Mastering Embedded System Online Diploma

<https://www.learn-in-depth.com/>

First Term (Final Project 1)

Eng. Mohamed Abd El-Naby Mohamed

My Profile:

<https://www.learn-in-depth.com/online-diploma/mahameda.naby@gmail.com>

Table of Contents

[LIST OF FIGURES 4](#_Toc101001696)

[Description 5](#_Toc101001697)

[System Specifications 5](#_Toc101001698)

[System Assumptions: 5](#_Toc101001699)

[System Architecture 5](#_Toc101001700)

[1- Case study 5](#_Toc101001701)

[2- Method 5](#_Toc101001702)

[3- Requirement 6](#_Toc101001703)

[4- Space exploration/partitioning 6](#_Toc101001704)

[5- System Analysis 7](#_Toc101001705)

[i- Use Case Diagram 7](#_Toc101001706)

[ii- Activity Diagram 8](#_Toc101001707)

[iii- Sequence Diagram (UML) 9](#_Toc101001708)

[6- System Design 9](#_Toc101001709)

[System Design in Details 10](#_Toc101001710)

[1- Pressure Sensor Logic 10](#_Toc101001711)

[2- Main Algo 11](#_Toc101001712)

[3- Memory Driver Logic 11](#_Toc101001713)

[4- Alarm Manger Logic 12](#_Toc101001714)

[5- Alarm Driver 13](#_Toc101001715)

[Blocks Implementation in C 14](#_Toc101001716)

[1- Pressure Sensor 14](#_Toc101001717)

[2- Main Algo 14](#_Toc101001718)

[3- Alarm Manger 15](#_Toc101001719)

[4- Alarm 16](#_Toc101001720)

[Boot Sequence 17](#_Toc101001721)

[Linker script file 18](#_Toc101001722)

[Startup 19](#_Toc101001723)

[Make 20](#_Toc101001724)

[Map File 21](#_Toc101001725)

[Memory dump 22](#_Toc101001726)

[Symbols 24](#_Toc101001727)

[ELF image details 25](#_Toc101001728)

[Hardware Simulation 26](#_Toc101001729)

# LIST OF FIGURES

[Figure 1:System Architecture 5](#_Toc101001545)

[Figure 2:System Requirement 6](file:///E:\COURSES\Learn-in-Depth\pressureControl.docx#_Toc101001546)

[Figure 3:System Partitioning 6](file:///E:\COURSES\Learn-in-Depth\pressureControl.docx#_Toc101001547)

[Figure 4:Use Case Diagram 7](#_Toc101001548)

[Figure 5:Activity Diagram 8](#_Toc101001549)

[Figure 6:UML Diagram 9](#_Toc101001550)

[Figure 7:System Design 9](#_Toc101001551)

[Figure 8:Pressure Sensor Logic 10](#_Toc101001552)

[Figure 9:Main Computational Logic 11](#_Toc101001553)

[Figure 10:Memory Storing Logic 11](#_Toc101001554)

[Figure 11:Alarm Manger Logic 12](#_Toc101001555)

[Figure 12:Alarm Logic 13](#_Toc101001556)

[Figure 13:Pressure C code 14](#_Toc101001557)

[Figure 14:MainAlgo C code 14](#_Toc101001558)

[Figure 15:Alarm Manger C code 15](#_Toc101001559)

[Figure 16:Alarm C code 16](#_Toc101001560)

[Figure 17:Sequence 17](#_Toc101001561)

[Figure 18:Memory Areas in Linker Script 18](#_Toc101001562)

[Figure 19:Memory Section in Linker script 18](#_Toc101001563)

[Figure 20:Reset Handler in startup code 19](#_Toc101001564)

[Figure 21:Vector Section in startup code 19](#_Toc101001565)

[Figure 22: vector functions 19](#_Toc101001566)

[Figure 23:Default Handler 20](#_Toc101001567)

[Figure 24: Memory configuration in Linkerscript and map file. 21](#_Toc101001568)

