



Pressure Detection System

1 NOV 2025

Eng - Belal Hani Abu Sabha

Learn in Depth Company

Profile:

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

Github:

https://github.com/belalsabha/Master_Embedded_Systems

Linkedin:

<https://www.linkedin.com/in/belal-sabha-06ba33349/>

Content	1
OverView and Case Study	2
Method	3
Requirements	4
Space Exploration	5
System Analysis	6
System Design	9
Implementation C Code	13
Symbols & Sections	22
Mabfile.map	26
Simulation	28
Conclusion	30

Overview

The Pressure Controller *System* is an embedded control system designed to monitor pressure in Room . The system continuously reads data from a pressure sensor detect if the value is more or less than threshold to control in alarm status .

The project demonstrates a complete embedded system design cycle followed by the implementation of modular C code and hardware-level integration on the STM32F103C6 microcontroller.

Case Study :

1-Specifications

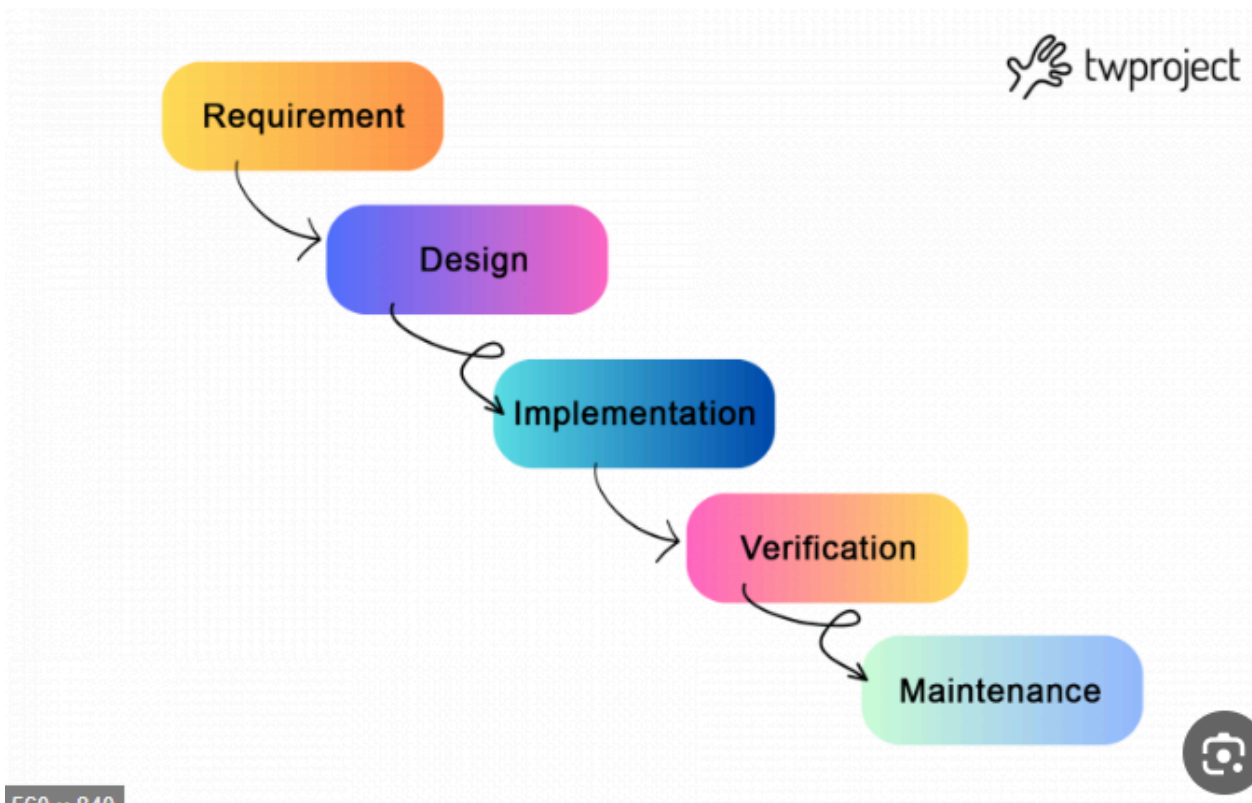
1. A pressure controller informs the crew of a cabin when the pressure exceeds 20 bars in the cabin
2. The alarm duration equals 60 seconds.

2-Assumptions :

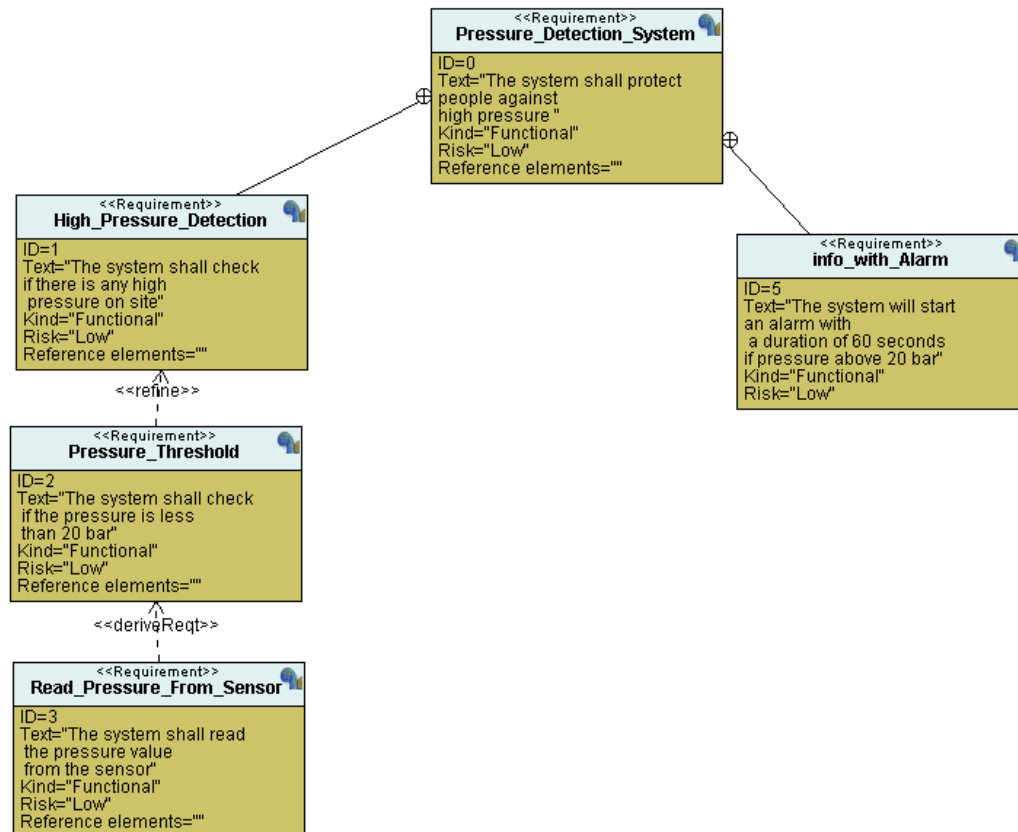
- 1-The controller set up and shutdown procedures are not modeled
- 2-The controller maintenance is not modeled
- 3-The pressure sensor never fails
- 4-The alarm never fails
- 5-The controller never faces power cut

Method :

I. we will be working on the waterfall method

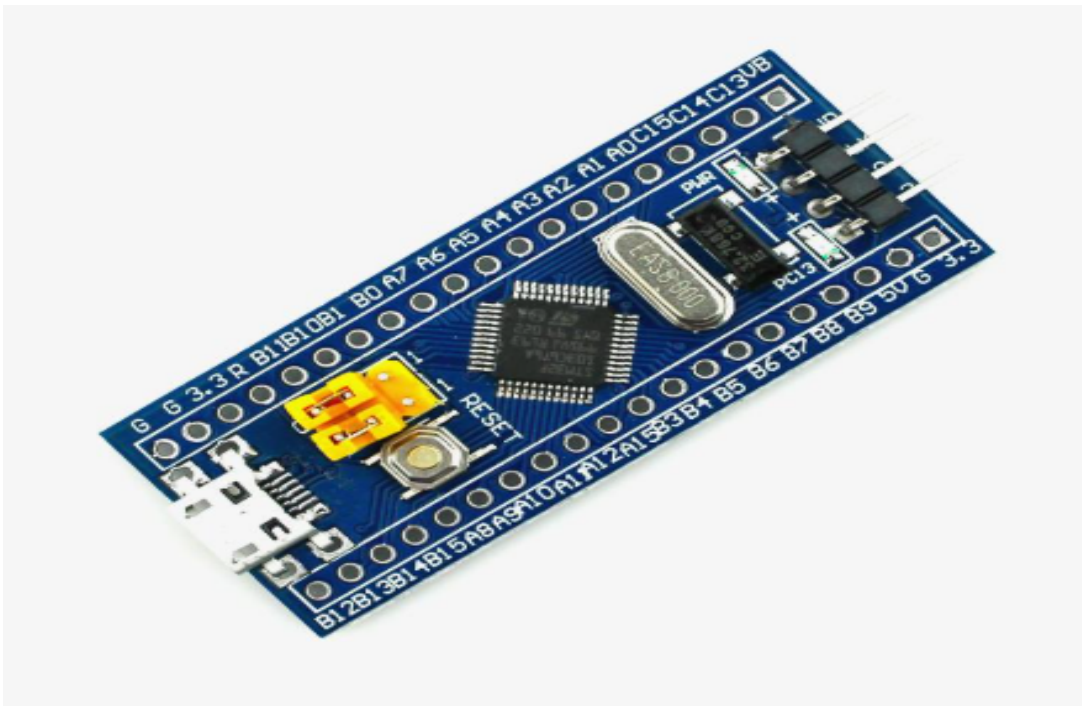


Requirements :



