Mastering Embedded System Online Diploma

First Term (Final Project 1): High-Pressure-Detector.

Name: Hossam Eid Ibrahim

My Profile:

https://www.learn-in-depth.com/online-diploma/eidhossam7@gmail.com

1- Case Study:

- After meeting with the customer, we figured out the main idea of the project which will be:
    - The system shall measure the pressure in a crew cabin.
    - An alarm should be turned on when the pressure exceeds 20 bar.
    - The alarm should stay on for 60 seconds.
    - The measured values of the pressure should be stored in a flash memory each with its own timestamp (optional).

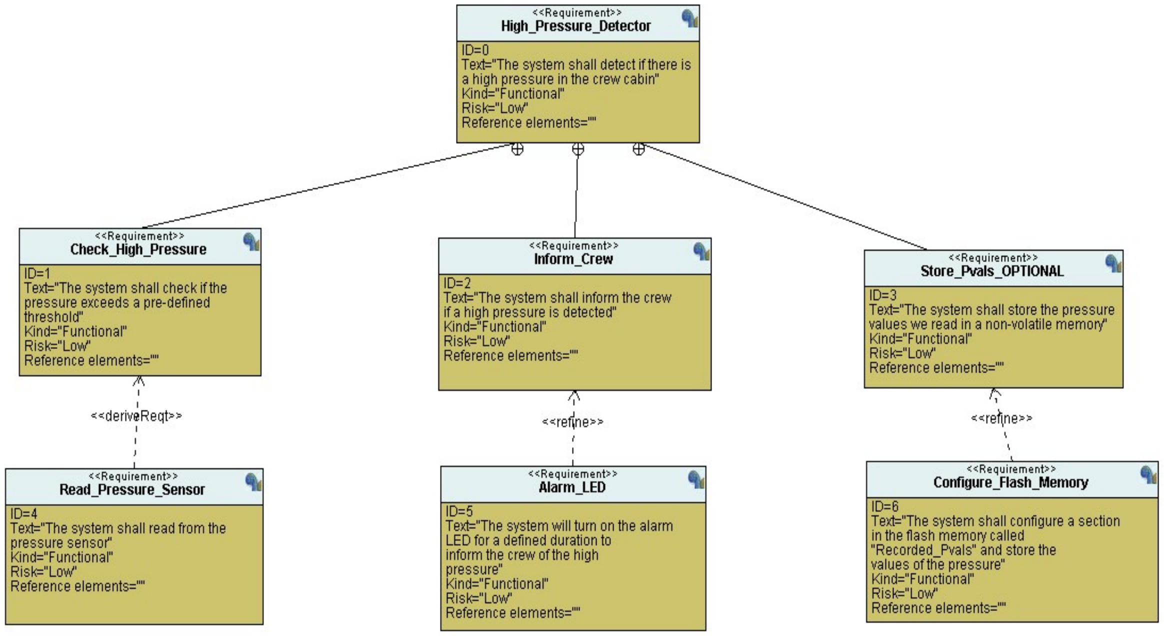
- The following assumptions have been made:
    - The sensor will never fail.
    - The alarm will never fail.
    - The MCU will always have power.
    - Maintaining the MCU isn't modelled.

- Versioning: Saving the captured values feature won't be in the fist version of the SW.

2- Method:

We will work with the V-Cycle method in this project as we saw it as the best option for a project like this.

3- Requirements:



<<Requirement>>
High_Pressure_Detector
ID=0
Text="The system shall detect if there is a high pressure in the crew cabin"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Check_High_Pressure
ID=1
Text="The system shall check if the pressure exceeds a pre-defined threshold"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Inform_Crew
ID=2
Text="The system shall inform the crew if a high pressure is detected"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Store_Pvals_OPTIONAL
ID=3
Text="The system shall store the pressure values we read in a non-volatile memory"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Read_Pressure_Sensor
ID=4
Text="The system shall read from the pressure sensor"
Kind="Functional"
Risk="Low"
Reference elements=""

<<Requirement>>
Alarm_LED
ID=5
Text="The system will turn on the alarm LED for a defined duration to inform the crew of the high pressure"
Kind="Functional"
Risk="Low"
Reference elements=""

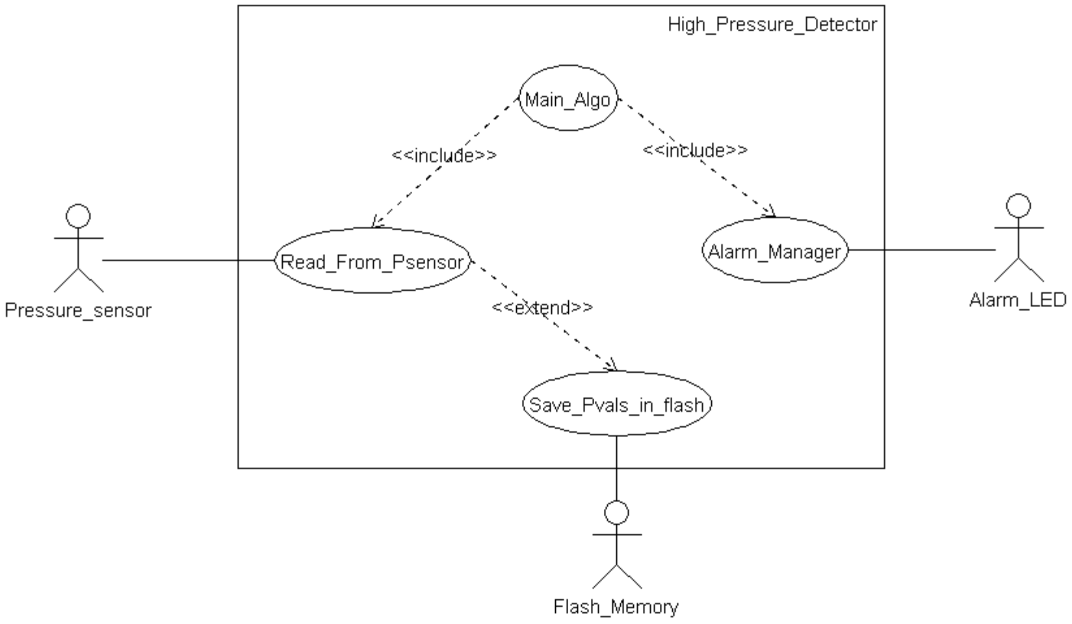
<<Requirement>>
Configure_Flash_Memory
ID=6
Text="The system shall configure a section in the flash memory called "Recorded_Pvals" and store the values of the pressure"
Kind="Functional"
Risk="Low"
Reference elements=""

4- Space exploration:

