



LIN\_EA  
DSD\_LIN\_EA

Division  
I B&S

## **LIN Driver**

**Title: SW Component LIN\_EA v1.5**



History				
Issue status (Index)	Maturity/Date (draft/invalid/valid) (dd-mm-yyyy)	Author Department	Check/Release Department	Description
1.0	Draft 27-Oct-15	Edgar Escayola	Adrián Zacarías	Creation of the document
1.1	Draft 28-Oct-15	Edgar Escayola	Adrián Zacarías	Changes at: Purpose, References, and realization constraints and targets.
1.2	Draft 29-Oct-15	Adrián Zacarías	Edgar Escayola	Addition of abbreviations and definitions, diagrams and descriptions.
1.3	Draft 30-Oct-15	Adrián Zacarías	Edgar Escayola	Class diagram and functions definitions.
1.4	Draft 30-Oct-15	Edgar Escayola	Adrián Zacarías	Implementation of traceability in the document.
1.5	Release 1-Oct-15	Edgar Escayola	Adrián Zacarías	Changes from the review implemented.

## Table of Contents

<b>1</b>	<b>PURPOSE .....</b>	<b>5</b>
<b>2</b>	<b>DEFINITIONS AND ABBREVIATIONS .....</b>	<b>5</b>
2.1	Definitions .....	5
2.2	Type name definition .....	5
<b>3</b>	<b>REALIZATION CONSTRAINTS AND TARGETS .....</b>	<b>6</b>
3.1	TRK-MPC5606B's features .....	6
<b>4</b>	<b>SW CONCEPTUAL DESIGN.....</b>	<b>7</b>
4.1	Architecture design .....	7
4.2	Use case diagram.....	8
4.3	Deployment Diagram.....	9
4.4	Sequence diagram .....	9
<b>5</b>	<b>SW COMPONENT INTERNAL BREAKDOWN.....</b>	<b>10</b>
5.1	Functional Decomposition .....	10
5.2	<b>LIN .....</b>	<b>11</b>
5.2.1	Function void TX_ISR (void).....	11
5.2.2	Function void RX_ISR (void) .....	13
5.2.3	Function void Init_LIN (void) .....	14
5.2.4	Function void Error_handler (void).....	14
5.3	<b>LIN_EA.....</b>	<b>15</b>
5.3.1	Function void LED_status (void) .....	15
5.3.2	Function void Set_SlaveState (T_UBYTE).....	16
5.4	<b>TASKS .....</b>	<b>17</b>
5.4.1	Function void Task0_SlaveState (void).....	17
5.4.2	Function void Task1_LED (void).....	17
5.5	<b>Init_Tasks .....</b>	<b>17</b>
5.5.1	Function void Global_Init (void).....	17
5.6	<b>TIMER.....</b>	<b>18</b>
5.6.1	Function void InitPIT (void).....	18
5.6.2	Function void InitPITChannel (T_UBYTE).....	18
5.7	<b>LED .....</b>	<b>19</b>
5.7.1	Function void Init_LED (void).....	19



5.7.2	Function T_UBYTE Get_LED_Status (void) .....	19
5.7.3	Function void Set_LED_Status (T_UBYTE) .....	19
<b>5.8</b>	<b>GPIO .....</b>	<b>19</b>
5.8.1	Function void Set_Pin_State (T_UBYTE, T_UBYTE) .....	19
5.8.2	Function void Set_Pin_Mode (T_UBYTE, T_UBYTE) .....	20
5.8.3	Function T_UBYTE Get_Pin_State (T_UBYTE) .....	20
<b>5.9</b>	<b>KERNEL.....</b>	<b>20</b>
5.9.1	Function void Tick_ISR (void) .....	20
5.9.2	Function void InitModesAndClock (void) .....	20
5.9.3	Function void main_Scheduler (void) .....	21

## 1 Purpose

This document has been created to describe the design specifications of the application LIN Driver. It consists on the implementation of the LIN protocol in the TRK-MPC5606B development board and an application, which will be a slave number 1 in a group of one master and four slaves.

Req. Id. 1.0, 1.1, 1.2

## 2 Definitions and abbreviations

### 2.1 Definitions

<b>Cmd_NONE</b>	Command to do nothing.
<b>Cmd_LED_on</b>	Command to turn the LED on.
<b>Cmd_LED_off</b>	Command to turn the LED off.
<b>Cmd_LED_toggling</b>	Command to trigger the blinking LED of the sequence.
<b>Cmd_disable_slv</b>	Command to disable the slave mode in the slave node.
<b>Data</b>	The response of a frame carries one to eight data bytes, collectively called data.
<b>Dominant</b>	A zero value in the LIN bus.
<b>Frame</b>	A frame consists of a header and a response. The reply frame for a node configuration or a diagnostic request is a response.
<b>Header</b>	A header is the first part of a frame; it is always sent by the master task.
<b>ISR</b>	Interrupt Service Routine
<b>LED</b>	Light Emitting Diode
<b>LIN</b>	Local interconnect Network
<b>LIN_EA</b>	Name of the system designed in this project.
<b>Node</b>	A node is an ECU (electronic control unit). However, a single ECU may be connected to multiple LIN clusters.
<b>PIT</b>	Periodic Interrupt Timer
<b>Recessive</b>	A one value in the LIN bus.
<b>Rx</b>	Reception.
<b>Slave node</b>	A node that contains a slave task only, i.e. it does not contain a master task.
<b>TOGGLING</b>	Status where the LED blinks.
<b>Tx</b>	Transmission.

