Sprints

Basic Communication Manager

Design

Bassel Yasser Mahmoud

2023

Contents

Project Introduction

The Basic Communication Manager module has a capability to work with different serial communication protocol using ISR with the highest possible throughput.

In this design Docs we'll discuss layered architecture, module description, drivers' documentation and UML on High Level Design.

We'll discuss also flowchart of each module, Pre-compiling configuration and Linking configuration on Low Level Design

High Level Design

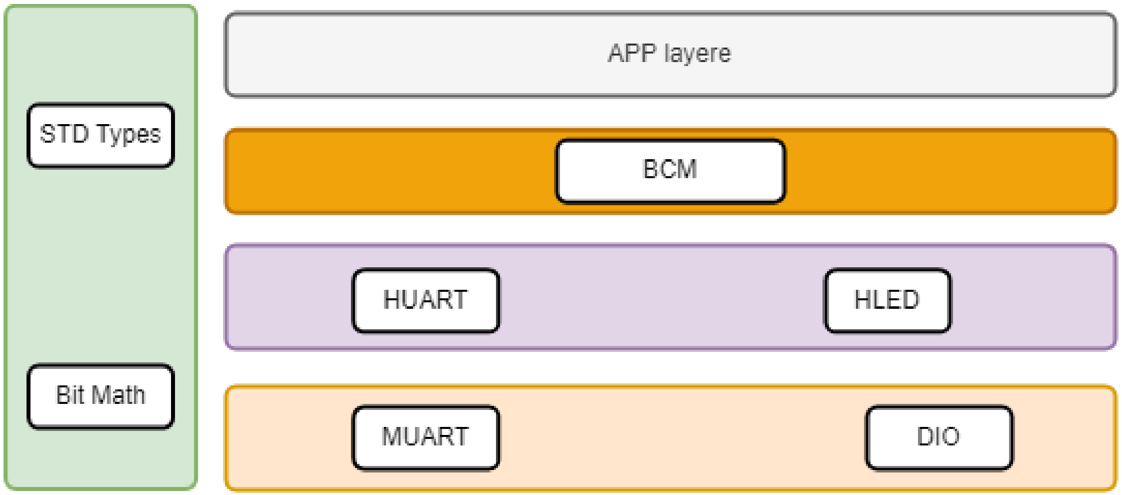
Layered Architecture

APP Layer: written in high level languages like java, C++, C# with rich GUI support. Application layer calls the middleware api in response to action by the user or an event.

HAL Layer: are a way to provide an interface between hardware and software so applications can be device independent.

MCAL Layer: is a software module that directly accesses on-chip MCU peripheral modules and external devices that are mapped to memory, and makes the upper software layer independent of the MCU. Details of the MCAL software module are shown below.

Common Layer: is the layer which consists of BIT_MATH and STD types



Module Description

- **APP Layer**
  - **App:** written in high level languages like java, C++, C# with rich GUI support. Application layer calls the middleware api in response to action by the user or an event.
- **SERVICE Layer**
  - **Sbcm:** In this module configure communication protocol selection
- **HAL Layer**
  - **Huart:** this module communicates with Muart on MCAL layer
  - **Led:** this led module configure selected pin as output and generate volt
- **MCAL Layer**
  - **Muart:** this module having configuration and Initialization for UART which communicate to hardware register directly
- **COMMON Layer**
  - **std_types:** having basic standard types like (Uint32_t, Uint8_t, …..)
  - **bit_math :** Consist of bit manipulation like (SetBit, ClrBit, GetBit, ..)

Drivers’ documentation

APP

APP_vidInit

Service name	APP_vidInit
Description	This Function Make Modules Initialization
Syntax	void APP_vidInit (void)
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	void
Parameters (out)	None
Return	void
Available via	app.h

APP_vidStart

Service name	APP_vidStart
Description	This Function Start the Application.
Syntax	void APP_vidStart (void)
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	void
Parameters (out)	None
Return	void
Available via	app.h

SERVICE

BCM module

bcm_init

Service name	**bcm_init**
Description	This Function Initialize Specific communication protocol
Syntax	enu_system_status_t **bcm_init** (str_bcm_instance_t* ptr_str_instance_t);
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	**ptr_str_instance_t:** address of BCM instance
Parameters (out)	None
Return	enu_system_status_t
Available via	bcm.h

bcm_deinit

Service name	**bcm_deinit**
Description	This Function De-Initialize Specific communication protocol
Syntax	enu_system_status_t **bcm_deinit** (str_bcm_instance_t* ptr_str_instance_t)
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	**ptr_str_instance_t:** address of BCM instance
Parameters (out)	None
Return	enu_system_status_t
Available via	bcm.h

bcm_send

Service name	bcm_send
Description	This Function Send One byte of data
Syntax	`enu_system_status_t bcm_send (str_bcm_instance_t* ptr_str_instance_t, Uint8_t u8_one_byte_data)`
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	**ptr_str_instance_t:** address of BCM instance **u8_one_byte_data:** Copy of data
Parameters (out)	None
Return	`enu_system_status_t`
Available via	bcm.h

bcm_send_n

Service name	bcm_send_n
Description	This Function send N byte of data
Syntax	`enu_system_status_t bcm_send_n (str_bcm_instance_t* ptr_str_instance_t, Uint8_t* ptr_u8_n_byte_data, Uint16_t u16_byte_length)`
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	**ptr_str_instance_t:** address of BCM instance **ptr_u8_n_byte_data:** Copy of array **u8_byte_length:** Length of array
Parameters (out)	None
Return	`enu_system_status_t`
Available via	bcm.h

bcm_dispatcher

Service name	**bcm_dispatcher**
Description	Is periodic function and notifies the user with need event
Syntax	`enu_system_status_t bcm_dispatcher (str_bcm_instance_t* ptr_str_instance_t)`
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	**ptr_str_instance_t:** address of BCM instance
Parameters (out)	None
Return	`enu_system_status_t`
Available via	bcm.h

