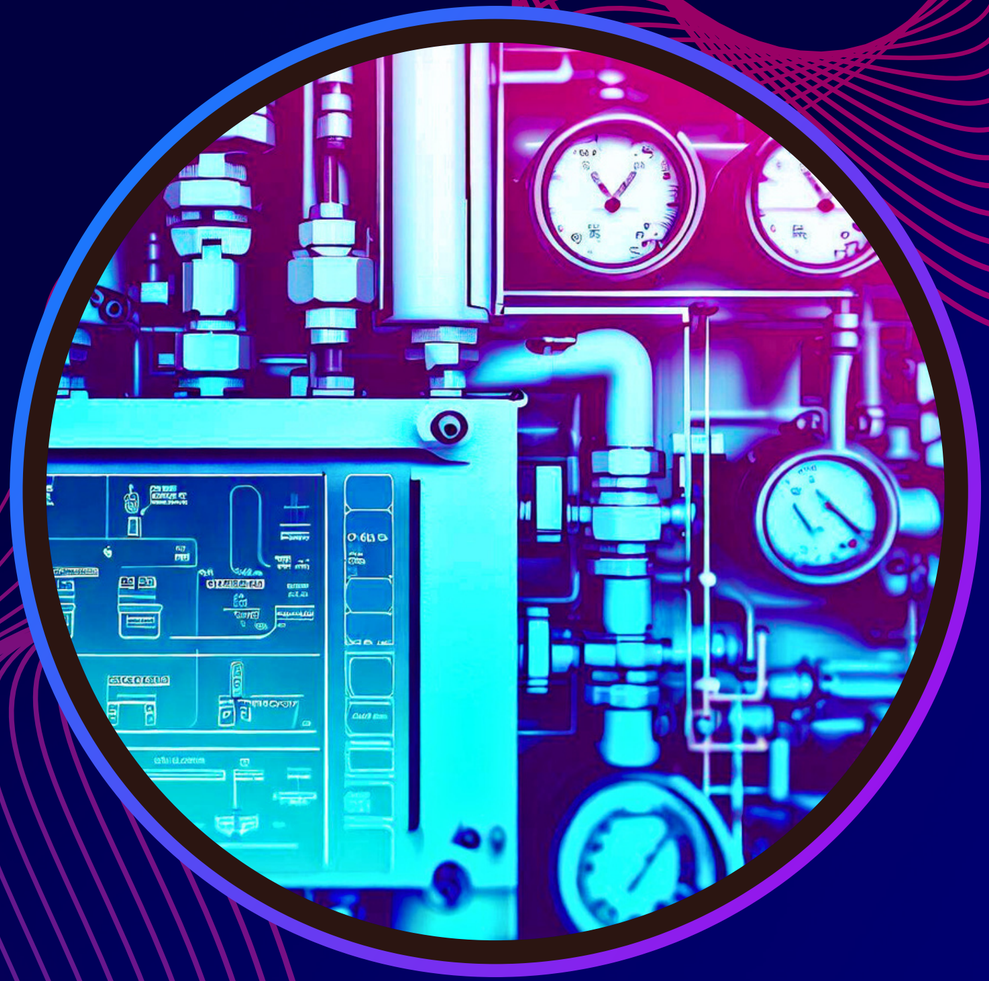
lean in depth

Mastring Embedded Systems Online Diploma

PRESSURE CONTROLLER



PRESENTED BY

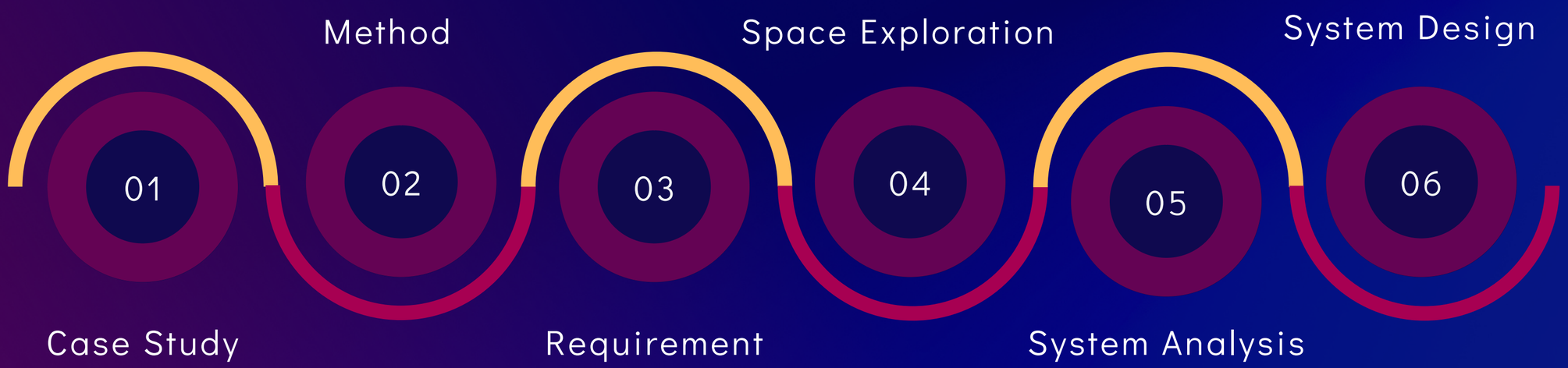
Mohamed Ebrahim Belal

PRESENTED TO

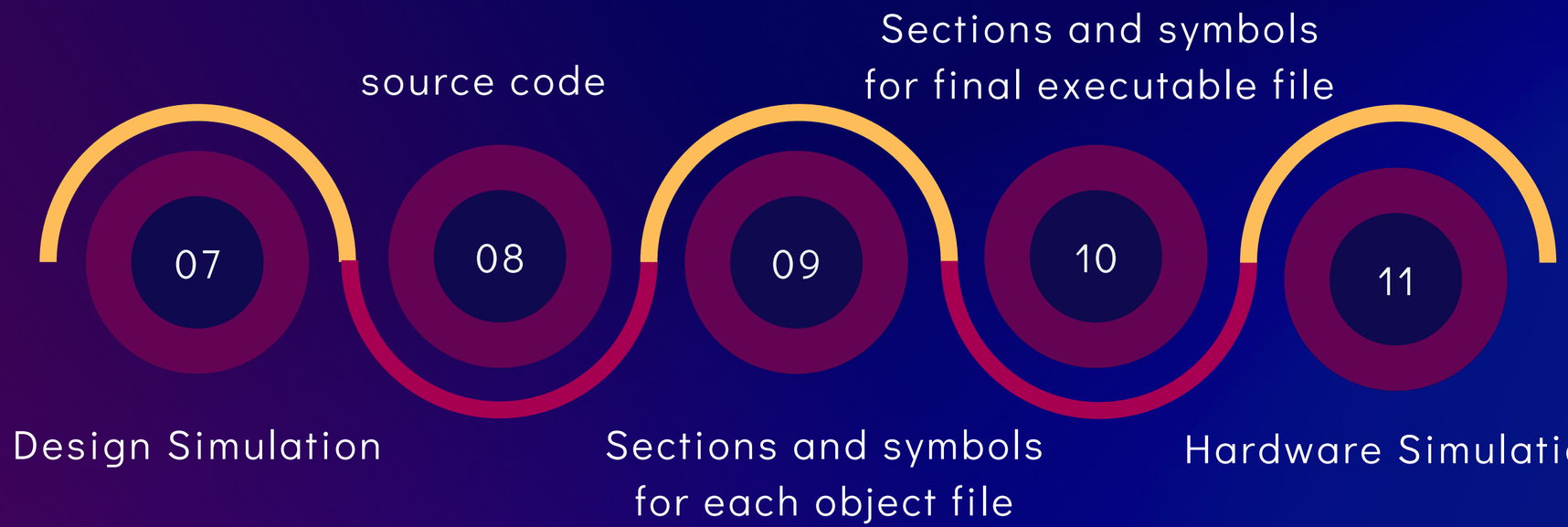
ENG : keroles Shenouda

22/7/2022

System Architecting & Design Sequence



System Architecting & Design Sequence



01

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:

1. The system setup and shutdown procedures are not modeled.
2. The system maintenance is not modeled.
3. The pressure sensor never fails.
4. The alarm never fails.
5. The system never faces power cut.

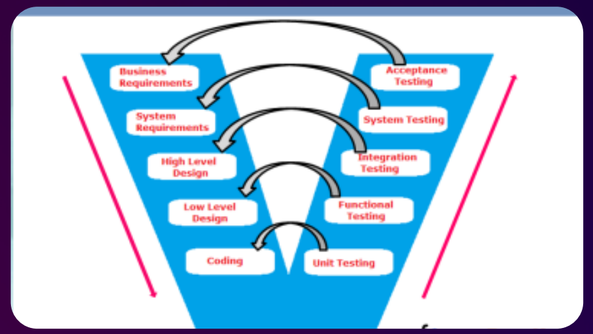
02

Methodology

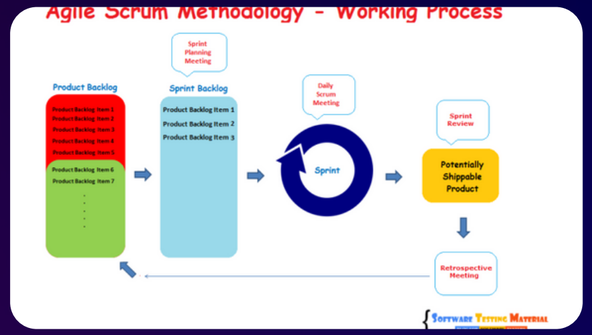
Since the system has multiple modules that are no easy to integrate

There are many methods, such as :

