Pressure Controller

project

STUDENT: YOUSSEF SAMY YOUSSEF

EMAIL: yosefsamy019@gmail.com

INSTRUCTOR: ENG. KERLOES KHALIL

DIPLOMA: LEARN IN DEPTH

GOALS

- **This Project targets:**
  - Embedded C
  - Understanding Build Process
  - MakeFile
  - Startup
  - Linker Script
  - UML: unified modeling Language

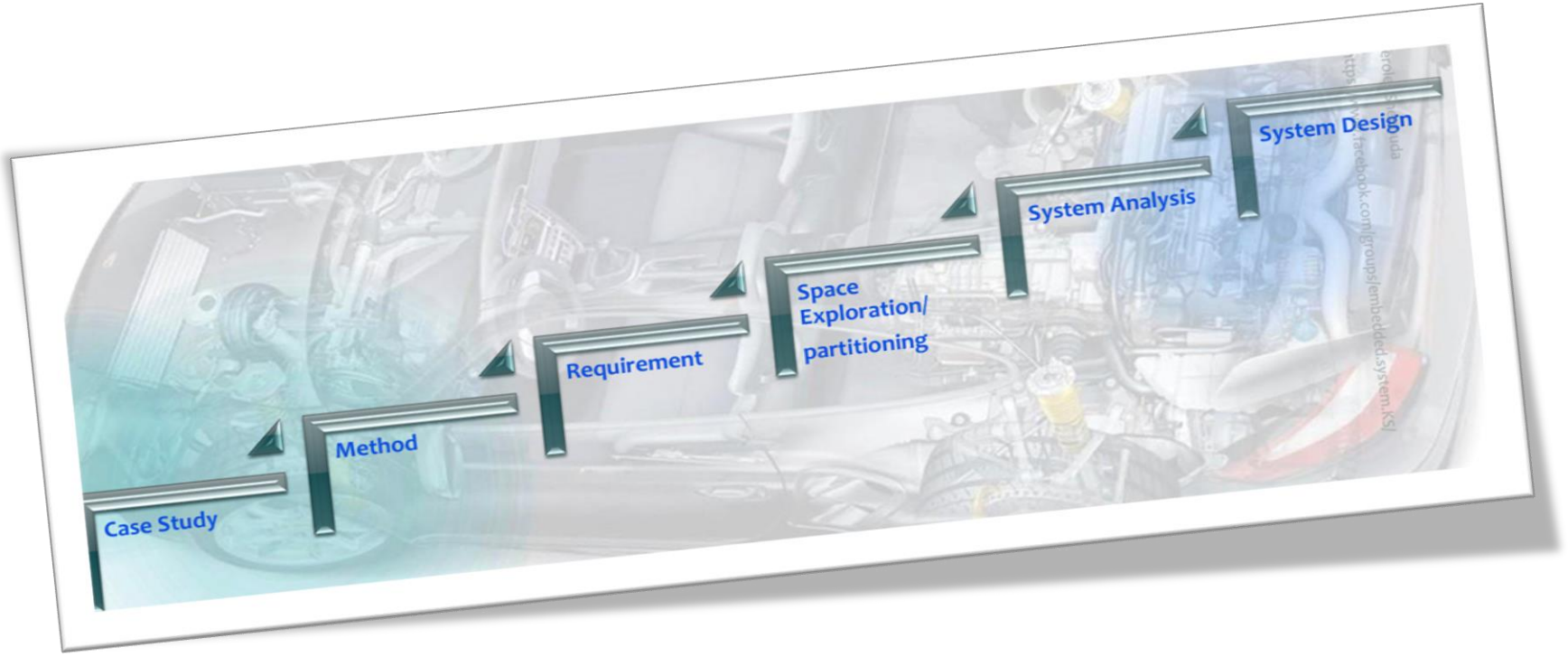
- **Board:** STM32 (arm processor)
- **Tool Chain:** Arm-none-eabi

Project Idea

- A ”client” expects you to deliver the software of the following system:
  - A pressure controller with an alarm in the cabin informs the crew of a cabin when the pressure exceeds 20 bars.
  - The Alarm duration has non specific duration.
  - The Alarm can be LED or Buzzer.

System Architecting

- Case Study
- Method
- Requirements
- Space Exploration
- System Analysis
- System Design



1. Case Study

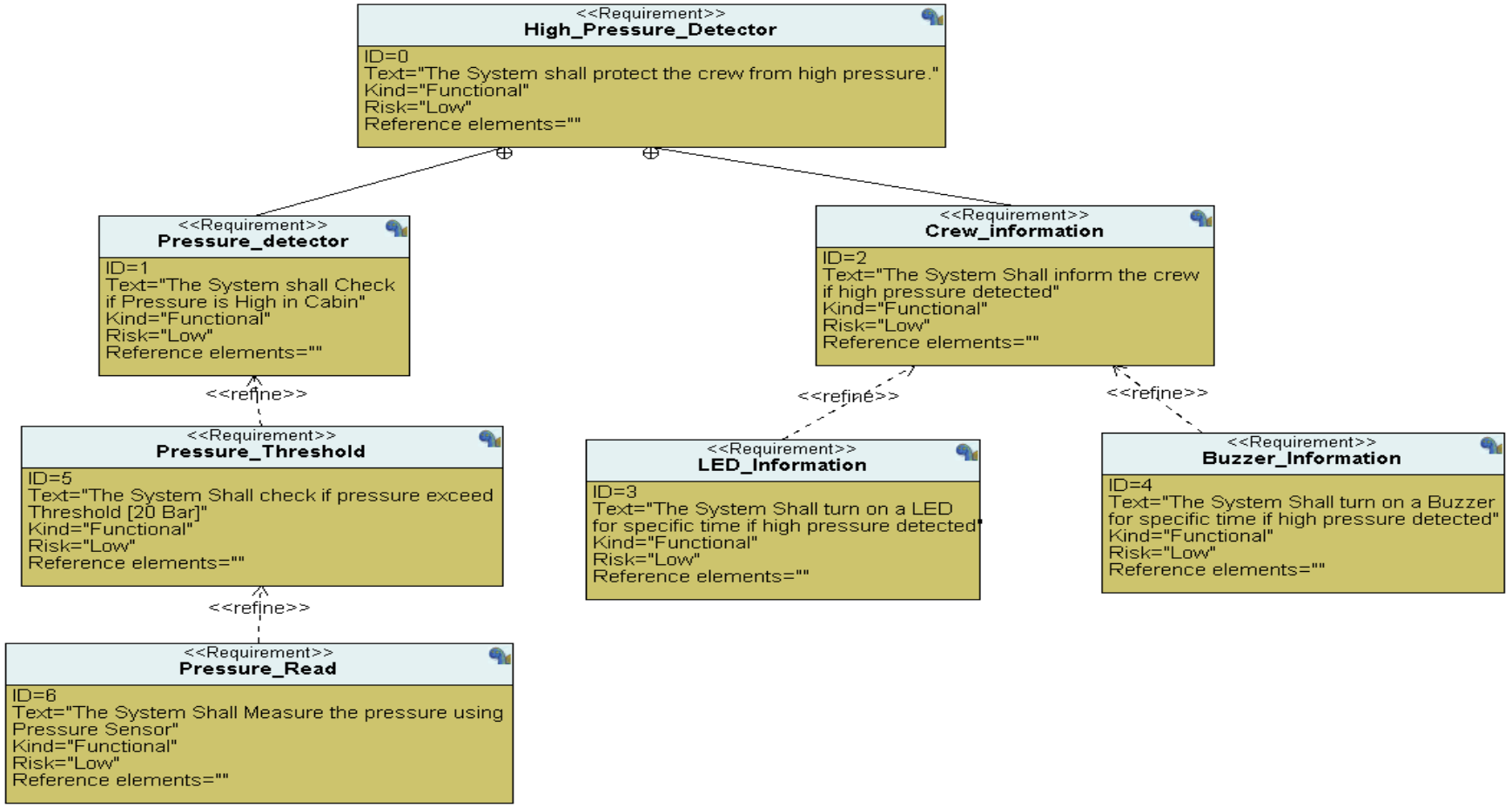
- Idea: A pressure controller with an alarm in the cabin informs the crew of a cabin when the pressure exceeds 20 bars.

