Mastering Embedded Systems Online Diploma

https://www.learn-in-depth-store.com/

First Term (Final Project 1- Pressure Controller)

Eng. Samar Gamal Mahmoud Eid

My Profile:

https://www.learn-in-depth-store.com/certificate/samargamalmahmoud%40gmail.com

Pressure Control

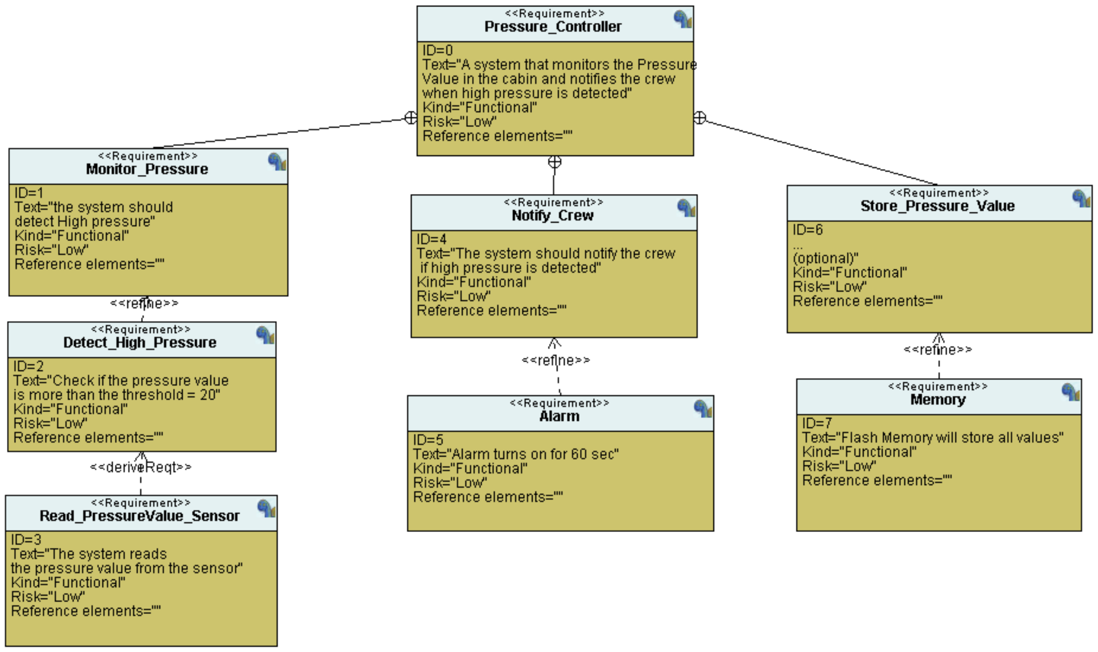
Case Study

A system to notify the crew when high pressure is detected in the cabin.