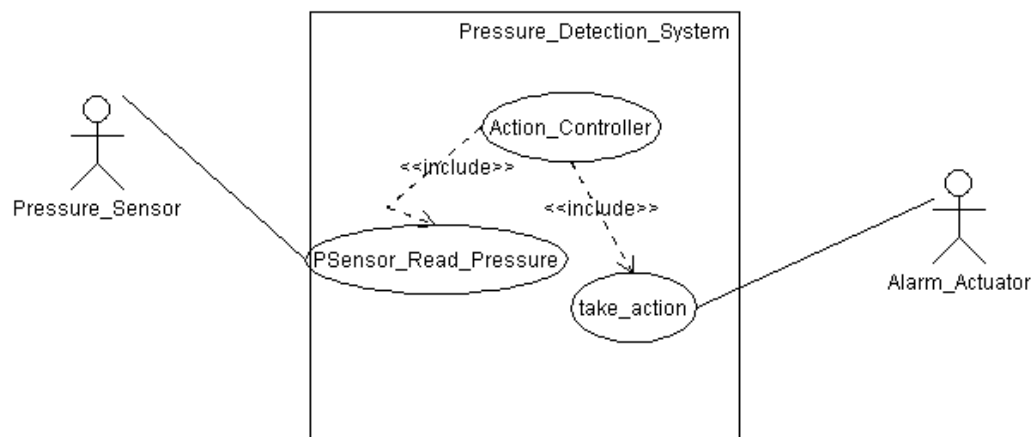
Space Exploration :

1-Use STM32F103C6 Microcontroller with Cortex-M3

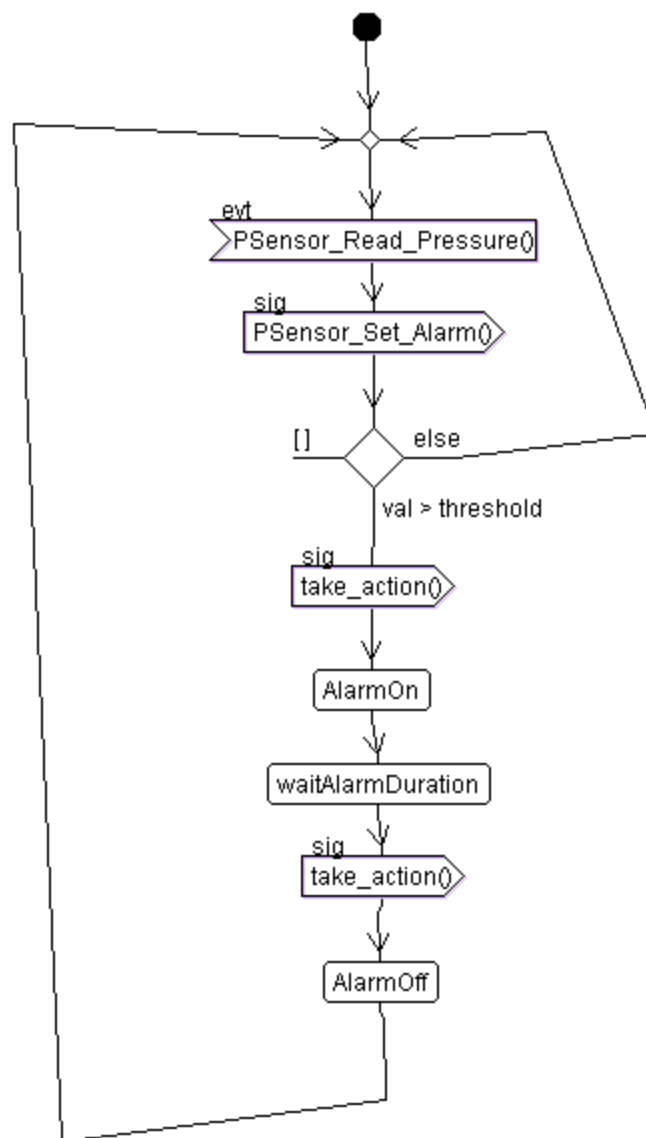


System Analysis :

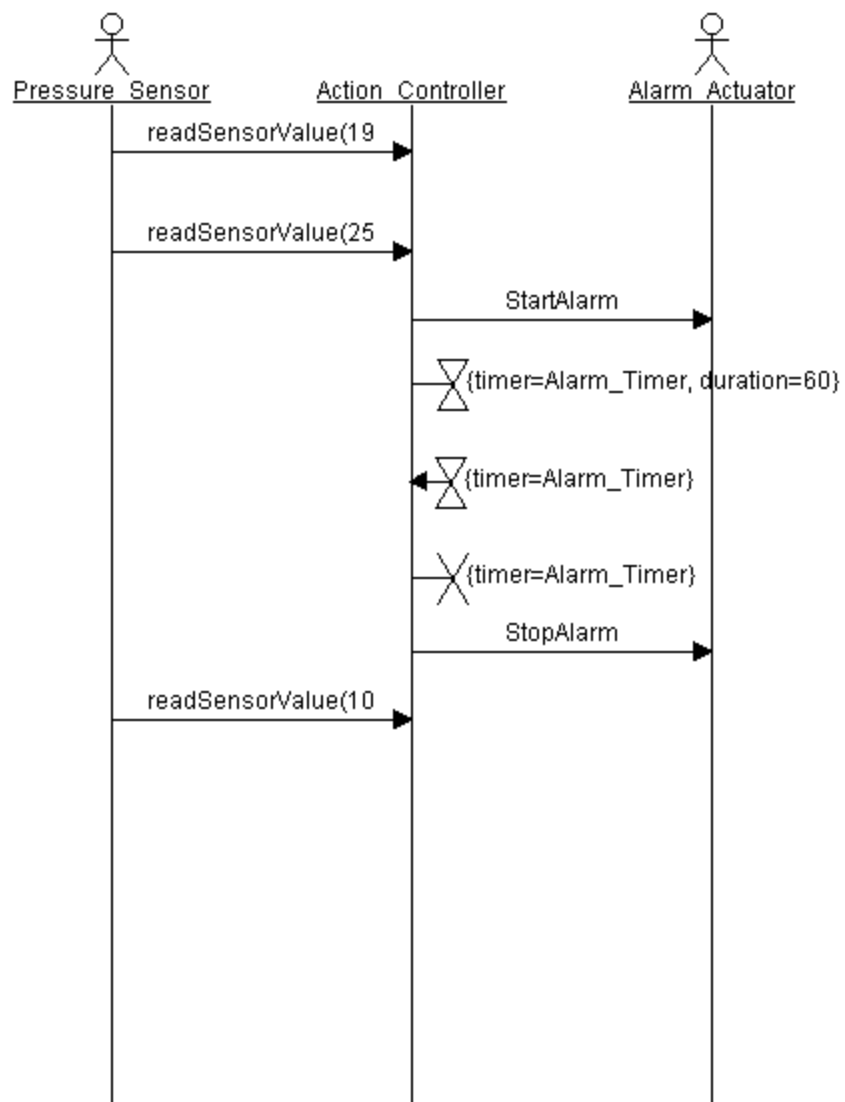
1-Use-Case Diagram :



2-Activity Diagram :