- 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

2. Methods (SDLC)

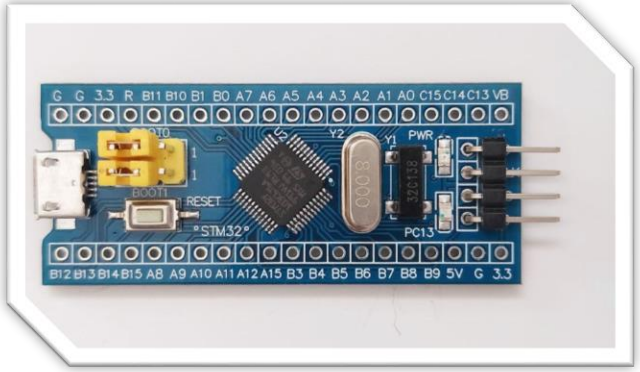
- The SW development can follow one of these life cycle:
  - Waterfall
  - V-model
  - Agile
  - Spiral
- Actually, Selecting the SDLC is out of project scope right now.

3. Requirements



4. Space Exploration

- The system will be implemented using STM32 board.



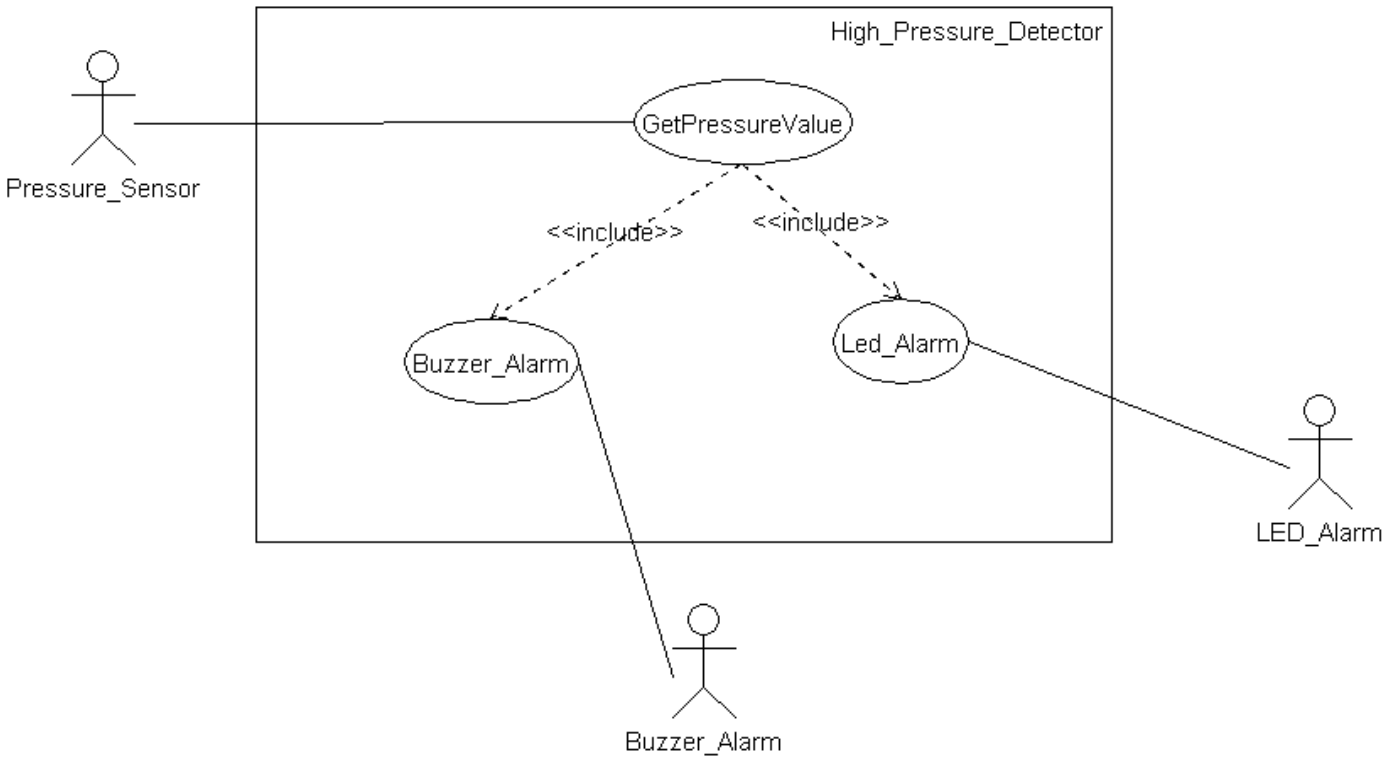
- The processor used is: ARM-CORTEX-M

5. System Analysis

- The System analysis is divided into 3 diagrams:
  - Case Diagram
  - Activity Diagram
  - Sequence Diagram