HAL

HUART module

HUART_enInit

Service name	HUART_enInit
Description	This Function call MUART_enInit on MCAL layer
Syntax	`en_huartErrStat_t HUART_enInit (Uint32_t copy_u32BaudRateH)`
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	`copy_u32BaudRateH: Copy of Baudrate`
Parameters (out)	None
Return	`en_huartErrStat_t: HUART_OK, HUART_NOK`
Available via	huart_Interface.h

HUART_enDeInit

Service name	HUART_enDeInit
Description	This Function de Initialize UART
Syntax	`en_huartErrStat_t HUART_enDeInit(void)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	`void`
Parameters (out)	None
Return	`en_huartErrStat_t: HUART_OK, HUART_NOK`
Available via	huart_Interface.h

HUART_enSyncSendData

Service name	HUART_enSyncSendData
Description	This Function call **MUART_enSyncSendData** on MCAL layer
Syntax	`en_huartErrStat_t HUART_enSyncSendData(Uint8_t Copy_u8DataH)`
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	**Copy_u8DataH:** Copy of One Byte Data
Parameters (out)	None
Return	en_huartErrStat_t: *HUART_OK*, *HUART_NOK*
Available via	huart_Interface.h

HUART_enAsyncSendData

Service name	HUART_enAsyncSendData
Description	This Function call **MUART_enAsyncSendData** on MCAL layer
Syntax	`en_huartErrStat_t HUART_enAsyncSendData (Uint8_t Copy_u8DataH)`
Sync/Async	Asynchronous
Reentrancy	Non-Reentrant
Parameters (in)	**Copy_u8DataH:** Copy of One Byte Data
Parameters (out)	None
Return	en_huartErrStat_t: *HUART_OK*, *HUART_NOK*
Available via	huart_Interface.h

HUART_enRecieveData

Service name	**HUART_enReieveData**
Description	This Function call MUART_enRecieveData on MCAL layer
Syntax	`en_huartErrStat_t HUART_enRecieveData (Uint8_t* Ref_u8DataH)`
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	**None**
Parameters (out)	**Ref_u8DataH:** Address of variable which data to be stored
Return	en_huartErrStat_t: *HUART_OK*, *HUART_NOK*
Available via	huart_Interface.h

HUART_sendSyncString

Service name	**HUART_sendSyncString**
Description	This Function call **MUART_sendSyncString**on MCAL layer
Syntax	`void HUART_sendSyncString (Uint8_t * Hstr, Uint8_t u8_arr_size)`
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	**p_u8_string:** Copy of array of char or String **u8_arr_size: Copy of array size**
Parameters (out)	None
Return	void
Available via	huart_Interface.h

HUART_sendAsyncString

Service name	HUART_sendAsyncString
Description	This Function call **MUART_sendAsyncString** MCAL layer
Syntax	`void HUART_sendAsyncString (Uint8_t * Hstr, Uint16_t u16_arr_size)`
Sync/Async	Asynchronous
Reentrancy	Non-Reentrant
Parameters (in)	**p_u8_string:** Copy of array of char or String **u8_arr_size: Copy of array size**
Parameters (out)	None
Return	`void`
Available via	huart_Interface.h

HUART_receiveSTRING

Service name	HUART_receiveSTRING
Description	This Function call MUART_ receiveSTRIN**G** on MCAL layer
Syntax	`void HUART_receiveSTRING (Uint8_t * p_u8_arr, Uint8_t p_u8_arr_size)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	**p_u8_arr:** Empty Array which data to be stored **p_u8_arr_size:** Array size
Parameters (out)	None
Return	`void`
Available via	huart_Interface.h

HUART_receiveAsyncString

Service name	**HUART_receiveAsyncString**
Description	This Function call MUART_receiveAsyncString on MCAL layer
Syntax	`void HUART_receiveAsyncString (Uint16_t u16_arr_size)`
Sync/Async	S\Asynchronous
Reentrancy	Reentrant
Parameters (in)	**u16_arr_size:** buffer size that data to be stored
Parameters (out)	None
Return	`void`
Available via	huart_Interface.h

HUART_enEnableInterrupt

Service name	**HUART_enEnableInterrupt**
Description	This Function call **MUART_enEnableInterrupt** on MCAL layer
Syntax	`en_huartErrStat_t HUART_enEnableInterrupt(en_huart_tx_rx_sel_t en_huart_tx_rx_sel)`
Sync/Async	Asynchronous
Reentrancy	Reentrant
Parameters (in)	**en_huart_tx_rx_sel:** Take kind of operation (TX or RX)
Parameters (out)	None
Return	`en_huartErrStat_t`
Available via	huart_Interface.h

HLED module

HLed_Init

Service name	HLed_Init
Description	This Function Init LED dio pin as output
Syntax	`enu_ledError_t HLed_Init (enu_pin en_pinNum)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	`en_pinNum: dio pin selection`
Parameters (out)	None
Return	`en_ledError_t`
Available via	hled.h

HLed_on

Service name	HLed_on
Description	This Function give LED pin logic 1
Syntax	`enu_ledError_t HLed_on (enu_pin en_pinx);`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	`en_pinNum: dio pin selection`
Parameters (out)	None
Return	`en_ledError_t`
Available via	hled.h

