SIMULATION & IMPLEMENTATION TOOLS

This Project is Firstly Designed and verified using "TTool" then implemented using simple "Editor"

Build & Compiled → ARM Cross Tool-Chain

Debugger → ARM GDB Debugger

MASTERING EMBEDDED SYSTEM 1ST TERM PROJECT

BY

Omar Ahmed Hassan

CONTACT ME

Phone| (+20) 1141563470 |
omarsaudiy@gmail.com
github.com/OmarSaudiy
linkedin.com/in/omarsaudiy

PROJECT NAME

PRESSURE DETECTION

An embedded system Pressure detection project involves designing and implementing a system that can detect objects in-front and stop vehicle accordingly.

SYSTEM ARCHITECTURE

- ## CASE STUDY

The system detects if the pressure exceeds 20bars in the cabin, then fire alarm to inform the crew for 60 sec.

Assumption: power, sensors and actuators never fail.

Optionally: Log the pressure values in ROM.

- ## METHODOLOGY

Water-fall Model for 2 reasons

1. I am both Customer and Developer, that’s grantee there is not any misunderstanding between the end-user and the developer.
2. The Water-fall Model suffers from teams’ dependency and blocking; However, I am working alone and that means whatever Model I choose there will be blocking anyway.

- ## REQUIREMENTS

The Requirements Diagram is identified next page.

- ## SPACE-EXPLORATION/ PARTITIONING

We will use blue-bill Target (STM32F103C6) Processor ARM Cortex-M3 for this project.

- ## SYSTEM ANALYSIS

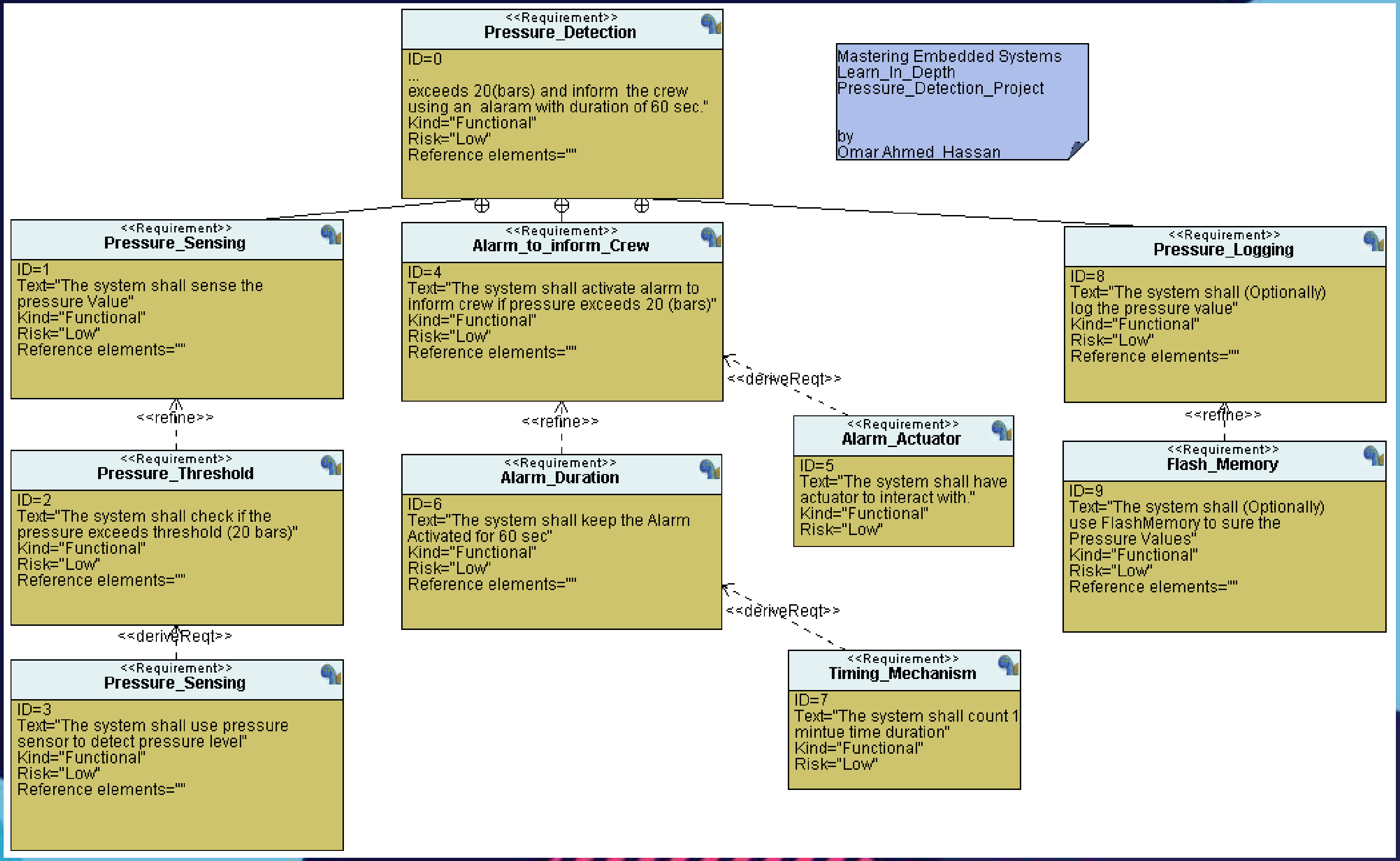
The system analysis Diagrams are identified in the following pages.

- ## SYSTEM DESIGN

The system Design Diagrams are identified in the following pages.

> **“Learn in depth and you will be deeply happy.”**
>
> \- Omar Ahmed -

REQUIREMENTS DIAGRAM



<<Requirement>>
Pressure_Detection
ID=0
...
exceeds 20(bars) and inform the crew
using an alaram with duration of 60 sec."
Kind="Functional"
Risk="Low"
Reference elements=""

Mastering Embedded Systems
Learn_In_Depth
Pressure_Detection_Project

by
Omar Ahmed Hassan

<<Requirement>>
Pressure_Sensing
ID=1
Text="The system shall sense the
pressure Value"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Alarm_to_inform_Crew
ID=4
Text="The system shall activate alarm to
inform crew if pressure exceeds 20 (bars)"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Pressure_Logging
ID=8
Text="The system shall (Optionally)
log the pressure value"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Pressure_Threshold
ID=2
Text="The system shall check if the
pressure exceeds threshold (20 bars)"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Alarm_Duration
ID=6
Text="The system shall keep the Alarm
Activated for 60 sec"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Alarm_Actuator
ID=5
Text="The system shall have
actuator to interact with."
Kind="Functional"
Risk="Low"