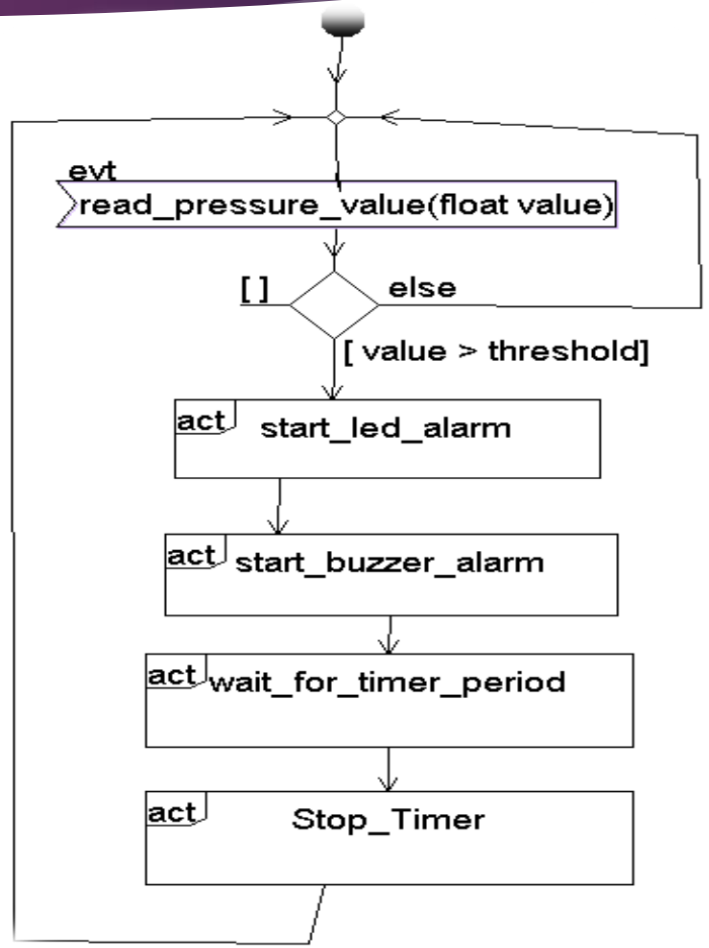
5.1 Case Diagram

- This Diagram defines the boundary of the system:



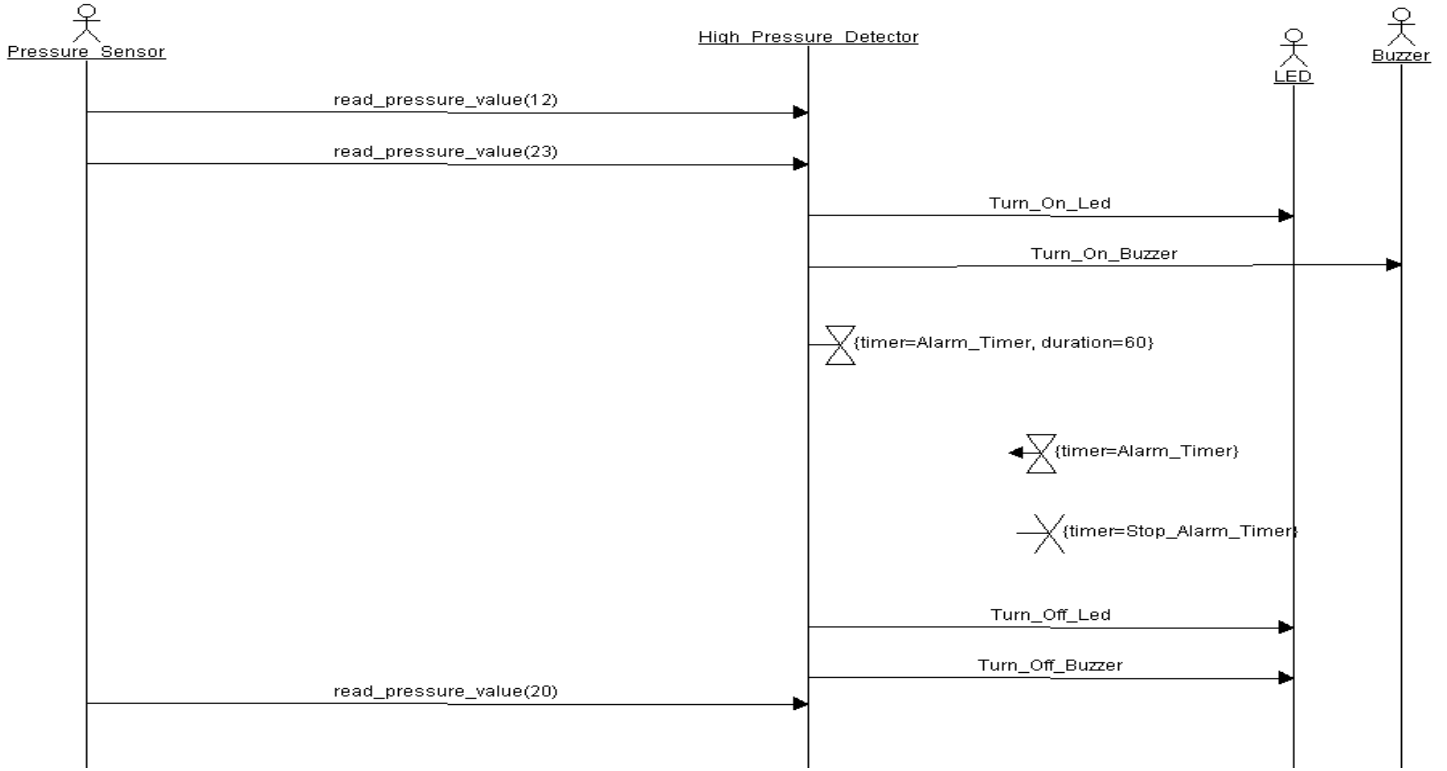
5.2 Activity Diagram

- This Diagram describe the workflow behavior of a system:



5.3 Sequence Diagram

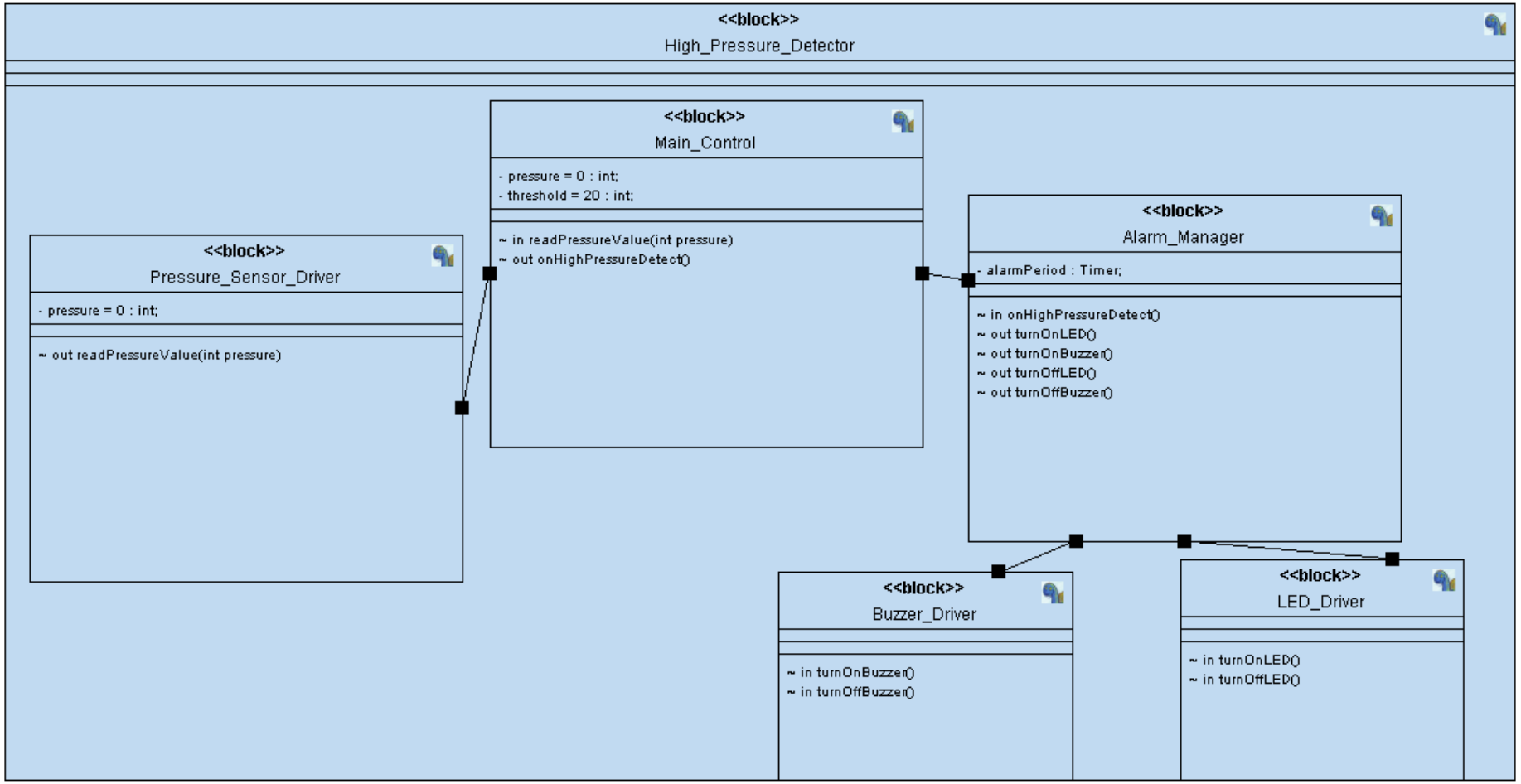
- An interaction diagram that details how operations are carried out.



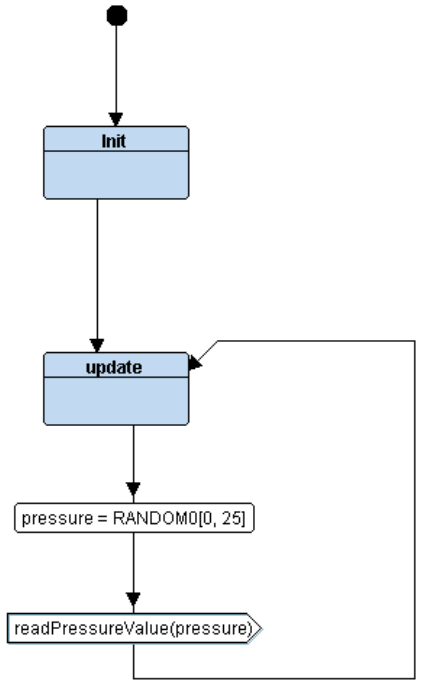
6. System Design

- Design required the system using block diagrams and state machine.
- Use TTool Program to make design