HLed_off

Service name	**HLed_off**
Description	This Function give LED pin logic 0
Syntax	`enu_ledError_t HLed_off (enu_pin en_pinx)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	**`en_pinNum: dio pin selection`**
Parameters (out)	None
Return	`en_ledError_t`
Available via	hled.h

HLed_toggle

Service name	**HLed_toggle**
Description	This Function Change previous state of LED pin
Syntax	`enu_ledError_t HLed_toggle (enu_pin en_pinx)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	**`en_pinNum: dio pin selection`**
Parameters (out)	None
Return	`en_ledError_t`
Available via	hled.h

MCAL

MUART module

MUART_enInit

Service name	MUART_enInit
Description	This Function Initialize UART configuration
Syntax	`en_uartErrStat_t MUART_enInit (Uint32_t copy_u32BaudRateH)`
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	`copy_u32BaudRateH: Copy of Baudrate`
Parameters (out)	None
Return	`en_uartErrStat_t: MUART_OK, MUART_NOK`
Available via	muart_Interface.h

MUART_en_TX_Enable

Service name	MUART_en_TX_Enable
Description	This Function Transmitter Enable
Syntax	`void MUART_en_TX_Enable(void)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	`void`
Parameters (out)	None
Return	`void`
Available via	muart_Interface.h

MUART_en_RX_Enable

Service name	MUART_en_TX_Enable
Description	This Function Receiver Enable
Syntax	`void MUART_en_RX_Enable(void)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	`void`
Parameters (out)	None
Return	`void`
Available via	muart_Interface.h

MUART_en_TX_Disable

Service name	MUART_en_TX_Disable
Description	This Function Disable Transmitter
Syntax	`void MUART_en_TX_Disable(void)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	`void`
Parameters (out)	None
Return	`void`
Available via	muart_Interface.h

MUART_en_RX_Disable

Service name	**MUART_en_RX_Disable**
Description	This Function Disable Receiver
Syntax	`void MUART_en_RX_Disable(void)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	`void`
Parameters (out)	None
Return	`void`
Available via	muart_Interface.h

MUART_en_TX_RX_Enable

Service name	**MUART_en_TX_RX_Enable**
Description	This Function Enable Transmitter & Receiver
Syntax	`void MUART_en_TX_RX_Enable(void)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	`void`
Parameters (out)	None
Return	`void`
Available via	muart_Interface.h

MUART_en_TX_RX_Disable

Service name	MUART_en_TX_RX_Disable
Description	This Function Disable Transmitter & Receiver
Syntax	`void MUART_en_TX_RX_Disable(void)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	`void`
Parameters (out)	None
Return	`void`
Available via	muart_Interface.h

MUART_enSyncSendData

Service name	MUART_enSyncSendData
Description	This Function Send data via UDR register
Syntax	`en_uartErrStat_t MUART_enSyncSendData (Uint8_t Copy_u8Data)`
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	**Copy_u8Data:** Copy of One Byte Data
Parameters (out)	None
Return	`en_uartErrStat_t: MUART_OK, MUART_NOK`
Available via	muart_Interface.h

MUART_enAsyncSendData

Service name	MUART_enAsyncSendData
Description	This Function send data and this function is non blocking
Syntax	`en_huartErrStat_t HUART_enAsyncSendData (Uint8_t Copy_u8DataH)`
Sync/Async	Asynchronous
Reentrancy	Non-Reentrant
Parameters (in)	**Copy_u8DataH:** Copy of One Byte Data
Parameters (out)	None
Return	en_uartErrStat_t: *MUART_OK*, *MUART_NOK*
Available via	huart_Interface.h

MUART_enRecieveData

Service name	MUART_enReieveData
Description	This Function Receive data via UDR register
Syntax	`en_uartErrStat_t MUART_enRecieveData (Uint8_t* Ref_u8DataH)`
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	**None**
Parameters (out)	**Ref_u8DataH:** Address of variable which data to be stored
Return	en_uartErrStat_t: *MUART_OK*, *MUART_NOK*
Available via	muart_Interface.h

MUART_sendSyncString

Service name	**MUART_sendSyncString**
Description	This Function Send group of char
Syntax	`void MUART_sendSyncString(Uint8_t * str, Uint8_t u8_arr_size)`
Sync/Async	Synchronous
Reentrancy	Non-Reentrant
Parameters (in)	**p_u8_string:** Copy of array of char or String **u8_arr_size:** array or string length to be sent
Parameters (out)	None
Return	`void`
Available via	muart_Interface.h

MUART_sendAsyncString

Service name	**MUART_sendAsyncString**
Description	This Function Send group of char
Syntax	`void MUART_sendAsyncString (Uint8_t * str, Uint16_t u16_arr_size)`
Sync/Async	Asynchronous
Reentrancy	Non-Reentrant
Parameters (in)	**p_u8_string:** Copy of array of char or String **u8_arr_size:** array or string length to be sent
Parameters (out)	None
Return	`void`
Available via	muart_Interface.h

MUART_receiveAsyncString

Service name	MUART_receiveAsyncString
Description	This Function Send group of char
Syntax	`void MUART_receiveAsyncString(Uint16_t u16_arr_size)`
Sync/Async	Asynchronous
Reentrancy	Non-Reentrant
Parameters (in)	**u16_arr_size:** array or string length to be received
Parameters (out)	None
Return	`void`
Available via	muart_Interface.h

MUART_receiveSTRING

Service name	MUART_receiveSTRING
Description	This Function Receive group of char
Syntax	`void MUART_receiveSTRING (Uint8_t * p_u8_arr, Uint8_t p_u8_arr_size)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	**p_u8_arr:** Empty Array which data to be stored **p_u8_arr_size:** Array size
Parameters (out)	None
Return	`void`
Available via	muart_Interface.h

