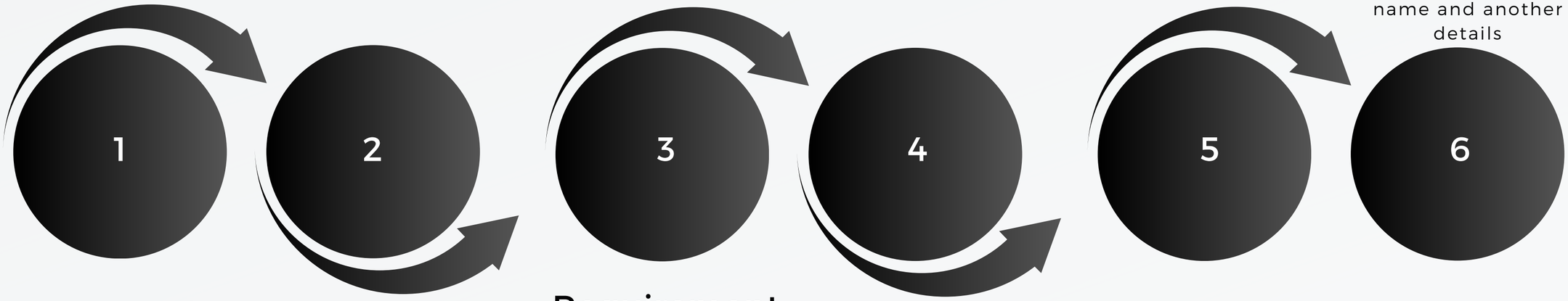
PRESSURE MANGEMENT SYSTEM PROJECT

PRESENTED TO
Keroles Shenouda

PRESENTED BY
Guirguis Hedia Henan

1-8-2023



case Study

Listen Carefully to the Customer and Write What he Want

Testing Method

Choose A suitable Testing Method like V-model , Agile Model , waterfall model Or spiral model

Requirement

Draw and Document The requirement of the Custom To insure that you understand what the customer want

Space Exploration

Choose the suitable microcontroller and choose Your Hardware

System Analysis

This Step You can share with the Customer you should implement **The Use case Diagram** ,**The Activity Diagram** and **The Sequence Diagram**

System Design

Draw State Diagram to each module Before starting coding and Determining how the modules communicate with each other and the APIs name and another details

Source Code

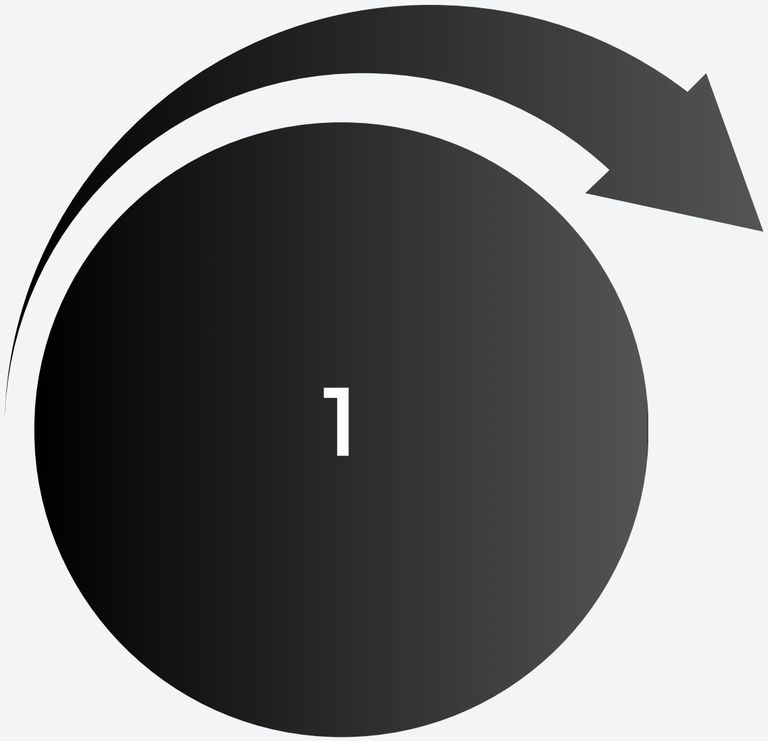
Begin to Write Your Code

Sections and symbols for each object file

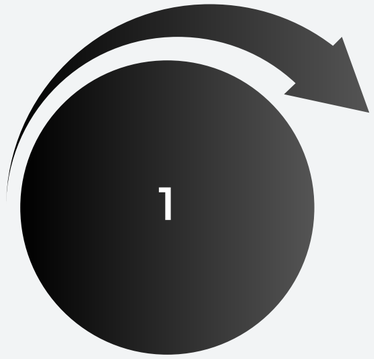
Sections and symbols for Final execution File

Hardware Simulation

Simulate Your Code to make sure it work correctly in proteus or KeilMirovision or another simulation Tool



Case Study



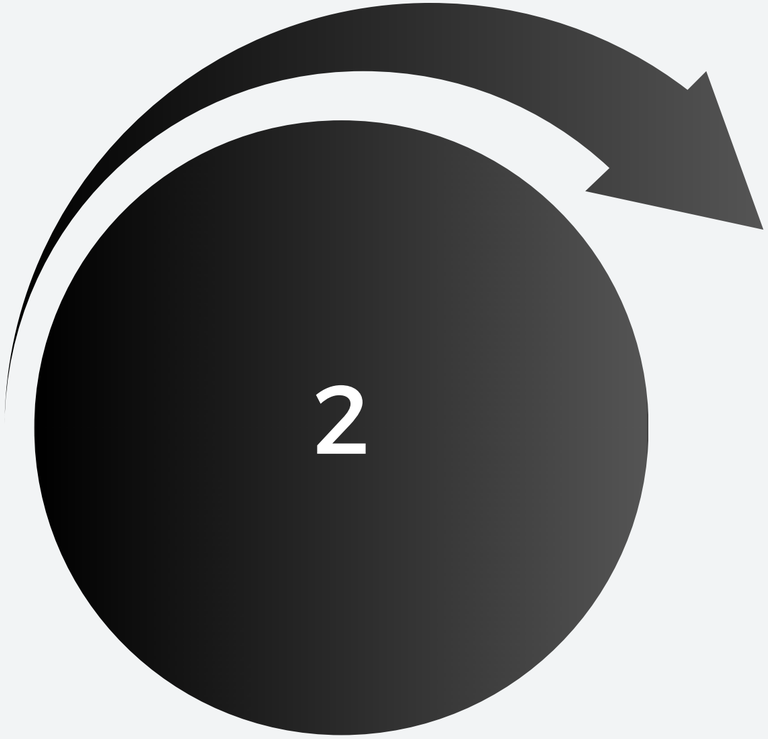
case Study

A pressure controller informs the crew of a cabin with an alarm when the pressure exceeds 20 bars in the cabin. The alarm duration equals 5 seconds.

Assumptions:

1- The Pressure Sensor Never Failed
2- The Alarm Never Failed

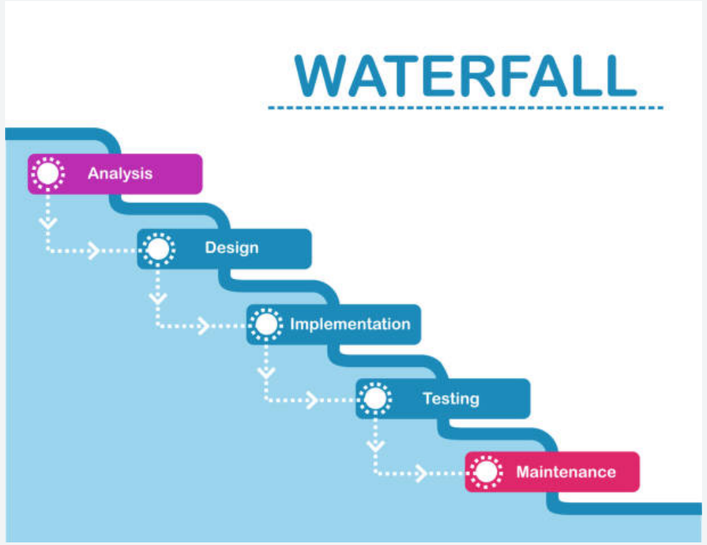
AGENDA

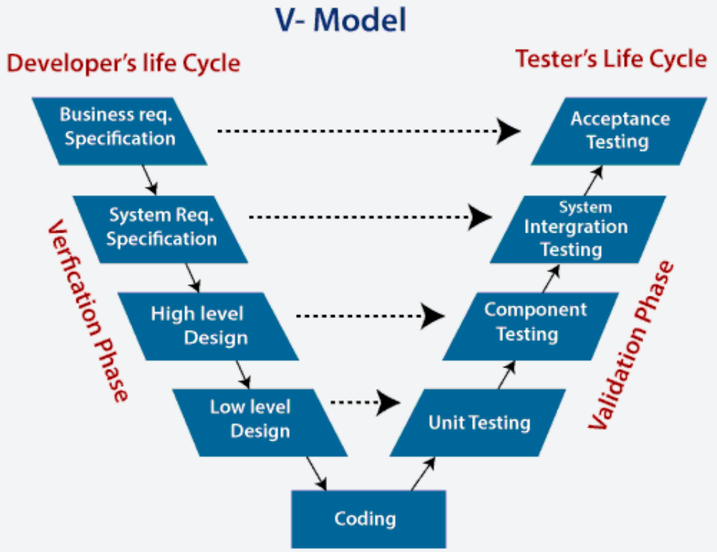


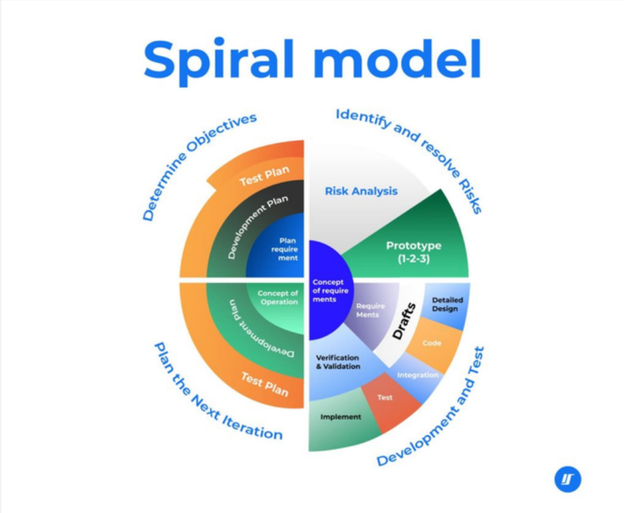
Testing Method

2

Testing Method







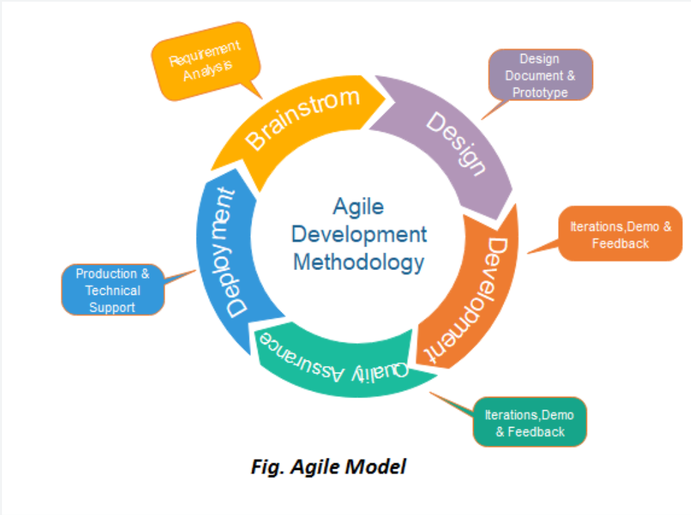
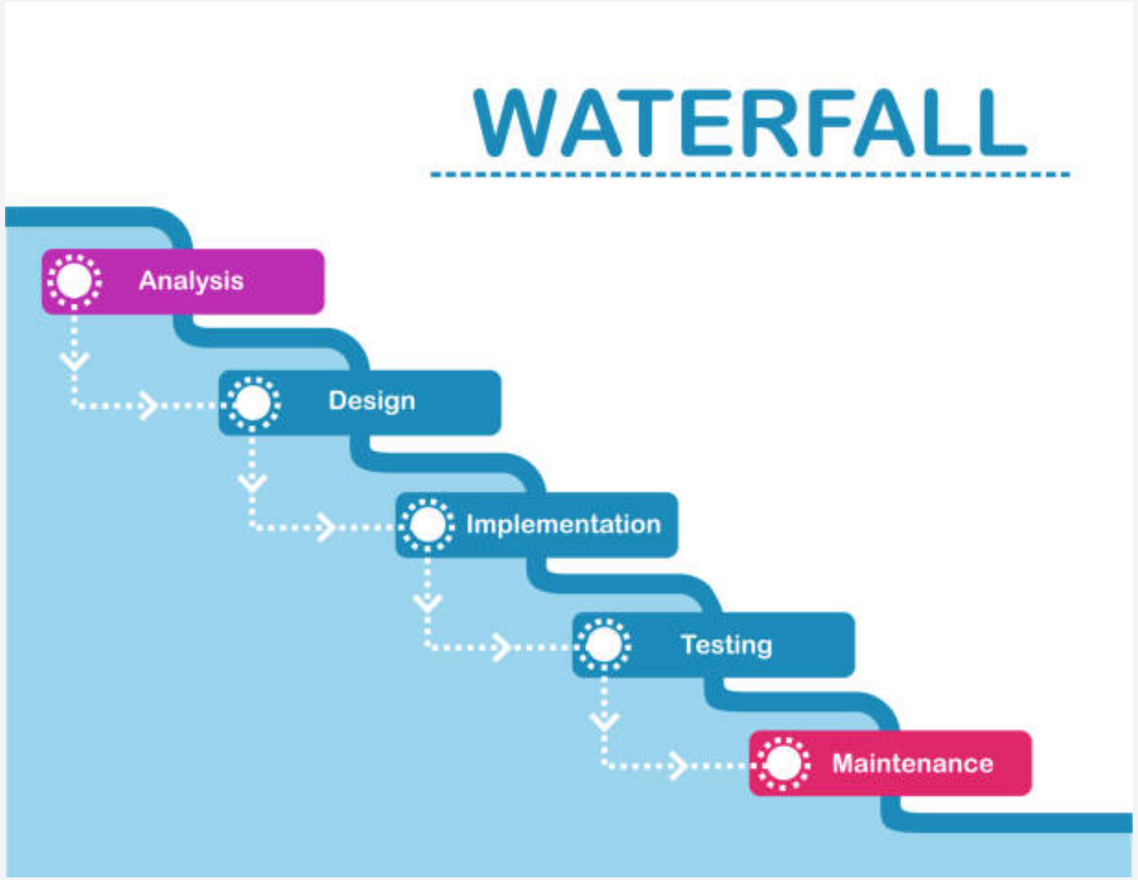