[Figure 25:Vector table position in map file 21](#_Toc101001569)

[Figure 26:Disassemble of elf image 22](#_Toc101001570)

[Figure 27:Memory Dump with debug section 22](#_Toc101001571)

[Figure 28:position of .data section in flash and sram 23](#_Toc101001572)

[Figure 29:Memory dump without dung sections 23](#_Toc101001573)

[Figure 30:Symbols of ELF image 24](#_Toc101001574)

[Figure 31:ELF image details 25](#_Toc101001575)

[Figure 32:ELF image attributes 25](#_Toc101001576)

[Figure 33:simulation test case 1 26](#_Toc101001577)

[Figure 34:simulation test case 2 27](#_Toc101001578)

Pressure Detection System

# Description

This System deliver software that detect high pressure and alarm if pressure is high.

# System Specifications

1. Pressure detection at 60 bar inform with alarm.
2. Alarm duration 60 second
3. Optional Keeps track the measured values.

# System Assumptions:

1. Controller setup and shutdown procedure are not modeled.
2. Controller maintenance is not modeled.
3. Pressure sensor never fails.
4. Alarm never fails.
5. All components never face power cut.

# System Architecture

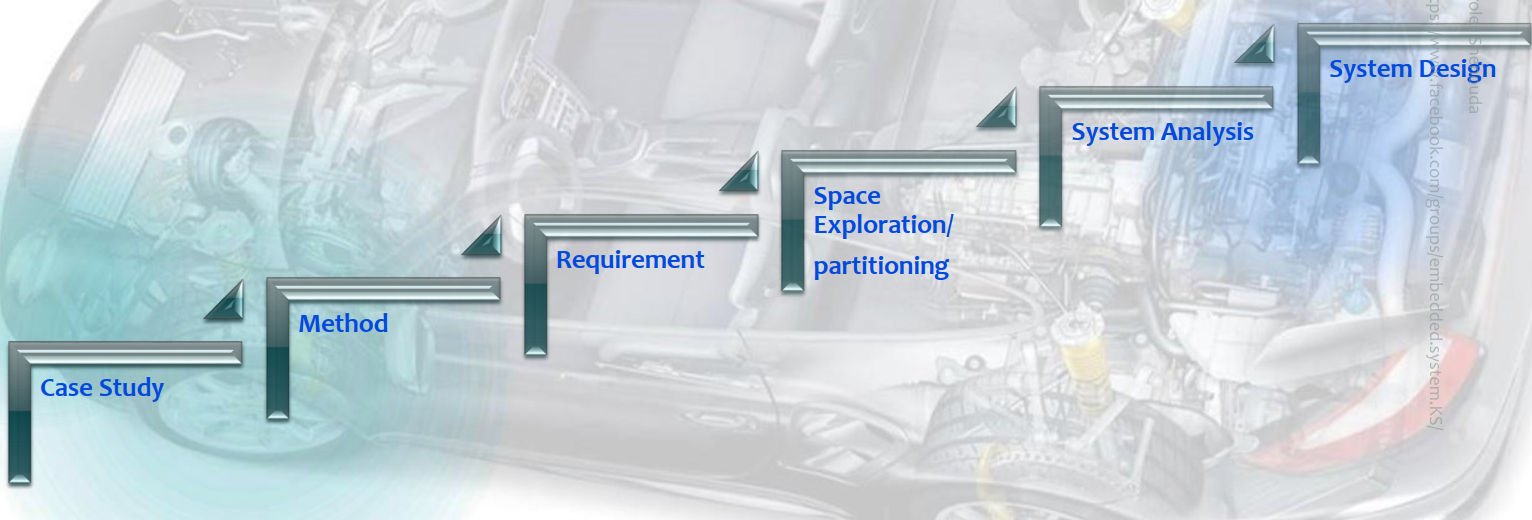


Figure :System Architecture

## Case study

software that detects high pressure and alarm for info.

