# ENG. MOHAMED HAMDY

MASTERING EMBEDDED SYSTEM ONLINE DIPLOMA
WWW.LEARN-IN-DEPTH.COM

## FIRST TERM (FINAL PROJECT 1) – PRESSURE DETECTION SYSTEM

MY PROFILE:
HTTPS://WWW.LEARN-IN-DEPTH-STORE.COM/PROFILE/MOHAMEDHAMDY-M-H-M-A/PROFILE

GITHUB REPO:
HTTPS://GITHUB.COM/MOHAMED-HAMDY-MA/MASTERING_EMBEDDED_SYSTEM.GIT

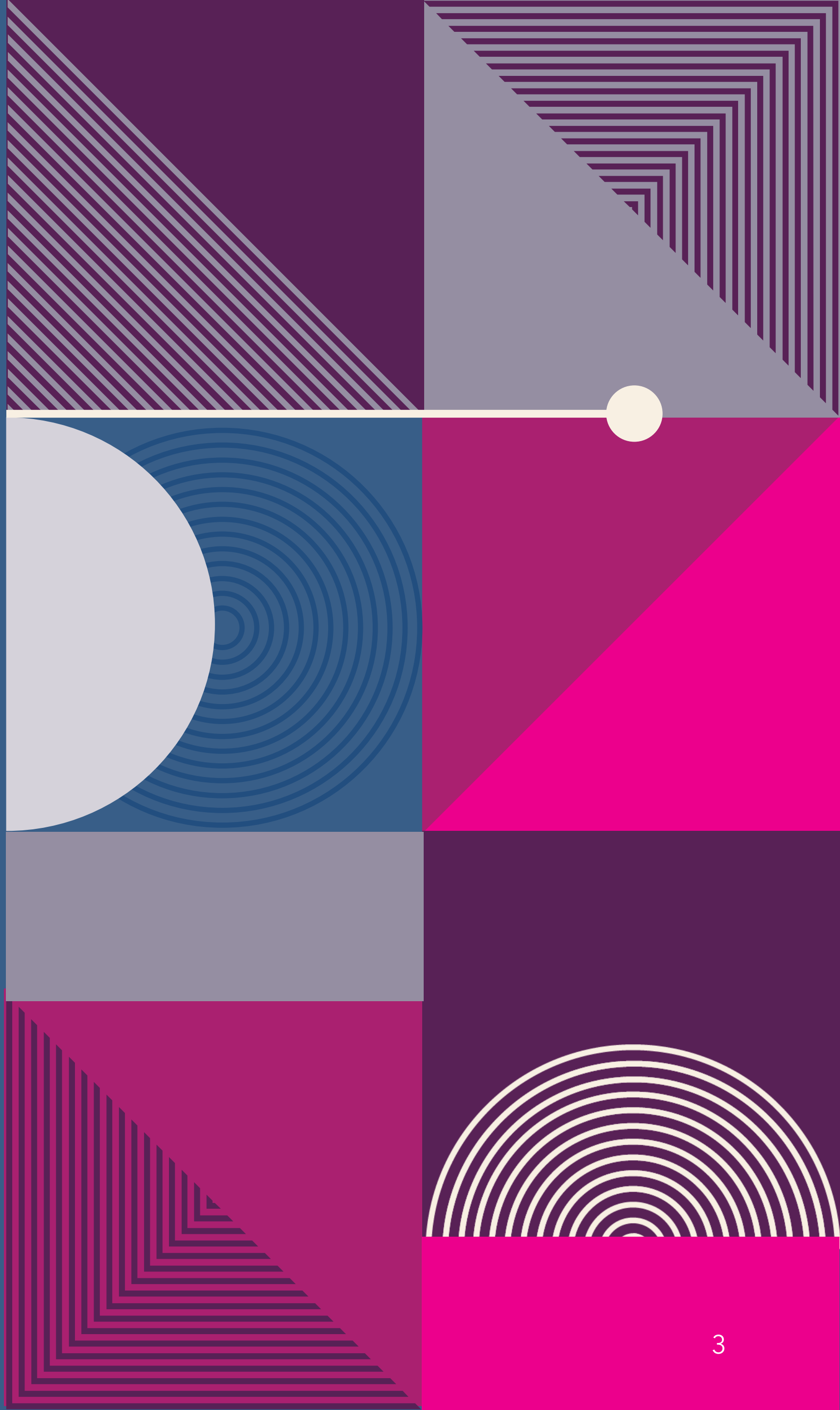
# AGENDA

1. Project objective
2. Project idea
3. System architecture
4. Implementation
5. Simulation
6. Binary utilities

# 1.PROJECT OBJECTIVE

The project covers topics including:

- Embedded C.
- Bare-metal software build process.
- Arm-none-eabi toolchain
- Startup.
- Linker script.
- Make File.
- System architecture using UML.



# 2.PROJECT IDEA

A “client” expects you to deliver the software of the following system:

- ➢ A pressure detection system informs the crew of a cabin with alarm when the pressure exceeds 15 bars in the cabin.
- ➢ The alarm duration equals 60 seconds.

# 3.SYSTEM ARCHITECTURE

- 3.1.Case Study
- 3.2.Method.
- 3.3.Requirements.
  - Requirements Diagram
- 3.4.Space Exploration
- 3.5.System Analysis
  - Use Case Diagram.
  - Activity Diagram.
  - Sequence Diagram.
- 3.6.System Design
  - Block Diagram
  - State machine Design

# 3.1.CASE STUDY

Pressure Detection System Assumptions:

- ➢ The controller set up and shutdown procedures are not modeled.
- ➢ The controller maintenance is not modeled.
- ➢ The pressure sensor never fails.
- ➢ The alarm never fails.
- ➢ The controller never faces power cut.

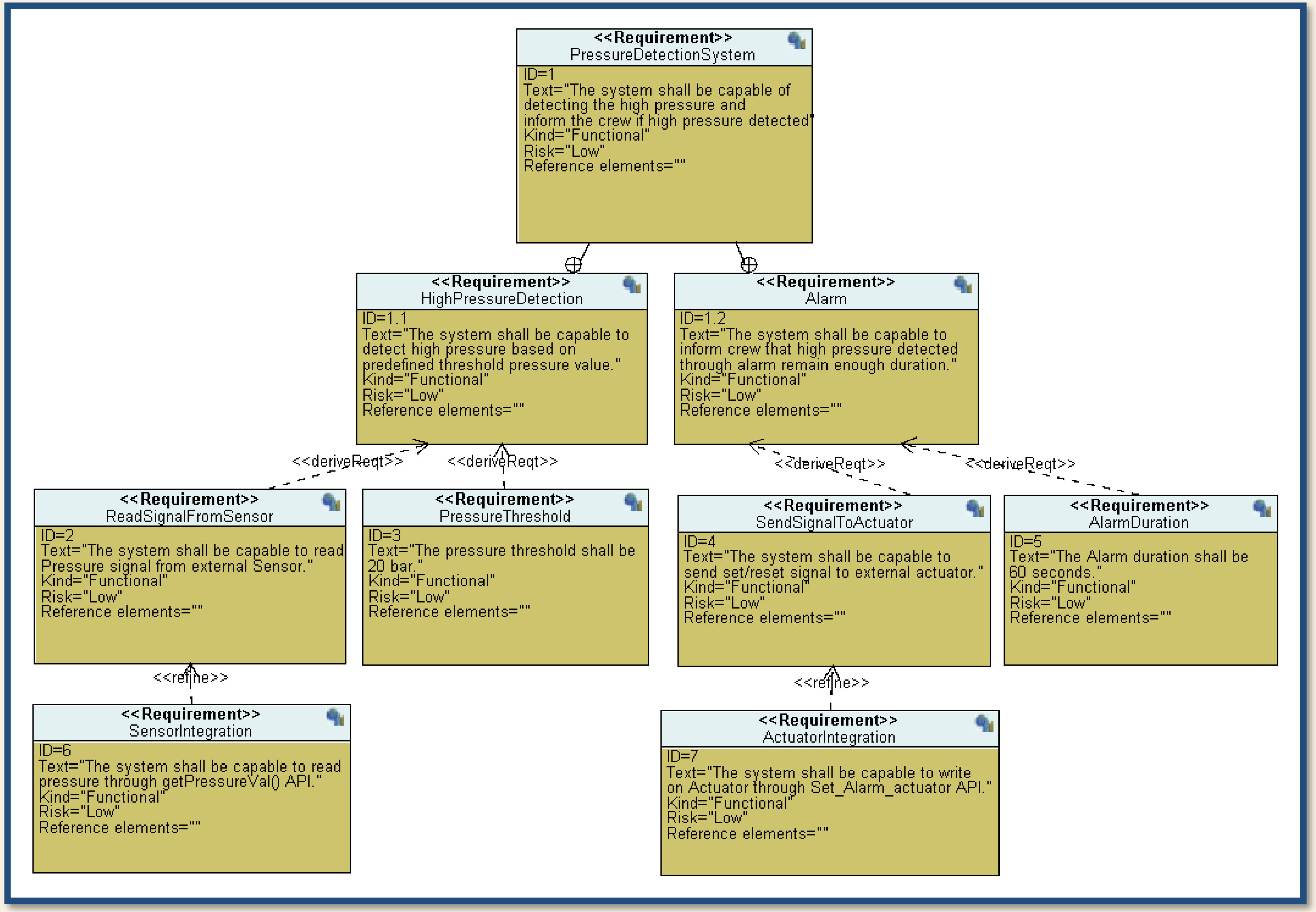
# 3.2.METHOD

The SW development can follow one of these life cycle:

- Waterfall
- V-model
- Agile
- Rapid Application Development(RAD)
- Spiral

# 3.3.REQUIREMENTS

## ➢ 3.3.1.REQUIREMENTS DIAGRAM



<<Requirement>>
PressureDetectionSystem
ID=1
Text="The system shall be capable of detecting the high pressure and inform the crew if high pressure detected"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
HighPressureDetection
ID=1.1
Text="The system shall be capable to detect high pressure based on predefined threshold pressure value."
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Alarm
ID=1.2
Text="The system shall be capable to inform crew that high pressure detected through alarm remain enough duration."
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
ReadSignalFromSensor
ID=2
Text="The system shall be capable to read Pressure signal from external Sensor."
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
PressureThreshold
ID=3
Text="The pressure threshold shall be 20 bar."
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
SendSignalToActuator
ID=4
Text="The system shall be capable to send set/reset signal to external actuator."
Kind="Functional"
Risk="Low"
Reference elements=""

