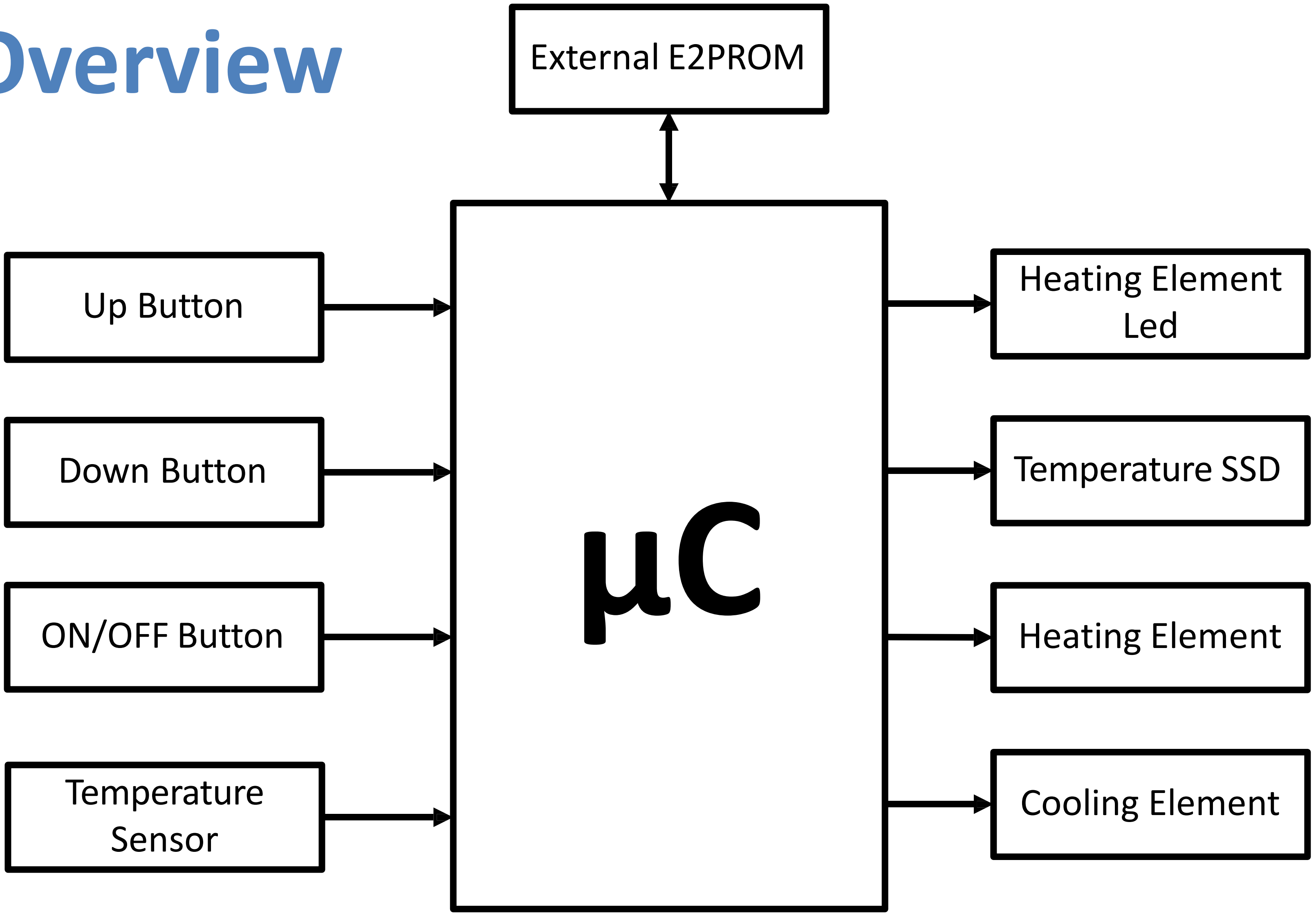
Case Study: Electric Water Heater

Abdelrhman Mosad Abdelhady
ECED up to level 4

Project - >

https://github.com/xMosad/swift_electric_water_heater

Overview


External E2PROM
Up Button
Down Button
ON/OFF Button
Temperature Sensor
µC
Heating Element Led
Temperature SSD
Heating Element
Cooling Element

Specifications – Temperature Setting

1. The “Up” or “Down” buttons are used to change the required water temperature (set temperature)
2. The first “Up” or “Down” button press, enters the temperature setting mode
3. After entering temperature setting mode, a single “Up” button press increase the set temperature by 5 degrees
4. After entering temperature setting mode, a single “Down” button press decrease the set temperature by 5 degrees
5. The minimum possible set temperature is 35 degrees
6. The maximum possible set temperature is 75 degrees
7. The “External E2PROM” should save the set temperature once set
8. If the electric water heater is turned OFF then ON, the stored set temperature should be retrieved from the “External E2PROM”
9. The initial set temperature is 60 degrees