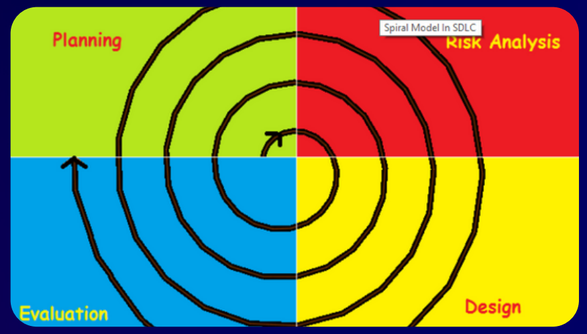
V-model



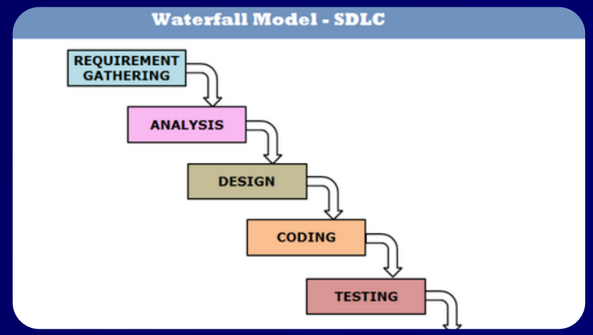
agile model



spiral model



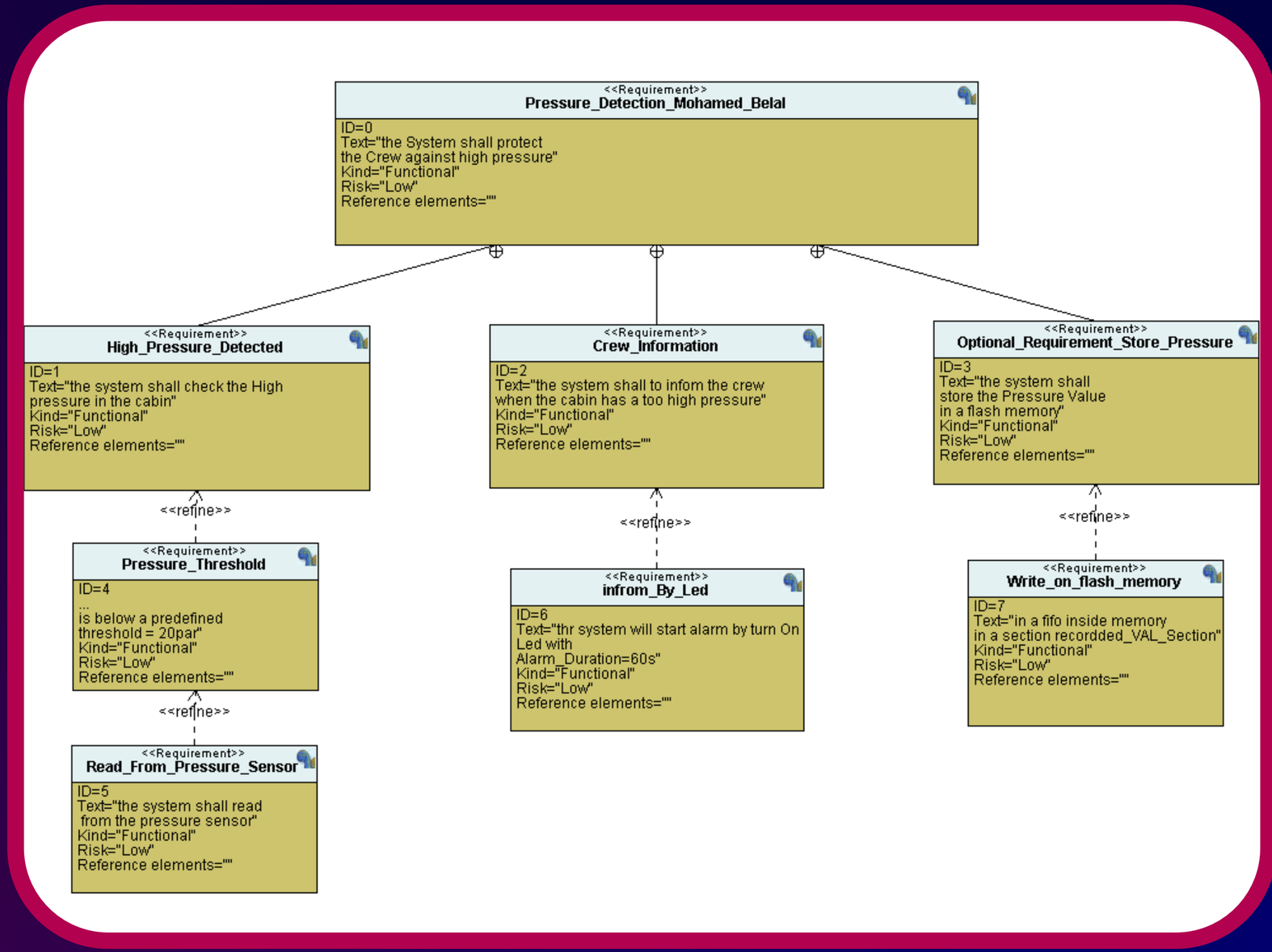
waterfall model



we Will Use V-Model

03

Requirement



04

Space Exploration

We Will Use STM32F103C6 microcontroller with a cortex-m3

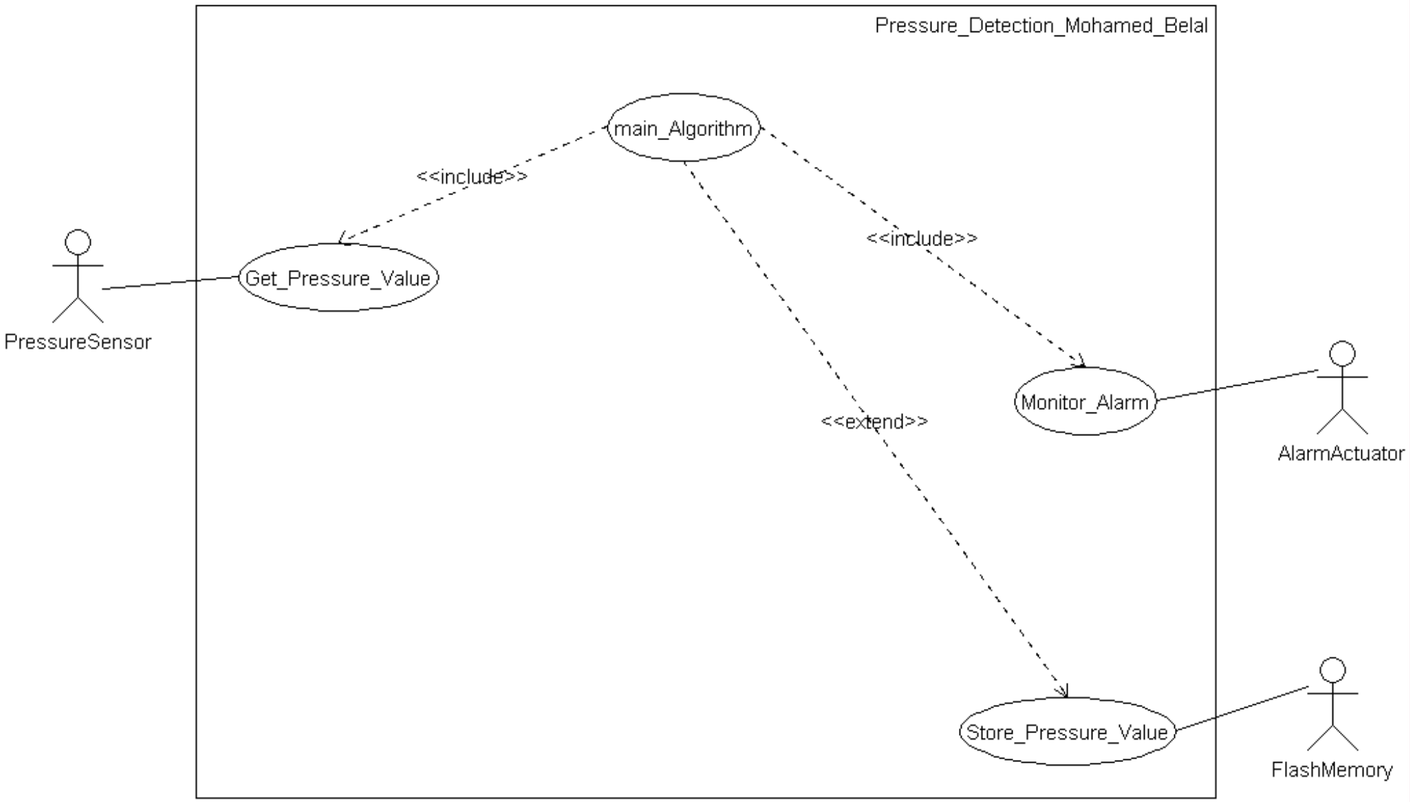
it's more than enough for this application

