SPRINTS

SOS Design Document

Arafa Arafa Abd ElMawgod Ali

Contents

Project Introduction 2
High Level Design 3
Layered Architecture 3
Modules Each Layer 4
Low Level 5
Application Layer 5
SOS Layer 8
SOS APIs 8

List of Figures

Figure 1 Layered Architecture 3
Figure 2 Sequence Diagram 5
Figure 3 Class Diagram 6
Figure 4 sos_sequence diagram 7

List of Tables

Table 1 sos_init API 8
Table 2 sos_deinit API 9
Table 3 sos_create_task API 10
Table 4 sos_delete_task API 11
Table 5 sos_modify_task API 12
Table 6 sos_run API 13
Table 7 sos_disable API 14
Table 8 sos_enable API 14

Project Introduction

The SOS(Small operating System) is simulate the basic role of OS Scheduler to handle multiple task and synchronize them.

In this documentation will discuss the layered architecture of the system, Modules APIs and their flowchart and

High Level Design

Layered Architecture

APPLICATION LAYER: In this layer the application or system will be implemented here, this layer that will be between User events and other layers.

Utilities LAYER: that have standard types will use with all entire system and bit manipulation functions

OS LAYER: this layer will apply multi-tasking behavior to the system

ECU LAYER: this layer has all hardware driver outside **MCU**.

MCA LAYER: this layer has all peripheral drivers that are inside **MCU**.