Specifications – ON/OFF Behavior

1. If power is connected to the heater, the electric water heater is in OFF state
2. If the “ON/OFF” button is released and the electric water heater is in OFF state, the electric water heater goes to ON state
3. If the “ON/OFF” button is released and the electric water heater is in ON state, the electric water heater goes to OFF state
4. In the OFF state, all display should be turned OFF

Specifications – Temperature Sensing

1. The temperature sensor measures the water temperature
2. The water temperature should increase, if the “Heating Element” is ON
3. The water temperature should decrease, if the “Cooling Element” is ON
4. Temperature should be sensed once every 100 ms
5. The decision to turn ON or OFF either the “Heating Element” or the “Cooling Element” based on the average of the last 10 temperature readings

Specifications – Heating/Cooling Elements

1. The “Heating Element” should be turned ON, if the current water temperature is less than the set temperature by 5 degrees
2. The “Cooling Element” should be turned OFF, if the current water temperature is less than the set temperature by 5 degrees
3. The “Heating Element” should be turned OFF, if the current water temperature is greater than the set temperature by 5 degrees
4. The “Cooling Element” should be turned ON, if the current water temperature is greater than the set temperature by 5 degrees

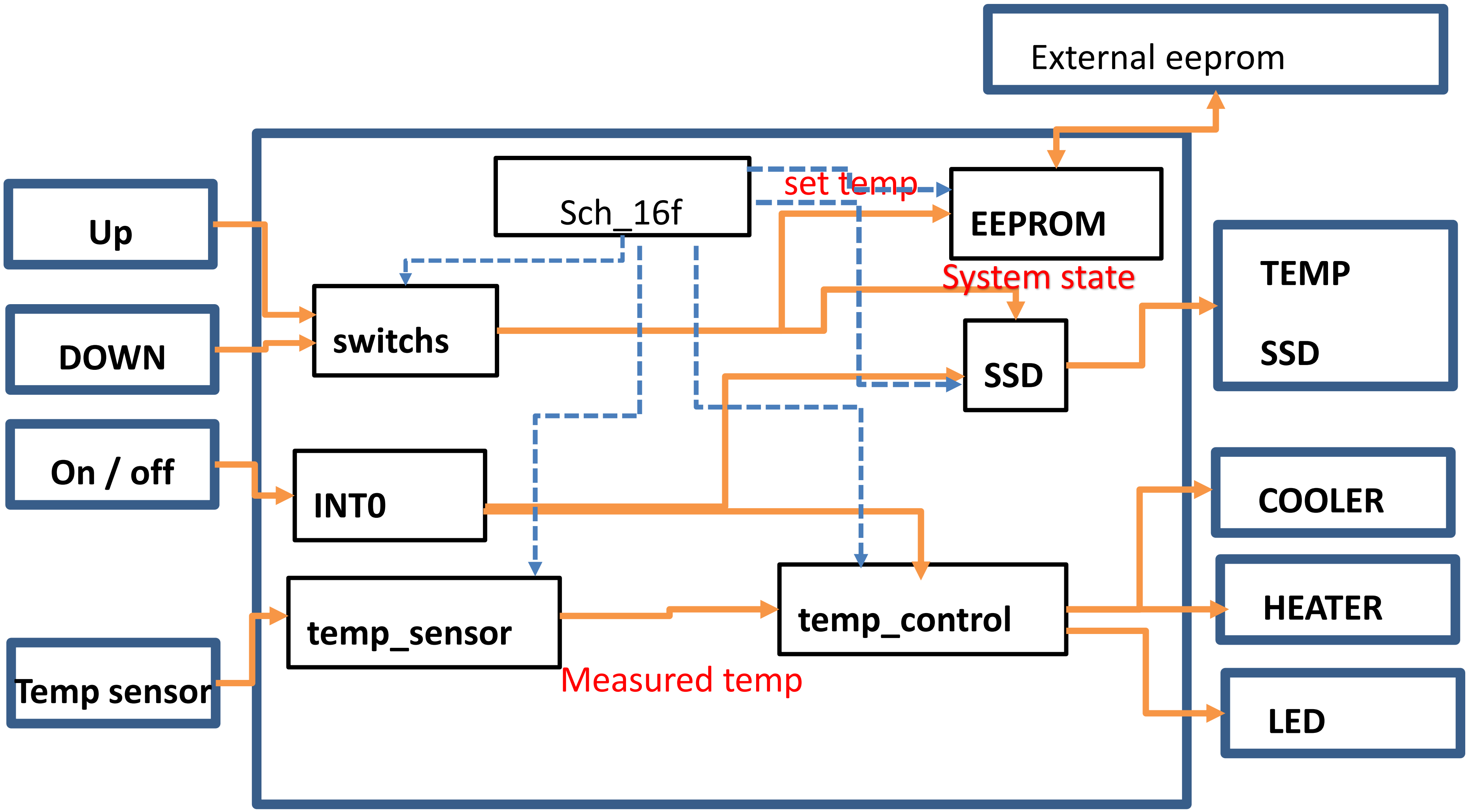
Specifications – Seven Segments

1. 2 seven segment by default show the current water temperature or the set temperature
2. By default, the 2 seven segment display are show the current water temperature
3. If the electric water heater is in the temperature setting mode, the 2 seven segment displays should blink every 1 second and show the set temperature
4. In the temperature setting mode, every change in the set temperature should be reflected on the 2 seven segment displays
5. The 2 seven segment display should exit the temperature setting mode, if the “UP” and “Down” buttons are not pressed for 5 seconds

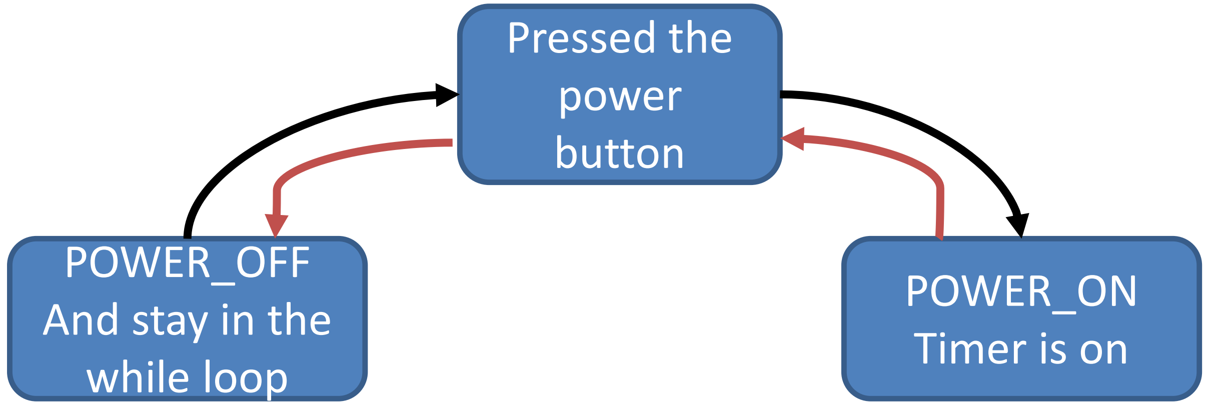
Specifications – Heating Element Led

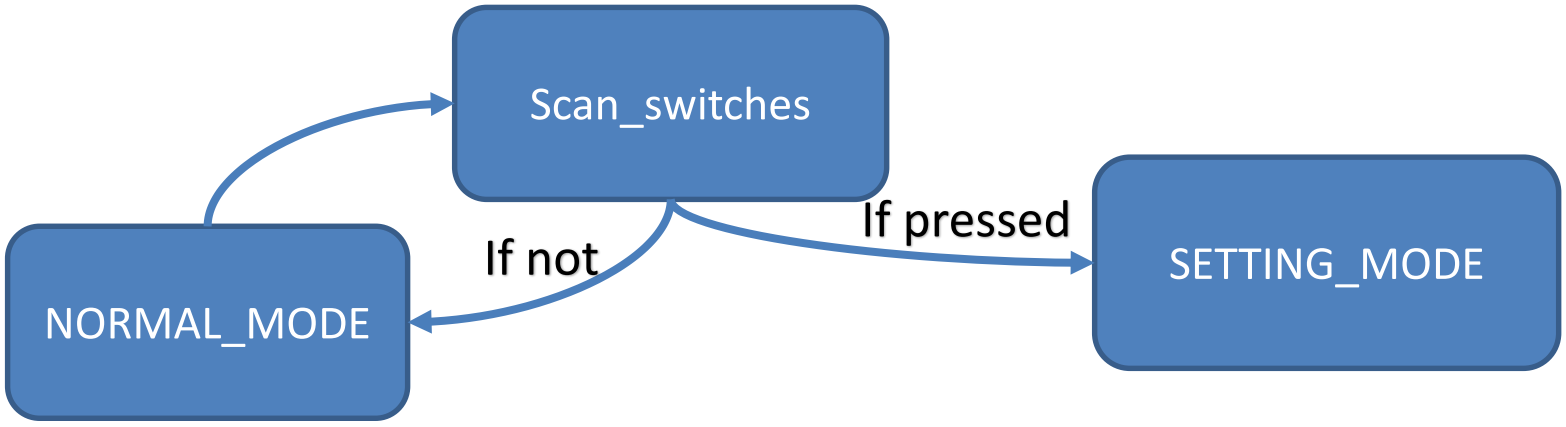
1. If the “Heating Element” is ON, the “Heating Element Led” should blink every 1 second
2. If the “Cooling Element” is OB, the “Heating Element Led” should be ON

