

### Contents

Case Study	2
Assumptions	2
Requirements	3
System Analysis : Use Case Diagram	3
System Analysis : Activity Diagram	4
System Analysis : Sequence Diagram	4
System Design	5
Pressure Sensor	5
Main Algorithm	6
Alarm Actuator	6
Monitor Alarm	7
Simulation trace from ttool	8
Implementation	9
Memory Sections	9
Symbol Table	10
PROTEUS	11
The output in case of Low-Pressure value, so the alarm is OFF	11
The output in case of High-Pressure value, so the alarm is ON	11

# Case Study

- A Pressure Controller informs the crew of cabin with an alarm when the pressure exceeds 20 Bars in the cabin.
- The Alarm duration equals 60 Seconds.
- Keep track of the measured values.

## Assumptions

- The System setup and shutdown procedures are not modeled.
- The System maintenance is not modeled.
- The Pressure Sensor never fails.
- The Alarm never fails.
- The System never faces power cut.

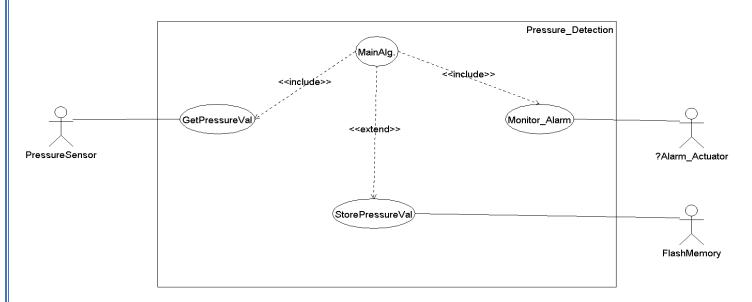
## Requirements <<Requirement>> Pressure\_Detection ID=0 Text="The System shall Protect the Crew against High\_Pressure" Kind="Functional" Risk="Low" Reference elements="" <<Requirement>> Store\_thePressure\_Values <<Requirement>> High\_Pressure\_Detection <<Requirement>> Crew\_Information ID=1 Text="the system shall check the High Pressure in the cabin" Kind="Functional" Risk="Low" Reference elements="" ID=3 Text="the system shall Store the Pressure values in a Flash Memory" Kind="Functional" Risk="Low" Reference elements="" ID=2 Text="the system shall to inform the Crew when the cabin has a too High\_Pressure" Kind="Functional" Risk="Low" Reference elements="" <<ref[ne>> <<ref(ne>> <<refine>> <<Requirement>> Pressure\_Threshold <<Requirement>> Write\_Flash\_Memory ID=4 Text="the system shall check if the pressure is below a predefined threshold = 20 par" Kind="Functional" Risk="Low" Reference elements="" <<Requirement>> Inform\_From\_Led ID=7 Text="the system shall save the recorded values in a fife inside flash memory in a section "Recoreded VAL\_Section"" Kind="Functional" Risk="Low" Reference elements="" ID=6 Text="the system will start alarm by Turn On led with Alarm\_Detection = 60s" Kind="Functional" Risk="Low" Reference elements=""

#### System Analysis: Use Case Diagram

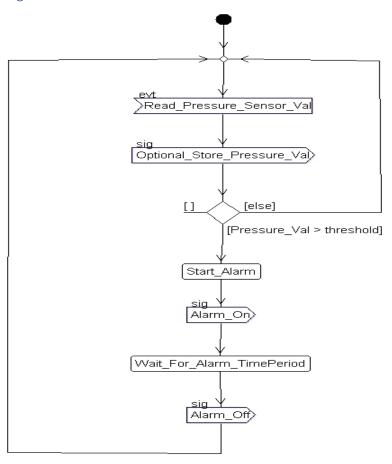
<<ref(ne>>

<<Requirement>>
Read\_from\_PressureSensor

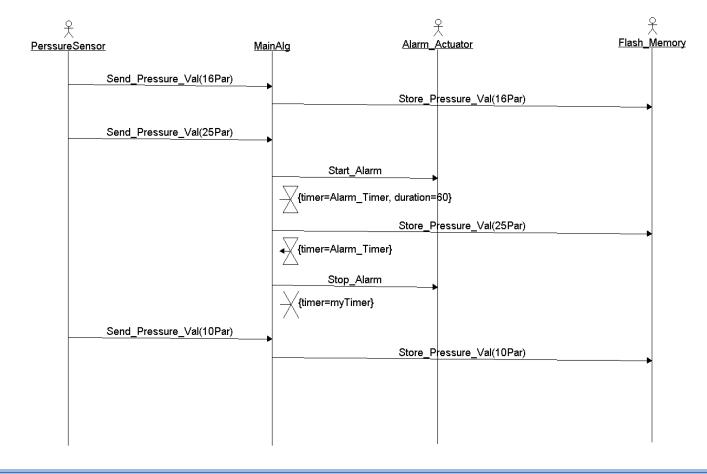
ו⊔=ס Text="the system shall Read from the Pressure Sensor" Kind="Functional" Risk="Low" Reference elements=""