MUART_enEnableInterrupt

Service name	**MUART_enEnableInterrupt**
Description	This Function Enable UART Interrupt
Syntax	`en_uartErrStat_t MUART_enEnableInterrupt (en_muart_interrupt_t en_muart_interrupt)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	**`en_muart_interrupt:`** choosing which INT fires (TX or RX)
Parameters (out)	None
Return	`en_uartErrStat_t`
Available via	muart_Interface.h

MUART_enDisableInterrupt

Service name	**MUART_enDisableInterrupt**
Description	This Function Disable UART Interrupt
Syntax	`en_uartErrStat_t MUART_enDisableInterrupt (en_muart_interrupt_t en_muart_interrupt)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	**`en_muart_interrupt:`** choosing which INT Disabled (TX or RX)
Parameters (out)	None
Return	`en_uartErrStat_t`
Available via	muart_Interface.h

DIO module

DIO_s8SETPinDir

Service name	DIO_s8SETPinDir
Description	This Function Initialize Pin Direction Input or Output
Syntax	`Sint8_t DIO_s8SETPinDir (enu_pin enPinCopy, enu_dir enPortDir)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	**enPinCopy:** Select pin and port [DIO_PINA_0,………] **enPortDir:** Select Pin direction [INPUT, OUTPUT]
Parameters (out)	None
Return	Sint8_t: DIO_OK, DIO_NOK
Available via	dio_Interface.h

DIO_s8SETPinVal

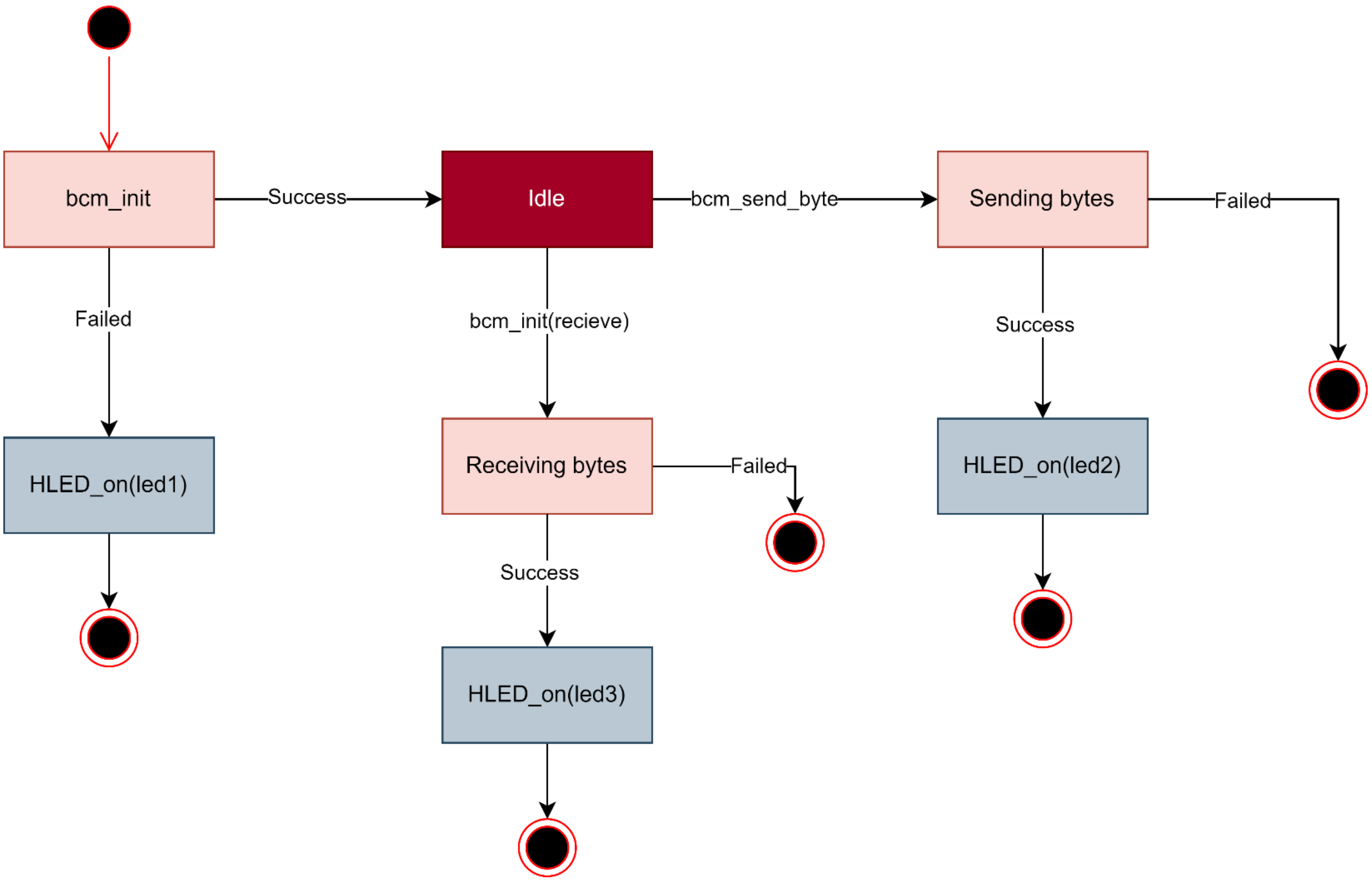
Service name	DIO_s8SETPinVal
Description	This Function Initialize Pin Value High or Low
Syntax	`Sint8_t DIO_s8SETPinVal (enu_pin enPinCopy, enu_dir enPortVal)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	**enPinCopy:** Select pin and port [DIO_PINA_0,………] **enPortDir:** Select Pin Value [HIGH, LOW]
Parameters (out)	None
Return	Sint8_t: DIO_OK, DIO_NOK
Available via	dio_Interface.h