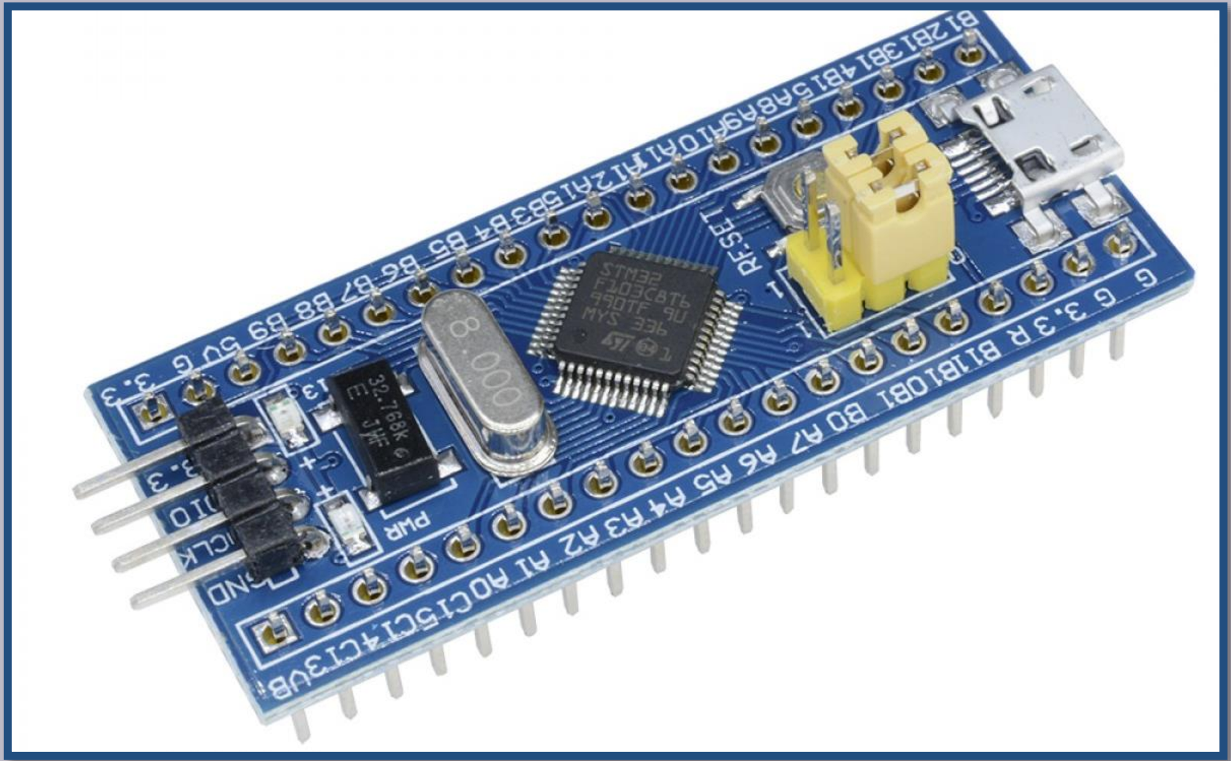
<<Requirement>>
AlarmDuration
ID=5
Text="The Alarm duration shall be 60 seconds."
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
SensorIntegration
ID=6
Text="The system shall be capable to read pressure through getPressureVal() API."
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
ActuatorIntegration
ID=7
Text="The system shall be capable to write on Actuator through Set_Alarm_actuator API."
Kind="Functional"
Risk="Low"
Reference elements=""

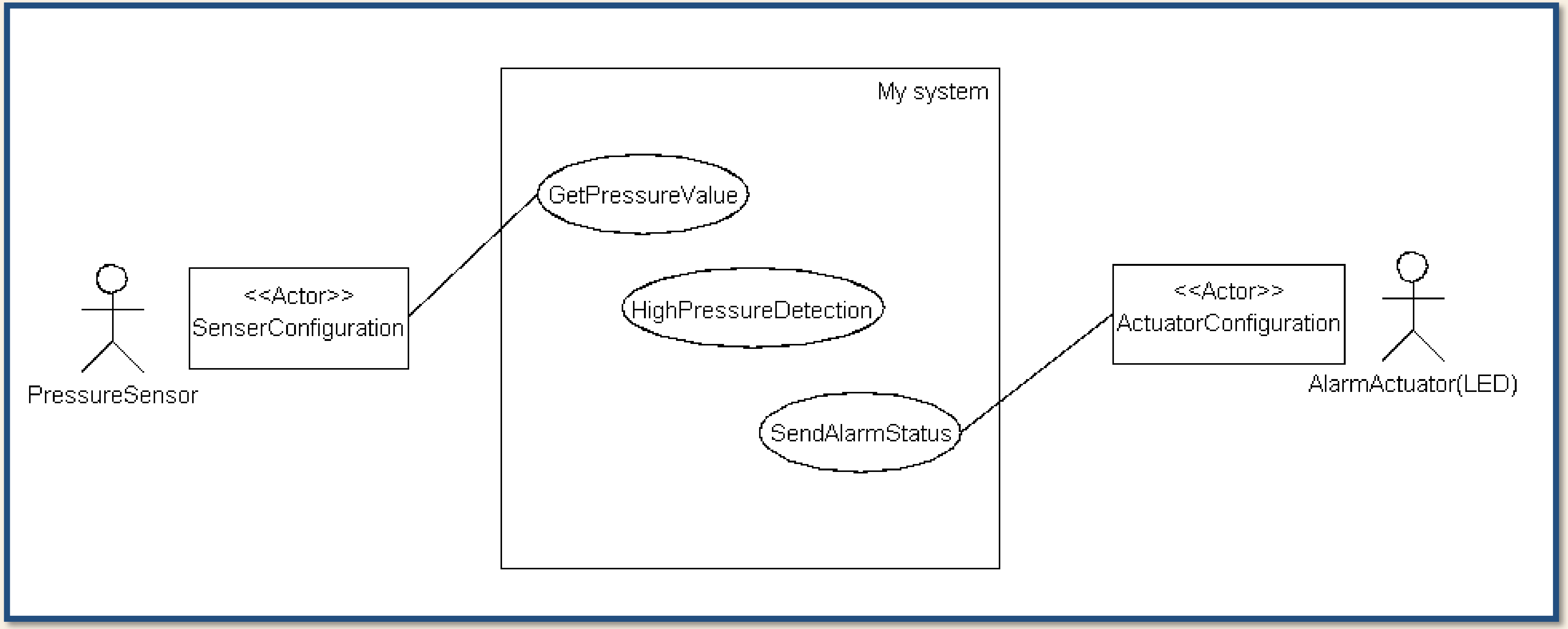
# 3.4.SPACE EXPLORATION

The development will be implemented using STM32 board with ARM-CORTEX-M3 core.


STM32
F103C8T6
8.000
32.768K
RESET
PWR
3.3
G
5V
B9 B8 B7 B6 B5 B4 B3 A15 A12 A11 A10 A9 A8 B15 B14 B13 B12
VB C13 C14 C15 A0 A1 A2 A3 A4 A5 A6 A7 B0 B1 B10 B11 R 3.3 G G
GND
CLK
DIO
3.3