Req. Id. 1.13, 1.14, 1.15, 1.17, 1.18

### 2.2 Type name definition

Type Name	Elements					
	0	1	2	3	4	5
<b>t_cmdType</b>	cmd_NONE	cmd_LED_on	cmd_LED_off	cmd_LED_toggling	cmd_disable_slv	cmd_enable_slv
<b>t_LEDstat</b>	OFF	ON	TOGGLING			
<b>t_boolean</b>	FALSE	TRUE				
<b>array</b>	AZSEEV					
<b>Scalar</b>	1					

Req. Id. 1.12, 1.16, 1.19, 1.20, 1.21



## References

N°	Document name	Reference	Revision
1	LIN Specification Package	LIN/Documents/LIN-Spec_2-2A.pdf	2.2A
2	Traceability Matrix – LIN_EA	LIN/Documents/1.0 Requirements/Traceability Matrix – LIN_EA.xls	1.7
3	MPC5607B Microcontroller Reference Manual	LIN/Documents/MPC5607BRM_Reference_Manual.pdf	7.2
4	Quick Start Guide TRK-MPC5606B	LIN/Documents/Quick_Start_Guide.pdf	3
5	LIN Network Definition	LIN/Documents/LIN_Network_Database.xls	1.0

## 3 Realization constraints and targets

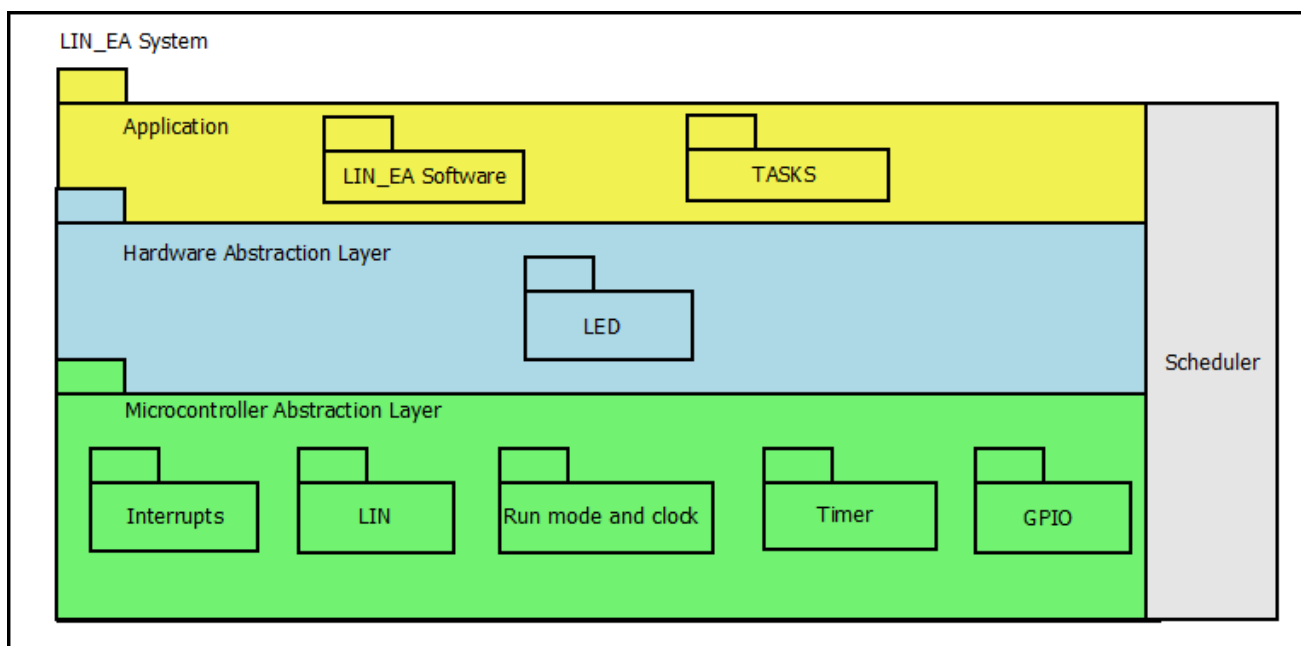
### 3.1 TRK-MPC5606B's features

- MPC5606B MCU (144-pin LQFP).
- On-board JTAG connection via open source OSBDM circuit using the MPC9S08JM MCU.
- MCZ3390S5EK system basis chip with advanced power management and integrated CAN transceiver and LIN 2.0 interface.
- CAN interface.
- LIN interface.
- Analog interface with potentiometer.
- High-efficiency green LEDs.
- 4 PushButtons.
- Serial communication interface.
- External power 9V DC to 12V DC regulated down to 5V DC.

