C


<<block>>
preasure_detection
<<block>>
main_algo
- flag = 20 : int;
- p_val = 0 : int;
~ in pVal(int p_val)
~ out highP()
<<block>>
alarm_monitor
- alarmTimer : Timer;
- alarmDuration : int;
~ in highP()
~ out startAlarm()
~ out stopAlarm()
<<block>>
alarmActuator
- init()
~ in startAlarm()
~ in stopAlarm()
<<block>>
PreasureSensorDriver
p_val = 0 : int;
pull_time : Timer;
init()
out setPval(int p_val)

SYSTEM CODE

```c
int main()
{

    uint32 (*Pressure_state)() = &pressure_sensor;
    void (*Alarm_Actuator_state)() = &ActuatorState;
    void (*Alarm_monitor_state)() = &MonitorState;
    void (*Algo_State)() = &AlgoState;
    GPIO_INITIALIZATION();
    while (1)
    {
        Pressure_state();
        Alarm_Actuator_state();
        Alarm_monitor_state();
        Algo_State();
    }
}
```

```
P_state Pstate = Pinit;
uint32 Pvalue;
uint32 pressure_sensor()
{
    switch (Pstate)
    {
    case Pinit:
        Pstate = Pread;
        break;
    case Pread:
        Pvalue = getPressureVal();
        Pstate = Pwait;
        break;
    case Pwait:
        Delay(1000);
        Pstate = Pread;
        break;
    default:
        break;
    }
    return Pvalue;
}
```

```
void AlgoState()
{
    switch (state)
    {
    case Algo_HIGH_PRESSURE:
        if (Pvalue > Algo_pressure_threshold)
        {
            state = Algo_HIGH_PRESSURE;
        }
        else
            state = Algo_NORMAL_PRESSURE;
        break;
    default:
        break;
    }
}
```

```
void Delay(int nCount)
{
    for(; nCount != 0; nCount--);
}

int getPressureVal(){
    return (GPIOA_IDR & 0xFF);
}

void Set_Alarm_actuator(int i){
    if (i == 1){
        SET_BIT(GPIOA_ODR,13);
    }
    else if (i == 0){
        RESET_BIT(GPIOA_ODR,13);
    }
}

void GPIO_INITIALIZATION (){
    SET_BIT(APB2ENR, 2);
    GPIOA_CRL &= 0xFF0FFFFF;
    GPIOA_CRL |= 0x00000000;
    GPIOA_CRH &= 0xFF0FFFFF;
    GPIOA_CRH |= 0x22222222;
}
```