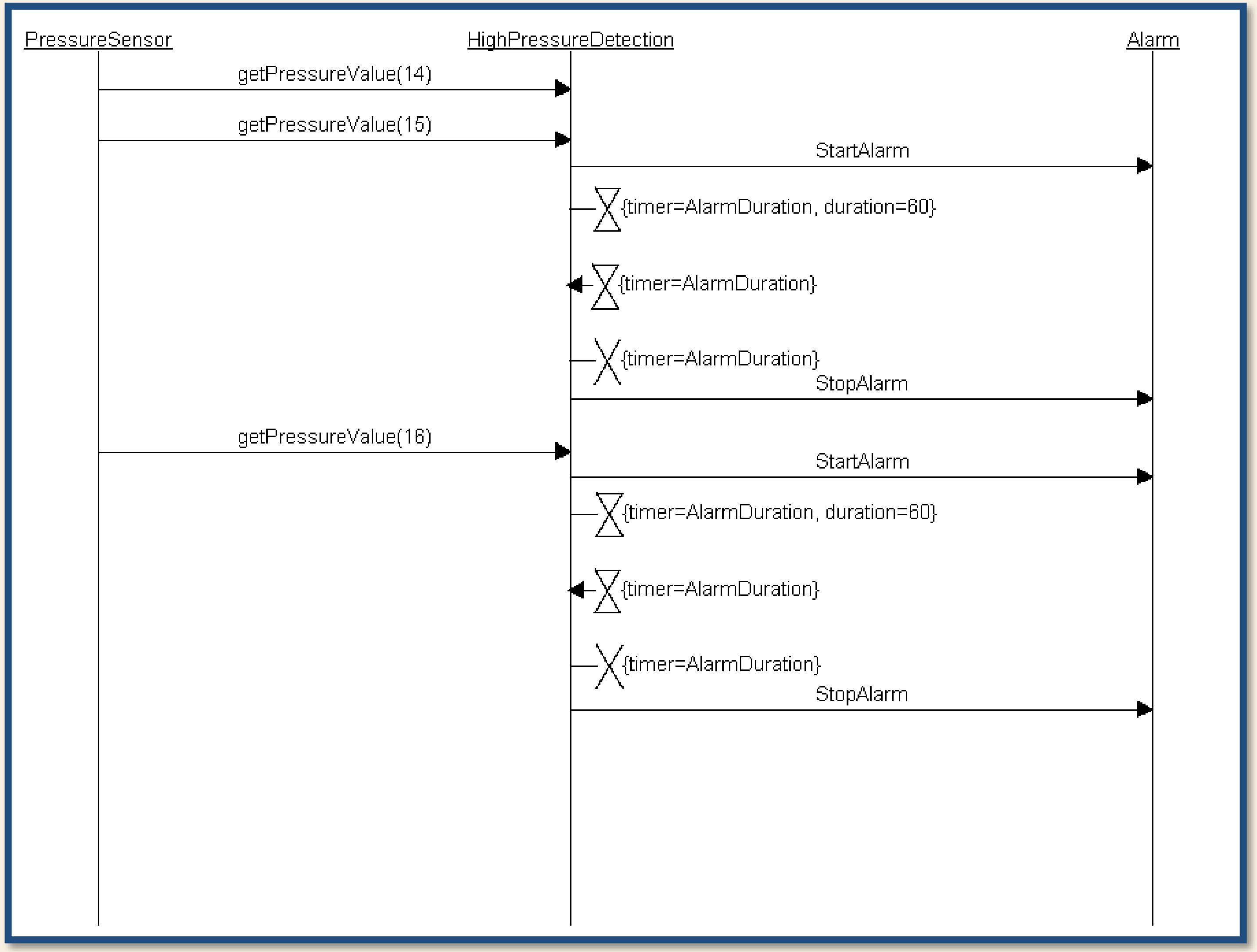
# 3.5.SYSTEM ANALYSIS

## ➢ 3.5.1.USE CASE DIAGRAM



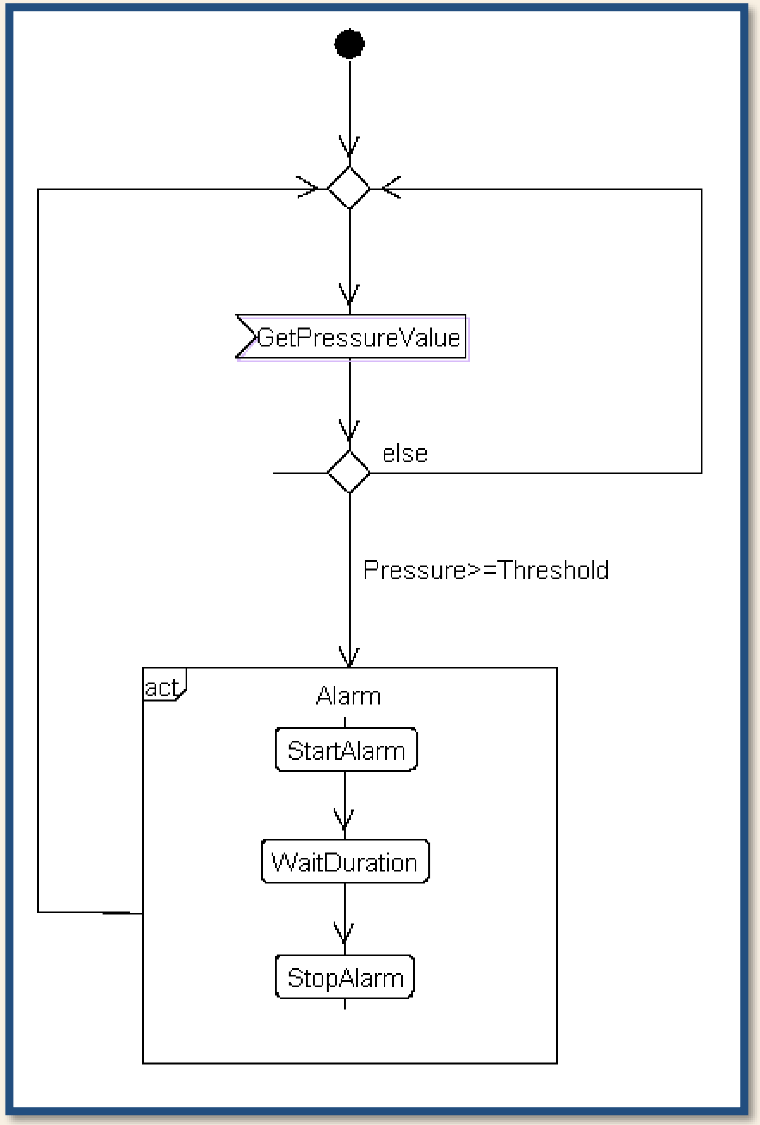
# 3.5.SYSTEM ANALYSIS

## ➢ 3.5.2.SEQUENCE DIAGRAM



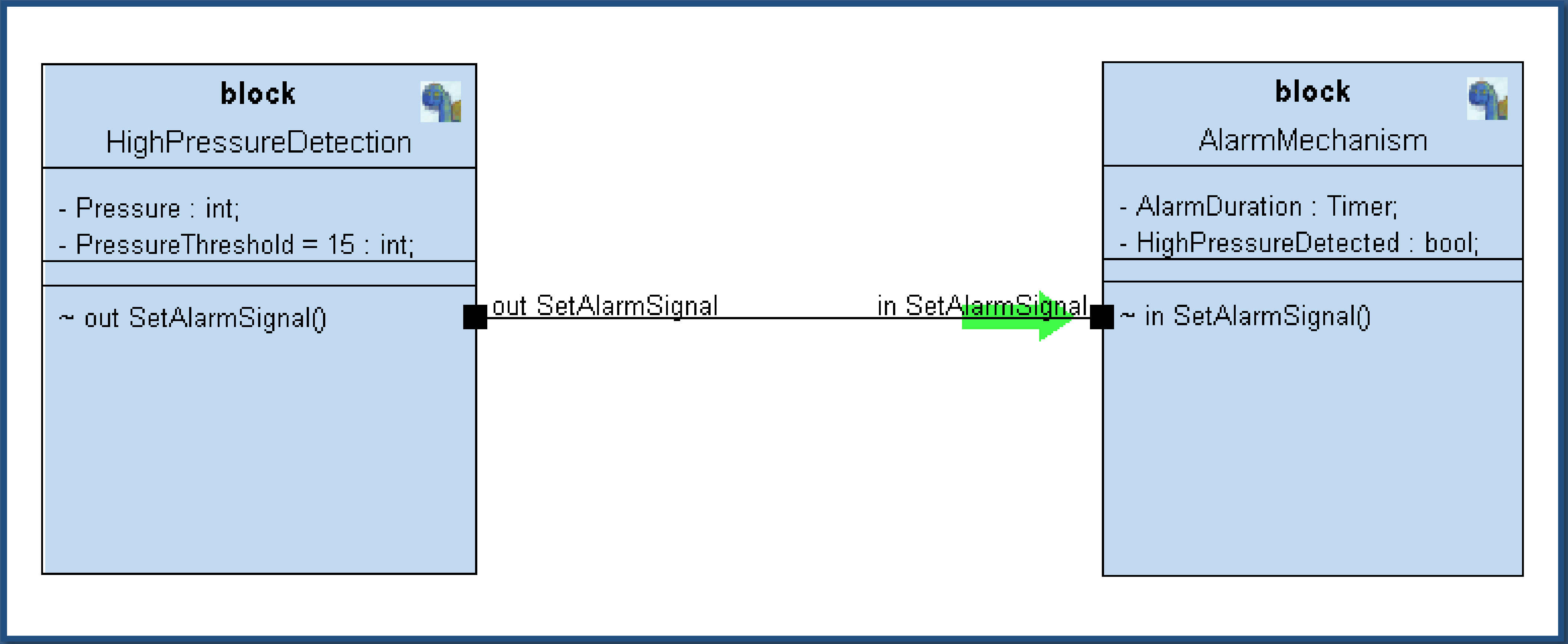
# 3.5.SYSTEM ANALYSIS

## ➢ 3.5.3.ACTIVITY DIAGRAM



# 3.6.SYSTEM DESIGN

## ➢ 3.6.1.BLOCK DIAGRAM



**block**

HighPressureDetection

- Pressure : int;
- PressureThreshold = 15 : int;

~ out SetAlarmSignal()

out SetAlarmSignal

in SetAlarmSignal

**block**

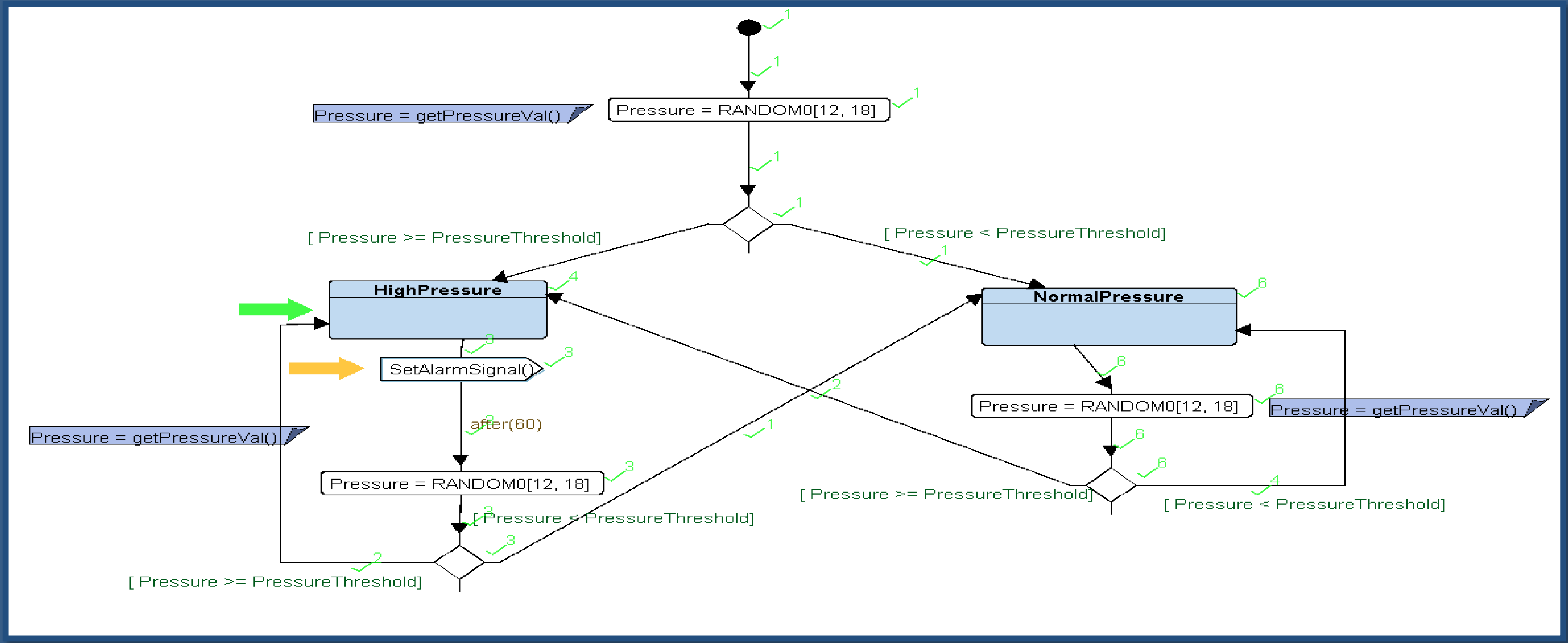
AlarmMechanism

- AlarmDuration : Timer;
- HighPressureDetected : bool;