05

System Analysis

1

Use Case Diagram

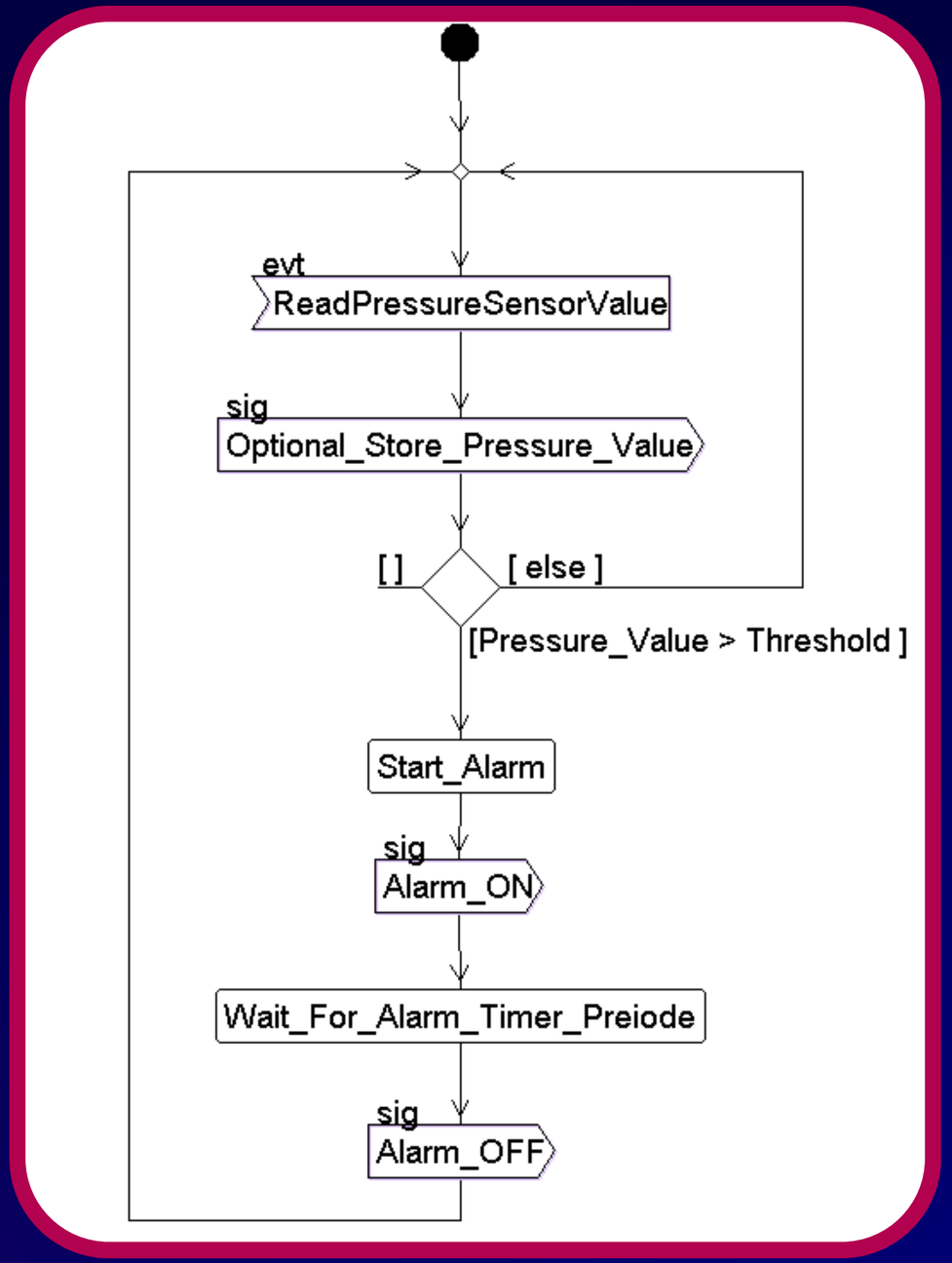


05

System Analysis

2

Activity Diagram

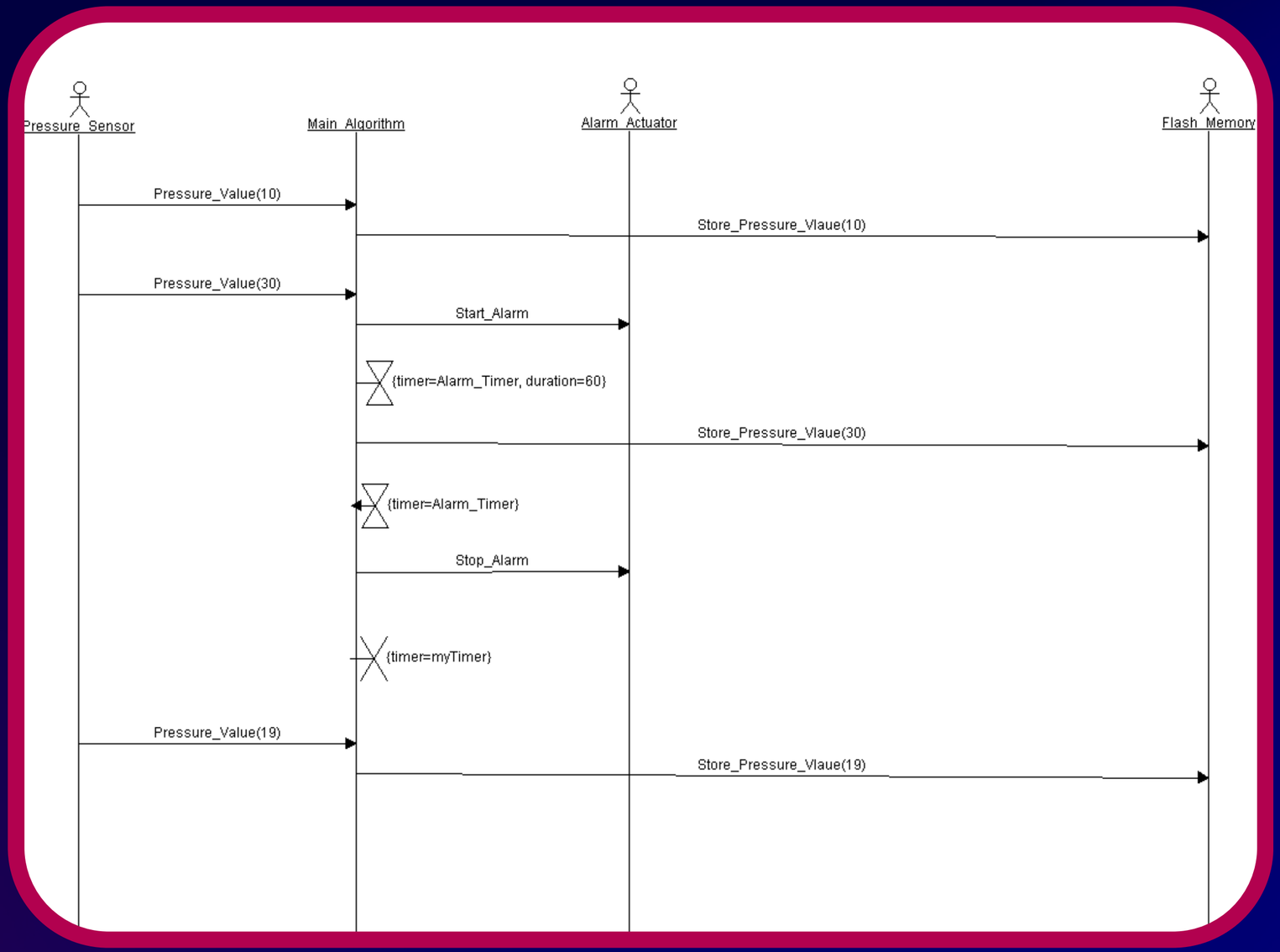


05

System Analysis

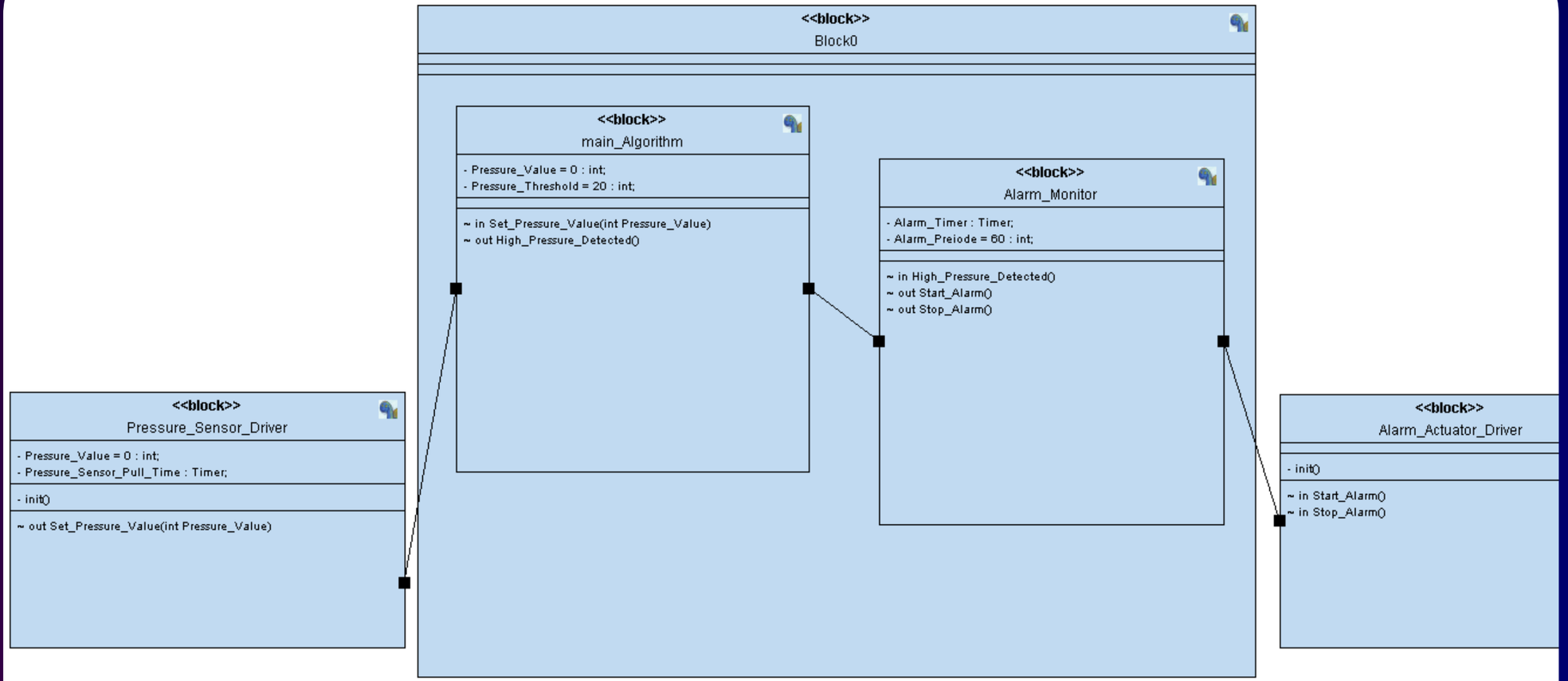
3

Sequence Diagram



06

System Design

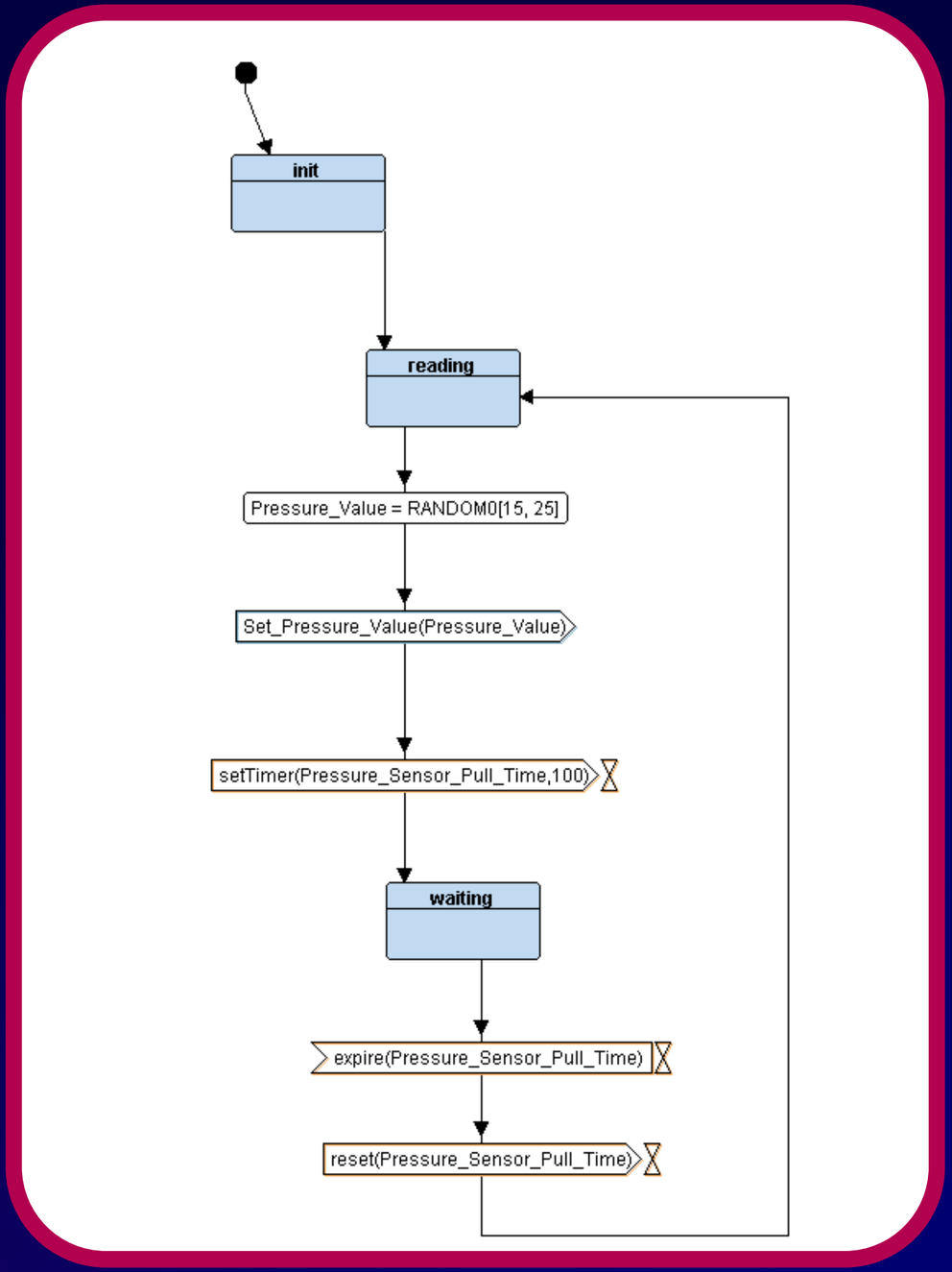


06

System Design

1

Pressure Sensor State Diagram

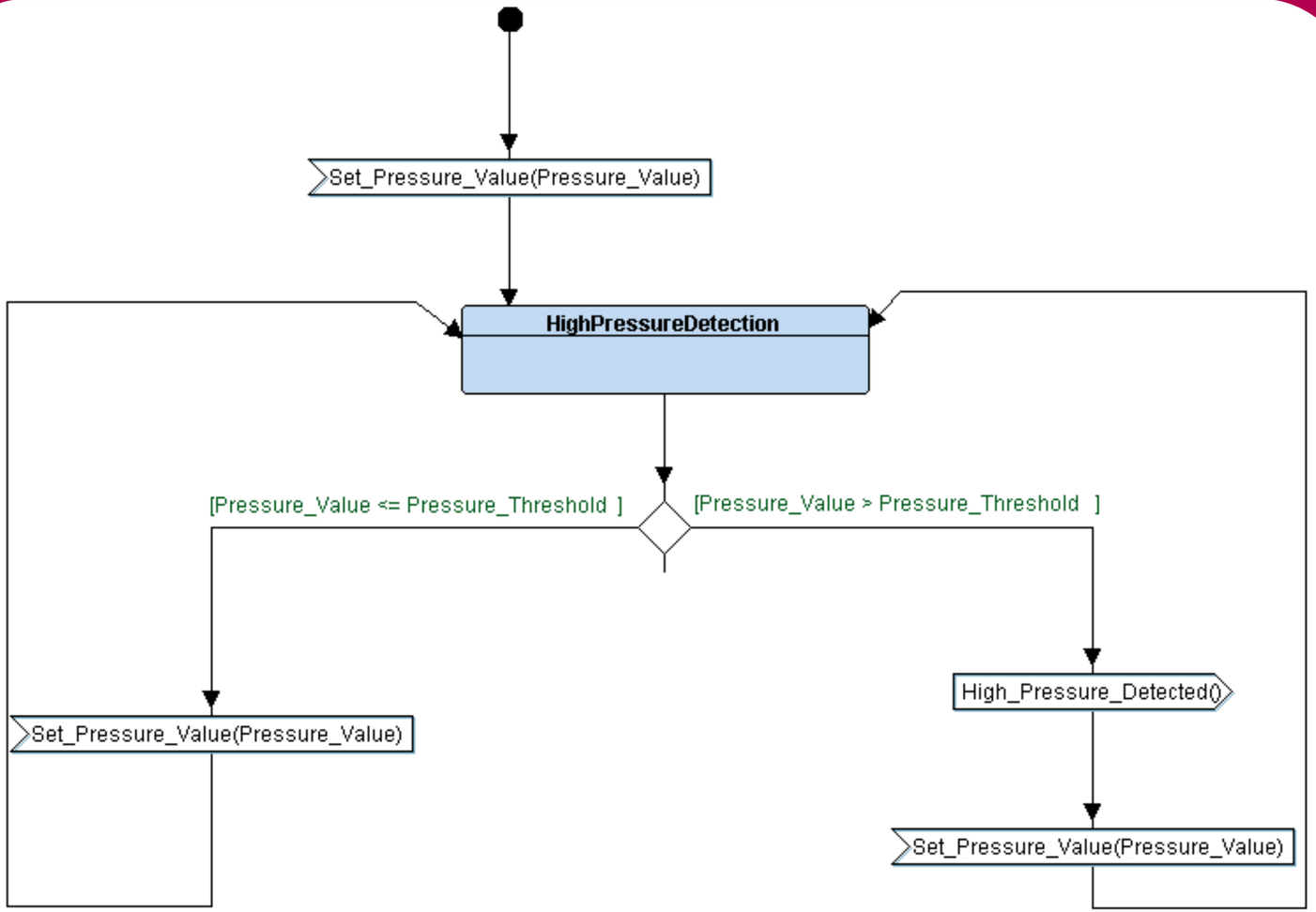


06

System Design

2

Main Program State Diagram

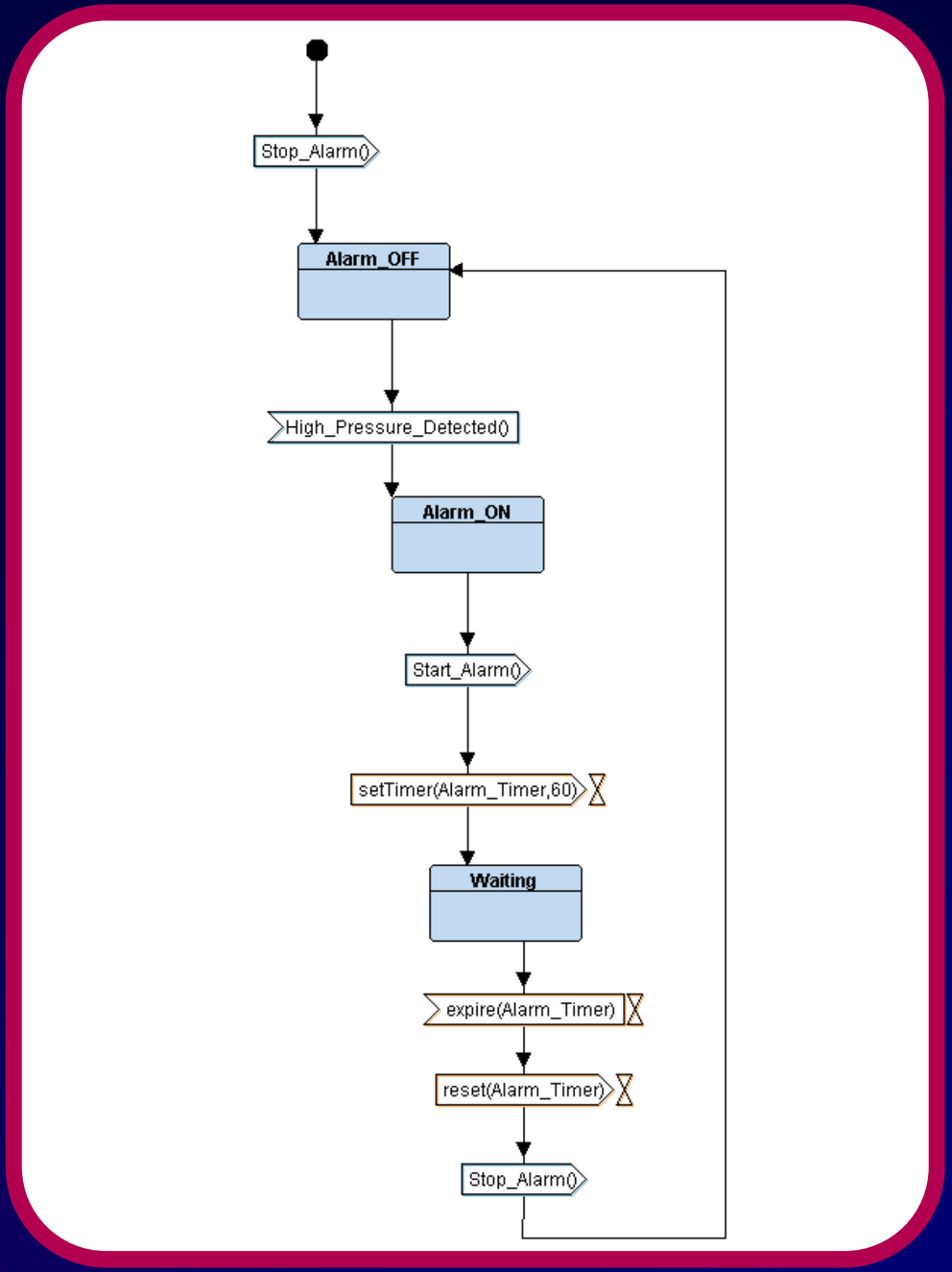


06

System Design

3

Alarm Monitor State Diagram

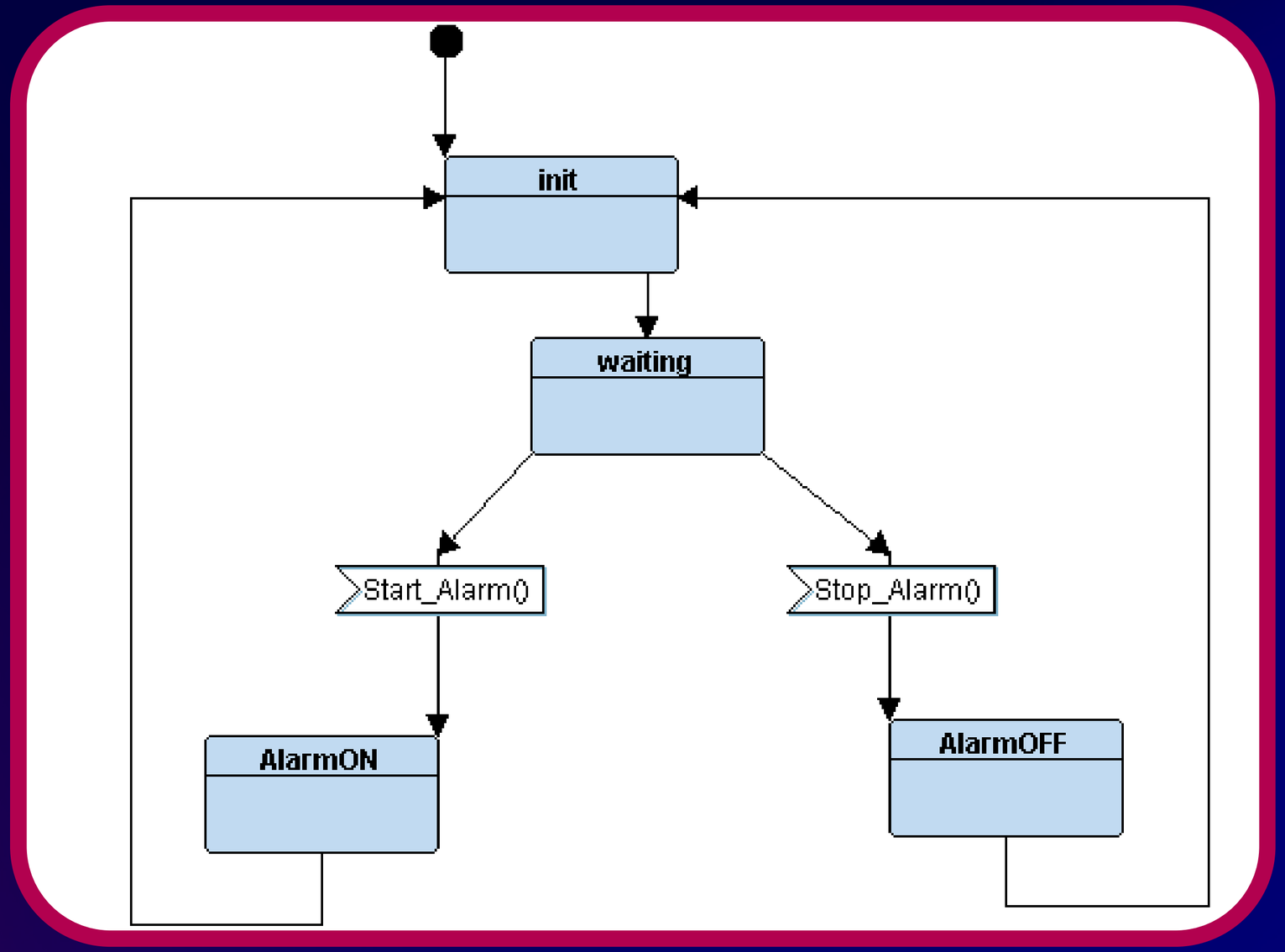


06

System Design

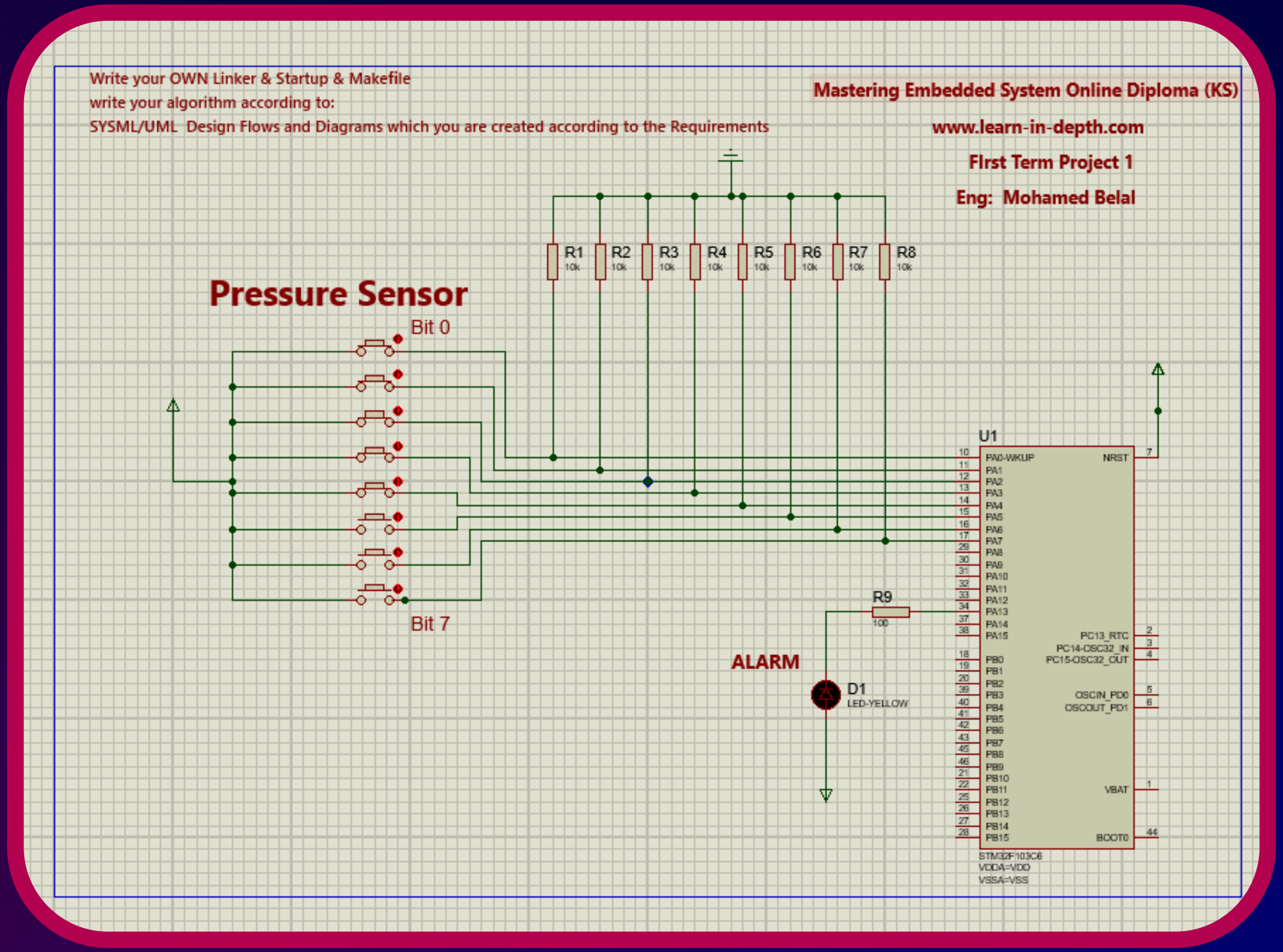
4

Alarm Actuator State Diagram