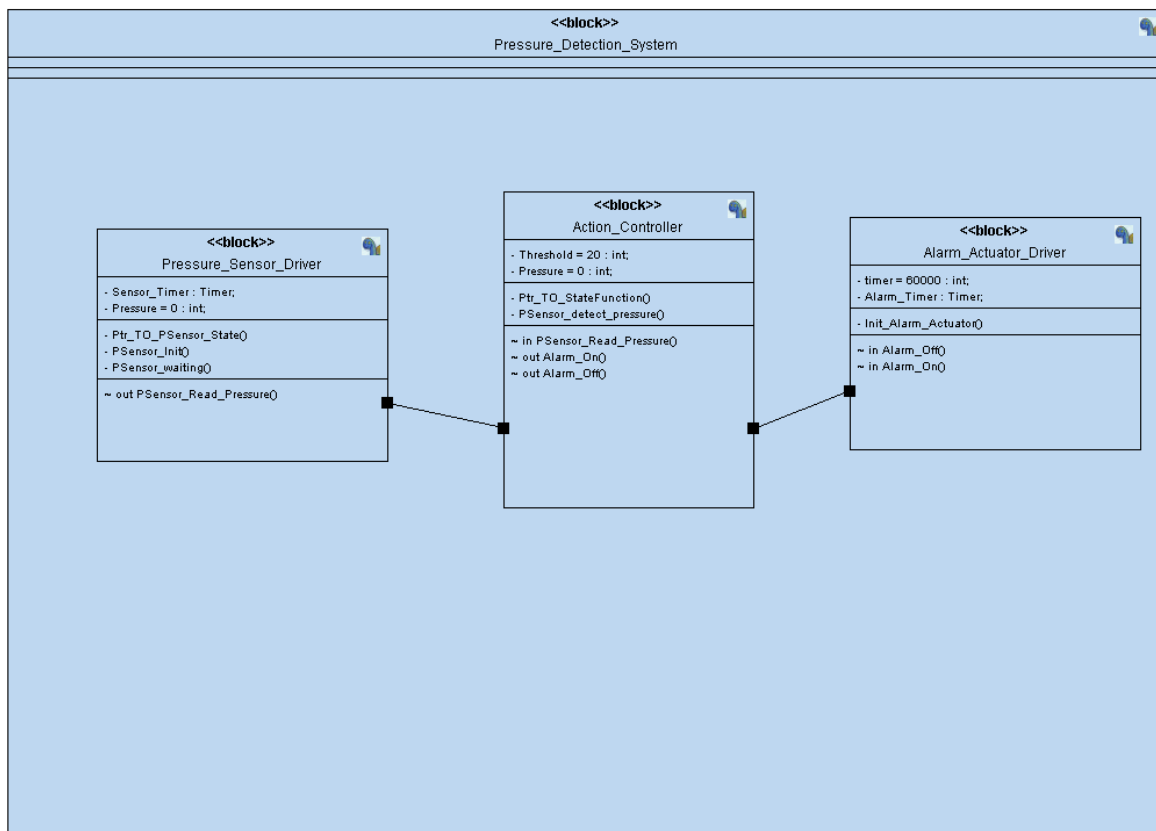


3-Sequence Diagram :



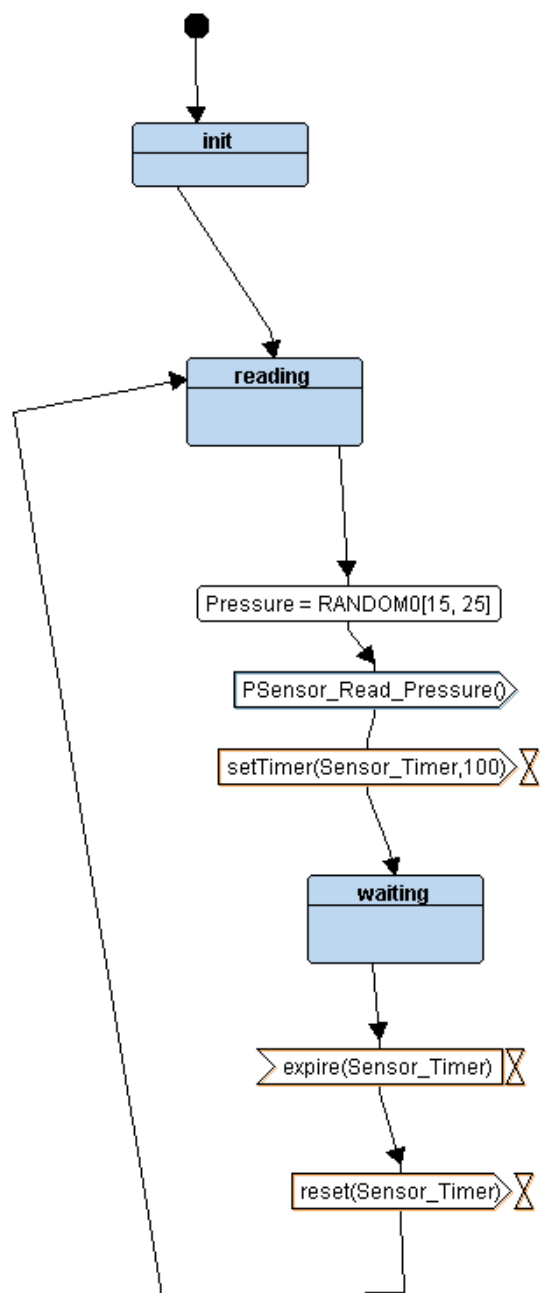
System Design :

1-Block Diagram :

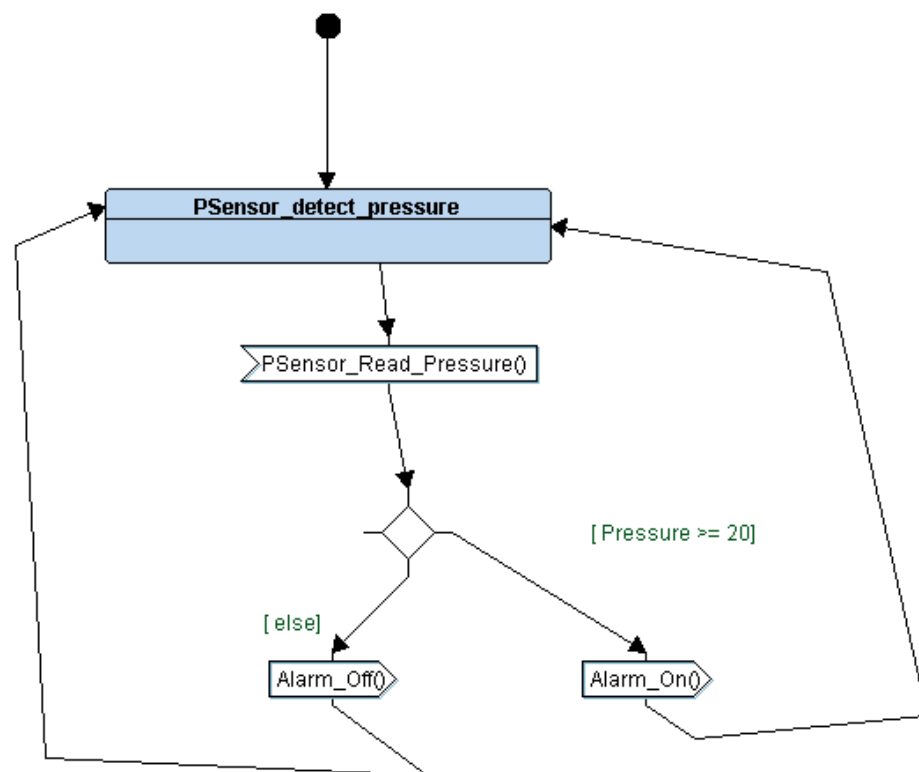


2-State Machine for every Module :

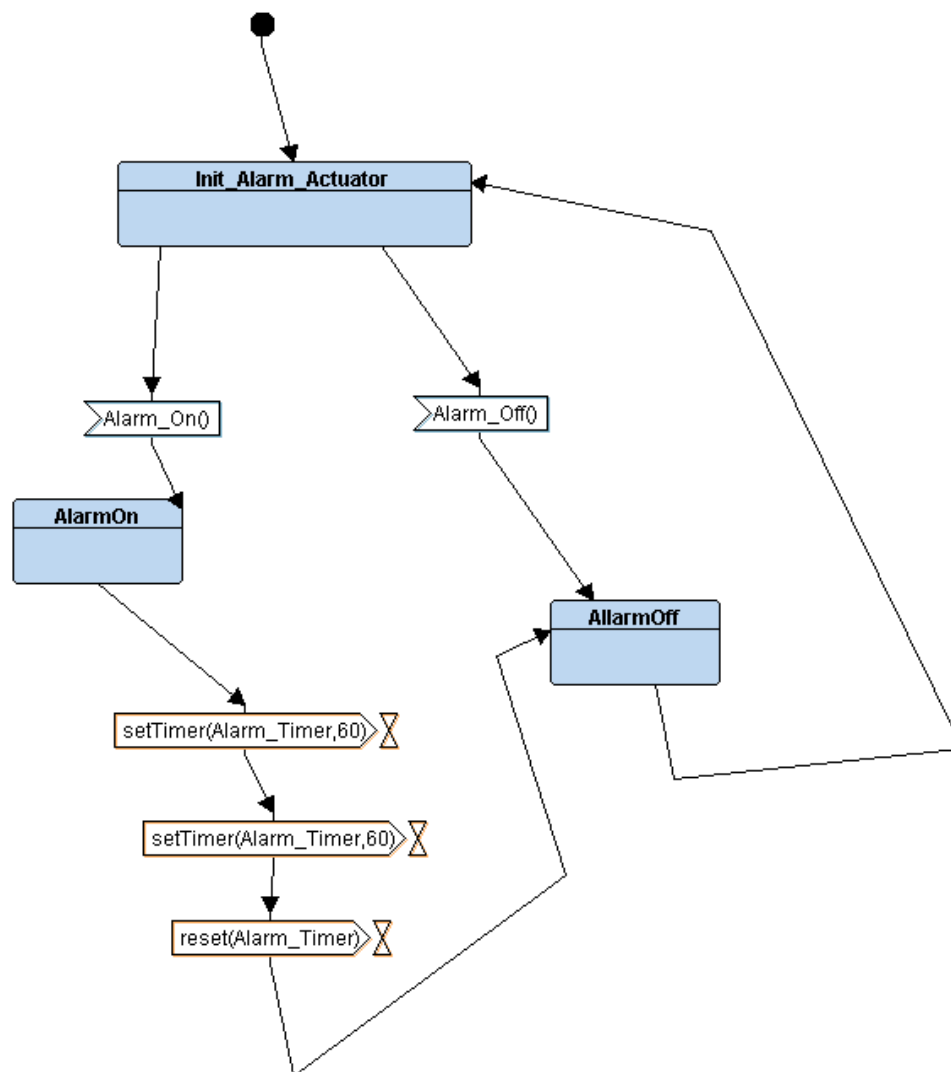
A-Pressure_Sensor_Driver :