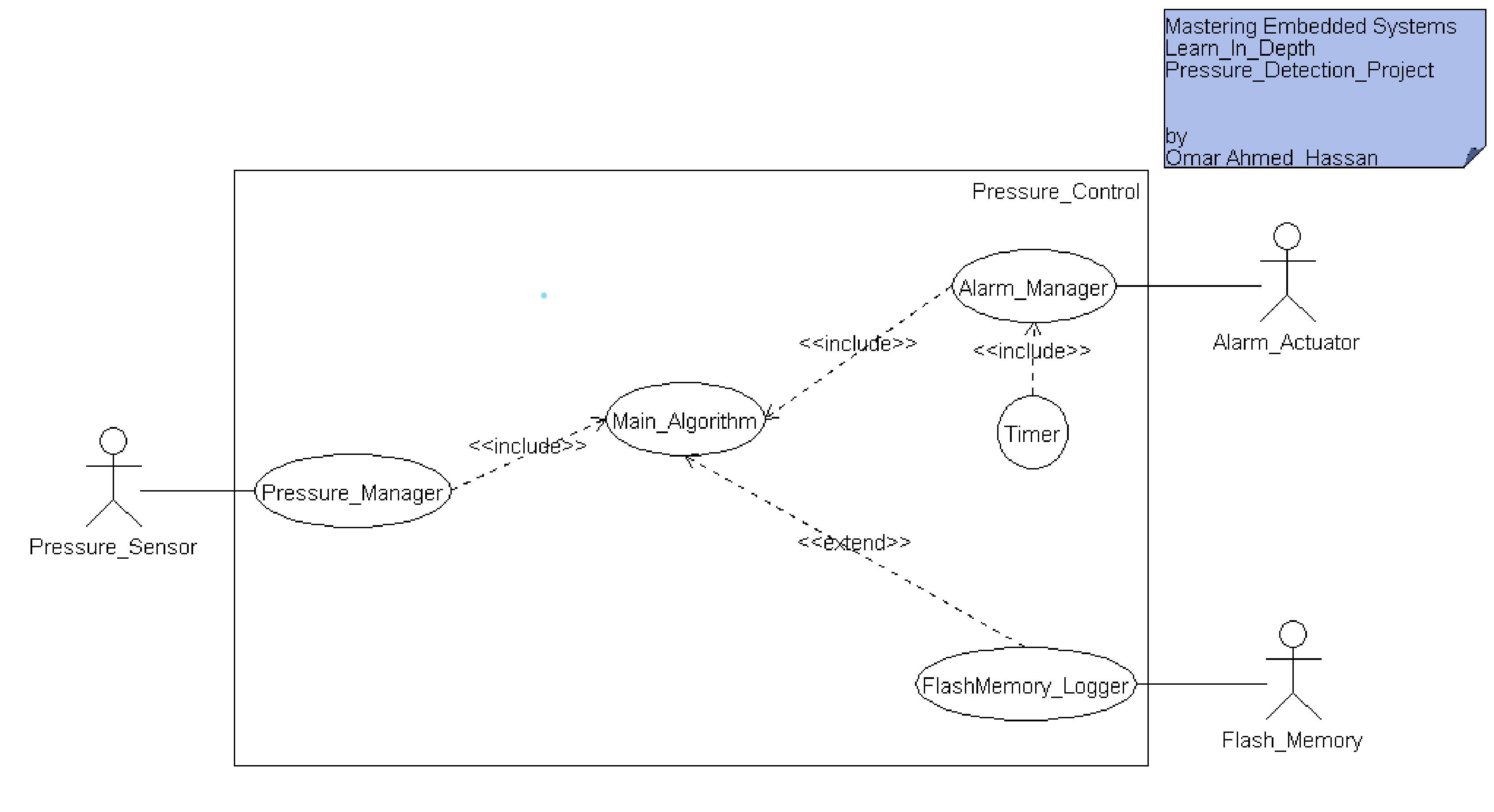
<<Requirement>>
Flash_Memory
ID=9
Text="The system shall (Optionally)
use FlashMemory to sure the
Pressure Values"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Pressure_Sensing
ID=3
Text="The system shall use pressure
sensor to detect pressure level"
Kind="Functional"
Risk="Low"
Reference elements=""

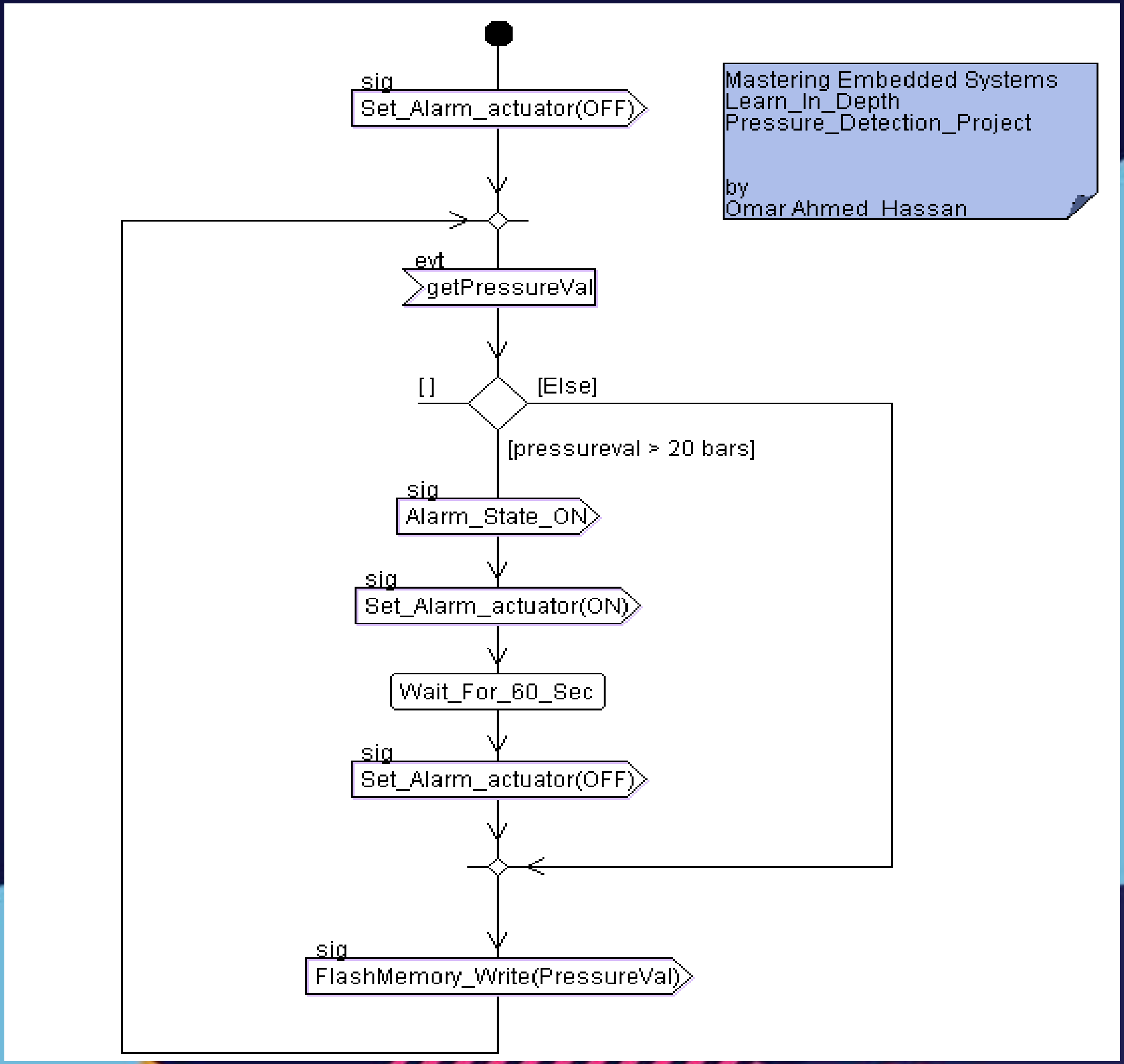
<<Requirement>>
Timing_Mechanism
ID=7
Text="The system shall count 1
mintue time duration"
Kind="Functional"
Risk="Low"