~ in SetAlarmSignal()

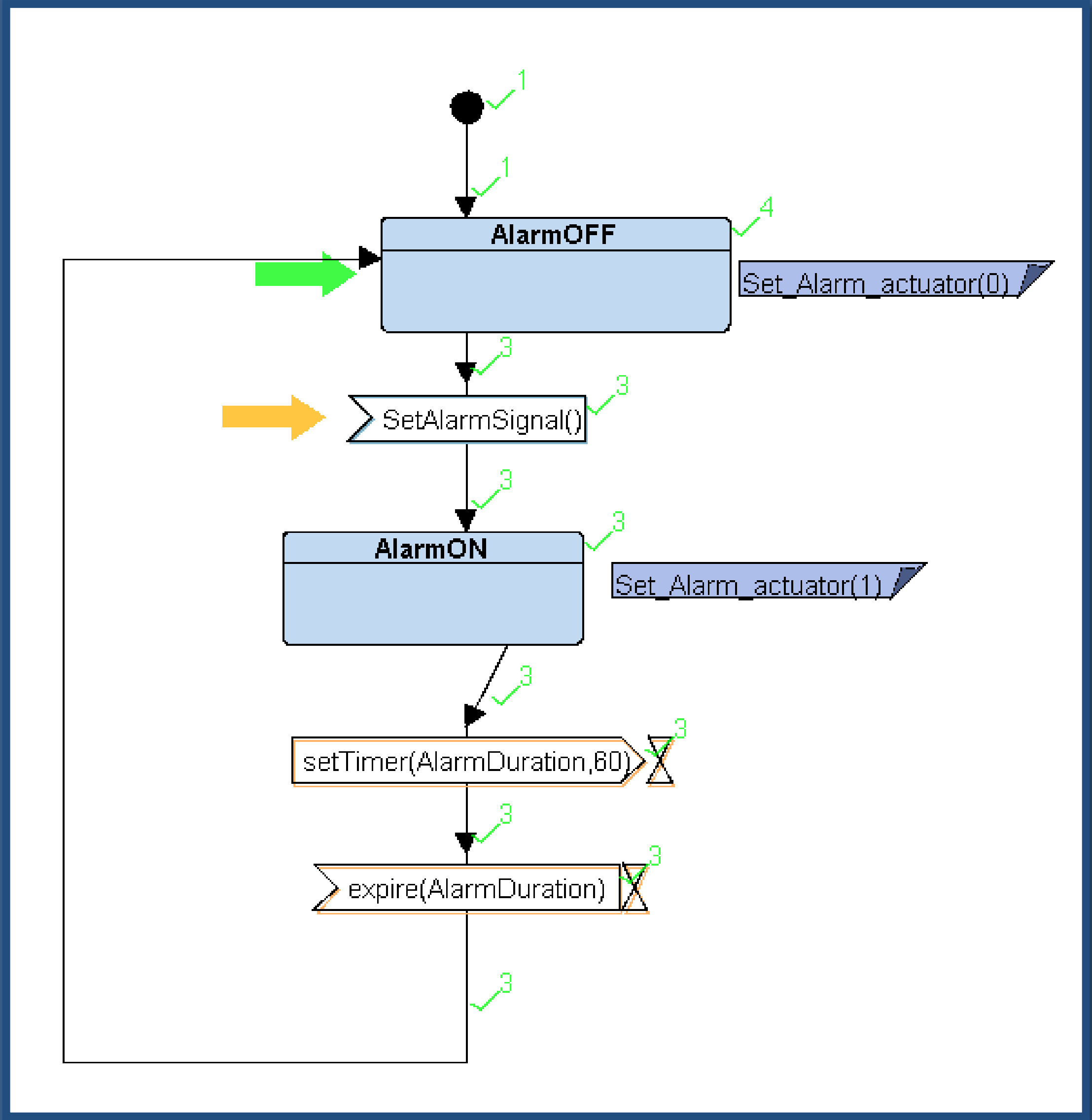
# 3.6.SYSTEM DESIGN

## ➢ 3.6.2.HIGH PRESSURE DETECTION STATE-MACHINE


Pressure = RANDOM0[12, 18]
Pressure = getPressureVal()
[ Pressure >= PressureThreshold]
[ Pressure < PressureThreshold]
HighPressure
NormalPressure
SetAlarmSignal()
after(60)
Pressure = getPressureVal()
Pressure = RANDOM0[12, 18]
[ Pressure < PressureThreshold]
[ Pressure >= PressureThreshold]
Pressure = RANDOM0[12, 18]
Pressure = getPressureVal()
[ Pressure >= PressureThreshold]
[ Pressure < PressureThreshold]

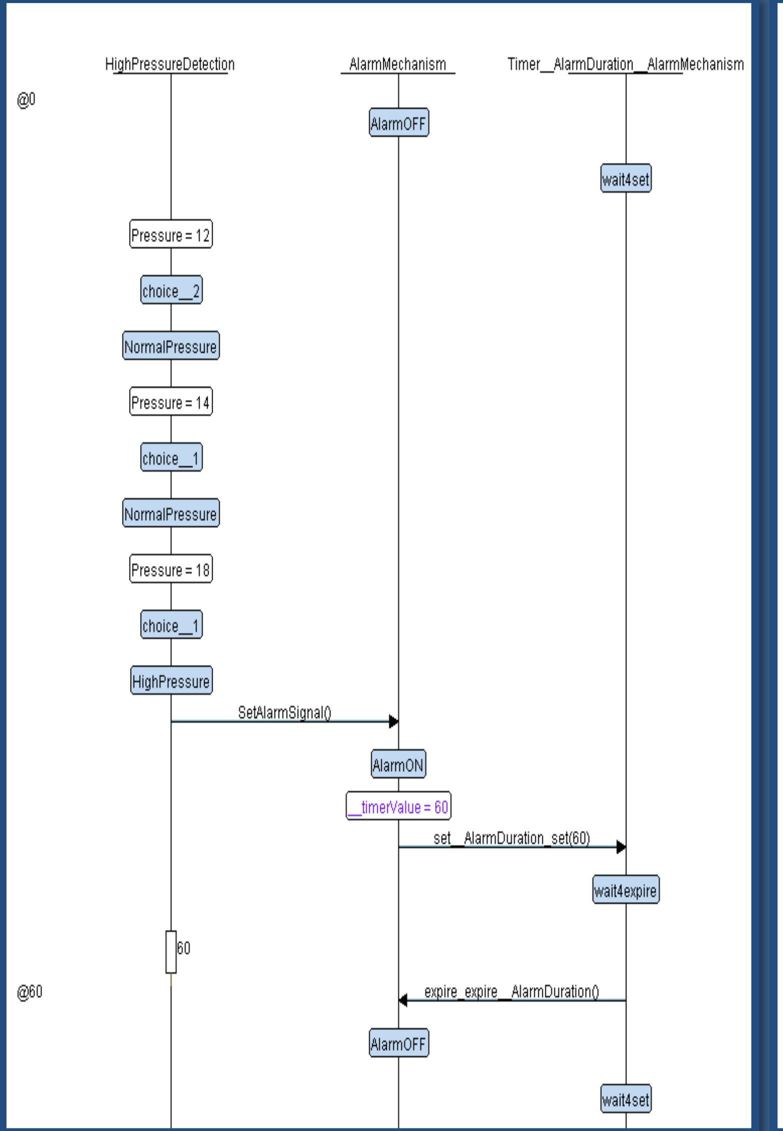
# 3.6.SYSTEM DESIGN

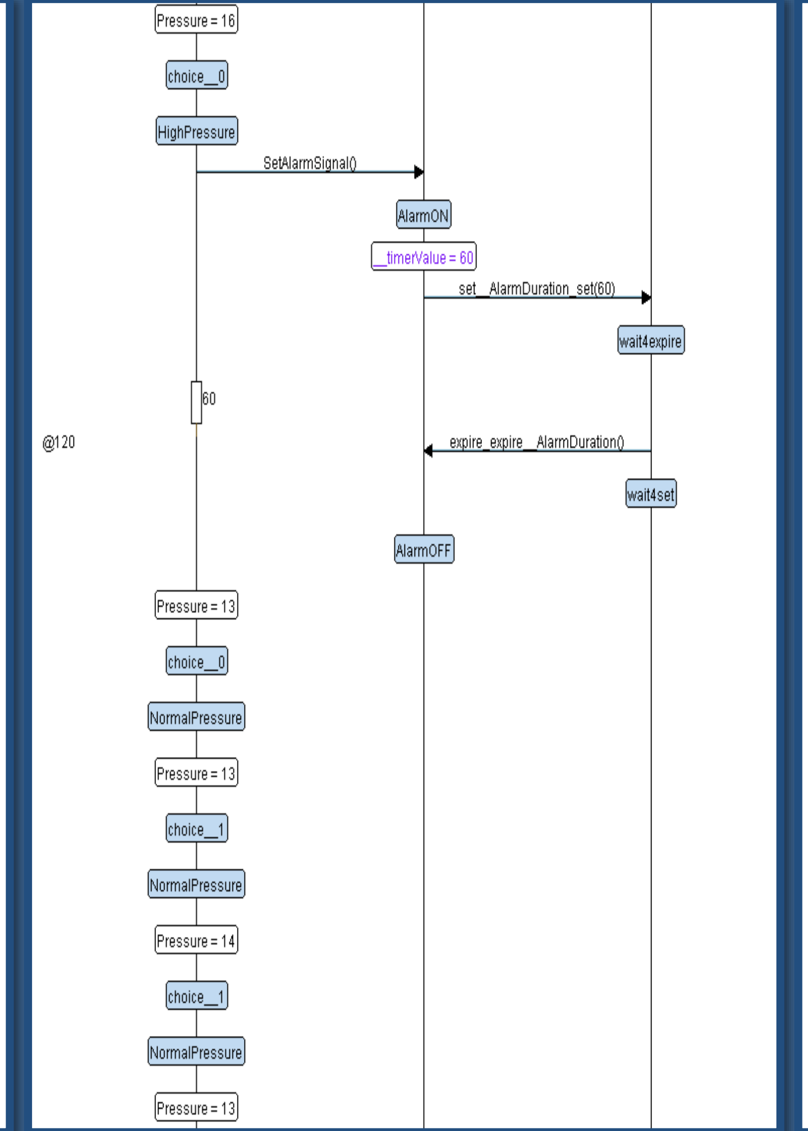
## ➢ 3.6.3.ALARM MECHANISM STATE-MACHINE