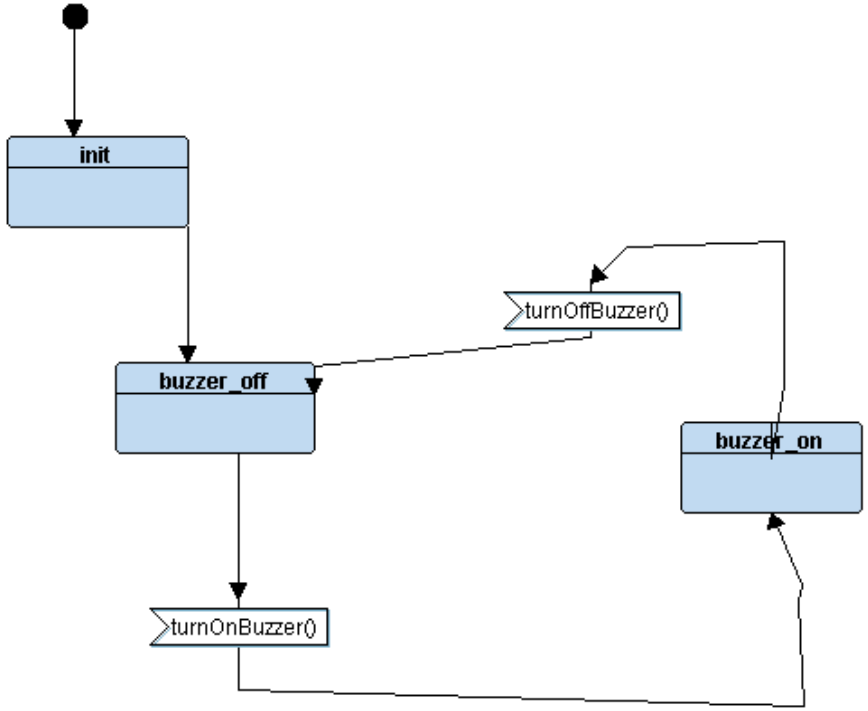
6.1 Block Diagram



6.2 State Machines

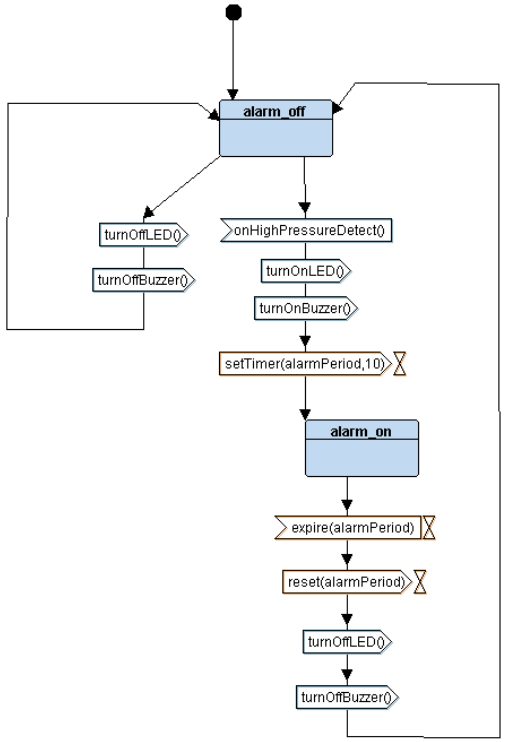


- Pressor Sensor Driver

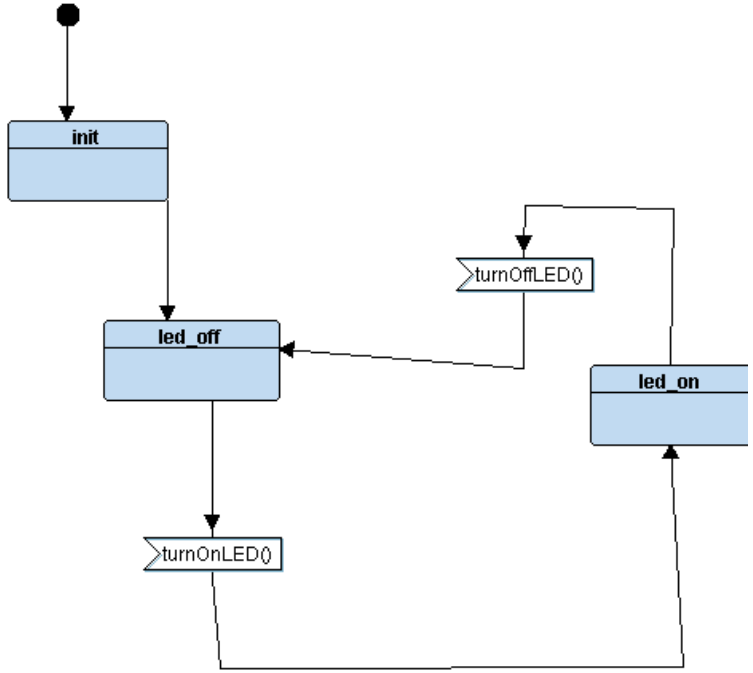


- Buzzer Driver

6.2 State Machines



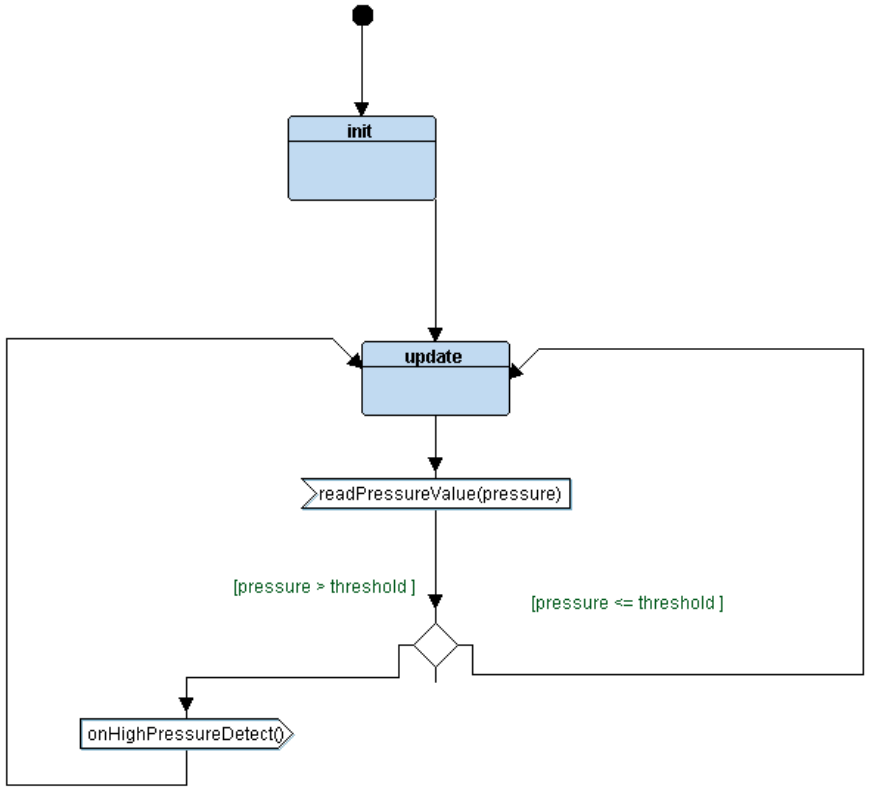
- Alarm manager