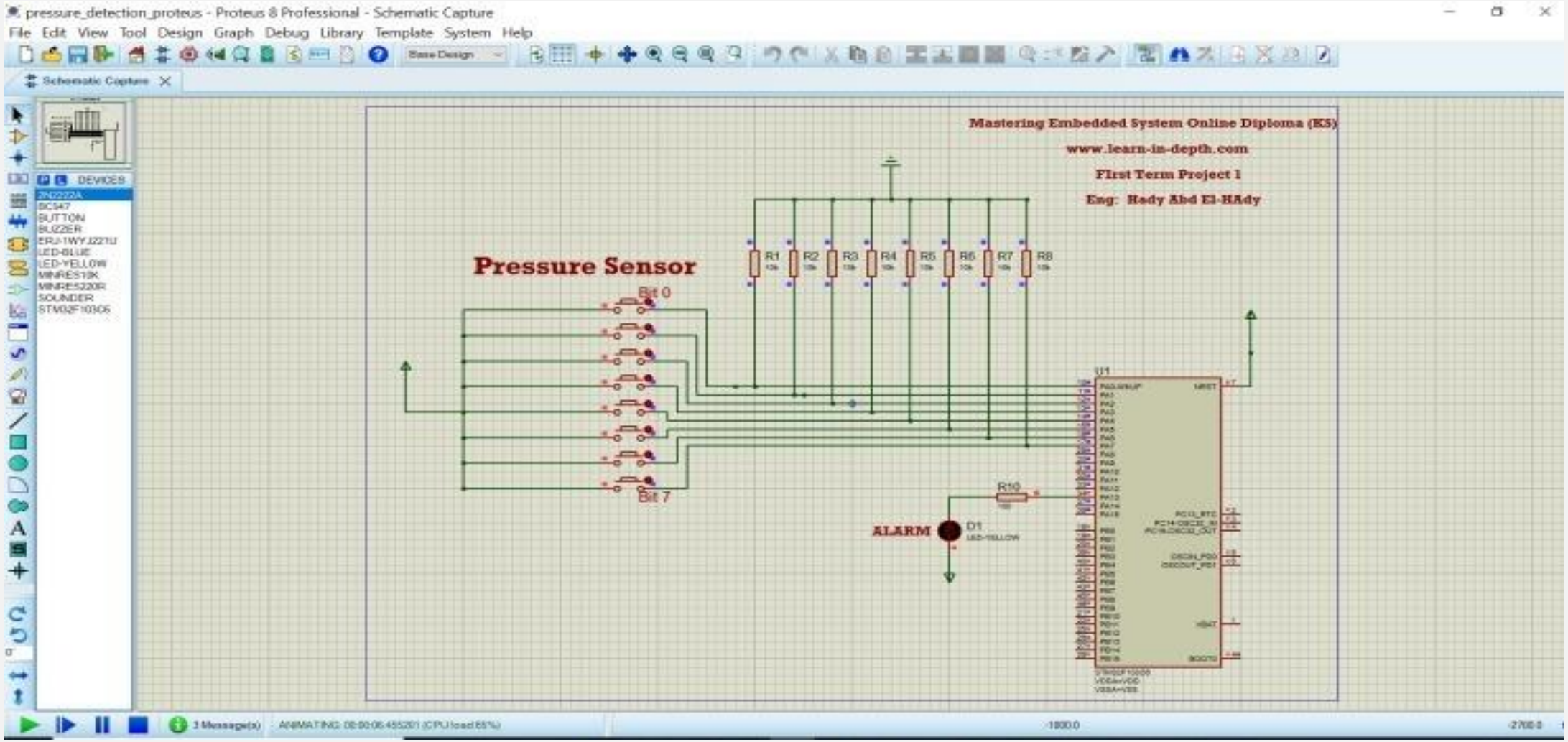
```
Mon_State Monitor_State=MonitorWait;
void MonitorState()
{

    switch (Monitor_State)
    {
    case MonitorOFF:
        ActState = AlarmOFF;
        if (Pvalue > Algo_pressure_threshold)
            Monitor_State = MonitorON;
        else
            Monitor_State = MonitorOFF;
        break;
    case MonitorON:
        ActState = AlarmON;
        Monitor_State = MonitorWait;
        break;
    case MonitorWait:
        Monitor_State = MonitorOFF;
        Delay(500);
        break;
    default:
        break;
    }
}
```