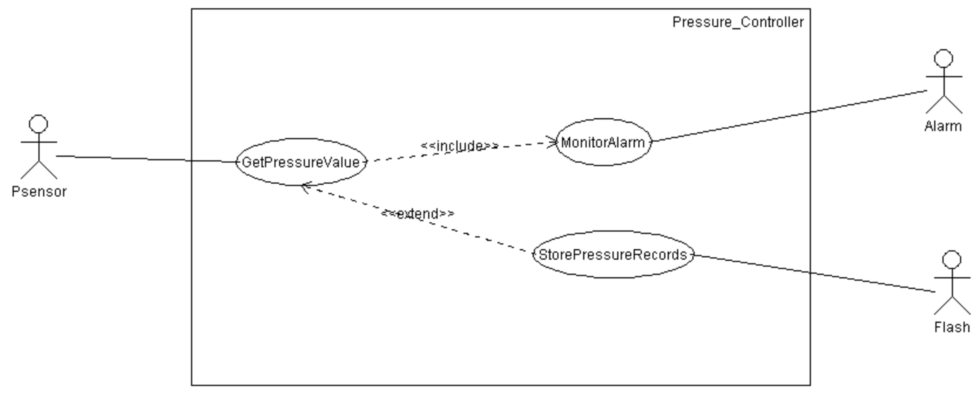
-Pressure threshold → 20 bars

-Alarm duration → 60 seconds

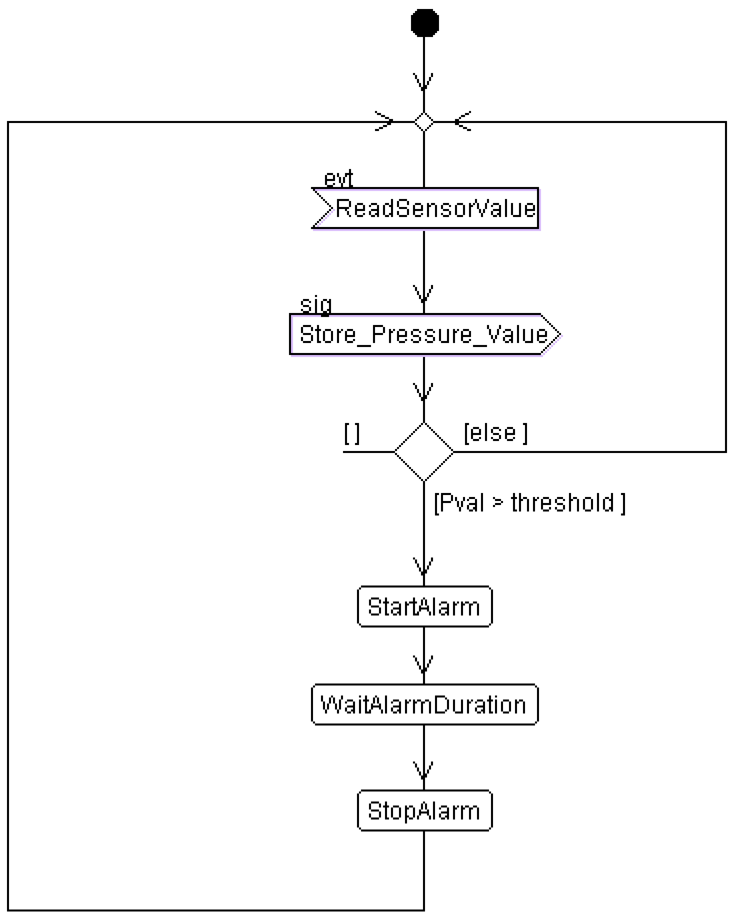
Requirement Diagram



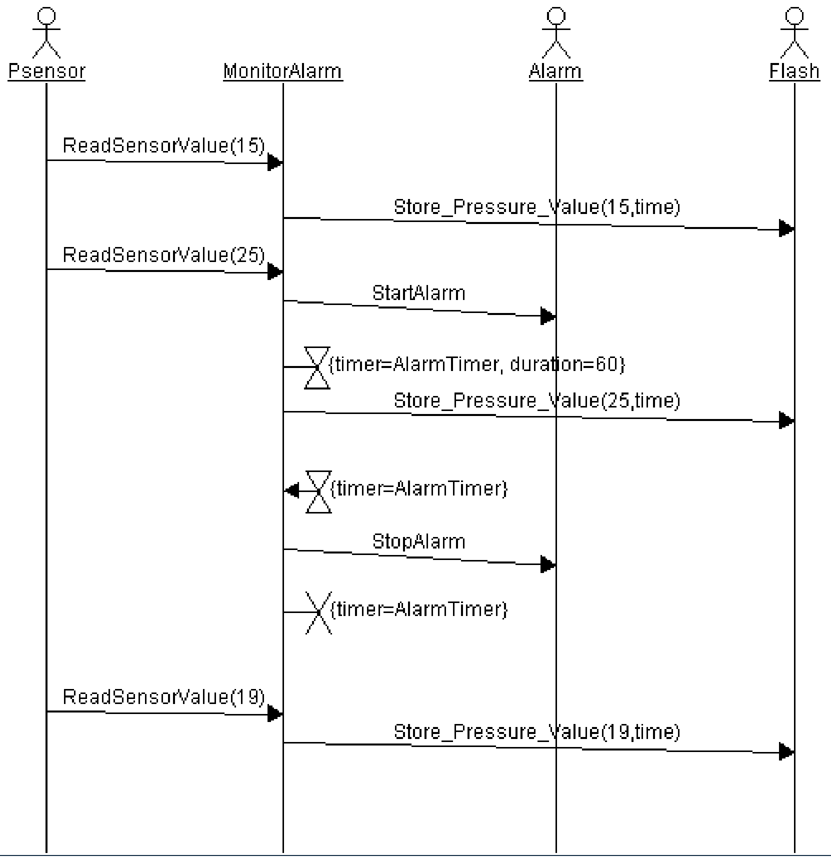
System Analysis : Use Case



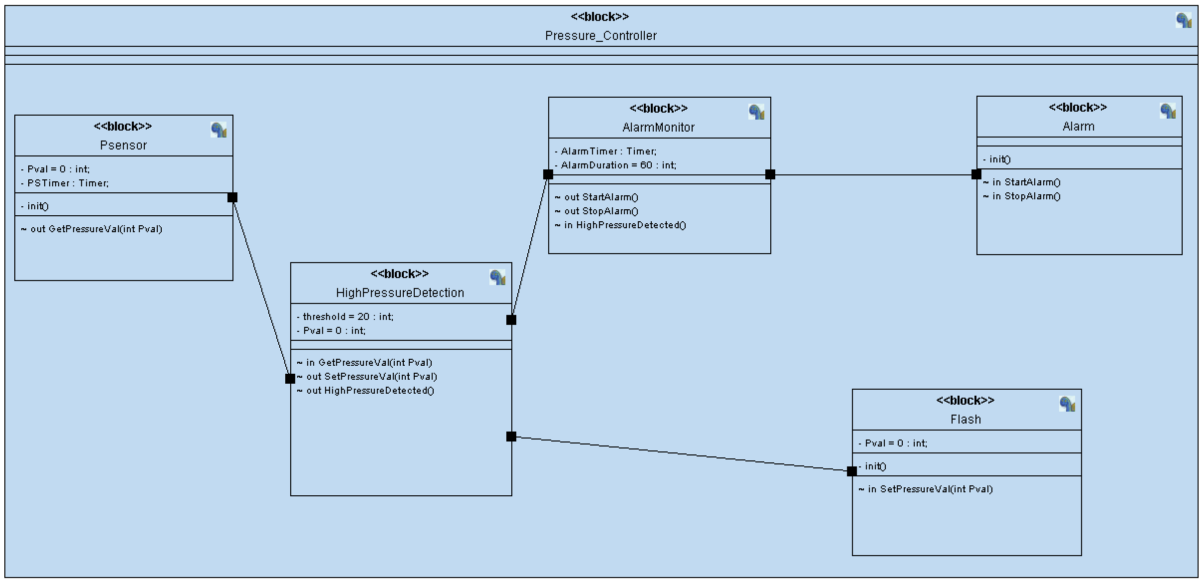
System Analysis: Activity Diagram



System Analysis: Sequence Diagram

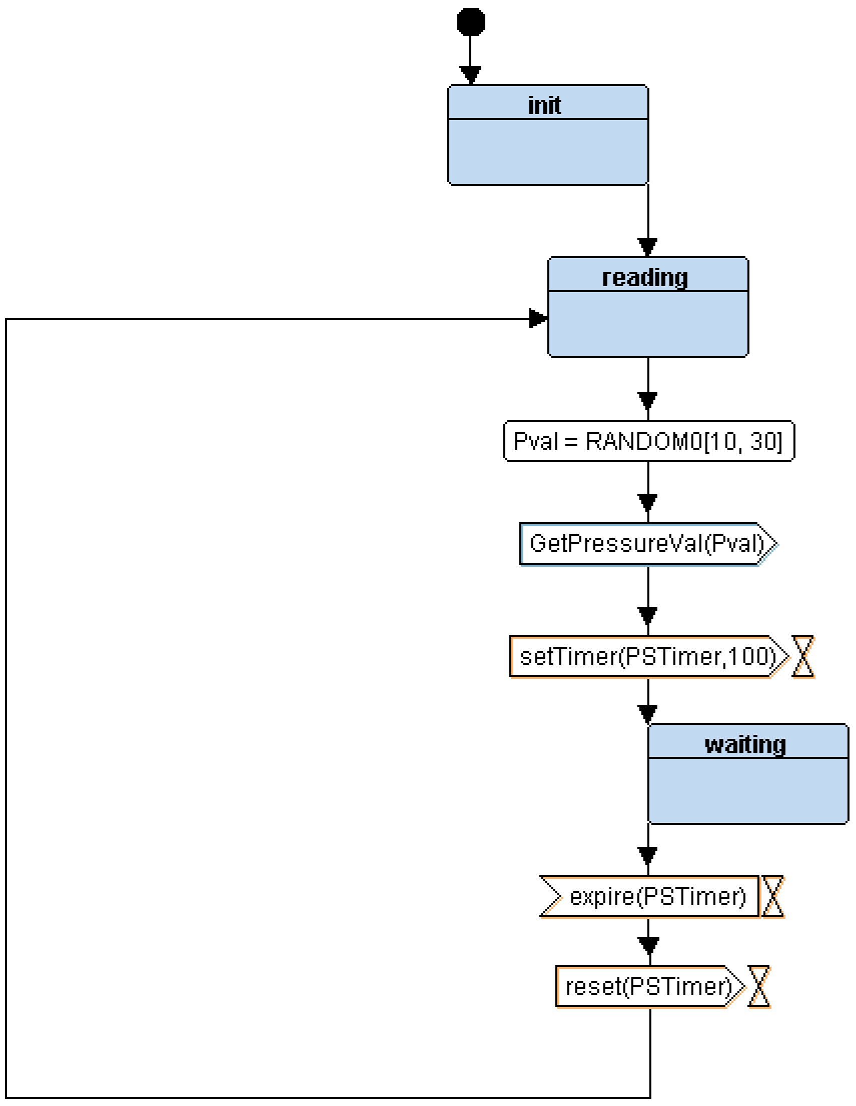


System Design: Block Diagram

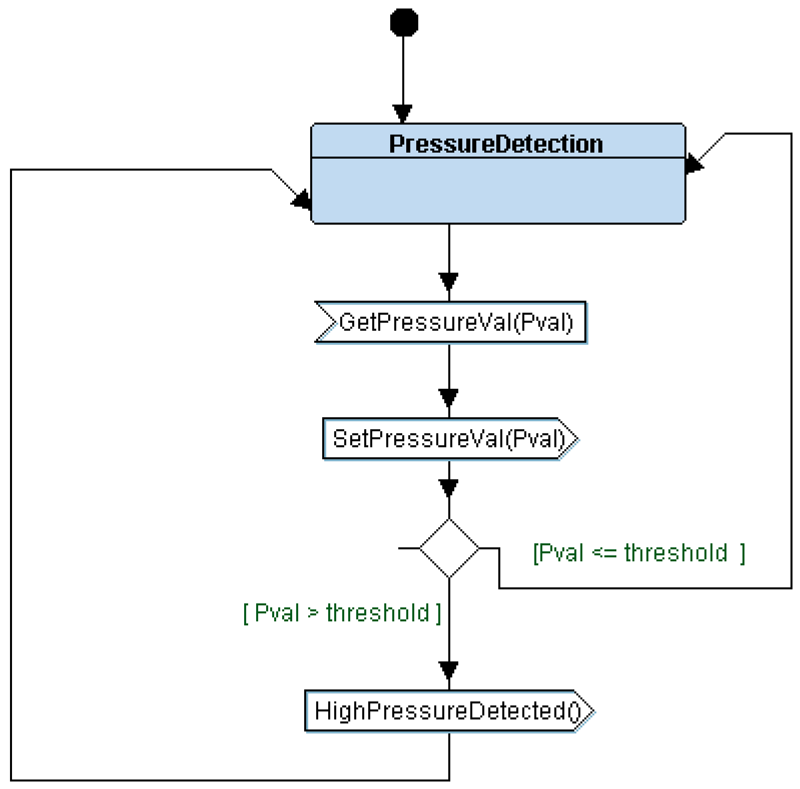


System Design: State Machines

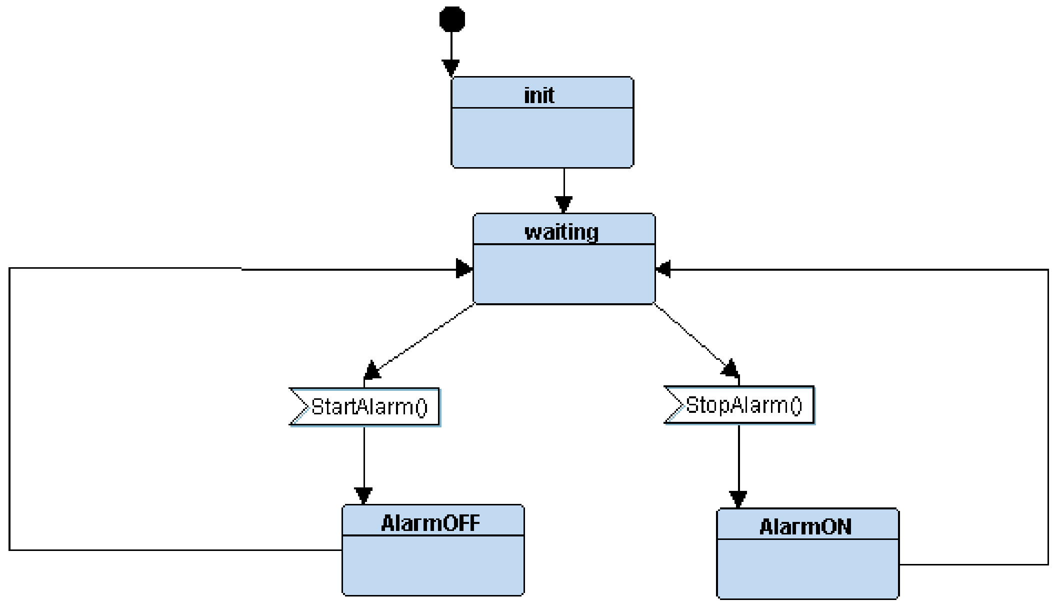
-Sensor



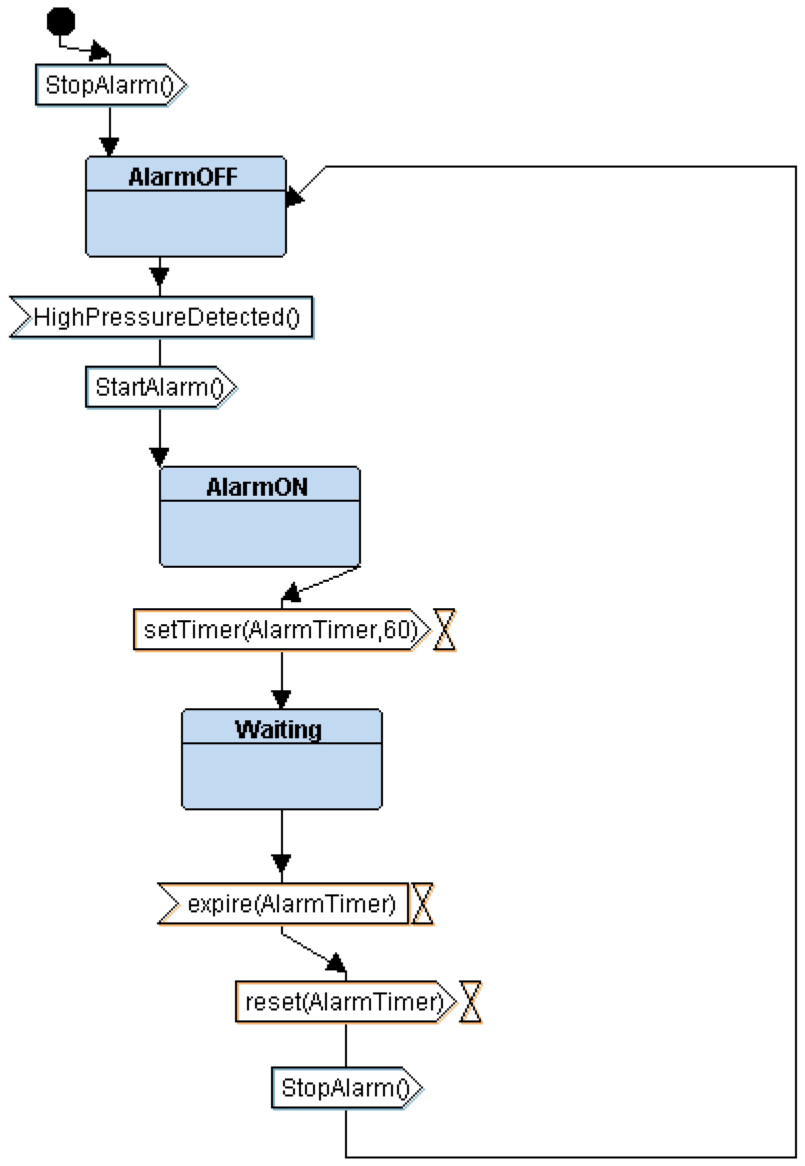
