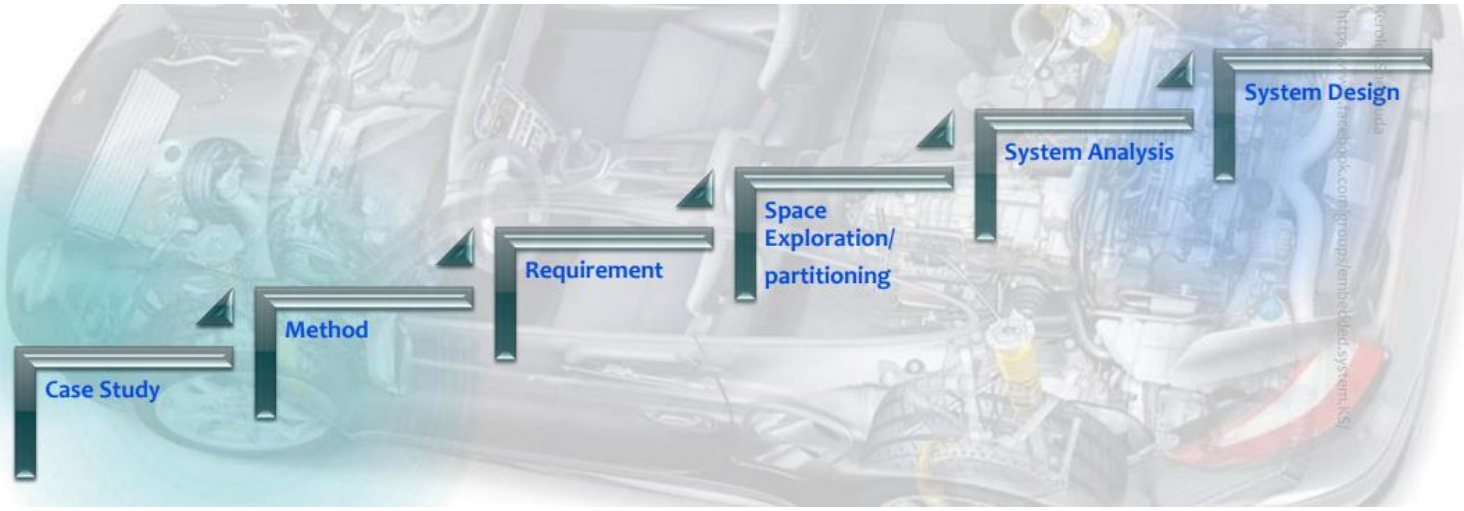
Pressure Detection Report

First Term Final Project 1

Eng. Youssef Mohamed Samy
My Profile: https://www.learn-in-depth.com/online-diploma/youssef.samy12345678%40gmail.com-

System Architecting/Design Sequence:



1- Case Study:

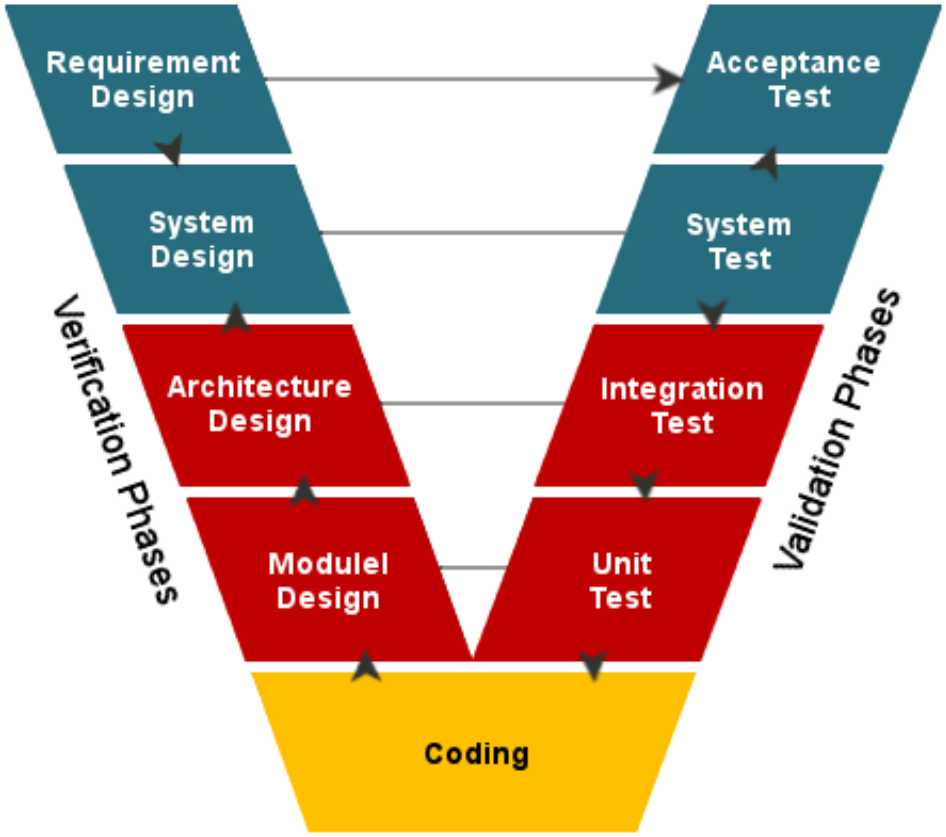
A Pressure Controller informs the crew of a cabin with an alarm when the pressure exceeds 20 bars in the cabin. The alarm duration equals 60 seconds.

Assumptions:

