Mastering Embedded System

Online Diploma

[www.learn-in-depth.com](https://www.learn-in-depth.com/)

First Term (Final Project 1)

Eng. Omar Khaled M. Anwer

My Profile:

# Case Study

The inside of an airplane

Description automatically generated with medium confidence

**Specifications:**

* A pressure controller should inform the crew of a cabin  
  with an alarm when the pressure exceeds 20 bars in the cabin.
* The alarm duration equals 60 seconds.
* The system keeps tracking the measured values.

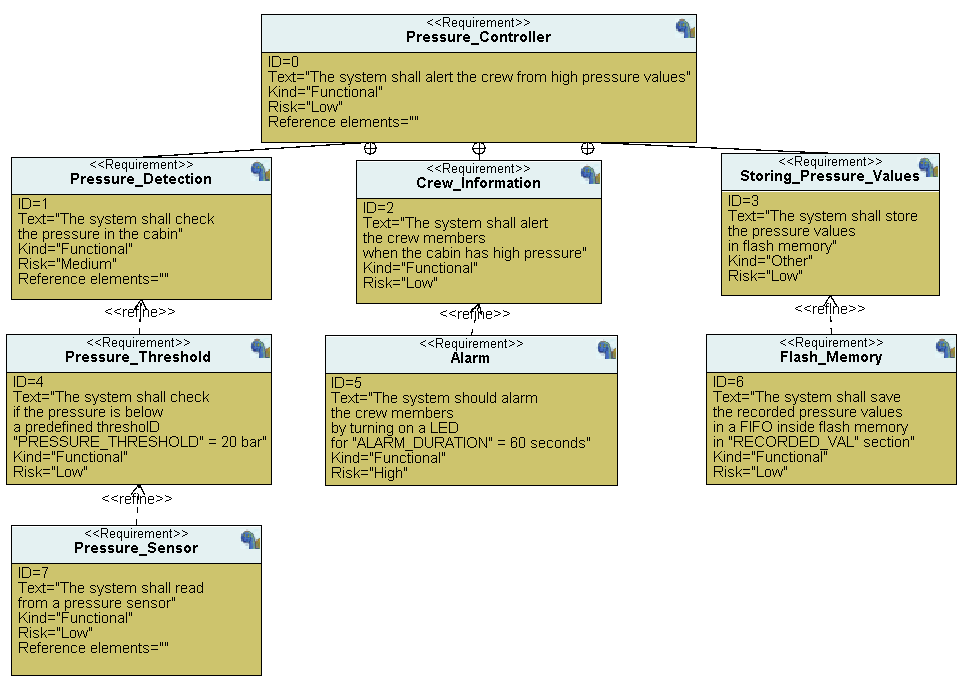
**Pressure Controller Assumptions**

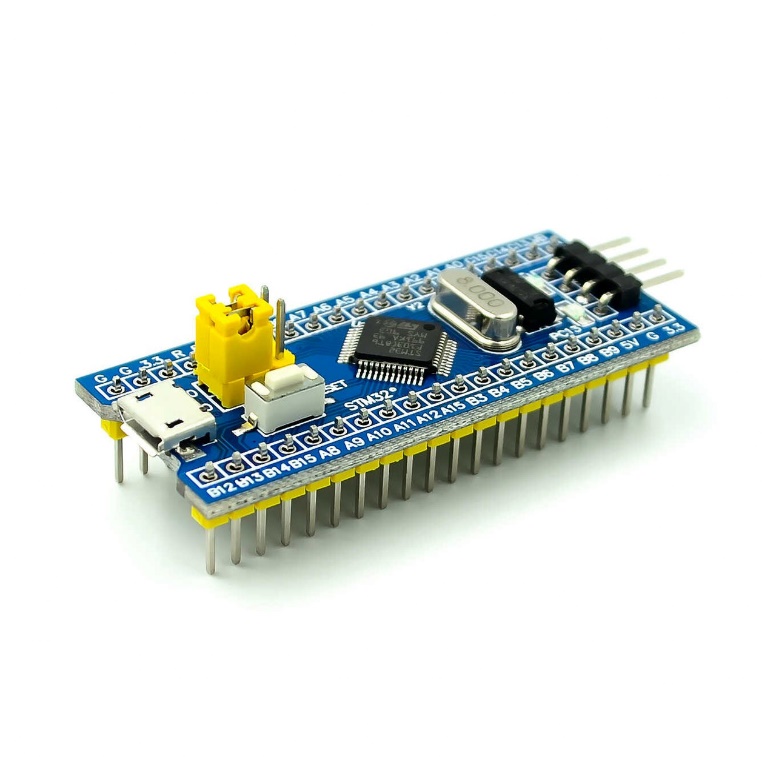
* The controller set up and shutdown procedures are not modeled.
* The controller maintenance is not modeled.
* The pressure sensor never fails.
* The alarm never fails.
* The controller never faces power cut.