Req. Id. 3.3

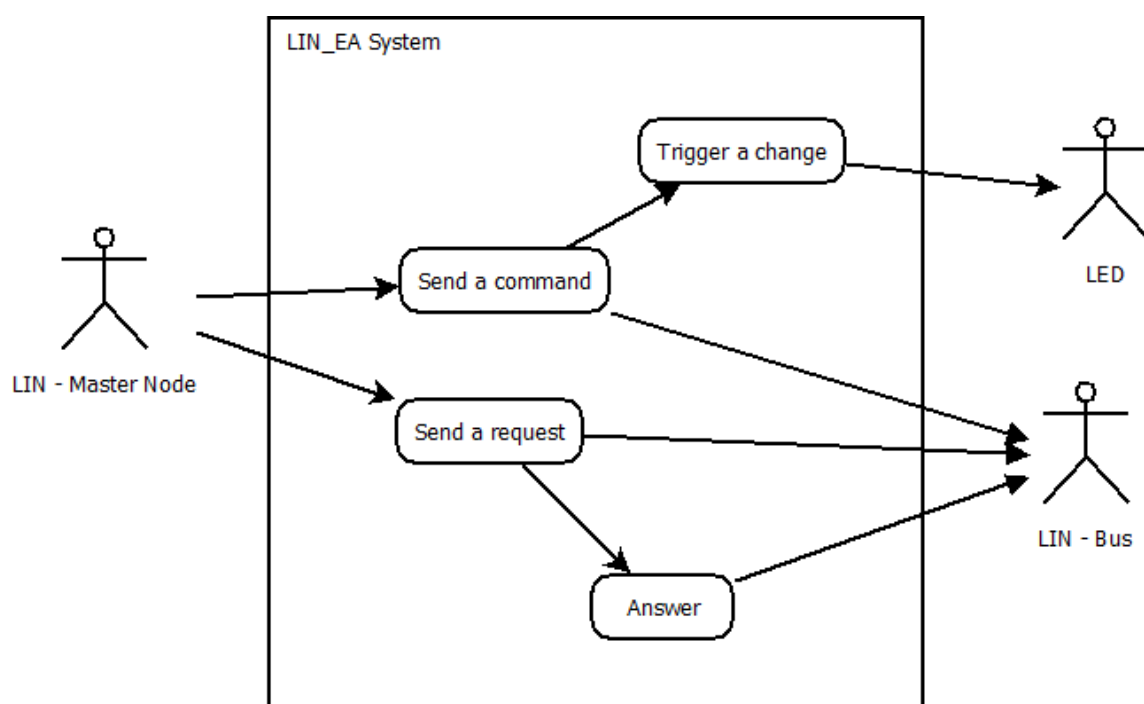
## 4 SW Conceptual design

### 4.1 Architecture design



The architecture diagram shows the layers in which the system is divided. In the Microcontroller Abstraction Layer, there are the modules where the interrupts, timers, GPIO, LIN interface and general configurations are handled. In the Hardware Abstraction Layer, the system manages the external modules as LEDs. In the Application layer, there is the main software and the tasks from the scheduler. Parallel to every layer there is the scheduler, which communicates with each of the layers.

## 4.2 Use case diagram



### Users

**LIN – Master Node**

### Description

The master node sends data through the LIN – Bus in order to affect the system.

**LIN – Bus**

The LIN – Bus receives every data sent through the LIN protocol.

**LED**

There is a LED which is controlled by the system.

### Case

**Send a command**

### Description

This case triggers a transmission by the master in order to communicate an order to be executed.

**Trigger a change**

This case evaluates the command sent.

**Send a request**

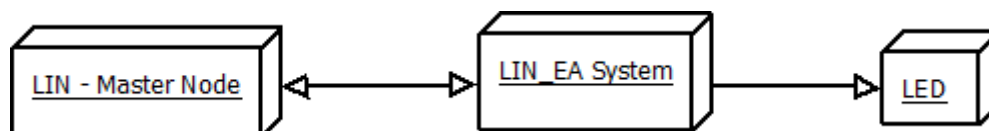
This case trigger a transmission by the master to communicate a request to receive status information.

**Answer**

This case triggers the transmission to the LIN – Bus of the solicited information.