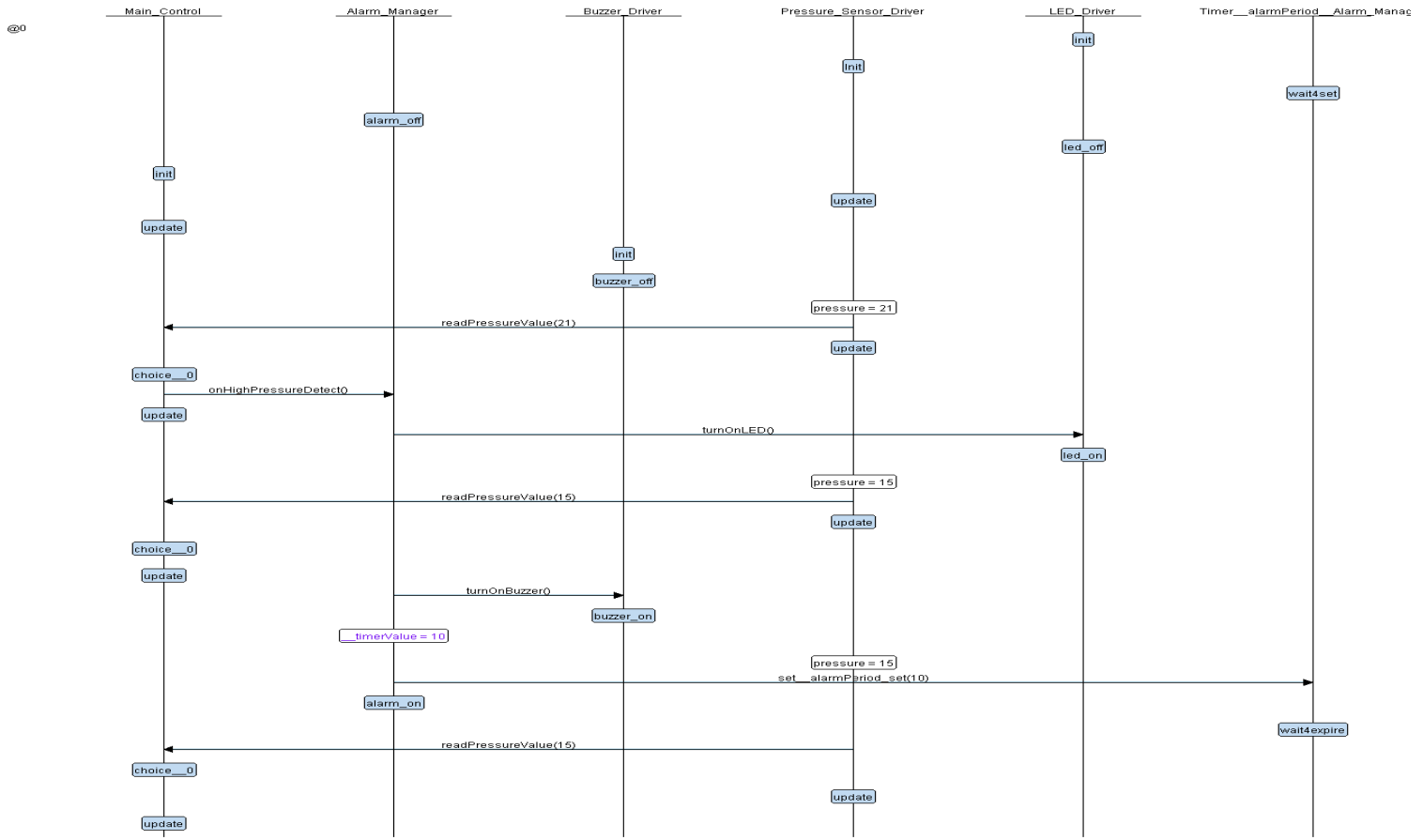
- Led Driver

6.2 State Machines

- Main Control



6.3 Simulate the system



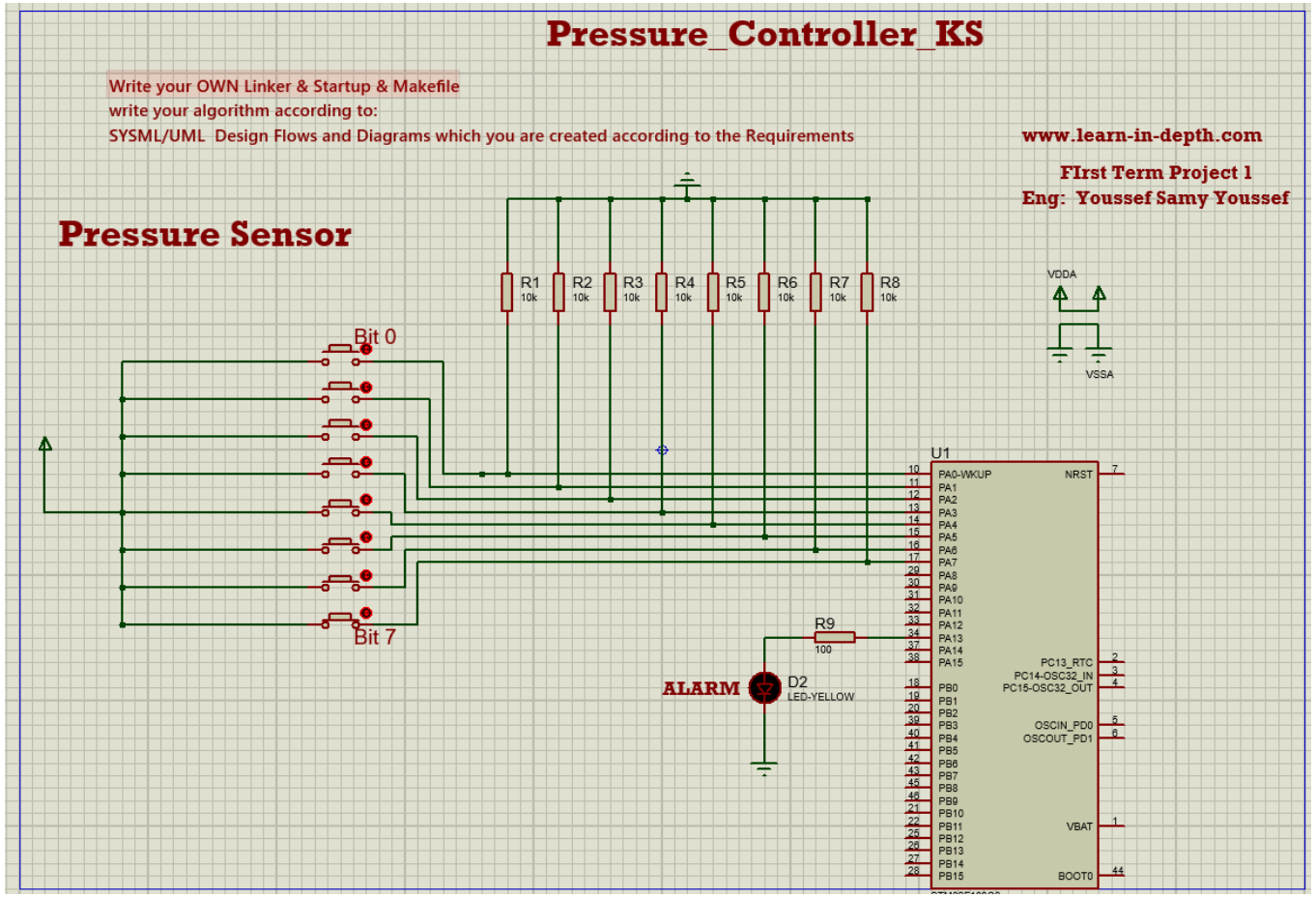
7. Project Implementation

- Language: C
- Board: STM32
- Write own Makefile, Linker Script, Startup
- Each block diagram is implemented in 2 files:
  - .c &.h