07

Design Simulation



08

Source Code

1

main

```c
/**
 ******************************************************************************
 * @project        : Pressure_Controller_MohamedBelal
 * @file           : main.c
 * @author         : mohamed belal
 * @Created on     : Jul 19, 2023
 ******************************************************************************
 */

#include <stdint.h>
#include <stdio.h>

#include "driver.h"
#include "Pressure_Sensor_Driver.h"
#include "main_Algorithm.h"
#include "Alarm_Monitor.h"
#include "Alarm_Actuator_Driver.h"

void setup()
{
    /*
     * 1 => init all the divers
     * 2 => init iqr .....
     * 3 => init hal us_driver dc_motor_driver
     * 4 => init block
     * 5 => set states pointers for each block
     * */
    pressure_sensor_driver_init();
    alarm_actuator_driver_init();

    pressure_sensor_driver_state = state_calling(pressure_sensor_driver_reading);
    main_algorithm_state = set_pressure_value;
    alarm_monitor_state = state_calling(alarm_monitor_alarm_off);
    alarm_actuator_driver_state = state_calling(alarm_actuator_driver_waiting);
}

int main()
{

    GPIO_INITIALIZATION();
    setup();
    while (1)
    {
        Pressure_Sensor_Driver_state();
        Main_Algorithm_state();
        Alarm_Monitor_state();
        Alarm_Actuator_Driver_state();
    }
    return 0;
}