-Pressure Detection



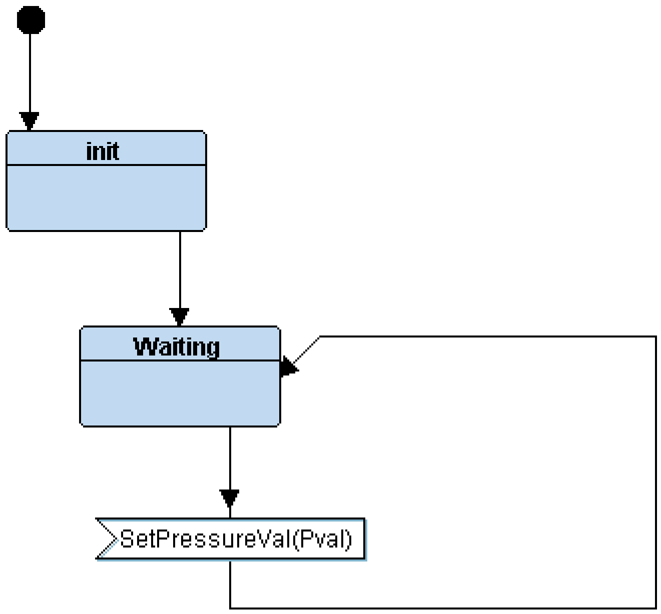
-Alarm



-Alarm Monitor

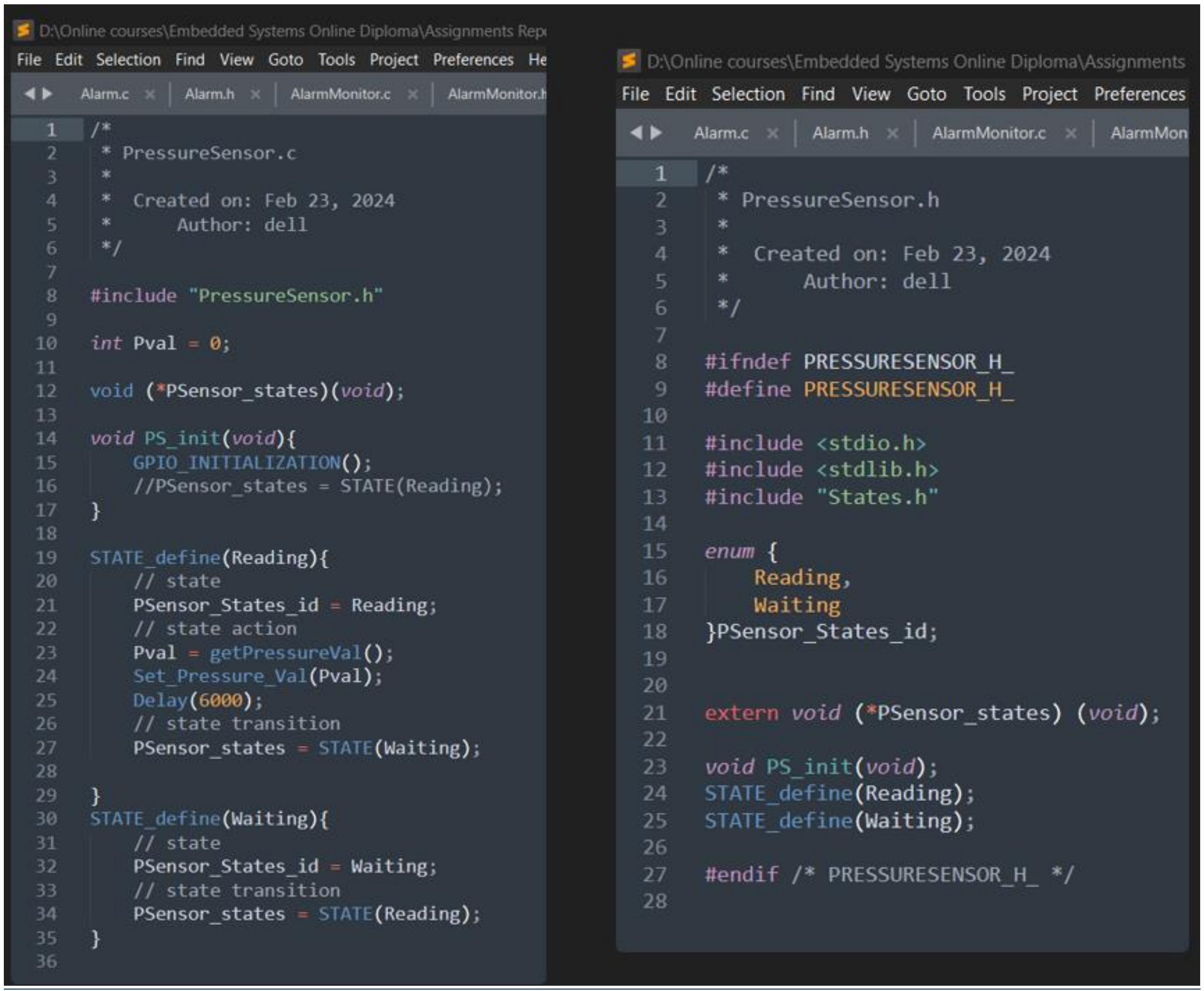


-Flash Memory (Optional – Not implemented)



Implementation

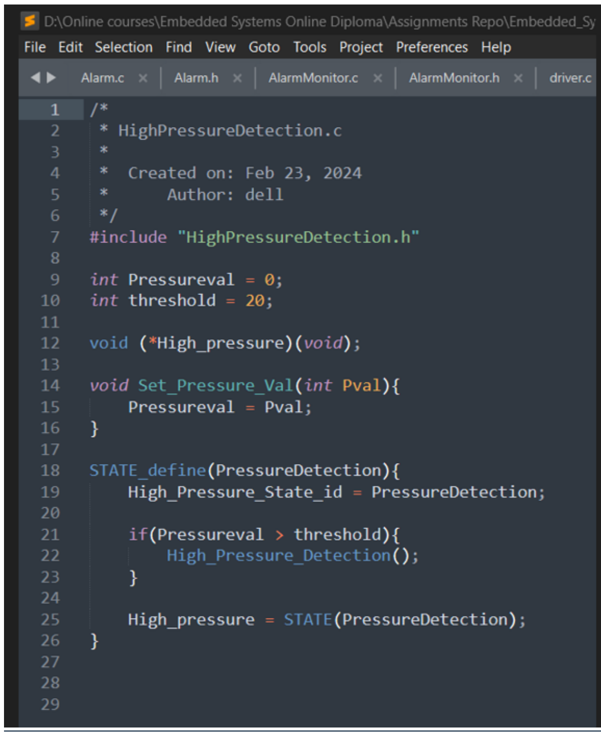
-Sensor


D:\Online courses\Embedded Systems Online Diploma\Assignments Rep
File Edit Selection Find View Goto Tools Project Preferences He
Alarm.c
Alarm.h
AlarmMonitor.c
AlarmMonitor.h
D:\Online courses\Embedded Systems Online Diploma\Assignments
File Edit Selection Find View Goto Tools Project Preferences
Alarm.c
Alarm.h
AlarmMonitor.c
AlarmMon