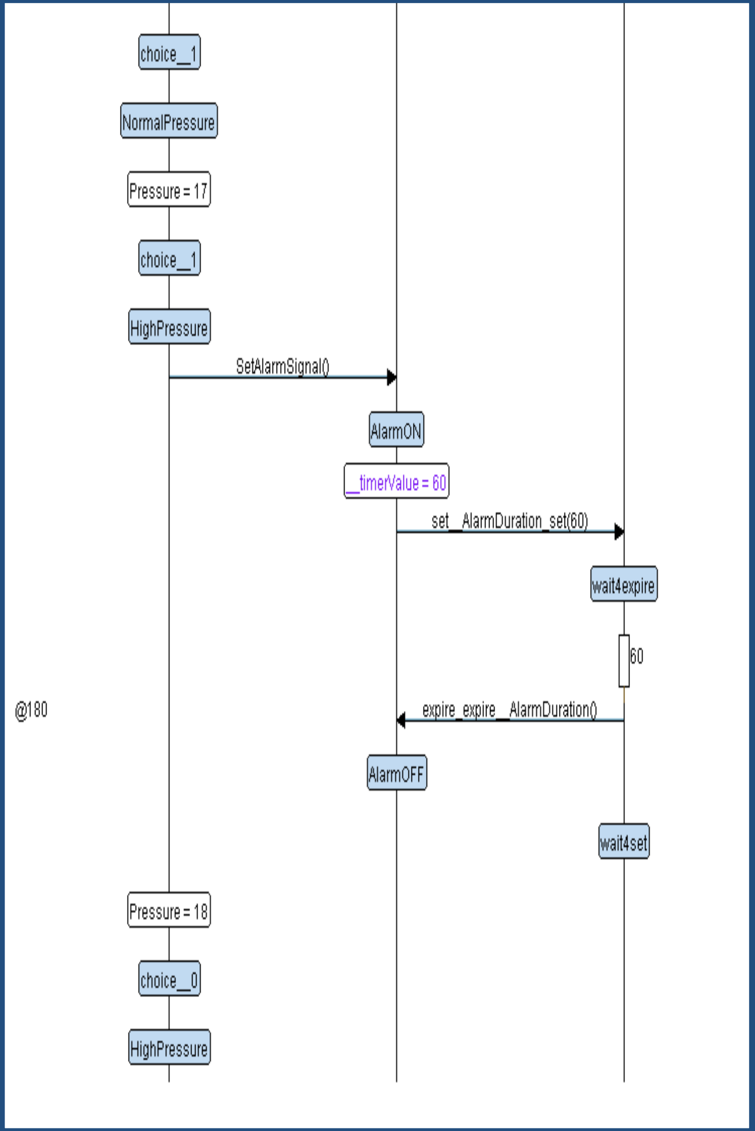


# 3.6.SYSTEM DESIGN

## ➢ 3.6.4.SIMULATION TRACE

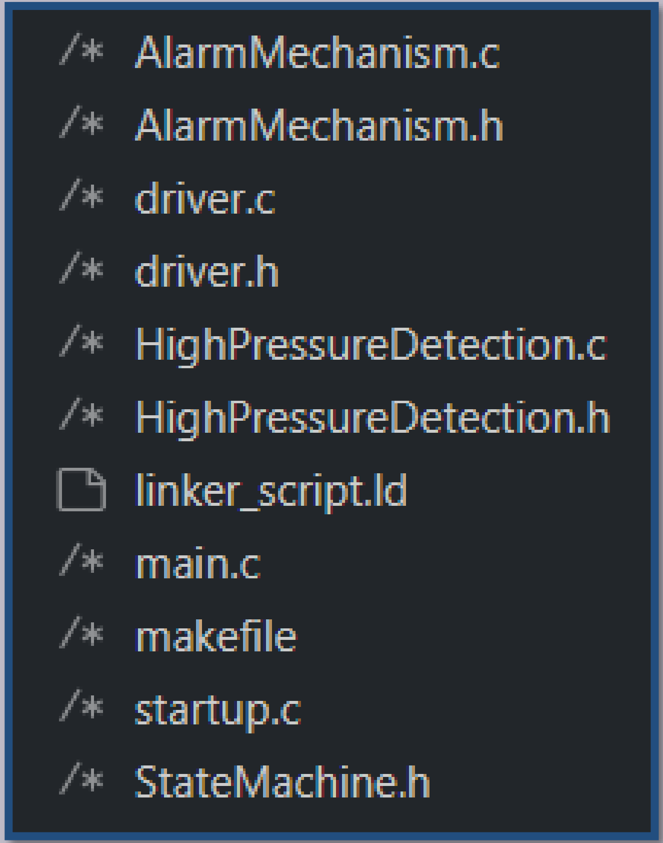






# 4.IMPLEMENTATION

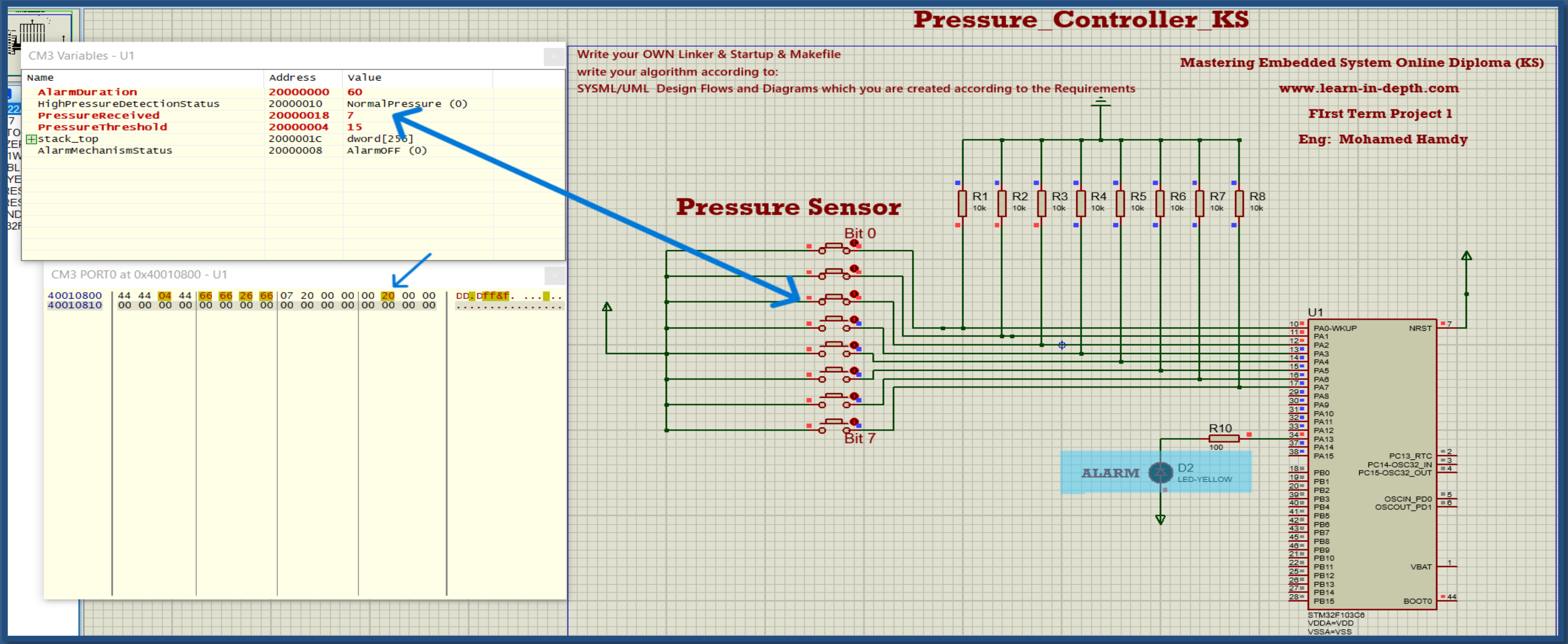
The code is available on GitHub Repo: https://github.com/Mohamed-Hamdy-MA/Mastering_Embedded_System.git



AlarmMechanism.c
AlarmMechanism.h
driver.c
driver.h
HighPressureDetection.c
HighPressureDetection.h
linker_script.ld
main.c
makefile
startup.c
StateMachine.h

# 5.SIMULATION

## ➢ NORMAL PRESSURE DETECTED



CM3 Variables - U1

```
Name                              Address    Value
  AlarmDuration                   20000000   60
  HighPressureDetectionStatus     20000010   NormalPressure (0)
  PressureReceived                20000018   7
  PressureThreshold               20000004   15
⊞stack_top                        2000001C   dword[256]
  AlarmMechanismStatus            20000008   AlarmOFF (0)
```

CM3 PORT0 at 0x40010800 - U1

```
40010800 | 44 44 04 44 | 66 66 26 66 | 07 20 00 00 | 00 20 00 00 | DD.Dff&f. .....
40010810 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | ................
```