```

08

Source Code

2

Pressure Sensor

```c
/**
 ******************************************************************************
 * @project        : Pressure_Controller_MohamedBelal
 * @file           : Pressure_Sensor_Driver.h
 * @author         : mohamed belal
 * @Created on     : Jul 19, 2023
 ******************************************************************************
 */

#include "Pressure_Sensor_Driver.h"

static int Pressure_Value = 0;
unsigned int Pressure_Sensor_Pull_Time = 100;
void (*Pressure_Sensor_Driver_state)();

void Pressure_Sensor_Driver_init()
{
    // GPIO_INITIALIZATION();
}

STATE_Define(Pressure_Sensor_Driver_Reading)
{
    // state_name
    Pressure_Sensor_Driver_state_id = Pressure_Sensor_Driver_Reading;

    Pressure_Value = getPressureVal();
    Set_Pressure_Value(Pressure_Value);

    Pressure_Sensor_Driver_state = STATE_calling(Pressure_Sensor_Driver_waiting);
}

STATE_Define(Pressure_Sensor_Driver_waiting)
{
    // state_name
    Pressure_Sensor_Driver_state_id = Pressure_Sensor_Driver_waiting;

    Delay(60000);

    Pressure_Sensor_Driver_state = STATE_calling(Pressure_Sensor_Driver_Reading);
}
```

```c
/**
 ******************************************************************************
 * @project        : Pressure_Controller_MohamedBelal
 * @file           : Pressure_Sensor_Driver.h
 * @author         : mohamed belal
 * @Created on     : Jul 19, 2023
 ******************************************************************************
 */

#ifndef PRESSURE_SENSOR_DRIVER_H_
#define PRESSURE_SENSOR_DRIVER_H_

#include "state.h"

// Define State
enum
{
    Pressure_Sensor_Driver_Reading,
    Pressure_Sensor_Driver_waiting,
}Pressure_Sensor_Driver_state_id;

// declare state function
void Pressure_Sensor_Driver_init();
STATE_Define(Pressure_Sensor_Driver_Reading);
STATE_Define(Pressure_Sensor_Driver_waiting);

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

#endif /* PRESSURE_SENSOR_DRIVER_H_ */
```

08

Source Code

3

main algorithm

```c
/**
 ******************************************************************************
 * @project        : Pressure_Controller_MohamedBelal
 * @file           : main_Algorithm.c
 * @author         : mohamed belal
 * @Created on     : Jul 19, 2023
 ******************************************************************************
 */
#include "main_Algorithm.h"
static int Main_Algorithm_Pressure_Value = 0;
static int Main_Algorithm_Pressure_Threshold = 20;
void (*Main_Algorithm_state)();

void Set_Pressure_Value(int Pressure_Val)
{
    Main_Algorithm_Pressure_Value = Pressure_Val;
    Main_Algorithm_state = STATE_calling(Main_Algorithm_High_Pressure_Detect);
}

// declare state function
STATE_Define(Main_Algorithm_High_Pressure_Detect)
{
    // state_name
    Main_Algorithm_state_id = Main_Algorithm_High_Pressure_Detect;


    if (Main_Algorithm_Pressure_Value <= Main_Algorithm_Pressure_Threshold)
    {
        Main_Algorithm_state = STATE_calling(Main_Algorithm_High_Pressure_Detect);
    }
    else
    {
        High_Pressure_Detected();
        Main_Algorithm_state = STATE_calling(Main_Algorithm_High_Pressure_Detect);
    }
}

```

```c
/**
 ******************************************************************************
 * @project        : Pressure_Controller_MohamedBelal
 * @file           : main_Algorithm.h
 * @author         : mohamed belal
 * @Created on     : Jul 19, 2023
 ******************************************************************************
 */

#ifndef MAIN_ALGORITHM_H_
#define MAIN_ALGORITHM_H_
#include "state.h"

// Define State
enum
{
    Main_Algorithm_High_Pressure_Detect,
} Main_Algorithm_state_id;


// declare state function
STATE_Define(Main_Algorithm_High_Pressure_Detect);


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

#endif /* MAIN_ALGORITHM_H_ */
```

08

Source Code

4

alarm Monitor

```h
/**
 ******************************************************************************
 * @project        : Pressure_Controller_MohamedBelal
 * @file           : Alarm_Monitor.h
 * @author         : mohamed belal
 * @Created on     : Jul 19, 2023
 ******************************************************************************
 */

#ifndef ALARM_MONITOR_H_
#define ALARM_MONITOR_H_

#include "state.h"

// Define State
enum
{
    Alarm_Monitor_alarm_off,
    Alarm_Monitor_alarm_on,
    Alarm_Monitor_waiting,
} Alarm_Monitor_state_id;


// declare state function
STATE_Define(Alarm_Monitor_alarm_off);
STATE_Define(Alarm_Monitor_alarm_on);
STATE_Define(Alarm_Monitor_waiting);


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

#endif /* ALARM_MONITOR_H_ */
```

```c
/**
 ******************************************************************************
 * @project        : Pressure_Controller_MohamedBelal
 * @file           : Alarm_Monitor.c
 * @author         : mohamed belal
 * @Created on     : Jul 19, 2023
 ******************************************************************************
 */

#include "Alarm_Monitor.h"
unsigned int Alarm_Periode = 600000;

void (*Alarm_Monitor_state)();

void High_Pressure_Detected()
{
    Alarm_Monitor_state = STATE_calling(Alarm_Monitor_alarm_on);
}

STATE_Define(Alarm_Monitor_alarm_off)
{
    Alarm_Monitor_state_id = Alarm_Monitor_alarm_off;

    Stop_Alarm();
    Alarm_Monitor_state = STATE_calling(Alarm_Monitor_alarm_off);
}

STATE_Define(Alarm_Monitor_alarm_on)
{
    Alarm_Monitor_state_id = Alarm_Monitor_alarm_on;

    Start_Alarm();
    Alarm_Monitor_state = STATE_calling(Alarm_Monitor_waiting);
}

STATE_Define(Alarm_Monitor_waiting)
{
    Alarm_Monitor_state_id = Alarm_Monitor_waiting;
    Stop_Alarm();
    Delay(Alarm_Periode);

    Alarm_Monitor_state = STATE_calling(Alarm_Monitor_alarm_off);
}
```

08

Source Code

5

alarm Actuator

```c
/**
 ******************************************************************************
 * @project        : Pressure_Controller_MohamedBelal
 * @file           : Alarm_Actuator_Driver.h
 * @author         : mohamed belal
 * @Created on     : Jul 19, 2023
 ******************************************************************************
 */

#ifndef ALARM_ACTUATOR_DRIVER_H_
#define ALARM_ACTUATOR_DRIVER_H_

#include "state.h"

// Define State
enum
{
    Alarm_Actuator_Driver_alarm_off,
    Alarm_Actuator_Driver_alarm_on,
    Alarm_Actuator_Driver_waiting,
} Alarm_Actuator_Driver_state_id;


// declare state function
void Alarm_Actuator_Driver_init();
STATE_Define(Alarm_Actuator_Driver_alarm_off);
STATE_Define(Alarm_Actuator_Driver_alarm_on);
STATE_Define(Alarm_Actuator_Driver_waiting);


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

#endif /* ALARM_ACTUATOR_DRIVER_H_ */
```

```c
/**
 ******************************************************************************
 * @project        : Pressure_Controller_MohamedBelal
 * @file           : Alarm_Actuator_Driver.c
 * @author         : mohamed belal
 * @Created on     : Jul 19, 2023
 ******************************************************************************
 */

#include "Alarm_Actuator_Driver.h"
void (*Alarm_Actuator_Driver_state)();

void Stop_Alarm()
{

    Alarm_Actuator_Driver_state = STATE_calling(Alarm_Actuator_Driver_alarm_off);
}

void Start_Alarm()
{

    Alarm_Actuator_Driver_state = STATE_calling(Alarm_Actuator_Driver_alarm_on);
}

// declare state function
void Alarm_Actuator_Driver_init()
{
    // GPIO_INITIALIZATION();
}

STATE_Define(Alarm_Actuator_Driver_waiting)
{
    Alarm_Actuator_Driver_state_id = Alarm_Actuator_Driver_waiting;

    Alarm_Actuator_Driver_state = STATE_calling(Alarm_Actuator_Driver_waiting);
}

STATE_Define(Alarm_Actuator_Driver_alarm_off)
{
    Alarm_Actuator_Driver_state_id = Alarm_Actuator_Driver_alarm_off;

    // we send (1) beacase of we connect led with power
    // then to reset the led we send (1) as (5 volte) and another bolarty is (5 volte) this main that (0 volte)  turn off led
    Set_Alarm_actuator(1);
    Alarm_Actuator_Driver_state = STATE_calling(Alarm_Actuator_Driver_waiting);
}

STATE_Define(Alarm_Actuator_Driver_alarm_on)
{
    Alarm_Actuator_Driver_state_id = Alarm_Actuator_Driver_alarm_on;

    // we send (0) beacase of we connect led with power
    // then to set the led we send (0) as (0 volte) and another bolarty is (5 volte) this main that (5 volte)  turn off led
    Set_Alarm_actuator(0);
    Alarm_Actuator_Driver_state = STATE_calling(Alarm_Actuator_Driver_waiting);
}