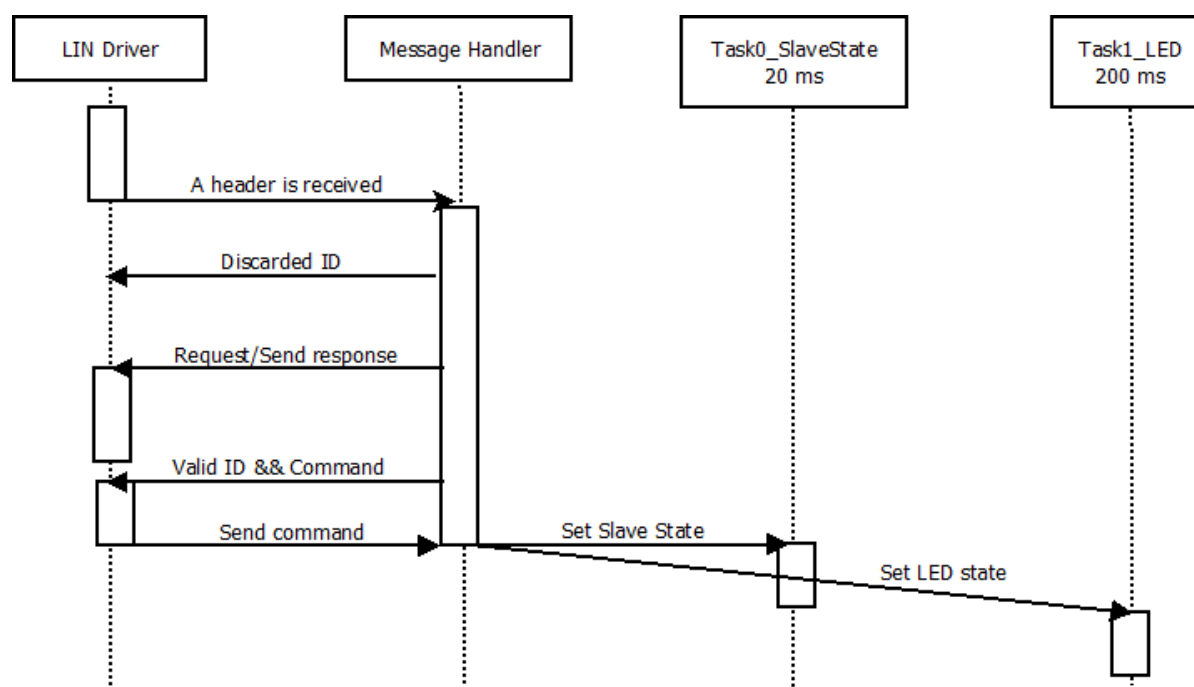


### 4.3 Deployment Diagram



The deployment diagram shows the interaction of the LIN\_EA system with external systems. There is a bidirectional communication with a LIN master node and an output to a LED. The master node sends commands, which can affect the system and LED state.

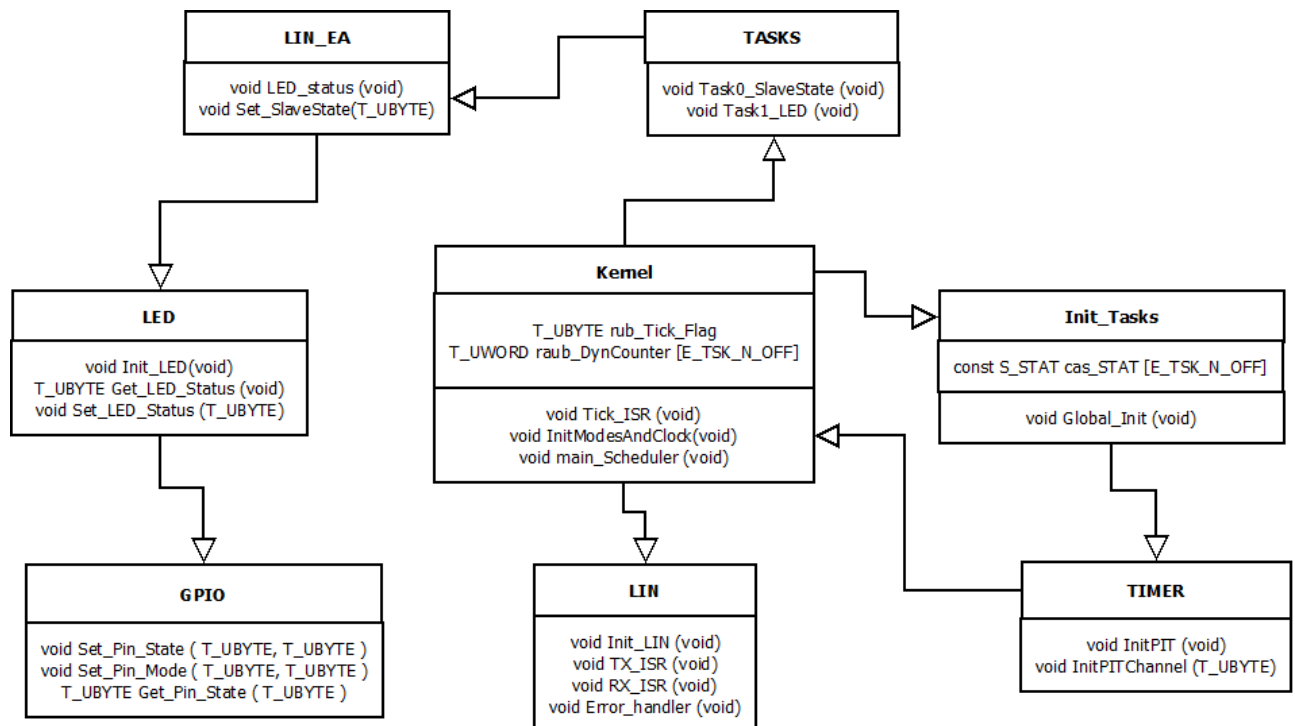
### 4.4 Sequence diagram



This diagram describes the interaction between different modules of the software. The LIN driver is in charge of receiving the messages that arrive to the system through the LIN protocol. It sends them to the message handler so it can decide whether it is an invalid ID, there should be a response from the system or it should receive a command from the master. In case that it receives a command from the master, the message handler delivers the tasks.

## 5 SW Component internal breakdown

### 5.1 Functional Decomposition



File	Description
<b>LIN_EA</b>	It is in charge of executing the main application of the program. It contains two functions, which also contain the state machines to control the LED status and the slave state.
<b>Kernel</b>	The scheduler is being executed here. It handles the main configurations and the tick interrupt.
<b>TASKS</b>	This module contains the periodic tasks that are executed by the scheduler.
<b>Init_Tasks</b>	This module contains the global initializations which are needed for the correct execution of the scheduler.
<b>LIN</b>	This module contains the LIN driver. It consists of an initialization of the LIN controller, an error handler, and an interrupt for transmission and reception.
<b>TIMER</b>	This file contains the configurations that must be done to achieve the periodic interrupt that gives the Ticks to the scheduler.
<b>GPIO</b>	This module handles the registers needed to configure ports and change state of pins.
<b>LED</b>	This module configures and handles the state of the LED.