B-Action_Controller :



C-Alarm_Actuator_Driver :



Implementation C Code :

Pressure_Sensor_Driver.c

```
Pressure_Sensor_Driver.c  x
/*
 * Pressure_Sensor_Driver.c
 *
 * Created on: 31 Oct 2025
 * Author: Belal
 */

#include "Pressure_Sensor_Driver.h"
#include "driver.h"
unsigned int Pressure= 0 ;
void (*Ptr_TO_PSensor_State)();

void PSensor_Init(){
    Ptr_TO_PSensor_State=PSensor_Read_Pressure;
}
void PSensor_Read_Pressure(){
    Pressure= (unsigned int)getPressureVal();
    PSensor_waiting();
    Ptr_TO_PSensor_State=PSensor_Read_Pressure;
}

void PSensor_waiting(){
    int delay=1000;
    Delay(delay);
}
```

Pressure_Sensor_Driver.h

File Edit Selection Find View Goto Tools Project Preferences Help

```
Pressure_Sensor_Driver.c x Pressure_Sensor_Driver.h x
1  /*
2   * Pressure_Sensor_Driver.h
3   *
4   * Created on: 31 Oct 2025
5   * Author: Belal
6   */
7
8  #ifndef PRESSURE_SENSOR_DRIVER_H_
9  #define PRESSURE_SENSOR_DRIVER_H_
10
11  void PSensor_Init();
12  void PSensor_waiting();
13  void PSensor_Read_Pressure();
14  extern unsigned int Pressure;
15  extern void (*Ptr_TO_PSensor_State)();
16
17
18
19
20
21
22
23
24  #endif /* PRESSURE_SENSOR_DRIVER_H_ */
25
26
27
```

Alarm_Actuator_Driver.c

```

1  /*
2  * Alarm_Actuator_Driver.c
3  *
4  * Created on: 31 Oct 2025
5  * Author: Belal
6  */
7  #include "Alarm_Actuator_Driver.h"
8  #include "driver.h"
9  unsigned int time=600000;
10
11 void Alarm_Off(){
12     Set_Alarm_actuator(1);
13 }
14 void Alarm_On(){
15     Set_Alarm_actuator(0);
16     Delay(time);
17     Alarm_Off();
18 }
19 void Init_Alarm_Actuator(){
20
21 }
22
23
24

```

Alarm_Actuator_Driver.h

```

1  /*
2  * Alarm_Actuator_Driver.h
3  *
4  * Created on: 31 Oct 2025
5  * Author: Belal
6  */
7
8  #ifndef ALARM_ACTUATOR_DRIVER_H_
9  #define ALARM_ACTUATOR_DRIVER_H_
10
11 #include "driver.h"
12 void Alarm_Off();
13 void Alarm_On();
14 void Init_Alarm_Actuator();
15 extern void (*Ptr_TO_Alarm_State);
16
17
18 #endif /* ALARM_ACTUATOR_DRIVER_H_ */
19
20

```


Action_Controller.c

```
1  /*
2   * Action_Controller.c
3   *
4   * Created on: 31 Oct 2025
5   * Author: Belal
6   */
7
8  #include "Action_Controller.h"
9  #include "Pressure_Sensor_Driver.h"
10 #include "Alarm_Actuator_Driver.h"
11
12
13 static const unsigned int Threshold = 20;
14
15 void (*Ptr_TO_StateFunction)();
16
17
18 void PSensor_detect_pressure(){
19
20     if(Pressure < Threshold )
21         Alarm_Off();
22     else
23         Alarm_On();
24
25 }
26
27
```

Action_Controller.h

```
1
2  * Action_Controller.h
3  *
4  * Created on: 31 Oct 2025
5  * Author: Belal
6  */
7
8  #ifndef ACTION_CONTROLLER_H_
9  #define ACTION_CONTROLLER_H_
10 #include "Pressure_Sensor_Driver.h"
11 #include "Alarm_Actuator_Driver.h"
12 extern void (*Ptr_TO_StateFunction)();
13 void set_alarm();
14 void reset_alarm();
15 void PSensor_detect_pressure();
16
17
18 #endif /* ACTION_CONTROLLER_H_ */
19
20
```

Main.c

```
1  /*
2  * main.c
3  *
4  * Created on: 31 Oct 2025
5  * Author: Belal
6  */
7
8
9  #include <stdint.h>
10 #include <stdio.h>
11
12 #include "driver.h"
13 #include "Pressure_Sensor_Driver.h"
14 #include "Action_Controller.h"
15 #include "Alarm_Actuator_Driver.h"
16
17
18 void setup(){
19     Ptr_TO_StateFunction=PSensor_detect_pressure;
20     Ptr_TO_PSensor_State=PSensor_Read_Pressure;
21     GPIO_INITIALIZATION();
22 }
23
24
25 int main (){
26     setup();
27     while (1)
28     {
29         Ptr_TO_PSensor_State();
30         Ptr_TO_StateFunction();
31
32
33
34     }
35 }
36
37
38
```

Driver.c

```
driver.c
1  #include "driver.h"
2  #include <stdint.h>
3  #include <stdio.h>
4  void Delay(int nCount)
5  {
6      for(; nCount != 0; nCount--);
7  }
8
9  int getPressureVal(){
10     return (GPIOA_IDR & 0xFF);
11 }
12
13 void Set_Alarm_actuator(int i){
14     if (i == 1){
15         SET_BIT(GPIOA_ODR,13);
16     }
17     else if (i == 0){
18         RESET_BIT(GPIOA_ODR,13);
19     }
20 }
21
22 void GPIO_INITIALIZATION (){
23     SET_BIT(APB2ENR, 2);
24     GPIOA_CRL &= 0xFF0FFFFFFF;
25     GPIOA_CRL |= 0x00000000;
26     GPIOA_CRH &= 0xFF0FFFFFFF;
27     GPIOA_CRH |= 0x22222222;
28 }
29
```

Driver.h

```
driver.c  x  driver.h  x

#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 ();
```

LinkerScript.ld

```
1
2  /*
3   * linker.ld
4   *
5   * Created on: 31 Oct 2025
6   * Author: Belal
7   */
8
9  ▼ MEMORY {
10     flash(rx) : ORIGIN = 0x08000000 , LENGTH = 128k
11     sram(rwx) : ORIGIN = 0x20000000 , LENGTH = 20k
12 }
13
14 ▼ SECTIONS {
15     ▼ .text : {
16         *(.vectors)
17         *(.text)
18         *(.rodata)
19         End_Text = . ;
20     }>flash
21
22     ▼ .data : {
23         Start_Data = . ;
24         *(.data)
25         End_Data = . ;
26     }>sram AT>flash
27
28     ▼ .bss : {
29         Start_Bss = . ;
30         *(.bss)
31         ▼ . = ALIGN(4) ;
32         End_Bss = . ;
33         . = ALIGN(4) ;
34         . = . + 0x1000; ;
35         Stack_Top = . ;
36     }>sram
37 }
```