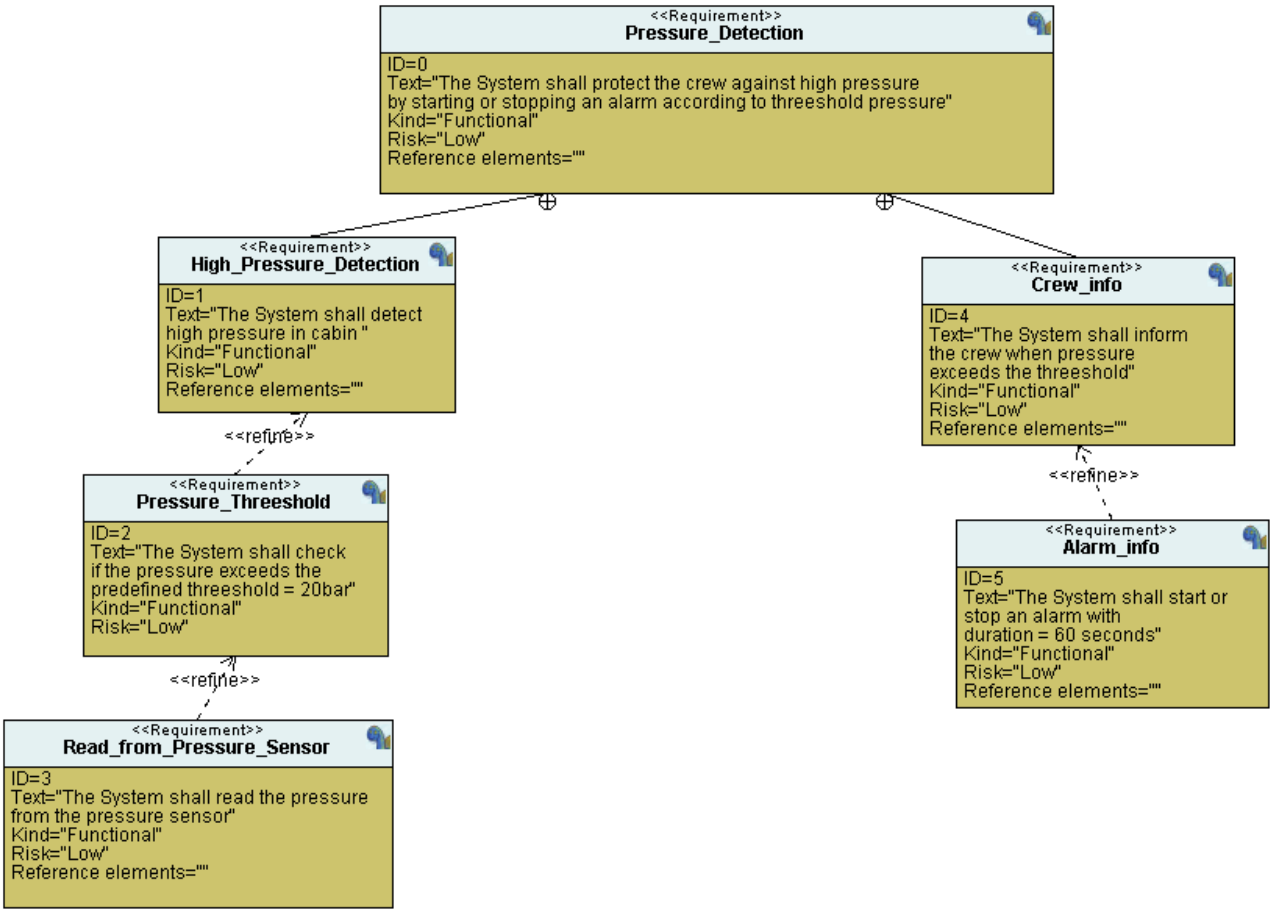
1- The system setup and shutdown procedures are not modeled.
2- The system maintenance is not modeled.
3- The pressure sensor never fails.
4- The alarm never fails.
5- The system never faces power cuts.
6- Store in Flash is not modeled in any diagram.

2-Method: V Model

The V-model is an SDLC model where execution of processes happens in a sequential manner in a V-shape. It is also known as Verification and Validation model.



3-Requirement Diagram:



Note:

Store data in flash is optional, may be implemented in future versions.

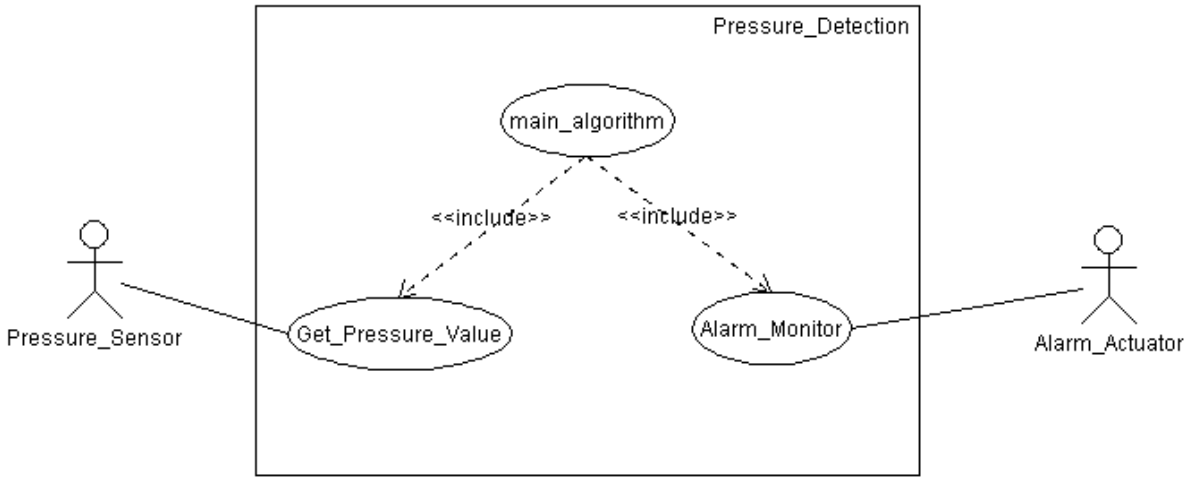
4-Space Exploration/Partitioning:

Hardware will be **STM32F103C6 MCU**

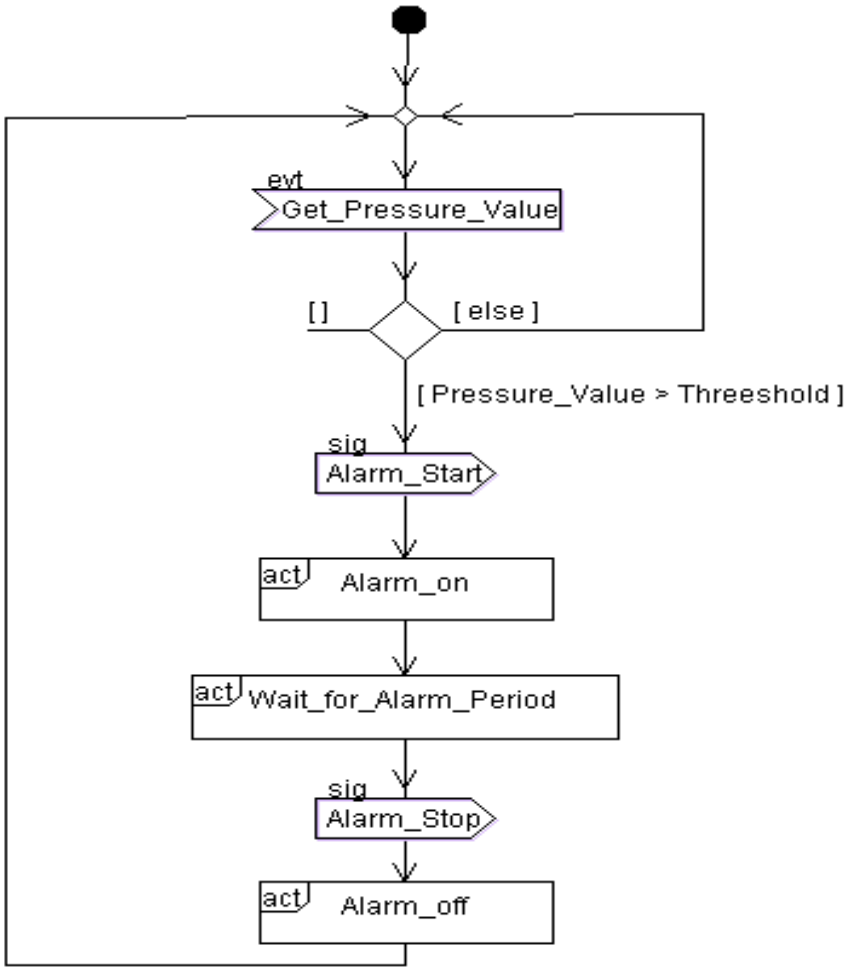
Based on **ARM-Cortex-M3** Processor

5-System Analysis:

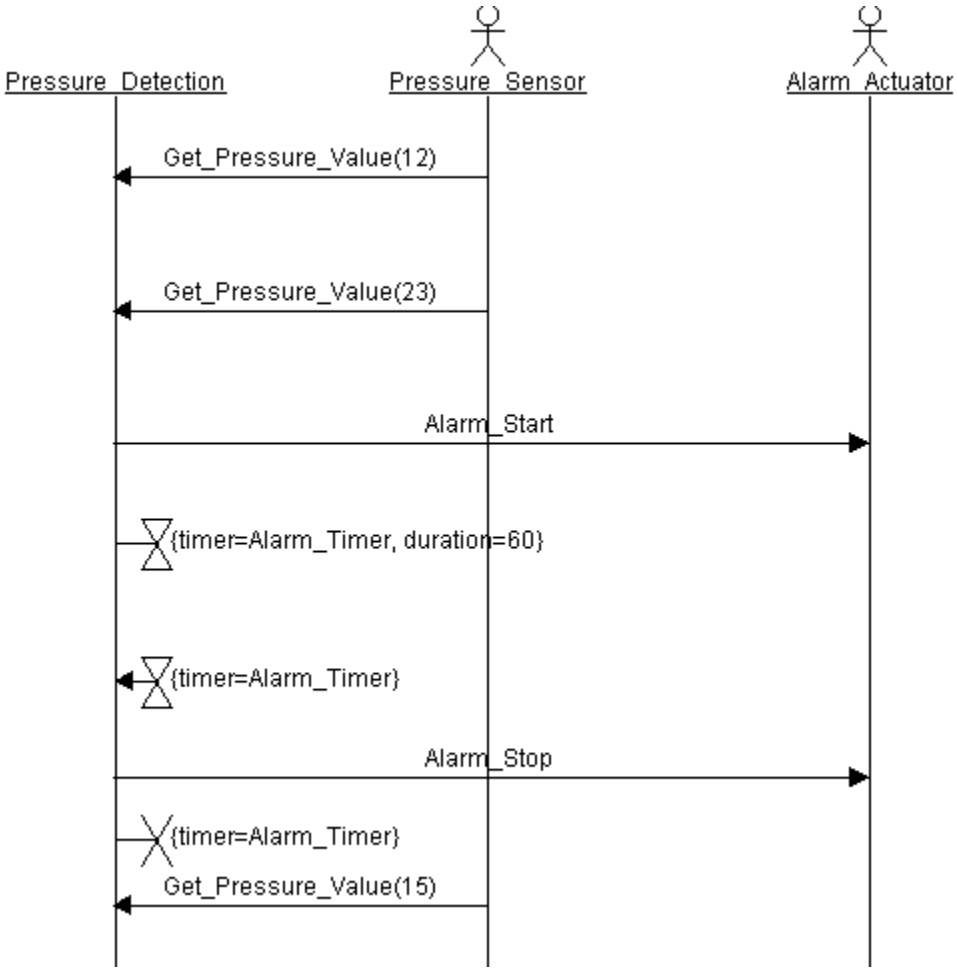
5.1) Use Case Diagram:



5.2) Activity Diagram:

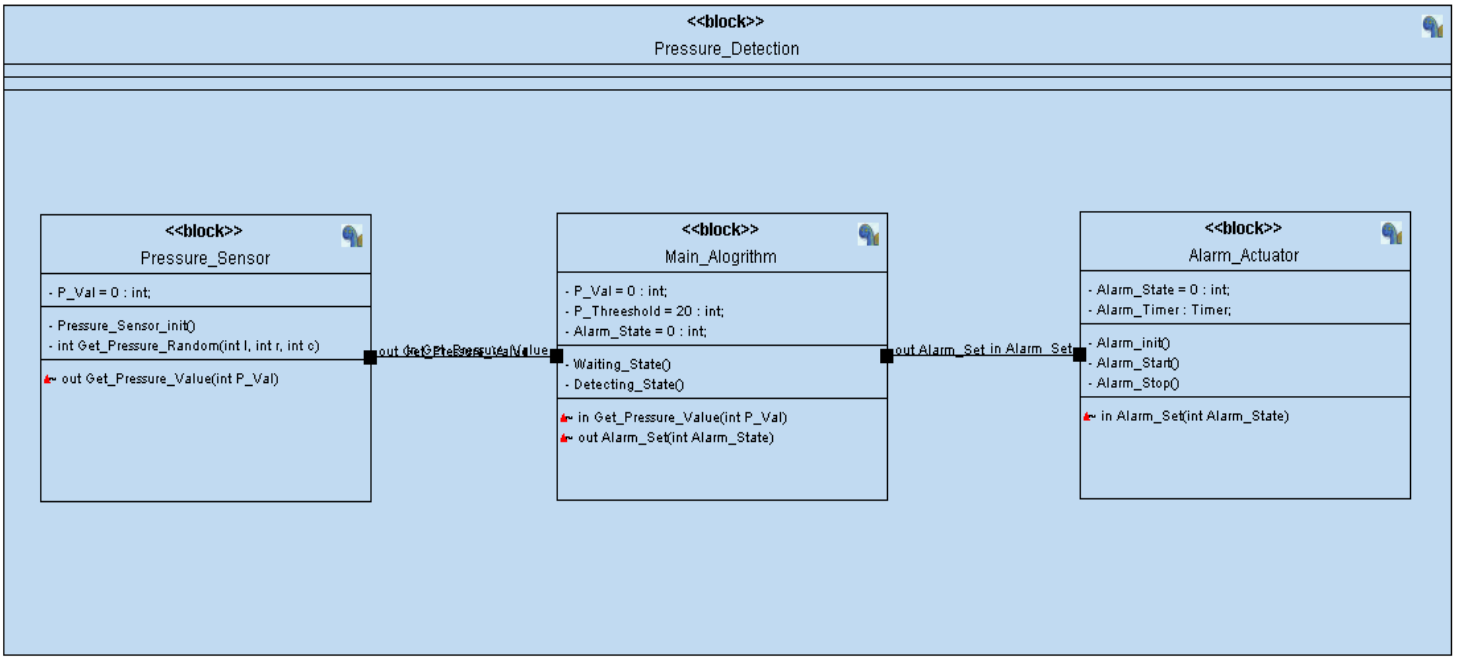


5.3) Sequence Diagram:



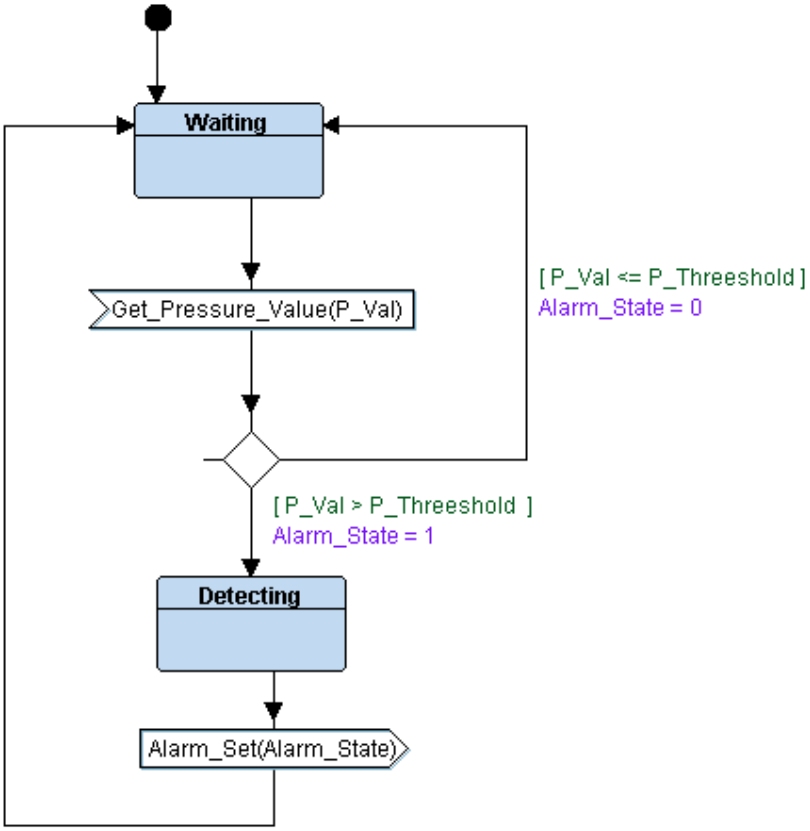
6-System Design:

6.1) System Block Diagram:

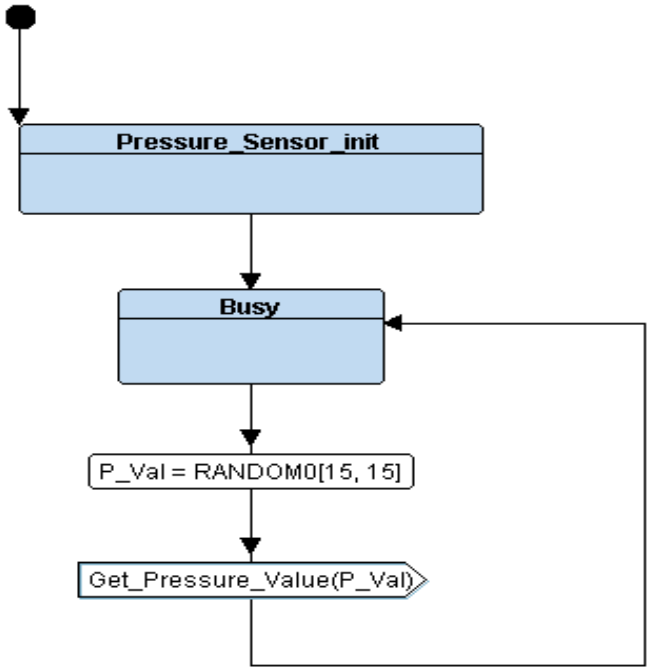


6.2) System Flow Charts:

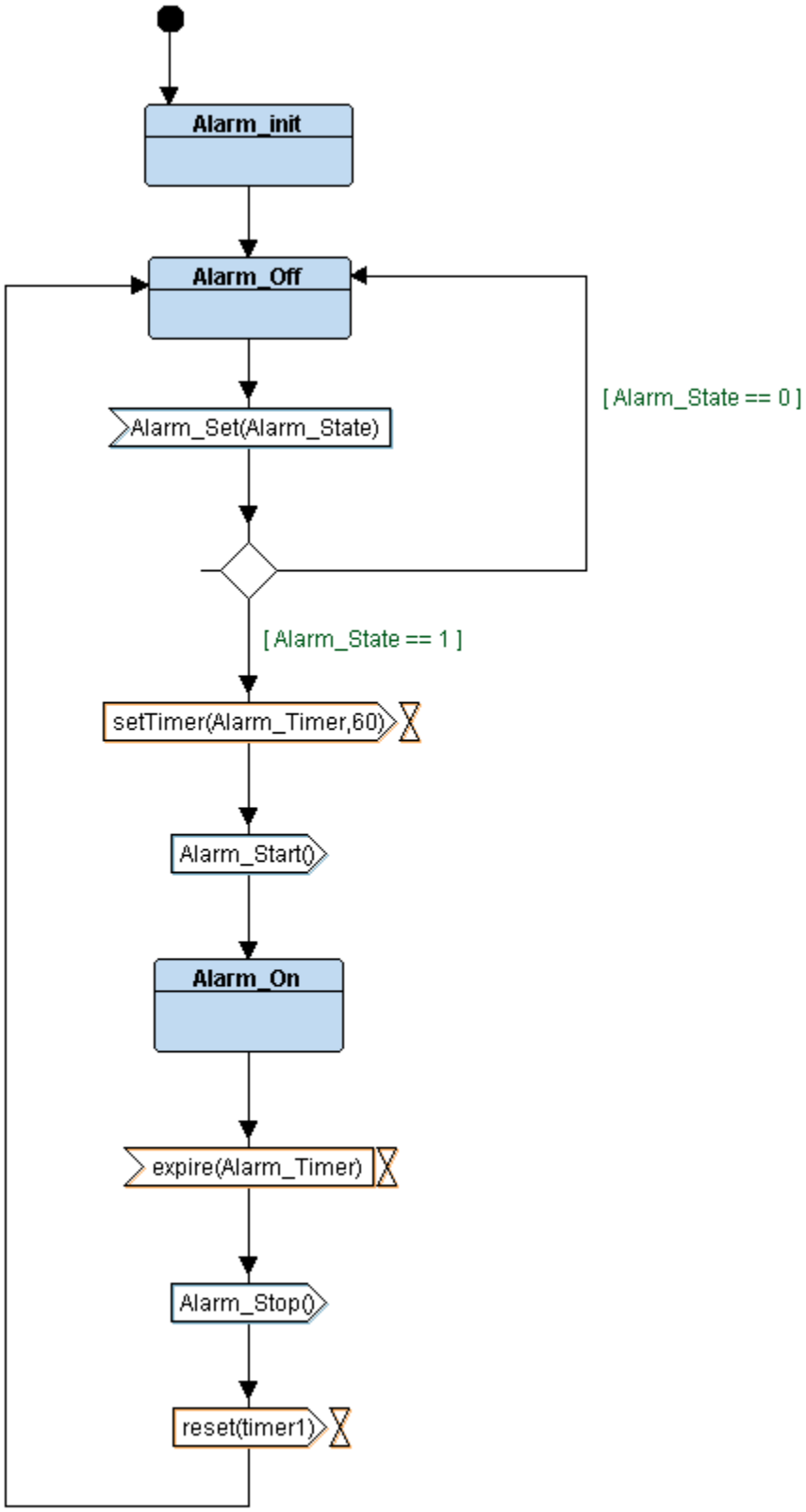
6.2.1) Main Algorithm:



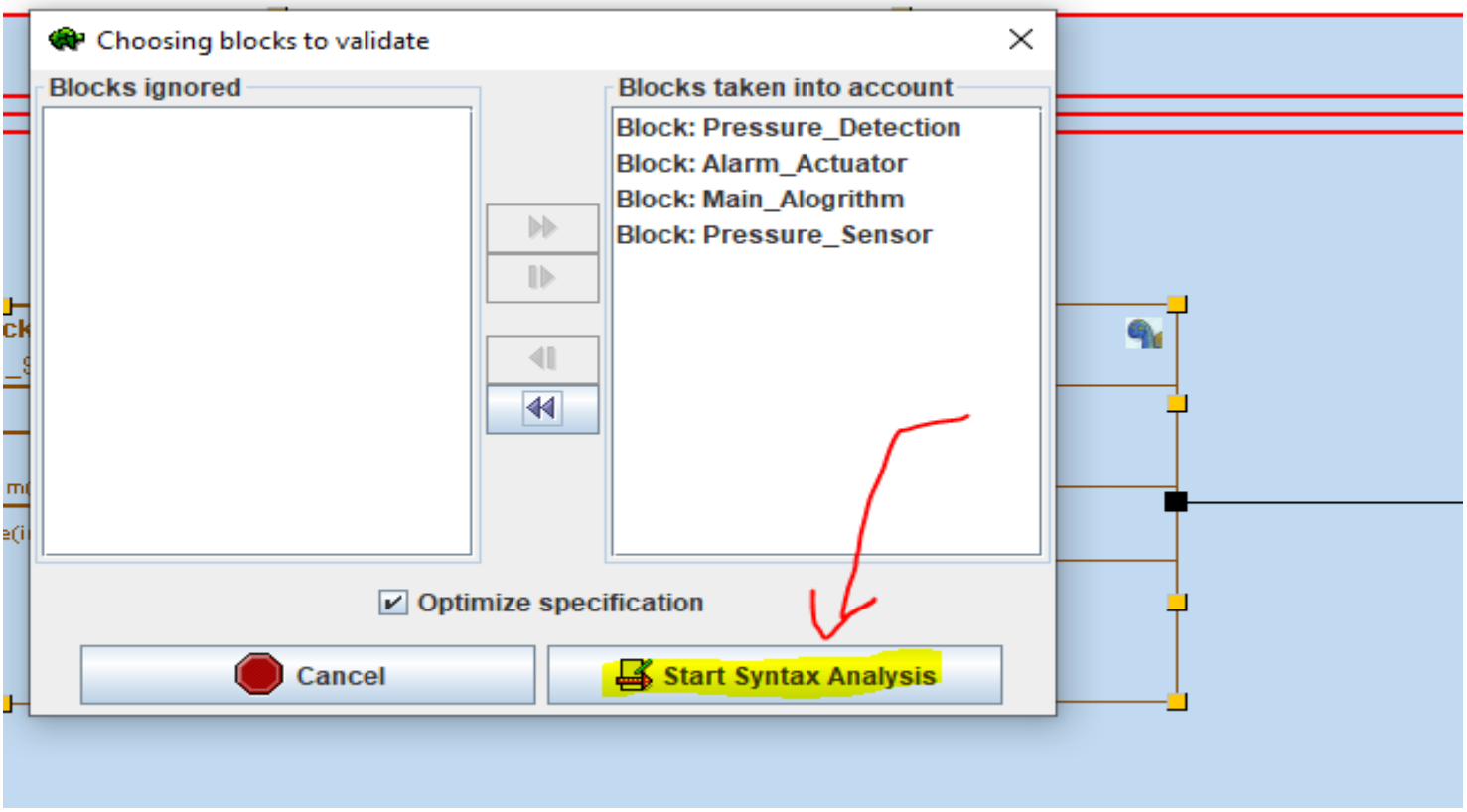
6.2.2) Pressure Sensor:



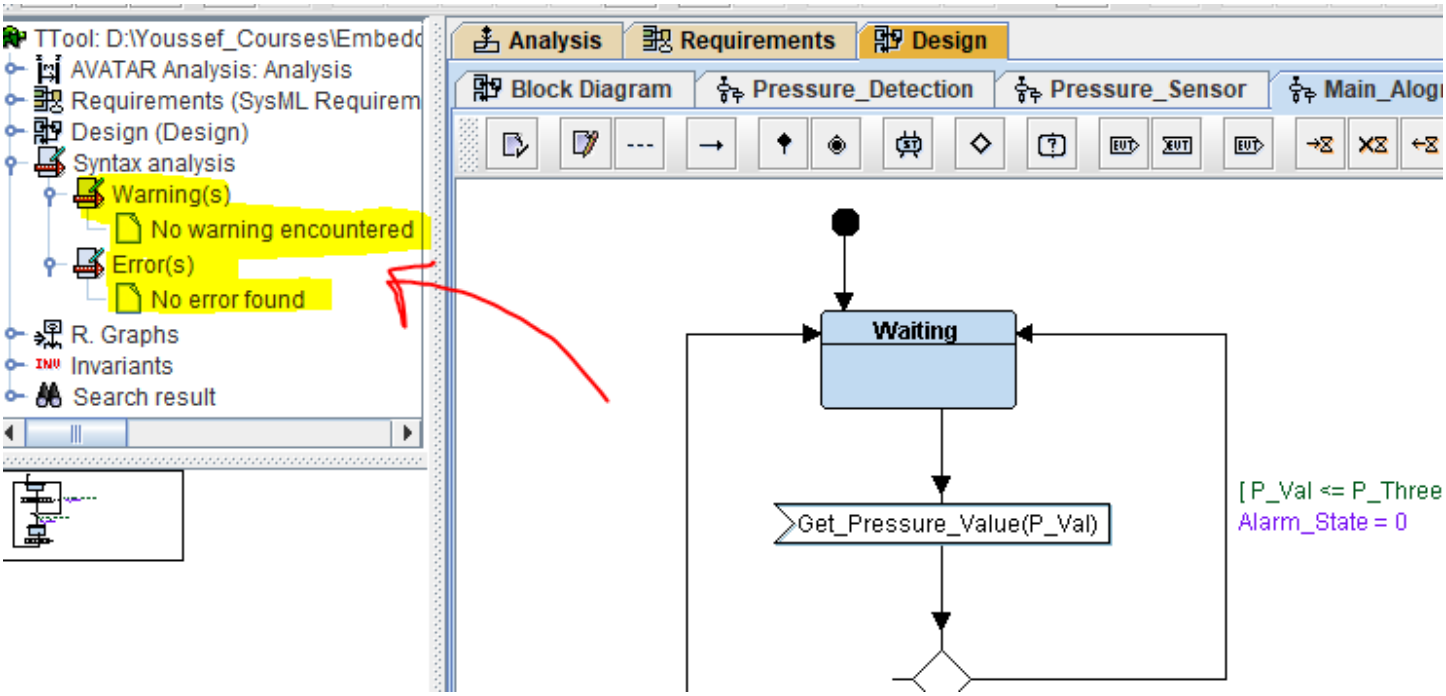
6.2.3) Alarm Actuator:



Check Syntax and Logic Errors (TTool):



NO ERRORS



5-Software & Testing:

Project Files:

PC › Data (D:) › Youssef_Courses › Embedded-Systems › Keroles_Diploma › First Term Final Projects › Pressure_Detection › Code