Electric water heater : Static Architecture



State Machine





Detailed Design

- switchs
  - switch_init
  - switch_scan
  - sw_action
- Sch_16f
  - SCH_Init
  - SCH_Dispatch_Tasks
  - SCH_Add_Task
  - SCH_Start
  - SCH_Stop
- temp_control
  - temp_control_init
  - temp_set
  - Led
  - temp_control_off
- **SSD**
  - ssd_init
  - display7s
  - ssd_update
  - ssd_blink
  - ssd_turn_off
- temp_sensor
  - temp_sensor_init
  - temp_sensor_read
- eeprom
  - EEPROM_init
  - EEPROM_write
  - EEPROM_read
  - get_set_temp

Switchs module

Function	Type
void switch_init(void)	**Initialization** - initialize the pins direction - Enable external interrupt
void switch_scan(void)	**Periodic Task** - check every 20 ms if a button was pressed
void sw_action(void)	**Global function** - if a button is pressed will take the right action responding to it

temp_sensor module

Function	Type
void temp_sensor_init(void)	**Initialization** - initialize the pins direction - Initialize the adc
uint8_t average (void)	**Private function** - return the average of the last ten values
uint8_t temp_sensor_read (void)	**Periodic Task** - read the temp every 100 ms and take average of the last ten values

temp_control module

function	Type
void temp_control_init(void);	**Initialization** - initialize the pins direction for cooler , heater , led
void temp_set(uint8_t temp);	**Periodic Task** - run evry time the ISR fire (1 ms) to keep the temp at the set temp In the normal_mode state
void led(void);	**Periodic Task** - blink every 1 s if heater is on
void temp_control_off(void);	**Global function** - turn off cooler , heater , led

SSD module

Function	type
void ssd_init(void)	**Initialization** - initialize the pins direction
uint8_t display7s(uint8_t v)	**Private function** - return the right data to be written in the port register
void ssd_update(uint8_t temp)	**Periodic Task** - update one of the ssd every 50 ms
void ssd_blink(uint8_t e_temp)	**Periodic Task** - in the setting_mode state blink the ssd every 1 s
void ssd_turn_off(void)	**Global function** - turn off the SSD

EEPROM

Function	Type
void EEPROM_init(void)	**Initialization** - initialize the I2C to run eeprom
void EEPROM_write (uint16_t address , uint8_t _x)	**Global function** - write to a specific address in the EEPROM and is called every time exiting the setting_mode
uint8_t EEPROM_read(uint16_t address)	**Global function** -read from specific address called only at the start
void get_set_temp(void)	**Global function** - one shot task

MCAL
For pic16f877a

I2C DRIVER

Function	operation
void I2C1_Init(uint32_t freq);	Initialize the module with specific frequency
void I2C_Wait(void);	Wait till the right bits get cleared
void I2C1_Start(void);	Send start condition
void I2C1_Stop(void);	Send stop condition
void I2C1_Wr(uint8_t _data);	Write to the I2C bus
uint8_t I2C1_Rd(void);	Read from The I2C bus

ADC DRIVER

function	operation
void ADC_Init(void);	Initialize the adc module
uint16_t ADC_Read (uint8_t channel)	Get the adc value from specific channel

Sch_16f

function	type
void SCH_Init(void);	**Initialization** - initialize the scheduler for Pic16f877a using timer 1
void SCH_Dispatch_Tasks(void);	**Global function** - called in the while loop to execute every function at the right time
tByte SCH_Add_Task(void (*) (void), const tWord, const tWord);	**Global function** - add tasks to scheduler
tByte SCH_Delete_Task(const tByte);	**Global function** - delete tasks from scheduler
void SCH_Start(void);	**Global function** - start scheduler
void SCH_Stop(void);	**Global function** - stop scheduler
void SCH_Report_Status(void);	**Global function** -report errors

