MASTERING EMBEDDED SYSTEMS ONLINE DIPLOMA

FIRST TERM (FINAL PROJECT 1)

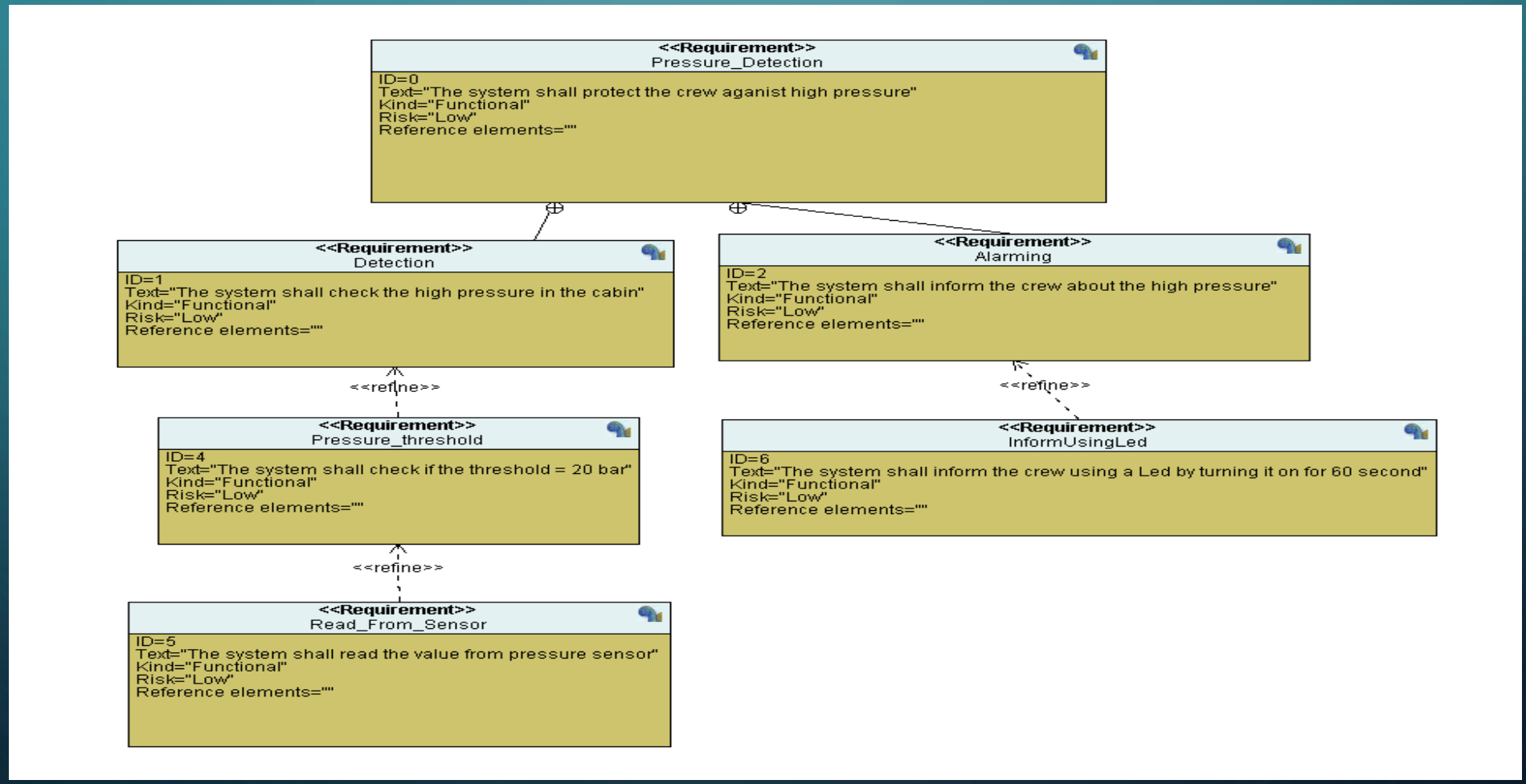
ENG. AMR ALI AHMED

HTTPS://WWW.LEARN-IN-DEPTH.COM/ONLINE-DIPLOMA/MEROMATRIX25%40GMAIL.COM