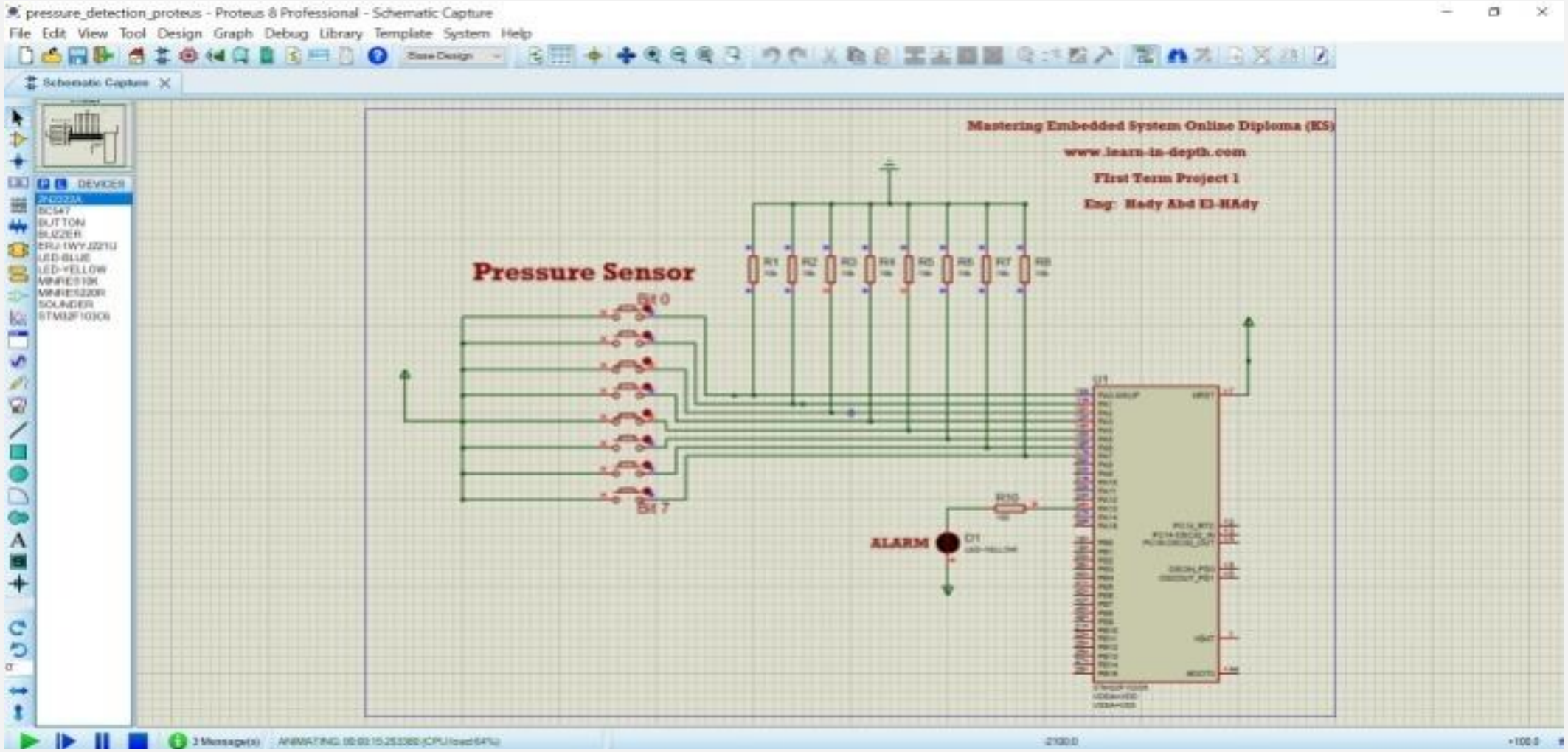
```
Act_State ActState=ActuatorINIT;
void ActuatorState()
{

    switch (ActState)
    {
    case ActuatorINIT:
    case ActuatorWAIT:
        ActState = ActuatorWAIT;
        break;
    case AlarmON:
        Set_Alarm_actuator(0);
        ActState = ActuatorWAIT;
        break;
    case AlarmOFF:
        Set_Alarm_actuator(1);
        ActState = ActuatorWAIT;
        break;
    default:
        break;
    }
}
```