```

08

Source Code

6

State

```c
/**
 ******************************************************************************
 * @project        : Pressure_Controller_MohamedBelal
 * @file           : state.h
 * @author         : mohamed belal
 * @Created on     : Jul 19, 2023
 ******************************************************************************
 */

#ifndef STATE_H_
#define STATE_H_

#include <stdio.h>
#include <stdlib.h>
#include "driver.h"

#define STATE_Define(_stateFUN_) void ST_##_stateFUN_()
#define STATE_calling(_stateFUN_) ST_##_stateFUN_

// state connection
//we use this for signal to connect eachblock with another
void Set_Pressure_Value(int Pressure_Val);
void High_Pressure_Detected();
void Stop_Alarm();
void Start_Alarm();

#endif /* STATE_H_ */

```

08

Source Code

7

alarm Actuator

```h
#include <stdint.h>
#include <stdio.h>

#define SET_BIT(ADDRESS,BIT)   ADDRESS |=  (1<<BIT)
#define RESET_BIT(ADDRESS,BIT) ADDRESS &= ~(1<<BIT)
#define TOGGLE_BIT(ADDRESS,BIT)  ADDRESS ^=  (1<<BIT)
#define READ_BIT(ADDRESS,BIT) ((ADDRESS) &   (1<<(BIT)))


#define GPIO_PORTA 0x40010800
#define BASE_RCC   0x40021000

#define APB2ENR   *(volatile uint32_t *)(BASE_RCC + 0x18)

#define GPIOA_CRL *(volatile uint32_t *)(GPIO_PORTA + 0x00)
#define GPIOA_CRH *(volatile uint32_t *)(GPIO_PORTA + 0X04)
#define GPIOA_IDR *(volatile uint32_t *)(GPIO_PORTA + 0x08)
#define GPIOA_ODR *(volatile uint32_t *)(GPIO_PORTA + 0x0C)


void Delay(int nCount);
int getPressureVal();
void Set_Alarm_actuator(int i);
void GPIO_INITIALIZATION ();

```

```c
#include "driver.h"
#include <stdint.h>
#include <stdio.h>
void Delay(int nCount)
{
    for (; nCount != 0; nCount--);
}

int getPressureVal()
{
    // Delay(40000);
    // TOGGLE_BIT(GPIOA_ODR, 13);
    // Delay(40000);
    return (GPIOA_IDR & 0xFF);
    // return 10;
}

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;
}

```

Sections for each object file

1 Main

```
Windows PowerShell
PS D:\00_Embedded System learn-in-depth\05_unit 5 first term exam\pr

main.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         00000098  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  000000cc  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000cc  2**0
                  ALLOC
  3 .debug_info   0000018b  00000000  00000000  000000cc  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 000000a5  00000000  00000000  00000257  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    00000058  00000000  00000000  000002fc  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000020  00000000  00000000  00000354  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line   000000b7  00000000  00000000  00000374  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str    000002dc  00000000  00000000  0000042b  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      00000012  00000000  00000000  00000707  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  00000719  2**0
                  CONTENTS, READONLY
 11 .debug_frame  00000048  00000000  00000000  0000074c  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

2 Pressure Sensor Driver

```
Windows PowerShell
PS D:\00_Embedded System learn-in-depth\05_unit 5 first term exam\
ensor_Driver.o

Pressure_Sensor_Driver.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         00000084  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000004  00000000  00000000  000000b8  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000004  00000000  00000000  000000bc  2**2
                  ALLOC
  3 .debug_info   00000122  00000000  00000000  000000bc  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 000000a1  00000000  00000000  000001de  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    00000084  00000000  00000000  0000027f  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000020  00000000  00000000  00000303  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line   00000070  00000000  00000000  00000323  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str    00000213  00000000  00000000  00000393  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      00000012  00000000  00000000  000005a6  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  000005b8  2**0
                  CONTENTS, READONLY
 11 .debug_frame  00000060  00000000  00000000  000005ec  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

3 main Algorithm

```
Windows PowerShell
PS D:\00_Embedded System learn-in-depth\05_unit 5 first term exam\
ithm.o

main_Algorithm.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         00000088  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000004  00000000  00000000  000000bc  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000004  00000000  00000000  000000c0  2**2
                  ALLOC
  3 .debug_info   0000011a  00000000  00000000  000000c0  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 000000b4  00000000  00000000  000001da  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    00000064  00000000  00000000  0000028e  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000020  00000000  00000000  000002f2  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line   0000005e  00000000  00000000  00000312  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str    000001df  00000000  00000000  00000370  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      00000012  00000000  00000000  0000054f  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  00000561  2**0
                  CONTENTS, READONLY
 11 .debug_frame  00000048  00000000  00000000  00000594  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

4 Alarm Monitor

```
Windows PowerShell
PS D:\00_Embedded System learn-in-depth\05_unit 5 first term exam\p
tor.o

Alarm_Monitor.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         000000b0  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000004  00000000  00000000  000000e4  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000e8  2**0
                  ALLOC
  3 .debug_info   0000012c  00000000  00000000  000000e8  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 00000092  00000000  00000000  00000214  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    000000b0  00000000  00000000  000002a6  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000020  00000000  00000000  00000356  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line   00000064  00000000  00000000  00000376  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str    000001e9  00000000  00000000  000003da  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      00000012  00000000  00000000  000005c3  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  000005d5  2**0
                  CONTENTS, READONLY
 11 .debug_frame  0000007c  00000000  00000000  00000608  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

5 Alarm Actuator Driver

```
Windows PowerShell
PS D:\00_Embedded System learn-in-depth\05_unit 5 first term exam\pr
ator_Driver.o

Alarm_Actuator_Driver.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         000000d0  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  00000104  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  00000104  2**0
                  ALLOC
  3 .debug_info   00000144  00000000  00000000  00000104  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 00000092  00000000  00000000  00000248  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    00000108  00000000  00000000  000002da  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000020  00000000  00000000  000003e2  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line   00000077  00000000  00000000  00000402  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str    0000023e  00000000  00000000  00000479  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      00000012  00000000  00000000  000006b7  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  000006c9  2**0
                  CONTENTS, READONLY
 11 .debug_frame  000000a8  00000000  00000000  000006fc  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

6 startup

```
Windows PowerShell
PS D:\00_Embedded System learn-in-depth\05_unit 5 first term exam\p

startup.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         000000c0  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  000000f4  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000f4  2**0
                  ALLOC
  3 .vectors      0000001c  00000000  00000000  000000f4  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, DATA
  4 .debug_info   00000164  00000000  00000000  00000110  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  5 .debug_abbrev 000000e1  00000000  00000000  00000274  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_loc    00000064  00000000  00000000  00000355  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_aranges 00000020  00000000  00000000  000003b9  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_line   00000069  00000000  00000000  000003d9  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  9 .debug_str    00000107  00000000  00000000  00000442  2**0
                  CONTENTS, READONLY, DEBUGGING
 10 .comment      00000012  00000000  00000000  00000549  2**0
                  CONTENTS, READONLY
 11 .ARM.attributes 00000033  00000000  00000000  0000055b  2**0
                  CONTENTS, READONLY
 12 .debug_frame  0000004c  00000000  00000000  00000590  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

Symbol Table for each object file

1 Main

```
Windows PowerShell
PS D:\00_Embedded System learn-in-depth\05_unit 5
         U Alarm_Actuator_Driver_init
         U Alarm_Actuator_Driver_state
00000001 C Alarm_Actuator_Driver_state_id
         U Alarm_Monitor_state
00000001 C Alarm_Monitor_state_id
         U GPIO_INITIALIZATION
00000058 T main
         U Main_Algorithm_state
00000001 C Main_Algorithm_state_id
         U Pressure_Sensor_Driver_init
         U Pressure_Sensor_Driver_state
00000001 C Pressure_Sensor_Driver_state_id
         U Set_Pressure_Value
00000000 T setup
         U ST_Alarm_Actuator_Driver_waiting
         U ST_Alarm_Monitor_alarm_off
         U ST_Pressure_Sensor_Driver_Reading
```

2 Pressure Sensor Driver

```
Windows PowerShell
PS D:\00_Embedded System learn-in-depth\05_unit 5 f
         U Delay
         U getPressureVal
00000000 T Pressure_Sensor_Driver_init
00000004 C Pressure_Sensor_Driver_state
00000001 C Pressure_Sensor_Driver_state_id
00000000 D Pressure_Sensor_Pull_Time
00000000 b Pressure_Value
         U Set_Pressure_Value
0000000c T ST_Pressure_Sensor_Driver_Reading
00000054 T ST_Pressure_Sensor_Driver_waiting
```

3 main Algorithm

```
Windows PowerShell
PS D:\00_Embedded System learn-in-depth\05_unit 5
         U High_Pressure_Detected
00000000 d Main_Algorithm_Pressure_Threshold
00000000 b Main_Algorithm_Pressure_Value
00000004 C Main_Algorithm_state
00000001 C Main_Algorithm_state_id
00000000 T Set_Pressure_Value
00000030 T ST_Main_Algorithm_High_Pressure_Detect
```

4 Alarm Monitor

```
Windows PowerShell
PS D:\00_Embedded System learn-in-depth\
00000004 C Alarm_Monitor_state
00000001 C Alarm_Monitor_state_id
00000000 D Alarm_Periode
         U Delay
00000000 T High_Pressure_Detected
0000001c T ST_Alarm_Monitor_alarm_off
00000048 T ST_Alarm_Monitor_alarm_on
00000074 T ST_Alarm_Monitor_waiting
         U Start_Alarm
         U Stop_Alarm
```

5 Alarm Actuator Driver

```
Windows PowerShell
PS D:\00_Embedded System learn-in-depth\05_unit 5 firs
00000038 T Alarm_Actuator_Driver_init
00000004 C Alarm_Actuator_Driver_state
00000001 C Alarm_Actuator_Driver_state_id
         U Set_Alarm_actuator
00000070 T ST_Alarm_Actuator_Driver_alarm_off
000000a0 T ST_Alarm_Actuator_Driver_alarm_on
00000044 T ST_Alarm_Actuator_Driver_waiting
0000001c T Start_Alarm
00000000 T Stop_Alarm
```