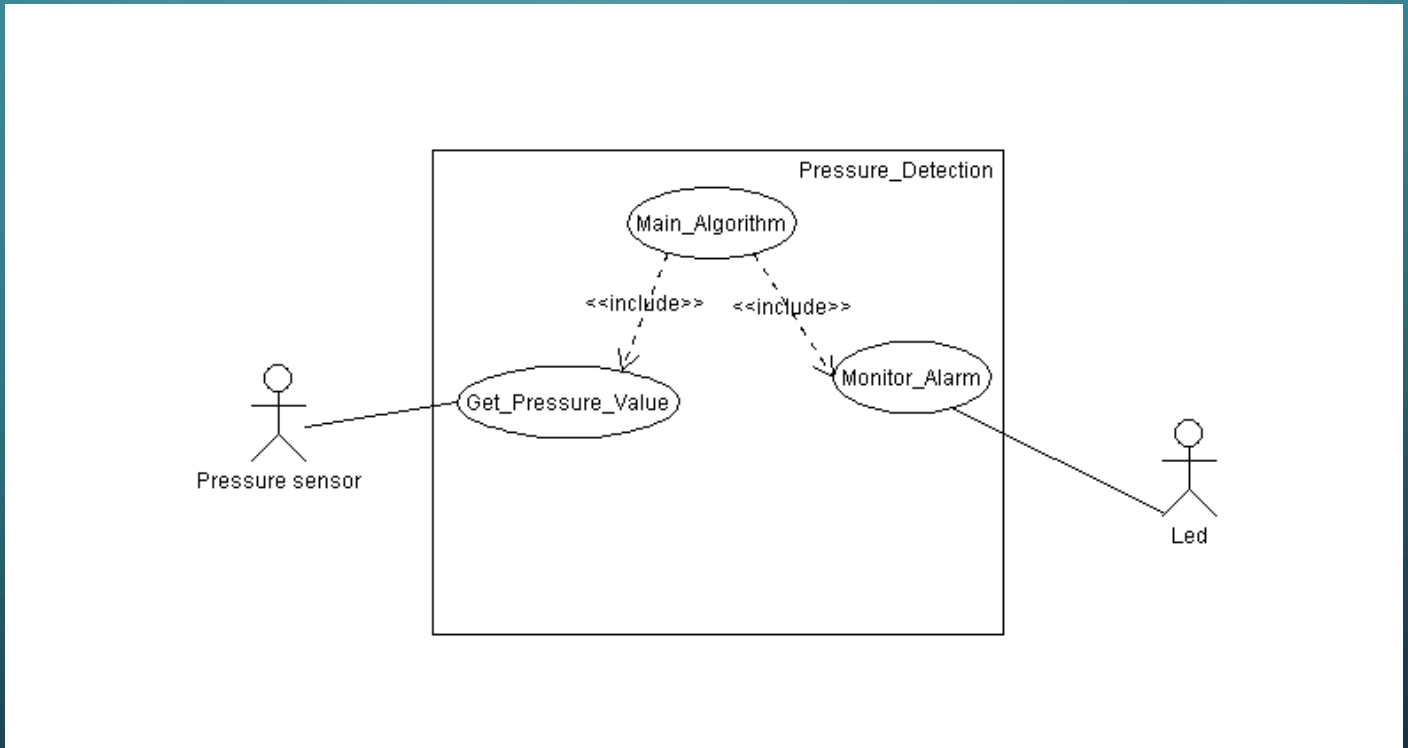
THE PROBLEM

- A ”**client**” expects you to deliver the software of Pressure Detection system using the following system specification:
  - 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