DIO_s8GETPinVal

Service name	DIO_s8GETPinVal
Description	This Function Get value from selected pin
Syntax	`Sint8_t DIO_s8GETPinVal (enu_pin enPinCopy, Uint8_t* pu8Val)`
Sync/Async	Synchronous
Reentrancy	Reentrant
Parameters (in)	**enPinCopy:** Select pin and port [DIO_PINA_0,………]
Parameters (out)	**pu8Val:** Address of variable which pin status to be stored
Return	Sint8_t: DIO_OK, DIO_NOK
Available via	dio_Interface.h

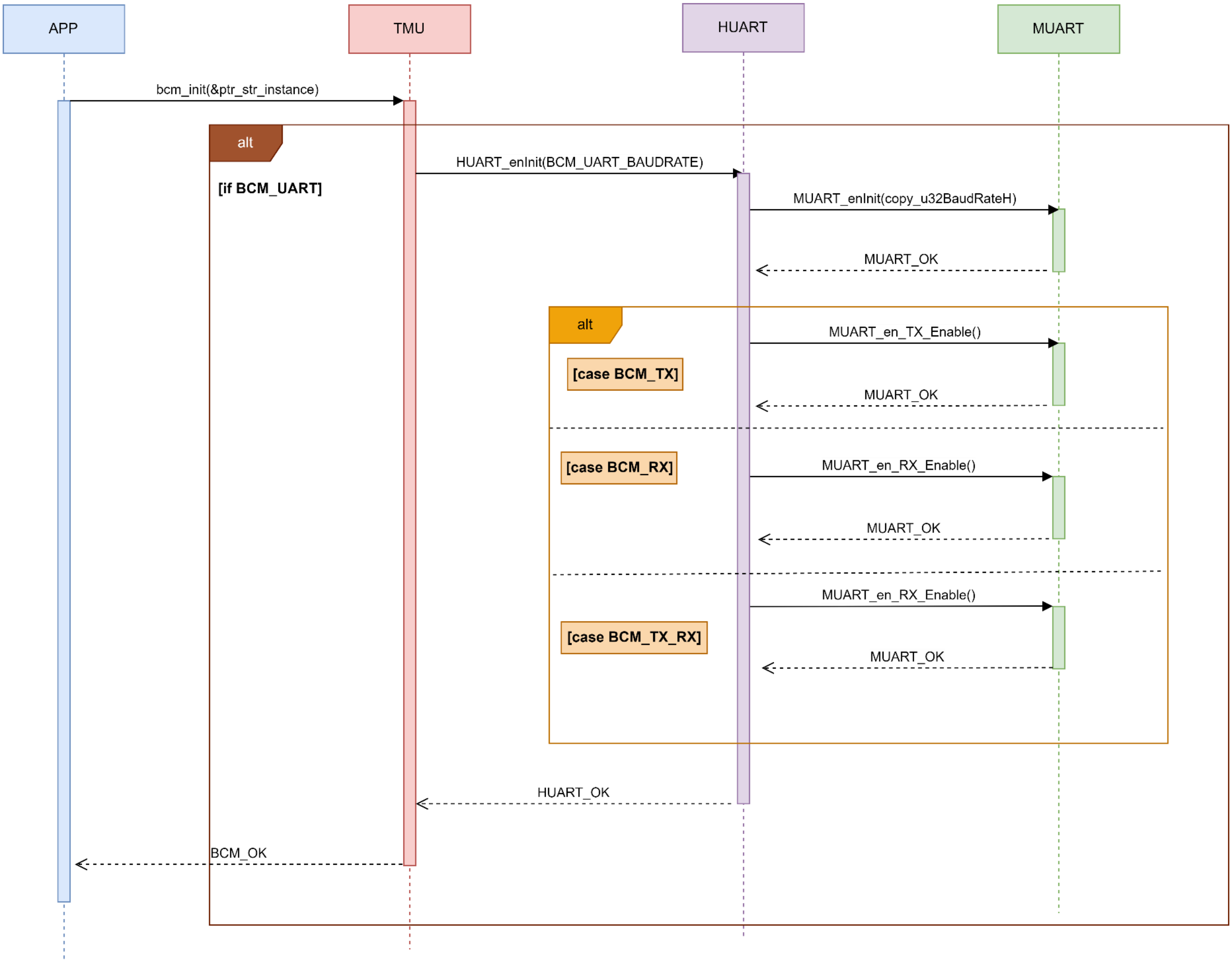
UML

State Machine

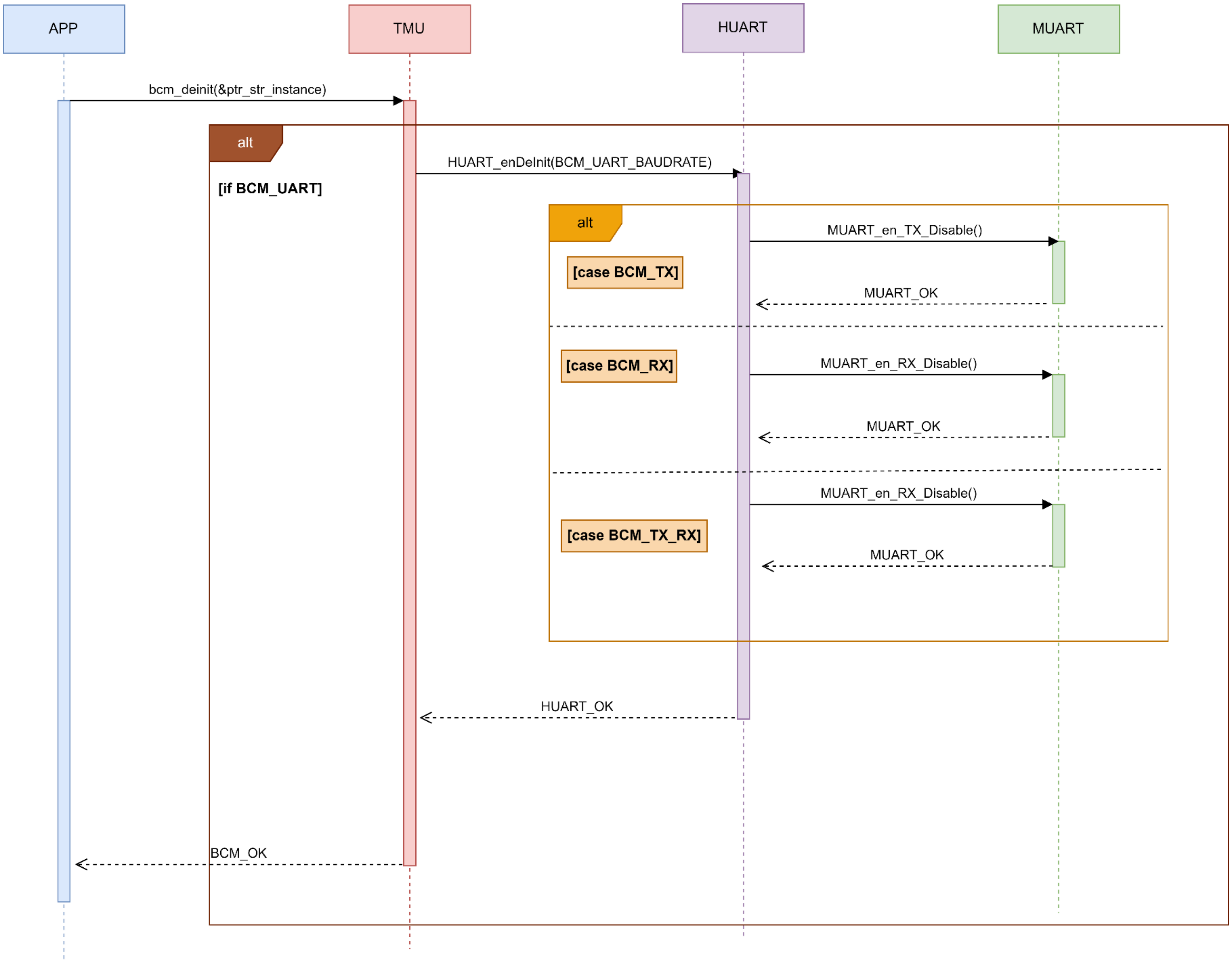


Sequence Diagram

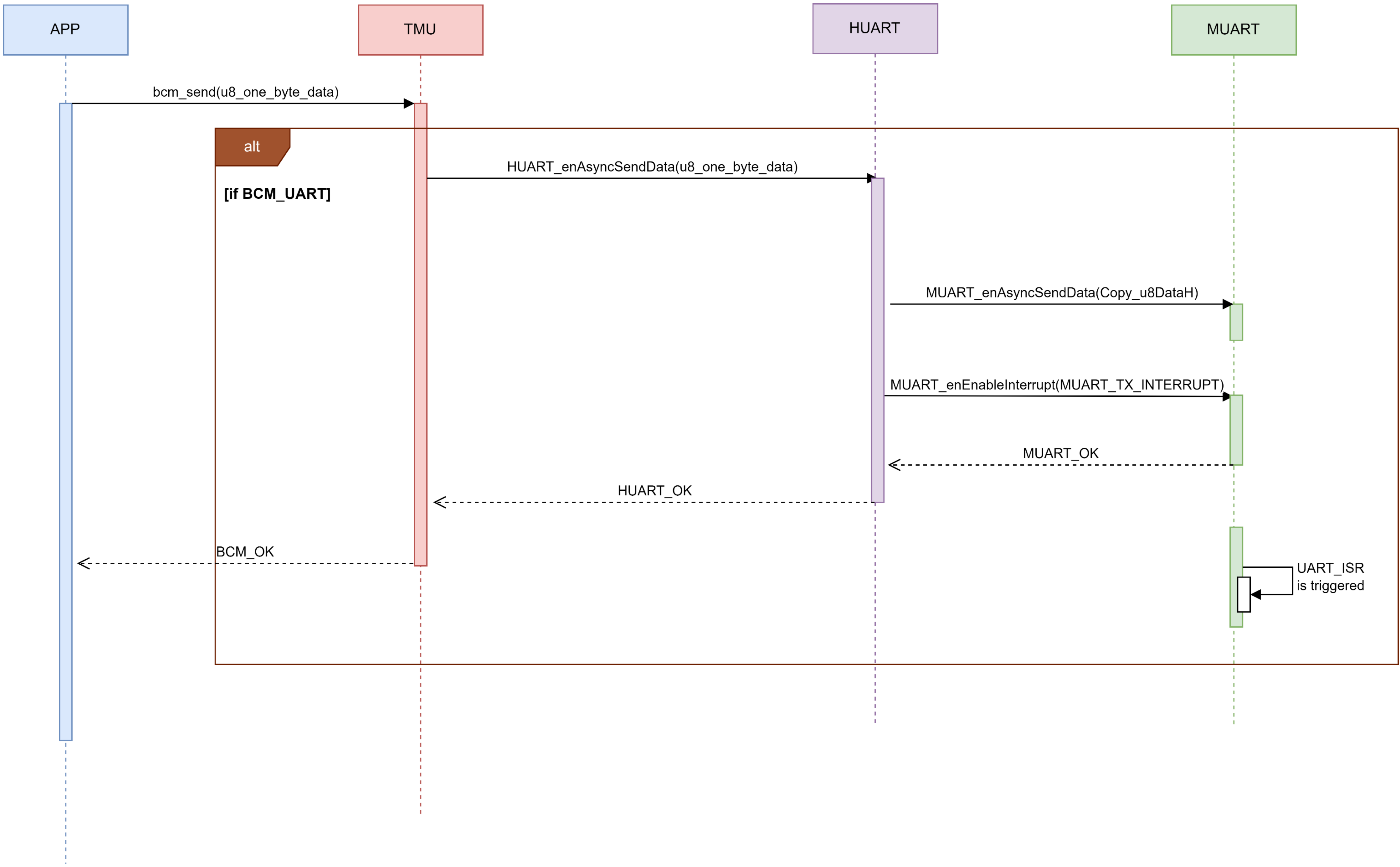
Communication Initialization



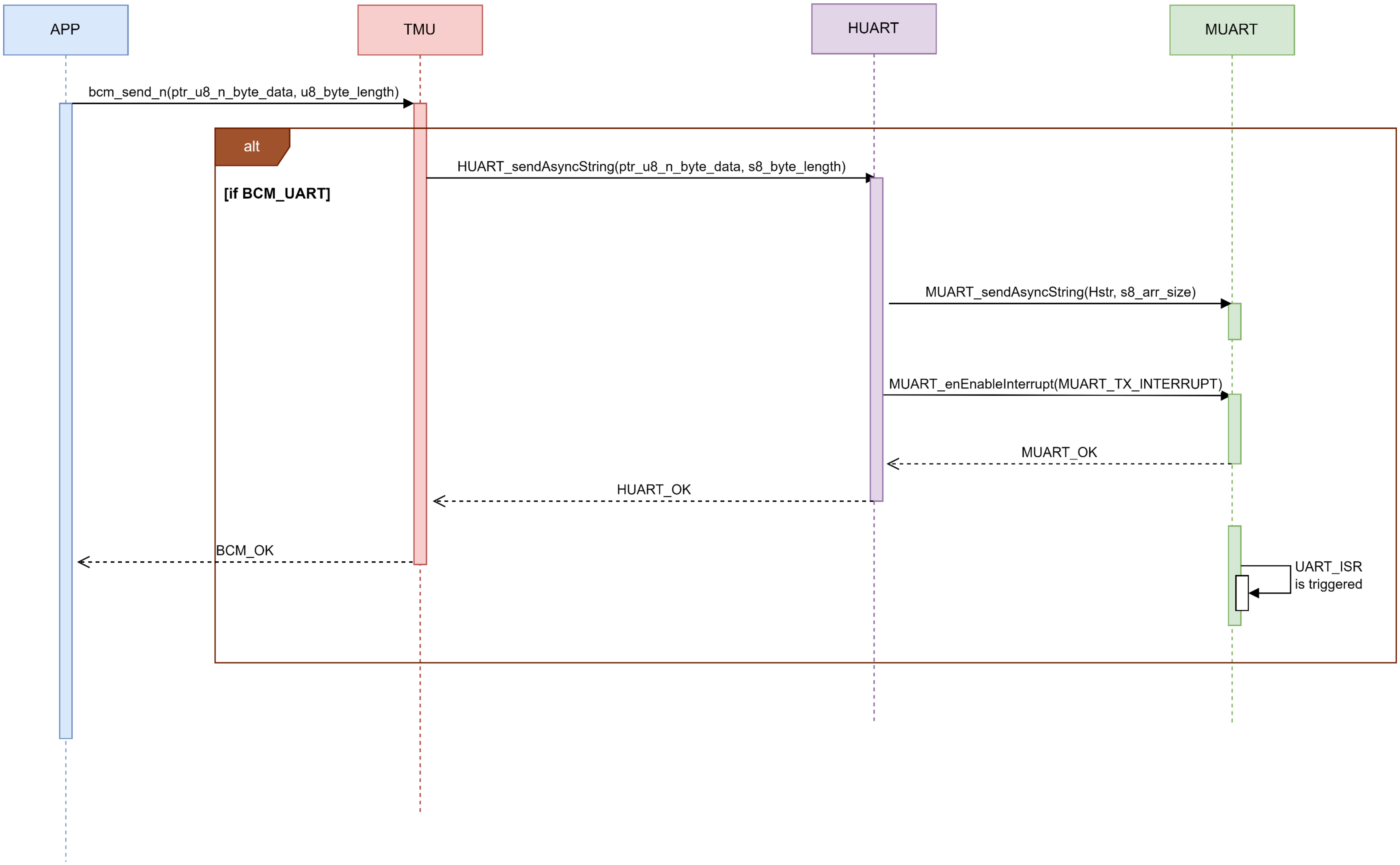
Communication De-Initialization