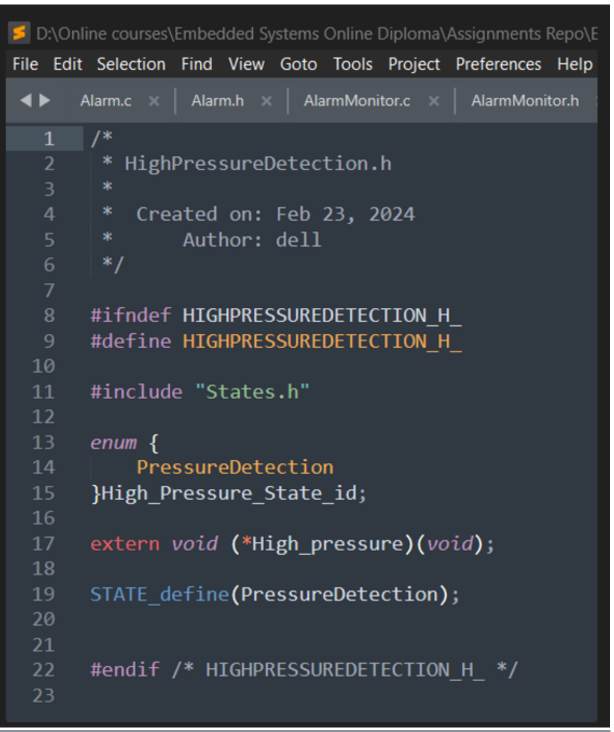
```c
/*
 * PressureSensor.c
 *
 *  Created on: Feb 23, 2024
 *      Author: dell
 */

#include "PressureSensor.h"

int Pval = 0;

void (*PSensor_states)(void);

void PS_init(void){
    GPIO_INITIALIZATION();
    //PSensor_states = STATE(Reading);
}

STATE_define(Reading){
    // state
    PSensor_States_id = Reading;
    // state action
    Pval = getPressureVal();
    Set_Pressure_Val(Pval);
    Delay(6000);
    // state transition
    PSensor_states = STATE(Waiting);

}
STATE_define(Waiting){
    // state
    PSensor_States_id = Waiting;
    // state transition
    PSensor_states = STATE(Reading);
}

```

```c
/*
 * PressureSensor.h
 *
 *  Created on: Feb 23, 2024
 *      Author: dell
 */

#ifndef PRESSURESENSOR_H_
#define PRESSURESENSOR_H_

#include <stdio.h>
#include <stdlib.h>
#include "States.h"

enum {
    Reading,
    Waiting
}PSensor_States_id;


extern void (*PSensor_states) (void);

void PS_init(void);
STATE_define(Reading);
STATE_define(Waiting);

#endif /* PRESSURESENSOR_H_ */

```

-Pressure Detection

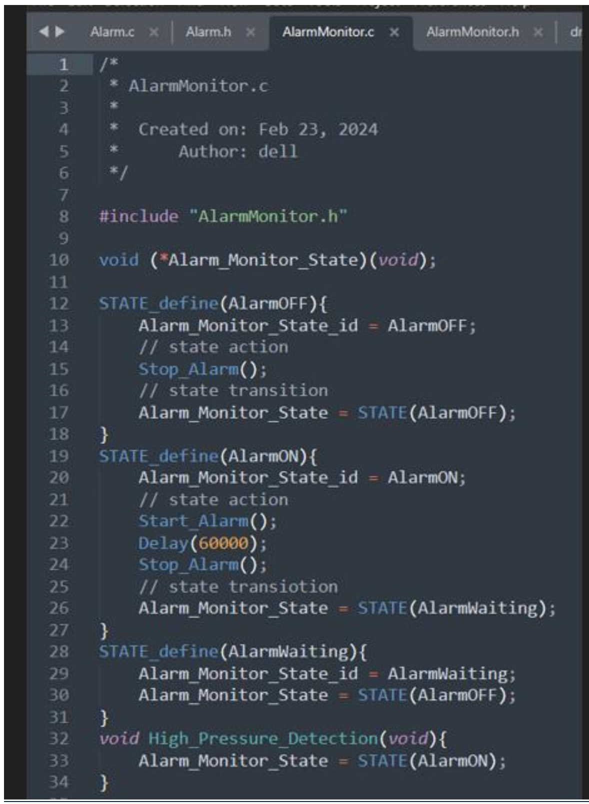

D:\Online courses\Embedded Systems Online Diploma\Assignments Repo\Embedded_Sy
File Edit Selection Find View Goto Tools Project Preferences Help
Alarm.c Alarm.h AlarmMonitor.c AlarmMonitor.h driver.c

```c
/*
 * HighPressureDetection.c
 *
 *  Created on: Feb 23, 2024
 *      Author: dell
 */
#include "HighPressureDetection.h"

int Pressureval = 0;
int threshold = 20;

void (*High_pressure)(void);

void Set_Pressure_Val(int Pval){
    Pressureval = Pval;
}

STATE_define(PressureDetection){
    High_Pressure_State_id = PressureDetection;

    if(Pressureval > threshold){
        High_Pressure_Detection();
    }

    High_pressure = STATE(PressureDetection);
}
```

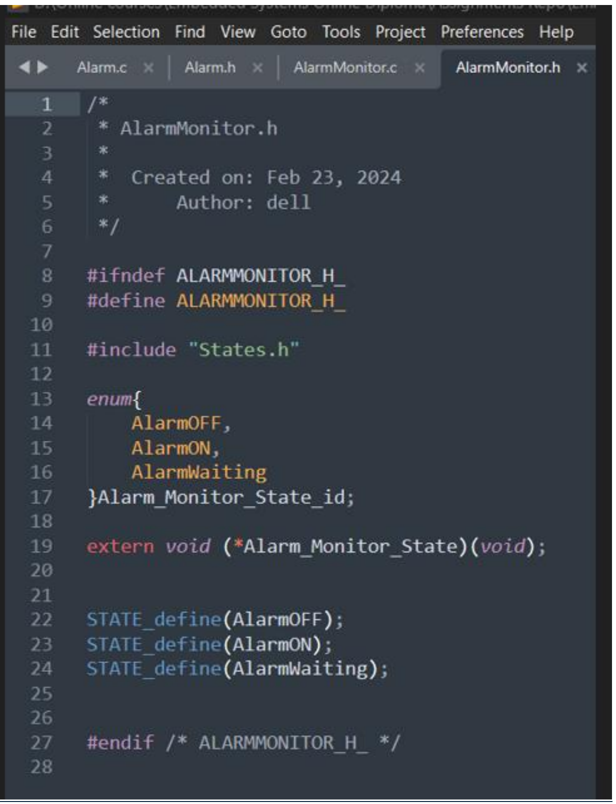

D:\Online courses\Embedded Systems Online Diploma\Assignments Repo\E
File Edit Selection Find View Goto Tools Project Preferences Help
Alarm.c Alarm.h AlarmMonitor.c AlarmMonitor.h

```c
/*
 * HighPressureDetection.h
 *
 *  Created on: Feb 23, 2024
 *      Author: dell
 */

#ifndef HIGHPRESSUREDETECTION_H_
#define HIGHPRESSUREDETECTION_H_

#include "States.h"

enum {
    PressureDetection
}High_Pressure_State_id;

extern void (*High_pressure)(void);

STATE_define(PressureDetection);


#endif /* HIGHPRESSUREDETECTION_H_ */
```