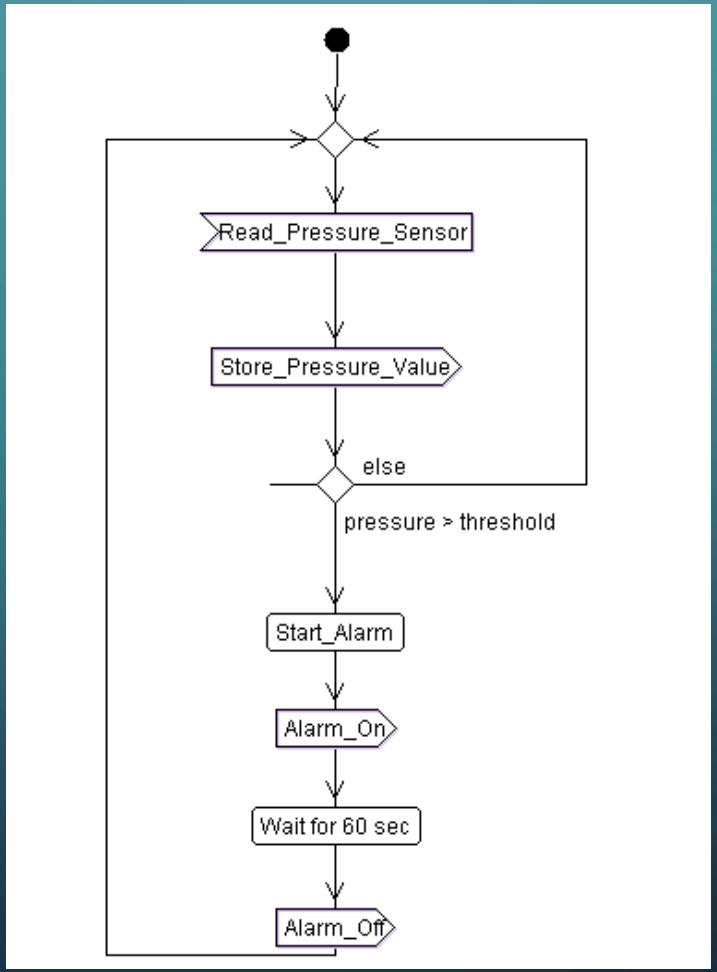
REQUIREMENT DIAGRAM



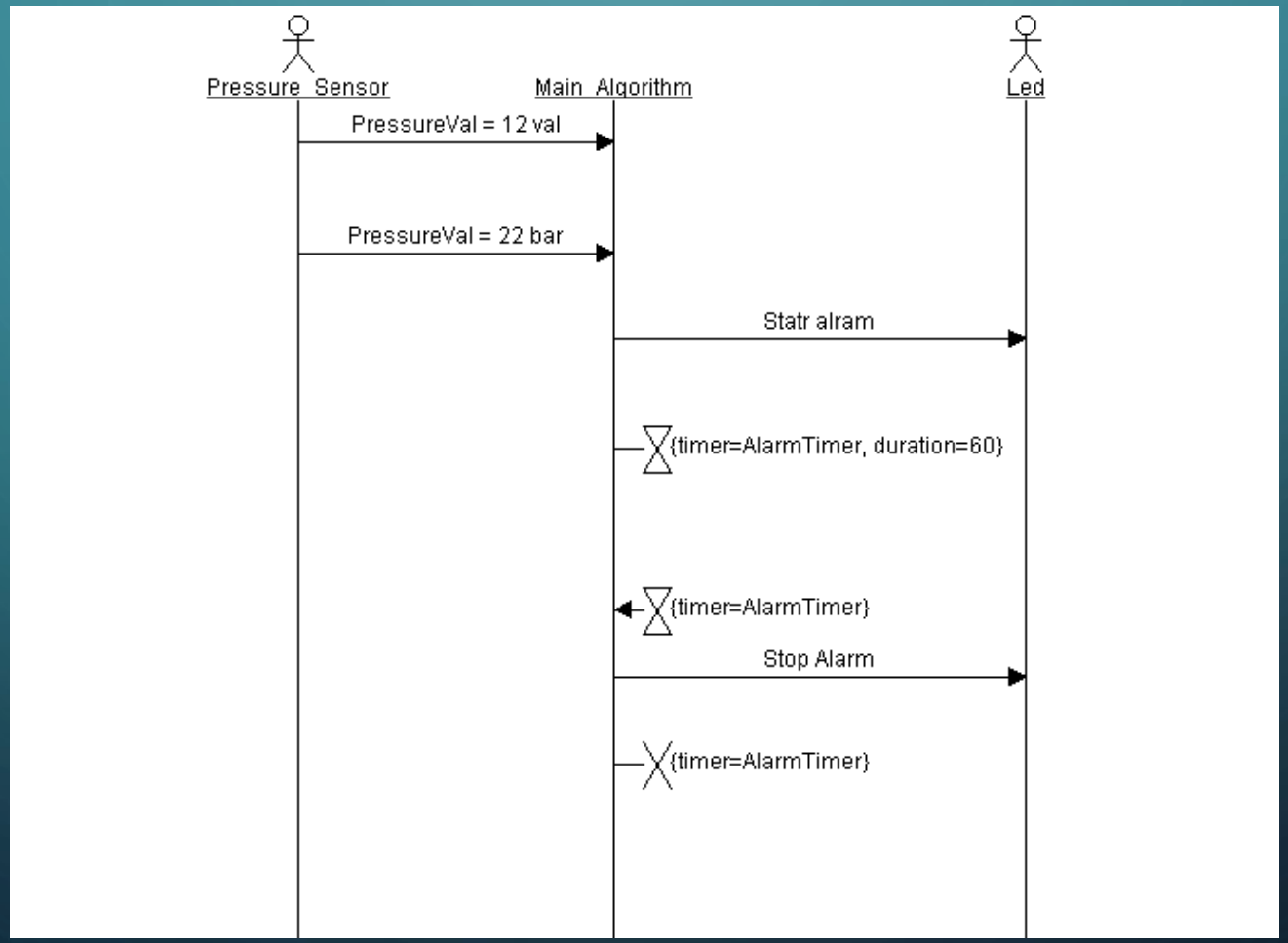
USE CASE DIAGRAM



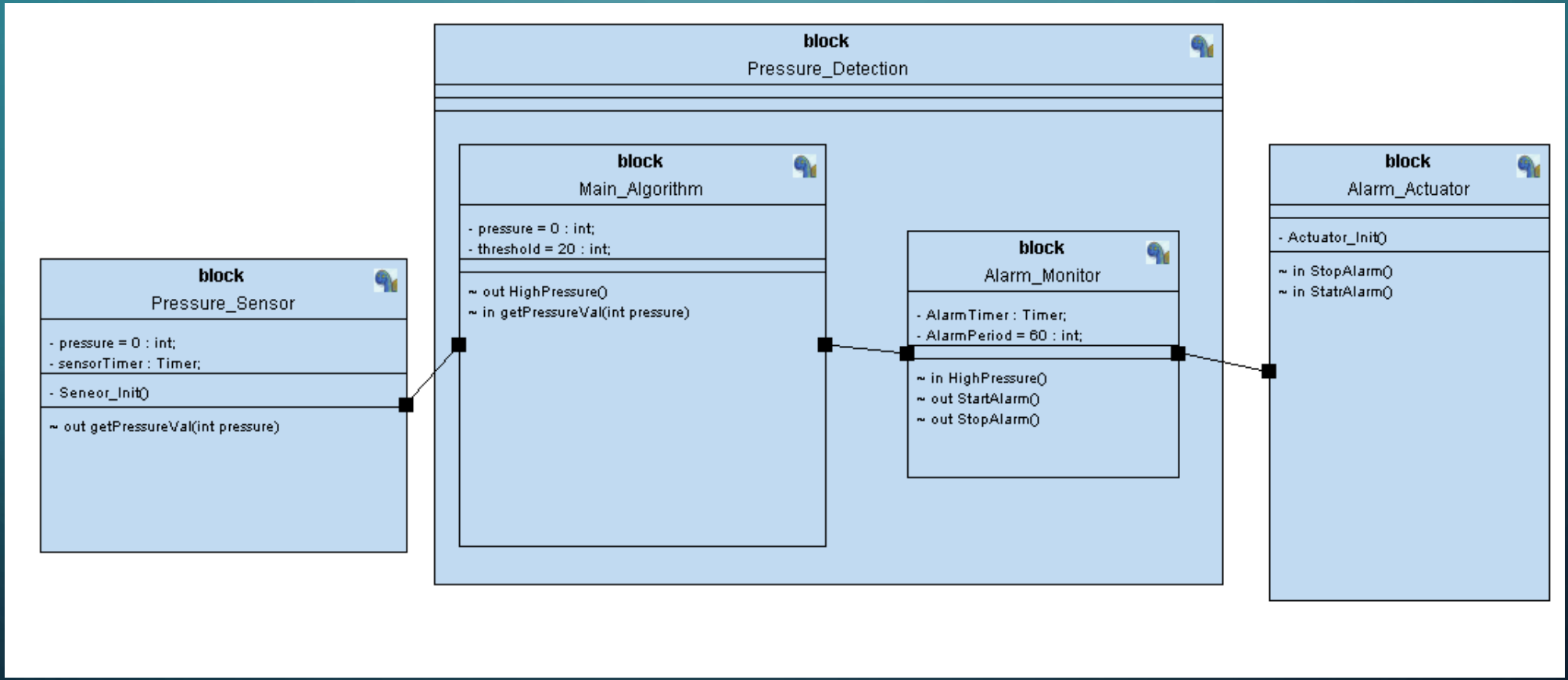
ACTIVITY DIAGRAM



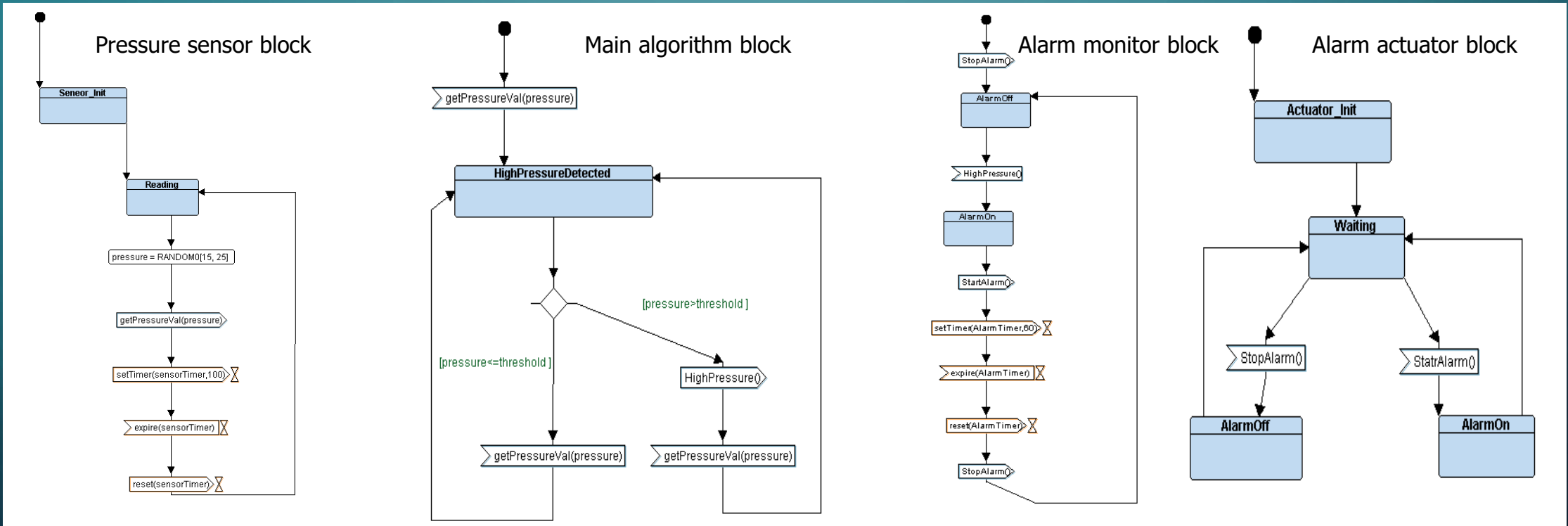
SEQUENCE DIAGRAM



SYSTEM DESIGN



STATE MACHINE FOR BLOCKS



IMPLEMENTATION

```c
/*
 * main.c
 *
 *  Created on: Jul 25, 2023
 *      Author: AmrAli
 */

#include "driver.h"
#include "alarm_monitor.h"
#include "alarm_actuator.h"
#include "algo.h"
#include "PrSensor.h"
#include "state.h"

void setup()
{
    //init all drivers
    GPIO_INITIALIZATION();
    //init block
    PrSensor_init();
    alg_state = STATE(HighPreDetected);
    AM_state = STATE(AlarmOff);
    Alarm_init();
}
void main()
{
    setup();
    while(1)
    {
        PrS_state();
        alg_state();