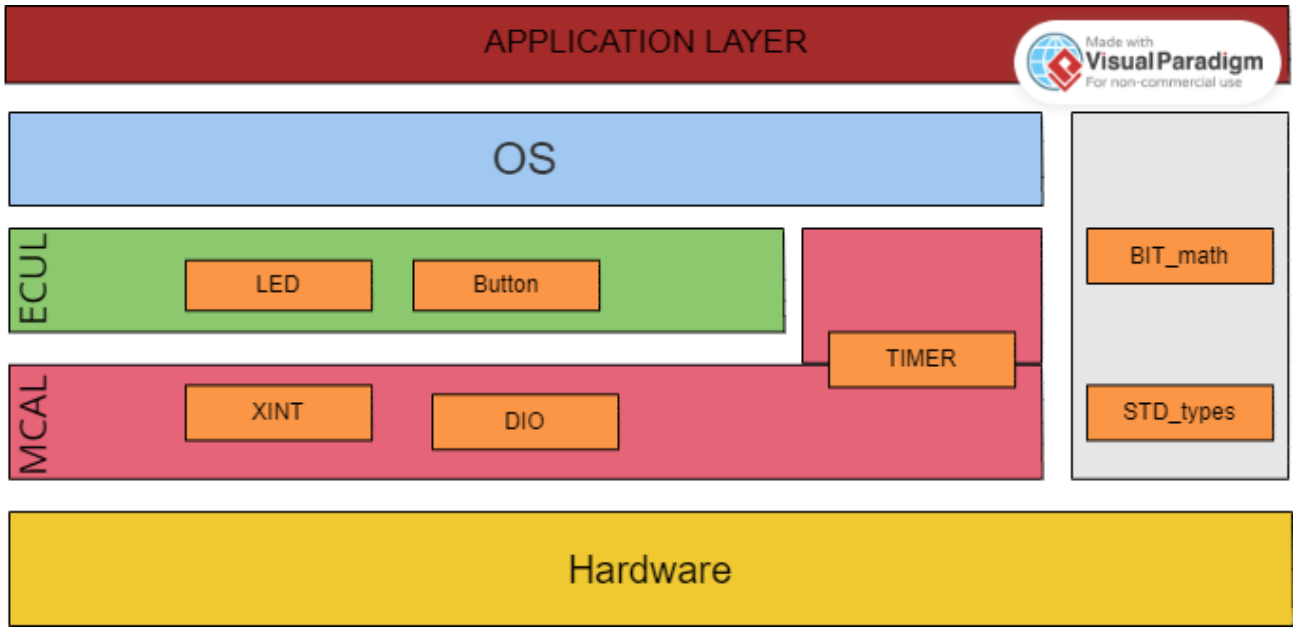

Figure 1 Layered Architecture

Modules Each Layer

1. APP Layer
   I.APPLICATION Module

2. OS Layer
   I.SOS Module is providing APIs to handle multiple tasks

3. Utilities Layer
   I.BIT_math Module provides bit manipulation.
   II.STD_types Module provides standard types

4. ECU Layer
   I.Button Module provides APIs to control all buttons.
   II.LED Module provides APIs to control LEDs.

5. MCA LAYER
   I.Timer Module provides APIs to all timers in MCU.
   II.DIO Module provides APIs to use GPIO and handle External interrupts.

