ELECTRIC HEATER

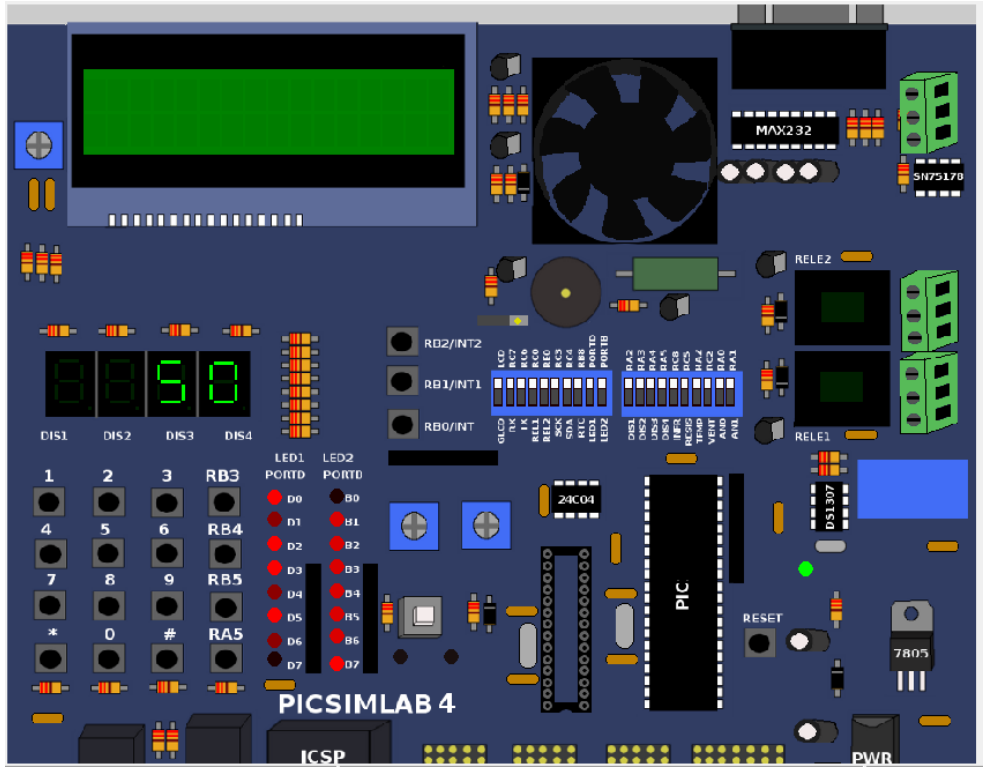
SWIFT ACT Q3 INTERNSHIP PROJECT

BY: ABDELRAHMAN REFAAT & MUHAMMED DIAA

CONTENT

- User manual
- Context Diagram
- Modules Details
  - Module Description
  - Module functions and variables
  - Module Logic
- Timing Analysis

USER MANUAL



- Push buttons

i. RB0 : ON/OFF push button.

ii. RB1 : Down push button.

iii. RB2 : Up push button.

- Sensors

i. RA2/AN2 : Temperature sensor (LM35).

- Temperature control elements

i. RC5 : Heater.

ii. RC2 : Cooler (Fan).

- Two multiplexed 7 Segment Display

i. PORTD : The two 7 segment data pins.

ii. RA4 : Middle Right 7 segment enable pin.

iii. RA5 : Right 7 segment enable pin.