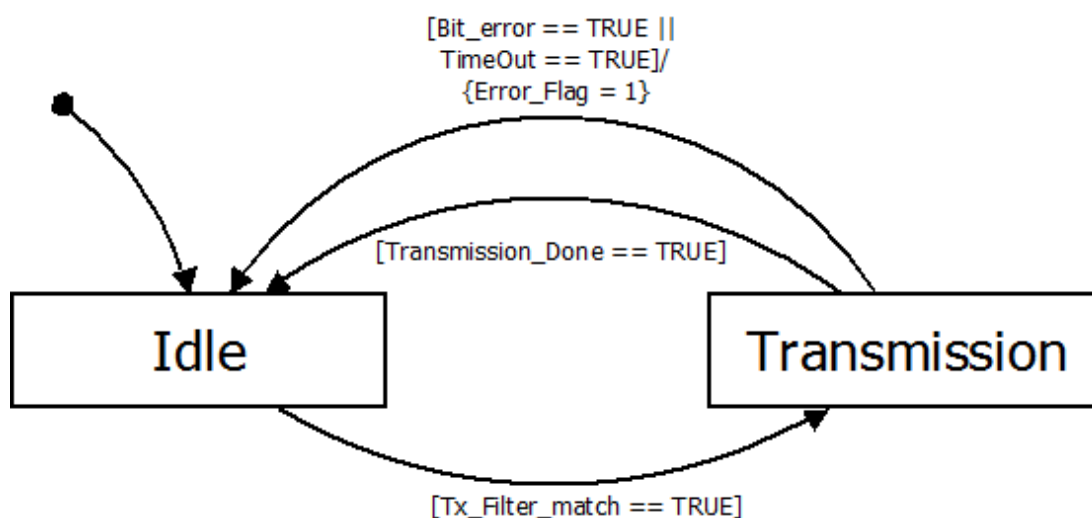
## Function Description and Dynamic Behavior

### 5.2 LIN

#### 5.2.1 Function void TX\_ISR (void)

<b>Description</b>	This function is called every time a transmission interrupt is generated from the LINFlex controller. It implements a state machine which controls the response's transmission after a valid command is received.
<b>Parameter 1</b>	Void
<b>Return Value</b>	Void
<b>Precondition</b>	Basic configuration of LINFlex must be done and the interrupts must be initialize in order to run this function.
<b>Post condition</b>	Does not apply
<b>Error Conditions</b>	Does not apply

### Dynamic Behavior State Chart



State	Description
<b>Idle</b>	State that waits for a valid command to send information using the LIN protocol. Once the id has been received the signal is considered transmitted by the master and it is ready to be readable to the filter. Req. Id. 2.3
<b>Transmission</b>	In this state, the corresponding data for the Filter Match Index is sent using the LIN protocol.



The following messages ID are considered valid. They should trigger a change of state to transmission:

Message name	Msg ID	Msg Data Length (byte)	Message publisher	Message subscriber	Signal length (bits)	Signal type	Signal Description
SLAVE1_RSP	0x20	2	Slave 1	Master	2	T_LEDstat	Return LED status
					1	T_boolean	Return node status
SLAVE1_ID	0xF0	7	Slave 1	Master	8	Scalar	Return team number
					48	Array	Return initials of members team

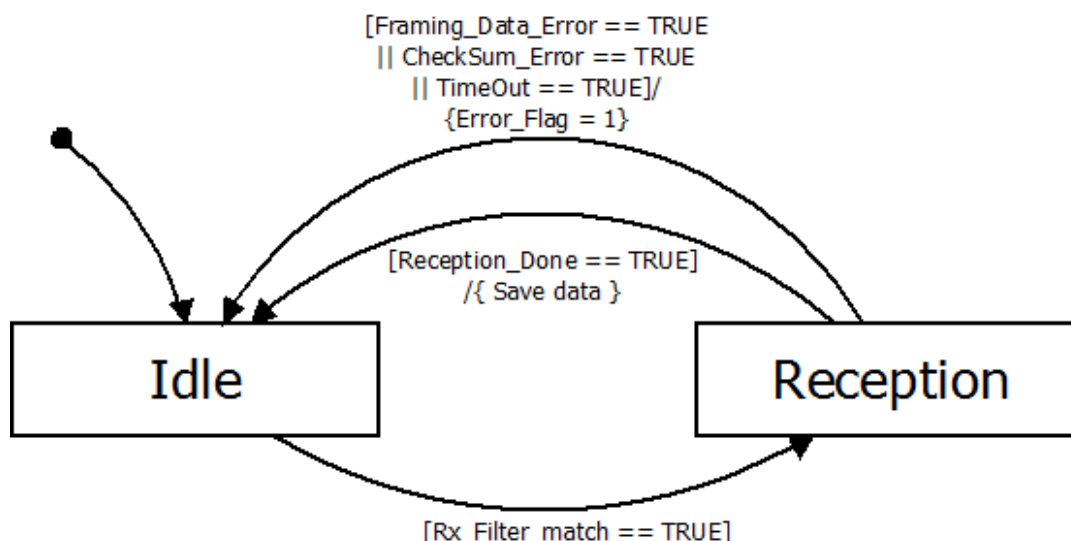
Req. Id. 1.3, 1.8, 1.9, 1.10, 1.11, 2.0, 2.1.