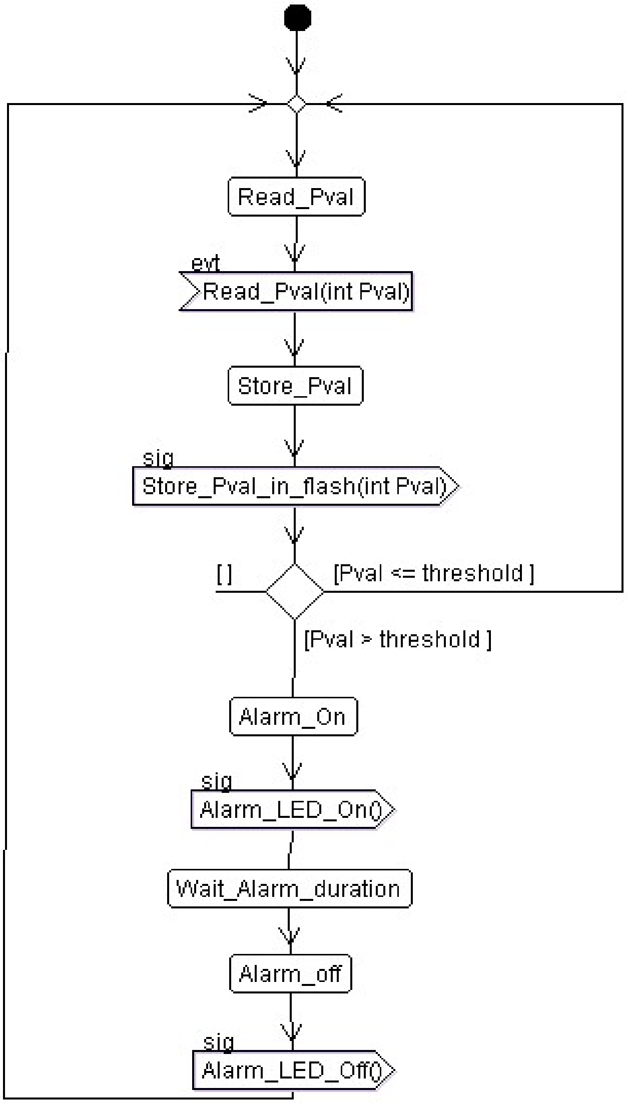
- A single SOC Stm32 microcontroller with a cortex-m3 processor will be used to implement this project.

5- System analysis:

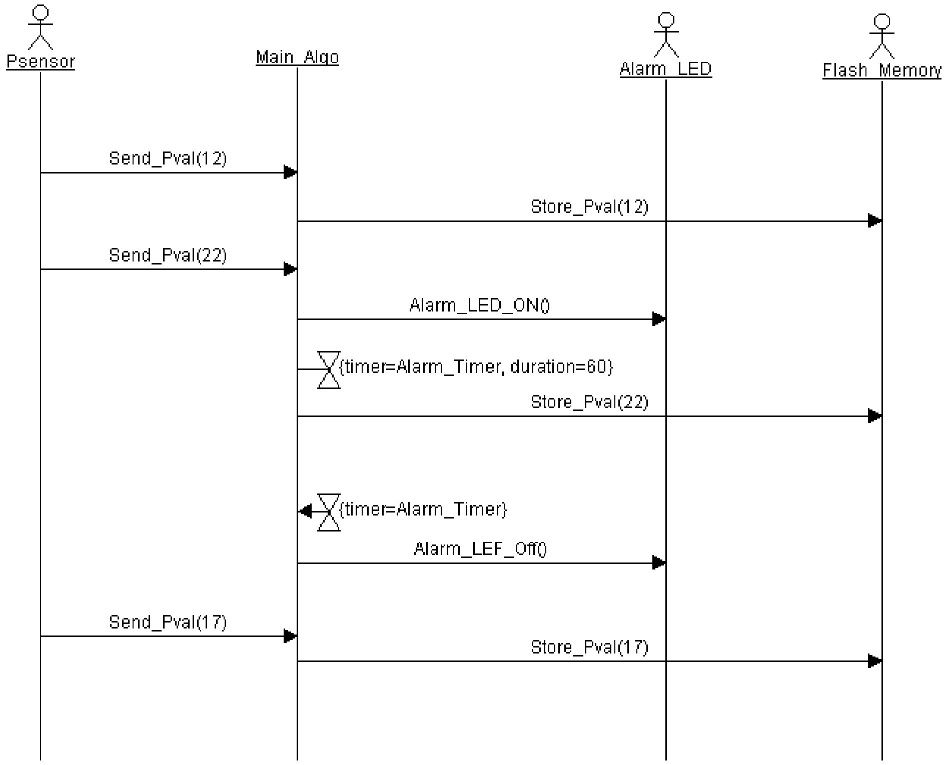
a- Use Case Diagram:



b- Activity Diagram:

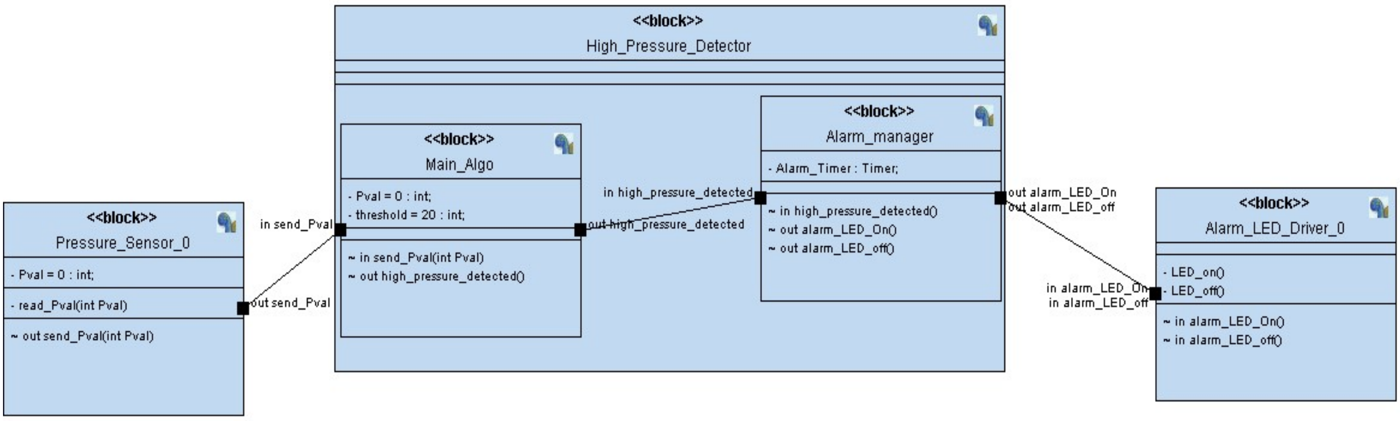


c- Sequence Diagram:

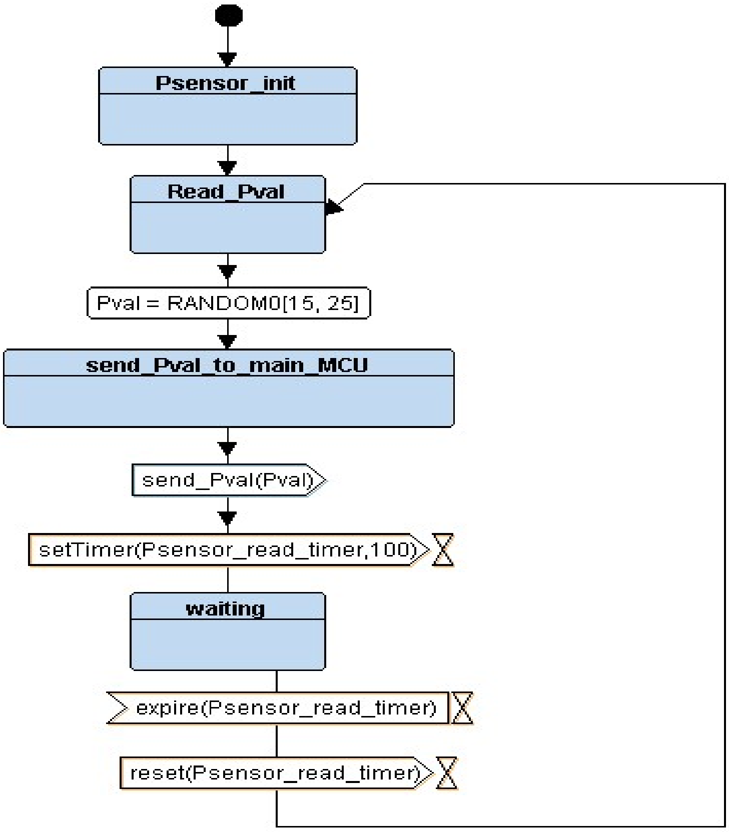


6- System Design:

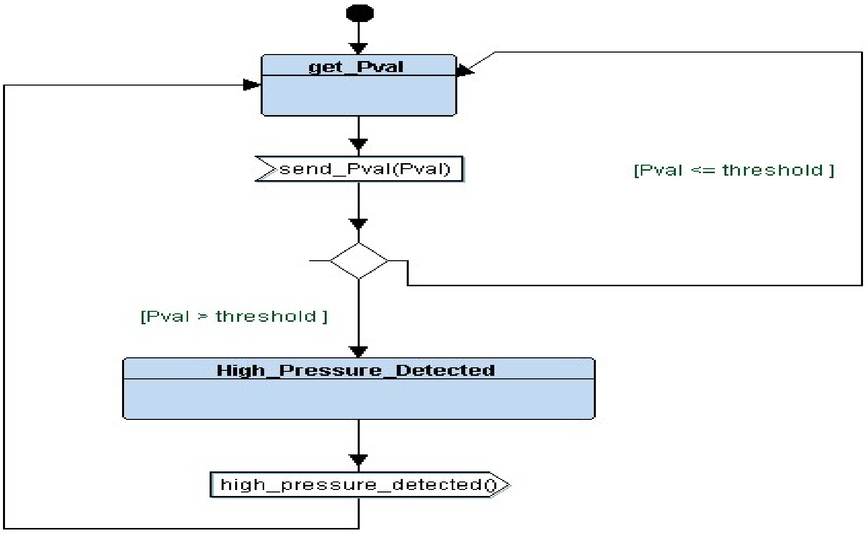
System Block diagram:



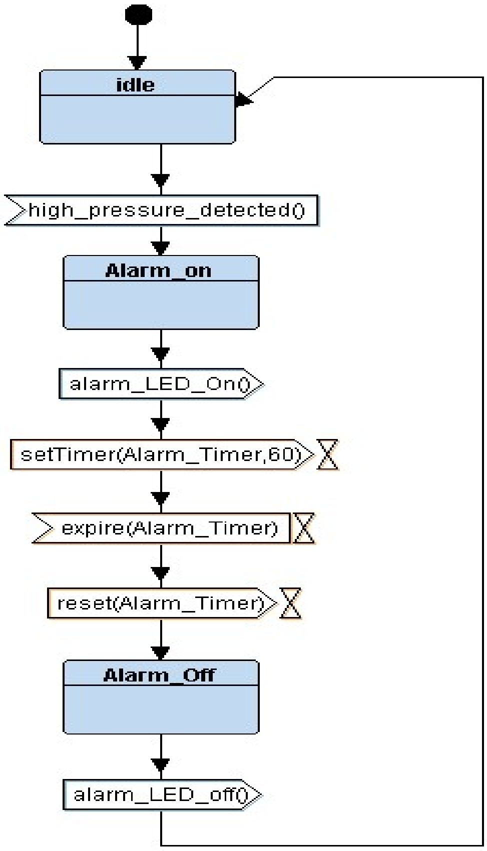
1 – Pressure Sensor:



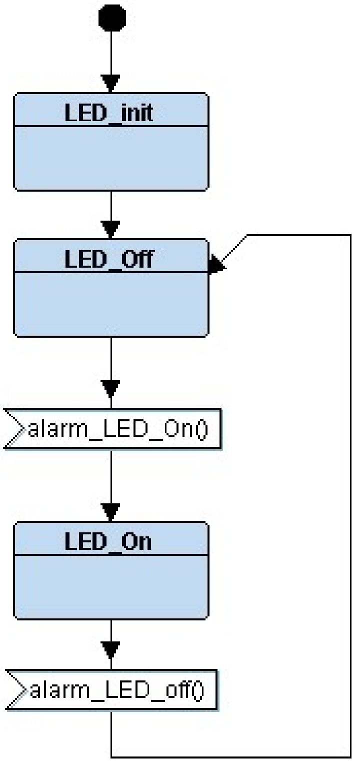
2- Main Algo:



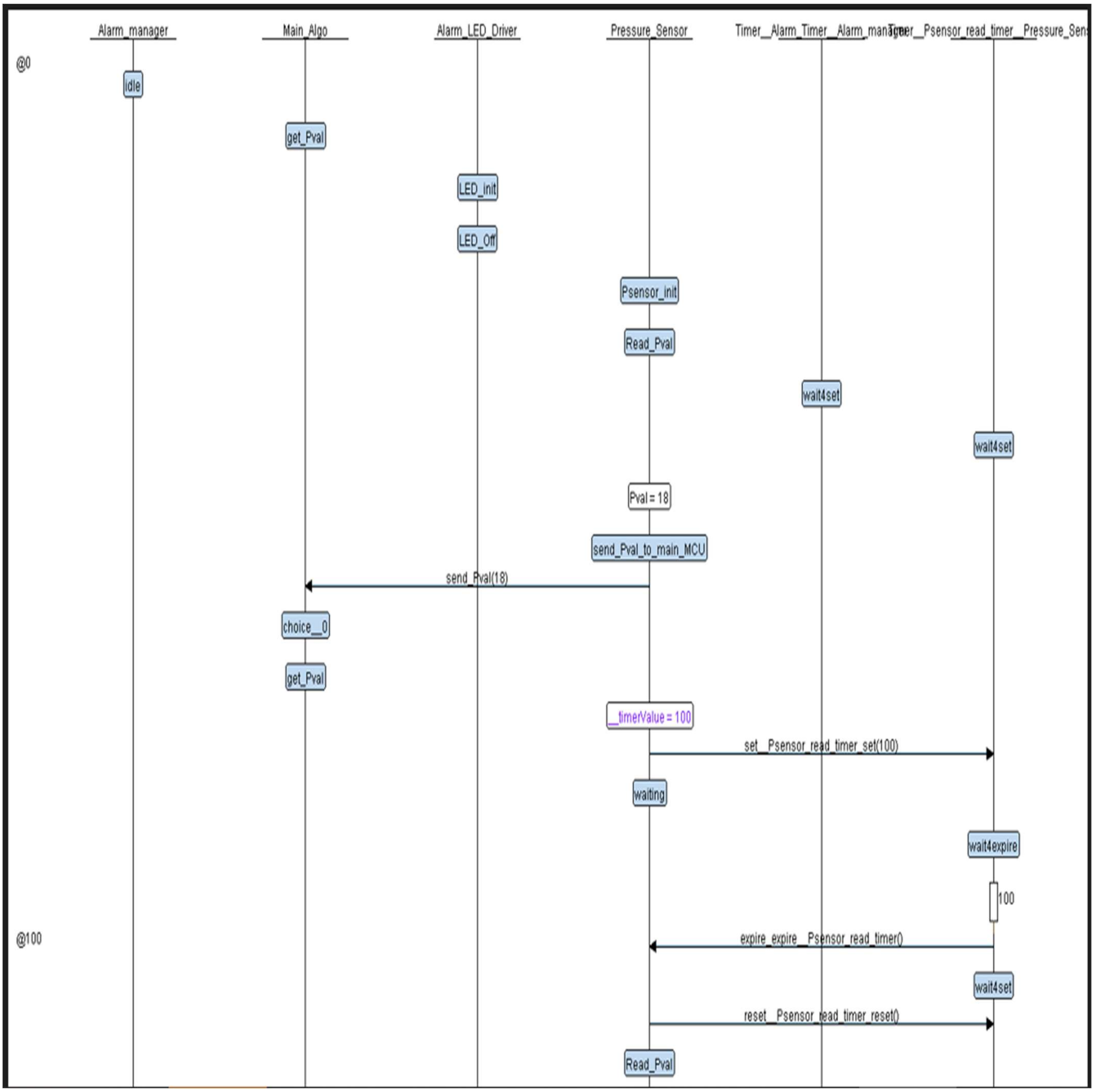
3- Alarm Manager:

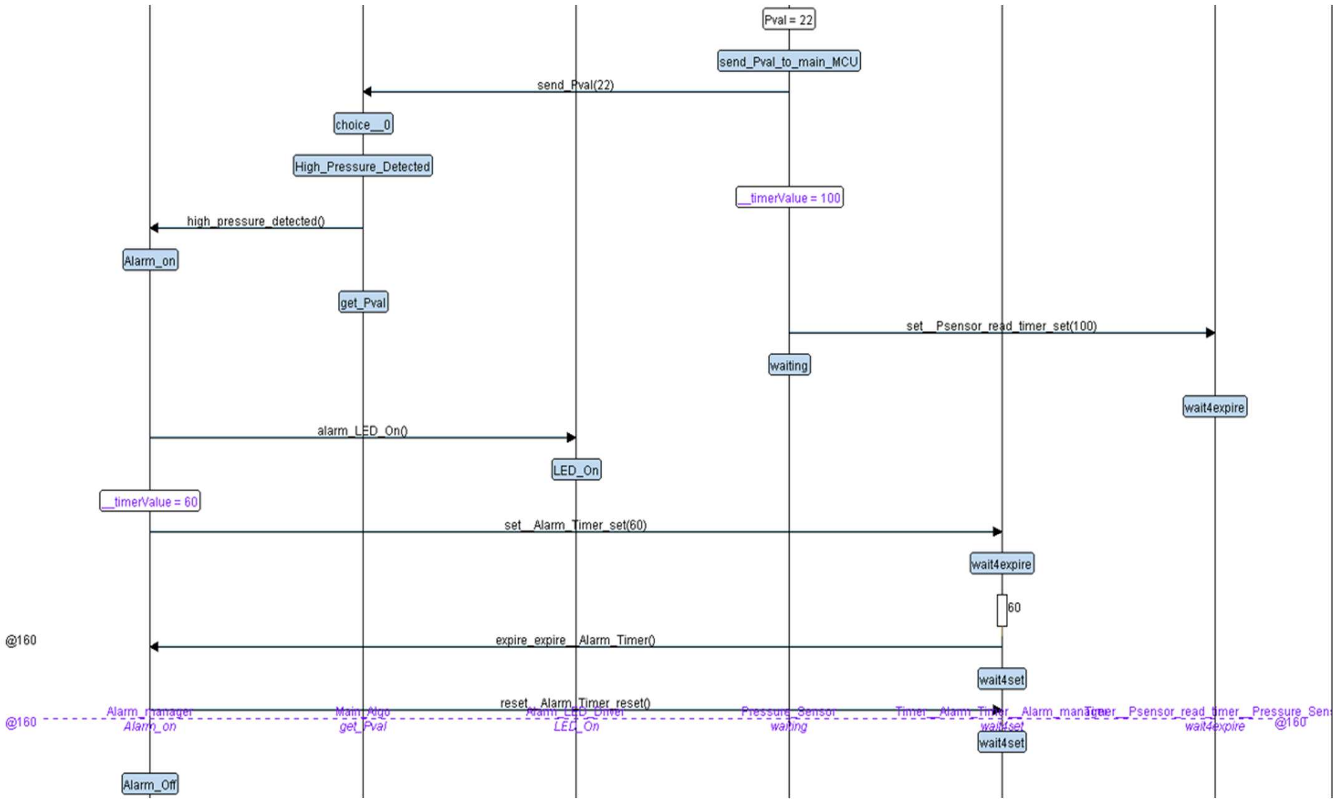


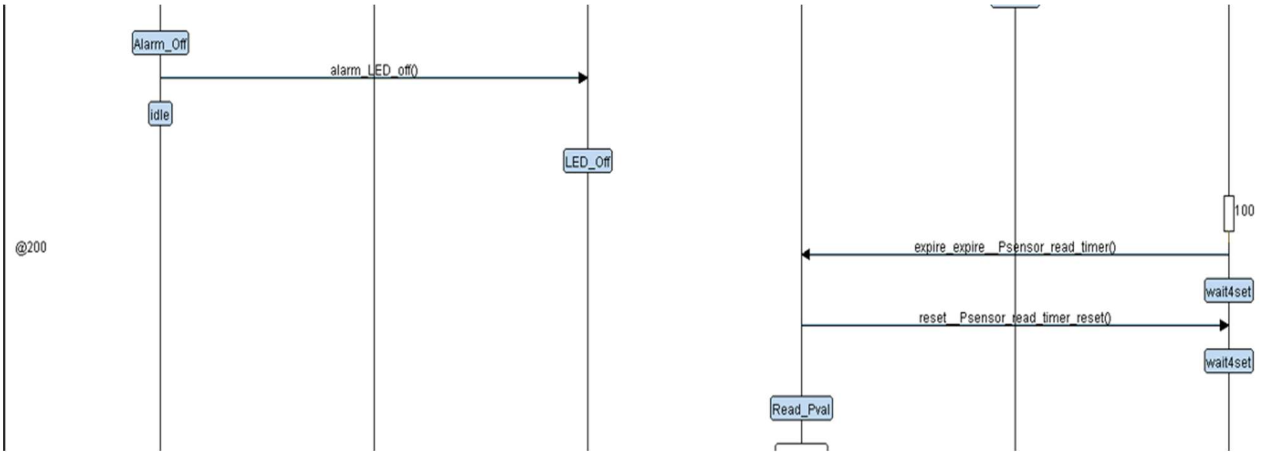
4- Alarm LED Driver:



7- Logic verification:

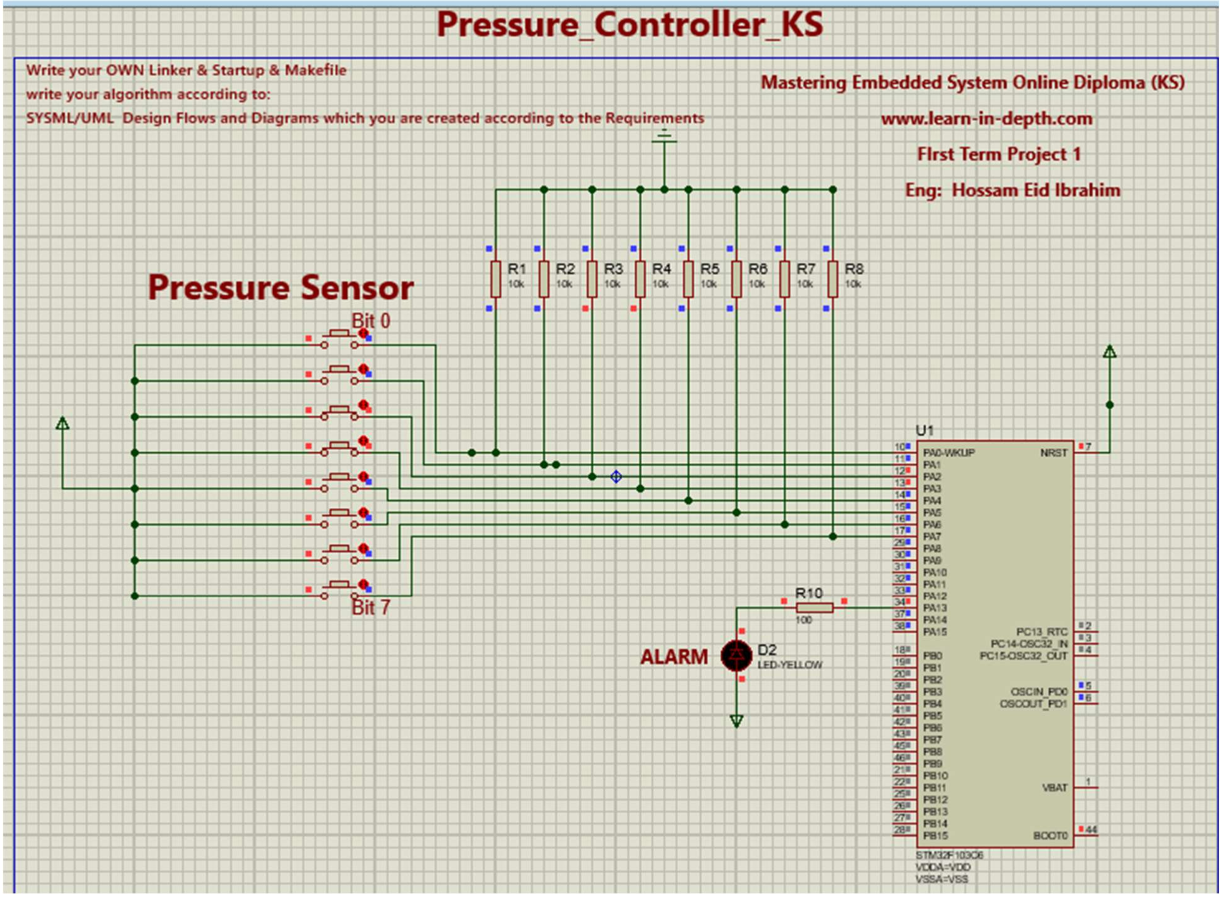



Pval = 22
send_Pval_to_main_MCU
send_Pval(22)
choice__0
High_Pressure_Detected
__timerValue = 100
high_pressure_detected()
Alarm_on
get_Pval
set__Psensor_read_timer_set(100)
waiting
wait4expire
alarm_LED_On()
LED_On
__timerValue = 60
set__Alarm_Timer_set(60)
wait4expire
60
@160
expire_expire__Alarm_Timer()
wait4set
reset__Alarm_Timer_reset()
@160
Alarm_manager
Alarm_on
Main_Algo
get_Pval
Alarm_LED_Driver
LED_On
Pressure_Sensor
waiting
Timer__Alarm_Timer__Alarm_manager
wait4set
Timer__Psensor_read_timer__Pressure_Sens
wait4expire
@160
wait4set
Alarm_Off


Alarm_Off
alarm_LED_off()
idle
LED_Off
100
@200
expire_expire__Psensor_read_timer()
wait4set
reset__Psensor_read_timer_reset()
wait4set
Read_Pval

8- Proteus simulation:

Case: 1

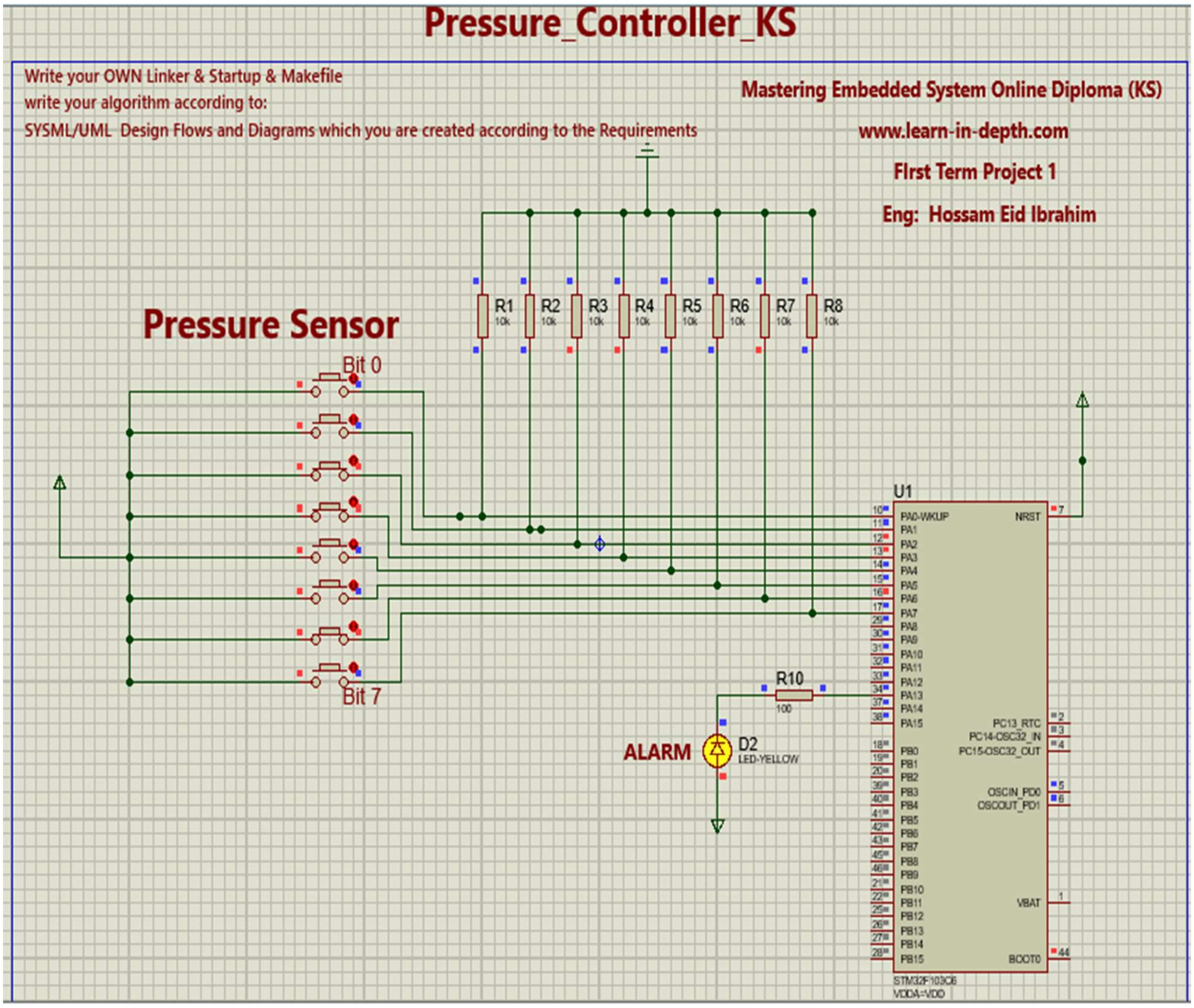


Pressure_Controller_KS

Write your OWN Linker & Startup & Makefile
write your algorithm according to:
SYSML/UML Design Flows and Diagrams which you are created according to the Requirements

Mastering Embedded System Online Diploma (KS)
www.learn-in-depth.com
FIrst Term Project 1
Eng: Hossam Eid Ibrahim

Case: 2



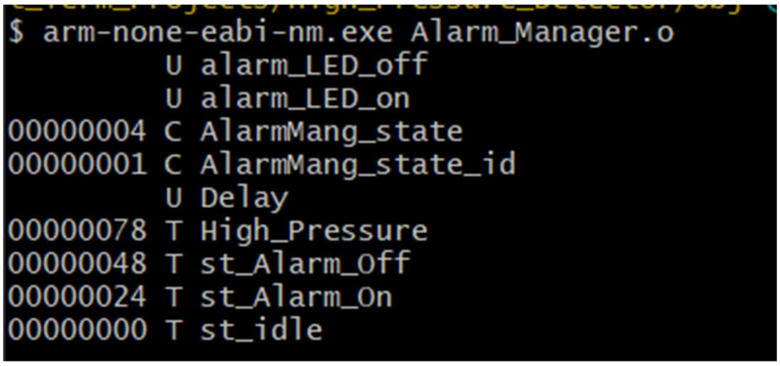
Pressure_Controller_KS

Write your OWN Linker & Startup & Makefile
write your algorithm according to:
SYSML/UML Design Flows and Diagrams which you are created according to the Requirements

Mastering Embedded System Online Diploma (KS)
www.learn-in-depth.com
FIrst Term Project 1
Eng: Hossam Eid Ibrahim

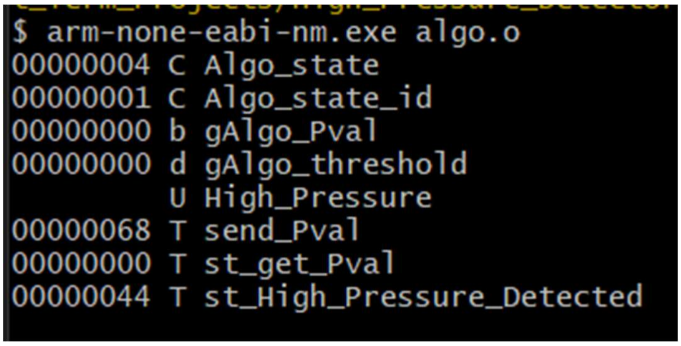
9- Symbols:

1- Alarm_Manager.o:



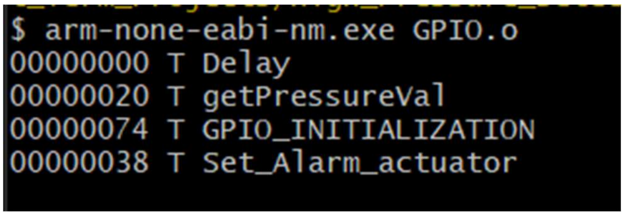
```
$ arm-none-eabi-nm.exe Alarm_Manager.o
         U alarm_LED_off
         U alarm_LED_on
00000004 C AlarmMang_state
00000001 C AlarmMang_state_id
         U Delay
00000078 T High_Pressure
00000048 T st_Alarm_Off
00000024 T st_Alarm_On
00000000 T st_idle
```