Fig. Agile Model

AGENDA

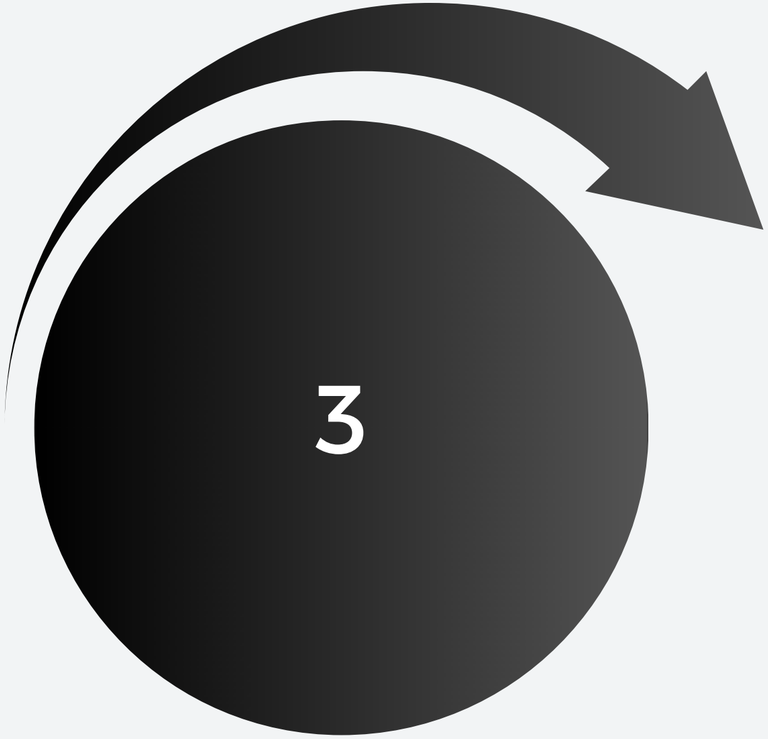
2

Testing
Method



We Choose WATERFALL Model Because the project still Small So this is the fastest Model to implement

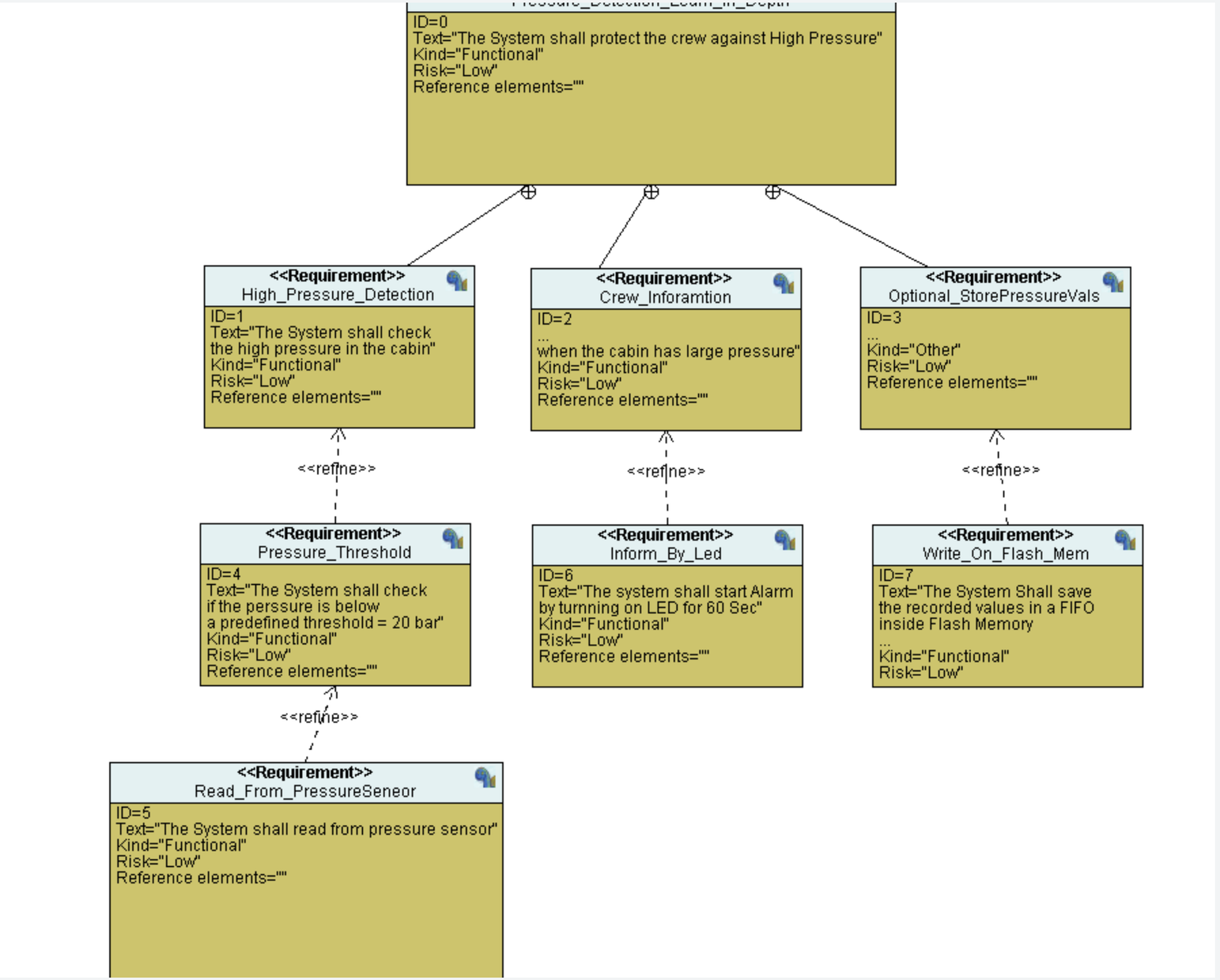
AGENDA

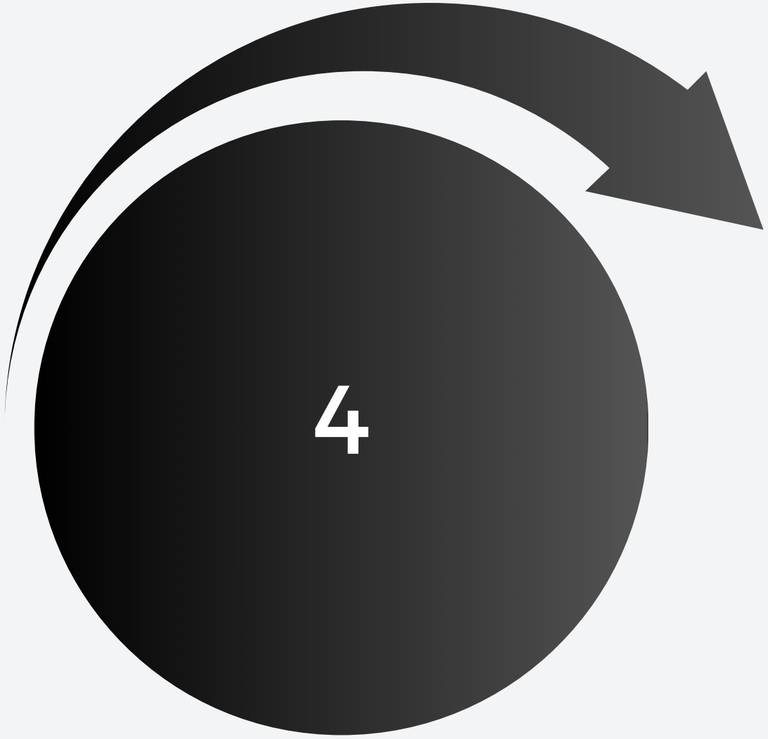


Requirement

3

Requirement

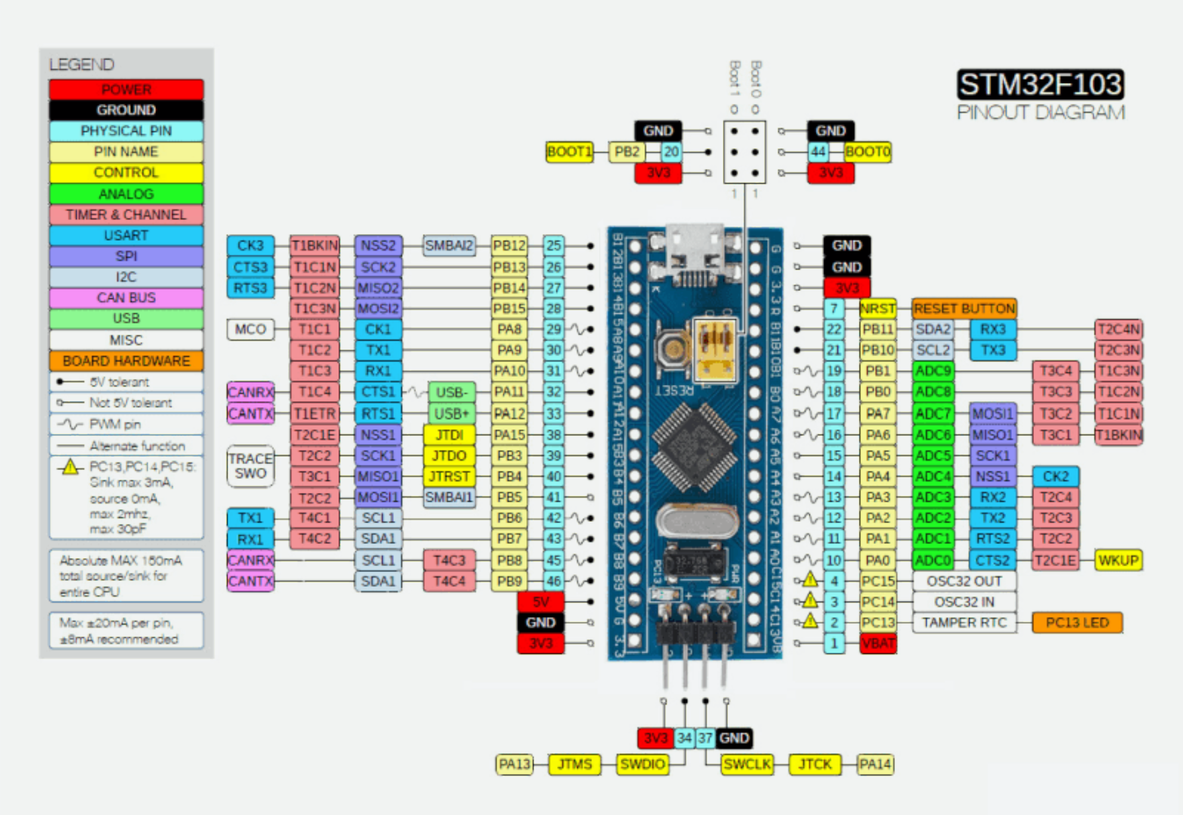




Space Exploration

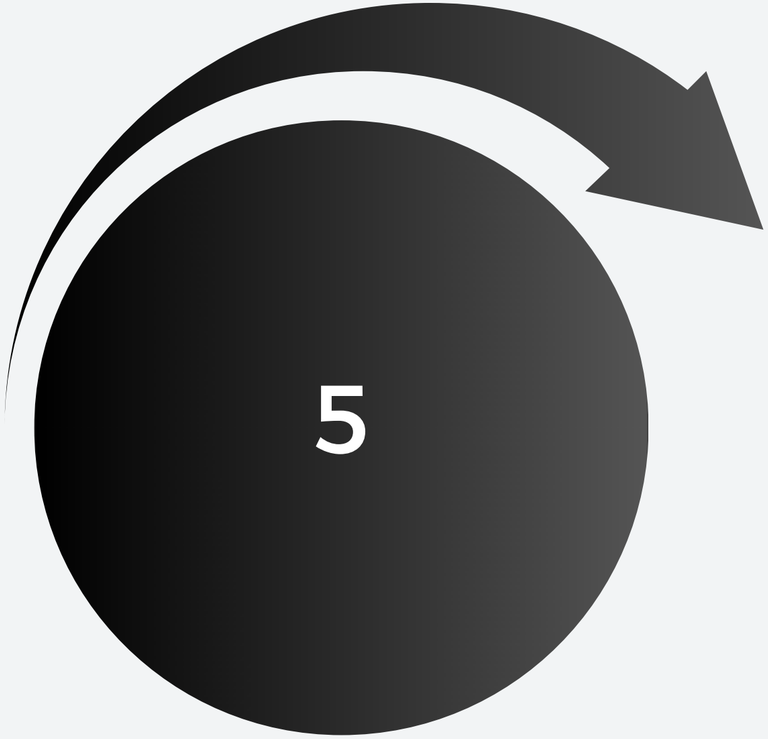
4

Space Exploration



We Choose blue pill ECU to Run the code on it
blue pill has Stm32f103c6 Microcontroller and Use ARM M3 processor
and this choice will be very enough for this project