- External EEPROM : SDA/SCK pins
- Heating Element LED : RB7

USER MANUAL

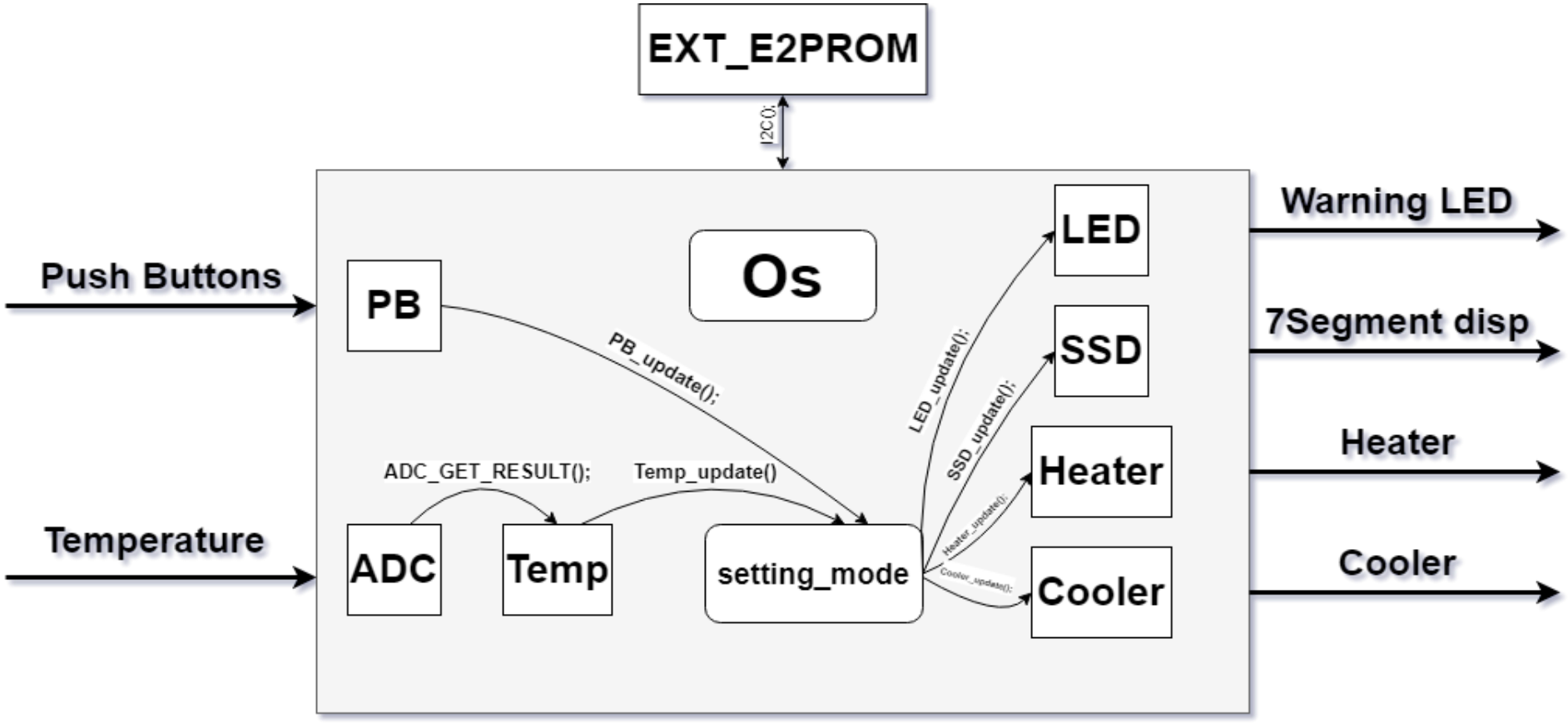
- The “Up” or “Down” buttons are used to change the required water temperature.
- If the “ON/OFF” button is released and the electric water heater is in OFF state, the electric water heater goes to ON state and vice versa.
- If the electric water heater is turned OFF then ON, the last temperature value you set will be displayed.
- If you want to change the water temperature, press up or down key. When you reach out the desired temperature value, Don't press any key for 5 Seconds.
- The desired set value will be displayed on the Two 7 segments.

USER MANUAL

HOW TO USE

1) Press power button to turn on the electric heater system.
2) Press up or down button to set your desired water temperature value.
3) After You set the desired value ,You should wait 5 seconds until the 7 segment get out from the blinking mode display to fixed mode display.
4) The system will process your desired temperature and the water temperature to reach out your desired temperature.
5) Finally, if you want to turn off the electric heater system, press on power button.

CONTEXT DIAGRAM



MODULES DETAILS

- Setting_mode module

❖ Module Description

✓ Setting_mode module is considered as the heart of the whole electric heater system , we can say that it is the processing unit of the system.

✓ This module communicates with most of other system modules through three public structures, one for flags ,one for counters and the last one for temperature readings(user set value & temp sensor value).

MODULES DETAILS

- Modules functions and variables
  - Variables
    - ✓ Counter Struct

variable	Description
FixedMode_counter	Count until 5 seconds then goes into fixed mode.
Blink_counter	Count the 1 second between ON and OFF states for SSD in Blinking mode.
Led_counter	Count the 1 second of Heating element Led Toggling.
Temp_counter	Count the 100 ms between every two readings of temp sensor.
e2prom_count	Counter used to make eeprom receive the set value when the system gets into fixed mode.

MODULES DETAILS

- Variables
  - ✓ Flag Struct

variable	Description
next_state	Update SSD state in Blinking mode.
Toggle_led	Update Heating element Led state (0:ON 1:Toggle 2:OFF)
Heater_Operation	Update Heater State.
Cooler_Operation	Update Cooler State.
Operation_Flag	Update Electric Heater State.
e2prom_flag	Check if this is the first use of the Electric Heater or not

MODULES DETAILS

- Variables
  - ✓ Reading Struct

variable	Description
temp_read	store the temp sensor current reading.
Set_value	store the set value that user entered.