## 5.2.2 Function void RX\_ISR (void)

<b>Description</b>	This function is called every time a reception interrupt is generated from the LINFlex controller. It has a state machine, which controls the response's reception after a valid command is received.
<b>Parameter 1</b>	Void
<b>Return Value</b>	Void
<b>Precondition</b>	Basic configuration of LINFlex must be done and the interrupts must be initialize in order to run this function.
<b>Post condition</b>	Does not apply
<b>Error Conditions</b>	Does not apply

### Dynamic Behavior

#### State Chart



State	Description
<b>Idle</b>	State that waits for a valid command to send information using the LIN protocol. Once the break/sync field sequence and the id arrives, the filter compares the value of the id with the ones configured. If the id is found in the registers, an interrupt is triggered. Req. Id. 2.4, 2.5, 2.8
<b>Transmission</b>	In this state, a response of a communication using the LIN protocol is received and the enhance checksum is calculated and compared to the received one. Then, the signal is considered received and available. Req. Id. 2.2, 2.6, 2.9, 2.10

The following messages ID are considered valid. They should trigger a change of state to reception:

Message name	Msg ID	Msg Data Length (byte)	Message publisher	Message subscriber	Signal length (bits)	Signal type	Signal Description
MASTER_CMD_ALL	0xCF	1	Master	Slave 1, 2, 3, 4.	4	T_cmdType	Command for all nodes
MASTER_CMD_SLV1	0x50	1	Master	Slave 1	4	T_cmdType	Command for node 1.

Req. Id. 1.3, 1.4, 1.5, 1.6, 1.7, 2.0, 2.1.

### 5.2.3 Function void Init\_LIN (void)

<b>Description</b>	This function initializes the LIN controller of the board.
<b>Parameter 1</b>	Void
<b>Return Value</b>	Void
<b>Precondition</b>	Does not apply
<b>Post condition</b>	Does not apply
<b>Error Conditions</b>	Does not apply

The LIN controller must be set to initialization mode in order to set its configurations. The break detection threshold is set to 11 dominant local slave bit times. Another configuration is to set the endianness to Little Endian. Then the filters must be set to the following values: 0xCF, 0x50, 0x20, and 0xF0. Before leaving the function, the software sets the LIN controller to normal mode.

Req. Id. 1.3, 1.30, 2.0, 2.1, 2.7.

### 5.2.4 Function void Error\_handler (void)

<b>Description</b>	This function handles the flags of error coming from the LIN controller.
<b>Parameter 1</b>	Void
<b>Return Value</b>	Void
<b>Precondition</b>	Does not apply
<b>Post condition</b>	Does not apply
<b>Error Conditions</b>	Does not apply

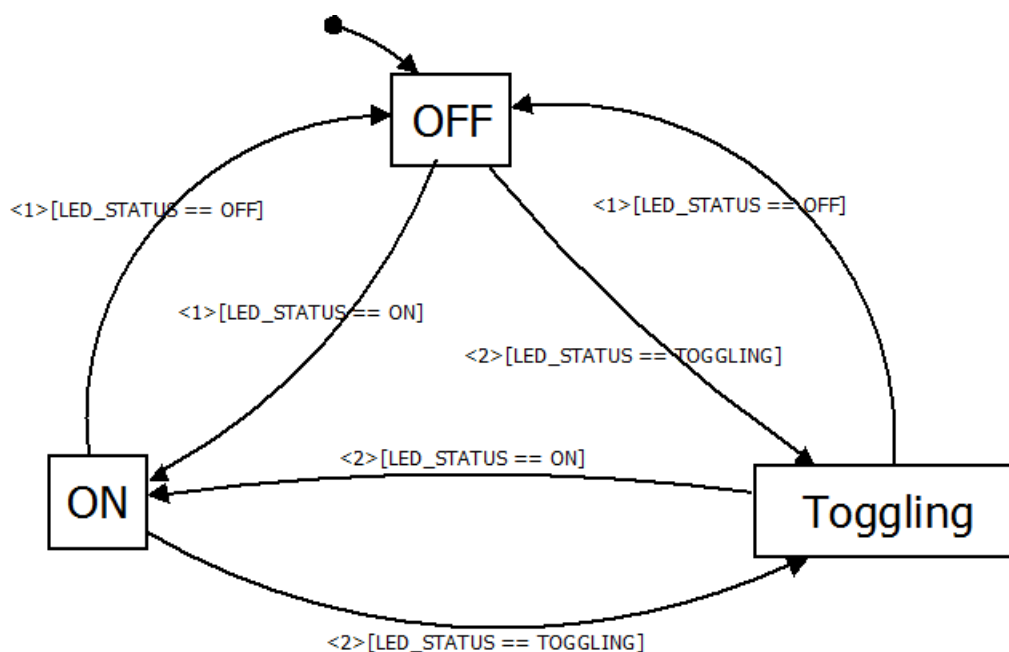
## 5.3 LIN\_EA

### 5.3.1 Function void LED\_status (void)

<b>Description</b>	This function is called every time the task 1 is executed. It has a state machine, which controls the response's reception after a valid command is received.
<b>Parameter 1</b>	Void
<b>Return Value</b>	Void
<b>Precondition</b>	Basic configuration of LINFlex must be done and the interrupts must be initialize in order to run this function.
<b>Post condition</b>	Does not apply
<b>Error Conditions</b>	Does not apply