-Alarm Monitor

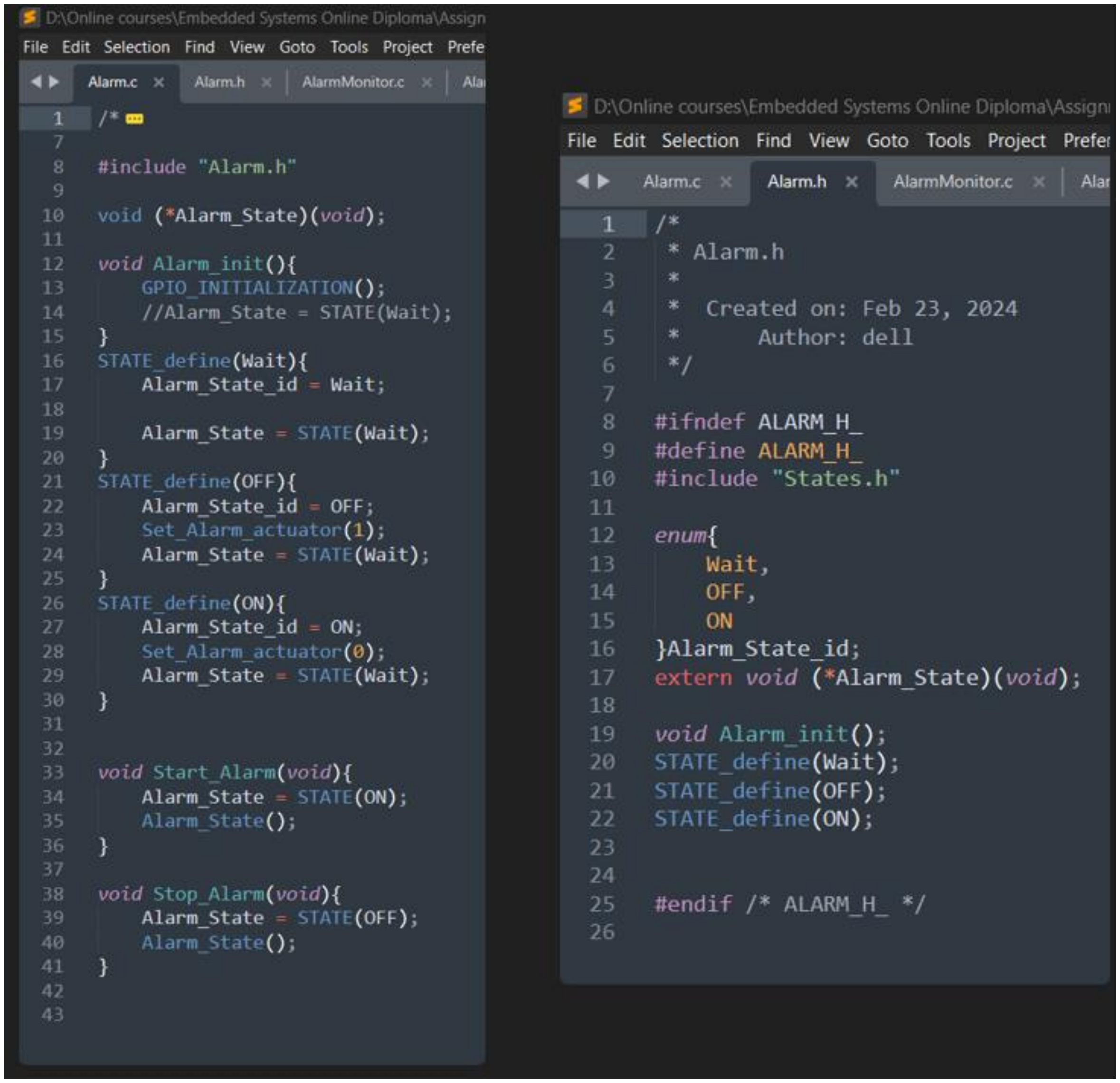


```c
/*
 * AlarmMonitor.c
 *
 *  Created on: Feb 23, 2024
 *      Author: dell
 */

#include "AlarmMonitor.h"

void (*Alarm_Monitor_State)(void);

STATE_define(AlarmOFF){
    Alarm_Monitor_State_id = AlarmOFF;
    // state action
    Stop_Alarm();
    // state transition
    Alarm_Monitor_State = STATE(AlarmOFF);
}
STATE_define(AlarmON){
    Alarm_Monitor_State_id = AlarmON;
    // state action
    Start_Alarm();
    Delay(60000);
    Stop_Alarm();
    // state transiotion
    Alarm_Monitor_State = STATE(AlarmWaiting);
}
STATE_define(AlarmWaiting){
    Alarm_Monitor_State_id = AlarmWaiting;
    Alarm_Monitor_State = STATE(AlarmOFF);
}
void High_Pressure_Detection(void){
    Alarm_Monitor_State = STATE(AlarmON);
}
```


File Edit Selection Find View Goto Tools Project Preferences Help
Alarm.c Alarm.h AlarmMonitor.c AlarmMonitor.h

```c
/*
 * AlarmMonitor.h
 *
 *  Created on: Feb 23, 2024
 *      Author: dell
 */

#ifndef ALARMMONITOR_H_
#define ALARMMONITOR_H_

#include "States.h"

enum{
    AlarmOFF,
    AlarmON,
    AlarmWaiting
}Alarm_Monitor_State_id;

extern void (*Alarm_Monitor_State)(void);


STATE_define(AlarmOFF);
STATE_define(AlarmON);
STATE_define(AlarmWaiting);


#endif /* ALARMMONITOR_H_ */
```

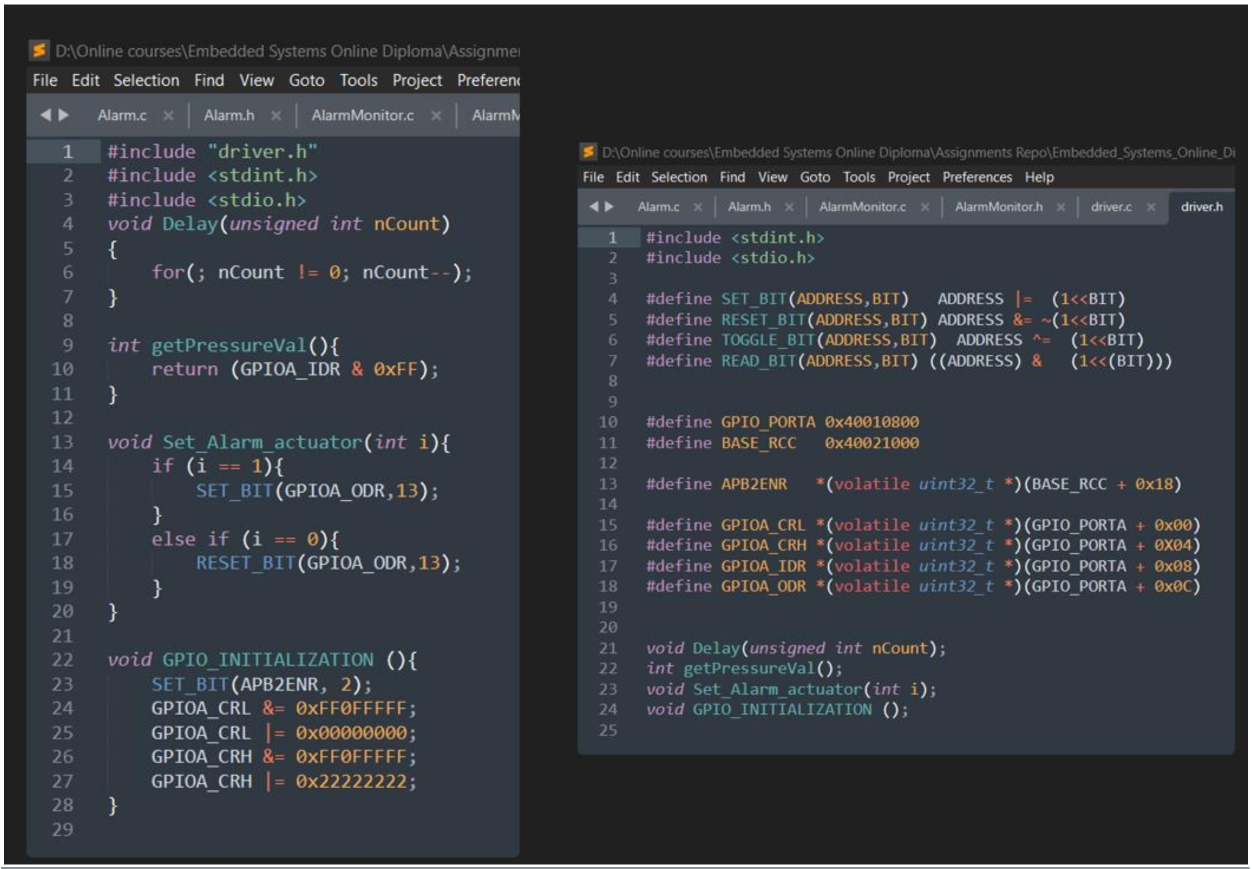
-Alarm


D:\Online courses\Embedded Systems Online Diploma\Assign
File Edit Selection Find View Goto Tools Project Prefe
Alarm.c
Alarm.h
AlarmMonitor.c

```c
/*

#include "Alarm.h"

void (*Alarm_State)(void);

void Alarm_init(){
    GPIO_INITIALIZATION();
    //Alarm_State = STATE(Wait);
}
STATE_define(Wait){
    Alarm_State_id = Wait;

    Alarm_State = STATE(Wait);
}
STATE_define(OFF){
    Alarm_State_id = OFF;
    Set_Alarm_actuator(1);
    Alarm_State = STATE(Wait);
}
STATE_define(ON){
    Alarm_State_id = ON;
    Set_Alarm_actuator(0);
    Alarm_State = STATE(Wait);
}


void Start_Alarm(void){
    Alarm_State = STATE(ON);
    Alarm_State();
}

void Stop_Alarm(void){
    Alarm_State = STATE(OFF);
    Alarm_State();
}
```

```c
/*
 * Alarm.h
 *
 *  Created on: Feb 23, 2024
 *      Author: dell
 */

#ifndef ALARM_H_
#define ALARM_H_
#include "States.h"

enum{
    Wait,
    OFF,
    ON
}Alarm_State_id;
extern void (*Alarm_State)(void);

void Alarm_init();
STATE_define(Wait);
STATE_define(OFF);
STATE_define(ON);


#endif /* ALARM_H_ */
```