AGENDA

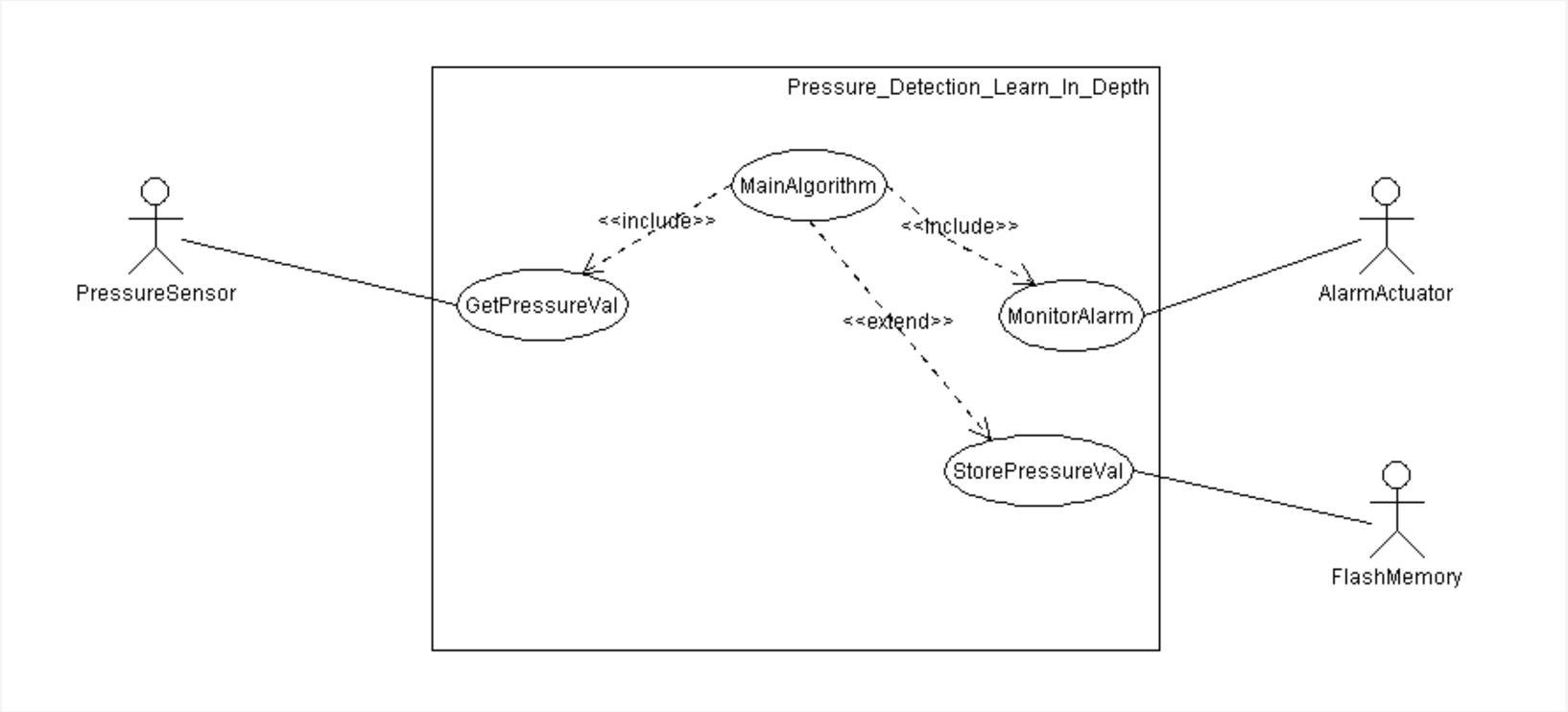


System Analysis

5

System Analysis

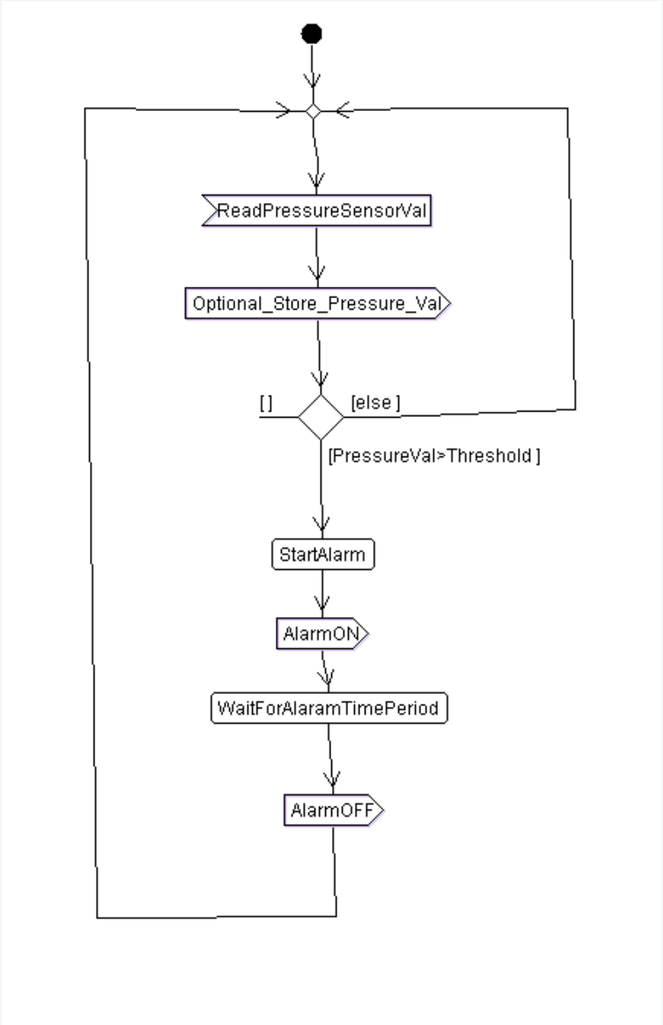
Use Case Diagram:



5

System Analysis

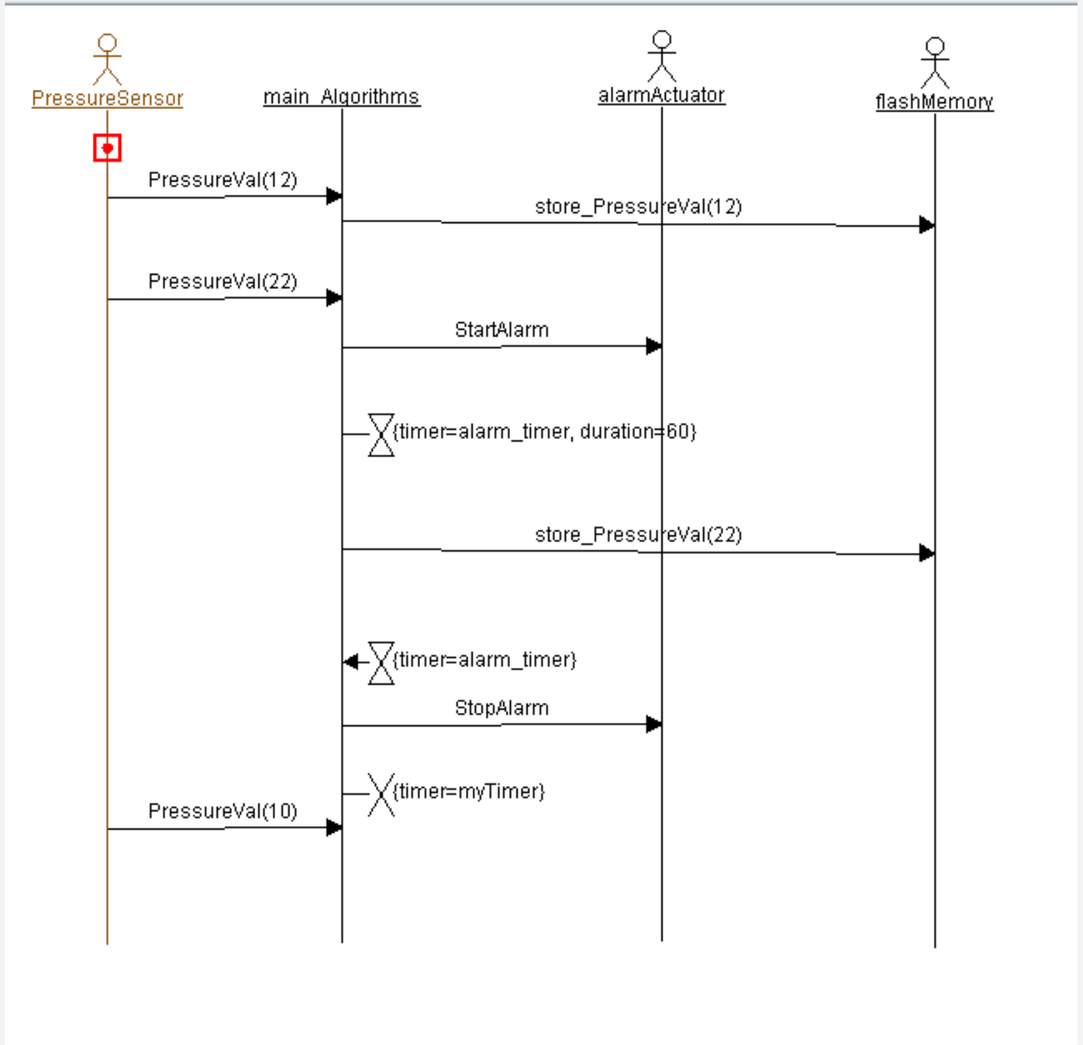
Activity Diagram:

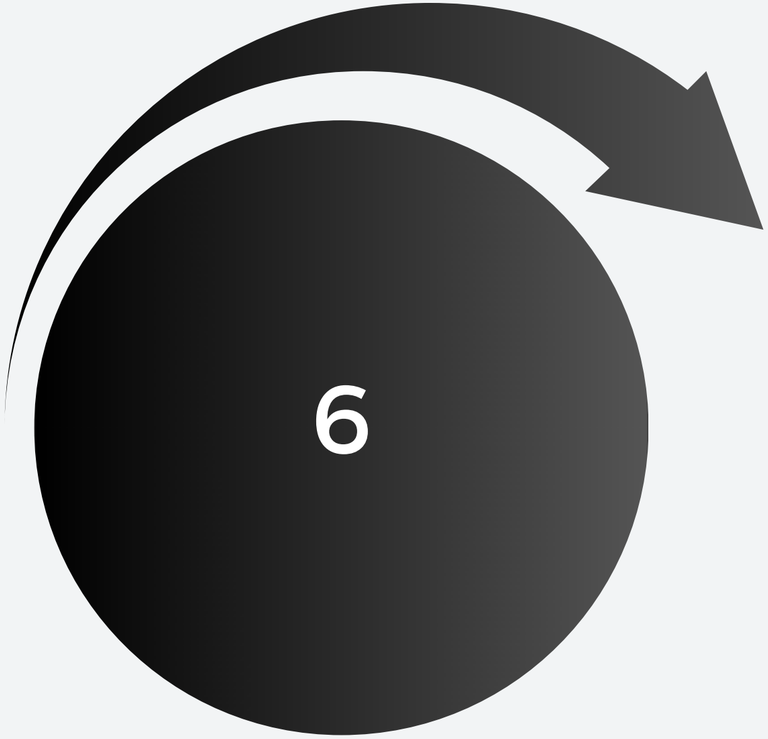


5

System Analysis

Sequence Diagram:





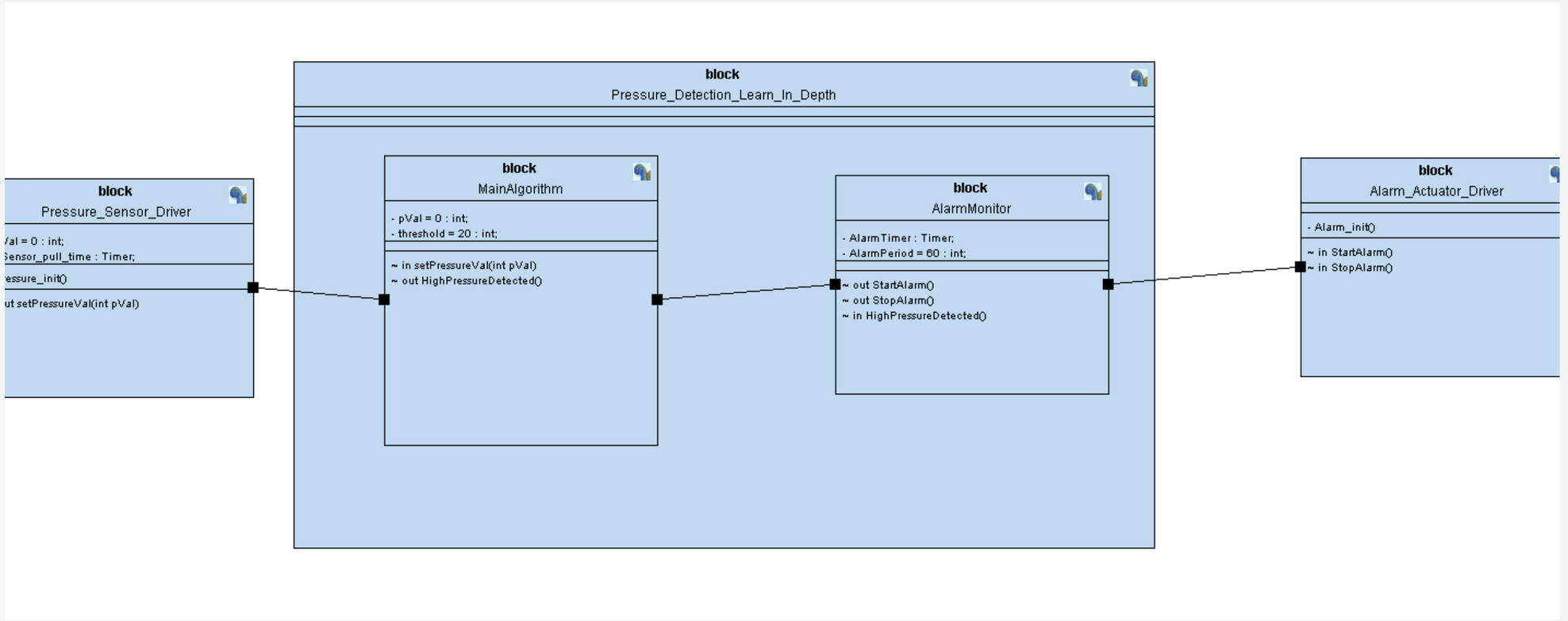
System Design

AGENDA

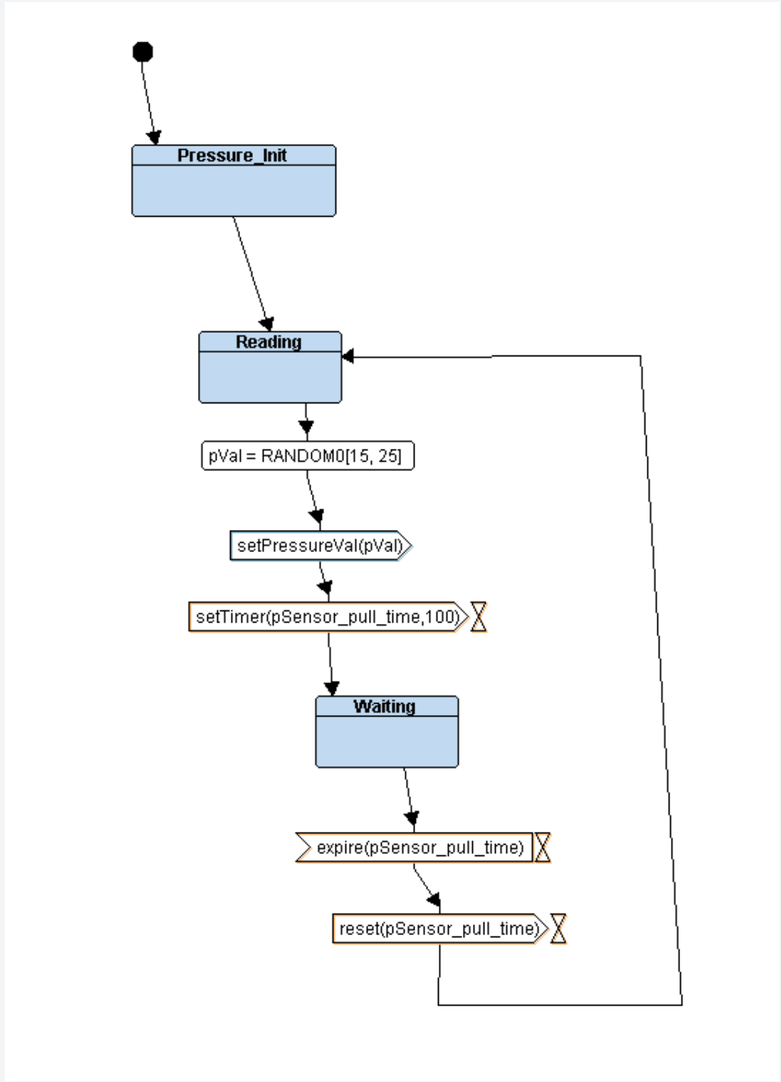
6

System Design

System Blocks :

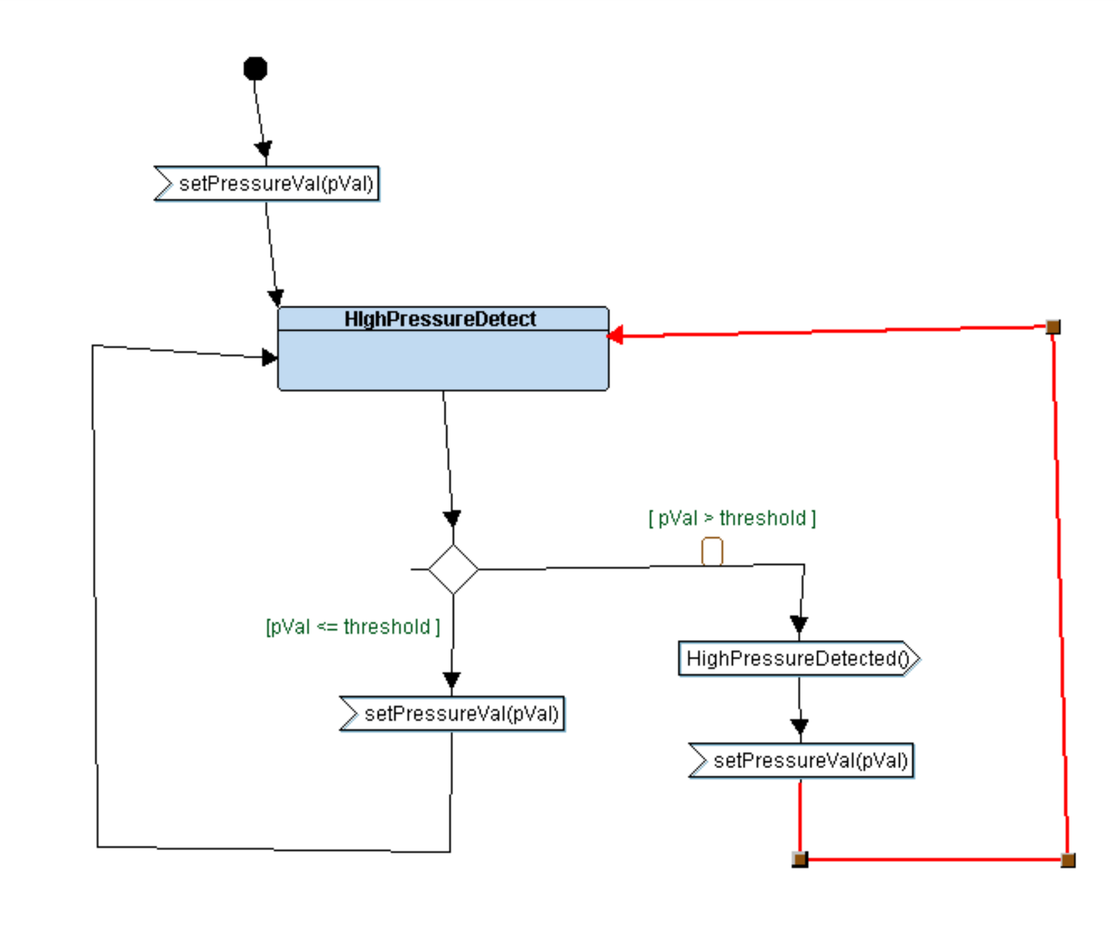


Pressure Sensor Driver State :



System Design

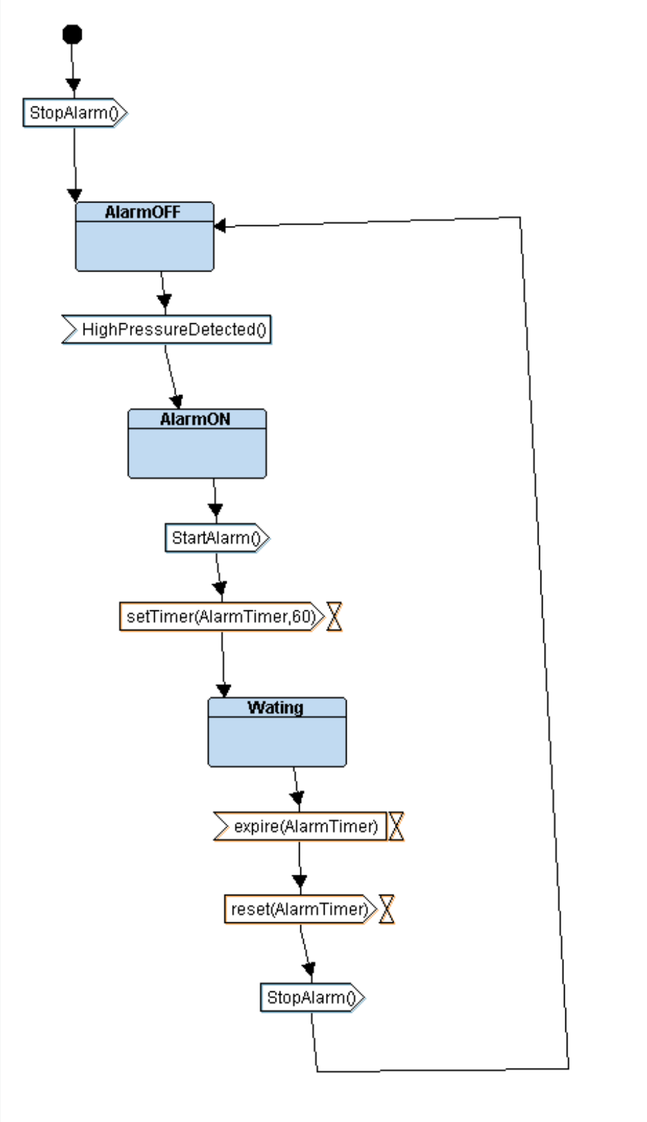
Main Algorithm State :