        AM_state();
        Led_state();
    }
}
```

```c
/*
 * state.h
 *
 *  Created on: Jul 25, 2023
 *      Author: AmrAli
 */

#ifndef STATE_H_
#define STATE_H_

#include "stdio.h"
#include "stdlib.h"
#include "platform_types.h"

// Automatic state function generator
#define STATE_define(_stFun_) void ST_##_stFun_()
#define STATE(_stFun_) ST_##_stFun_

// state connections
void setPressureVal(uint8 pressure);
void HighPressure();
void StartAlarm();
void StopAlarm();

#endif /* STATE_H_ */
```

IMPLEMENTATION

```c
/*
 * PrSensor.c
 *
 *  Created on: Jul 25, 2023
 *      Author: AmrAli
 */

#include "PrSensor.h"
#include "driver.h"

// variables
uint8 pressure = 0;

// State pointer to function
void (*PrS_state)();

// Flow of the program

void PrSensor_init()
{
    // initialize of the pressure sensor will be called from driver.h
    PrS_state = STATE(PrS_Reading);
}

STATE_define(PrS_Reading)
{
    PrS_state_id = PrS_Reading;
    pressure = getPressureVal();
    setPressureVal(pressure);
    PrS_state = STATE(PrS_Reading);
}
```

```c
/*
 * PrSensor.h
 *
 *  Created on: Jul 25, 2023
 *      Author: AmrAli
 */

#ifndef PRSENSOR_H_
#define PRSENSOR_H_

// libs includes

#include "state.h"

// define the states
enum{
    PrS_Reading
}PrS_state_id;

// declare state functions for pressure sensor
STATE_define(PrS_Reading);

// State pointer to function
extern void (*PrS_state)();

// APIs
void PrSensor_init();


#endif /* PRSENSOR_H_ */
```

IMPLEMENTATION

```c
/*
 * main_algorithem.c
 *
 *  Created on: Jul 25, 2023
 *      Author: AmrAli
 */

#include "algo.h"

//variabls
uint8 pVal = 0;
uint8 threshold = 20;

// state pointer to function
void (*alg_state)();

// connection abstraction
void setPressureVal(uint8 pressure)
{
    pVal = pressure;
    alg_state = STATE(HighPreDetected);
}

STATE_define(HighPreDetected)
{
    alg_state_id = HighPreDetected;
    if(pVal > threshold)
    {
        HighPressure();
        alg_state = STATE(HighPreDetected);
    }
    else
        alg_state = STATE(HighPreDetected);
}

```

```c
/*
 * main_algorithem.h
 *
 *  Created on: Jul 25, 2023
 *      Author: AmrAli
 */

#ifndef ALGO_H_
#define ALGO_H_

// includes
#include "state.h"


enum{
    HighPreDetected
}alg_state_id;

// state Pointer to function
extern void (*alg_state)();

//declare state function for main algorithem
STATE_define(HighPreDetected);

#endif /* ALGO_H_ */