Startup.c

```

1  /*
2   * @file      startup.c
3   * Created on: 31 Oct 2025
4   * Author: Belal
5   */
6  typedef unsigned int uint32_t ;
7  typedef unsigned char uint8_t;
8  #define Start_Stack_SP 0x20001000
9  extern int main(void);
10 void reset_handler(void);
11 void deaful_handler(void);
12 void NMI_handler(void)          __attribute__((weak, alias("deaful_handler")));
13 void H_fault_handler(void)      __attribute__((weak, alias("deaful_handler")));
14 void MM_fault_handler(void)     __attribute__((weak, alias("deaful_handler")));
15 void Bus_fault_(void)          __attribute__((weak, alias("deaful_handler")));
16 void Usage_fault_handler(void)  __attribute__((weak, alias("deaful_handler")));
17 uint32_t vectors[] __attribute__((section(".vectors"))) = {
18     Start_Stack_SP,
19     (uint32_t)&reset_handler,
20     (uint32_t)&deaful_handler,
21     (uint32_t)&NMI_handler,
22     (uint32_t)&H_fault_handler,
23     (uint32_t)&MM_fault_handler,
24     (uint32_t)&Bus_fault_,
25     (uint32_t)&Usage_fault_handler,
26 };
27 extern uint32_t End_Text ;
28 extern uint32_t Start_Data;
29 extern uint32_t End_Data;
30 extern uint32_t Start_Bss;
31 extern uint32_t End_Bss;
32 extern uint32_t Stack_Top;
33 void reset_handler(void){
34     // copying data from flash to sram
35     uint32_t Data_Size = (uint8_t*)&End_Data - (uint8_t*)&Start_Data;
36     uint8_t *Src = (uint8_t*)&End_Text;
37     uint8_t *Des = (uint8_t*)&Start_Data;
38     for (int i = 0; i < Data_Size; ++i)
39     {
40         *((uint8_t*)Des++) = *((uint8_t*)Src++);
41     }
42     // create .bss section and init with zero
43     uint32_t Bss_Size = (uint8_t*)&End_Bss - (uint8_t*)&Start_Bss;
44     Des = (uint8_t*)&Start_Bss;
45     for (int i = 0; i < Bss_Size; ++i)
46     {
47         *Des++ = 0;
48     }
49     // go to main in main.c file
50     main();
51 }
52 void deaful_handler(void){
53     reset_handler();
54 }

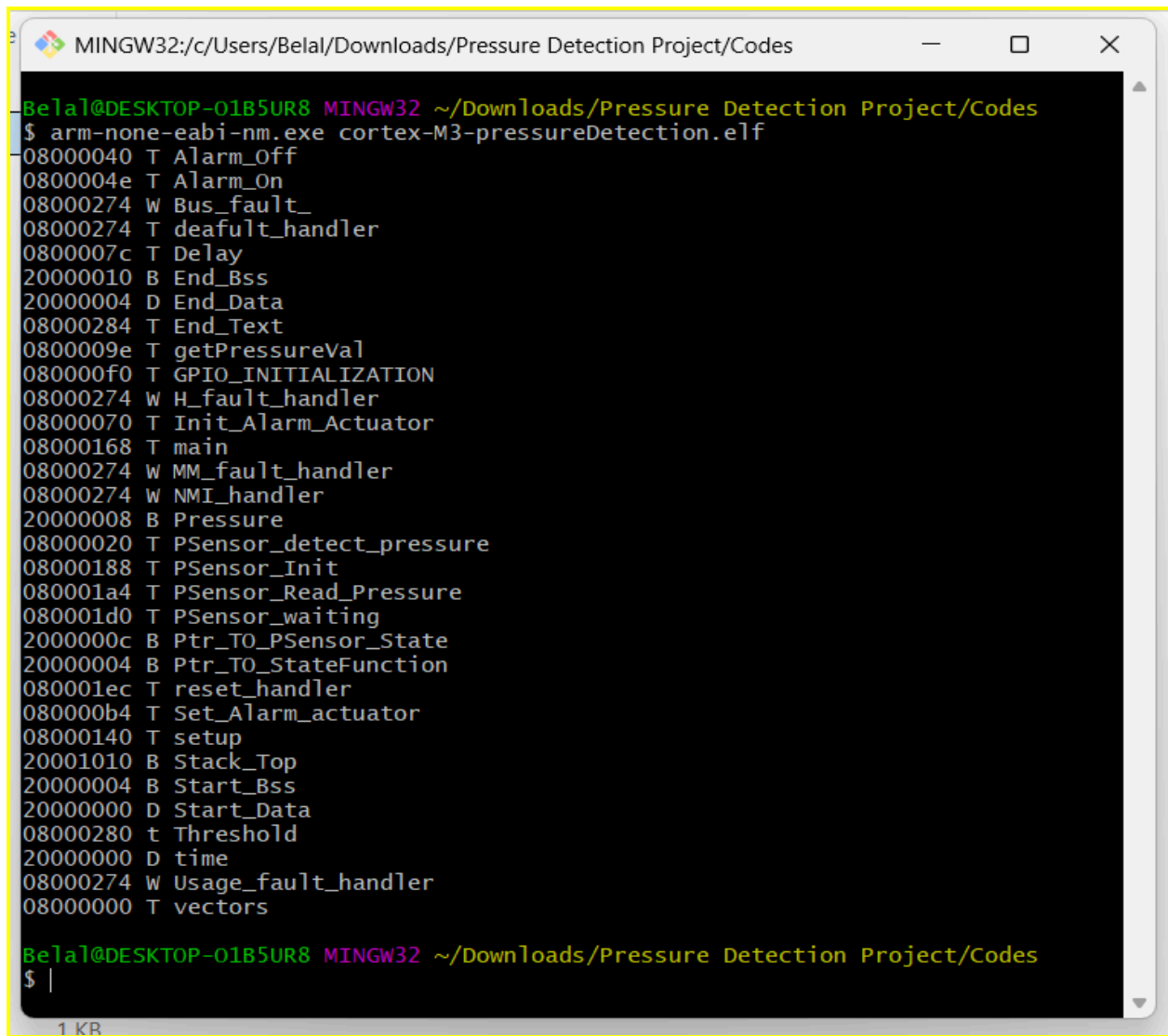
```

Makefile

```
1
2  #@copyright : Belal Hani Sabha
3
4  CC=arm-none-eabi-
5
6  CFLAGS=-gdwarf-2 -mcpu=cortex-m3
7
8  INCS=-I .
9
10 LIBS=
11
12 SRC=$(wildcard *.c)
13 OBJ=$(SRC:.c=.o)
14
15 AS=$(wildcard *.s)
16 ASO=$(AS:.s=.o)
17
18
19
20
21
22 PROJECT_NAME=cortex-M3-pressureDetection
23
24 all: $(PROJECT_NAME).bin
25     @echo "build is done"
26
27 %.o: %.c
28     $(CC)gcc.exe -c $(CFLAGS) $(INCS) $< -o $@
29
30 $(PROJECT_NAME).elf: $(OBJ) $(ASO)
31     $(CC)ld.exe -T LinkerScript.ld $(LIBS) $(OBJ) $(ASO) -o $@ -Map=mab_file.map
32
33 $(PROJECT_NAME).bin: $(PROJECT_NAME).elf
34     $(CC)objcopy.exe -O binary $< $@
35
36 clean_all:
37     rm *.o *.elf *.bin *.map
38     @echo "everything clean..."
39
40 clean:
41     rm *.bin *.elf
42     @echo "clean .elf and .bin files"
```

Symbols & Sections :

cortex-M3-pressureDetection.elf



```
MINGW32:/c/Users/Belal/Downloads/Pressure Detection Project/Codes
Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Pressure Detection Project/Codes
$ arm-none-eabi-nm.exe cortex-M3-pressureDetection.elf
08000040 T Alarm_Off
0800004e T Alarm_On
08000274 W Bus_fault_
08000274 T deaault_handler
0800007c T Delay
20000010 B End_Bss
20000004 D End_Data
08000284 T End_Text
0800009e T getPressureVal
080000f0 T GPIO_INITIALIZATION
08000274 W H_fault_handler
08000070 T Init_Alarm_Actuator
08000168 T main
08000274 W MM_fault_handler
08000274 W NMI_handler
20000008 B Pressure
08000020 T PSensor_detect_pressure
08000188 T PSensor_Init
080001a4 T PSensor_Read_Pressure
080001d0 T PSensor_waiting
2000000c B Ptr_TO_PSensor_State
20000004 B Ptr_TO_StateFunction
080001ec T reset_handler
080000b4 T Set_Alarm_actuator
08000140 T setup
20001010 B Stack_Top
20000004 B Start_Bss
20000000 D Start_Data
08000280 t Threshold
20000000 D time
08000274 W Usage_fault_handler
08000000 T vectors

Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Pressure Detection Project/Codes
$ |
```