SYSTEM ANALYSIS

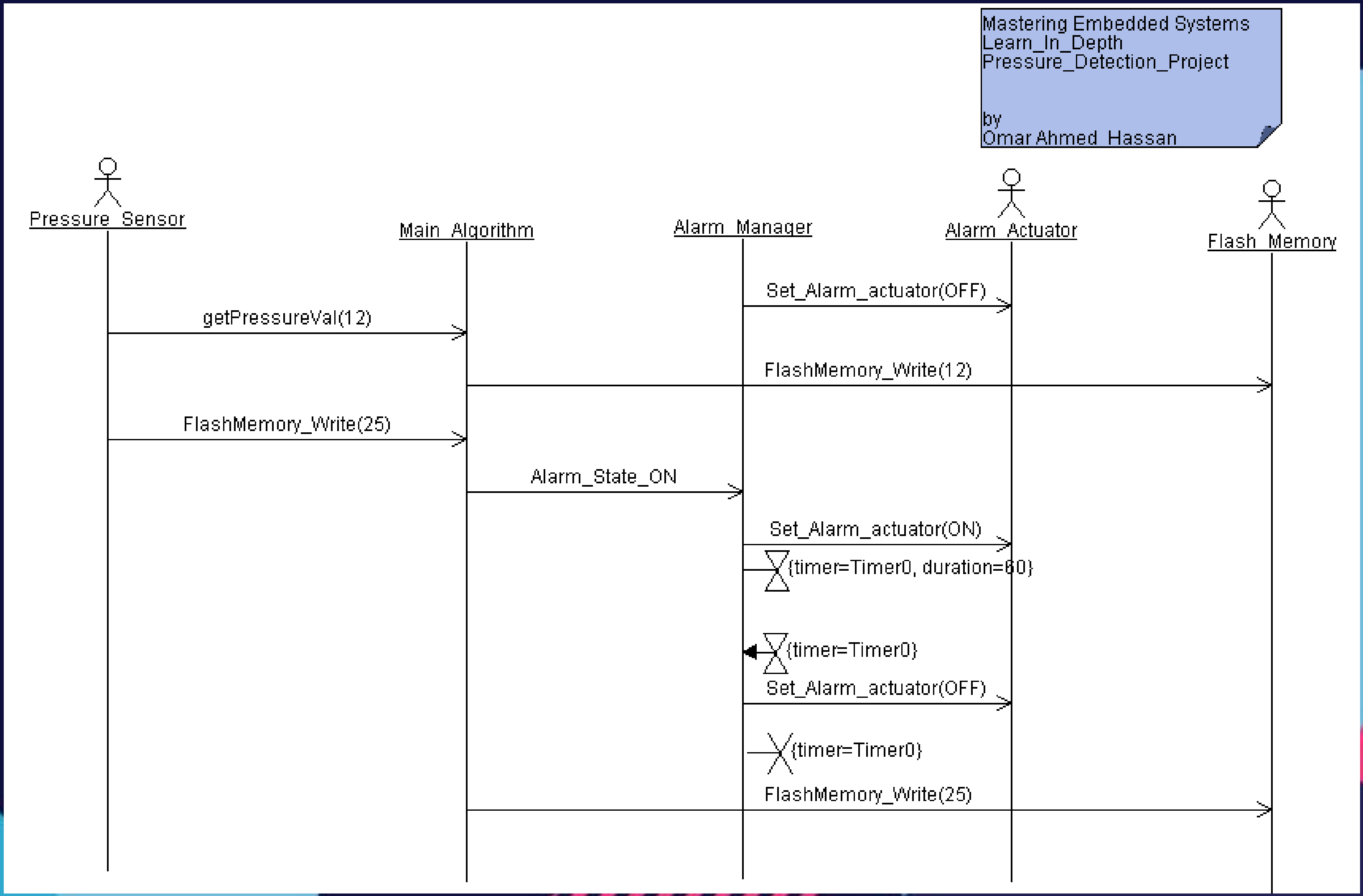
- USE CASE DIAGRAM



- ACTIVITY DIAGRAM

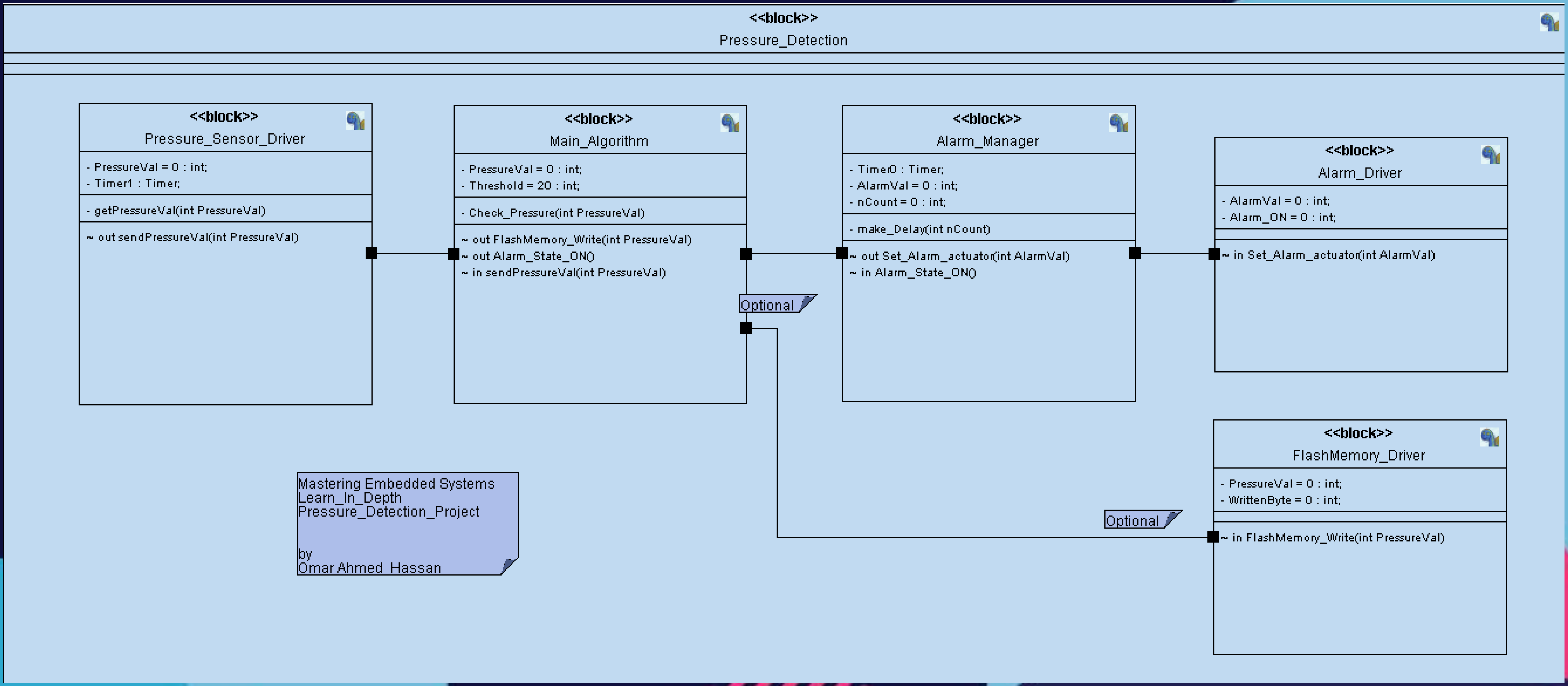


- SEQUENCE DIAGRAM



SYSTEM DESIGN

- BLOCK DIAGRAM



<<block>>
Pressure_Detection

<<block>>
Pressure_Sensor_Driver

- PressureVal = 0 : int;
- Timer1 : Timer;