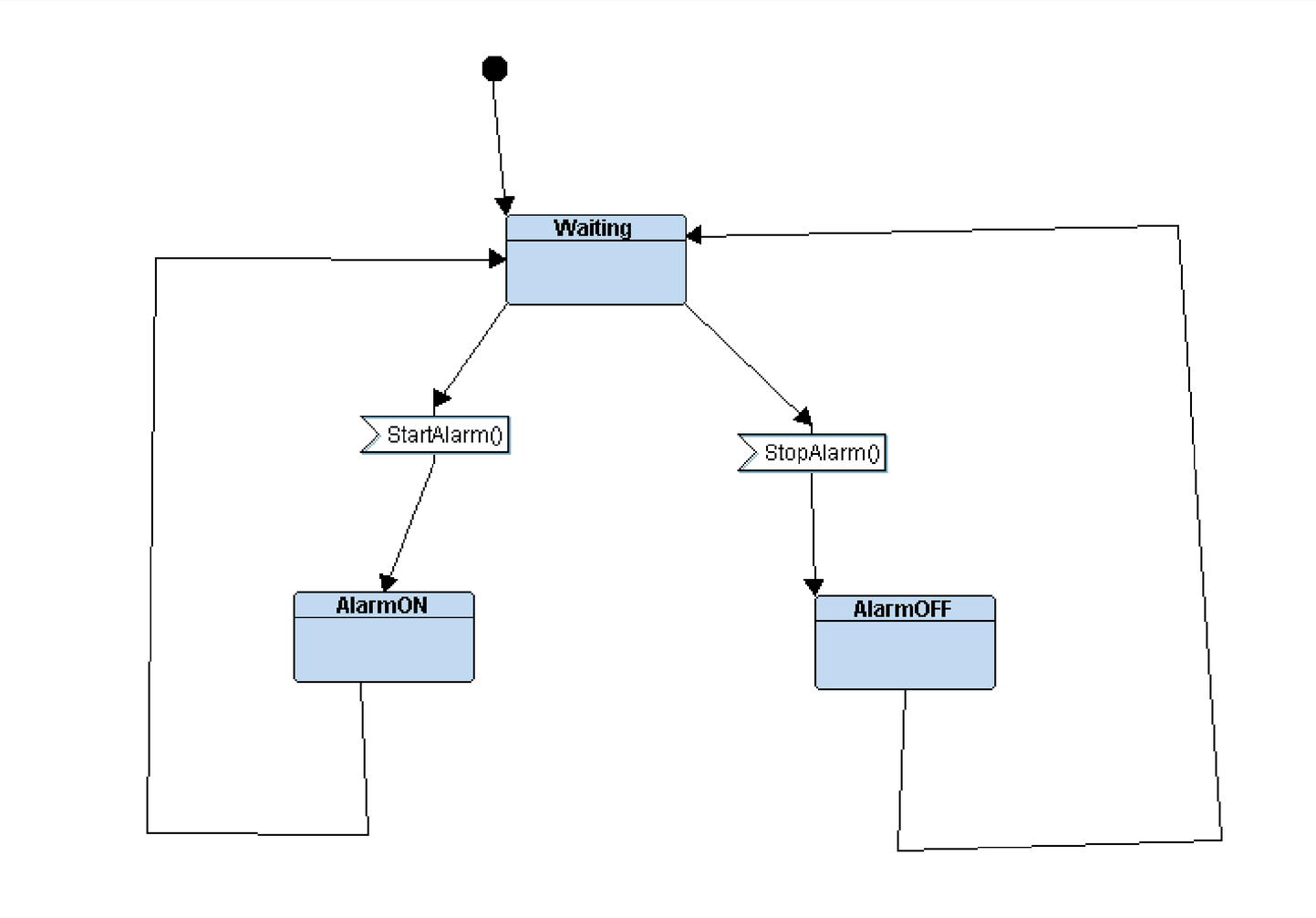
6

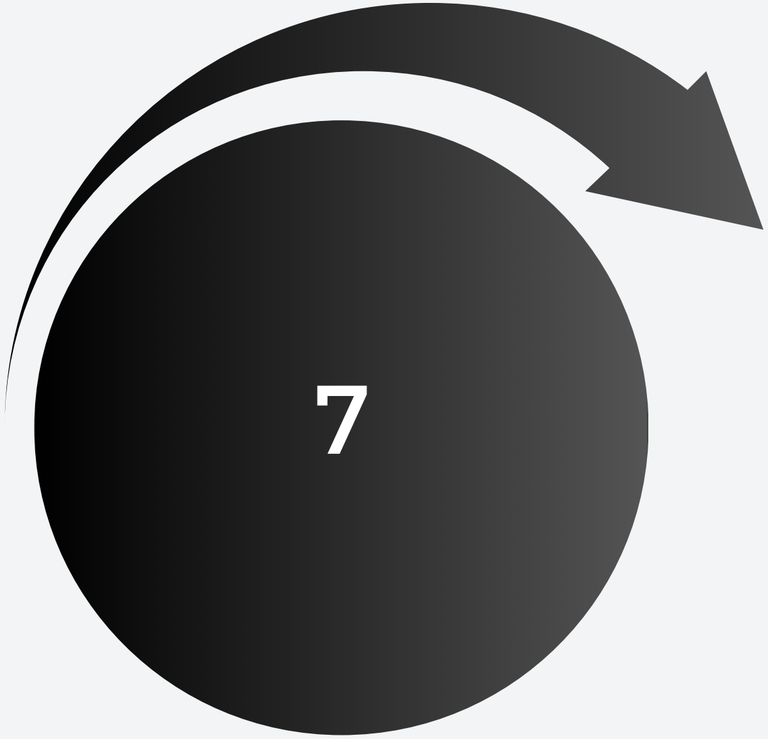
System Design

Alarm Monitor State :

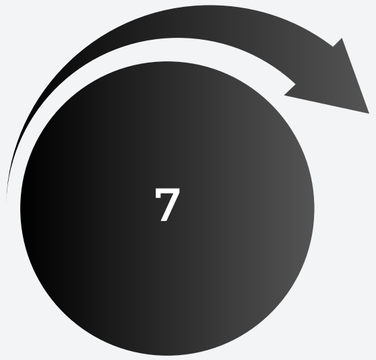


Alarm Actuator Driver State :





Source Code



Source Code

Folder Stucture

Alarm_Actuator_Driver.c	8/2/2023 1:25 PM	C Source File	2 KB
Alarm_Actuator_Driver.h	7/20/2023 3:55 AM	H File	1 KB
AlarmMonitor.c	8/2/2023 3:57 PM	C Source File	2 KB
AlarmMonitor.h	7/20/2023 3:55 AM	H File	1 KB
Backup Of Learn-In-depth_project_1.pds...	7/20/2023 5:45 AM	PDSBAK File	17 KB
BIT_MATH.h	2/27/2023 1:50 PM	H File	1 KB
GPIO_config.h	2/27/2023 4:45 PM	H File	1 KB
GPIO_interface.h	7/20/2023 4:07 AM	H File	6 KB
GPIO_private.h	3/1/2023 10:22 PM	H File	2 KB
GPIO_progam.c	7/20/2023 4:07 AM	C Source File	10 KB
Last Loaded Learn-In-depth_project_1.pd...	8/2/2023 6:20 PM	PDSBAK File	17 KB
learn_in_depth_cortex_m3.elf.asm	8/2/2023 12:18 PM	ASM File	1 KB
learn_in_depth_Project1.elf.asm	8/2/2023 3:57 PM	ASM File	1 KB
Learn-In-depth_project_1.pdsprj	8/2/2023 12:21 PM	Proteus Project	17 KB
Learn-In-depth_project_1.pdsprj.DESKTO...	8/2/2023 12:22 PM	WORKSPACE File	15 KB
linker_script.ld	7/20/2023 4:31 AM	LD File	1 KB
main.c	8/2/2023 3:42 PM	C Source File	2 KB
MainAlgorithm.c	7/20/2023 4:21 AM	C Source File	1 KB
MainAlgorithm.h	7/20/2023 3:55 AM	H File	1 KB
makefile	8/2/2023 12:20 PM	File	1 KB
Map_file.map	8/2/2023 6:28 PM	MAP File	4 KB
Pressure_Sensor_Driver.c	8/2/2023 12:23 PM	C Source File	2 KB
Pressure_Sensor_Driver.h	7/20/2023 3:55 AM	H File	1 KB
RCC_config.h	8/2/2023 1:05 PM	H File	4 KB
RCC_interface.h	3/4/2023 1:32 AM	H File	3 KB
RCC_private.h	2/27/2023 2:26 PM	H File	3 KB
RCC_program.c	3/1/2023 3:23 PM	C Source File	3 KB
startup.c	7/20/2023 4:37 AM	C Source File	2 KB
state.h	7/20/2023 3:55 AM	H File	1 KB
STD_TYPES.h	3/17/2023 4:27 PM	H File	1 KB
STK_config.h	3/5/2023 2:28 PM	H File	1 KB
STK_interface.h	3/5/2023 11:11 PM	H File	1 KB
STK_private.h	3/5/2023 10:43 PM	H File	1 KB
STK_program.c	8/2/2023 3:57 PM	C Source File	3 KB

AGENDA

Source Code

Startup.c

```c
//Startup.c
//Guirguis Hedia

#include <stdint.h>
#define STACK_Start_SP 0x20001000
extern int main (void);

void Reset_Handler(void) ;

void Default_Handler()
{
    Reset_Handler();
}

void NMI_Handler (void) __attribute__ ((weak, alias ("Default_Handler")));

void H_Fault_Handler(void) __attribute__ ((weak, alias ("Default_Handler")));

void MM_Fault_Handler(void) __attribute__ ((weak, alias ("Default_Handler")));

void Bus_Fault(void) __attribute__ ((weak, alias ("Default_Handler")));

void Usage_Fault_Handler(void) __attribute__ ((weak, alias ("Default_Handler")));


extern unsigned int _stack_top;
uint32_t vectors[] __attribute__((section(".vectors")))={

(uint32_t)  &_stack_top,
(uint32_t)  &Reset_Handler,
(uint32_t)  &NMI_Handler,
(uint32_t)  &H_Fault_Handler,
(uint32_t)  &MM_Fault_Handler,
(uint32_t)  &Bus_Fault,
(uint32_t)  &Usage_Fault_Handler
```

Source Code

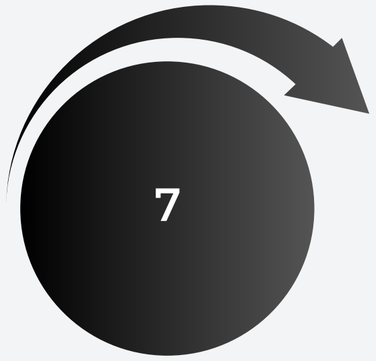
Startup.c

```
void Reset_Handler(void)
{
    int i;
    //copy data Section From Flash to Ram
    unsigned int DATA_size =(unsigned char*) &_E_DATA - (unsigned char*)&_S_DATA ;//
    unsigned char* P_src =(unsigned char*)&_E_text;
    unsigned char *P_dst =(unsigned char*)&_S_DATA;

    for(i=0;i<DATA_size;i++)
    {
        *((unsigned char *)P_dst++) = *((unsigned char *)P_src++) ;
    }
    //init .bss section in SRAM =0
    unsigned int bss_size =(unsigned char*) &_E_bss - (unsigned char*)&_S_bss ;
    P_dst=(unsigned char*)&_S_bss;
    for(i=0 ;i<bss_size;i++)
    {
        *((unsigned char *)P_dst++) = (unsigned char)0 ;
    }

    //jump main()
    main();
}
```

Source Code

linker_script.ld File :

```
/*Linker Script CortexM3
Guirguis Hedia
*/
MEMORY
{
flash(RX) : ORIGIN =0x08000000, LENGTH =128K
sram(RWX) : ORIGIN =0x20000000, LENGTH =20K

}

SECTIONS
{
    .text : {
            *(.vectors*)
            *(.text*)
            *(.rodata)
            _E_text = .;
    }>flash


    .data : {
    _S_DATA = . ;
    *(.data)
    . = ALIGN(4) ;
    _E_DATA = . ;

    }>sram AT> flash


    .bss : {
    _S_bss = . ;
    *(.bss*)
    _E_bss = . ;
    . = ALIGN(4);
    . = . +0x1000;
    _stack_top = .;

    }> sram
}
```



Source Code

MakeFile :

```
#@Copyright : Guiguis Hedia

CC=arm-none-eabi-
CFLAGS=-gdwarf-2 -mcpu=cortex-m3 -mthumb
INCS=-I .
LIBS=
SRC= $(wildcard *.c)
OBJ= $(SRC:.c=.o)
As= $(wildcard *.s)
AsOBJ= $(As:.s=.o)

Project_Name=learn_in_depth_Project1

all: $(Project_Name).bin
    @echo "============Build is Done============"
#startup.o: startup.s
#    $(CC)as.exe $(CFLAGS) $< -o $@

%.o: %.c
    $(CC)gcc.exe -c $(CFLAGS) $(INCS) $< -o $@


$(Project_Name).elf: $(OBJ) $(AsOBJ)
    $(CC)ld.exe -T linker_script.ld $(LIBS) $(OBJ) $(AsOBJ) -o $@ -Map=Map_file.map

$(Project_Name).bin: $(Project_Name).elf

    $(CC)objcopy.exe -O binary $< $@

clean_all:
        rm *.o *.elf *.bin

clean:
        rm *.elf *.bin
```

Source Code

State.h

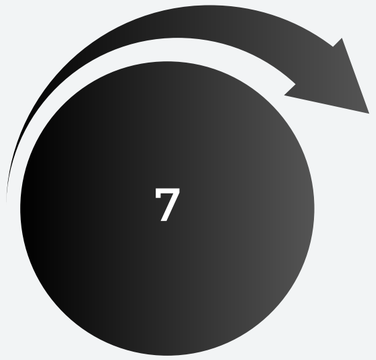
```c
/* state.h..

#ifndef STATE_H_
#define STATE_H_
#include "stdio.h"
#include "stdlib.h"

typedef enum
{
    PressureDetected,
    PressureNotDetected
}PressureState;


//Automatic State Function Generated
#define STATE_define(_statFUN_) void ST_##_statFUN_()
#define STATE(_statFUN_)     ST_##_statFUN_


//States Connection
extern void PressureSensorDriver_setPressureVal(int pVal);
extern void MainAlgorithm_HighPressureDetected(PressureState Pstate);
extern void AlarmMonitor_StartAlarm();
extern void AlarmMonitor_StopAlarm();



#endif /* STATE_H_ */