1 KB


```

MINGW32:/c/Users/Belal/Downloads/Pressure Detection Project/Codes
08000280 t Threshold
20000000 D time
08000274 w Usage_fault_handler
08000000 T vectors

Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Pressure Detection Project/Codes
$ arm-none-eabi-objdump.exe -h cortex-M3-pressureDetection.elf

cortex-M3-pressureDetection.elf:      file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text          00000284  08000000  08000000  00010000  2**2
    CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data           00000004  20000000  08000284  00020000  2**2
    CONTENTS, ALLOC, LOAD, DATA
  2 .bss            0000100c  20000004  08000288  00020004  2**2
    ALLOC
  3 .debug_info     00000610  00000000  00000000  00020004  2**0
    CONTENTS, READONLY, DEBUGGING, OCTETS
  4 .debug_abbrev   000003dc  00000000  00000000  00020614  2**0
    CONTENTS, READONLY, DEBUGGING, OCTETS
  5 .debug_loc      0000039c  00000000  00000000  000209f0  2**0
    CONTENTS, READONLY, DEBUGGING, OCTETS
  6 .debug_aranges  000000c0  00000000  00000000  00020d8c  2**0
    CONTENTS, READONLY, DEBUGGING, OCTETS
  7 .debug_line     00000403  00000000  00000000  00020e4c  2**0
    CONTENTS, READONLY, DEBUGGING, OCTETS
  8 .debug_str      000002b6  00000000  00000000  0002124f  2**0
    CONTENTS, READONLY, DEBUGGING, OCTETS
  9 .comment        00000049  00000000  00000000  00021505  2**0
    CONTENTS, READONLY
10 .ARM.attributes 0000002d  00000000  00000000  0002154e  2**0
    CONTENTS, READONLY
11 .debug_frame     0000023c  00000000  00000000  0002157c  2**2
    CONTENTS, READONLY, DEBUGGING, OCTETS

Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Pressure Detection Project/Codes
$

```

1 KB

Action_Controller.c

```

MINGW32:/c:/Users/Belal/Downloads/Src
Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Src
$ arm-none-eabi-nm.exe Action_Controller.o
                 U Alarm_Off
                 U Alarm_On
                 U Pressure
00000000 T PSensor_detect_pressure
00000000 B Ptr_TO_StateFunction
00000000 r Threshold

Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Src
$ |

```

```

MINGW32:/c:/Users/Belal/Downloads/Src
Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Src
$ arm-none-eabi-objdump.exe -h Action_Controller.o

Action_Controller.o:      file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text          00000020  00000000  00000000  00000034  2**2
                 CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data          00000000  00000000  00000000  00000054  2**0
                 CONTENTS, ALLOC, LOAD, DATA
  2 .bss           00000004  00000000  00000000  00000054  2**2
                 ALLOC
  3 .rodata        00000004  00000000  00000000  00000054  2**2
                 CONTENTS, ALLOC, LOAD, READONLY, DATA
  4 .debug_info    000000de  00000000  00000000  00000058  2**0
                 CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  5 .debug_abbrev  00000091  00000000  00000000  00000136  2**0
                 CONTENTS, READONLY, DEBUGGING, OCTETS
  6 .debug_loc     0000002c  00000000  00000000  000001c7  2**0
                 CONTENTS, READONLY, DEBUGGING, OCTETS
  7 .debug_aranges 00000020  00000000  00000000  000001f3  2**0
                 CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  8 .debug_line    0000008a  00000000  00000000  00000213  2**0
                 CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  9 .debug_str     0000016a  00000000  00000000  0000029d  2**0
                 CONTENTS, READONLY, DEBUGGING, OCTETS
10 .comment       0000004a  00000000  00000000  00000407  2**0
                 CONTENTS, READONLY
11 .debug_frame   0000002c  00000000  00000000  00000454  2**2
                 CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
12 .ARM.attributes 0000002d  00000000  00000000  00000480  2**0
                 CONTENTS, READONLY

```

Alarm_Actuator_Driver.c

```

MINGW32:/c/Users/Belal/Downloads/Src
Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Src
$ arm-none-eabi-nm.exe Alarm_Actuator_Driver.o
00000000 T Alarm_Off
0000000e T Alarm_On
          U Delay
00000030 T Init_Alarm_Actuator
          U Set_Alarm_actuator
00000000 D time
Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Src
$ |

```

```

MINGW32:/c/Users/Belal/Downloads/Src
Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Src
$ arm-none-eabi-objdump.exe -h Alarm_Actuator_Driver.o

Alarm_Actuator_Driver.o:      file format elf32-littlearm

Sections:
Idx Name          Size      VMA           LMA           File off  Algn
  0 .text          0000003c  00000000  00000000  00000034  2**2
CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data          00000004  00000000  00000000  00000070  2**2
CONTENTS, ALLOC, LOAD, DATA
  2 .bss           00000000  00000000  00000000  00000074  2**0
ALLOC
  3 .debug_info     000000cf  00000000  00000000  00000074  2**0
CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  4 .debug_abbrev   0000006d  00000000  00000000  00000143  2**0
CONTENTS, READONLY, DEBUGGING, OCTETS
  5 .debug_loc      0000009c  00000000  00000000  000001b0  2**0
CONTENTS, READONLY, DEBUGGING, OCTETS
  6 .debug_aranges  00000020  00000000  00000000  0000024c  2**0
CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  7 .debug_line     00000060  00000000  00000000  0000026c  2**0
CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  8 .debug_str      0000015a  00000000  00000000  000002cc  2**0
CONTENTS, READONLY, DEBUGGING, OCTETS
  9 .comment        0000004a  00000000  00000000  00000426  2**0
CONTENTS, READONLY
10 .debug_frame    00000068  00000000  00000000  00000470  2**2
CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
11 .ARM.attributes 0000002d  00000000  00000000  000004d8  2**0
CONTENTS, READONLY

```

Pressure_Sensor_Driver.c

```

MINGW32:/c/Users/Belal/Downloads/Src
Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Src
$ arm-none-eabi-nm.exe Pressure_Sensor_Driver.o
                 U Delay
                 U getPressureVal
00000000 B Pressure
00000000 T PSensor_Init
0000001c T PSensor_Read_Pressure
00000048 T PSensor_waiting
00000004 B Ptr_TO_PSensor_State

```

```

Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Src
$ arm-none-eabi-objdump.exe -h Pressure_Sensor_Driver.o

Pressure_Sensor_Driver.o:      file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text          00000064  00000000  00000000  00000034  2**2
CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data           00000000  00000000  00000000  00000098  2**0
CONTENTS, ALLOC, LOAD, DATA
  2 .bss            00000008  00000000  00000000  00000098  2**2
ALLOC
  3 .debug_info     00000115  00000000  00000000  00000098  2**0
CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  4 .debug_abbrev   000000bc  00000000  00000000  000001ad  2**0
CONTENTS, READONLY, DEBUGGING, OCTETS
  5 .debug_loc      000000c0  00000000  00000000  00000269  2**0
CONTENTS, READONLY, DEBUGGING, OCTETS
  6 .debug_aranges  00000020  00000000  00000000  00000329  2**0
CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  7 .debug_line     0000008d  00000000  00000000  00000349  2**0
CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  8 .debug_str      00000186  00000000  00000000  000003d6  2**0
CONTENTS, READONLY, DEBUGGING, OCTETS
  9 .comment        0000004a  00000000  00000000  0000055c  2**0
CONTENTS, READONLY
10 .debug_frame    00000070  00000000  00000000  000005a8  2**2
CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
11 .ARM.attributes 0000002d  00000000  00000000  00000618  2**0
CONTENTS, READONLY

```

Main.c

```

MINGW32:/c/Users/Belal/Downloads/Src
Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Src
$ arm-none-eabi-nm.exe main.o
00000028 T main
          U GPIO_INITIALIZATION
          U PSensor_detect_pressure
          U PSensor_Read_Pressure
          U Ptr_TO_PSensor_State
          U Ptr_TO_StateFunction
00000000 T setup

```

```

MINGW32:/c/Users/Belal/Downloads/Src
Belal@DESKTOP-01B5UR8 MINGW32 ~/Downloads/Src
$ arm-none-eabi-objdump.exe -h main.o

main.o:      file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text          00000048  00000000  00000000  00000034  2**2
             CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data           00000000  00000000  00000000  0000007c  2**0
             CONTENTS, ALLOC, LOAD, DATA
  2 .bss            00000000  00000000  00000000  0000007c  2**0
             ALLOC
  3 .debug_info     000000d3  00000000  00000000  0000007c  2**0
             CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  4 .debug_abbrev   00000084  00000000  00000000  0000014f  2**0
             CONTENTS, READONLY, DEBUGGING, OCTETS
  5 .debug_loc      00000058  00000000  00000000  000001d3  2**0
             CONTENTS, READONLY, DEBUGGING, OCTETS
  6 .debug_aranges  00000020  00000000  00000000  0000022b  2**0
             CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  7 .debug_line     0000008e  00000000  00000000  0000024b  2**0
             CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
  8 .debug_str      00000152  00000000  00000000  000002d9  2**0
             CONTENTS, READONLY, DEBUGGING, OCTETS
  9 .comment        0000004a  00000000  00000000  0000042b  2**0
             CONTENTS, READONLY
10 .debug_frame    00000048  00000000  00000000  00000478  2**2
             CONTENTS, RELOC, READONLY, DEBUGGING, OCTETS
11 .ARM.attributes 0000002d  00000000  00000000  000004c0  2**0
             CONTENTS, READONLY