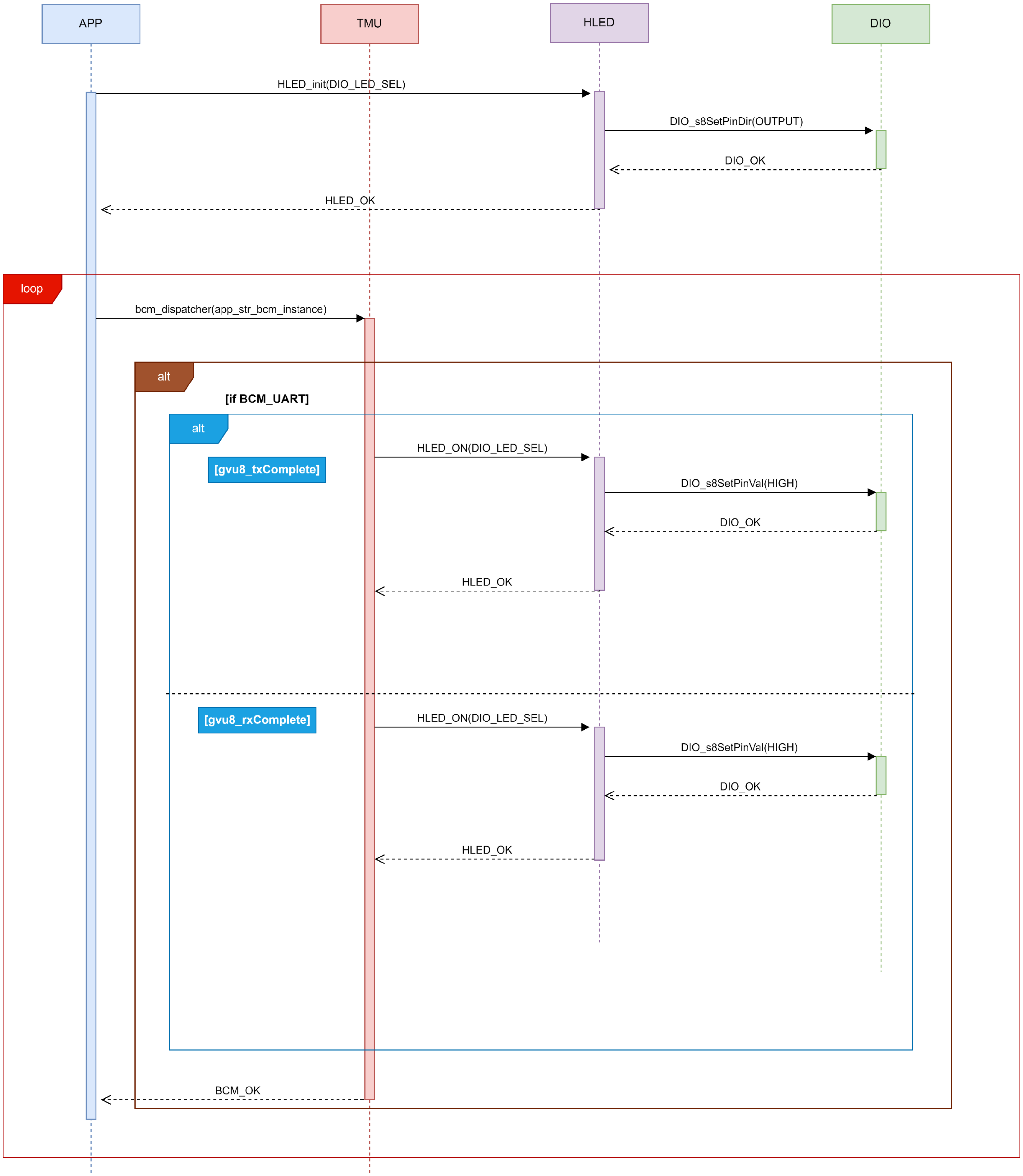
Sending Data (one byte)


APP
TMU
HUART
MUART
bcm_send(u8_one_byte_data)
alt
[if BCM_UART]
HUART_enAsyncSendData(u8_one_byte_data)
MUART_enAsyncSendData(Copy_u8DataH)
MUART_enEnableInterrupt(MUART_TX_INTERRUPT)
MUART_OK
HUART_OK
BCM_OK
UART_ISR
is triggered

Sending Data (N bytes)


APP
TMU
HUART