MODULES DETAILS

- Functions

Function name &Arguments	Return type	State	Description
SettingMode_Blinking_mode (void);	void	static	Responsible for blinking the SSD every one second. In this mode, function turns off heater ,cooler and Heating element led through changing their flag values. Related counters update theirself at the end of the function. This functions is called by setting mode update function every 5ms if the system operation mode is at ON mode.
SettingMode_Fixed_mode(void);	void	static	This function is called by setting mode update function if and only if the user doesn't press any key for 5s. This function Change the SSD state from blinking state to normal operation state. It also compares the user set value with the water temp value to determine which elements will be turned on or turned off.

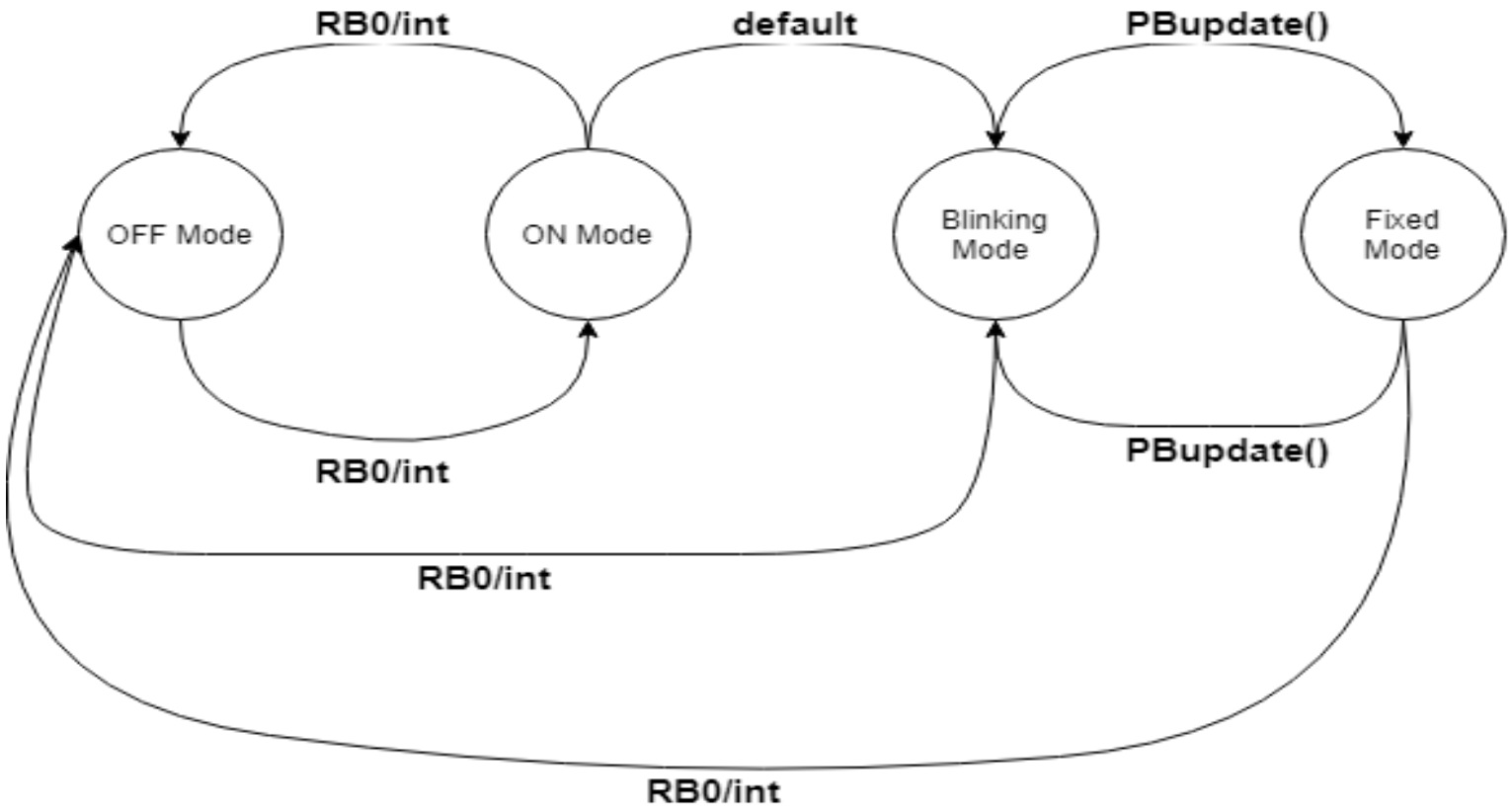
MODULE DETAILS

- Functions

Function name &Arguments	Return type	State	Description
SettingMode_OFF_mode(void);	void	static	This function is called if and only if the System operating flag changes from ON_MODE to OFF_MODE. It is responsible for turn off the whole the system and initialize any counter to zero because when the system is run again , it initialize the system to be at the beginning state.
SettingMode_update(void);	void	Public	In OFF_MODE,it calls OFF_mode function. In ON_MODE ,If the user doesn't press any key for 5s. It calls Fixed_mode function else it calls blinking mode function. This function is called by timer update function every 20ms

MODULES DETAILS

- Module logic
  - State machine diagram



MODULES DETAILS

- push_button module

❖ Module Description

✓ This module check for the up and down push buttons state to determine whether the set value will be incremented or be decremented.

✓ If the user sets a temperature value exceeds or becomes less than the determined temperature limits(35-75), the set value will get fixed.

✓ This module resets the Fixed mode counter when up or down key pressed to count new 5s.

MODULES DETAILS

- Module Functions and Variables
- Variables
- ✓ This module uses one variable from Readings struct(Set_value) and one variable from Counters struct (FixedMode_counter).
- Data types
- ✓ This module defines two unique data types:-

  tPB:represents the push buttons of the system (Up,Down).

  tPB_state: represents the state of the push_button(PB_RELEASED, PB_PRE_PRESSED,PB_PRESSED,PB_PRE_RELEASED).

MODULES DETAILS

- Functions

Function name	Return type	state	Description