Req. Id. 3.0, 3.1

**Dynamic Behavior**  
State Chart



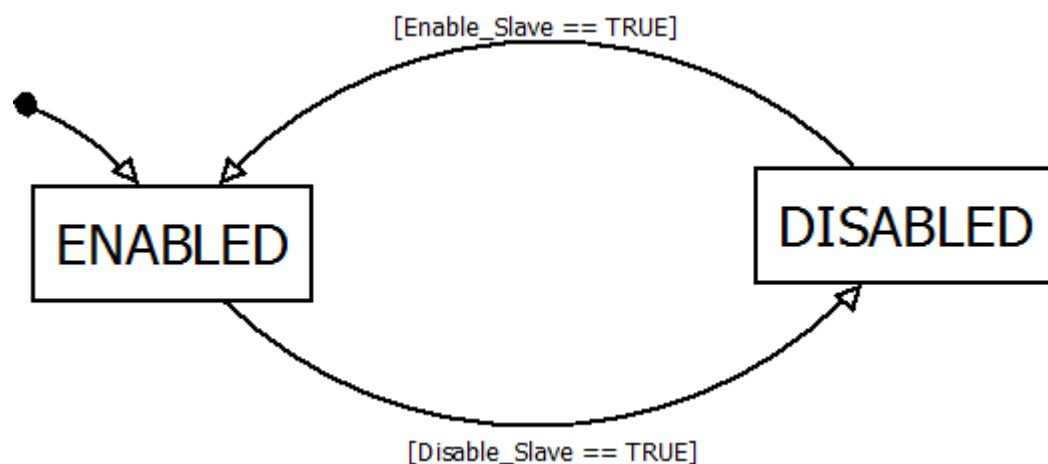
State	Description
<b>OFF</b>	State in which the LED is OFF. Req. Id. 1.24
<b>ON</b>	State in which the LED is ON. Req. Id. 1.23
<b>Toggling</b>	State in which the LED toggles. Req. id. 1.25

### 5.3.2 Function void Set\_SlaveState (T\_UBYTE)

<b>Description</b>	This function is executed every 20 milliseconds in the task 0. It has a state machine, which controls the response's reception after a valid command is received.
<b>Parameter 1</b>	T_UBYTE. Flag which might trigger a change in the internal state machine.
<b>Return Value</b>	Void
<b>Precondition</b>	Does not apply
<b>Post condition</b>	Does not apply
<b>Error Conditions</b>	Does not apply

Req. Id. 3.2

**Dynamic Behavior**  
State Chart



State	Description
<b>Enabled</b>	State in which the slave node acts fully in slave mode. The state transition is triggered by the command cmd_disable_slv. The command cmd_none shall not trigger any change. Req. Id. 1.26, 1.28
<b>Disabled</b>	State in which the system will not accept any other command than the command cmd_enable_slv. The command cmd_none shall not trigger any change. Req. Id. 1.29

Req. Id. 1.22



## 5.4 TASKS

### 5.4.1 Function void Task0\_SlaveState (void)

<b>Description</b>	This task is executed every 20 milliseconds. Its main function is to call the state machine that handles the slave status of the system.
<b>Parameter 1</b>	Void
<b>Return Value</b>	Void
<b>Precondition</b>	Does not apply
<b>Post condition</b>	Does not apply
<b>Error Conditions</b>	Does not apply

### 5.4.2 Function void Task1\_LED (void)

<b>Description</b>	This task is executed every 200 milliseconds. Its main function is to call the state machine that handles the status of the LED.
<b>Parameter 1</b>	Void
<b>Return Value</b>	Void
<b>Precondition</b>	Does not apply
<b>Post condition</b>	Does not apply
<b>Error Conditions</b>	Does not apply

## 5.5 Init\_Tasks

### 5.5.1 Function void Global\_Init (void)

<b>Description</b>	This function initializes the scheduler, and all the other peripherals needed to run the whole software.
<b>Parameter 1</b>	Void
<b>Return Value</b>	Void
<b>Precondition</b>	This function should be called first in the main program.
<b>Post condition</b>	The scheduler's functionalities can be used.
<b>Error Conditions</b>	Does not apply.

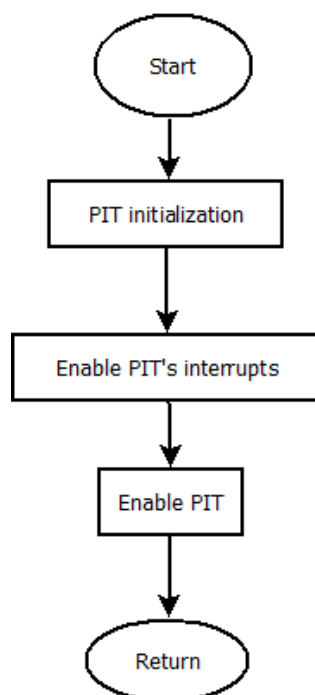
## 5.6 TIMER

### 5.6.1 Function void InitPIT (void)

<b>Description</b>	PIT is initialized.
<b>Parameter 1</b>	Void
<b>Return Value</b>	Void
<b>Precondition</b>	This function is called in the beginning of the main program to initialize the PIT.
<b>Post condition</b>	The interrupts every 10ms will be generated.
<b>Error Conditions</b>	Does not apply