```

Pressure_Sensor_Driver

Source Code

main.c | Pressure_Sensor_Driver.c | MainAlgorithm.c | AlarmMonitor.c | Alarm_Actuator_Driver.c | STK_prc

```c
#include "state.h"
#include "Pressure_Sensor_Driver.h"

//Drivers For Hardware
#include "STD_TYPES.h"
#include "BIT_MATH.h"
#include "RCC_interface.h"
#include "GPIO_interface.h"
#include "STK_interface.h"


static int PressureSensorVal;



void (*PressureSensor_state )();


void PressureSensor_init()
{
    //Here We Will Initialize HW for Pressure Sensor
    //Pressure Sensor Connect with MicroController in PORTA PINs from 0 to 7
    MGPIO_voidSetByteDir(GPIOA,GPIO_BYTE_POSITION_LOW,GPIO_BYTE_INPUT_FLOATING);
}

//Declare States Functions PressureSensor
STATE_define(PressureSensor_Reading)
{
    //Set Enum Value for Debugging
    PressureSensor_state_id = PressureSensorReading;

    //Get Pressure Value
    PressureSensorVal = MGPIO_voidGetByteVal(GPIOA,GPIO_BYTE_POSITION_LOW);


    //Send Pressure Value to Main Algorithm Module
    PressureSensorDriver_setPressureVal(PressureSensorVal);

    //Set Next State
    PressureSensor_state = STATE(PressureSensor_Reading);

}
```



Source Code

MainAlgorithm.c

main.c | Pressure_Sensor_Driver.c | MainAlgorithm.c | AlarmMonitor.c | Alarm_Actuator_Driver

```c
/* MainAlgorithm.c..

#include "state.h"
#include "MainAlgorithm.h"

//Drivers For Hardware
#include "STD_TYPES.h"
#include "BIT_MATH.h"
#include "RCC_interface.h"
#include "GPIO_interface.h"
#include "STK_interface.h"


static int MainAlgorithmPressureVal;
static int MainAlgorithmThreshold = 20;


void (*MainAlgorithm_state )();

void PressureSensorDriver_setPressureVal(int pVal)
{
    MainAlgorithmPressureVal = pVal;
}


STATE_define(MainAlgorithm_HighPressureDetect)
{

    //Set Enum Value for Debugging
    MainAlgorithm_state_id = MainAlgorithmHighPressureDetect;

    if (MainAlgorithmPressureVal > MainAlgorithmThreshold)
    {
        MainAlgorithm_HighPressureDetected(PressureDetected);
    }
    else
    {
        MainAlgorithm_HighPressureDetected(PressureNotDetected);
    }
    //Set Next State
    MainAlgorithm_state = STATE(MainAlgorithm_HighPressureDetect);

}
```

7

Source Code

AlarmMonitor.c

```c

#include "state.h"
#include "AlarmMonitor.h"

//Drivers For Hardware
#include "STD_TYPES.h"
#include "BIT_MATH.h"
#include "RCC_interface.h"
#include "GPIO_interface.h"
#include "STK_interface.h"

void (*AlarmMonitor_state )();

void MainAlgorithm_HighPressureDetected(PressureState Pstate)
{
    switch(Pstate)
    {
    case PressureDetected:
        AlarmMonitor_state = STATE(AlarmMonitor_ON);
        break;
    case PressureNotDetected:
        AlarmMonitor_state = STATE(AlarmMonitor_OFF);
        break;
    default:
        break;
    }

}
```

main.c | Pressure_Sensor_Driver.c | MainAlgorithm.c | AlarmMonitor.c | Alarm_Actu

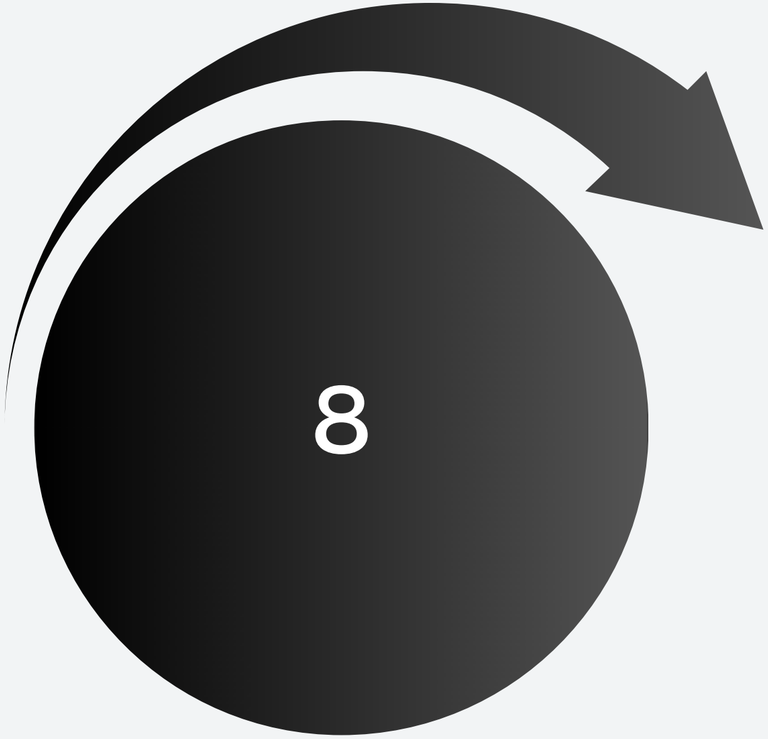
```c
}

STATE_define(AlarmMonitor_ON)
{
    //Set Enum Value for Debugging
    AlarmMonitor_state_id = AlarmMonitorON;


    AlarmMonitor_StartAlarm();

    //Turn on AlarmMonitor_Waiting State
    STATE(AlarmMonitor_Waiting)();

    //Set Next State
    AlarmMonitor_state = STATE(AlarmMonitor_OFF);

}

STATE_define(AlarmMonitor_OFF)
{
    //Set Enum Value for Debugging
    AlarmMonitor_state_id = AlarmMonitorOFF;

    //Stop Alarm
    AlarmMonitor_StopAlarm();

    //Set Next State
    AlarmMonitor_state = STATE(AlarmMonitor_OFF);
}

STATE_define(AlarmMonitor_Waiting)
{

    //Set Enum Value for Debugging
    AlarmMonitor_state_id = AlarmMonitorWaiting;

    //Set Timer
    MSTK_voidSetBusyWait(5000000);

    //Stop Alarm
    AlarmMonitor_StopAlarm();

    //Set Next State
    AlarmMonitor_state = STATE(AlarmMonitor_OFF);

}
```

Alarm_Actuator_Driver.c

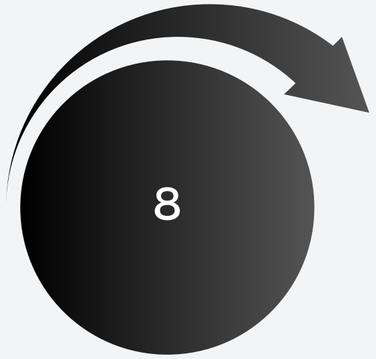
Source Code

main.c | Pressure_Sensor_Driver.c | MainAlgorithm.c | AlarmMonitor.c | Alarm_Actuator_Driver.c | STK_program.c

```c
void (*AlarmActuatorDriver_state )();

void AlarmActuatorDriver_init(void)
{
    //Initialization for HW of Buzzer which is Connected with MicroController with pin 13
    MGPIO_voidSetPinDir(GPIOA,GPIO_PIN13,GPIO_OUTPUT_SPEED_2MHz_PP);
}


void AlarmMonitor_StartAlarm()
{
    AlarmActuatorDriver_state =STATE(AlarmActuatorDriver_ON);
    AlarmActuatorDriver_state();
}
void AlarmMonitor_StopAlarm()
{
    AlarmActuatorDriver_state =STATE(AlarmActuatorDriver_OFF);
    AlarmActuatorDriver_state();
}

//Declare States Functions CA
STATE_define(AlarmActuatorDriver_ON)
{
    //Turn on Buzzer
    MGPIO_voidSetPinVal(GPIOA,GPIO_PIN13,GPIO_LOW);
    AlarmActuatorDriver_state =STATE(AlarmActuatorDriver_Waiting);

}
STATE_define(AlarmActuatorDriver_OFF)
{
    //Turn off Buzzer
    MGPIO_voidSetPinVal(GPIOA,GPIO_PIN13,GPIO_HIGH);
    AlarmActuatorDriver_state =STATE(AlarmActuatorDriver_Waiting);

}
STATE_define(AlarmActuatorDriver_Waiting)
{
    //NO Thing
    //Just Waiting for any change
}
```



Sections and symbols for each object file



Sections and symbols for each object file

```
Ephraim@DESKTOP-H8B8O6K MINGW32 /d/Learn_in_Depth/Learn_in_Depth_Unit5/Learn-In-depth_project_1
$ mingw32-make.exe
arm-none-eabi-gcc.exe -c -gdwarf-2 -mcpu=cortex-m3 -mthumb -I . GPIO_progam.c -o GPIO_progam.o
arm-none-eabi-gcc.exe -c -gdwarf-2 -mcpu=cortex-m3 -mthumb -I . startup.c -o startup.o
arm-none-eabi-gcc.exe -c -gdwarf-2 -mcpu=cortex-m3 -mthumb -I . MainAlgorithm.c -o MainAlgorithm.o
arm-none-eabi-gcc.exe -c -gdwarf-2 -mcpu=cortex-m3 -mthumb -I . RCC_program.c -o RCC_program.o
arm-none-eabi-gcc.exe -c -gdwarf-2 -mcpu=cortex-m3 -mthumb -I . Pressure_Sensor_Driver.c -o Pressure_Sensor_Driver.o
arm-none-eabi-gcc.exe -c -gdwarf-2 -mcpu=cortex-m3 -mthumb -I . main.c -o main.o
arm-none-eabi-gcc.exe -c -gdwarf-2 -mcpu=cortex-m3 -mthumb -I . STK_program.c -o STK_program.o
arm-none-eabi-gcc.exe -c -gdwarf-2 -mcpu=cortex-m3 -mthumb -I . Alarm_Actuator_Driver.c -o Alarm_Actuator_Driver.o
arm-none-eabi-gcc.exe -c -gdwarf-2 -mcpu=cortex-m3 -mthumb -I . AlarmMonitor.c -o AlarmMonitor.o
arm-none-eabi-ld.exe -T linker_script.ld  GPIO_progam.o startup.o MainAlgorithm.o RCC_program.o Pressure_Sensor_Driver.o main.o STK_p
rogram.o Alarm_Actuator_Driver.o AlarmMonitor.o  -o learn_in_depth_Project1.elf -Map=Map_file.map
arm-none-eabi-objcopy.exe -O binary learn_in_depth_Project1.elf learn_in_depth_Project1.bin
============Build is Done============
```

Build is Done

```
$ arm-none-eabi-nm.exe main.o
         U AlarmActuatorDriver_init
         U AlarmActuatorDriver_state
00000001 C AlarmActuatorDriver_state_id
         U AlarmMonitor_state
00000001 C AlarmMonitor_state_id
00000000 T HW_voidInit
00000078 T main
         U MainAlgorithm_state
00000001 C MainAlgorithm_state_id
         U MSTK_voidInit
         U PressureSensor_init
         U PressureSensor_state
00000001 C PressureSensor_state_id
         U RCC_u8EnableClock
         U RCC_voidInitSystemClk
0000001c T setup
         U ST_AlarmActuatorDriver_OFF
         U ST_AlarmMonitor_OFF
         U ST_MainAlgorithm_HighPressureDetect
         U ST_PressureSensor_Reading
```

main.o Symbol Table

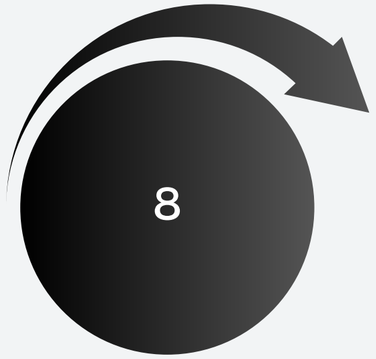
```
Ephraim@DESKTOP-H8B8O6K MINGW32 /d/Learn_in_Dep
$ arm-none-eabi-nm.exe AlarmMonitor.o
         U AlarmMonitor_StartAlarm
00000004 C AlarmMonitor_state
00000001 C AlarmMonitor_state_id
         U AlarmMonitor_StopAlarm
00000000 T MainAlgorithm_HighPressureDetected
         U MSTK_voidSetBusyWait
0000007c T ST_AlarmMonitor_OFF
0000004c T ST_AlarmMonitor_ON
000000a8 T ST_AlarmMonitor_Waiting
```

AlarmMointor.o Symbol Table

```
$ arm-none-eabi-nm.exe Alarm_Actuator_Driver.o
00000000 T AlarmActuatorDriver_init
00000004 C AlarmActuatorDriver_state
00000001 C AlarmActuatorDriver_state_id
00000018 T AlarmMonitor_StartAlarm
0000003c T AlarmMonitor_StopAlarm
         U MGPIO_voidSetPinDir
         U MGPIO_voidSetPinVal
00000088 T ST_AlarmActuatorDriver_OFF
00000060 T ST_AlarmActuatorDriver_ON
000000b0 T ST_AlarmActuatorDriver_Waiting
```

Alarm_Actuator_Driver.o Symbol Table

AGENDA



Sections and symbols for each object file

```
$ arm-none-eabi-nm.exe Pressure_Sensor_Driver.o
         U MGPIO_voidGetByteVal
         U MGPIO_voidSetByteDir
00000000 T PressureSensor_init
00000004 C PressureSensor_state
00000001 C PressureSensor_state_id
         U PressureSensorDriver_setPressureVal
00000000 b PressureSensorVal
00000018 T ST_PressureSensor_Reading
```