-driver

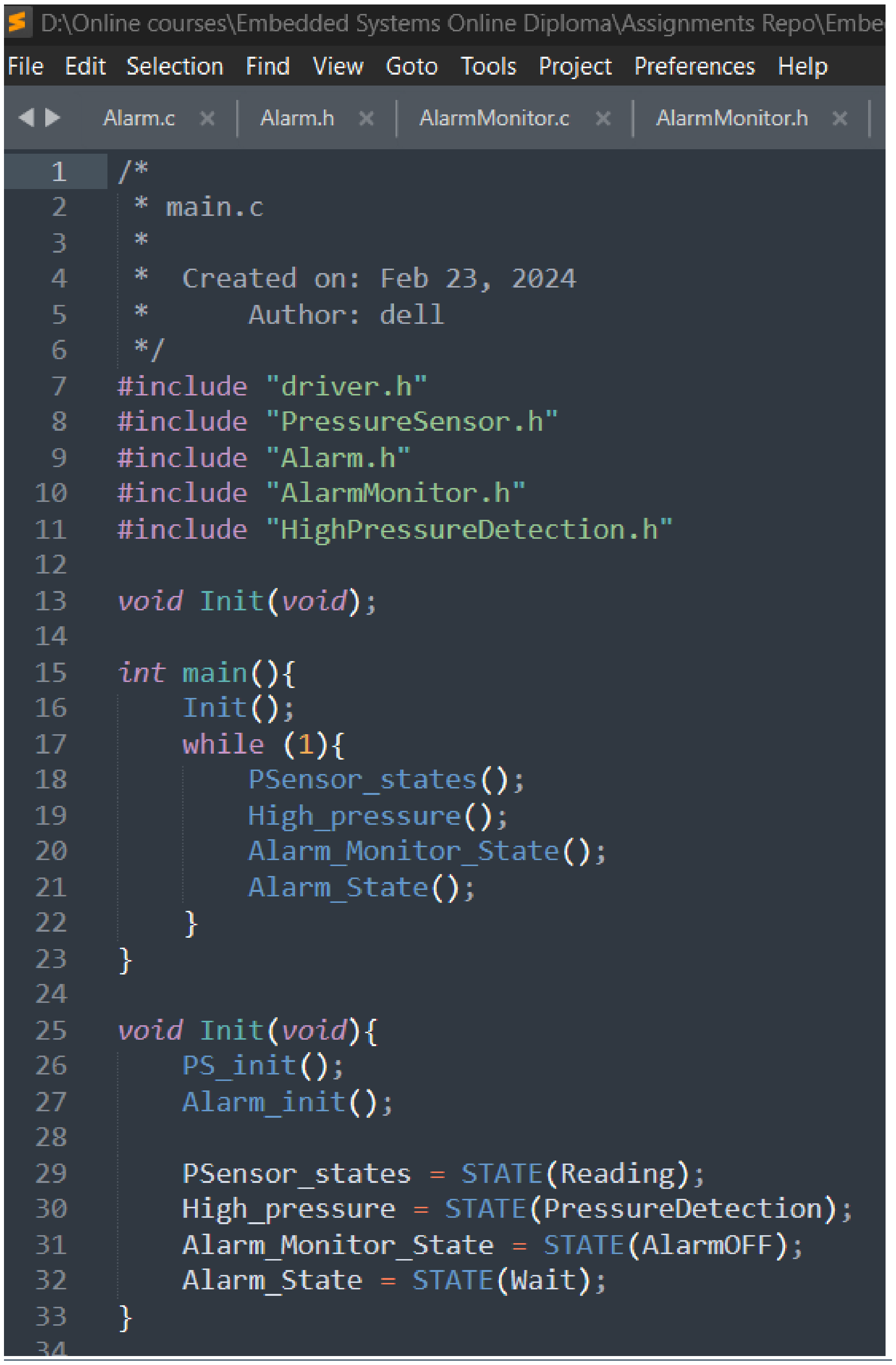

D:\Online courses\Embedded Systems Online Diploma\Assignme
File Edit Selection Find View Goto Tools Project Preferen
Alarm.c
Alarm.h
AlarmMonitor.c
AlarmM

```c
#include "driver.h"
#include <stdint.h>
#include <stdio.h>
void Delay(unsigned int nCount)
{
    for(; nCount != 0; nCount--);
}

int getPressureVal(){
    return (GPIOA_IDR & 0xFF);
}

void Set_Alarm_actuator(int i){
    if (i == 1){
        SET_BIT(GPIOA_ODR,13);
    }
    else if (i == 0){
        RESET_BIT(GPIOA_ODR,13);
    }
}

void GPIO_INITIALIZATION (){
    SET_BIT(APB2ENR, 2);
    GPIOA_CRL &= 0xFF0FFFFF;
    GPIOA_CRL |= 0x00000000;
    GPIOA_CRH &= 0xFF0FFFFF;
    GPIOA_CRH |= 0x22222222;
}

```

D:\Online courses\Embedded Systems Online Diploma\Assignments Repo\Embedded_Systems_Online_Di
File Edit Selection Find View Goto Tools Project Preferences Help
Alarm.c
Alarm.h
AlarmMonitor.c
AlarmMonitor.h
driver.c
driver.h

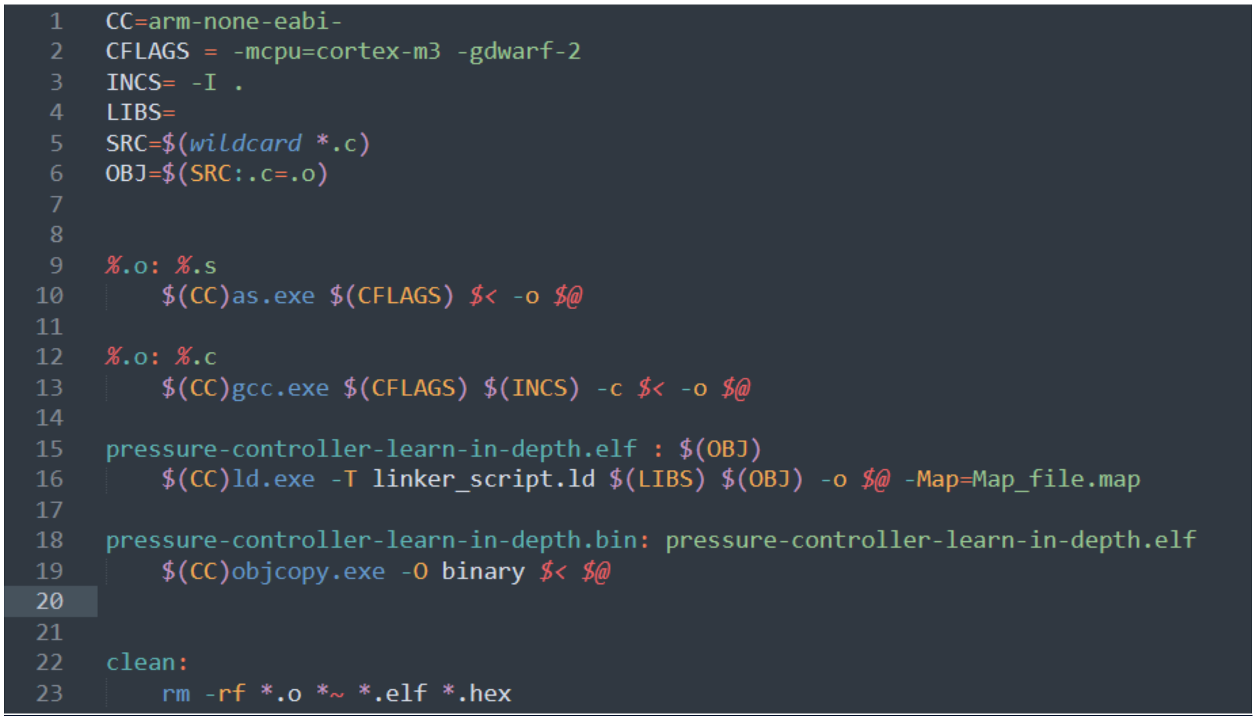
```c
#include <stdint.h>
#include <stdio.h>

#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(unsigned int nCount);
int getPressureVal();
void Set_Alarm_actuator(int i);
void GPIO_INITIALIZATION ();

```

-Main


D:\Online courses\Embedded Systems Online Diploma\Assignments Repo\Embe
File Edit Selection Find View Goto Tools Project Preferences Help
Alarm.c
Alarm.h
AlarmMonitor.c
AlarmMonitor.h

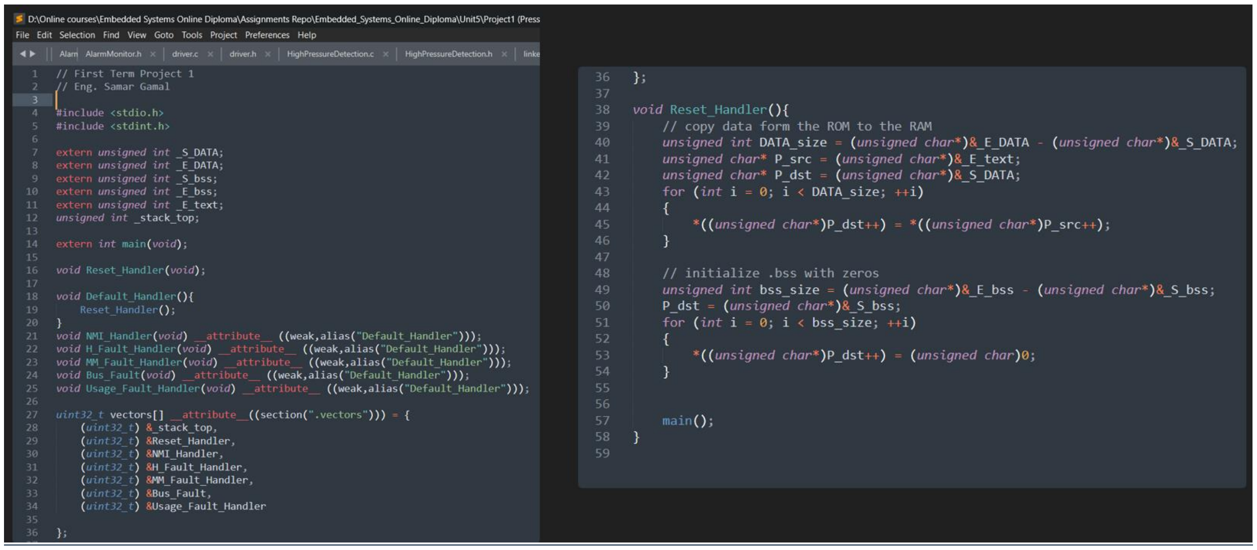
```c
/*
 * main.c
 *
 *  Created on: Feb 23, 2024
 *      Author: dell
 */
#include "driver.h"
#include "PressureSensor.h"
#include "Alarm.h"
#include "AlarmMonitor.h"
#include "HighPressureDetection.h"

void Init(void);

int main(){
    Init();
    while (1){
        PSensor_states();
        High_pressure();
        Alarm_Monitor_State();
        Alarm_State();
    }
}

void Init(void){
    PS_init();
    Alarm_init();

    PSensor_states = STATE(Reading);
    High_pressure = STATE(PressureDetection);
    Alarm_Monitor_State = STATE(AlarmOFF);
    Alarm_State = STATE(Wait);
}
```