Low Level

Application Layer

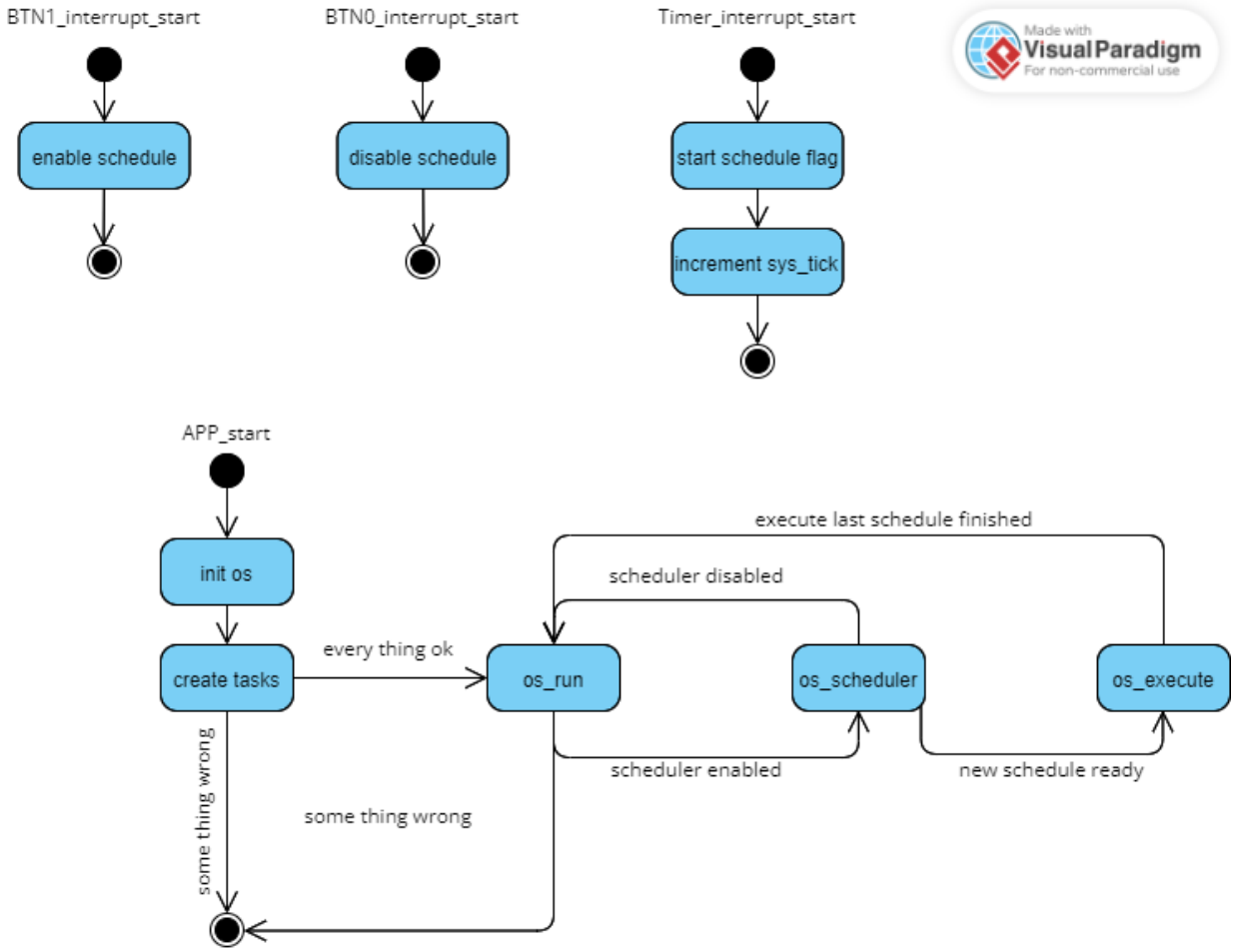

Figure 2 Sequence Diagram

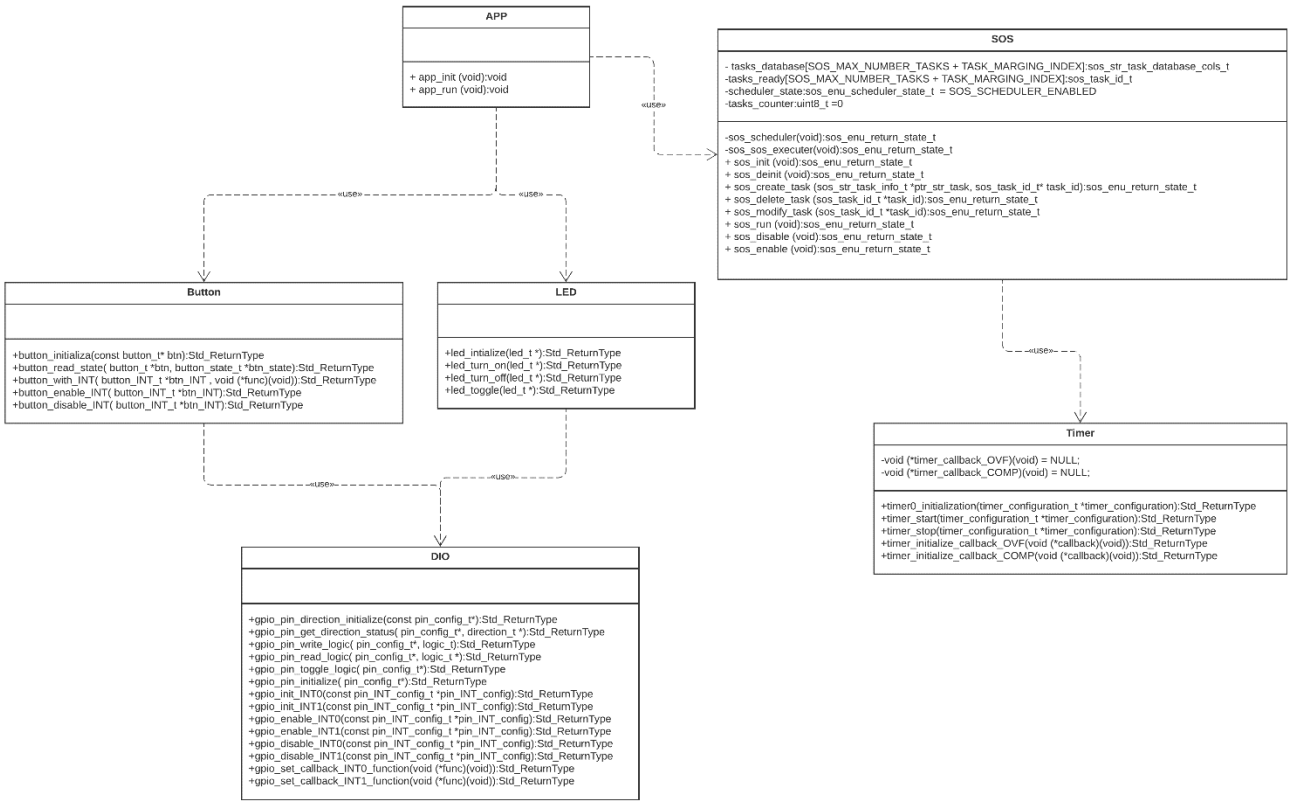

Figure 3 Class Diagram

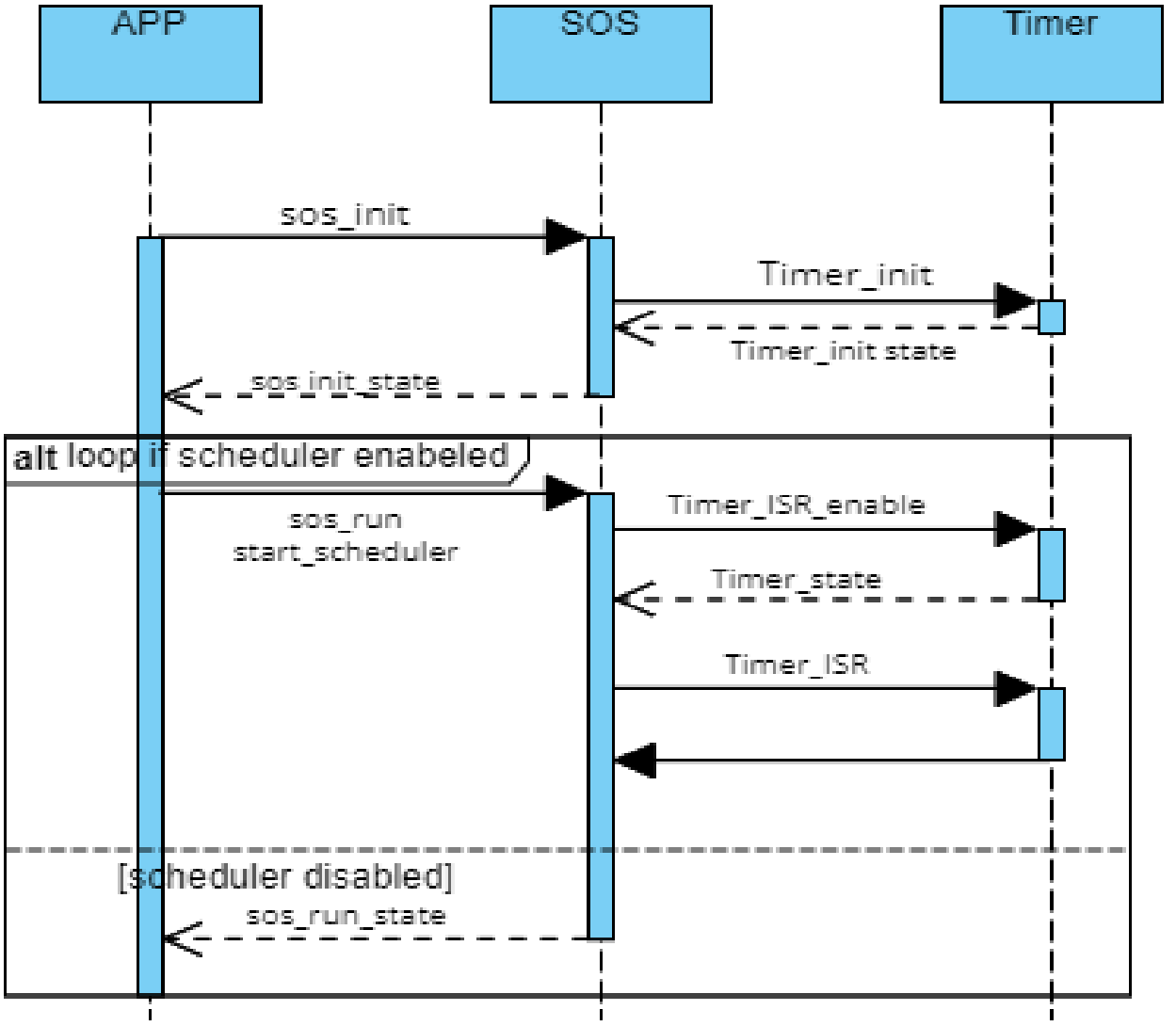

Figure 4 sos_sequence diagram

SOS Layer

SOS APIs

Table 1 sos_init API

API name	**sos_init**
Description	This API initialize the SOS
Syntax	`sos_enu_return_state_t sos_init (void)`
Sync/Async	Sync
Reentrancy	Reentrant
Parameters (in)	None
Parameters (out)	None
Parameters (in&out)	None
Return	`SOS_E_NOT_OK:` there is something wrong but couldn't defined it `SOS_EXCEED_NUMBER_OF_TASKS:` the max number of tasks exceeded