```

mabfile.map :

```

C:\Users\Belaf\Downloads\Src\mab_file.map - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
LinkerScriptId startup.c Makefile mab_file.map
1
2 Memory Configuration
3
4 Name Origin Length Attributes
5 flash 0x00000000 0x00020000 xrw
6 sram 0x20000000 0x00005000 xrw
7 *default* 0x00000000 0xffffffff
8
9 Linker script and memory map
10
11
12 .text 0x00000000 0x2B4
13 *(.vectors)
14 .vectors 0x00000000 0x20 startup.o
15 *(.text)
16
17 .text 0x00000020 0x20 Action_Controller.o
18 0x00000020 PSensor_detect_pressure
19 0x00000040 0x3c Alarm_Actuator_Driver.o
20 0x00000040 Alarm_Off
21 0x00000040 Alarm_On
22 0x00000070 Init_Alarm_Actuator
23 .text 0x0000007c 0xc4 driver.o
24 0x0000007c Delay
25 0x00000080 getPressureVal
26 0x00000084 Set_Alarm_actuator
27 0x000000f0 GPIO_INITIALIZATION
28 .text 0x00000140 0x48 main.o
29 0x00000140 setup
30 0x00000180 main
31 .text 0x00000188 0x64 Pressure_Sensor_Driver.o
32 0x00000188 PSensor_Init
33 0x000001a4 PSensor_Read_Pressure
34 0x000001d0 PSensor_Waiting
35 .text 0x000001e0 0x04 startup.o
36 0x000001ec reset_handler
37 0x00000274 Bus_fault_
38 0x00000274 default_handler
39 0x00000274 Usage_fault_handler
40 0x00000274 M1_fault_handler
41 0x00000274 H_fault_handler
42 0x00000274 HW_handler
43 *(.rodata)
44 .rodata 0x00000280 0x4 Action_Controller.o
45 0x00000284 End_Text = .
46
47 .glue_7 0x00000284 0x0
48 .glue_7 0x00000284 0x0 linker stubs
49
50 .glue_7t 0x00000284 0x0
51 .glue_7t 0x00000284 0x0 linker stubs
52
53 .vfp11_veneer 0x00000284 0x0
54 .vfp11_veneer 0x00000284 0x0 linker stubs
55

```

```

C:\Users\Belaf\Downloads\Src\mab_file.map - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
LinkerScriptId startup.c Makefile mab_file.map
64
65 .data 0x20000000 0x4 load address 0x00000284
66 0x20000000 Start_Data = .
67 *(.data)
68 .data 0x20000000 0x0 Action_Controller.o
69 0x20000000 0x4 Alarm_Actuator_Driver.o
70 0x20000000 Time
71 .data 0x20000004 0x0 driver.o
72 .data 0x20000004 0x0 main.o
73 .data 0x20000004 0x0 Pressure_Sensor_Driver.o
74 .data 0x20000004 0x0 startup.o
75 0x20000004 End_Data = .
76
77 .igot.plt 0x20000004 0x0 load address 0x00000288
78 .igot.plt 0x20000004 0x0 Action_Controller.o
79
80 .bss 0x20000004 0x100c load address 0x00000288
81 0x20000004 Start_Bss = .
82 *(.bss)
83 .bss 0x20000004 0x4 Action_Controller.o
84 0x20000004 Ptc_To_StateFunction
85 .bss 0x20000008 0x0 Alarm_Actuator_Driver.o
86 .bss 0x20000008 0x0 driver.o
87 .bss 0x20000008 0x0 main.o
88 .bss 0x20000008 0x0 Pressure_Sensor_Driver.o
89 0x20000008 Pressure
90 0x2000000c Ptc_To_PSensor_State
91 .bss 0x20000010 0x0 startup.o
92 0x20000010 . = ALIGN (0x4)
93 0x20000010 End_Bss = .
94 0x20000010 . = ALIGN (0x4)
95 0x20001010 . = (. + 0x1000)
96 0x20000010 0x1000
97 0x20001010 Stack_Top = .
98 LOAD Action_Controller.o
99 LOAD Alarm_Actuator_Driver.o
100 LOAD driver.o
101 LOAD main.o
102 LOAD Pressure_Sensor_Driver.o
103 LOAD startup.o
104 OUTPUT(cortex-M3-pressureDetection.elf elf32-littlearm)
105 LOAD linker stubs
106
107 .debug_info 0x00000000 0x610
108 .debug_info 0x00000000 0x6e Action_Controller.o
109 .debug_info 0x0000000e 0xc3 Alarm_Actuator_Driver.o
110 .debug_info 0x000001ad 0x12 driver.o
111 .debug_info 0x0000021f 0x3d main.o
112 .debug_info 0x00000392 0x15 Pressure_Sensor_Driver.o
113 .debug_info 0x000004a7 0x109 startup.o
114
115 .debug_abbrev 0x00000000 0x3dc
116 .debug_abbrev 0x00000000 0x61 Action_Controller.o
117 .debug_abbrev 0x00000091 0x6d Alarm_Actuator_Driver.o
118 .debug_abbrev 0x000000fe 0xc3 driver.o

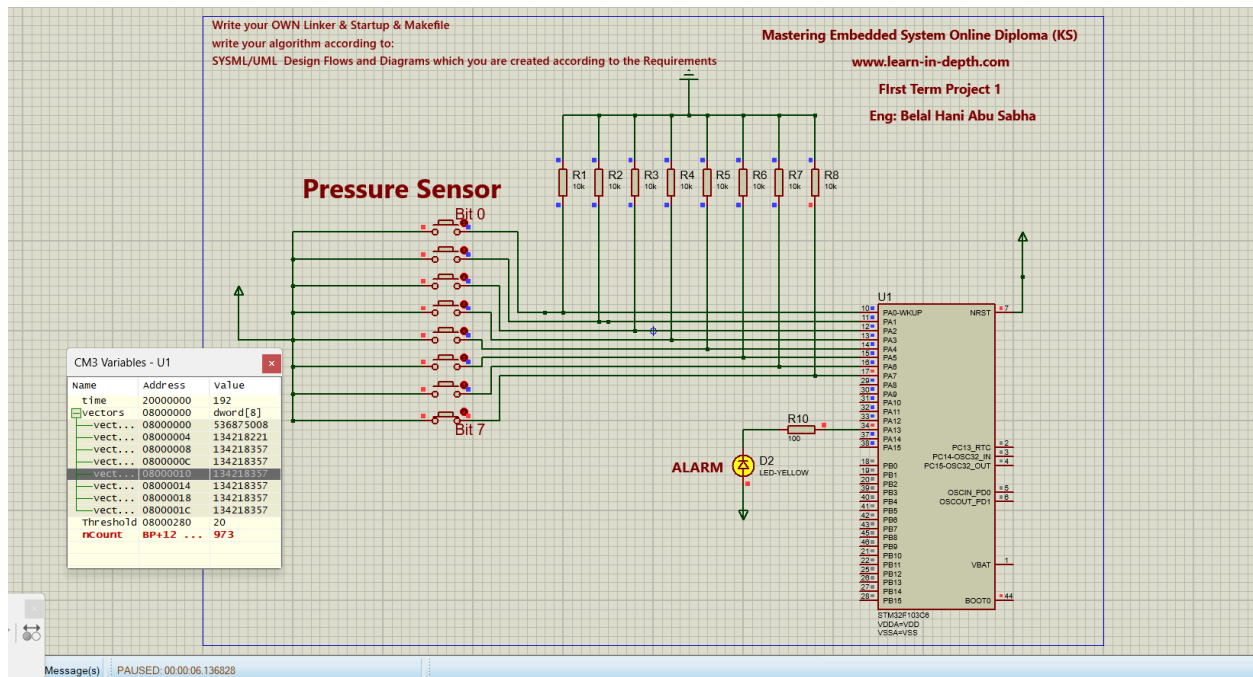
```

```
C:\Users\Belah\Downloads\Src\mab_file.map - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

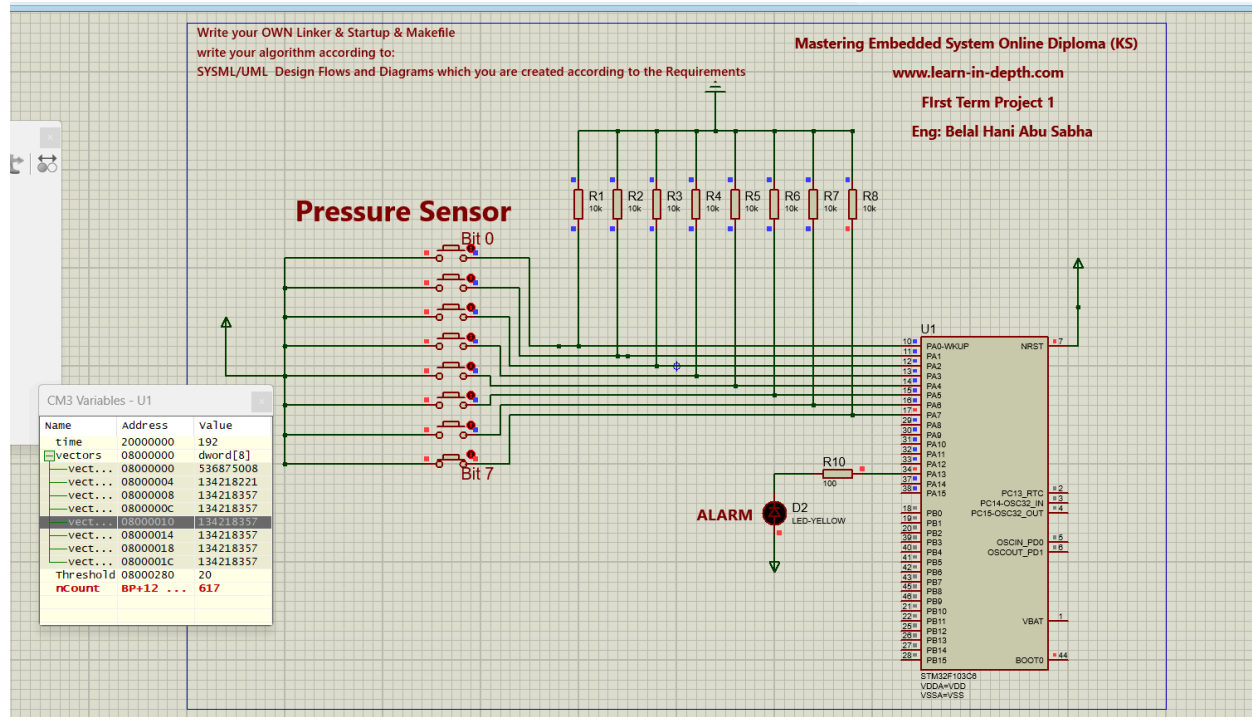
LinkerScript.mab x startup.o Makefile mab_file.map
145 .debug_line 0x00000000 0x003
146 .debug_line 0x00000000 0x00a Action_Controller.o
147 .debug_line 0x00000000 0x008 Alarm_Actuator_Driver.o
148 .debug_line 0x00000000 0x14f driver.o
149 .debug_line 0x00000239 0x00e main.o
150 .debug_line 0x000002c7 0x00d Pressure_Sensor_Driver.o
151 .debug_line 0x00000354 0x00f startup.o
152
153 .debug_str 0x00000000 0x2b6
154 .debug_str 0x00000000 0x128 Action_Controller.o
155 .debug_str 0x00000000 0x16a (size before relaxing)
156 .debug_str 0x00000128 0x04d Alarm_Actuator_Driver.o
157 .debug_str 0x0000016c 0x15a (size before relaxing)
158 .debug_str 0x0000016c 0x57 driver.o
159 .debug_str 0x000001c3 0x176 (size before relaxing)
160 .debug_str 0x000001c3 0x27 main.o
161 .debug_str 0x000001ea 0x152 (size before relaxing)
162 .debug_str 0x000001ea 0x52 Pressure_Sensor_Driver.o
163 .debug_str 0x000001ea 0x186 (size before relaxing)
164 .debug_str 0x0000023c 0x7a startup.o
165 .debug_str 0x0000023c 0x11f (size before relaxing)
166
167 .comment 0x00000000 0x49
168 .comment 0x00000000 0x00a Action_Controller.o
169 .comment 0x00000000 0x008 Alarm_Actuator_Driver.o
170 .comment 0x00000049 0x4d driver.o
171 .comment 0x00000049 0x15a (size before relaxing)
172 .comment 0x00000049 0x57 driver.o
173 .comment 0x00000049 0x176 (size before relaxing)
174 .comment 0x00000049 0x27 main.o
175 .comment 0x00000049 0x152 (size before relaxing)
176 .comment 0x00000049 0x52 Pressure_Sensor_Driver.o
177 .comment 0x00000049 0x186 (size before relaxing)
178 .comment 0x00000049 0x7a startup.o
179 .comment 0x00000049 0x11f (size before relaxing)
180
181 .ARM.attributes 0x00000000 0x2d
182 .ARM.attributes 0x00000000 0x2d Action_Controller.o
183 .ARM.attributes 0x0000002d 0x2d Alarm_Actuator_Driver.o
184 .ARM.attributes 0x0000005a 0x2d driver.o
185 .ARM.attributes 0x0000005a 0x2d main.o
186 .ARM.attributes 0x00000087 0x2d main.o
187 .ARM.attributes 0x000000b4 0x2d Pressure_Sensor_Driver.o
188 .ARM.attributes 0x000000b4 0x2d startup.o
189 .ARM.attributes 0x000000e1 0x2d startup.o
190
191 .debug_frame 0x00000000 0x23c
192 .debug_frame 0x00000000 0x2c Action_Controller.o
193 .debug_frame 0x0000002c 0x68 Alarm_Actuator_Driver.o
194 .debug_frame 0x00000034 0xa0 driver.o
195 .debug_frame 0x00000124 0x48 main.o
196 .debug_frame 0x0000017c 0x70 Pressure_Sensor_Driver.o
197 .debug_frame 0x000001ec 0x58 startup.o
198
199
```