**Versioning:**

The “keeps tracking the measured values” option is not modeled in the first version of the design.

# Requirement Diagram



System Analysis

# Use Case Diagram

Diagram

Description automatically generated

# Activity Diagram

Diagram

Description automatically generated

# Sequence Diagram

Timeline

Description automatically generated

This sequence diagram describes a scenario of the system.

System Design

A picture containing application

Description automatically generated

# Application

Diagram

Description automatically generated

# Pressure Sensor

Diagram

Description automatically generated

# Alarm

Diagram

Description automatically generated

Compilation

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

Table

Description automatically generated

# Makefile

#@copyright : Omar Anwer

#Project name

**PROJECT\_NAME** = Pressure\_Controller

#Architectures Specific Flags

**CPU** = cortex-m3

**CFLAGS\_ARCH** = **-mcpu**=$(CPU)

#CFLAGS\_ARCH = -mcpu=$(CPU) -m$(ARCH) --specs=$(SPECS)

#Compiler Flags and Defines

**CC** = arm-none-eabi-

**DBGCFLAGS** = -g -gdwarf-2

**CFLAGS** = -ansi **-std**=c89 -O0 -Wall $(CFLAGS\_ARCH) $(DBGCFLAGS)

#Linker Flags

**LINKER\_FILE** = linker\_script.ld

**LDFLAGS\_ARCH** = -T $(LINKER\_FILE)

**STARTUP\_FILE** = startup.s

#includes

**INCS** = -I .

**LIBS** =

#.c and .s files

**SRC** = **$(wildcard** \*.c**)**

**AS** = **$(wildcard** \*.s**)**

**PRE** = $(SRC:.c=.i)

**SRCOBJ** = $(SRC:.c=.o)

**ASOBJ** = $(AS:.s=.o)

all: $(PROJECT\_NAME).bin

@echo ""

@$(CC)size.exe $(PROJECT\_NAME).elf

@echo ""

@echo "Building done..."

%.o: %.s

$(CC)as.exe $(INCS) $< -o $@

@echo ""

%.o: %.c

$(CC)gcc.exe -S $(INCS) $<

@echo ""

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

@echo ""

%.i: %.c

$(CC)gcc.exe -E $(INCS) $< -o $@

@echo ""

$(PROJECT\_NAME).elf: $(ASOBJ) $(SRCOBJ) $(PRE)

$(CC)ld.exe $(LDFLAGS\_ARCH) $(LIBS) $(ASOBJ) $(SRCOBJ) -o $@ **-Map**=$(PROJECT\_NAME).map

@echo ""

$(CC)objdump.exe -h $@

@echo ""

$(CC)readelf.exe -S $@

@echo ""

@cp $(PROJECT\_NAME).elf $(PROJECT\_NAME).axf

$(PROJECT\_NAME).bin: $(PROJECT\_NAME).elf

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

@echo ""

$(CC)objcopy.exe -O ihex $< $(PROJECT\_NAME).hex

@echo ""

clean:

@rm \*.bin \*.hex \*.elf \*.axf \*.map

clean-all:

#@rm -rf $(filter-out $(STARTUP\_FILE), $(AS))

@rm \*.s

@rm \*.i \*.o \*.bin \*.hex \*.elf \*.axf \*.map

@echo "All cleaned..."