- getPressureVal(int PressureVal)

~ out sendPressureVal(int PressureVal)

<<block>>
Main_Algorithm

- PressureVal = 0 : int;
- Threshold = 20 : int;

- Check_Pressure(int PressureVal)

~ out FlashMemory_Write(int PressureVal)
~ out Alarm_State_ON()
~ in sendPressureVal(int PressureVal)

<<block>>
Alarm_Manager

- Timer0 : Timer;
- AlarmVal = 0 : int;
- nCount = 0 : int;

- make_Delay(int nCount)

~ out Set_Alarm_actuator(int AlarmVal)
~ in Alarm_State_ON()

<<block>>
Alarm_Driver

- AlarmVal = 0 : int;
- Alarm_ON = 0 : int;

~ in Set_Alarm_actuator(int AlarmVal)

<<block>>
FlashMemory_Driver

- PressureVal = 0 : int;
- WrittenByte = 0 : int;

~ in FlashMemory_Write(int PressureVal)

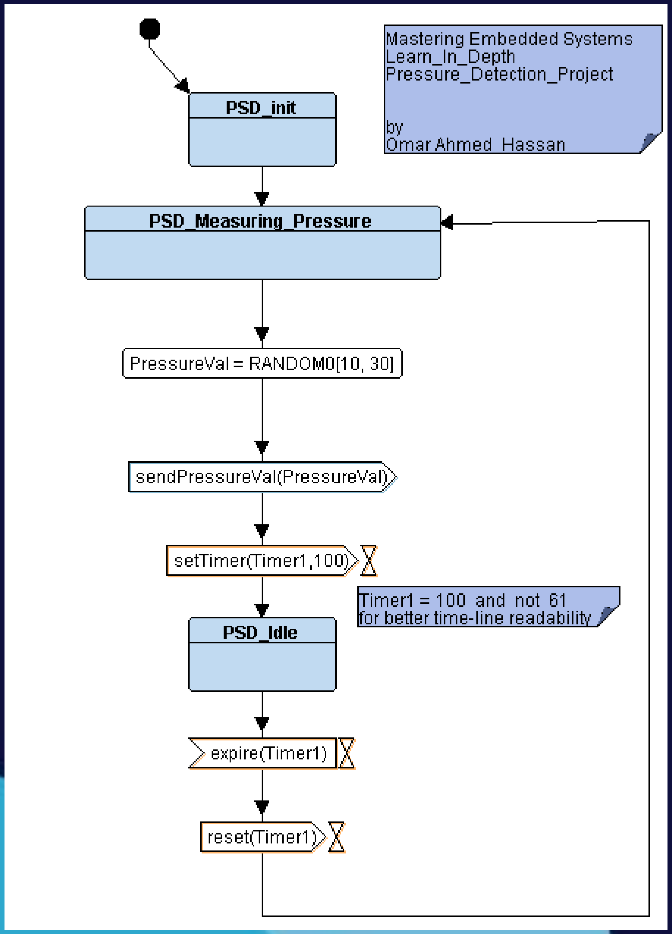
Optional

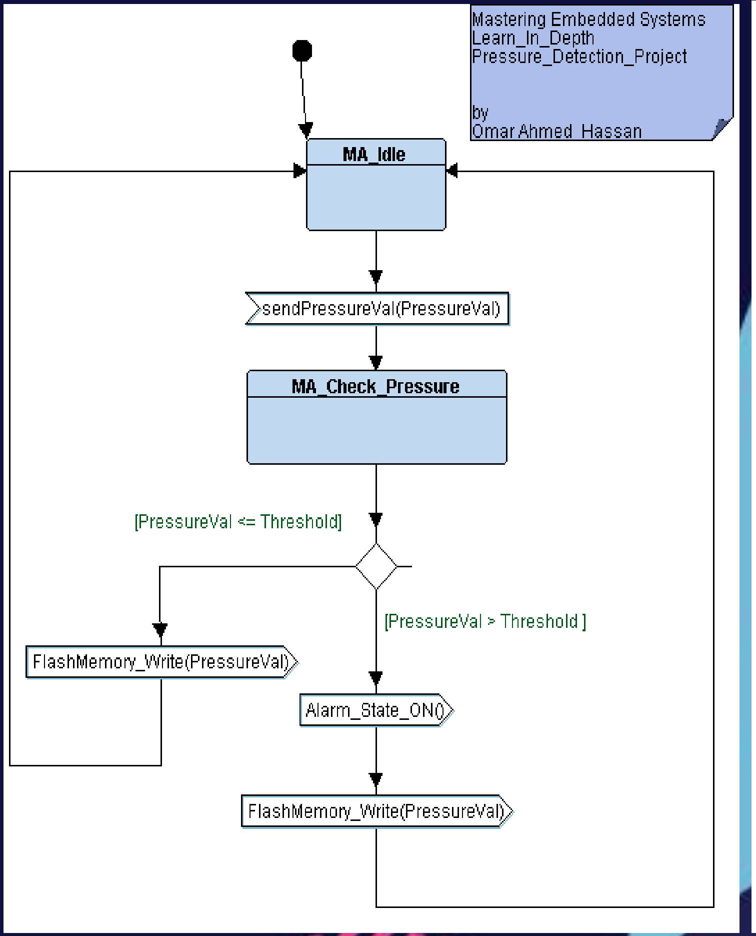
Optional

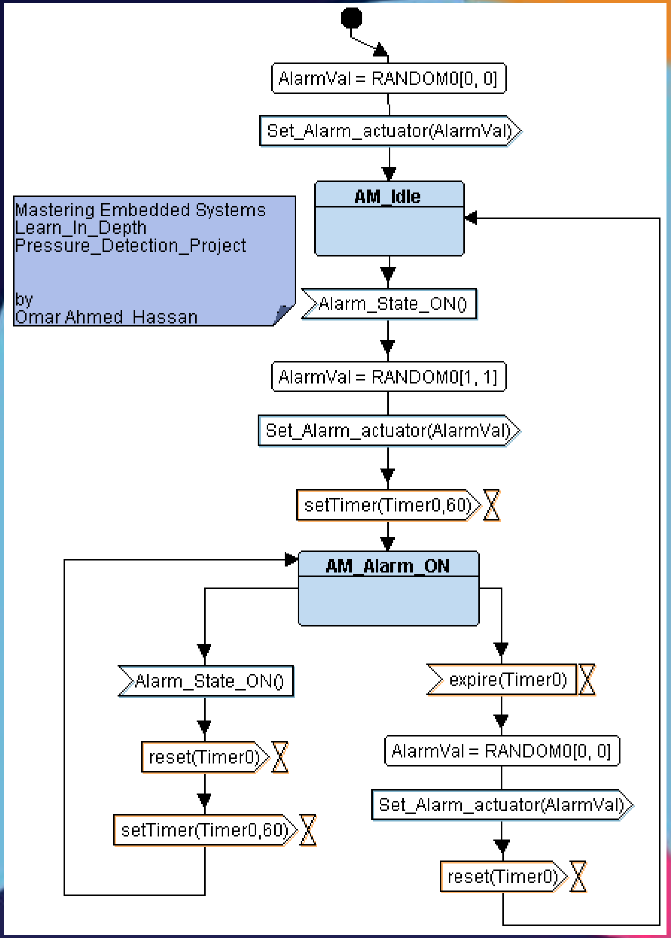
Mastering Embedded Systems
Learn_In_Depth
Pressure_Detection_Project

by
Omar Ahmed Hassan

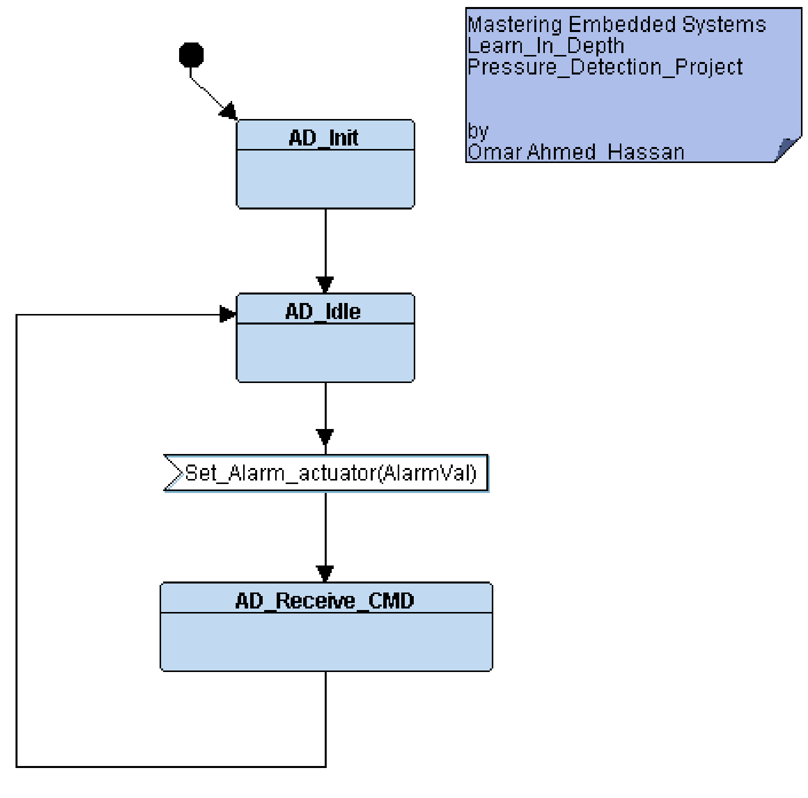
- STATE DIAGRAM OF EACH BLOCK
  - Pressure Sensor Driver
  - Main Algorithm
  - Alarm Manager


Mastering Embedded Systems
Learn_In_Depth
Pressure_Detection_Project
by
Omar Ahmed Hassan
PSD_init
PSD_Measuring_Pressure
PressureVal = RANDOM0[10, 30]
sendPressureVal(PressureVal)
setTimer(Timer1,100)
Timer1 = 100 and not 61
for better time-line readability
PSD_Idle
expire(Timer1)
reset(Timer1)


Mastering Embedded Systems
Learn_In_Depth
Pressure_Detection_Project
by
Omar Ahmed Hassan
MA_Idle
sendPressureVal(PressureVal)
MA_Check_Pressure
[PressureVal <= Threshold]
[PressureVal > Threshold]
FlashMemory_Write(PressureVal)
Alarm_State_ON()
FlashMemory_Write(PressureVal)