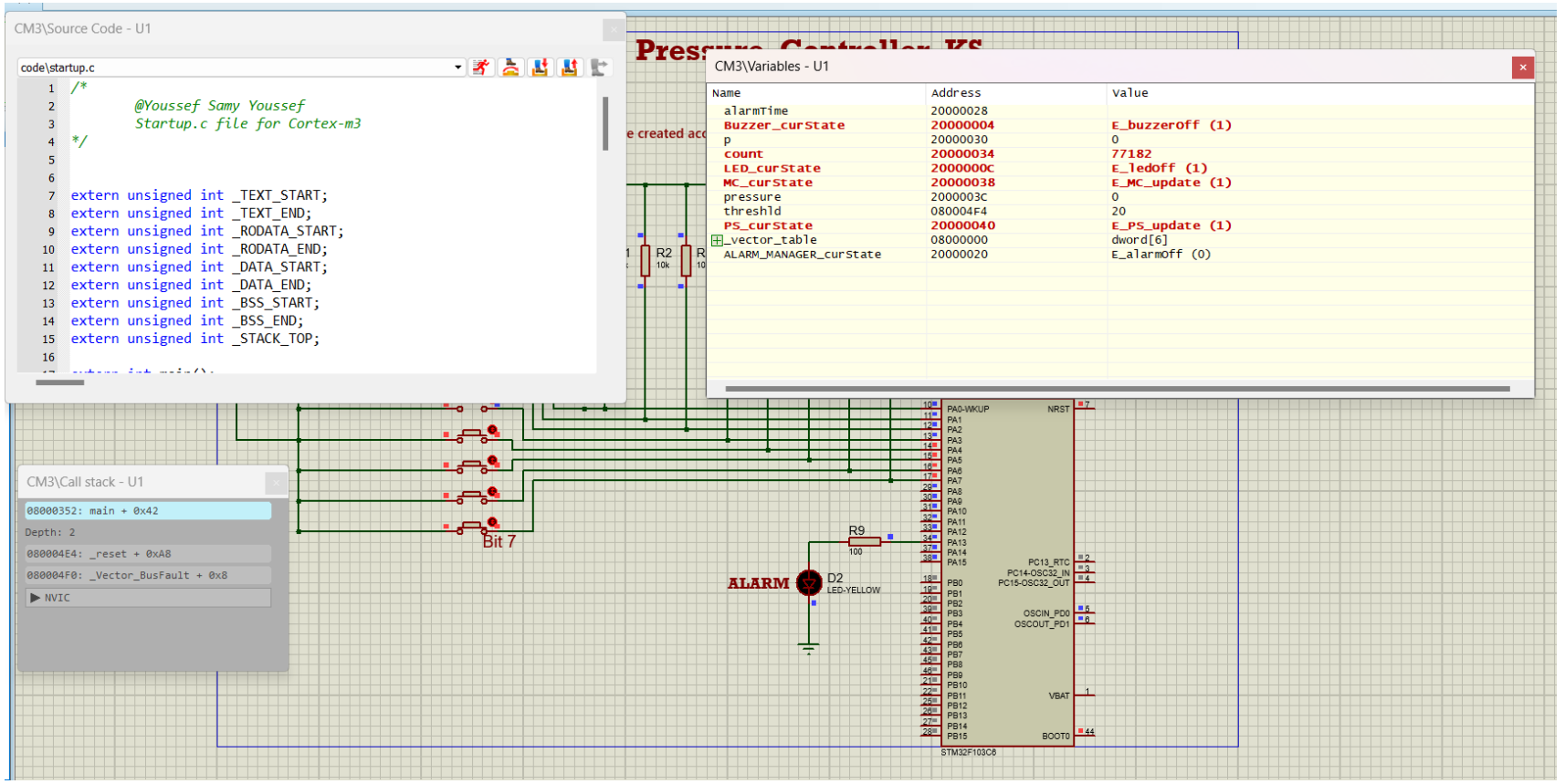
7. Project Implementation

The code is available on my Account on GitHub.

8. Proteus Simulation



8. Proteus Simulation



9. Binary Utilitis (nm)

```
arm-none-eabi-nm pressure_controller.elf
```

```
20000020 B _BSS_START
2000001c D _DATA_END
20000000 D _DATA_START
0800043c T _reset
080004f8 R _RODATA_END
080004f4 R _RODATA_START
20001044 B _STACK_TOP
080004f4 T _TEXT_END
08000000 T _TEXT_START
080004e8 W _Vector_BusFault
080004e8 T _Vector_deafultHandler
080004e8 W _Vector_HardFault
080004e8 W _Vector_MemManage
080004e8 W _Vector_NMI
08000000 t _vector_table
080004e8 W _Vector_UsageFault
20000000 D ALARM_MANAGER_curCall
20000020 b ALARM_MANAGER_curState
20000028 b alarmTime
20000008 D Buzzer_curCall
20000004 d Buzzer_curState
20000034 B count
```

```
08000164 T getPressureVal
080001fc T GPIO_INITIALIZATION
20000010 D LED_curCall
2000000c d LED_curState
08000310 T main
20000014 D MC_curCall
20000038 b MC_curState
20000030 B p
2000003c b pressure
20000018 D PS_curCall
20000040 b PS_curState
080001ac T Set_Alarm_actuator
080000b0 T Signal_onHighPressureDetect
080003b4 T Signal_readPressureSensor
08000148 T Signal_turnOffBuzzer
080002f4 T Signal_turnOffLed
0800012c T Signal_turnOnBuzzer
080002d8 T Signal_turnOnLed
08000018 T state_define_E_alarmOff
08000034 T state_define_E_alarmOn
080000f4 T state_define_E_buzzerOff
08000110 T state_define_E_buzzerOn
080002a0 T state_define_E_ledOff
```

```
080002bc T state_define_E_ledOn
08000358 T state_define_E_MC_init
08000384 T state_define_E_MC_update
080003d4 T state_define_E_PS_init
08000400 T state_define_E_PS_update
080004f4 R threshld
```

9. Binary Utilitis (size)

```
arm-none-eabi-size pressure_controller.elf
   text    data     bss     dec     hex filename
   1272      28    4132    5432    1538 pressure_controller.elf
```

9. Binary Utilitis (readelf)

```
arm-none-eabi-readelf -a pressure_controller.elf
ELF Header:
  Magic:   7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00 00
  Class:                             ELF32
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              EXEC (Executable file)
  Machine:                           ARM
  Version:                           0x1
  Entry point address:               0x8000000
  Start of program headers:          52 (bytes into file)
  Start of section headers:          73644 (bytes into file)
  Flags:                             0x5000002, has entry point, Version5 EABI
  Size of this header:               52 (bytes)
  Size of program headers:           32 (bytes)
  Number of program headers:         2
  Size of section headers:           40 (bytes)
  Number of section headers:         17
  Section header string table index: 14
```

```
Section Headers:
  [Nr] Name              Type            Addr     Off    Size   ES Flg Lk Inf Al
  [ 0]                   NULL            00000000 000000 000000 00      0   0  0
  [ 1] .text             PROGBITS        08000000 008000 0004f4 00  AX  0   0  4
  [ 2] .rodata           PROGBITS        080004f4 0084f4 000004 00   A  0   0  4
  [ 3] .data             PROGBITS        20000000 010000 00001c 00  WA  0   0  4
  [ 4] .bss              NOBITS          20000020 01001c 001024 00  WA  0   0  8
  [ 5] .debug_info       PROGBITS        00000000 01001c 0008f9 00      0   0  1
  [ 6] .debug_abbrev     PROGBITS        00000000 010915 000553 00      0   0  1
  [ 7] .debug_loc        PROGBITS        00000000 010e68 000404 00      0   0  1
  [ 8] .debug_aranges    PROGBITS        00000000 01126c 000100 00      0   0  1
  [ 9] .debug_line       PROGBITS        00000000 01136c 000408 00      0   0  1
  [10] .debug_str        PROGBITS        00000000 011774 000475 01  MS  0   0  1
  [11] .comment          PROGBITS        00000000 011be9 000011 01  MS  0   0  1
  [12] .ARM.attributes   ARM_ATTRIBUTES  00000000 011bfa 000033 00      0   0  1
  [13] .debug_frame      PROGBITS        00000000 011c30 0002d4 00      0   0  4
  [14] .shstrtab         STRTAB          00000000 011f04 0000a5 00      0   0  1
  [15] .symtab           SYMTAB          00000000 012254 000660 10     16  58  4
  [16] .strtab           STRTAB          00000000 0128b4 0003b4 00      0   0  1
```