-makefile



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


%.o: %.s
	$(CC)as.exe $(CFLAGS) $< -o $@

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

pressure-controller-learn-in-depth.elf : $(OBJ)
	$(CC)ld.exe -T linker_script.ld $(LIBS) $(OBJ) -o $@ -Map=Map_file.map

pressure-controller-learn-in-depth.bin: pressure-controller-learn-in-depth.elf
	$(CC)objcopy.exe -O binary $< $@


clean:
	rm -rf *.o *~ *.elf *.hex
```

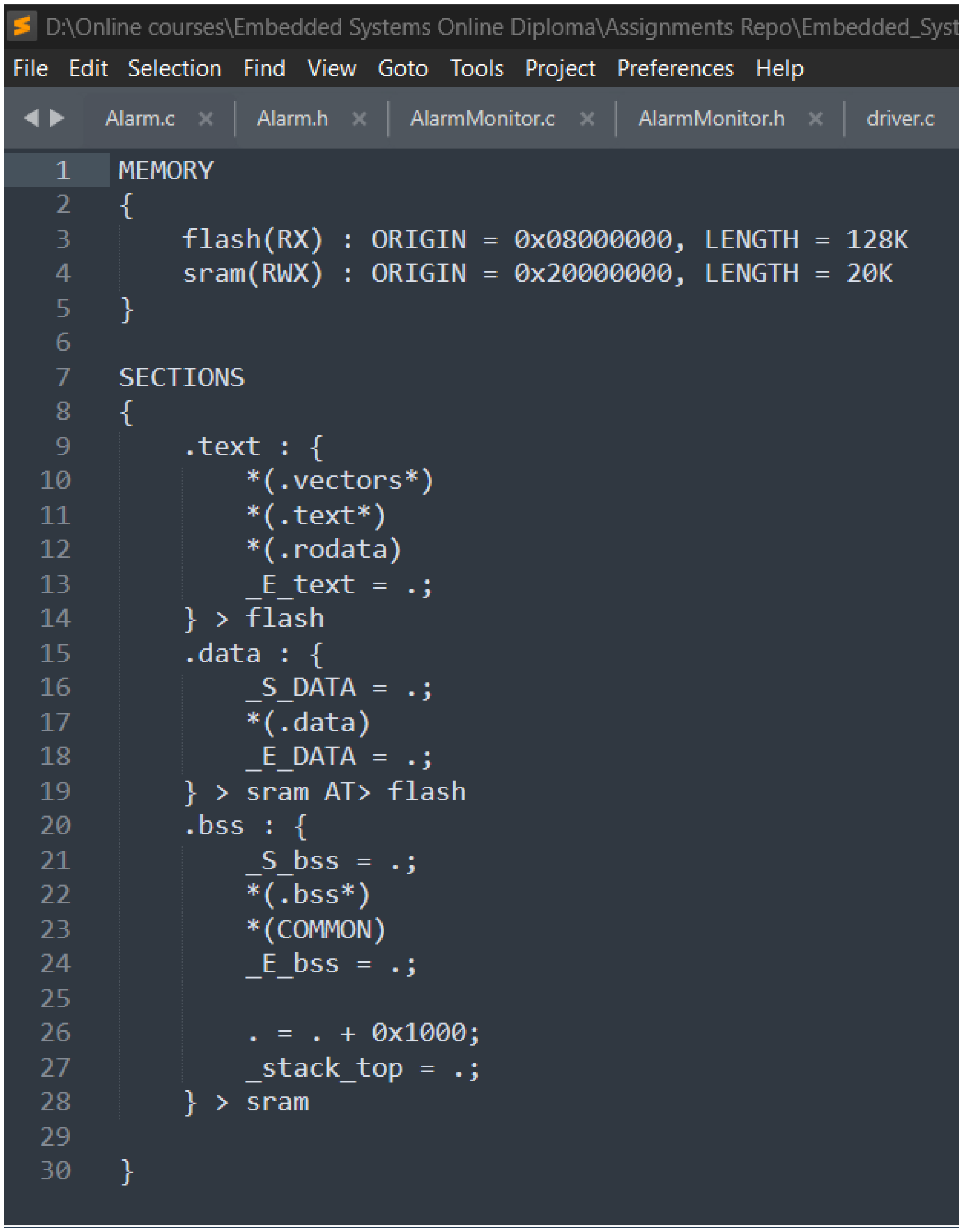
-Startup



D:\Online courses\Embedded Systems Online Diploma\Assignments Repo\Embedded_Systems_Online_Diploma\Unit5\Project1 (Press
File Edit Selection Find View Goto Tools Project Preferences Help
Alarm AlarmMonitor.h driver.c driver.h HighPressureDetection.c HighPressureDetection.h linke

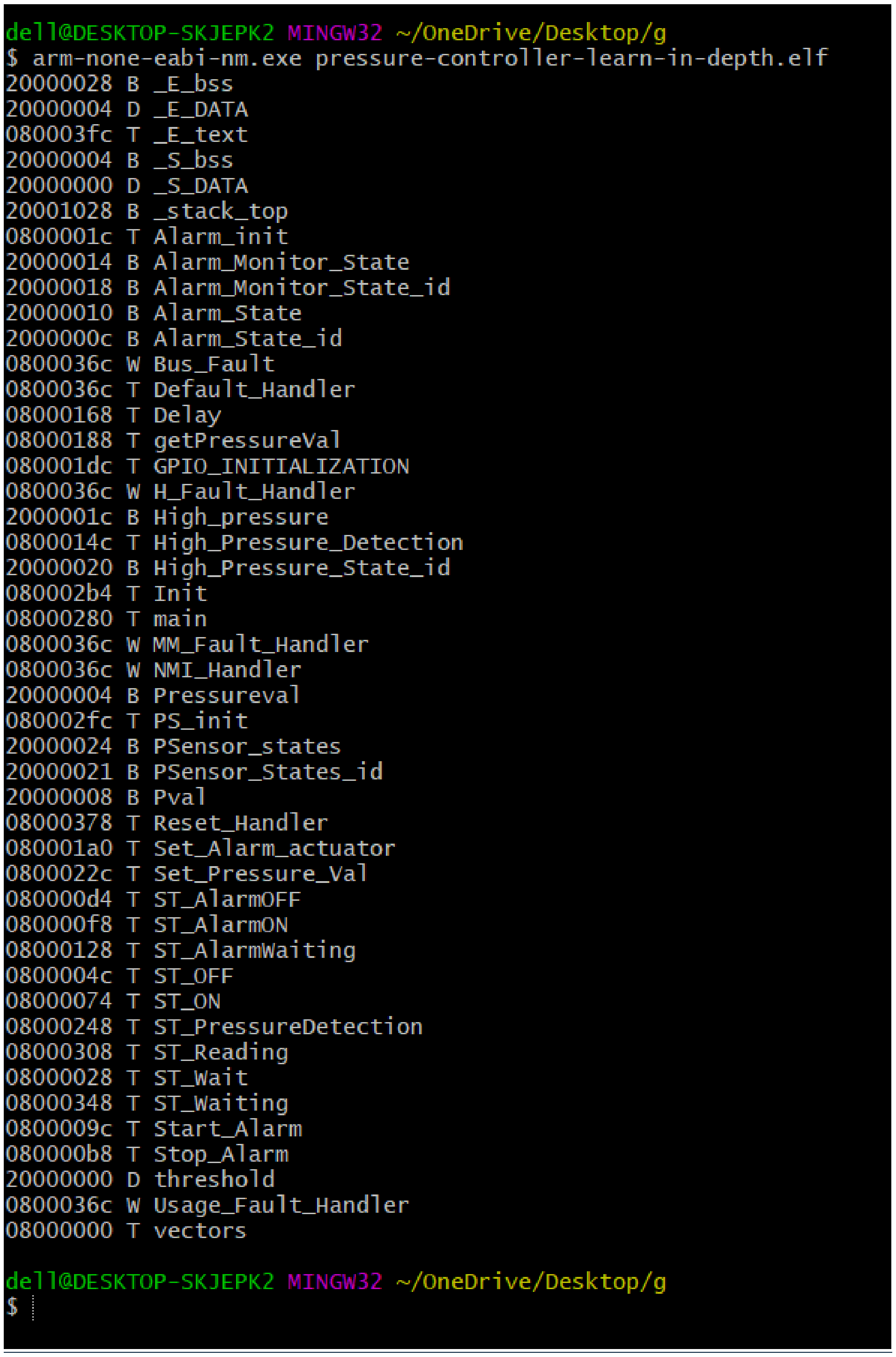
```c
// First Term Project 1
// Eng. Samar Gamal

#include <stdio.h>
#include <stdint.h>

extern unsigned int _S_DATA;
extern unsigned int _E_DATA;
extern unsigned int _S_bss;
extern unsigned int _E_bss;
extern unsigned int _E_text;
unsigned int _stack_top;

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")));