PB_Init(tPB pb, tPB_State initial_state);	void	public	Configure the push buttons pins as input pins and initialize the state of push buttons with the desired initial state.
PB_Update(void);	void	public	Updates push buttons states and calls SetValue_update() function. This function is called by timer every 5ms.
PB_GetState(tPB pb);	tPB_State	public	Return the state of the push button wether it is pressed or released
SetValue_update(void);	void	static	Check for the up and down push buttons state. If anyone of them is pressed ,the Fixed counter will be initialized to 0.Up button increment the set_value five degrees.Down button decrement it five degrees.If set_value reach out the temprature uppper or lower limit , it get fixed until you press the opposite button again.

MODULES DETAILS

- SSD module

❖ Description

✓ This module displays the user set value

✓ It alternates between two differrent modes, Fixed mode ,the mode that the set value is normally displayed, Blinking mode ,the mode that the set value is displayed for one second and turned off for one second.

✓ SSD module has an extra feature,it can display current temprature reading instead of the set value in fixed mode.you can adapt it through setting a flag in the module header file.

MODULES DETAILS

- Modules Functions and Variables
- Variables
  - ✓ Array symbols[11]: Seven segment display lookup table.
  - ✓ Array SSD_symbols[2]: stores the data that should be displayed on the two 7 segments.
  - ✓ This module uses two variables from Readings public struct(Set_value,Temp_read) and one variable from Flags public struct (next_state).
- Data types
  - ✓ tSSD: represents the SSDs that connected to the system(SSD_MR,SSD_R).
  - ✓ SSD_symbol:represents the symbols that will be displayed on the 7 segments and used to access the lookup table.

MODULES DETAILS

- Functions

Function name	Return type	state	Description
SSD_Init(tSSD ssd ,SSD_STATE initial_state , SSD_symbol initial_symbol);	void	public	Configure SSD Data pins as output pins and SSD Enable pin as output pin then it initialize the SSD state with the passed symbol.
SSD_SET_Symbol(tSSD ssd ,SSD_symbol symbol);	void	public	Initialize SSD_symbols array with the passed symbol in the proper SSD element.
SSD_SET_state(tSSD ssd ,SSD_STATE state);	void	public	Enable or disable the passed SSD.
SSD_GET_state(tSSD ssd);	SSD_STATE	public	Return the passed SSD state
SSD_GET_Symbol(tSSD ssd)	SSD_symbol	public	Return the passed SSD symbol
SSD_update();	void	public	Updates SSD_symbols array with the set_value.Then,displays the updated value.