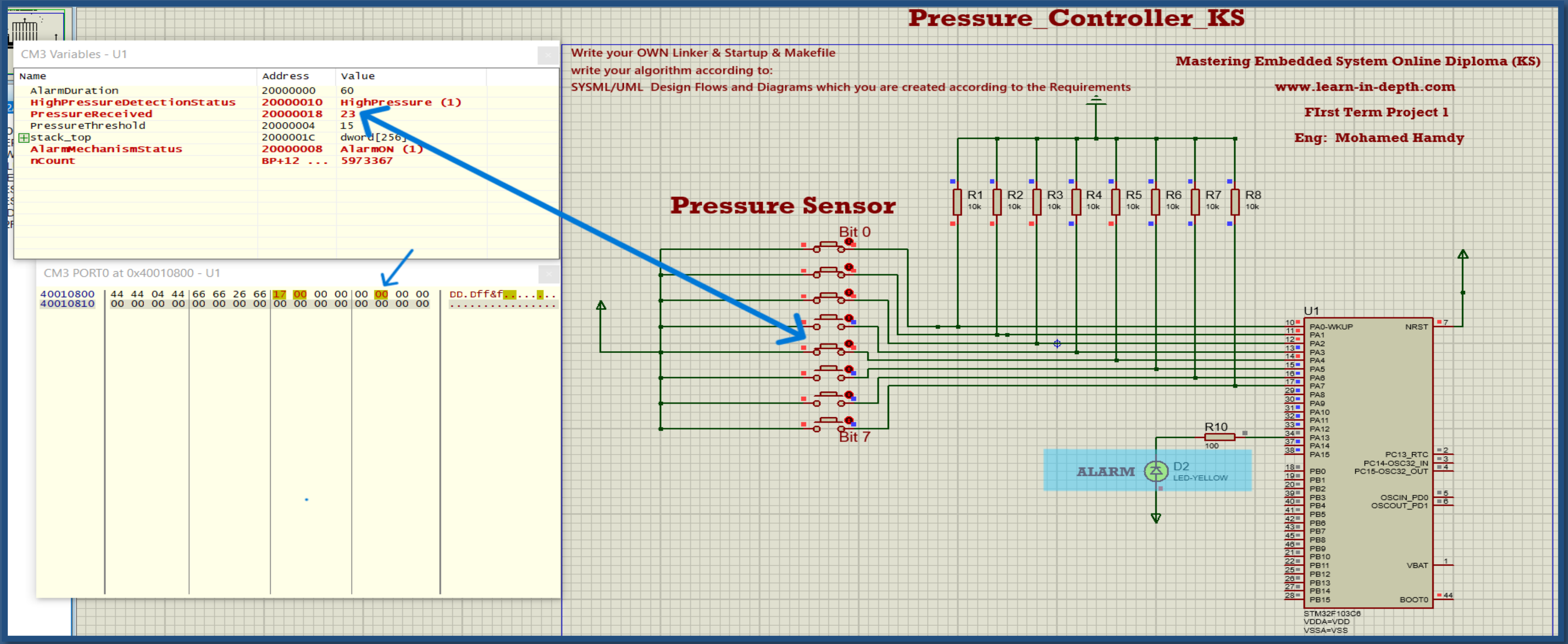
Pressure_Controller_KS

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: Mohamed Hamdy

# 5.SIMULATION

## ➢ HIGH PRESSURE DETECTED



CM3 Variables - U1

| Name | Address | Value |
|---|---|---|
| AlarmDuration | 20000000 | 60 |
| HighPressureDetectionStatus | 20000010 | HighPressure (1) |
| PressureReceived | 20000018 | 23 |
| PressureThreshold | 20000004 | 15 |
| ⊞stack_top | 2000001C | dword[256] |
| AlarmMechanismStatus | 20000008 | AlarmON (1) |
| nCount | BP+12 ... | 5973367 |

CM3 PORT0 at 0x40010800 - U1

```
40010800 | 44 44 04 44 | 66 66 26 66 | 17 00 00 00 | 00 00 00 00 | DD.Dff&f........
40010810 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | ................
```

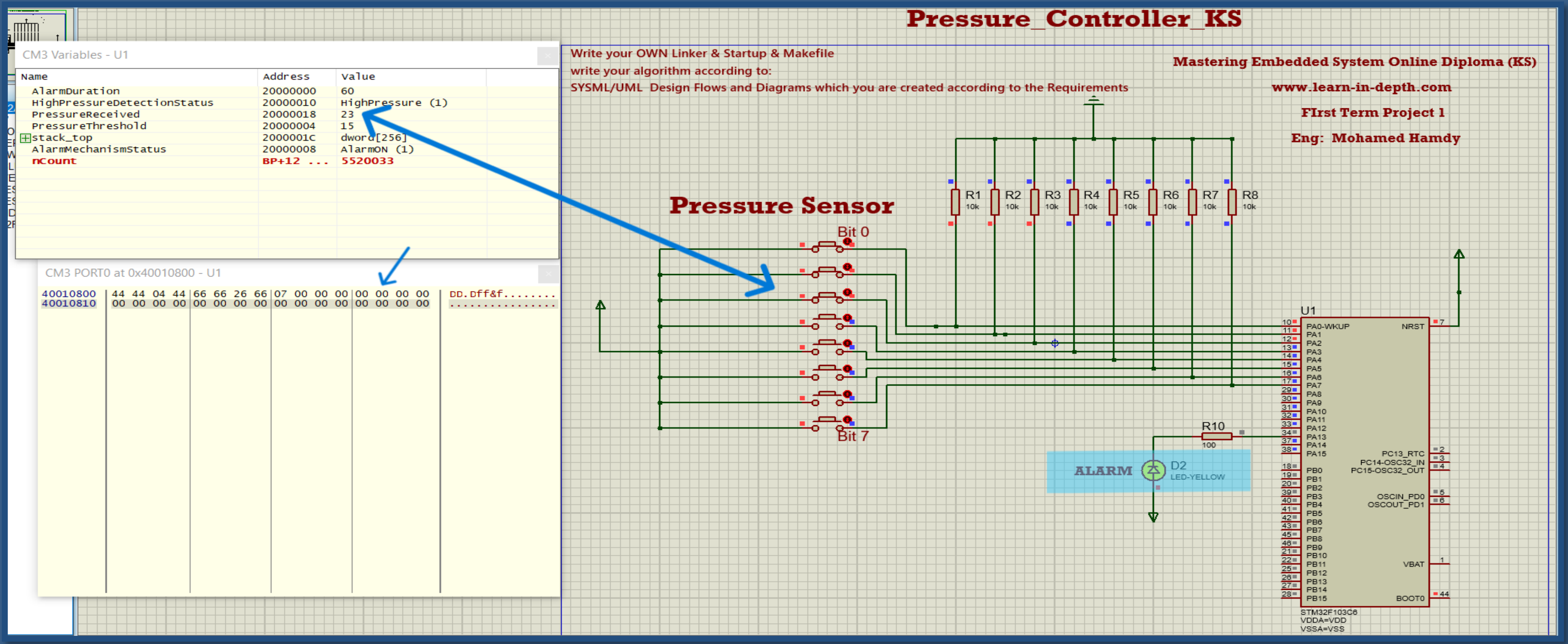
Pressure_Controller_KS

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: Mohamed Hamdy

# 5.SIMULATION

## ➢ ALARM MECHANISM



CM3 Variables - U1

| Name | Address | Value |
|---|---|---|
| AlarmDuration | 20000000 | 60 |
| HighPressureDetectionStatus | 20000010 | HighPressure (1) |
| PressureReceived | 20000018 | 23 |
| PressureThreshold | 20000004 | 15 |
| ⊞stack_top | 2000001C | dword[256] |
| AlarmMechanismStatus | 20000008 | AlarmON (1) |
| nCount | BP+12 ... | 5520033 |

CM3 PORT0 at 0x40010800 - U1

```
40010800 | 44 44 04 44 | 66 66 26 66 | 07 00 00 00 | 00 00 00 00 | DD.Dff&f........
40010810 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | 00 00 00 00 | ................
```