2- algo.o:



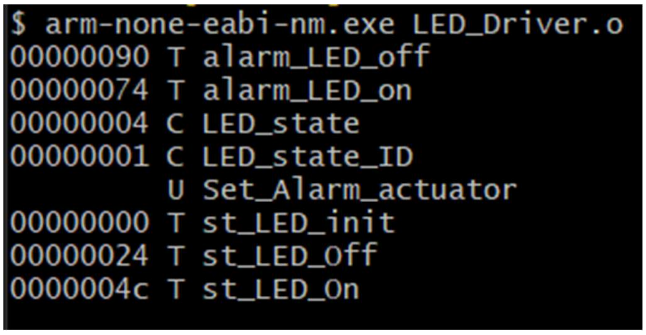
```
$ arm-none-eabi-nm.exe algo.o
00000004 C Algo_state
00000001 C Algo_state_id
00000000 b gAlgo_Pval
00000000 d gAlgo_threshold
         U High_Pressure
00000068 T send_Pval
00000000 T st_get_Pval
00000044 T st_High_Pressure_Detected
```

3- GPIO.o:



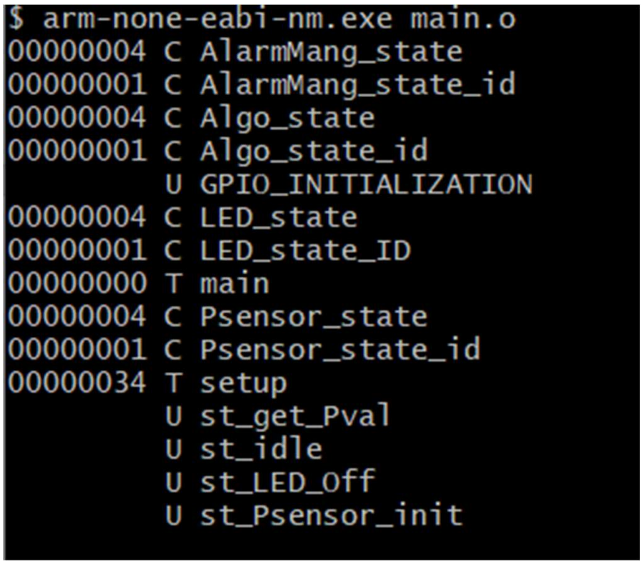
```
$ arm-none-eabi-nm.exe GPIO.o
00000000 T Delay
00000020 T getPressureVal
00000074 T GPIO_INITIALIZATION
00000038 T Set_Alarm_actuator
```

4- LED_Driver.o:



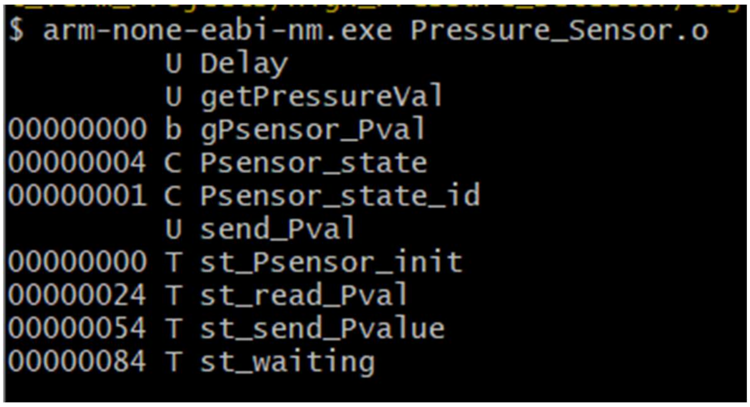
```
$ arm-none-eabi-nm.exe LED_Driver.o
00000090 T alarm_LED_off
00000074 T alarm_LED_on
00000004 C LED_state
00000001 C LED_state_ID
         U Set_Alarm_actuator
00000000 T st_LED_init
00000024 T st_LED_Off
0000004c T st_LED_On
```

5- main.o:



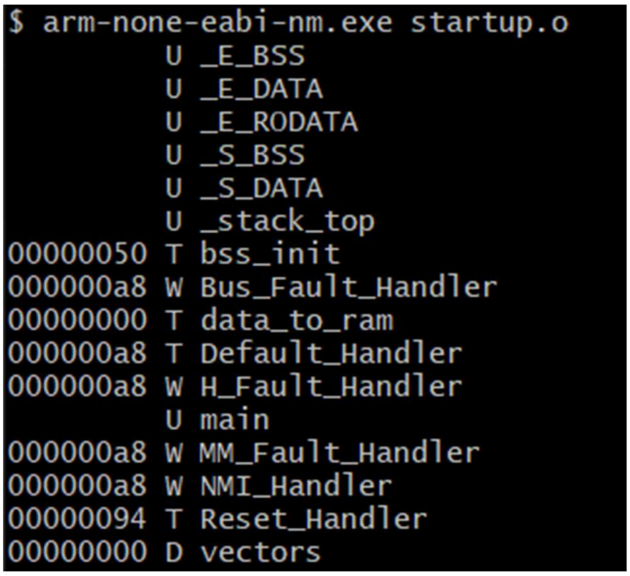
```
$ arm-none-eabi-nm.exe main.o
00000004 C AlarmMang_state
00000001 C AlarmMang_state_id
00000004 C Algo_state
00000001 C Algo_state_id
         U GPIO_INITIALIZATION
00000004 C LED_state
00000001 C LED_state_ID
00000000 T main
00000004 C Psensor_state
00000001 C Psensor_state_id
00000034 T setup
         U st_get_Pval
         U st_idle
         U st_LED_Off
         U st_Psensor_init
```

6- Pressure_Sensor.o:



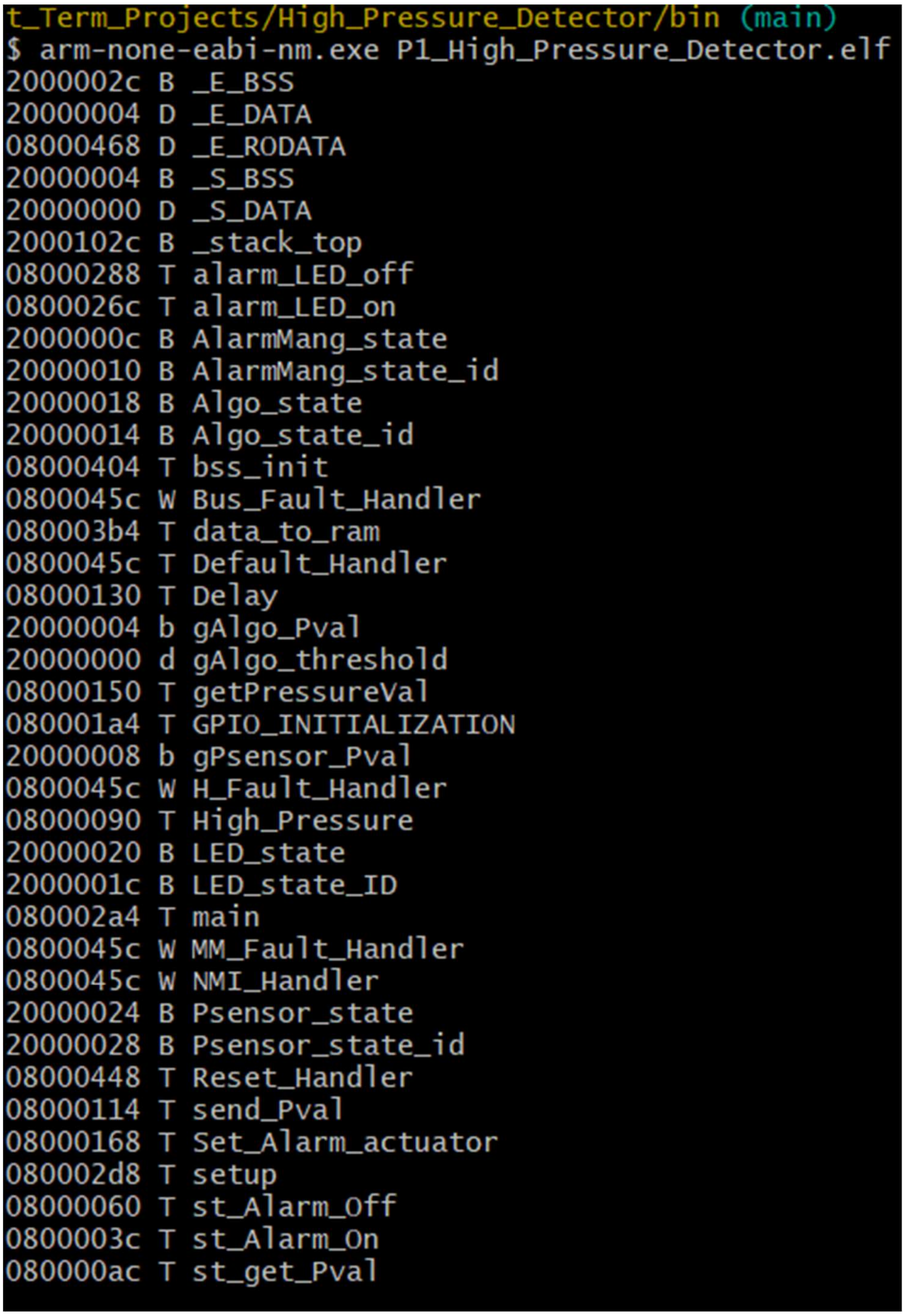
```
$ arm-none-eabi-nm.exe Pressure_Sensor.o
         U Delay
         U getPressureVal
00000000 b gPsensor_Pval
00000004 C Psensor_state
00000001 C Psensor_state_id
         U send_Pval
00000000 T st_Psensor_init
00000024 T st_read_Pval
00000054 T st_send_Pvalue
00000084 T st_waiting
```

7- Startup.o:

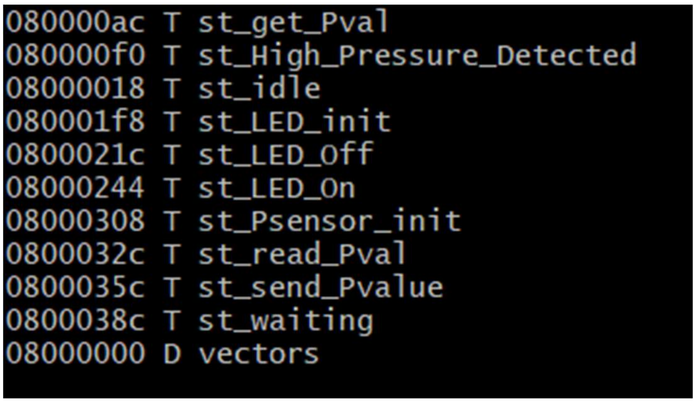


```
$ arm-none-eabi-nm.exe startup.o
         U _E_BSS
         U _E_DATA
         U _E_RODATA
         U _S_BSS
         U _S_DATA
         U _stack_top
00000050 T bss_init
000000a8 W Bus_Fault_Handler
00000000 T data_to_ram
000000a8 T Default_Handler
000000a8 W H_Fault_Handler
         U main
000000a8 W MM_Fault_Handler
000000a8 W NMI_Handler
00000094 T Reset_Handler
00000000 D vectors
```

8- Final elf:



```
t_Term_Projects/High_Pressure_Detector/bin (main)
$ arm-none-eabi-nm.exe P1_High_Pressure_Detector.elf
2000002c B _E_BSS
20000004 D _E_DATA
08000468 D _E_RODATA
20000004 B _S_BSS
20000000 D _S_DATA
2000102c B _stack_top
08000288 T alarm_LED_off
0800026c T alarm_LED_on
2000000c B AlarmMang_state
20000010 B AlarmMang_state_id
20000018 B Algo_state
20000014 B Algo_state_id
08000404 T bss_init
0800045c W Bus_Fault_Handler
080003b4 T data_to_ram
0800045c T Default_Handler
08000130 T Delay
20000004 b gAlgo_Pval
20000000 d gAlgo_threshold
08000150 T getPressureVal
080001a4 T GPIO_INITIALIZATION
20000008 b gPsensor_Pval
0800045c W H_Fault_Handler
08000090 T High_Pressure
20000020 B LED_state
2000001c B LED_state_ID
080002a4 T main
0800045c W MM_Fault_Handler
0800045c W NMI_Handler
20000024 B Psensor_state
20000028 B Psensor_state_id
08000448 T Reset_Handler
08000114 T send_Pval
08000168 T Set_Alarm_actuator
080002d8 T setup
08000060 T st_Alarm_Off
0800003c T st_Alarm_On
080000ac T st_get_Pval
```



```
080000ac T st_get_Pval
080000f0 T st_High_Pressure_Detected
08000018 T st_idle
080001f8 T st_LED_init
0800021c T st_LED_Off
08000244 T st_LED_On
08000308 T st_Psensor_init
0800032c T st_read_Pval
0800035c T st_send_Pvalue
0800038c T st_waiting
08000000 D vectors
```