MODULES DETAILS

- Temp module
- Module Description

✓ This module is responsible for updating temp_read value every 100ms according to the following equation:-

temp_read =(ADC_Current_value*150*5)/(1.5*1023)

MODULES DETAILS

- Module Functions and Variables
- Variables
- ✓ This module uses one variable from Readings public struct(Temp_read) and one variable from Counters public struct (Temp_counter).
- Functions

Function name &Arguments	Return type	State	Description
Temprature_Init(void);	void	public	This function only calls the ADC_Init function to initialize the ADC.
Temprature_Read(void);	void	Static	Responsible of mapping ADC_Result value to temperature value.
Temprature_update(void);	void	public	This function only Calls Temprature_Read(void);

MODULES DETAILS

- Module logic

- ✓ Conversion equation explanation.

```
LM35 calculations
=================
5v   (vref+)       ===> 1023(ADC_max)
1.5v(max volt)     ===> max_ADC_value_for_temp_sensor

150(max_temp_value) ===> max_ADC_value_for_temp_sensor
temp                ===> ADC_READ_RESULT()
```

- ✓ Temperature_Read function updates temp_read variable with the current sensor reading value every 100ms. Temp_counter handles this task.

MODULES DETAILS

- Heater module

❖ Module description

✓ This module control the state of the heater element in various modes of system operation.

❖ Module Functions and Variables

❖ variables

✓ This module uses one variable from Flags public struct(Heater_Operation).

MODULES DETAILS

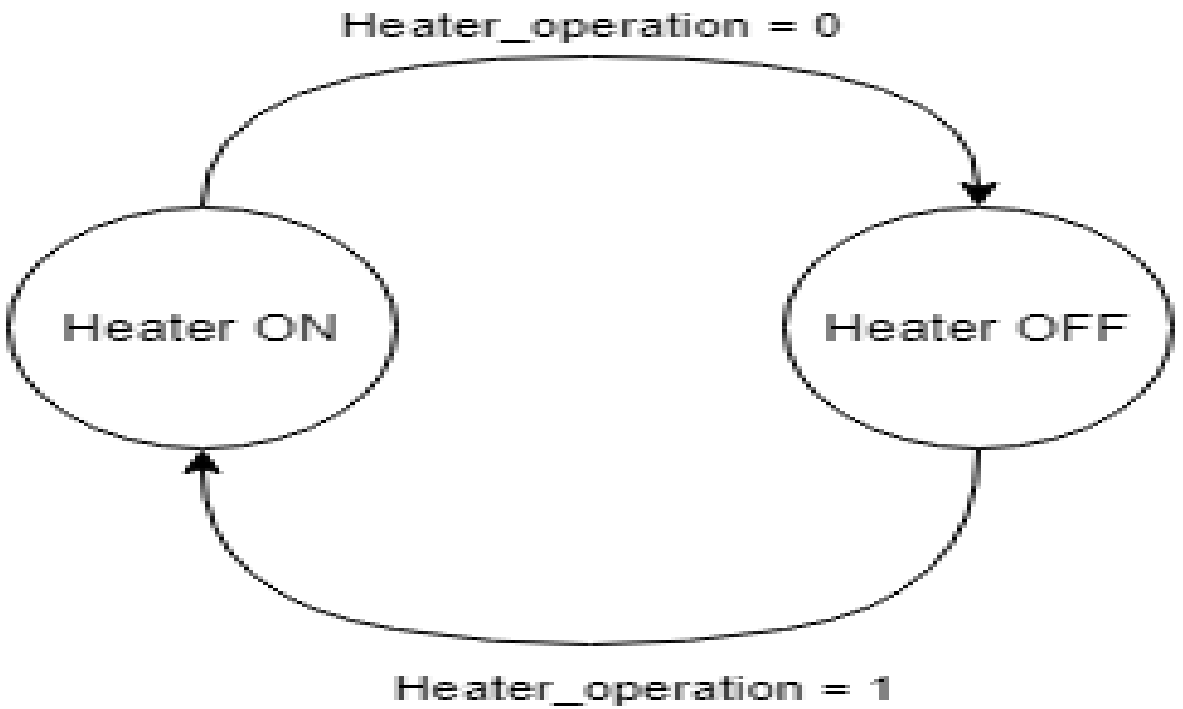
- Functions

Function name &Arguments	Return type	State	Description
Heater_Init(void);	void	public	Configure heater element pin to be output pin.
Heater_SetState(unsigned char state);	void	Static	Controls the state of heater element
Heater_GetState(void);	unsigned char	public	Return the state of heater element
Heater_update(void);	void	public	Calls Heater_SetState function and pass to it the current state.

MODULES DETAILS

- Module Logic

✓ Setting_mode module updates Heater_opertation flags according to the system current mode.



MODULES DETAILS

- Cooler module

❖ Module description

✓ This module control the state of the Cooler element in various modes of system operation.