6 startup

```
Windows PowerShell
PS D:\00_Embedded System learn-in-depth\05_unit 5 first term exam\p

startup.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         000000c0  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  000000f4  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000f4  2**0
                  ALLOC
  3 .vectors      0000001c  00000000  00000000  000000f4  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, DATA
  4 .debug_info   00000164  00000000  00000000  00000110  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  5 .debug_abbrev 000000e1  00000000  00000000  00000274  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_loc    00000064  00000000  00000000  00000355  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_aranges 00000020  00000000  00000000  000003b9  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_line   00000069  00000000  00000000  000003d9  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  9 .debug_str    00000107  00000000  00000000  00000442  2**0
                  CONTENTS, READONLY, DEBUGGING
 10 .comment      00000012  00000000  00000000  00000549  2**0
                  CONTENTS, READONLY
 11 .ARM.attributes 00000033  00000000  00000000  0000055b  2**0
                  CONTENTS, READONLY
 12 .debug_frame  0000004c  00000000  00000000  00000590  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

sections

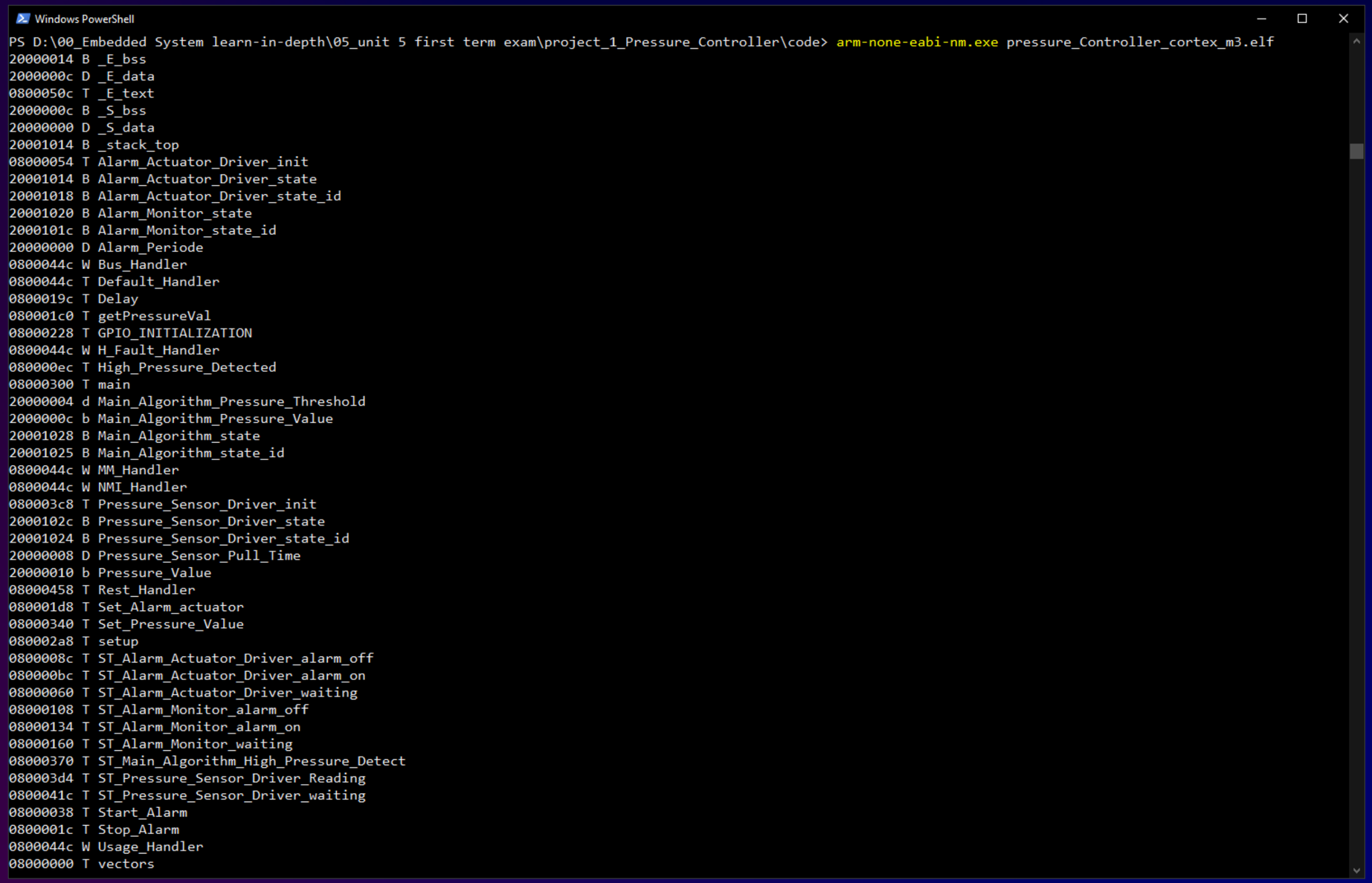
Windows PowerShell

```
PS D:\00_Embedded System learn-in-depth\05_unit 5 first term exam\project_1_Pressure_Controller\code> arm-none-eabi-objdump.exe -h pressure_Controller_cortex_m3.elf

pressure_Controller_cortex_m3.elf:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         0000050c  08000000  08000000  00008000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         0000000c  20000000  0800050c  00010000  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00001024  2000000c  08000518  0001000c  2**2
                  ALLOC
  3 .debug_info   0000089e  00000000  00000000  0001000c  2**0
                  CONTENTS, READONLY, DEBUGGING
  4 .debug_abbrev 0000049c  00000000  00000000  000108aa  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    00000424  00000000  00000000  00010d46  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 000000e0  00000000  00000000  0001116a  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_line   00000362  00000000  00000000  0001124a  2**0
                  CONTENTS, READONLY, DEBUGGING
  8 .debug_str    000004d1  00000000  00000000  000115ac  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      00000011  00000000  00000000  00011a7d  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  00011a8e  2**0
                  CONTENTS, READONLY
 11 .debug_frame  000002d8  00000000  00000000  00011ac4  2**2
                  CONTENTS, READONLY, DEBUGGING
```

Symbol Table



Hardware Simulation

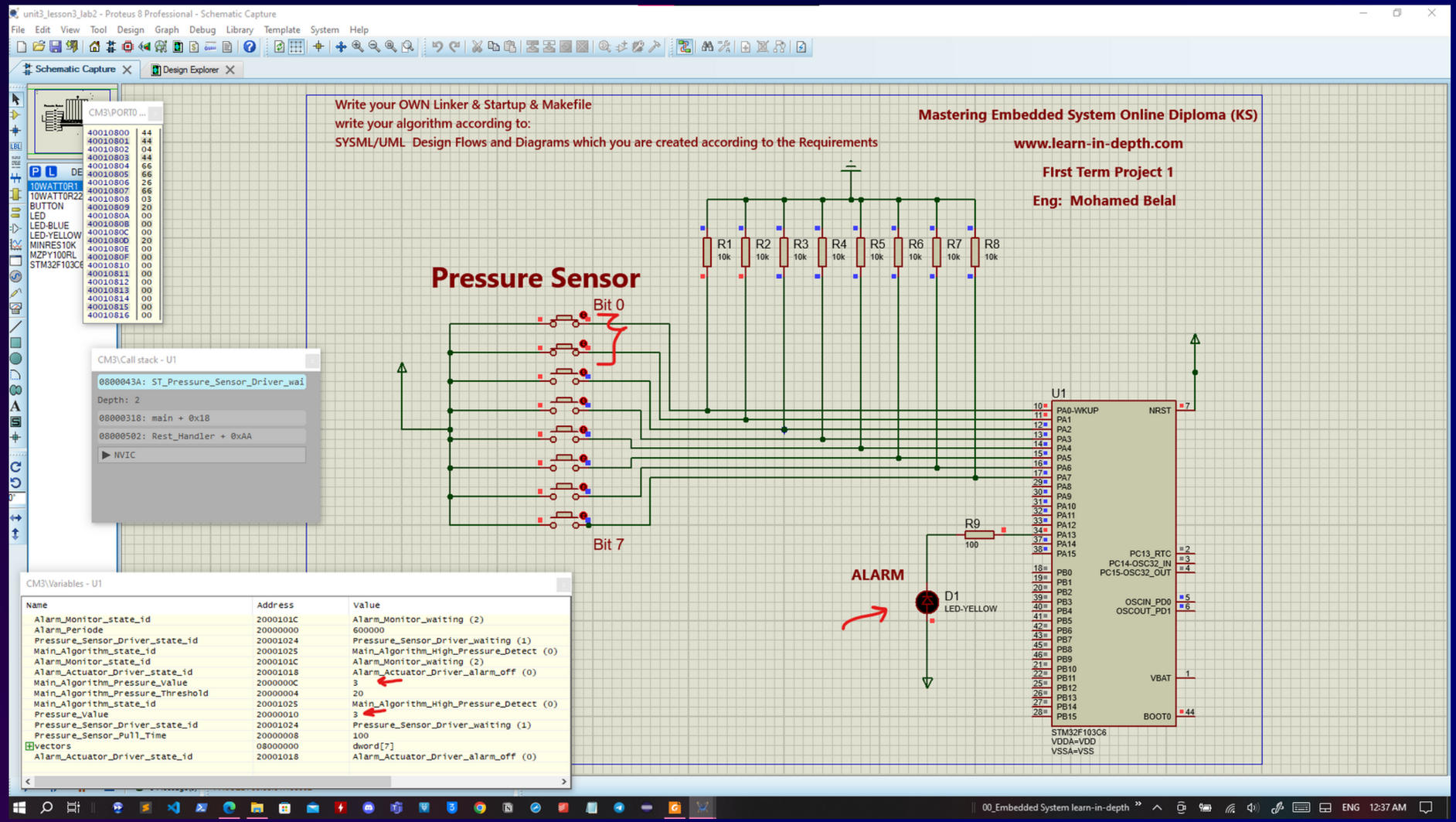
1

If

pressureVal <= Threshold

Alarm OFF

Threshold = 20



Hardware Simulation

2

If

pressureVal > Threshold

Alarm ON

Threshold = 20