## Method

Using V-Model

## Requirement

Figure :System Requirement

## Space exploration/partitioning

Figure :System Partitioning

STM 32-BIT MCU

ADC

LED info.

Storage

Compute Pressure

Main Algo

Air Pressure

Pressure Value

PressureVal > threshold

LED State

Pressure Sensor

Alarm

Memory

Optional

I used STM32F103C6 MCU which based on ARM Cortex m3 microprocessor its specification

1. ARM 32-bit Cortex™-M3 CPU Core
2. 72 MHz maximum frequency
3. Single-cycle multiplication and hardware division.
4. Memories
5. 32 Kbytes of Flash memory
6. 10 Kbytes of SRAM
7. Clock, reset and supply management
8. 2.0 to 3.6 V application supply and I/Os.
9. 4-to-16 MHz crystal oscillator.
10. 32 kHz oscillator for RTC with calibration

## System Analysis

### Use Case Diagram

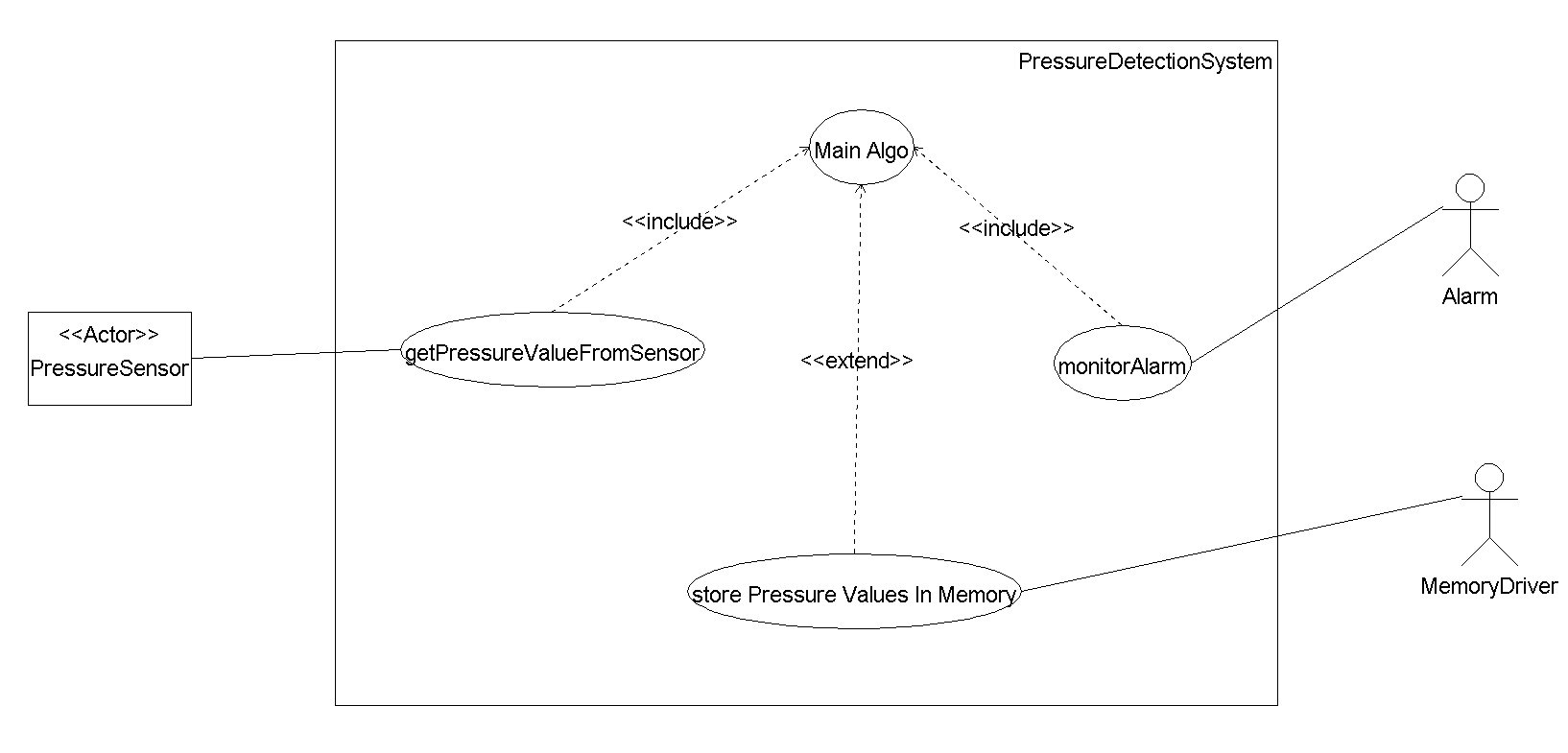


Figure :Use Case Diagram

### Activity Diagram

Diagram

Description automatically generated

Figure :Activity Diagram