Name	Date modified	Type	Size
Proteus Simulation	9/17/2023 5:30 AM	File folder	
Alarm_Actuator.c	9/16/2023 4:39 AM	C Source File	2 KB
Alarm_Actuator.h	9/16/2023 4:39 AM	C Header Source F...	1 KB
BIT_MATH.h	9/15/2023 11:03 PM	C Header Source F...	1 KB
GPIO.c	9/15/2023 11:57 PM	C Source File	1 KB
GPIO.h	9/15/2023 11:58 PM	C Header Source F...	1 KB
linker_script.ld	8/30/2023 1:03 AM	LD File	1 KB
main.c	9/16/2023 4:40 AM	C Source File	1 KB
Main_Algorithm.c	9/16/2023 4:40 AM	C Source File	1 KB
Main_Algorithm.h	9/16/2023 3:02 AM	C Header Source F...	1 KB
makefile	9/16/2023 5:09 AM	File	1 KB
Platform_Types.h	9/15/2023 11:15 PM	C Header Source F...	1 KB
Pressure_Sensor.c	9/16/2023 4:37 AM	C Source File	2 KB
Pressure_Sensor.h	9/16/2023 4:37 AM	C Header Source F...	1 KB
startup.c	9/16/2023 4:56 AM	C Source File	2 KB
State.h	9/16/2023 4:34 AM	C Header Source F...	1 KB

Makefile (to Automate the Building Process):

PC › Data (D:) › Youssef_Courses › Embedded-Systems › Keroles_Diploma › First Term Final Projects › Pressure_Detection › Code

Name	Date modified	Type	Size
Alarm_Actuator.c	9/16/2023 4:39 AM	C Source File	2 KB
Alarm_Actuator.h			
Alarm_Actuator.o			
BIT_MATH.h			
GPIO.c			
GPIO.h			
GPIO.o			
linker_script.ld			
main.c			
main.o			
Main_Algorithm.c			
Main_Algorithm.h			
Main_Algorithm.o			
makefile			
Map_file.map			
Platform_Types.h			
Pressure_Detection_STM32.bin			
Pressure_Detection_STM32.elf			
Pressure_Sensor.c			
Pressure_Sensor.h			
Pressure_Sensor.o			
startup.c			
startup.o	9/16/2023 5:09 AM	O File	5 KB
State.h	9/16/2023 4:34 AM	C Header Source F...	1 KB

MINGW32:/d/Youssef_Courses/Embedded-Systems/Keroles_Diploma/First Term Final Projects/Pressure_Detection/Code

```
youssef@DESKTOP-OB4M27D MINGW32 /d/Youssef_Courses/Embedded-Systems/Keroles_Dipl
oma/First Term Final Projects/Pressure_Detection/Code
$ mingw32-make.exe
arm-none-eabi-gcc.exe -mcpu=cortex-m3 -gdwarf-2 -g   -I . -c startup.c -o startu
p.o
arm-none-eabi-gcc.exe -mcpu=cortex-m3 -gdwarf-2 -g   -I . -c Pressure_Sensor.c -
o Pressure_Sensor.o
arm-none-eabi-gcc.exe -mcpu=cortex-m3 -gdwarf-2 -g   -I . -c GPIO.c -o GPIO.o
arm-none-eabi-gcc.exe -mcpu=cortex-m3 -gdwarf-2 -g   -I . -c main.c -o main.o
arm-none-eabi-gcc.exe -mcpu=cortex-m3 -gdwarf-2 -g   -I . -c Alarm_Actuator.c -o
 Alarm_Actuator.o
arm-none-eabi-gcc.exe -mcpu=cortex-m3 -gdwarf-2 -g   -I . -c Main_Algorithm.c -o
 Main_Algorithm.o
arm-none-eabi-ld.exe -T linker_script.ld  startup.o Pressure_Sensor.o GPIO.o mai
n.o Alarm_Actuator.o Main_Algorithm.o  -o Pressure_Detection_STM32.elf -Map=Map_
file.map
arm-none-eabi-objcopy.exe -O binary Pressure_Detection_STM32.elf Pressure_Detect
ion_STM32.bin
==============> Building is Done <==============

youssef@DESKTOP-OB4M27D MINGW32 /d/Youssef_Courses/Embedded-Systems/Keroles_Dipl
oma/First Term Final Projects/Pressure_Detection/Code
$
```

Object Files Symbols: (using “nm” Binary Utility)

Startup:

```
youssef@DESKTOP-OB4M27D MINGW32 /d/Youssef_Courses/Embedded-Systems/Keroles_Diploma/First Term
Final Projects/Pressure_Detection/Code
$ 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
```

GPIO:

```
youssef@DESKTOP-OB4M27D MINGW32 /d/Youssef_Courses/Embedded-Systems/Keroles_Diploma/First Term
Final Projects/Pressure_Detection/Code
$ arm-none-eabi-nm.exe GPIO.o
00000050 T GPIO_Delay
00000070 T GPIO_Get_Pressure_Value
00000000 T GPIO_Init
00000088 T GPIO_Start_Alarm_Actuator
000000a8 T GPIO_Stop_Alarm_Actuator
```

Pressure Sensor:

```
youssef@DESKTOP-OB4M27D MINGW32 /d/Youssef_Courses/Embedded-Systems/Keroles_Diploma/First Term
Final Projects/Pressure_Detection/Code
$ arm-none-eabi-nm.exe Pressure_Sensor.o
         U GPIO_Get_Pressure_Value
00000000 b P_Val
00000038 T PS_Init
00000004 b PS_State_ID
00000004 C ptr_PS_State
         U Set_Pressure_Val
00000000 T ST_PS_BUSY
```

Main Algorithm:

```
youssef@DESKTOP-OB4M27D MINGW32 /d/Youssef_Courses/Embedded-Systems/Keroles_Diploma/First Term
Final Projects/Pressure_Detection/Code
$ arm-none-eabi-nm.exe Main_Algorithm.o
00000004 b Alarm_State
         U Alarm_State_Set
00000008 b MA_State_ID
00000000 d P_threeshold
00000000 b P_Val
00000004 C ptr_Alarm_Actuator_State
00000004 C ptr_MA_state
00000050 T Set_Pressure_Val
00000028 T ST_MA_Detecting
00000000 T ST_MA_Waiting
```

Alarm Actuator:

```
youssef@DESKTOP-OB4M27D MINGW32 /d/Youssef_Courses/Embedded-Systems/Keroles_Diploma/First Term
Final Projects/Pressure_Detection/Code
$ arm-none-eabi-nm.exe Alarm_Actuator.o
00000048 T Alarm_Actuator_Init
00000004 b Alarm_Actuator_State_ID
00000000 b Alarm_State
00000054 T Alarm_State_Set
         U GPIO_Start_Alarm_Actuator
         U GPIO_Stop_Alarm_Actuator
00000004 C ptr_Alarm_Actuator_State
00000024 T ST_Alarm_Actuator_BUSY
00000000 T ST_Alarm_Actuator_IDLE
```

Main:

```
youssef@DESKTOP-OB4M27D MINGW32 /d/Youssef_Courses/Embedded-Systems/Keroles_Diploma/First Term
Final Projects/Pressure_Detection/Code
$ arm-none-eabi-nm.exe main.o
         U Alarm_Actuator_Init
         U GPIO_Init
00000040 T main
         U PS_Init
00000004 C ptr_Alarm_Actuator_State
00000004 C ptr_MA_state
00000004 C ptr_PS_State
00000000 T setup
         U ST_Alarm_Actuator_IDLE
         U ST_MA_Waiting
         U ST_PS_BUSY
```