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

};

void Reset_Handler(){
    // copy data form the ROM to the 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 (int i = 0; i < DATA_size; ++i)
    {
        *((unsigned char*)P_dst++) = *((unsigned char*)P_src++);
    }

    // initialize .bss with zeros
    unsigned int bss_size = (unsigned char*)&_E_bss - (unsigned char*)&_S_bss;
    P_dst = (unsigned char*)&_S_bss;
    for (int i = 0; i < bss_size; ++i)
    {
        *((unsigned char*)P_dst++) = (unsigned char)0;
    }


    main();
}

```

-LinkerScript


D:\Online courses\Embedded Systems Online Diploma\Assignments Repo\Embedded_Syst
File Edit Selection Find View Goto Tools Project Preferences Help
Alarm.c
Alarm.h
AlarmMonitor.c
AlarmMonitor.h
driver.c

```
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)
        _E_DATA = .;
    } > sram AT> flash
    .bss : {
        _S_bss = .;
        *(.bss*)
        *(COMMON)
        _E_bss = .;

        . = . + 0x1000;
        _stack_top = .;
    } > sram

}
```

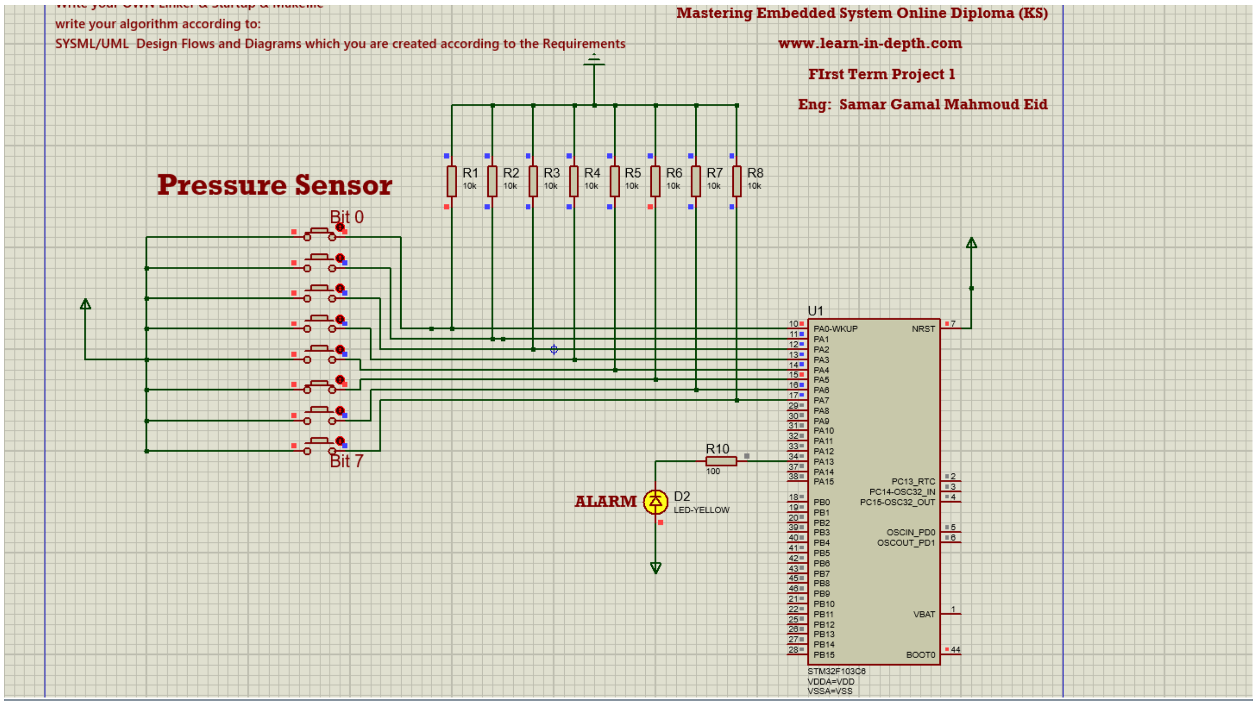
Symbol Table



```
dell@DESKTOP-SKJEPK2 MINGW32 ~/OneDrive/Desktop/g
$ arm-none-eabi-nm.exe pressure-controller-learn-in-depth.elf
20000028 B _E_bss
20000004 D _E_DATA
080003fc T _E_text
20000004 B _S_bss
20000000 D _S_DATA
20001028 B _stack_top
0800001c T Alarm_init
20000014 B Alarm_Monitor_State
20000018 B Alarm_Monitor_State_id
20000010 B Alarm_State
2000000c B Alarm_State_id
0800036c W Bus_Fault
0800036c T Default_Handler
08000168 T Delay
08000188 T getPressureVal
080001dc T GPIO_INITIALIZATION
0800036c W H_Fault_Handler
2000001c B High_pressure
0800014c T High_Pressure_Detection
20000020 B High_Pressure_State_id
080002b4 T Init
08000280 T main
0800036c W MM_Fault_Handler
0800036c W NMI_Handler
20000004 B Pressureval
080002fc T PS_init
20000024 B PSensor_states
20000021 B PSensor_States_id
20000008 B Pval
08000378 T Reset_Handler
080001a0 T Set_Alarm_actuator
0800022c T Set_Pressure_Val
080000d4 T ST_AlarmOFF
080000f8 T ST_AlarmON
08000128 T ST_AlarmWaiting
0800004c T ST_OFF
08000074 T ST_ON
08000248 T ST_PressureDetection
08000308 T ST_Reading
08000028 T ST_Wait
08000348 T ST_Waiting
0800009c T Start_Alarm
080000b8 T Stop_Alarm
20000000 D threshold
0800036c W Usage_Fault_Handler
08000000 T vectors

dell@DESKTOP-SKJEPK2 MINGW32 ~/OneDrive/Desktop/g
$
```

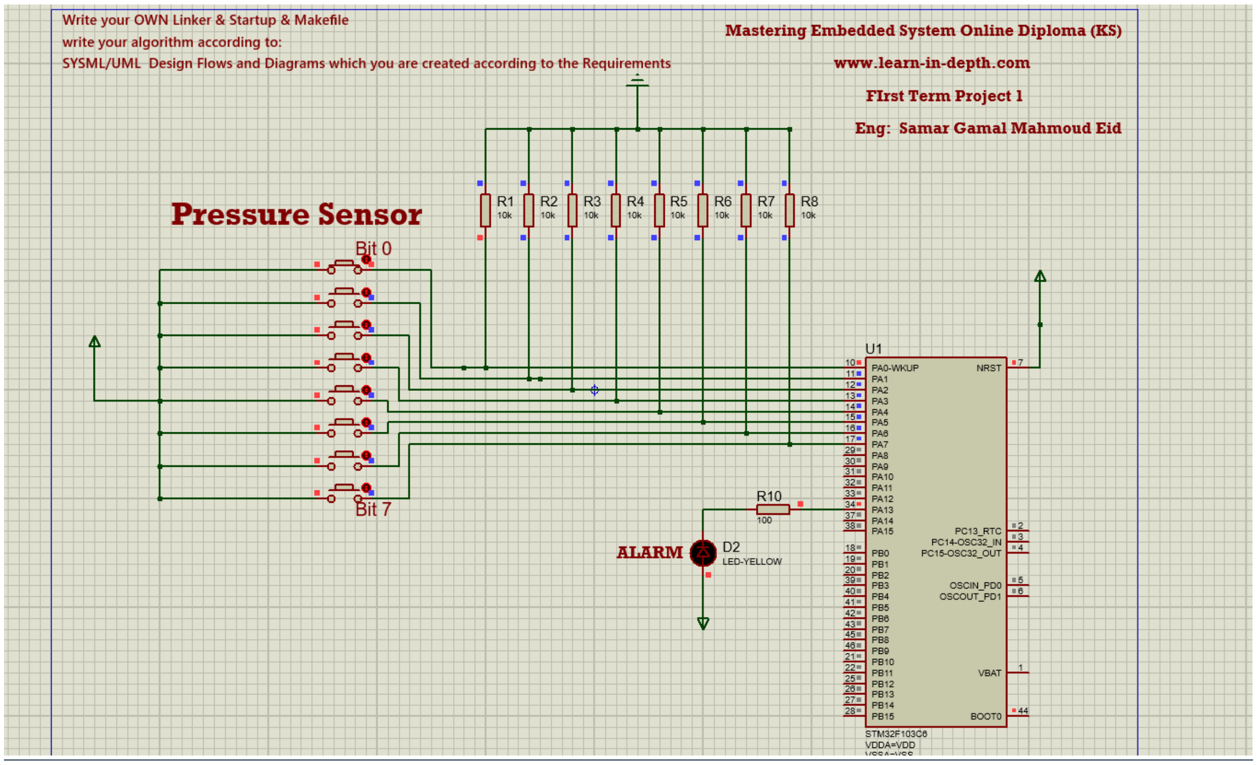
Simulation

-Pressure = 33 bars



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: Samar Gamal Mahmoud Eid
Pressure Sensor
Bit 0
Bit 7
R1 10k
R2 10k
R3 10k
R4 10k
R5 10k
R6 10k
R7 10k
R8 10k
U1
R10 100
ALARM
D2 LED-YELLOW
STM32F103C6
VDDA=VDD
VSSA=VSS

-Pressure = 1 bar



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: Samar Gamal Mahmoud Eid
Pressure Sensor
Bit 0
Bit 7
R1 10k
R2 10k
R3 10k
R4 10k
R5 10k
R6 10k
R7 10k
R8 10k
U1
R10 100
ALARM
D2 LED-YELLOW
STM32F103C6
VDDA=VDD