#### Dynamic Behavior

Activity diagram



### 5.6.2 Function void InitPITChannel (T\_UBYTE)

<b>Description</b>	This function configures the given channel of the PIT timer.
<b>Parameter 1</b>	T_UBYTE PIT channel which must be configured.
<b>Parameter 2..n</b>	Does not apply
<b>Return Value</b>	Void
<b>Precondition</b>	This function should be called in the beginning of the main application.
<b>Post condition</b>	The initialized channel of the PIT will be ready to use.
<b>Error Conditions</b>	Does not apply.

## 5.7 LED

### 5.7.1 Function void Init\_LED (void)

<b>Description</b>	This function initializes the pin used for the LED.
<b>Parameter 1</b>	Void
<b>Return Value</b>	Void
<b>Precondition</b>	Does not apply
<b>Post condition</b>	The LED will be able to be changed.
<b>Error Conditions</b>	Does not apply

### 5.7.2 Function T\_UBYTE Get\_LED\_Status (void)

<b>Description</b>	This function returns the state of the LED.
<b>Parameter 1</b>	Void
<b>Return Value</b>	T_UBYTE. Corresponds to the logic state of the LED. It can be either 0 or 1, meaning off or on, respectively.
<b>Precondition</b>	Does not apply
<b>Post condition</b>	Does not apply
<b>Error Conditions</b>	Does not apply

### 5.7.3 Function void Set\_LED\_Status (T\_UBYTE)

<b>Description</b>	This function changes the LED status depending on the parameter received.
<b>Parameter 1</b>	T_UBYTE. It can receive either 0 or 1 to turn it off or on respectively.
<b>Return Value</b>	Void
<b>Precondition</b>	The function Init_LED should be executed before.
<b>Post condition</b>	The LED status will change to the one received in the parameter.
<b>Error Conditions</b>	Does not apply

## 5.8 GPIO

### 5.8.1 Function void Set\_Pin\_State (T\_UBYTE, T\_UBYTE)

<b>Description</b>	Changes the logic level of the output pin selected.
<b>Parameter 1</b>	T_UBYTE. Corresponds to the pin number that should be affected.
<b>Parameter 2</b>	T_UBYTE. It can receive either 0 or 1 to turn the pin off or on respectively.
<b>Return Value</b>	Void
<b>Precondition</b>	The mode of the pin selected should be OUTPUT.
<b>Post condition</b>	The logic level for the selected pin, will be the one selected in the second parameter.
<b>Error Conditions</b>	Does not apply

### 5.8.2 Function void Set\_Pin\_Mode (T\_UBYTE, T\_UBYTE)

<b>Description</b>	This function changes the pin mode of the selected pin to the selected mode.
<b>Parameter 1</b>	T_UBYTE. Corresponds to the pin number that should be affected.
<b>Parameter 2</b>	T_UBYTE. It can receive either 0 or 1 to set the pin mode to OUTPUT or INPUT respectively.
<b>Return Value</b>	Void
<b>Precondition</b>	Does not apply
<b>Post condition</b>	The selected pin will act as the selected pin mode.
<b>Error Conditions</b>	Does not apply

### 5.8.3 Function T\_UBYTE Get\_Pin\_State (T\_UBYTE)

<b>Description</b>	This function returns the state of the given pin.
<b>Parameter 1</b>	T_UBYTE. Corresponds to the pin number which state is unknown.
<b>Return Value</b>	T_UBYTE. Corresponds to the logic state of the pin. It can be either 0 or 1, meaning off or on, respectively.
<b>Precondition</b>	Does not apply
<b>Post condition</b>	Does not apply
<b>Error Conditions</b>	Does not apply

## 5.9 KERNEL

### 5.9.1 Function void Tick\_ISR (void)

<b>Description</b>	This function is the one that handles the clock Ticks in order to trigger the tasks. This interrupt runs periodically every 10 milliseconds according to the configuration of the PIT.
<b>Parameter 1</b>	Void
<b>Return Value</b>	Void
<b>Precondition</b>	This function is called when a PIT interrupt is generated.
<b>Post condition</b>	It will literally interrupt the flow of the program to implement its code.
<b>Error Conditions</b>	Does not apply

### 5.9.2 Function void InitModesAndClock (void)

<b>Description</b>	It initializes the mode of operation and clocks.
<b>Parameter 1</b>	Void
<b>Parameter 2</b>	Does not apply.
<b>Return Value</b>	Void
<b>Precondition</b>	This should be the first function called in the main program.
<b>Post condition</b>	It will be possible to use the oscillator with the configurations done.
<b>Error Conditions</b>	Does not apply.

### 5.9.3 Function void main\_Scheduler (void)

<b>Description</b>	This function contains the main function of the scheduler which controls the timing for each of the tasks.
<b>Parameter 1</b>	Void
<b>Return Value</b>	Void
<b>Precondition</b>	The Global_Init function should be executed before.
<b>Post condition</b>	Does not apply
<b>Error Conditions</b>	Does not apply