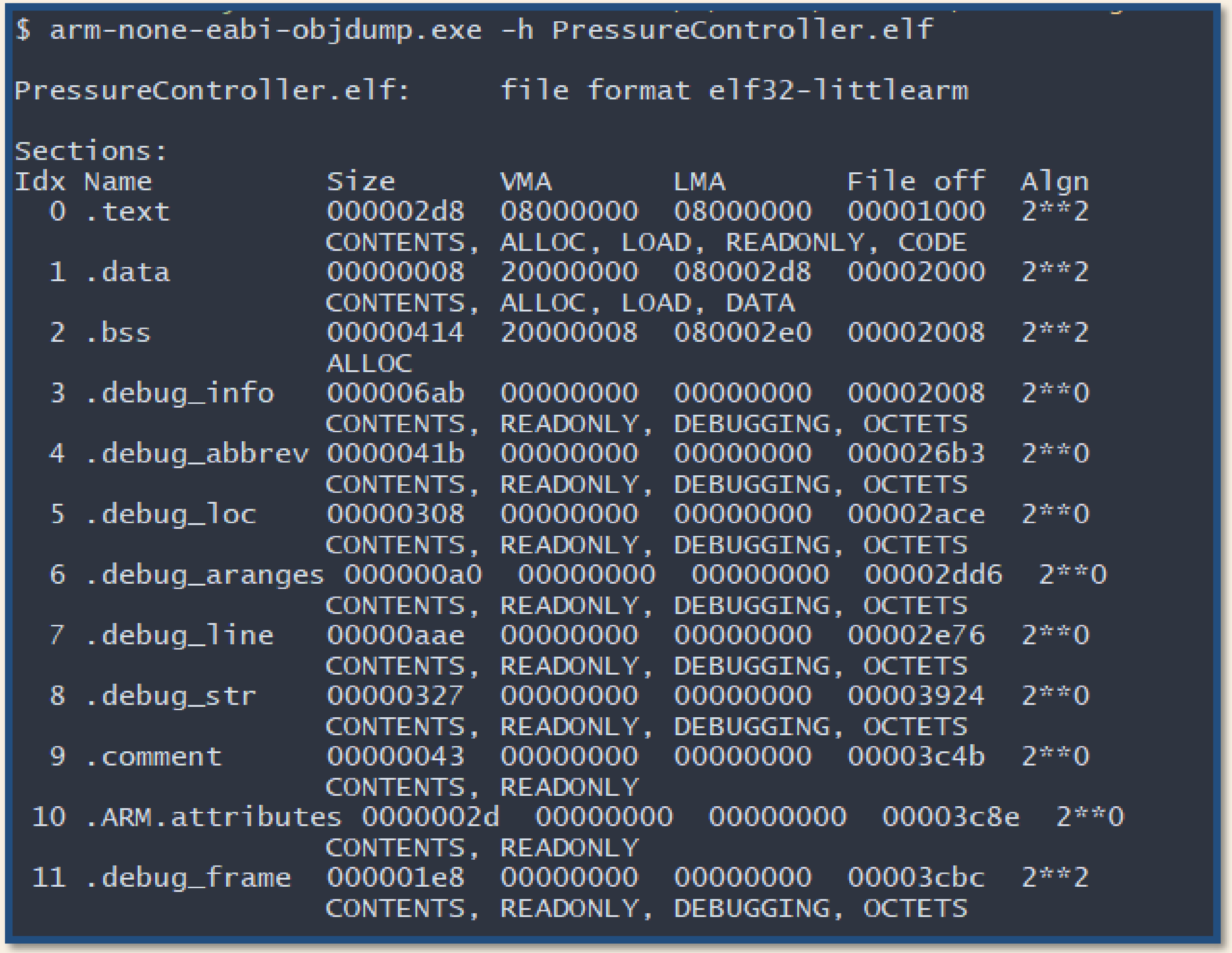
Pressure_Controller_KS

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:  Mohamed Hamdy

# 6.BINARY UTILITIES

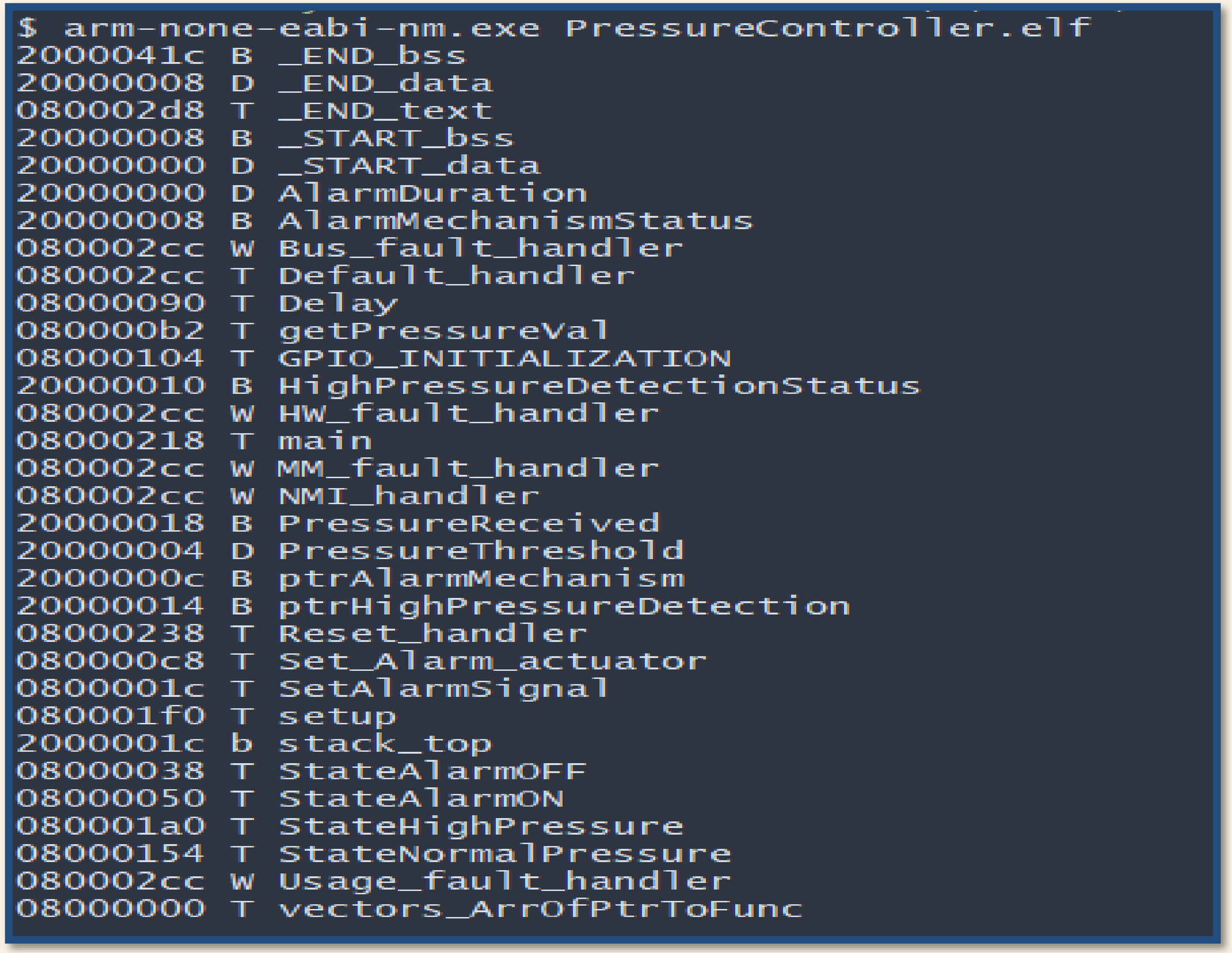
## ➢ 6.1.HEADERS



```
$ arm-none-eabi-objdump.exe -h PressureController.elf

PressureController.elf:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         000002d8  08000000  08000000  00001000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000008  20000000  080002d8  00002000  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000414  20000008  080002e0  00002008  2**2
                  ALLOC
  3 .debug_info   000006ab  00000000  00000000  00002008  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
  4 .debug_abbrev 0000041b  00000000  00000000  000026b3  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
  5 .debug_loc    00000308  00000000  00000000  00002ace  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
  6 .debug_aranges 000000a0  00000000  00000000  00002dd6  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
  7 .debug_line   00000aae  00000000  00000000  00002e76  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
  8 .debug_str    00000327  00000000  00000000  00003924  2**0
                  CONTENTS, READONLY, DEBUGGING, OCTETS
  9 .comment      00000043  00000000  00000000  00003c4b  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 0000002d  00000000  00000000  00003c8e  2**0
                  CONTENTS, READONLY
 11 .debug_frame  000001e8  00000000  00000000  00003cbc  2**2
                  CONTENTS, READONLY, DEBUGGING, OCTETS
```