All the Combined Symbols in the .elf file:

```
youssef@DESKTOP-OB4M27D MINGW32 /d/Youssef_Courses/Embedded-Systems/Keroles_Dipl
oma/First Term Final Projects/Pressure_Detection/Code
$ arm-none-eabi-nm.exe Pressure_Detection_STM32.elf
20000020 B _E_bss
20000004 D _E_data
0800036c T _E_text
20000004 B _S_bss
20000000 D _S_data
20001020 B _stack_top
08000284 T Alarm_Actuator_Init
20000010 b Alarm_Actuator_State_ID
2000000c b Alarm_State
20000018 b Alarm_State
08000290 T Alarm_State_Set
0800001c W Bus_Fault
0800001c T Default_Handler
08000148 T GPIO_Delay
08000168 T GPIO_Get_Pressure_Value
080000f8 T GPIO_Init
08000180 T GPIO_Start_Alarm_Actuator
080001a0 T GPIO_Stop_Alarm_Actuator
0800001c W H_Fault_Handler
2000001c b MA_State_ID
080001fc T main
0800001c W MM_Fault_Handler
0800001c W NMI_Handler
20000000 d P_threeshold
20000004 b P_Val
20000014 b P_Val
080000ec T PS_Init
20000008 b PS_State_ID
20001028 B ptr_Alarm_Actuator_State
20001024 B ptr_MA_state
20001020 B ptr_PS_State
08000028 T Reset_Handler
08000324 T Set_Pressure_Val
080001bc T setup
08000260 T ST_Alarm_Actuator_BUSY
0800023c T ST_Alarm_Actuator_IDLE
080002fc T ST_MA_Detecting
080002d4 T ST_MA_Waiting
080000b4 T ST_PS_BUSY
0800001c W Usage_Fault_Handler
08000000 T vectors
```

Sections in .elf file:

(by passing '-h' command to Binary Utility "objdump")

```
youssef@DESKTOP-OB4M27D MINGW32 /d/Youssef_Courses/Embedded-Systems/Keroles_Diploma/First Term Final Projects/Pressure_Detection/Code
$ arm-none-eabi-objdump.exe -h Pressure_Detection_STM32.elf

Pressure_Detection_STM32.elf:     file format elf32-littlearm

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         0000036c  08000000  08000000  00010000  2**2
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000004  20000000  0800036c  00020000  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  2 .bss          00001028  20000004  08000370  00020004  2**2
                  ALLOC
  3 .debug_info   0000070e  00000000  00000000  00020004  2**0
                  CONTENTS, READONLY, DEBUGGING
  4 .debug_abbrev 00000449  00000000  00000000  00020712  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_loc    00000474  00000000  00000000  00020b5b  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_aranges 000000c0  00000000  00000000  00020fcf  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_line   0000031b  00000000  00000000  0002108f  2**0
                  CONTENTS, READONLY, DEBUGGING
  8 .debug_str    000003c2  00000000  00000000  000213aa  2**0
                  CONTENTS, READONLY, DEBUGGING
  9 .comment      0000007e  00000000  00000000  0002176c  2**0
                  CONTENTS, READONLY
 10 .ARM.attributes 00000033  00000000  00000000  000217ea  2**0
                  CONTENTS, READONLY
 11 .debug_frame  000002a0  00000000  00000000  00021820  2**2
                  CONTENTS, READONLY, DEBUGGING
```

Testing:

To be able to test through terminal by using printf function,
I made a function that generates Random values in range **[15:25]**,
to act as the Pressure Sensor readings. Since the threshold Pressure is **20Bar** as mentioned in the requirements, there are many test cases that can be checked on the terminal.

```
Select C:\Windows\System32\cmd.exe - out
Microsoft Windows [Version 10.0.19045.3448]
(c) Microsoft Corporation. All rights reserved.

D:\Youssef_Courses\Embedded-Systems\Keroles_Diploma\First Term Final Projects\Pressure_Detection\Code>gcc main.c Alarm_Actuator.c Pressure_Sensor.c Main_Algorithm.c -o
out.exe

D:\Youssef_Courses\Embedded-Systems\Keroles_Diploma\First Term Final Projects\Pressure_Detection\Code>out


==========> Pressure Sensor Driver Initialization <==========
==========> Alarm Actuator Driver Initialization <==========


PS_BUSY state : Pressure= 23
Pressure Sensor ========> Pressure=23 ========> Main Algorithm


MA_Waiting state : Pressure=23
Main Algorithm ==========> State=1 ==========> Alarm_Actuator


Alarm Actuator BUSY State : State=1


PS_BUSY state : Pressure= 24
Pressure Sensor ========> Pressure=24 ========> Main Algorithm


MA_Waiting state : Pressure=24
Main Algorithm ==========> State=1 ==========> Alarm_Actuator


Alarm Actuator BUSY State : State=1


PS_BUSY state : Pressure= 24
Pressure Sensor ========> Pressure=24 ========> Main Algorithm
```

Case1: Pressure <= Threshold:

```
PS_BUSY state : Pressure= 16
Pressure Sensor ========> Pressure=16 ========> Main Algorithm


MA_Waiting state : Pressure=16
Main Algorithm ==========> State=0 ==========> Alarm_Actuator


Alarm Actuator BUSY State : State=0
```