Table 2 sos_deinit API

API name	**sos_deinit**
Description	This API deinitialize the SOS
Syntax	`sos_enu_return_state_t sos_deinit (void)`
Sync/Async	Sync
Reentrancy	Reentrant
Parameters (in)	None
Parameters (out)	None
Parameters (in&out)	None
Return	`SOS_E_NOT_OK:` there is something wrong but couldn't defined it

Table 3 sos_create_task API

API name	**sos_create_task**
Description	This API to create task and add it to sos database
Syntax	`sos_enu_return_state_t sos_create_task(sos_str_task_info_t *ptr_str_task, sos_task_id_t* task_id)`
Sync/Async	Sync
Reentrancy	Reentrant
Parameters (in)	*`ptr_str_task:` address of struct has all task info
Parameters (out)	* `task_id:` address of varible to return id of created task
Parameters (in&out)	None
Return	`SOS_E_NOT_OK:` there is something wrong but couldn't defined it `SOS_NULL_PTR:` null pointer used `SOS_PRIORITY_NOT_ALLOW:` priority not allow to use `SOS_TIME_LIMIT_NOT:` periodicity isn't allowed `SOS_EXCEED_NUMBER_OF_TASKS:` the max number of tasks exceeded

Table 4 sos_delete_task API

API name	**sos_delete_task**
Description	This API to create task from sos database
Syntax	`sos_enu_return_state_t sos_delete_task(sos_task_id_t *task_id)`
Sync/Async	Sync
Reentrancy	Reentrant
Parameters (in)	* `task_id:` address of varible to return id of created task
Parameters (out)	None
Parameters (in&out)	None
Return	`SOS_E_NOT_OK:` there is something wrong but couldn't defined it `SOS_NULL_PTR:` null pointer used `SOS_TASK_NOT_FOUND:` there isn't task in database by provided task_id

Table 5 sos_modify_task API

API name	**sos_modify_task**
Description	This API to modify task is sos database
Syntax	`sos_enu_return_state_t sos_modify_task(sos_task_id_t *task_id, sos_str_task_info_t *ptr_str_task);`
Sync/Async	Sync
Reentrancy	Reentrant
Parameters (in)	* `task_id:` address of varible to return id of created task *`ptr_str_task:` address of struct has all task info
Parameters (out)	None
Parameters (in&out)	None
Return	`SOS_E_NOT_OK:` there is something wrong but couldn't defined it `SOS_NULL_PTR:` null pointer used `SOS_TASK_NOT_FOUND:` there isn't task in database by provided task_id `SOS_PRIORITY_NOT_ALLOW:` priority not allow to use `SOS_TIME_LIMIT_NOT:` periodicity isn't allowed `SOS_EXCEED_NUMBER_OF_TASKS:` the max number of tasks exceeded

Table 6 sos_run API

API name	**sos_run**
Description	This API to run sos scheduler and executer
Syntax	`void sos_run(void)`
Sync/Async	Sync
Reentrancy	Reentrant
Parameters (in)	None
Parameters (out)	None
Parameters (in&out)	None
Return	None

Table 7 sos_disable API

API name	**sos_disable**
Description	This API to disable scheduler
Syntax	`void sos_disable(void)`
Sync/Async	Sync
Reentrancy	Reentrant
Parameters (in)	None
Parameters (out)	None
Parameters (in&out)	None
Return	None

Table 8 sos_enable API

API name	**sos_enable**
Description	This API to enable scheduler
Syntax	`void sos_enable(void)`
Sync/Async	Sync
Reentrancy	Reentrant
Parameters (in)	None
Parameters (out)	None
Parameters (in&out)	None
Return	None