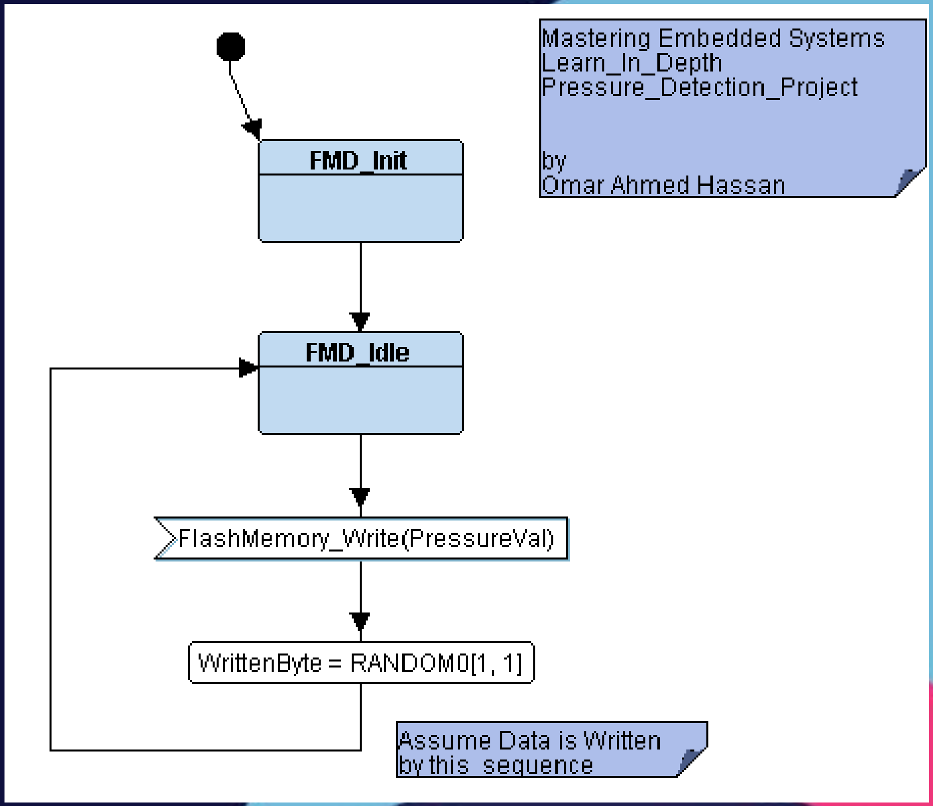


- STATE DIAGRAM OF EACH BLOCK
  - Alarm Driver
  - Flash Memory Driver



Mastering Embedded Systems
Learn_In_Depth
Pressure_Detection_Project

by
Omar Ahmed Hassan



Mastering Embedded Systems
Learn_In_Depth
Pressure_Detection_Project

by
Omar Ahmed Hassan

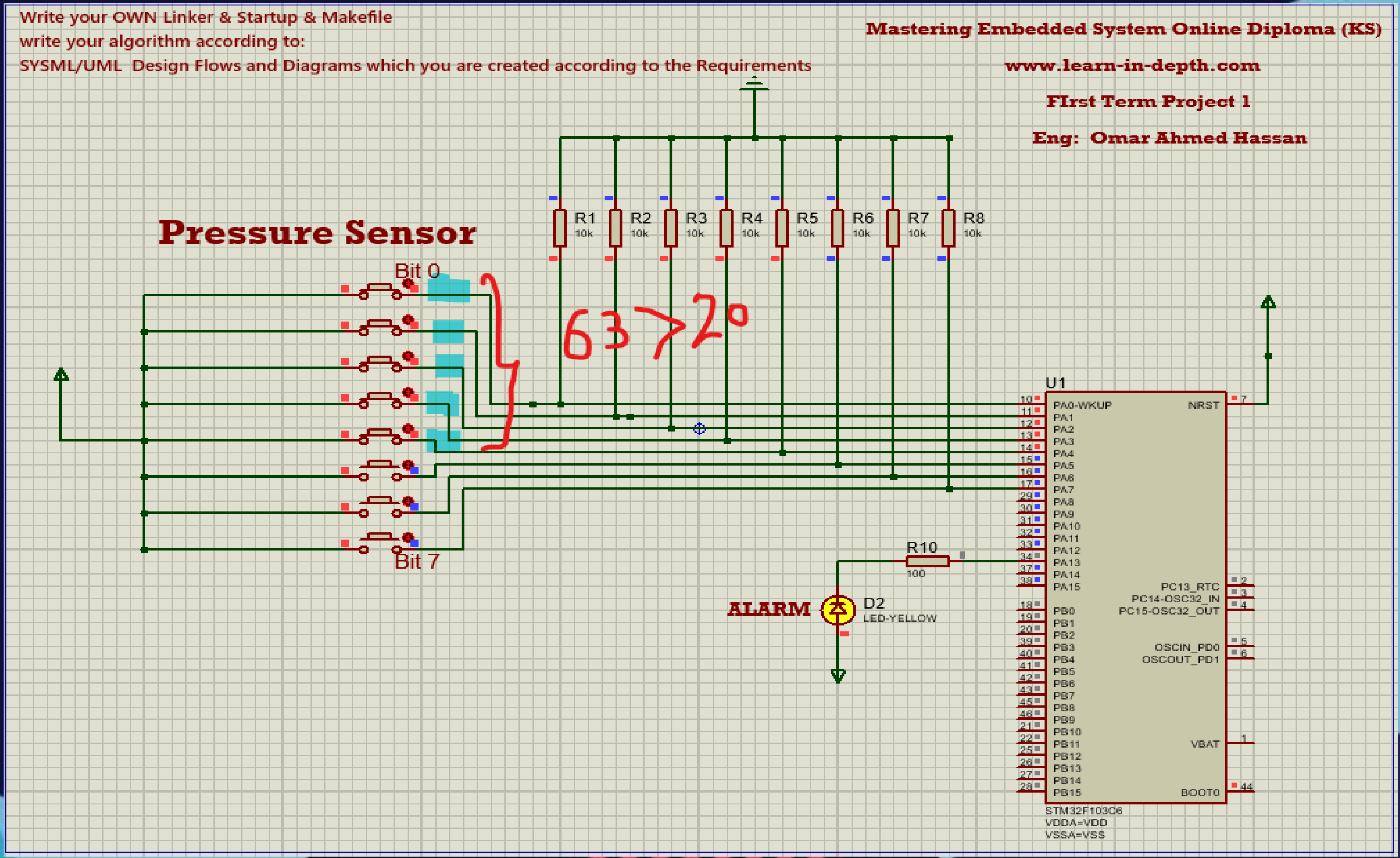
Assume Data is Written
by this sequence

INTERACTIVE SIMULATION