# Startup code

**#include** <stdint.h>

/\*#define stack\_top 0x20001000\*/

**extern** **int** **main**(**void**);

**extern** uint32\_t \_stack\_top;

**extern** uint32\_t \_E\_text;

**extern** uint32\_t \_S\_data;

**extern** uint32\_t \_E\_data;

**extern** uint32\_t \_S\_bss;

**extern** uint32\_t \_E\_bss;

**void** **Reset\_Handler**(**void**)

{

/\* copy .data section byte by byte from FLASH to SRAM \*/

uint8\_t\* pSrc = (uint8\_t\*)(&\_E\_text);

uint8\_t\* pDst = (uint8\_t\*)(&\_S\_data);

uint32\_t DATA\_SIZE = (uint8\_t\*)(&\_E\_data) - (uint8\_t\*)(&\_S\_data);

uint32\_t i;

**for**(i = 0; i < DATA\_SIZE; ++i)

{

\*pDst = \*pSrc;

pSrc++;

pDst++;

}

/\* initialize .data section in SRAM \*/

pDst = (uint8\_t\*)(&\_S\_bss);

uint32\_t BSS\_SIZE = (uint8\_t\*)(&\_E\_bss) - (uint8\_t\*)(&\_S\_bss);

**for**(i = 0; i < BSS\_SIZE; ++i)

{

\*pDst = 0;

pDst++;

}

/\* jump to main() \*/

main();

}

/\* initialize vectors \*/

**void** **Default\_Handler**(**void**)

{

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\_Handler**(**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\_Handler,

(uint32\_t) &Usage\_fault\_Handler,

};

# Linker Script

/\*

Linker script Cortex-M3

By Eng.Omar

\*/

ENTRY(Reset\_Handler)

MEMORY

{

FLASH (rx) : ORIGIN = 0x08000000, LENGTH = 64K

SRAM (rwx) : ORIGIN = 0x20000000, LENGTH = 20K

}

SECTIONS

{

.text :

{

\*(.vectors\*)

\*(.text\*)

\*(.rodata)

. = ALIGN(4);

\_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 = .; /\*stack top after 4 KB\*/

}>SRAM

}

# Map file

Memory Configuration

Name Origin Length Attributes

FLASH 0x08000000 0x00010000 xr

SRAM 0x20000000 0x00005000 xrw

\*default\* 0x00000000 0xffffffff

Linker script and memory map

.text 0x08000000 0x1f8

\*(.vectors\*)

.vectors 0x08000000 0x1c startup.o

0x08000000 vectors

\*(.text\*)

.text 0x0800001c 0x64 Alarm.o

0x0800001c Alarm\_Init

0x08000028 Set\_Alarm\_actuator

0x08000064 StartAlarm

0x08000072 StopAlarm

.text 0x08000080 0x38 Pressure\_Sensor.o

0x08000080 PressureSensor\_Init

0x08000098 getPressureVal

.text 0x080000b8 0x74 driver.o

0x080000b8 Delay

0x080000da GPIO\_INITIALIZATION

.text 0x0800012c 0x34 main.o

0x0800012c main

.text 0x08000160 0x98 startup.o

0x08000160 Reset\_Handler

0x080001ec MM\_fault\_Handler

0x080001ec Usage\_fault\_Handler

0x080001ec Bus\_fault\_Handler

0x080001ec Default\_Handler

0x080001ec H\_fault\_Handler

0x080001ec NMI\_Handler

\*(.rodata)

0x080001f8 . = ALIGN (0x4)

0x080001f8 \_E\_text = .

.glue\_7 0x080001f8 0x0

.glue\_7 0x080001f8 0x0 linker stubs

.glue\_7t 0x080001f8 0x0

.glue\_7t 0x080001f8 0x0 linker stubs

.vfp11\_veneer 0x080001f8 0x0

.vfp11\_veneer 0x080001f8 0x0 linker stubs

.v4\_bx 0x080001f8 0x0

.v4\_bx 0x080001f8 0x0 linker stubs

.iplt 0x080001f8 0x0

.iplt 0x080001f8 0x0 Alarm.o

.rel.dyn 0x080001f8 0x0

.rel.iplt 0x080001f8 0x0 Alarm.o

.data 0x20000000 0x0 load address 0x080001f8

0x20000000 \_S\_data = .

\*(.data)

.data 0x20000000 0x0 Alarm.o

.data 0x20000000 0x0 Pressure\_Sensor.o

.data 0x20000000 0x0 driver.o

.data 0x20000000 0x0 main.o

.data 0x20000000 0x0 startup.o

0x20000000 . = ALIGN (0x4)

0x20000000 \_E\_data = .

.igot.plt 0x20000000 0x0 load address 0x080001f8

.igot.plt 0x20000000 0x0 Alarm.o

.bss 0x20000000 0x1004 load address 0x080001f8

0x20000000 \_S\_bss = .