### System Analysis: Activity Diagram

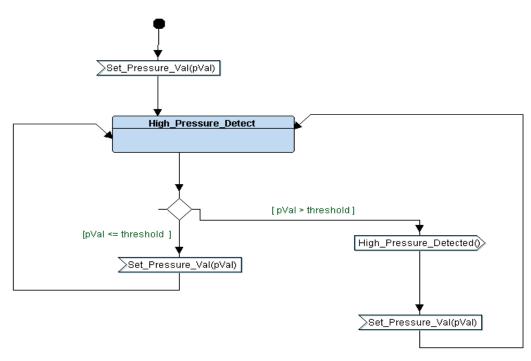


### System Analysis: Sequence Diagram

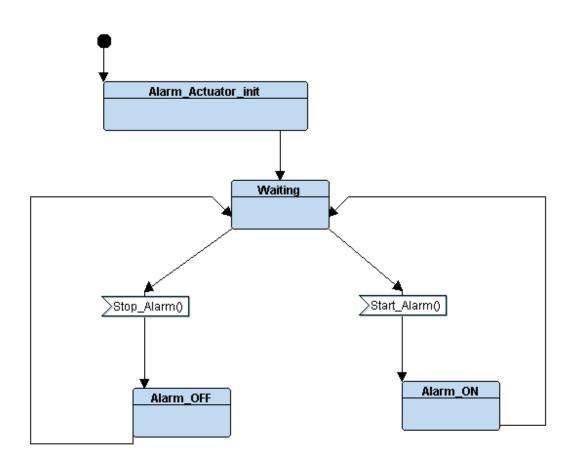


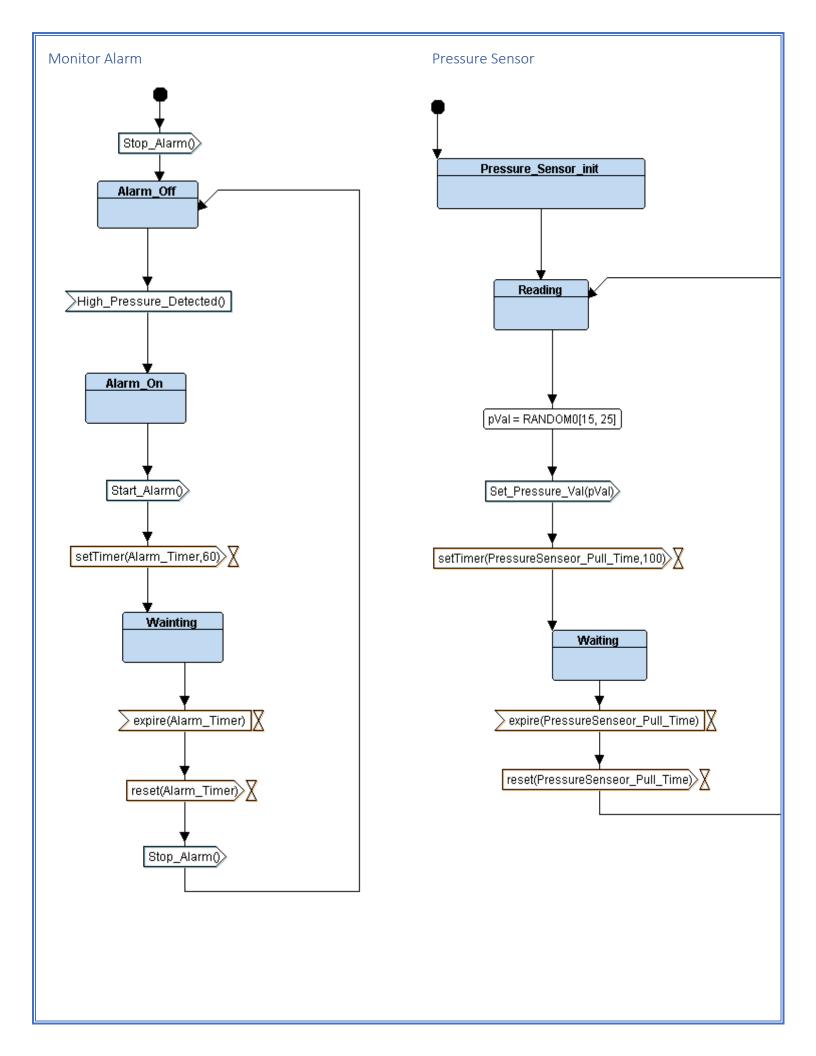
# System Design Pressure\_Detection <<bl>block>><br/>Main\_Alg <<ble><<ble>Monitor\_Alarm - pVal = 0 : int; - threshold = 20 : int; r in Set\_Pressure\_Val(int pVal) r out High\_Pressure\_Detected() <<br/>viock>><br/>Pressure\_Sensor\_Driver <<br/> <br/> Alarm\_Actuator\_Driver 94 - pVal = 0 : int; - PressureSenseor\_Pull\_Time : Timer Pressure\_Sensor\_init() Out Set\_Pressure\_Val(int pVal) Pressure Sensor Pressure\_Sensor\_init Reading pVal = RANDOM0[15, 25] Set\_Pressure\_Val(pVal)> setTimer(PressureSenseor\_Pull\_Time,100)>X Waiting > expire(PressureSenseor\_Pull\_Time) |X reset(PressureSenseor\_Pull\_Time)