9. Binary Utilitis (objdump)

```
arm-none-eabi-objdump -x pressure_controller.elf
```

```
SYMBOL TABLE:
08000000 l    d  .text  00000000 .text
080004f4 l    d  .rodata        00000000 .rodata
20000000 l    d  .data  00000000 .data
20000020 l    d  .bss   00000000 .bss
00000000 l    d  .debug_info    00000000 .debug_info
00000000 l    d  .debug_abbrev  00000000 .debug_abbrev
00000000 l    d  .debug_loc     00000000 .debug_loc
00000000 l    d  .debug_aranges 00000000 .debug_aranges
00000000 l    d  .debug_line    00000000 .debug_line
00000000 l    d  .debug_str     00000000 .debug_str
00000000 l    d  .comment       00000000 .comment
00000000 l    d  .ARM.attributes        00000000 .ARM.attributes
00000000 l    d  .debug_frame   00000000 .debug_frame
00000000 l    df *ABS*  00000000 startup.c
08000000 l     O .text  00000018 _vector_table
00000000 l    df *ABS*  00000000 alarmManager.c
20000020 l       .bss   00000000 ALARM_MANAGER_curState
20000028 l       .bss   00000000 alarmTime
00000000 l    df *ABS*  00000000 buzzerDriver.c
20000004 l     O .data  00000001 Buzzer_curState
00000000 l    df *ABS*  00000000 driver.c
00000000 l    df *ABS*  00000000 ledDriver.c
```

```
2000000c l     O .data  00000001 LED_curState
00000000 l    df *ABS*  00000000 main.c
00000000 l    df *ABS*  00000000 mainControl.c
20000038 l       .bss   00000000 MC_curState
2000003c l       .bss   00000000 pressure
00000000 l    df *ABS*  00000000 pressureSensorDriver.c
20000040 l       .bss   00000000 PS_curState
00000000 l    df *ABS*  00000000
08000000 g       .text  00000000 _TEXT_START
20000030 g     O .bss   00000004 p
080001fc g     F .text  000000a4 GPIO_INITIALIZATION
080000f4 g     F .text  0000001c state_define_E_buzzerOff
080004e8 g     F .text  0000000a _Vector_deafultHandler
080002bc g     F .text  0000001c state_define_E_ledOn
080000b0 g     F .text  00000042 Signal_onHighPressureDetect
080002f4 g     F .text  0000001c Signal_turnOffLed
080004e8  w    F .text  0000000a _Vector_MemManage
08000110 g     F .text  0000001c state_define_E_buzzerOn
08000400 g     F .text  0000003c state_define_E_PS_update
20000018 g     O .data  00000004 PS_curCall
080004e8  w    F .text  0000000a _Vector_NMI
20000008 g     O .data  00000004 Buzzer_curCall
0800012c g     F .text  0000001c Signal_turnOnBuzzer
```

```
20000014 g     O .data  00000004 MC_curCall
080004e8  w    F .text  0000000a _Vector_UsageFault
08000358 g     F .text  0000002a state_define_E_MC_init
08000164 g     F .text  00000048 getPressureVal
080004e8  w    F .text  0000000a _Vector_HardFault
080003b4 g     F .text  0000001e Signal_readPressureSensor
20000000 g       .data  00000000 _DATA_START
080003d4 g     F .text  0000002a state_define_E_PS_init
080004f4 g       .text  00000000 _TEXT_END
08000034 g     F .text  0000007c state_define_E_alarmOn
080001ac g     F .text  00000050 Set_Alarm_actuator
080004f8 g       .rodata        00000000 _RODATA_END
080004f4 g     O .rodata        00000004 threshld
08000310 g     F .text  00000046 main
20000044 g       .bss   00000000 _BSS_END
20000034 g     O .bss   00000004 count
080002d8 g     F .text  0000001c Signal_turnOnLed
20001044 g       .bss   00000000 _STACK_TOP
08000384 g     F .text  00000030 state_define_E_MC_update
080004e8  w    F .text  0000000a _Vector_BusFault
0800043c g     F .text  000000aa _reset
20000000 g     O .data  00000004 ALARM_MANAGER_curCall
080004f4 g       .rodata        00000000 _RODATA_START
```

```
08000148 g     F .text  0000001c Signal_turnOffBuzzer
2000001c g       .data  00000000 _DATA_END
08000018 g     F .text  0000001c state_define_E_alarmOff
080002a0 g     F .text  0000001c state_define_E_ledOff
20000010 g     O .data  00000004 LED_curCall
20000020 g       .bss   00000000 _BSS_START
```

10. Map File

```
Memory Configuration

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

```
.text           0x08000000      0x4f4
                0x08000000                _TEXT_START = .
 *(.vectors*)
 .vectors       0x08000000      0x18 startup.o
 *(.text*)
```

```
.rodata         0x080004f4        0x4
                0x080004f4                _RODATA_START = .
 *(.rodata*)
 .rodata        0x080004f4        0x4 mainControl.o
                0x080004f4                threshld
                0x080004f8                . = ALIGN (0x4)
                0x080004f8                _RODATA_END = .
```

```
*(.data*)
.data           0x20000000        0x4 alarmManager.o
                0x20000000                ALARM_MANAGER_curCall
.data           0x20000004        0x8 buzzerDriver.o
                0x20000008                Buzzer_curCall
.data           0x2000000c        0x0 driver.o
.data           0x2000000c        0x8 ledDriver.o
                0x20000010                LED_curCall
.data           0x20000014        0x0 main.o
.data           0x20000014        0x4 mainControl.o
                0x20000014                MC_curCall
.data           0x20000018        0x4 pressureSensorDriver.o
                0x20000018                PS_curCall
.data           0x2000001c        0x0 startup.o
                0x2000001c                . = ALIGN (0x4)
                0x2000001c                _DATA_END = .
```

```
.bss            0x20000020     0x1024 load address 0x08000518
                0x20000020                _BSS_START = .
*(.bss*)
.bss            0x20000020       0x10 alarmManager.o
.bss            0x20000030        0x0 buzzerDriver.o
.bss            0x20000030        0x8 driver.o
                0x20000030                p
                0x20000034                count
.bss            0x20000038        0x0 ledDriver.o
.bss            0x20000038        0x0 main.o
.bss            0x20000038        0x8 mainControl.o
.bss            0x20000040        0x1 pressureSensorDriver.o
.bss            0x20000041        0x0 startup.o
*(COMMON)
                0x20000044                . = ALIGN (0x4)
```

Thank You