```

IMPLEMENTATION

```c
/*
 * alarm_monitor.c
 *
 *  Created on: Jul 26, 2023
 *      Author: AmrAli
 */


#include "alarm_monitor.h"

void (*AM_state)();

void HighPressure()
{
    AM_state = STATE(AlarmOn);
}

STATE_define(AlarmOff)
{
    AM_state_id = AlarmOff;
    StopAlarm();
}

STATE_define(AlarmOn)
{
    AM_state_id = AlarmOn;
    StartAlarm();
    AM_state = STATE(AlarmOff);
}


```

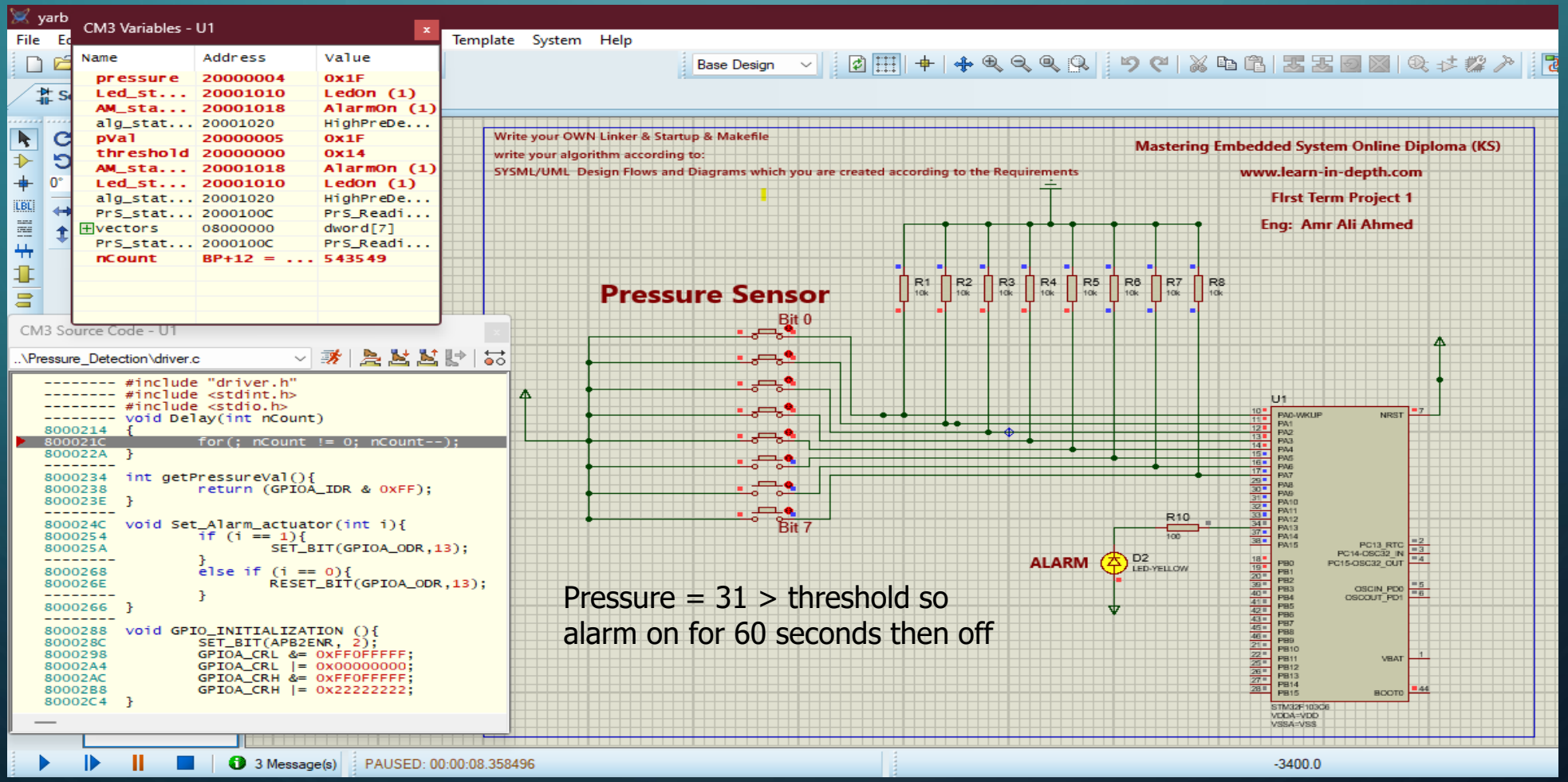
```c
/*
 * alarm_monitor.h
 *
 *  Created on: Jul 26, 2023
 *      Author: AmrAli
 */

#ifndef ALARM_MONITOR_H_
#define ALARM_MONITOR_H_

#include "state.h"

enum{
    AlarmOff,
    AlarmOn
}AM_state_id;

// state pointer to function
extern void (*AM_state)();

// state function for alarm monitor
STATE_define(AlarmOn);
STATE_define(AlarmOff);

#endif /* ALARM_MONITOR_H_ */