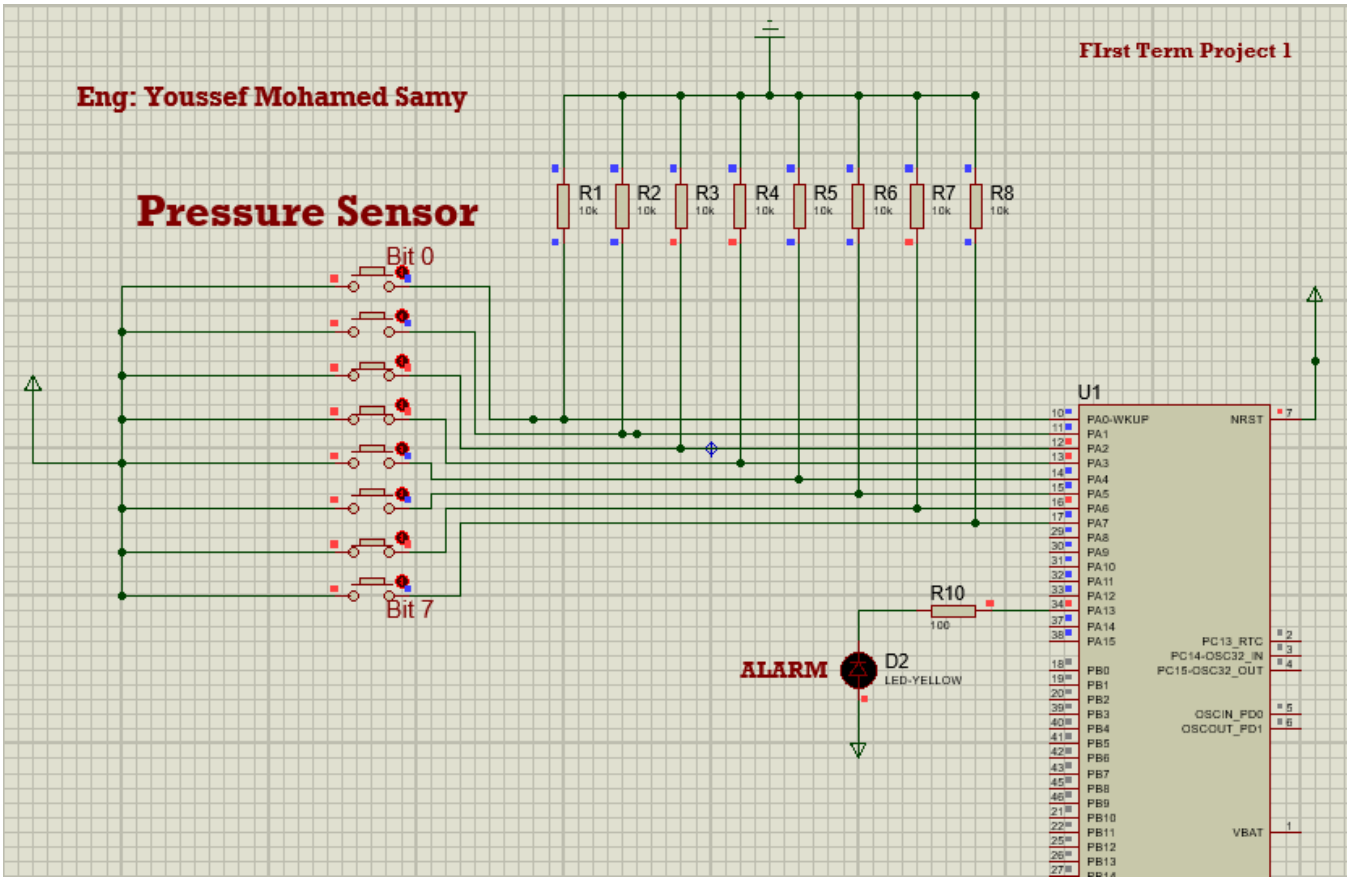
In this case, the Pressure = 16 which is less than the predefined threshold (20Bar).
So The Pressure Sensor sent this value to the Main Algorithm
and the Main Algorithm sent to Alarm Actuator “State=0” which states for **Alarm OFF** (Alarm Stop).

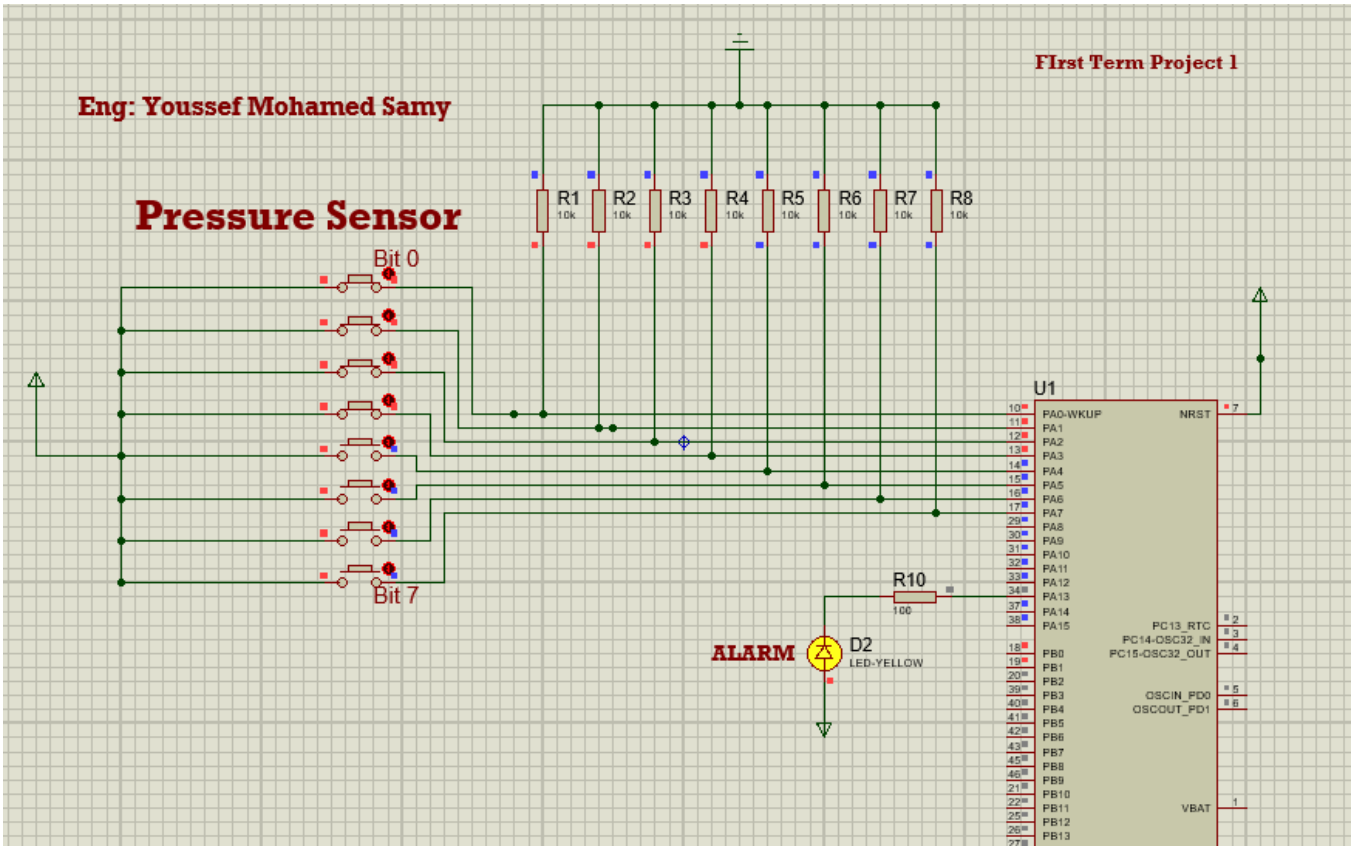
Case2: Pressure > Threshold:

```
PS_BUSY state : Pressure= 23
Pressure Sensor ========> Pressure=23 ========> Main Algorithm


MA_Waiting state : Pressure=23
Main Algorithm ==========> State=1 ==========> Alarm_Actuator


Alarm Actuator BUSY State : State=1
```

In this case, the Pressure = 23 which exceeds the predefined threshold (20Bar).
So The Pressure Sensor sent this value to the Main Algorithm
and the Main Algorithm sent to Alarm Actuator “State=1” which states for **Alarm ON** (Alarm Start).

Simulation:




FIrst Term Project 1
Eng: Youssef Mohamed Samy
Pressure Sensor
Bit 0
Bit 7
R1 10k
R2 10k
R3 10k
R4 10k
R5 10k
R6 10k
R7 10k
R8 10k
R10 100
ALARM
D2 LED-YELLOW
U1
PA0-WKUP
PA1
PA2
PA3
PA4
PA5
PA6
PA7
PA8
PA9
PA10
PA11
PA12
PA13
PA14
PA15
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
PB8
PB9
PB10
PB11
PB12
PB13
NRST
PC13_RTC
PC14-OSC32_IN
PC15-OSC32_OUT
OSCIN_PD0
OSCOUT_PD1
VBAT