\*(.bss\*)

.bss 0x20000000 0x0 Alarm.o

.bss 0x20000000 0x4 Pressure\_Sensor.o

0x20000000 g\_pressureVal

.bss 0x20000004 0x0 driver.o

.bss 0x20000004 0x0 main.o

.bss 0x20000004 0x0 startup.o

0x20000004 \_E\_bss = .

0x20000004 . = ALIGN (0x4)

0x20001004 . = (. + 0x1000)

\*fill\* 0x20000004 0x1000

0x20001004 \_stack\_top = .

LOAD Alarm.o

LOAD Pressure\_Sensor.o

LOAD driver.o

LOAD main.o

LOAD startup.o

OUTPUT(Pressure\_Controller.elf elf32-littlearm)

LOAD linker stubs

.debug\_info 0x00000000 0x4be

.debug\_info 0x00000000 0xf9 Alarm.o

.debug\_info 0x000000f9 0xcc Pressure\_Sensor.o

.debug\_info 0x000001c5 0xc8 driver.o

.debug\_info 0x0000028d 0x9c main.o

.debug\_info 0x00000329 0x195 startup.o

.debug\_abbrev 0x00000000 0x28d

.debug\_abbrev 0x00000000 0x80 Alarm.o

.debug\_abbrev 0x00000080 0x66 Pressure\_Sensor.o

.debug\_abbrev 0x000000e6 0x7c driver.o

.debug\_abbrev 0x00000162 0x59 main.o

.debug\_abbrev 0x000001bb 0xd2 startup.o

.debug\_loc 0x00000000 0x2b0

.debug\_loc 0x00000000 0xe0 Alarm.o

.debug\_loc 0x000000e0 0x88 Pressure\_Sensor.o

.debug\_loc 0x00000168 0xa0 driver.o

.debug\_loc 0x00000208 0x2c main.o

.debug\_loc 0x00000234 0x7c startup.o

.debug\_aranges 0x00000000 0xa0

.debug\_aranges

0x00000000 0x20 Alarm.o

.debug\_aranges

0x00000020 0x20 Pressure\_Sensor.o

.debug\_aranges

0x00000040 0x20 driver.o

.debug\_aranges

0x00000060 0x20 main.o

.debug\_aranges

0x00000080 0x20 startup.o

.debug\_line 0x00000000 0x4b4

.debug\_line 0x00000000 0x10b Alarm.o

.debug\_line 0x0000010b 0x101 Pressure\_Sensor.o

.debug\_line 0x0000020c 0x106 driver.o

.debug\_line 0x00000312 0x4e main.o

.debug\_line 0x00000360 0x154 startup.o

.debug\_str 0x00000000 0x24b

.debug\_str 0x00000000 0x14a Alarm.o

0x18c (size before relaxing)

.debug\_str 0x0000014a 0x43 Pressure\_Sensor.o

0x194 (size before relaxing)

.debug\_str 0x0000018d 0x2a driver.o

0x17b (size before relaxing)

.debug\_str 0x000001b7 0xc main.o

0x157 (size before relaxing)

.debug\_str 0x000001c3 0x88 startup.o

0x1dc (size before relaxing)

.comment 0x00000000 0x49

.comment 0x00000000 0x49 Alarm.o

0x4a (size before relaxing)

.comment 0x00000049 0x4a Pressure\_Sensor.o

.comment 0x00000049 0x4a driver.o

.comment 0x00000049 0x4a main.o

.comment 0x00000049 0x4a startup.o

.ARM.attributes

0x00000000 0x2d

.ARM.attributes

0x00000000 0x2d Alarm.o

.ARM.attributes

0x0000002d 0x2d Pressure\_Sensor.o

.ARM.attributes

0x0000005a 0x2d driver.o

.ARM.attributes

0x00000087 0x2d main.o

.ARM.attributes

0x000000b4 0x2d startup.o

.debug\_frame 0x00000000 0x1b0

.debug\_frame 0x00000000 0x8c Alarm.o

.debug\_frame 0x0000008c 0x50 Pressure\_Sensor.o

.debug\_frame 0x000000dc 0x58 driver.o

.debug\_frame 0x00000134 0x2c main.o

.debug\_frame 0x00000160 0x50 startup.o

Simulation

Diagram, schematic

Description automatically generatedDiagram, schematic

Description automatically generated

Diagram, schematic

Description automatically generated