❖ Module Functions and Variables

❖ variables

✓ This module uses one variable from Flags public struct(Cooler_Operation).

MODULES DETAILS

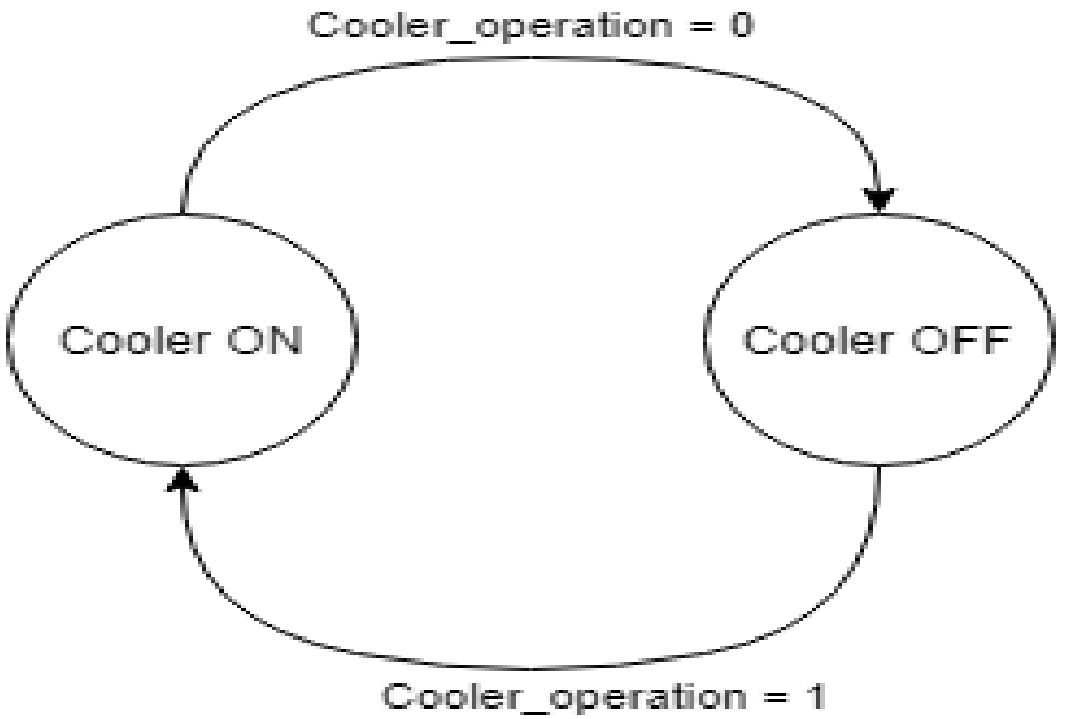
- Functions

Function name &Arguments	Return type	State	Description
Cooler_Init(void);	void	public	Configure cooler element pin to be output pin.
Cooler_SetState(unsigned char state);	void	Static	Controls the state of cooler element
Cooler_GetState(void);	unsigned char	public	Return the state of cooler element
Cooler_update(void);	void	public	Calls Cooler_SetState function and pass to it the current state.

MODULES DETAILS

- Module Logic

✓ Setting_mode module updates Cooler_opertation flags according to the current mode.



MODULES DETAILS

- LED module
- ❖ Module description
- ✓ This module control the state of the Heating element LED in various modes of system operation.

- ❖ Module Functions and Variables
- ❖ variables
- ✓ This module uses one variable from Flags public struct(Toggle_LED) and one variable from counters public struct(LED_counter).
- ❖ Data types
- ✓ tLED: store the number of led you want to access (LED_0,LED_1, ... , LED_7).
- ✓ TLED_State: store the led state(LED_OFF,LED_OFF).