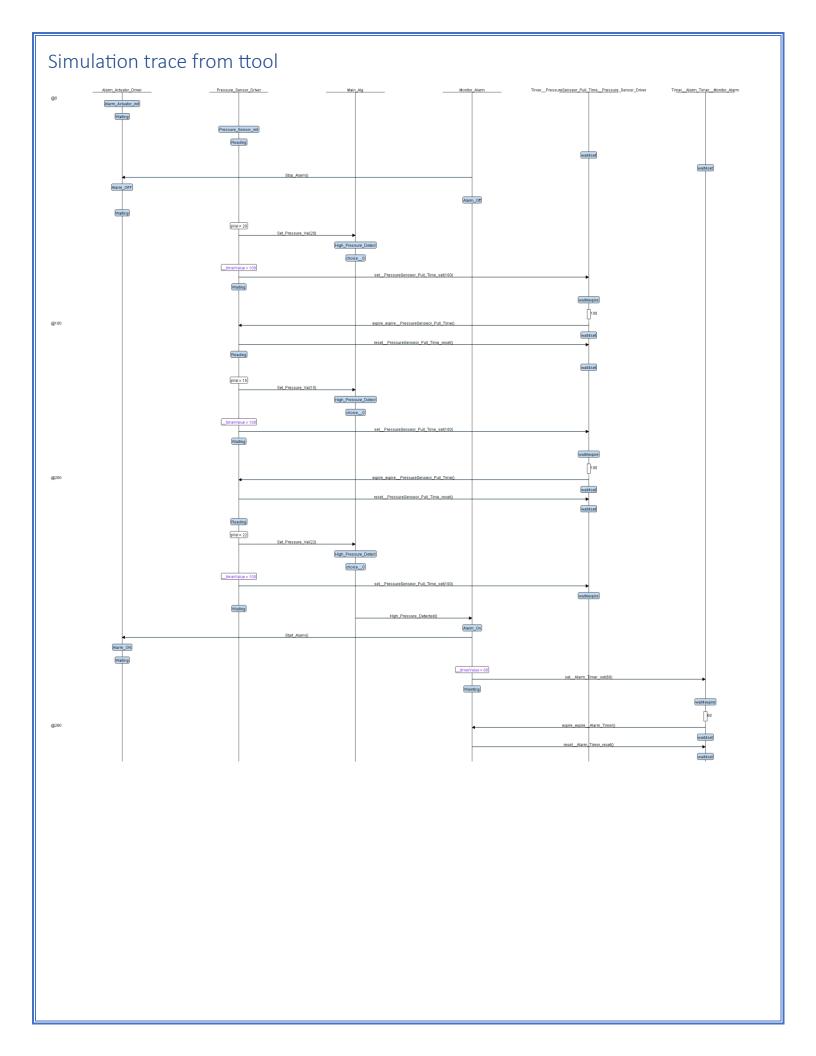
### Main Algorithm



#### Alarm Actuator







# Implementation Press the Link below.

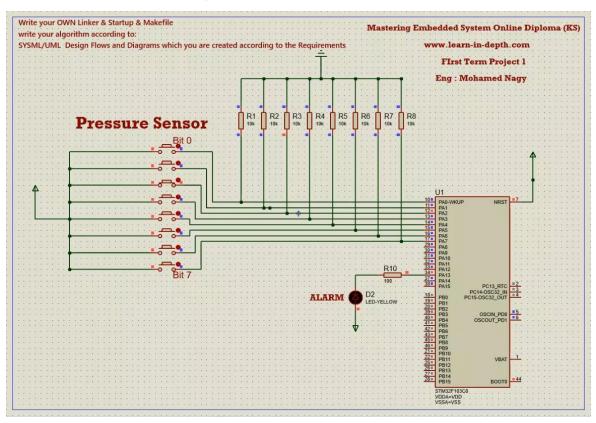
# Memory Sections

•	MINGW64:/e/New	folder/New fo	older/Source(	Code/bin		-		×		
Mohamed Nagy@LAPTOP-R4QD4OLM MINGW64 /e/New folder/New folder/SourceCode/bin \$ arm-none-eabi-objdump -h Pressure_Controller.elf										
Pressure_Controller.elf: file format elf32-littlearm										
	ions:									
	Name	Size	VMA	LMA	File off	Algn				
0	.text	0000041c	08000000	08000000	00010000	2**2				
		CONTENTS,		AD, READONI						
1	.data	0000000c	20000000		00020000	2**2				
		CONTENTS,	ALLOC, LO							
2	.bss	00001020	2000000c	08000428	0002000c	2**2				
		ALLOC								
3	.debug_info	00003f49	00000000	00000000	0002000c	2**0				
		CONTENTS,	READONLY,	DEBUGGING	, OCTETS					
4	.debug_abbrev	00000c1f	00000000	00000000	00023f55	2**0				
			READONLY,	DEBUGGING	, OCTETS					
5	.debug_loc	00000568	00000000	00000000	00024b74	2**0				
	J-			DEBUGGING	. OCTETS					
6	.debug_aranges		00000000		000250dc	2**0				
	· · y y			DEBUGGING						
7	.debug_line	00000ad1	00000000	00000000	000251bc	2**0				
	racoagc			DEBUGGING						
8	.debug_str	00000777	00000000	00000000	00025c8d	2**0				
•	. debug_5 ci			DEBUGGING.		2 0				
9	.comment	0000007e	00000000	00000000	•	2**0				
	. commerre	CONTENTS,		0000000	00020101	2 0				
10	.ARM.attribute			0 00000000	00026482	2 2**0				
10	.AM. acci ibac	CONTENTS,		0 0000000	00020402	2 2 0				
11	.debug_frame	00000340	00000000	00000000	000264b8	2**2				
	. debug_11 dile			DEBUGGING		2 2				
		contients,	REALDONET,	DEDUCATIVA	, ocilio					
Mohamed Nagy@LAPTOP-R4QD4OLM MINGW64 /e/New folder/New folder/SourceCode/bin										