Dynamic Design

Using **Time Triggered** scheduler , with 1 ms period

Task	Task action	period
get_set_temp	Read temp from external EEPROM	One shot
temp_sensor_read	Read the temp and averaging it	100 ms
ssd_update	Update the SSD (one of the SSD)	50 ms
led	Blink the led if heater is on	1000 ms
temp_set	Keep the temp at set_temp	100 ms
switch_scan	Scan switches and take the right action based on it	20 ms
ssd_blink	Blink SSD if SETTING_MODE is on	1000 ms

Schedulability Check

🟧 switchs Task @ 20 ms

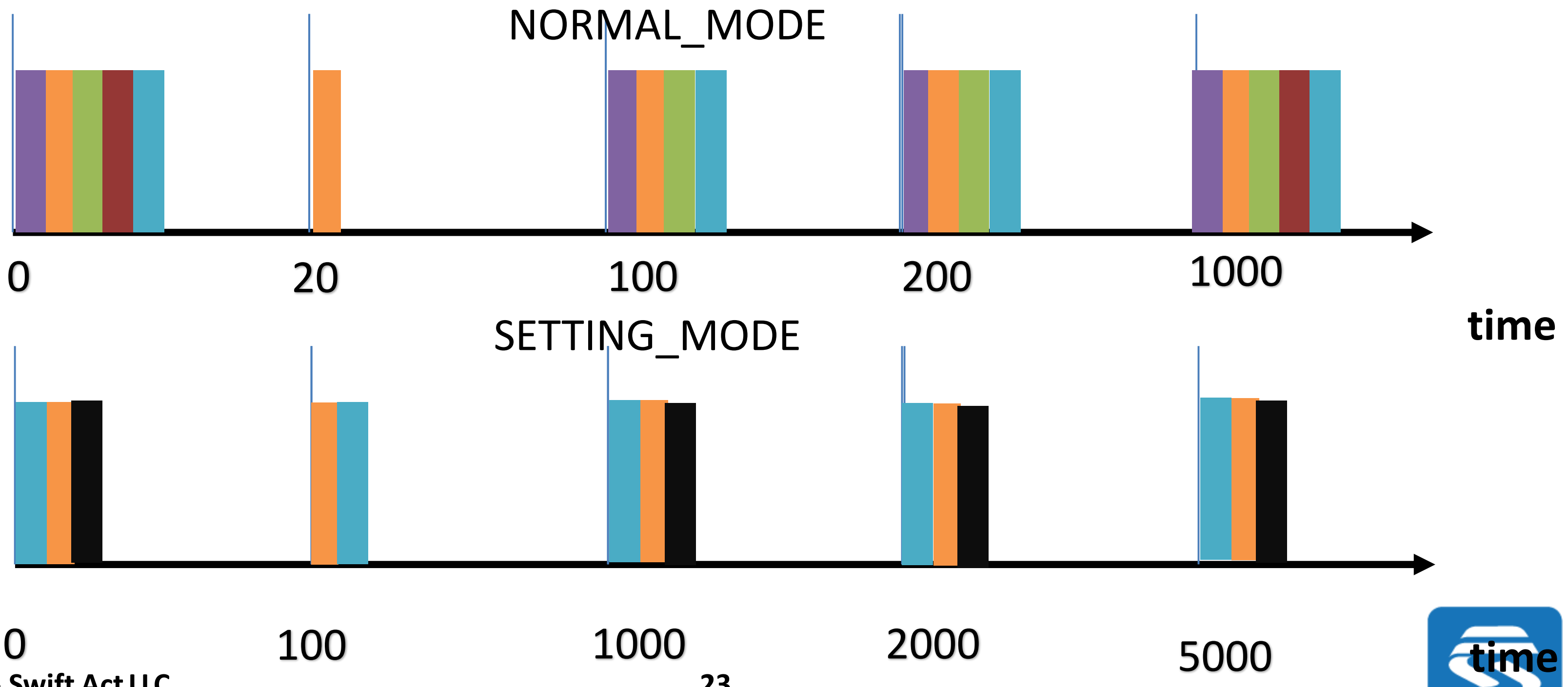
🟪 Sensor_read Task @ 100 ms

🟥 led Task @ 1000 ms If heater is on

🟦 SSD update Task @ 50 ms

🟩 Set_temp Task @ 100 ms

⬛ Ssd_blink@ 1000 ms If heater is on



thank you
gracias
merci
danke
obrigado
go raibh maith agat
teşekkür ederim
dziękuję
mochchakkeram
sukriya
kop khun krap
terima kasih
grazie
arigatō
dakujem
bedankt
спасибо
謝謝
감사합니다
ngiyabonga
tapadh leat
dank je
takk
мерси
ευχαριστώ