Due to the considerably large size of the figure,
it is not feasible to include it within the confines of this page. However, it can be accessed via Interactive Simulation.

PROTEUS SIMULATION

- POSITIVE TEST (PRESSURE > THRESHOLD)

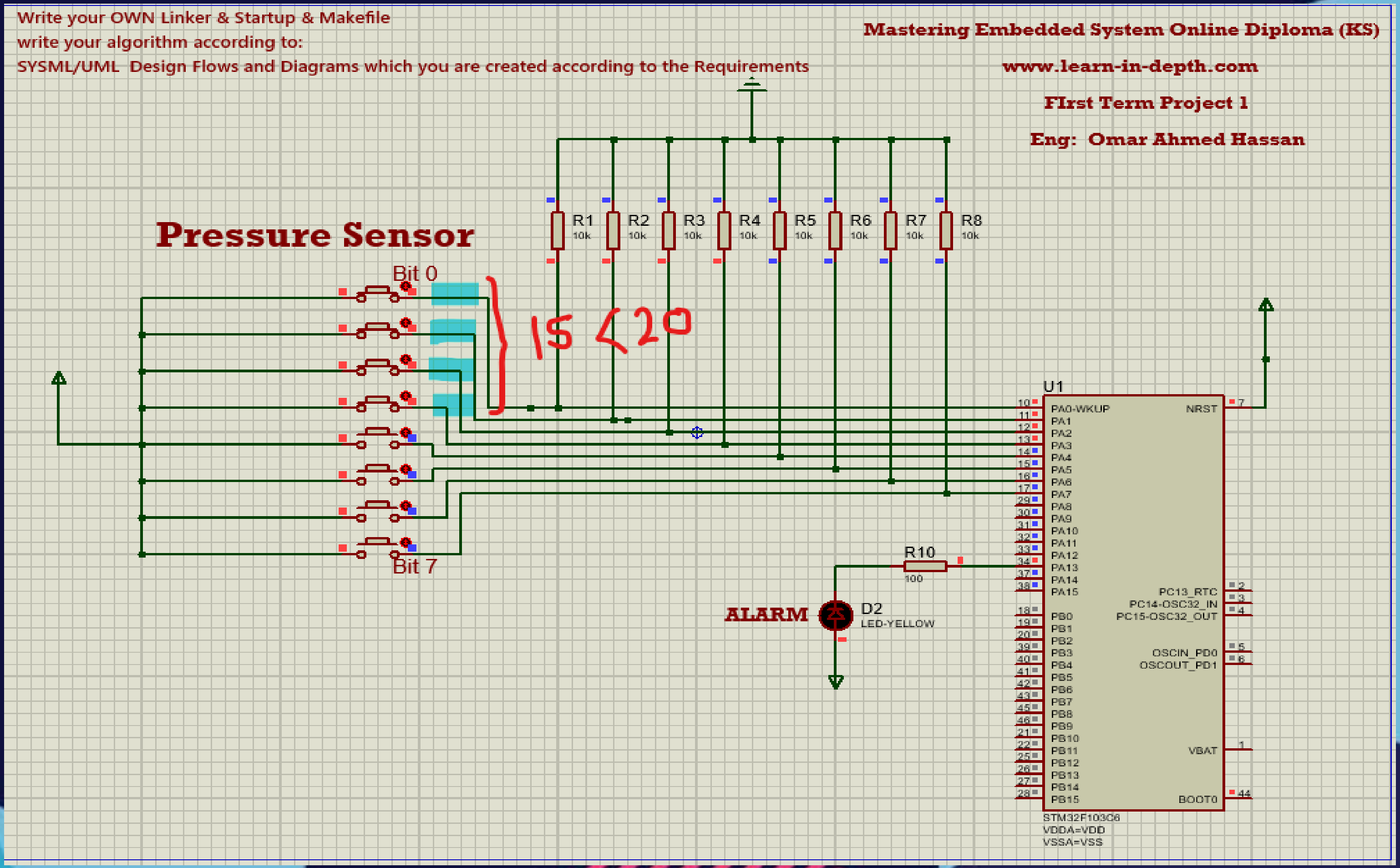


Write your OWN Linker & Startup & Makefile
write your algorithm according to:
SYSML/UML Design Flows and Diagrams which you are created according to the Requirements