MODULES DETAILS

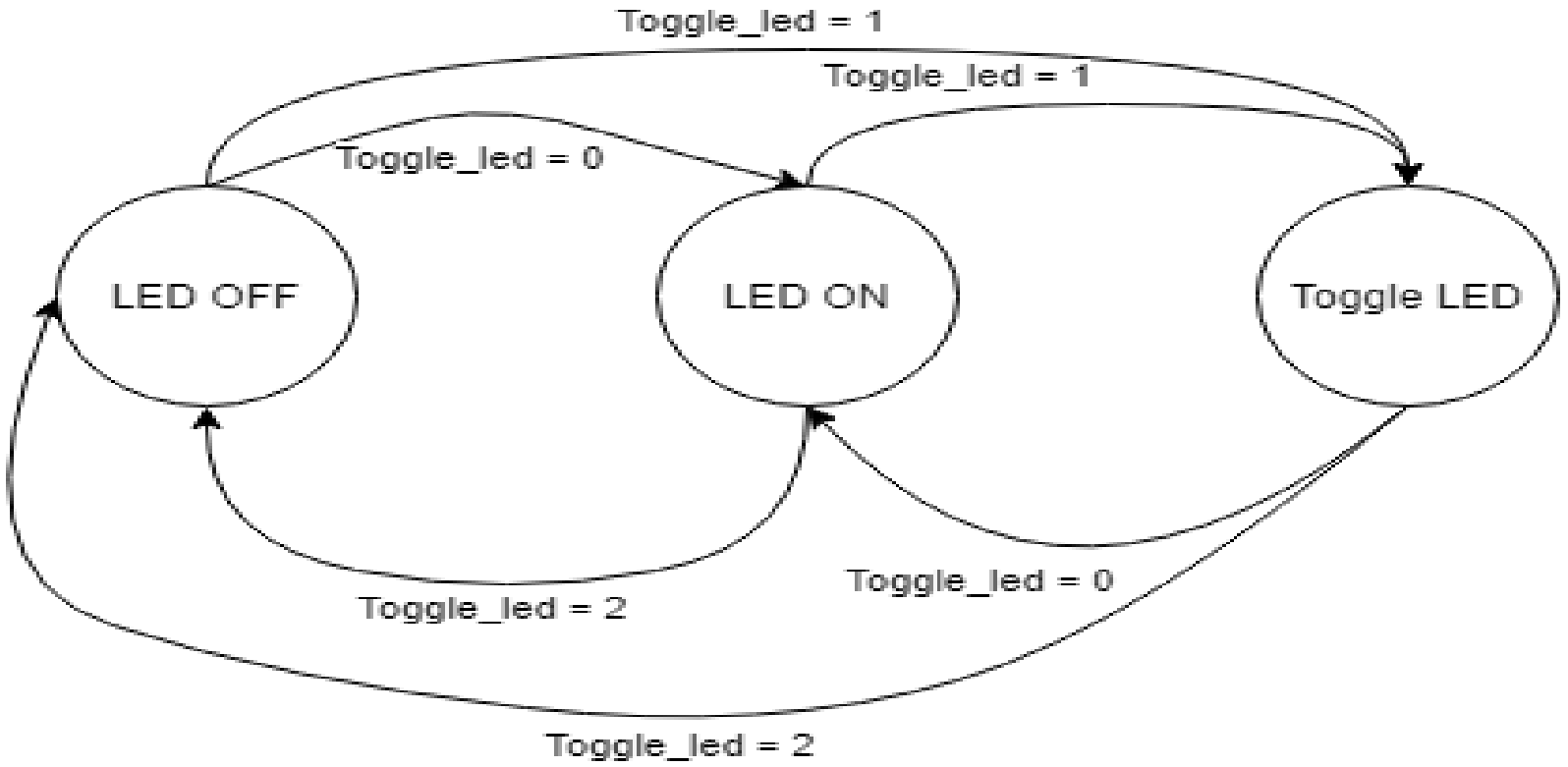
- Functions

Function name &Arguments	Return type	State	Description
LED_Init(tLED led, tLED_State initial_state);	void	public	Configure Heating element LED pin to be output pin and give it initial state.
LED_SetState(tLED led, tLED_State state);	void	Static	Controls the state of Heating element LED
LED_GetState(tLED led);	tLED_State	public	Return the state of Heating element LED
LED_Toggle(tLED led);	void	static	Change the state of Led every call.
void LED_Update(void);	void	public	Timer call this function every 5ms. Function check for the current led state and Set the proper state to the led

MODULES DETAILS

- Module Logic

✓ Setting_mode module updates Toggle_led flag according to the current mode.



MODULES DETAILS

- eeprom_ext module

❖ Description

✓ This module communicate with the system microcontroller throughout I2C communication protocol to store the last set value user entered before the system is turned off or power failure occurs.In the first use, Electric heater set_value is set by 60.

❖ Module Functions and Variables

❖ Variables

✓ This module uses one variable from Readings public struct(Set_value) and Two variable from counters public struct(e2prom_count,FixedMode_counter) and one variable from Flags public struct(e2prom_flag).