Pressure_Sensor_Driver.o Symbol Table

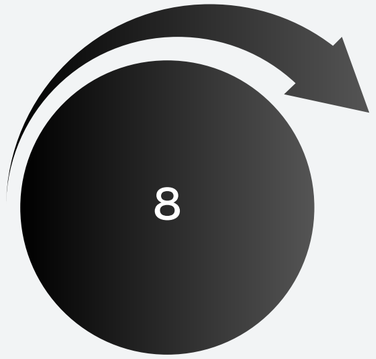
```
$ arm-none-eabi-nm.exe MainAlgorithm.o
         U MainAlgorithm_HighPressureDetected
00000004 C MainAlgorithm_state
00000001 C MainAlgorithm_state_id
00000000 b MainAlgorithmPressureVal
00000000 d MainAlgorithmThreshold
00000000 T PressureSensorDriver_setPressureVal
00000020 T ST_MainAlgorithm_HighPressureDetect
```

MainAlgorithm.o Symbol Table

```
$ arm-none-eabi-nm.exe startup.o
         U _E_bss
         U _E_DATA
         U _E_text
         U _S_bss
         U _S_DATA
         U _stack_top
00000000 W Bus_Fault
00000000 T Default_Handler
00000000 W H_Fault_Handler
         U main
00000000 W MM_Fault_Handler
00000000 W NMI_Handler
0000000c T Reset_Handler
00000000 W Usage_Fault_Handler
00000000 D vectors
```

Startup.o Symbol Table

AGENDA



Sections and symbols for each object file

```
$ 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         000000a8  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  000000dc  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000dc  2**0
                  ALLOC
  3 .debug_info   0000019c  00000000  00000000  000000dc  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 000000a7  00000000  00000000  00000278  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    00000084  00000000  00000000  0000031f  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000020  00000000  00000000  000003a3  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line   000000b5  00000000  00000000  000003c3  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str    00000265  00000000  00000000  00000478  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      00000012  00000000  00000000  000006dd  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  000006ef  2**0
                  CONTENTS, READONLY
 11 .debug_frame  00000064  00000000  00000000  00000724  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

main.o file Sections

```
$ arm-none-eabi-objdump.exe -h startup.o

startup.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         000000bc  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  000000f0  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000f0  2**0
                  ALLOC
  3 .vectors      0000001c  00000000  00000000  000000f0  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, DATA
  4 .debug_info   00000168  00000000  00000000  0000010c  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  5 .debug_abbrev 000000c2  00000000  00000000  00000274  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_loc    00000064  00000000  00000000  00000336  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_aranges 00000020  00000000  00000000  0000039a  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_line   000000ae  00000000  00000000  000003ba  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  9 .debug_str    00000162  00000000  00000000  00000468  2**0
                  CONTENTS, READONLY, DEBUGGING
 10 .comment      00000012  00000000  00000000  000005ca  2**0
                  CONTENTS, READONLY
 11 .ARM.attributes 00000033  00000000  00000000  000005dc  2**0
                  CONTENTS, READONLY
 12 .debug_frame  0000004c  00000000  00000000  00000610  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

startup.o file Sections

```
ephraim@DESKTOP-H8B8O6K MINGW32 /d/Learn_in_Depth/Learn_in_Depth
$ 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         000000bc  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  000000f0  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  000000f0  2**0
                  ALLOC
  3 .debug_info   00000145  00000000  00000000  000000f0  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 000000aa  00000000  00000000  00000235  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    00000108  00000000  00000000  000002df  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000020  00000000  00000000  000003e7  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line   00000076  00000000  00000000  00000407  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str    00000211  00000000  00000000  0000047d  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      00000012  00000000  00000000  0000068e  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  000006a0  2**0
                  CONTENTS, READONLY
 11 .debug_frame  000000b4  00000000  00000000  000006d4  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

Alarm_Actuator_Driver.o File Section

```
$ arm-none-eabi-objdump.exe -h AlarmMonitor.o

AlarmMonitor.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         000000e0  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  00000114  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000000  00000000  00000000  00000114  2**0
                  ALLOC
  3 .debug_info   0000014e  00000000  00000000  00000114  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 000000b2  00000000  00000000  00000262  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    000000bc  00000000  00000000  00000314  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000020  00000000  00000000  000003d0  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line   00000073  00000000  00000000  000003f0  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str    000001e5  00000000  00000000  00000463  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      00000012  00000000  00000000  00000648  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  0000065a  2**0
                  CONTENTS, READONLY
 11 .debug_frame  00000080  00000000  00000000  00000690  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

AlarmMonitor.o File Sections

AGENDA

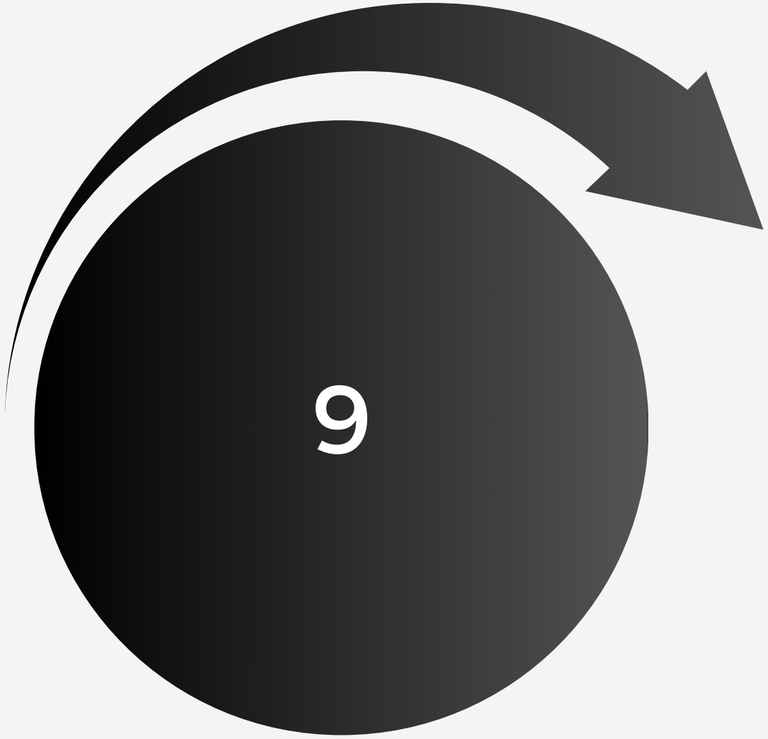
Sections and symbols for each object file

```
$ arm-none-eabi-objdump.exe -h MainAlgorithm.o

MainAlgorithm.o:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         00000070  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000004  00000000  00000000  000000a4  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000004  00000000  00000000  000000a8  2**2
                  ALLOC
  3 .debug_info   0000012f  00000000  00000000  000000a8  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 000000b4  00000000  00000000  000001d7  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    00000064  00000000  00000000  0000028b  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000020  00000000  00000000  000002ef  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line   00000066  00000000  00000000  0000030f  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str    000001db  00000000  00000000  00000375  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      00000012  00000000  00000000  00000550  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  00000562  2**0
                  CONTENTS, READONLY
 11 .debug_frame  00000048  00000000  00000000  00000598  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

MainAlgorithm.o File Section

```
$ 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         00000068  00000000  00000000  00000034  2**2
                  CONTENTS, ALLOC, LOAD, RELOC, READONLY, CODE
  1 .data         00000000  00000000  00000000  0000009c  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00000004  00000000  00000000  0000009c  2**2
                  ALLOC
  3 .debug_info   00000100  00000000  00000000  0000009c  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  4 .debug_abbrev 00000098  00000000  00000000  0000019c  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    00000058  00000000  00000000  00000234  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000020  00000000  00000000  0000028c  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  7 .debug_line   0000007b  00000000  00000000  000002ac  2**0
                  CONTENTS, RELOC, READONLY, DEBUGGING
  8 .debug_str    0000017a  00000000  00000000  00000327  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      00000012  00000000  00000000  000004a1  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  000004b3  2**0
                  CONTENTS, READONLY
 11 .debug_frame  00000048  00000000  00000000  000004e8  2**2
                  CONTENTS, RELOC, READONLY, DEBUGGING
```

Pressure_Sensor_Driver.o File Section



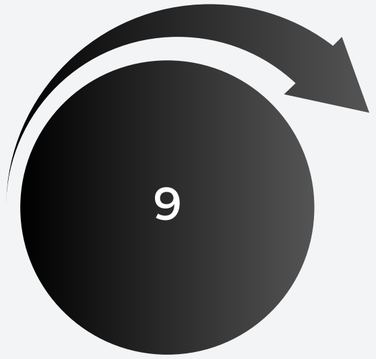
Sections and symbols for Final execution File

AGENDA

Sections and symbols for Final execution File

```
$ arm-none-eabi-objdump.exe -h learn_in_depth_Project1.elf

learn_in_depth_Project1.elf:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         000016fc  08000000  08000000  00008000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000008  20000000  080016fc  00010000  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00001028  20000008  08001704  00010008  2**2
                  ALLOC
  3 .debug_info   00000e0b  00000000  00000000  00010008  2**0
                  CONTENTS, READONLY, DEBUGGING
  4 .debug_abbrev 000006a1  00000000  00000000  00010e13  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    00000848  00000000  00000000  000114b4  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 00000120  00000000  00000000  00011cfc  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_line   0000066c  00000000  00000000  00011e1c  2**0
                  CONTENTS, READONLY, DEBUGGING
  8 .debug_str    00000848  00000000  00000000  00012488  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      00000011  00000000  00000000  00012cd0  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  00012ce1  2**0
                  CONTENTS, READONLY
 11 .debug_frame  0000052c  00000000  00000000  00012d14  2**2
                  CONTENTS, READONLY, DEBUGGING
```

Sections for elf File

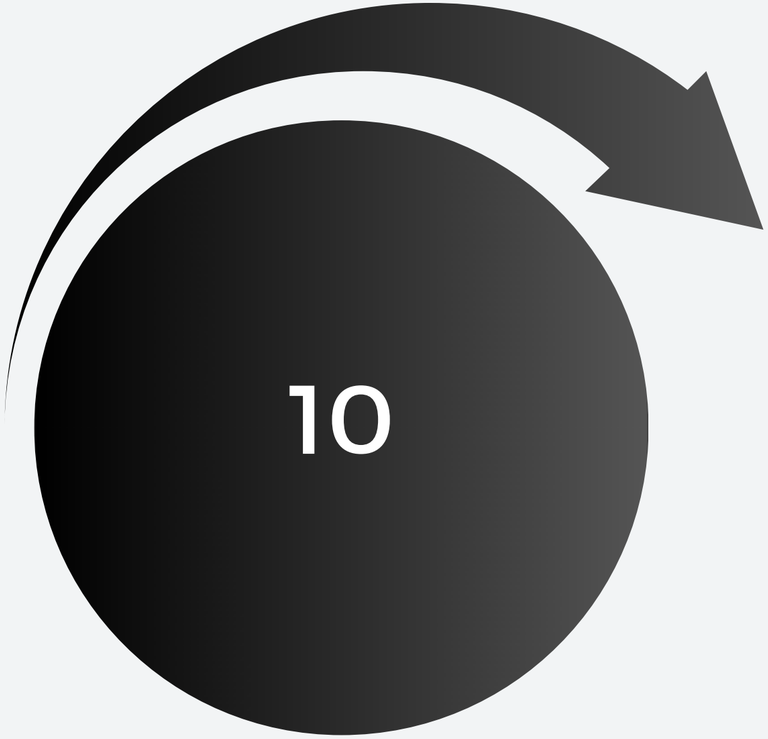


Sections and symbols for Final execution File

Symbol Table for elf File

```
08000de0 W NMI_Handler
08001150 T PressureSensor_init
2000101c B PressureSensor_state
20001020 B PressureSensor_state_id
08000e9c T PressureSensorDriver_setPressureVal
2000000c b PressureSensorVal
080010b0 T RCC_u8DisableClock
0800101c T RCC_u8EnableClock
08000f0c T RCC_voidEnableCLKs
08000ff0 T RCC_voidInitSystemClk
08000f9c T RCC_voidSetMCOSource
08000f60 T RCC_voidSetSystemClk
08000dec T Reset_Handler
080011d4 T setup
080015e8 T ST_AlarmActuatorDriver_OFF
080015c0 T ST_AlarmActuatorDriver_ON
08001610 T ST_AlarmActuatorDriver_Waiting
08001698 T ST_AlarmMonitor_OFF
08001668 T ST_AlarmMonitor_ON
080016c4 T ST_AlarmMonitor_Waiting
08000ebc T ST_MainAlgorithm_HighPressureDetect
08001168 T ST_PressureSensor_Reading
20000004 D STK
20001024 B STKCallBack
08001520 T SysTick_Handler
08000de0 W Usage_Fault_Handler
08000000 T vectors
```