Mastering Embedded System Online Diploma (KS)
www.learn-in-depth.com
FIrst Term Project 1
Eng: Omar Ahmed Hassan

PROTEUS SIMULATION

- NEGATIVE TEST (PRESSURE > THRESHOLD)



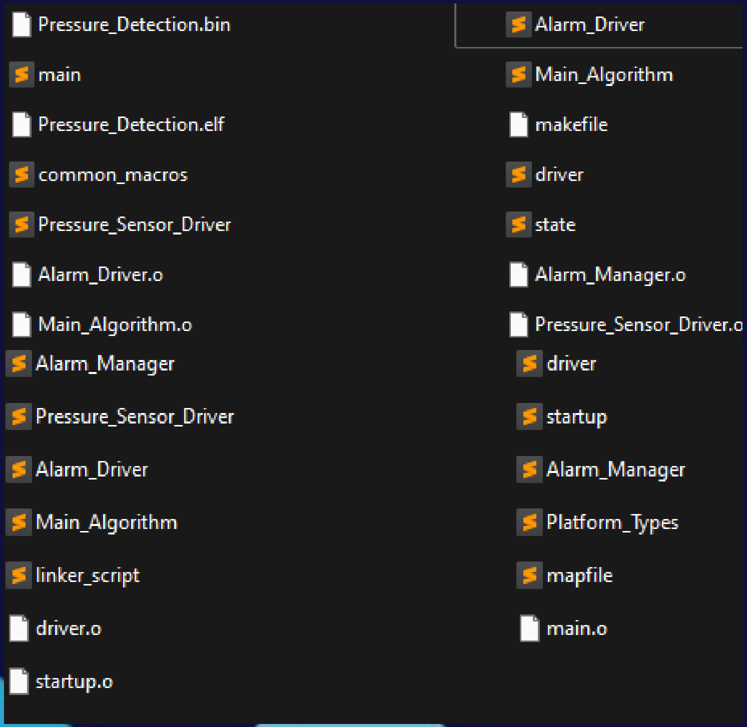
Write your OWN Linker & Startup & Makefile
write your algorithm according to:
SYSML/UML Design Flows and Diagrams which you are created according to the Requirements

Mastering Embedded System Online Diploma (KS)
www.learn-in-depth.com
FIrst Term Project 1
Eng: Omar Ahmed Hassan

SOURCE FILES

- **Source codes were developed according to C89 Standards.**
- **Code was developed perfectly aligned to the System Design.**
- **Source files were fully implemented and debugged manually using only Editor (Sublime) + ARM Cross Toolchain.**

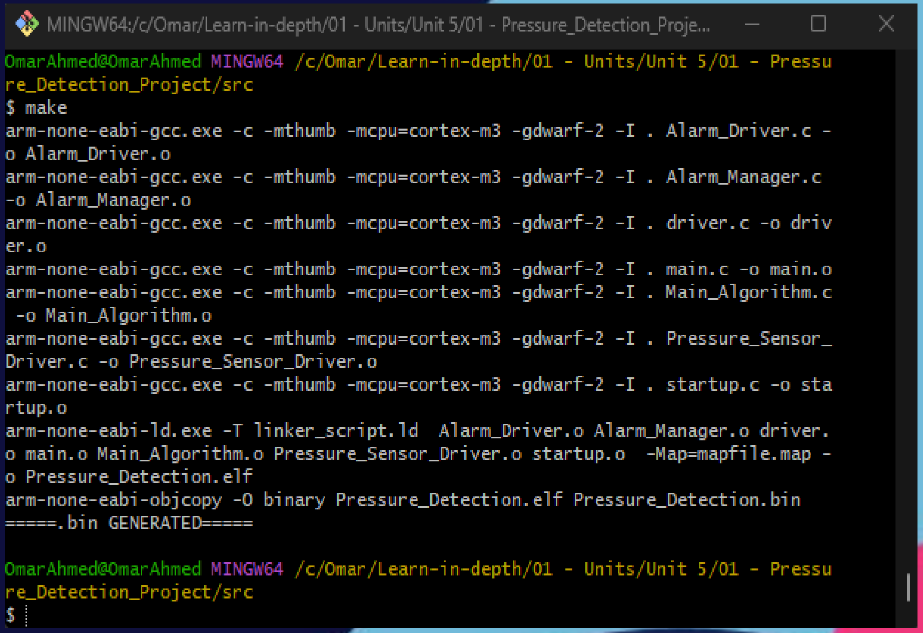
- Source Files



Pressure_Detection.bin
main
Pressure_Detection.elf
common_macros
Pressure_Sensor_Driver
Alarm_Driver.o
Main_Algorithm.o
Alarm_Manager
Pressure_Sensor_Driver
Alarm_Driver
Main_Algorithm
linker_script
driver.o
startup.o

Alarm_Driver
Main_Algorithm
makefile
driver
state
Alarm_Manager.o
Pressure_Sensor_Driver.o
driver
startup
Alarm_Manager
Platform_Types
mapfile
main.o

- **Code Generation**



MINGW64:/c/Omar/Learn-in-depth/01 - Units/Unit 5/01 - Pressure_Detection_Proje...