# 6.BINARY UTILITIES

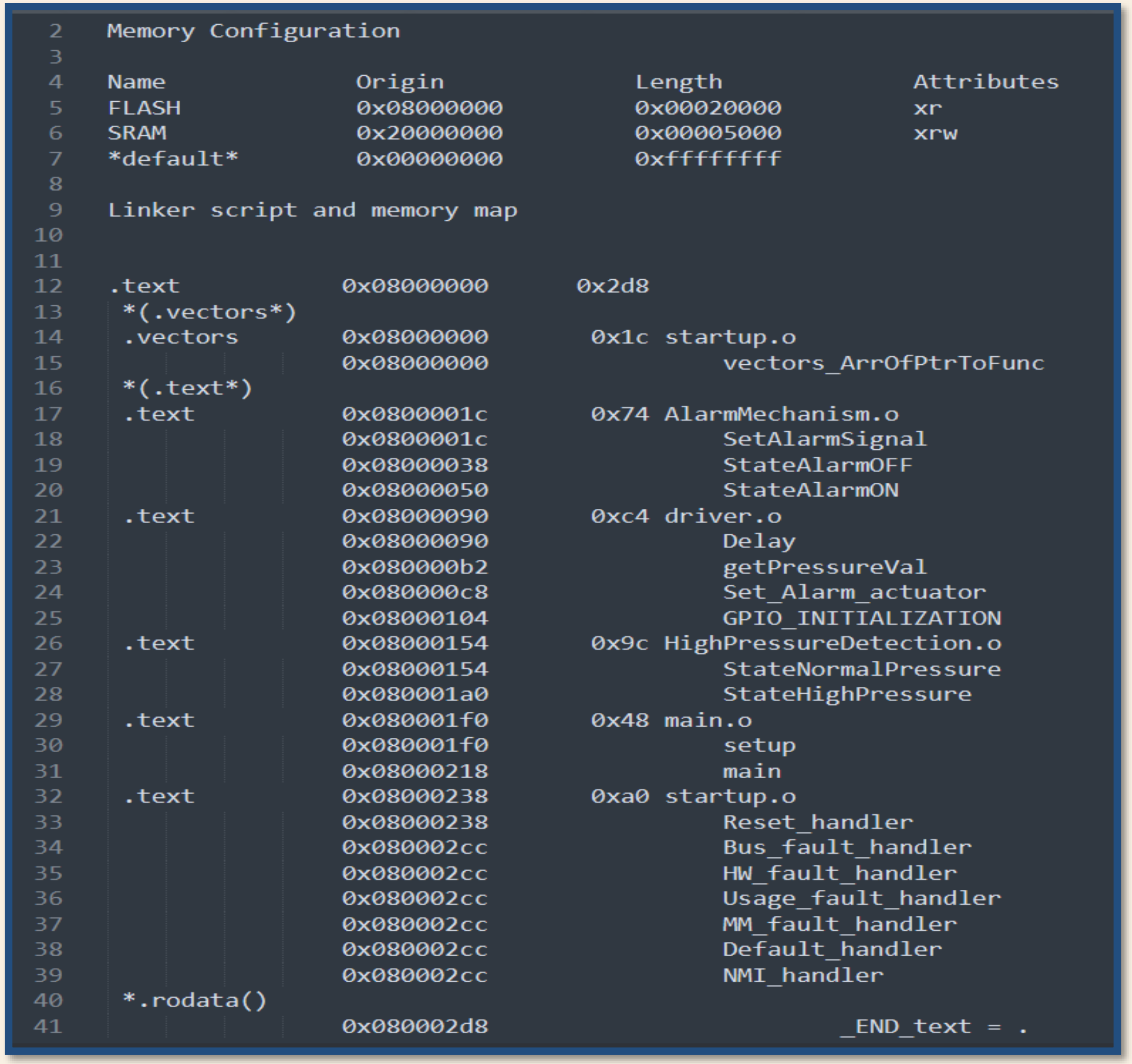
## ➢ 6.2.SYMBOLS



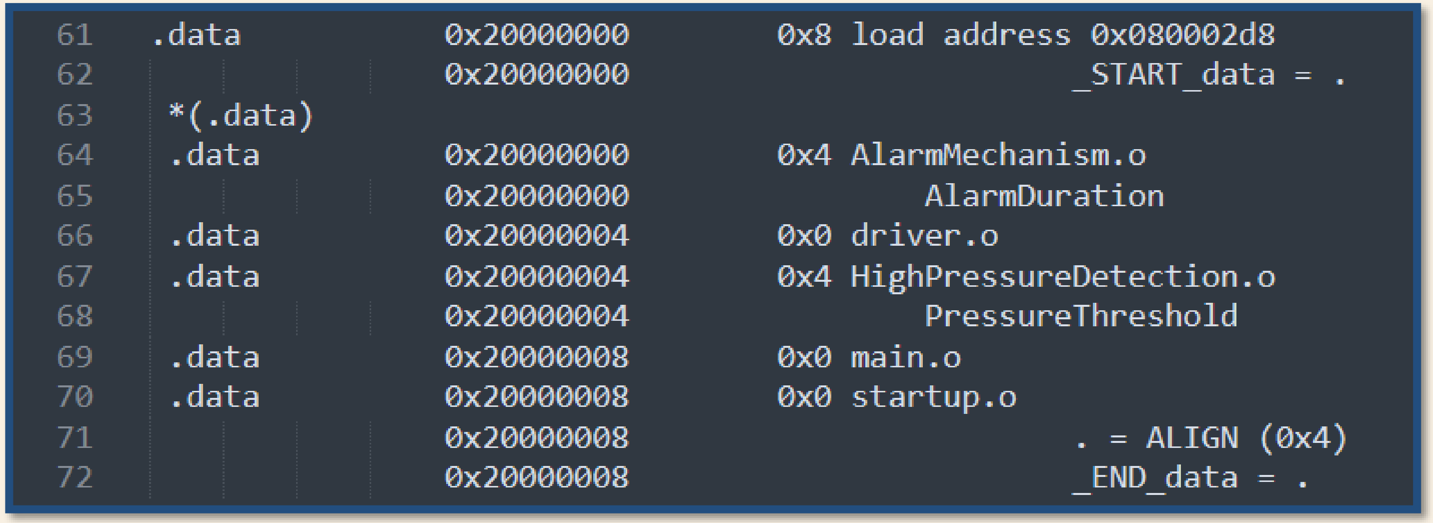
```
$ arm-none-eabi-nm.exe PressureController.elf
2000041c B _END_bss
20000008 D _END_data
080002d8 T _END_text
20000008 B _START_bss
20000000 D _START_data
20000000 D AlarmDuration
20000008 B AlarmMechanismStatus
080002cc W Bus_fault_handler
080002cc T Default_handler
08000090 T Delay
080000b2 T getPressureVal
08000104 T GPIO_INITIALIZATION
20000010 B HighPressureDetectionStatus
080002cc W HW_fault_handler
08000218 T main
080002cc W MM_fault_handler
080002cc W NMI_handler
20000018 B PressureReceived
20000004 D PressureThreshold
2000000c B ptrAlarmMechanism
20000014 B ptrHighPressureDetection
08000238 T Reset_handler
080000c8 T Set_Alarm_actuator
0800001c T SetAlarmSignal
080001f0 T setup
2000001c b stack_top
08000038 T StateAlarmOFF
08000050 T StateAlarmON
080001a0 T StateHighPressure
08000154 T StateNormalPressure
080002cc W Usage_fault_handler
08000000 T vectors_ArrOfPtrToFunc
```

# 6.BINARY UTILITIES

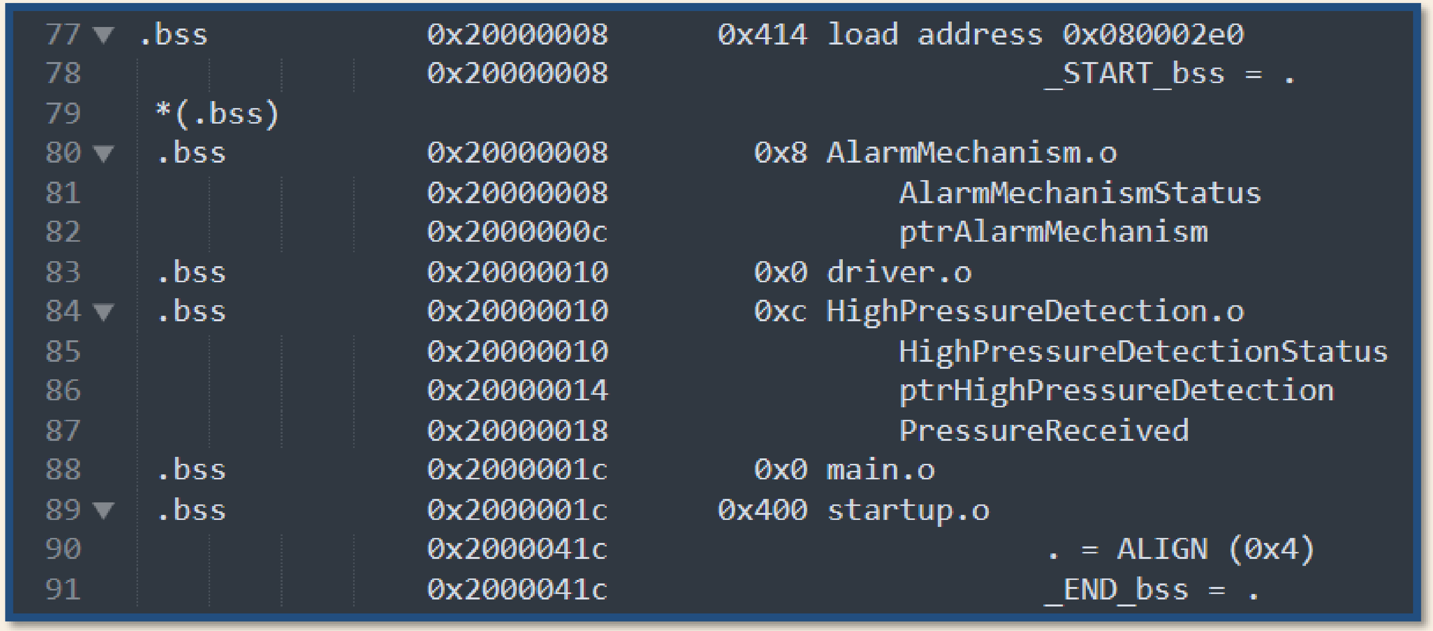
## ➢ 6.3.MAP FILE



```
Memory Configuration

Name             Origin             Length             Attributes
FLASH            0x08000000         0x00020000         xr
SRAM             0x20000000         0x00005000         xrw
*default*        0x00000000         0xffffffff

Linker script and memory map


.text           0x08000000      0x2d8
 *(.vectors*)
 .vectors       0x08000000       0x1c startup.o
                0x08000000                vectors_ArrOfPtrToFunc
 *(.text*)
 .text          0x0800001c       0x74 AlarmMechanism.o
                0x0800001c                SetAlarmSignal
                0x08000038                StateAlarmOFF
                0x08000050                StateAlarmON
 .text          0x08000090       0xc4 driver.o
                0x08000090                Delay
                0x080000b2                getPressureVal
                0x080000c8                Set_Alarm_actuator
                0x08000104                GPIO_INITIALIZATION
 .text          0x08000154       0x9c HighPressureDetection.o
                0x08000154                StateNormalPressure
                0x080001a0                StateHighPressure
 .text          0x080001f0       0x48 main.o
                0x080001f0                setup
                0x08000218                main
 .text          0x08000238       0xa0 startup.o
                0x08000238                Reset_handler
                0x080002cc                Bus_fault_handler
                0x080002cc                HW_fault_handler
                0x080002cc                Usage_fault_handler
                0x080002cc                MM_fault_handler
                0x080002cc                Default_handler
                0x080002cc                NMI_handler
 *.rodata()
                0x080002d8                        _END_text = .
```



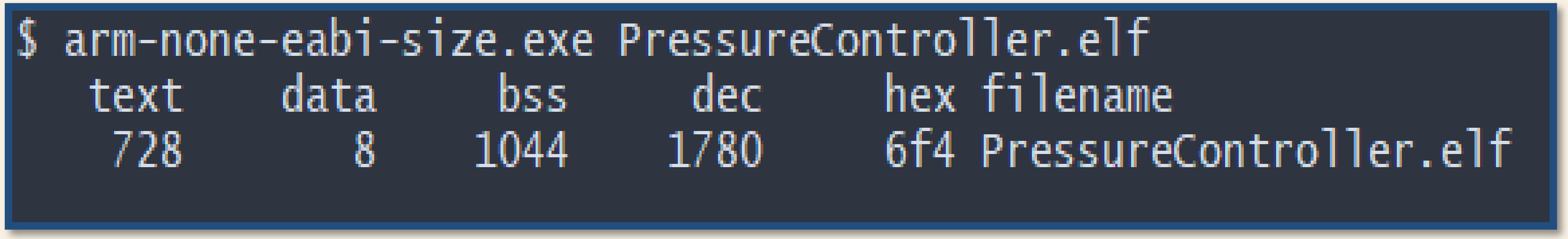
```
.data           0x20000000         0x8 load address 0x080002d8
                0x20000000                        _START_data = .
 *(.data)
 .data          0x20000000         0x4 AlarmMechanism.o
                0x20000000                 AlarmDuration
 .data          0x20000004         0x0 driver.o
 .data          0x20000004         0x4 HighPressureDetection.o
                0x20000004                 PressureThreshold
 .data          0x20000008         0x0 main.o
 .data          0x20000008         0x0 startup.o
                0x20000008                        . = ALIGN (0x4)
                0x20000008                        _END_data = .
```



```
.bss            0x20000008       0x414 load address 0x080002e0
                0x20000008                        _START_bss = .
 *(.bss)
 .bss           0x20000008         0x8 AlarmMechanism.o
                0x20000008                 AlarmMechanismStatus
                0x2000000c                 ptrAlarmMechanism
 .bss           0x20000010         0x0 driver.o
 .bss           0x20000010         0xc HighPressureDetection.o
                0x20000010                 HighPressureDetectionStatus
                0x20000014                 ptrHighPressureDetection
                0x20000018                 PressureReceived
 .bss           0x2000001c         0x0 main.o
 .bss           0x2000001c       0x400 startup.o
                0x2000041c                        . = ALIGN (0x4)
                0x2000041c                        _END_bss = .
```

# 6.BINARY UTILITIES

## ➢ 6.4.SIZE



```
$ arm-none-eabi-size.exe PressureController.elf
   text    data     bss     dec     hex filename
    728       8    1044    1780     6f4 PressureController.elf
```

# THANK YOU

Mohamed Hamdy

Mohamedhamdy.m.h.m.a@gmail.com