```

IMPLEMENTATION

```c
 * alarm_actuator.c
 *
 *  Created on: Jul 26, 2023
 *      Author: AmrAli
 */
#include "alarm_actuator.h"
#include "driver.h"

void (*Led_state)();
void Alarm_init()
{
    // init the alarm
    Led_state = STATE(Waiting);
}
void StartAlarm()
{
    Led_state = STATE(LedOn);
}
void StopAlarm()
{
    Led_state = STATE(LedOff);
}
STATE_define(LedOn)
{
    Led_state_id = LedOn;
    Set_Alarm_actuator(0);
    Delay(1500000);
    Set_Alarm_actuator(1);
    Delay(2000000);
    Led_state = STATE(Waiting);
}
STATE_define(LedOff)
{
    Led_state_id = LedOff;
    Set_Alarm_actuator(1);
    Led_state = STATE(Waiting);
}
STATE_define(Waiting)
{
    Led_state_id = Waiting;
    Set_Alarm_actuator(1);
    Delay(10000);
}
```

```c
/*
 * alarm_actuator.h
 *
 *  Created on: Jul 26, 2023
 *      Author: AmrAli
 */

#ifndef ALARM_ACTUATOR_H_
#define ALARM_ACTUATOR_H_

#include "state.h"

enum{
    LedOff,
    LedOn,
    Waiting
}Led_state_id;

// state pointer to function
extern void (*Led_state)();

// state function for alarm monitor
STATE_define(LedOn);
STATE_define(LedOff);
STATE_define(Waiting);

// APIs
void Alarm_init();

#endif /* ALARM_ACTUATOR_H_ */
```

SIMULATION


CM3 Variables - U1
Name Address Value
pressure 20000004 0x1F
Led_st... 20001010 LedOn (1)
AM_sta... 20001018 AlarmOn (1)
alg_stat... 20001020 HighPreDe...
pVal 20000005 0x1F
threshold 20000000 0x14
AM_sta... 20001018 AlarmOn (1)
Led_st... 20001010 LedOn (1)
alg_stat... 20001020 HighPreDe...
PrS_stat... 2000100C PrS_Readi...
vectors 08000000 dword[7]
PrS_stat... 2000100C PrS_Readi...
nCount BP+12 = ... 543549
CM3 Source Code - U1
..\Pressure_Detection\driver.c
#include "driver.h"
#include <stdint.h>
#include <stdio.h>
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;
Base Design
Write your OWN Linker & Startup & Makefile
write your algorithm according to:
SYSML/UML Design Flows and Diagrams which you are created according to the Requirements
Mastering Embedded System Online Diploma (KS)
www.learn-in-depth.com
FIrst Term Project 1
Eng: Amr Ali Ahmed
Pressure Sensor
Bit 0
Bit 7
R1 R2 R3 R4 R5 R6 R7 R8
R10
ALARM
D2
LED-YELLOW
U1
Pressure = 31 > threshold so
alarm on for 60 seconds then off
3 Message(s)
PAUSED: 00:00:08.358496
-3400.0