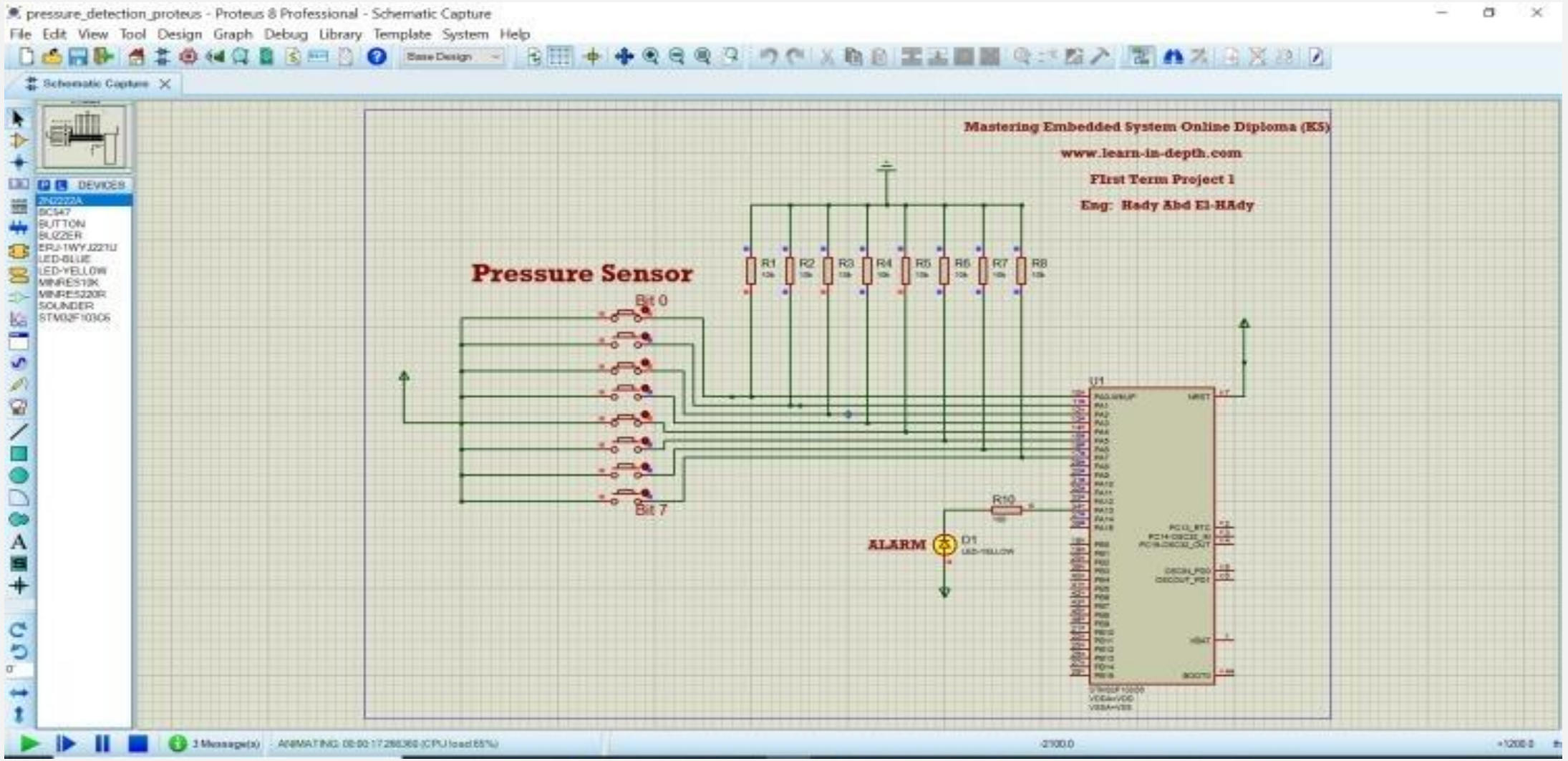
SCHEMATIC CIRCUIT


pressure_detection_proteus - Proteus 8 Professional - Schematic Capture
File Edit View Tool Design Graph Debug Library Template System Help
Mastering Embedded System Online Diploma (K5)
www.learn-in-depth.com
First Term Project 1
Eng: Hady Abd El-HAdy
Pressure Sensor
Bit 0
Bit 7
R1
R2
R3
R4
R5
R6
R7
R8
R10
ALARM
D1
U1

FOR 0 INPUT



FOR 20 AS INPUT



FOR 21 AS INPUT