MODULES DETAILS

- Functions

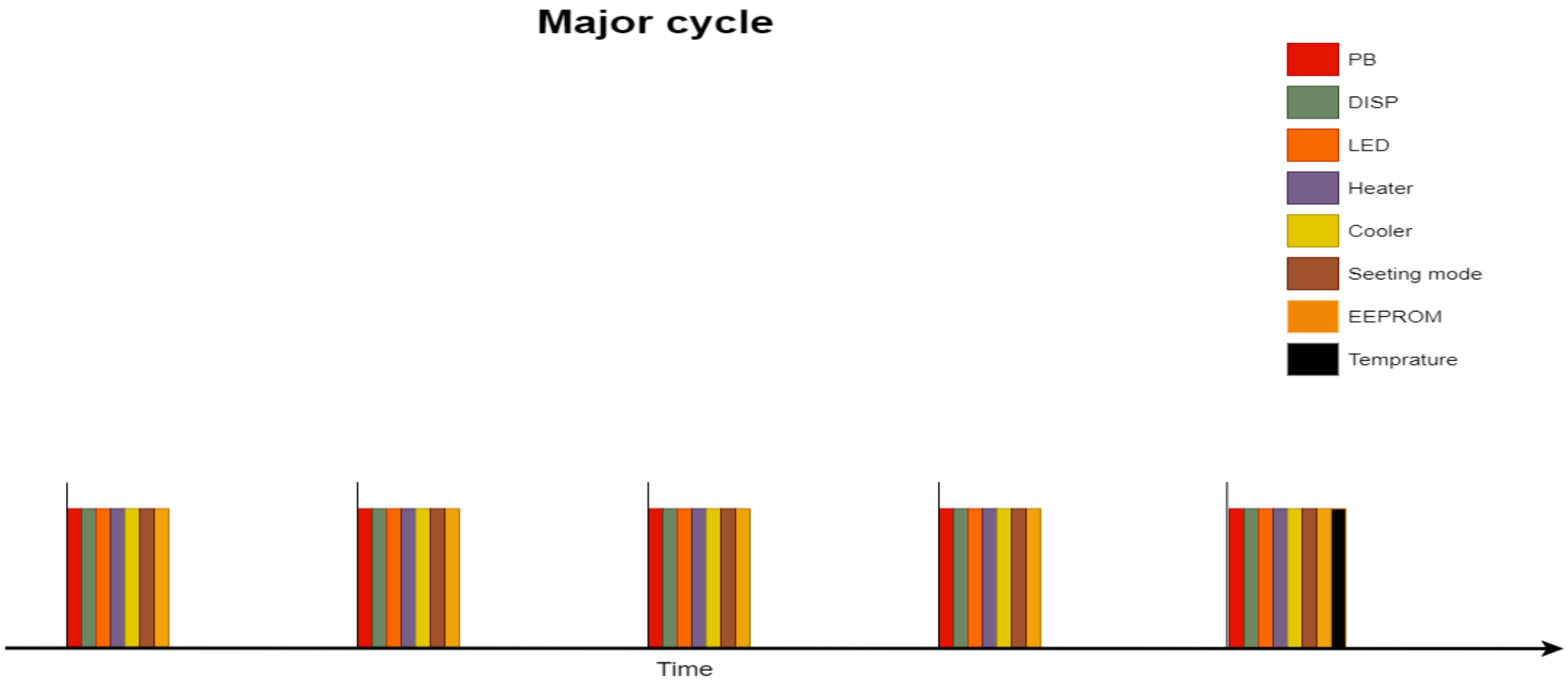
Function name &Arguments	Return type	State	Description
e2pext_r(unsigned int addr);	unsigned char	public	Read the data from the passed address.
e2pext_w(unsigned int addr, unsigned char val);	void	public	Write the passed data to the passed address.
e2pex_update(void);	void	public	If it's the first use of the electric heater.,it will assign the default value(60) to set_value.if not it will write the last Set_value the user set once the system get into the fixed mode and read the set value once the power is turned on.

TIME ANALYSIS

Task	Actions	BCET (ms)	WCET (ms)	Period of Action(ms)	Period of task(ms)
PB	Update samples Update set_value	0 0	0 0	20 20	20
DISP	Update SSD symbols Update display	0 4	0 4	20 20	20
Temprature	Update temperature	0	9	100	100
LED	Update LED state	0	0	20	20
Heater	Update Heater state	0	0	20	20
Cooler	Update Cooler state	0	0	20	20
SettingMode	Update Fixed_mode Update Blinking_mode	0 0	0.08 0.06	20 20	20
EEPROM	Read the first use set_value (60) Store the last set_value user entered Read the last set_value stored in eeprom	0 0 0	6.5 2.9 1.5	20 20 20	20
				TICK(ms) Major cycle	GCD = 20 MLS =100

TIMING ANALYSIS

- Major cycle = 5 minor cycle



THANK YOU

MUHAMMEDDIAA23@GMAIL.COM

ABDELRAHMANREFAAT601@GMAIL.COM