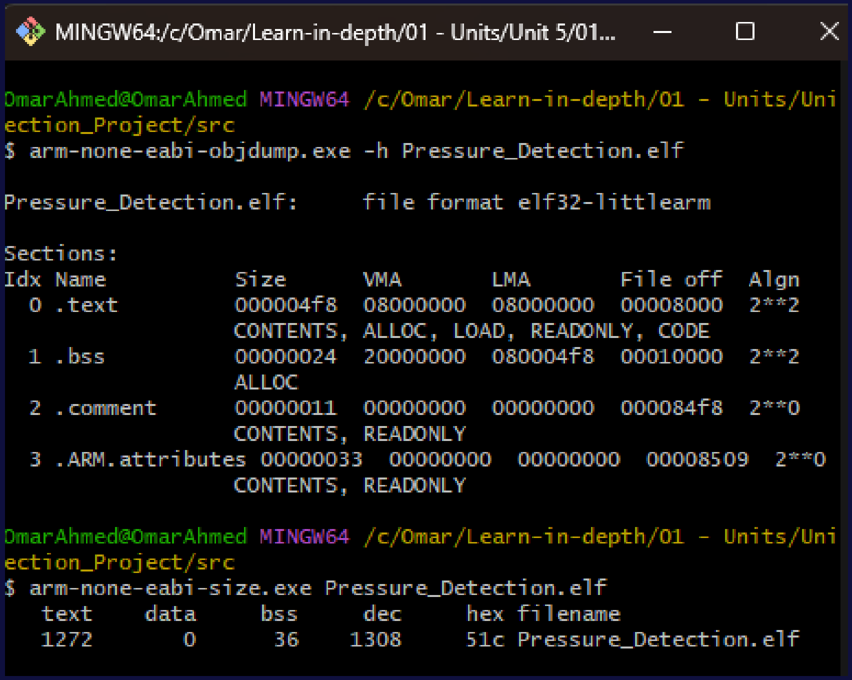
```
OmarAhmed@OmarAhmed MINGW64 /c/Omar/Learn-in-depth/01 - Units/Unit 5/01 - Pressu
re_Detection_Project/src
$ make
arm-none-eabi-gcc.exe -c -mthumb -mcpu=cortex-m3 -gdwarf-2 -I . Alarm_Driver.c -
o Alarm_Driver.o
arm-none-eabi-gcc.exe -c -mthumb -mcpu=cortex-m3 -gdwarf-2 -I . Alarm_Manager.c
-o Alarm_Manager.o
arm-none-eabi-gcc.exe -c -mthumb -mcpu=cortex-m3 -gdwarf-2 -I . driver.c -o driv
er.o
arm-none-eabi-gcc.exe -c -mthumb -mcpu=cortex-m3 -gdwarf-2 -I . main.c -o main.o
arm-none-eabi-gcc.exe -c -mthumb -mcpu=cortex-m3 -gdwarf-2 -I . Main_Algorithm.c
 -o Main_Algorithm.o
arm-none-eabi-gcc.exe -c -mthumb -mcpu=cortex-m3 -gdwarf-2 -I . Pressure_Sensor_
Driver.c -o Pressure_Sensor_Driver.o
arm-none-eabi-gcc.exe -c -mthumb -mcpu=cortex-m3 -gdwarf-2 -I . startup.c -o sta
rtup.o
arm-none-eabi-ld.exe -T linker_script.ld  Alarm_Driver.o Alarm_Manager.o driver.
o main.o Main_Algorithm.o Pressure_Sensor_Driver.o startup.o  -Map=mapfile.map -
o Pressure_Detection.elf
arm-none-eabi-objcopy -O binary Pressure_Detection.elf Pressure_Detection.bin
=====.bin GENERATED=====

OmarAhmed@OmarAhmed MINGW64 /c/Omar/Learn-in-depth/01 - Units/Unit 5/01 - Pressu
re_Detection_Project/src
$
```

- Please feel free to access the source code on my GitHub profile : Repositoy

EXECUTABLE SPECS

- Executable Sections & its size



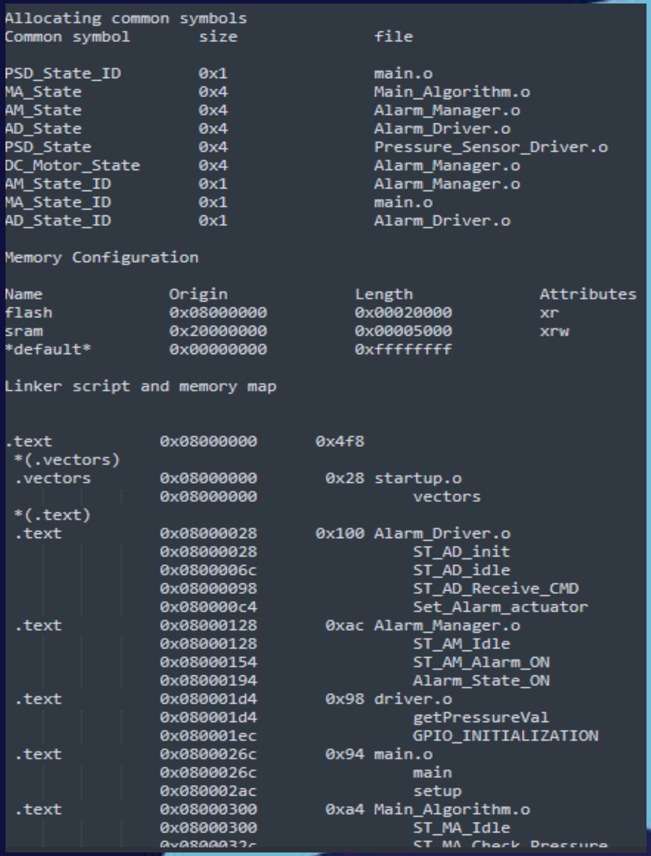
```
OmarAhmed@OmarAhmed MINGW64 /c/Omar/Learn-in-depth/01 - Units/Uni
ection_Project/src
$ arm-none-eabi-objdump.exe -h Pressure_Detection.elf

Pressure_Detection.elf:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         000004f8  08000000  08000000  00008000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .bss          00000024  20000000  080004f8  00010000  2**2
                  ALLOC
  2 .comment      00000011  00000000  00000000  000084f8  2**0
                  CONTENTS, READONLY
  3 .ARM.attributes 00000033  00000000  00000000  00008509  2**0
                  CONTENTS, READONLY

OmarAhmed@OmarAhmed MINGW64 /c/Omar/Learn-in-depth/01 - Units/Uni
ection_Project/src
$ arm-none-eabi-size.exe Pressure_Detection.elf
   text    data     bss     dec     hex filename
   1272       0      36    1308     51c Pressure_Detection.elf
```

- Map File



```
Allocating common symbols
Common symbol       size              file

PSD_State_ID        0x1               main.o
MA_State            0x4               Main_Algorithm.o
AM_State            0x4               Alarm_Manager.o
AD_State            0x4               Alarm_Driver.o
PSD_State           0x4               Pressure_Sensor_Driver.o
DC_Motor_State      0x4               Alarm_Manager.o
AM_State_ID         0x1               Alarm_Manager.o
MA_State_ID         0x1               main.o
AD_State_ID         0x1               Alarm_Driver.o

Memory Configuration

Name             Origin             Length             Attributes
flash            0x08000000         0x00020000         xr
sram             0x20000000         0x00005000         xrw
*default*        0x00000000         0xffffffff

Linker script and memory map


.text           0x08000000      0x4f8
 *(.vectors)
 .vectors       0x08000000       0x28 startup.o
                0x08000000                vectors
 *(.text)
 .text          0x08000028      0x100 Alarm_Driver.o
                0x08000028                ST_AD_init
                0x0800006c                ST_AD_idle
                0x08000098                ST_AD_Receive_CMD
                0x080000c4                Set_Alarm_actuator
 .text          0x08000128       0xac Alarm_Manager.o
                0x08000128                ST_AM_Idle
                0x08000154                ST_AM_Alarm_ON
                0x08000194                Alarm_State_ON
 .text          0x080001d4       0x98 driver.o
                0x080001d4                getPressureVal
                0x080001ec                GPIO_INITIALIZATION
 .text          0x0800026c       0x94 main.o
                0x0800026c                main
                0x080002ac                setup
 .text          0x08000300       0xa4 Main_Algorithm.o
                0x08000300                ST_MA_Idle
                0x0800032c                ST_MA_Check_Pressure
```

Mapfile : *mapfile*.