```
$ arm-none-eabi-nm.exe learn_in_depth_Project1.elf
20000011 B _E_bss
20000008 D _E_DATA
080016fc T _E_text
20000008 B _S_bss
20000000 D _S_DATA
20001014 B _stack_top
08001560 T AlarmActuatorDriver_init
20001028 B AlarmActuatorDriver_state
20001022 B AlarmActuatorDriver_state_id
08001578 T AlarmMonitor_StartAlarm
2000102c B AlarmMonitor_state
20001021 B AlarmMonitor_state_id
0800159c T AlarmMonitor_StopAlarm
08000de0 W Bus_Fault
08000de0 T Default_Handler
20000010 B Gobal_u8IntervalMode
08000de0 W H_Fault_Handler
080011b8 T HW_voidInit
08001230 T main
0800161c T MainAlgorithm_HighPressureDetected
20001014 B MainAlgorithm_state
20001018 B MainAlgorithm_state_id
20000008 b MainAlgorithmPressureVal
20000000 d MainAlgorithmThreshold
08000d18 T MGPIO_voidGetByteVal
08000394 T MGPIO_voidGetPinVal
08000410 T MGPIO_voidSetByteDir
080004c0 T MGPIO_voidSetByteVal
080006e8 T MGPIO_voidSetNibbleDir
08000a14 T MGPIO_voidSetNibbleVal
0800001c T MGPIO_voidSetPinDir
08000260 T MGPIO_voidSetPinVal
08000658 T MGPIO_voidTogglePin
08000de0 W MM_Fault_Handler
080014cc T MSTK_u32GetElapsedTime
080014fc T MSTK_u32GetRemainingTime
08001260 T MSTK_voidInit
08001358 T MSTK_voidResetTimerVal
080012c0 T MSTK_voidSetBusyWait
0800144c T MSTK_voidSetIntervalPeriodic
080013a0 T MSTK_voidSetIntervalSingle
08001374 T MSTK_voidStopTimer
```



Hardware Simulation

Prove of Concept Simulation using C Code with Native ToolChain

10

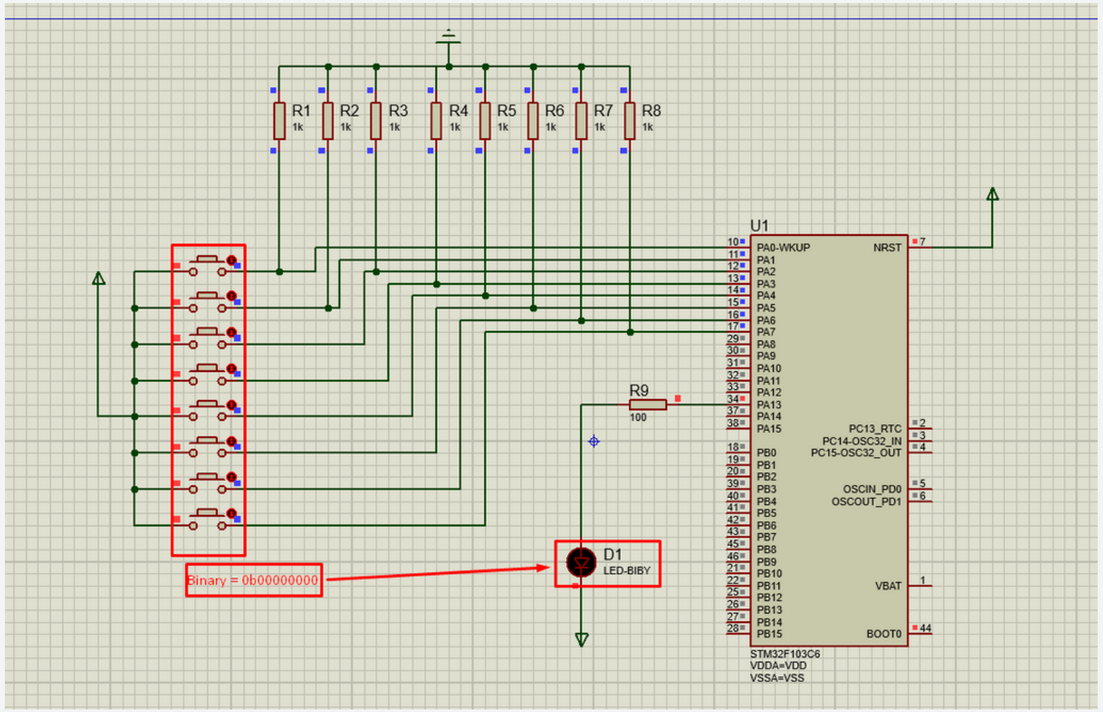
Hardware Simulation

```
============================================================

Pressure Sensor Val --> 19
Set Main Algorithm Pressure Value =19
19 < Main Algorithm Threshold
AlarmMonitor_StopAlarm
Alarm OFF
============================================================

Pressure Sensor Val --> 20
Set Main Algorithm Pressure Value =20
20 < Main Algorithm Threshold
AlarmMonitor_StopAlarm
Alarm OFF
============================================================

Pressure Sensor Val --> 17
Set Main Algorithm Pressure Value =17
17 < Main Algorithm Threshold
AlarmMonitor_StopAlarm
Alarm OFF
============================================================

Pressure Sensor Val --> 18
Set Main Algorithm Pressure Value =18
18 < Main Algorithm Threshold
AlarmMonitor_StopAlarm
Alarm OFF
============================================================

Pressure Sensor Val --> 15
Set Main Algorithm Pressure Value =15
15 < Main Algorithm Threshold
AlarmMonitor_StopAlarm
Alarm OFF
============================================================

Pressure Sensor Val --> 22
Set Main Algorithm Pressure Value =22
22 > Main Algorithm Threshold
AlarmMonitor_StartAlarm
Alarm ON
Set Timer
Turn on Waiting State
Waiting Till Timer Expired
Reset Timer
```

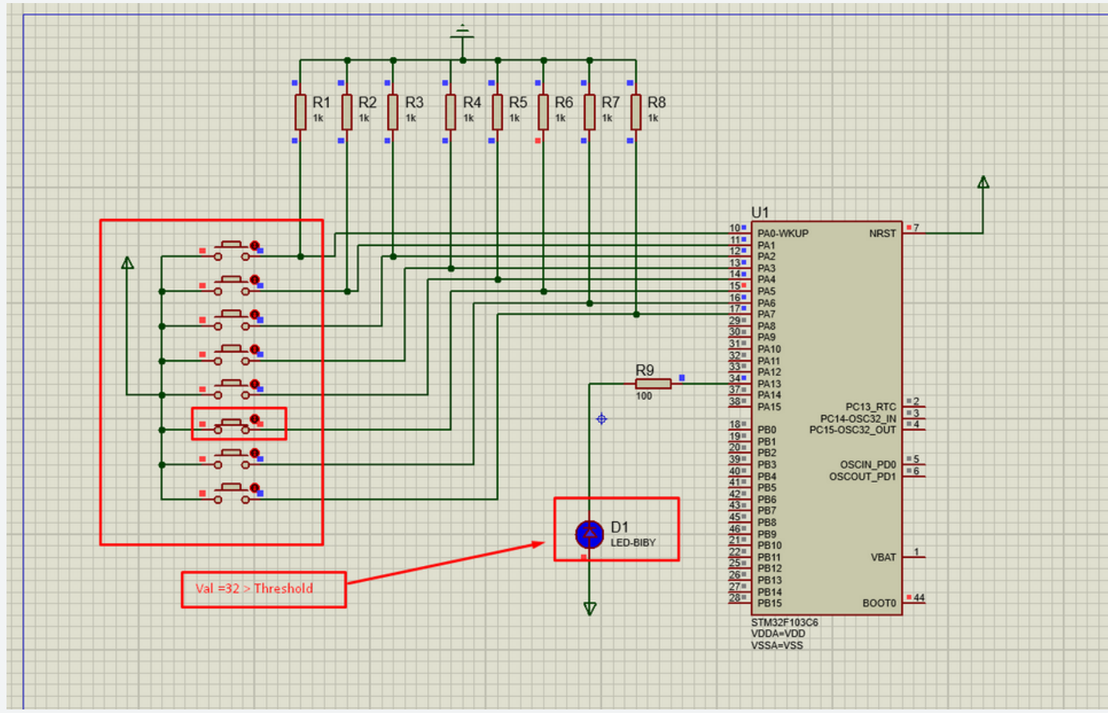
```
============================================================

Pressure Sensor Val --> 20
Set Main Algorithm Pressure Value =20
20 < Main Algorithm Threshold
AlarmMonitor_StopAlarm
Alarm OFF
============================================================

Pressure Sensor Val --> 21
Set Main Algorithm Pressure Value =21
21 > Main Algorithm Threshold
AlarmMonitor_StartAlarm
Alarm ON
Set Timer
Turn on Waiting State
Waiting Till Timer Expired
Reset Timer
Stop Alarm
Alarm OFF
============================================================

Pressure Sensor Val --> 19
Set Main Algorithm Pressure Value =19
19 < Main Algorithm Threshold
AlarmMonitor_StopAlarm
Alarm OFF
============================================================

Pressure Sensor Val --> 24
Set Main Algorithm Pressure Value =24
24 > Main Algorithm Threshold
AlarmMonitor_StartAlarm
Alarm ON
Set Timer
Turn on Waiting State
Waiting Till Timer Expired
Reset Timer
Stop Alarm
Alarm OFF
============================================================
```

Proteus Simulation

Hardware Simulation



Push Button Value =0
Less Than the threshold
so the LED OFF

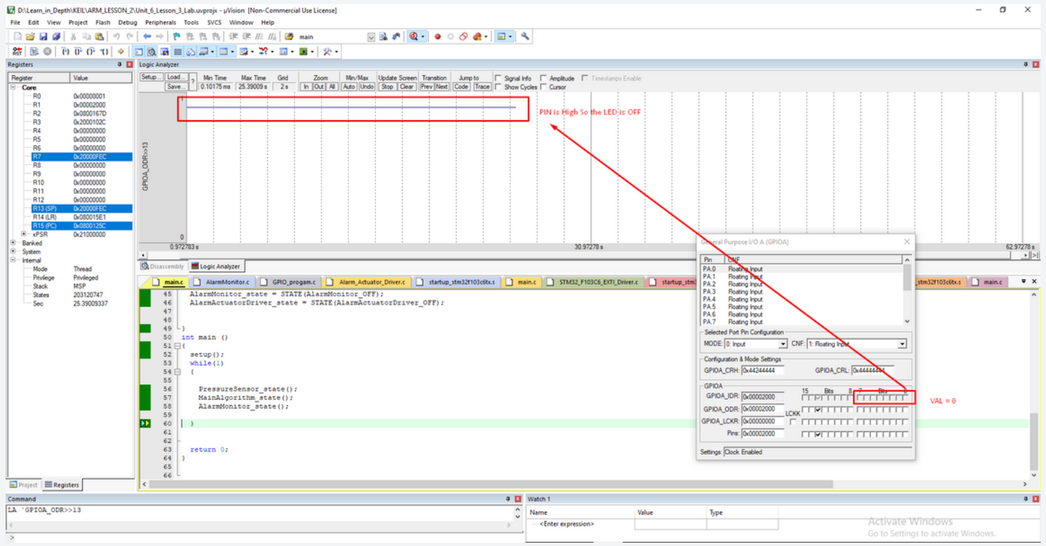


Push Button Value =32
More Than the threshold
so the LED ON

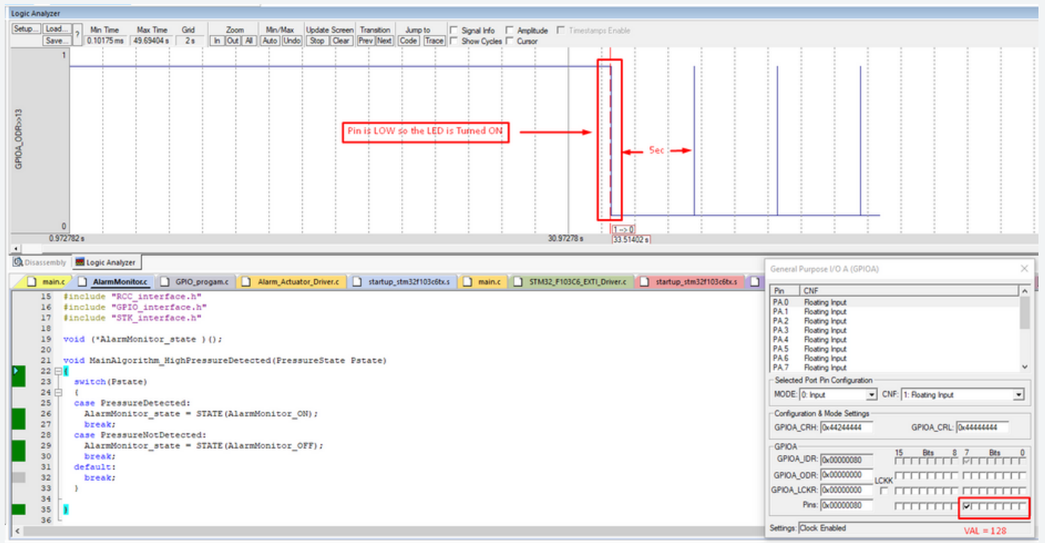
AGENDA

Keil Simulation

Hardware Simulation



Push Button Value =0
Less Than the threshold
so the LED OFF



Push Button Value =128
More Than the threshold
so the LED ON

THANK YOU

AGENDA