<b>\$</b>	illed Hugyethi II	OI KTQDTOLI	" PILITONOT	, c, New TOTE	aci/New 10	raci / Jour ecco	ac/bill			
Ψ										

```
MINGW64:/e/New folder/New folder/SourceCode/bin
                                                                         X
Mohamed Nagy@LAPTOP-R4QD4OLM MINGW64 /e/New folder/New folder/SourceCode/bin
$ arm-none-eabi-nm Pressure_Controller.elf
20000014 B _E_Bss
2000000c D <u>E</u> Data
0800041c T _E_Text
2000000c B _S_Bss
200000000 D _S_Data
20001014 B _Stack_Top
0800001c T AC_init
20001018 B AC_State
20001014 B AC State id
0800038c W Bus_Fault
0800038c T Default_Handler
080000d4 T Delay
080000f4 T getPressureVal
08000148 T GPIO_INITIALIZATION
0800038c W H_Fault_Handler
08000284 T High_Pressure_Detected
20000000 D MA_Pressure_Threshold
2000000c B MA_Pressure_Value
20001020 B MA_State
2000101c B MA_State_id
080001e0 T main
0800038c W MM_Fault_Handler
20000004 D MoA Periode
20001024 B MoA_State
2000101e B MoA_State_id
0800038c W NMI_Handler
0800031c T PS_init
20000008 D PS_Pull_Time
20001028 B PS_State
2000101d B PS_State_id
20000010 B PS_Value
08000398 T Reset_Handler
0800010c T Set_Alarm_actuator
08000218 T Set Pressure Value
08000198 T Setup
080000ac T ST_AC_Alarm_OFF
08000084 T ST_AC_Alarm_ON
08000060 T ST_AC_Waiting
08000244 T ST_MA_High_Pressure_Detect
080002a0 T ST_MoA_Alarm_OFF
080002c4 T ST_MoA_Alarm_ON
080002e8 T ST_MoA_Waiting
08000328 T ST_PS_Reading
08000364 T ST_PS_Waiting
08000044 T Start_Alarm
08000028 T Stop_Alarm
0800038c W Usage_Fault_Handler
08000000 T vectors
Mohamed Nagy@LAPTOP-R4QD4OLM MINGW64 /e/New folder/New folder/SourceCode/bin
```

### **PROTEUS**

The output in case of Low-Pressure value, so the alarm is OFF



The output in case of High-Pressure value, so the alarm is ON