### Sequence Diagram (UML)

Diagram

Description automatically generated

Figure :UML Diagram

## System Design

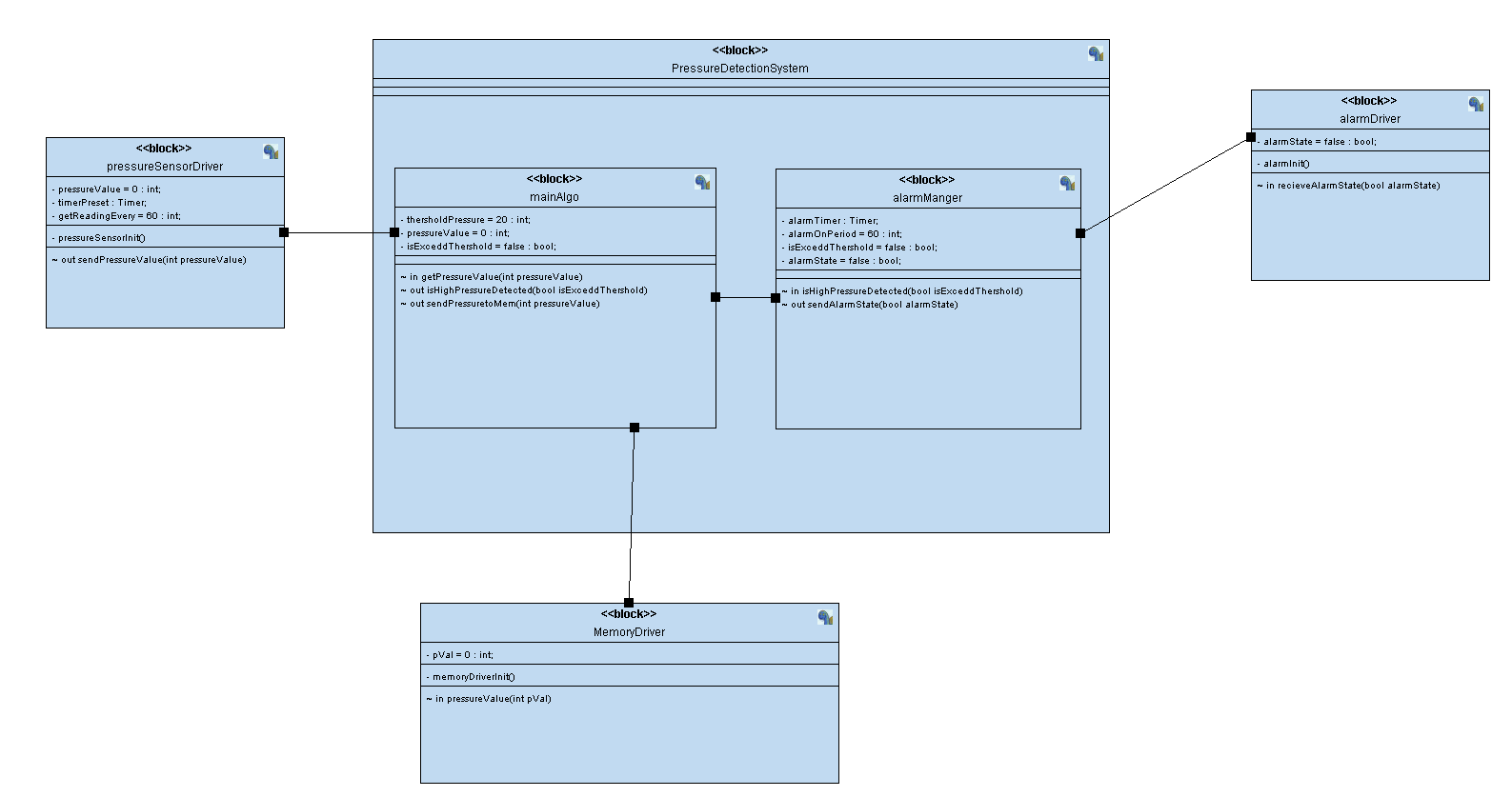


Figure :System Design

# System Design in Details

## Pressure Sensor Logic

Diagram

Description automatically generated

It emulates sensor value

Figure :Pressure Sensor Logic

## Main Algo

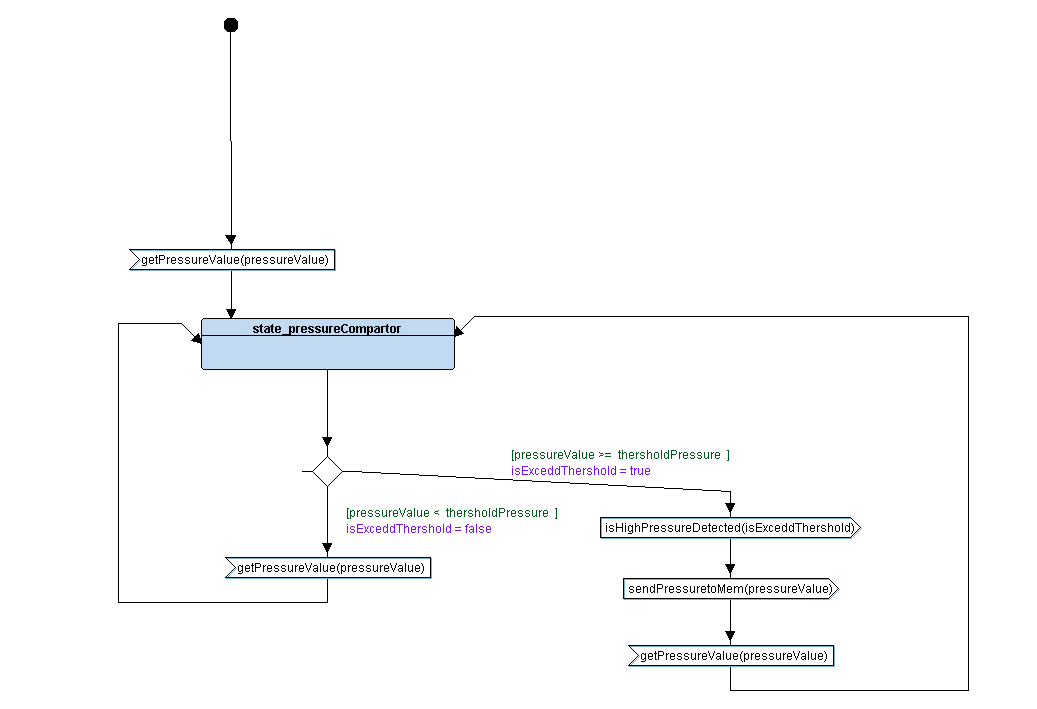


Figure :Main Computational Logic

## Memory Driver Logic

Diagram

Description automatically generated

Figure :Memory Storing Logic

## Alarm Manger Logic

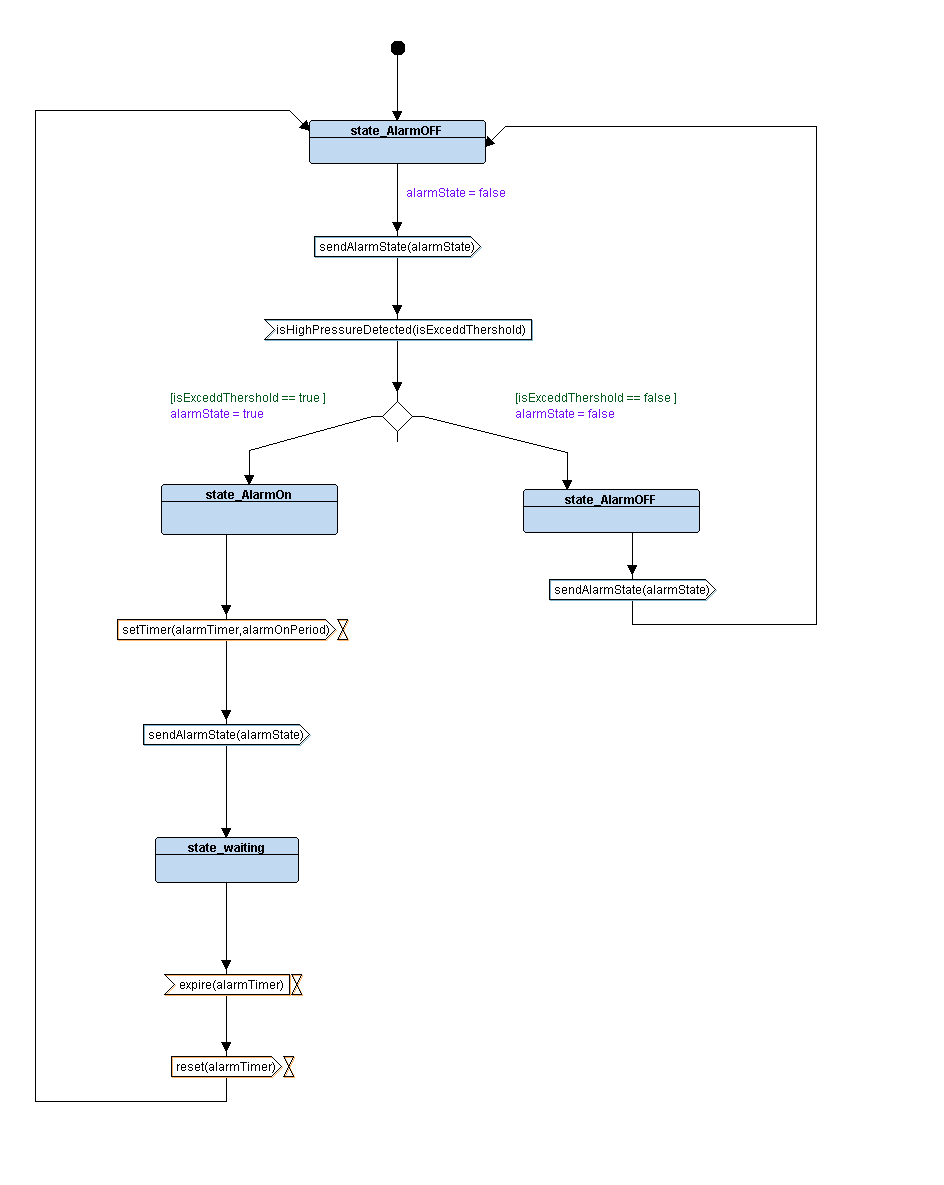


Figure :Alarm Manger Logic

## Alarm Driver

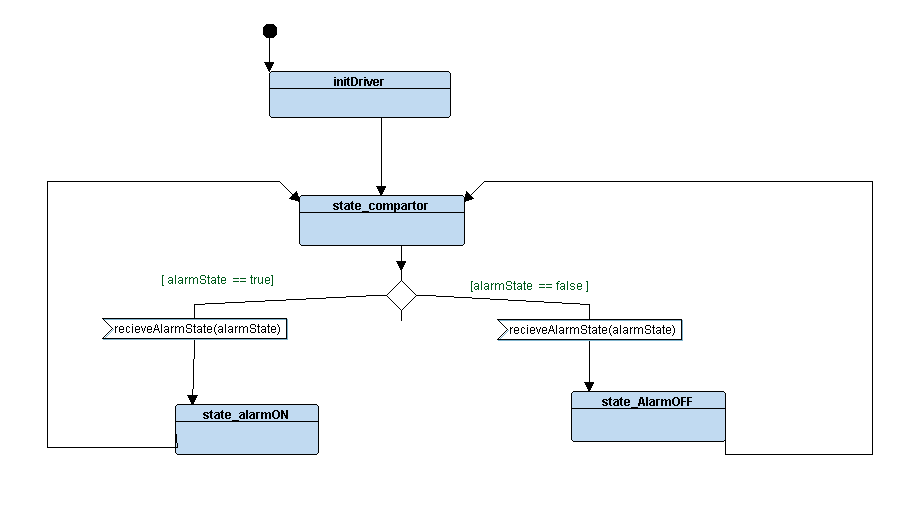


Figure :Alarm Logic

# Blocks Implementation in C

## Pressure Sensor

Text

Description automatically generated

Figure :Pressure C code

## Main Algo

Text

Description automatically generated

Figure :MainAlgo C code

## Alarm Manger

A screenshot of a computer