Simulation Using Proteus :

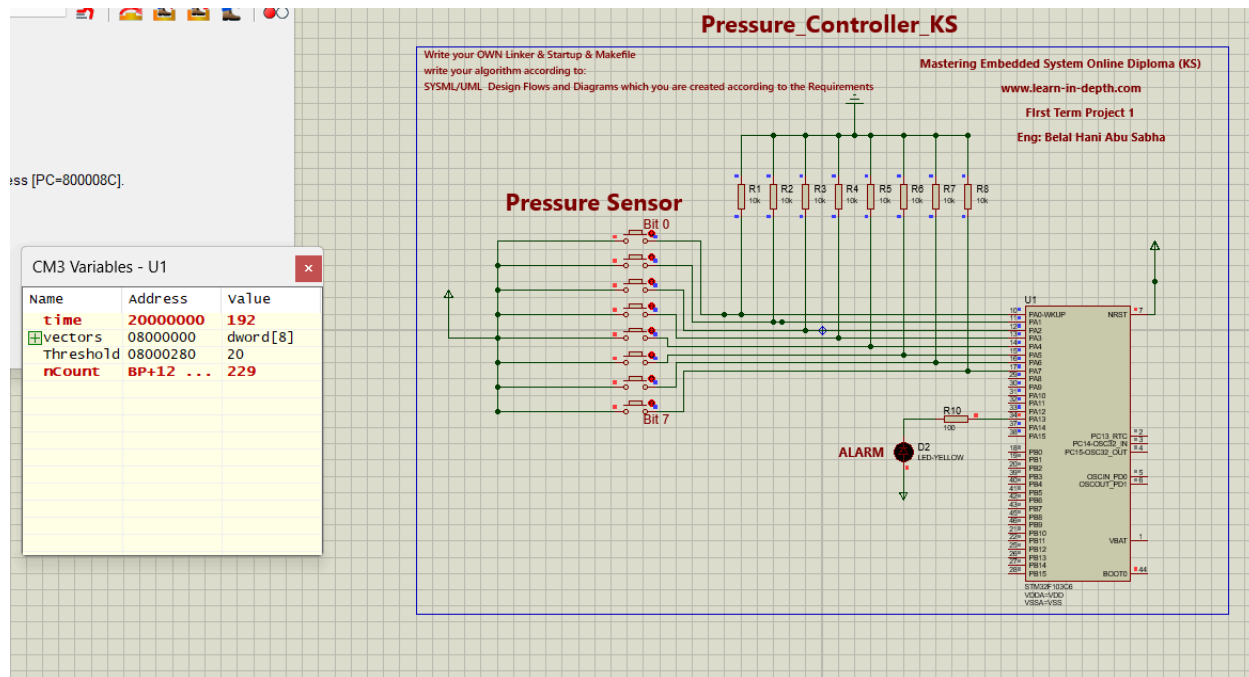
-if Pressure above or equal 20 bar then Alarm will turn on for 60 seconds



Alarm is off after 60 seconds :



else :



Conclusion :

Results:

- 1-Successfully implemented pressure detection system on STM32F103C6
- 2-System automatically senses pressure and controls alarm
- 3-All functional requirements achieved

Future Improvements:

- 1-Store pressure value in Memory