Description automatically generated with medium confidence

Figure :Alarm Manger C code

## Alarm

Text

Description automatically generated

Figure :Alarm C code

For Source Code

# Boot Sequence

Entry Point (Reset Section)

Reset section

FLASH MEM

BareMetal SW

A picture containing map

Description automatically generated

Figure :Sequence

Our entry point is reset handler that move .data from FLASH to SRAM and reserve .bss section in SRAM.

# Linker script file

Memory Areas and stack size

A screenshot of a computer

Description automatically generated with medium confidence

Figure :Memory Areas in Linker Script

Sections

Text

Description automatically generated

Figure :Memory Section in Linker script

# Startup

resetHandler()

Text

Description automatically generated

Figure :Reset Handler in startup code

Vector section

A screenshot of a computer

Description automatically generated with medium confidence

Figure :Vector Section in startup code

Vector table functions

Text

Description automatically generated

Figure : vector functions

Weak --> to be overwritten, alias to make declaration emitted to be alias for another function.

Default handler

Shape

Description automatically generated with low confidence

Figure :Default Handler

# Make

Text

Description automatically generated

Graphical user interface

Description automatically generated with medium confidence

# Map File

* Memory Configuration

A screenshot of a computer

Description automatically generated

In LinkerScript.ld

In map file

Figure : Memory configuration in Linkerscript and map file.

* Memory map

Graphical user interface, text, application

Description automatically generated



Start address of flash memory



Figure :Vector table position in map file

Disassemble elf image

Text

Description automatically generated

Figure :Disassemble of elf image

# Memory dump

Graphical user interface, text

Description automatically generated

Debug info

Figure :Memory Dump with debug section

Text

Description automatically generated

Figure :position of .data section in flash and sram

Sections without debug

Text

Description automatically generated

Figure :Memory dump without dung sections

# Symbols

Text

Description automatically generated

Figure :Symbols of ELF image

All symbols successfully resolved

For each symbol check [link](https://sourceware.org/binutils/docs/binutils/nm.html)

# ELF image details

Text

Description automatically generated

Figure :ELF image details

Text

Description automatically generated

Figure :ELF image attributes

# Hardware Simulation

Diagram, schematic

Description automatically generated

Figure :simulation test case 1

I/P Pressure = 12 bar < 20 BAR

O/P Alarm = OFF

Diagram, schematic

Description automatically generated

Figure :simulation test case 2

I/P Pressure = 32 bar > 20 BAR

O/P Alarm = ON FOR 60 SEC