MUART
bcm_send_n(ptr_u8_n_byte_data, u8_byte_length)
alt
[if BCM_UART]
HUART_sendAsyncString(ptr_u8_n_byte_data, s8_byte_length)
MUART_sendAsyncString(Hstr, s8_arr_size)
MUART_enEnableInterrupt(MUART_TX_INTERRUPT)
MUART_OK
HUART_OK
BCM_OK
UART_ISR
is triggered

Dispatcher function

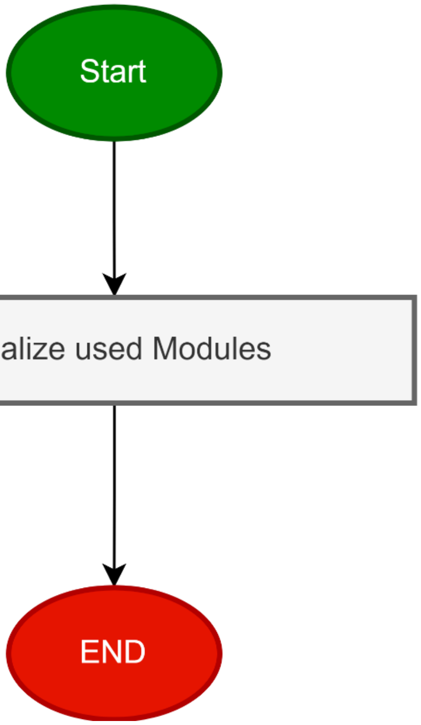


Low Level Design

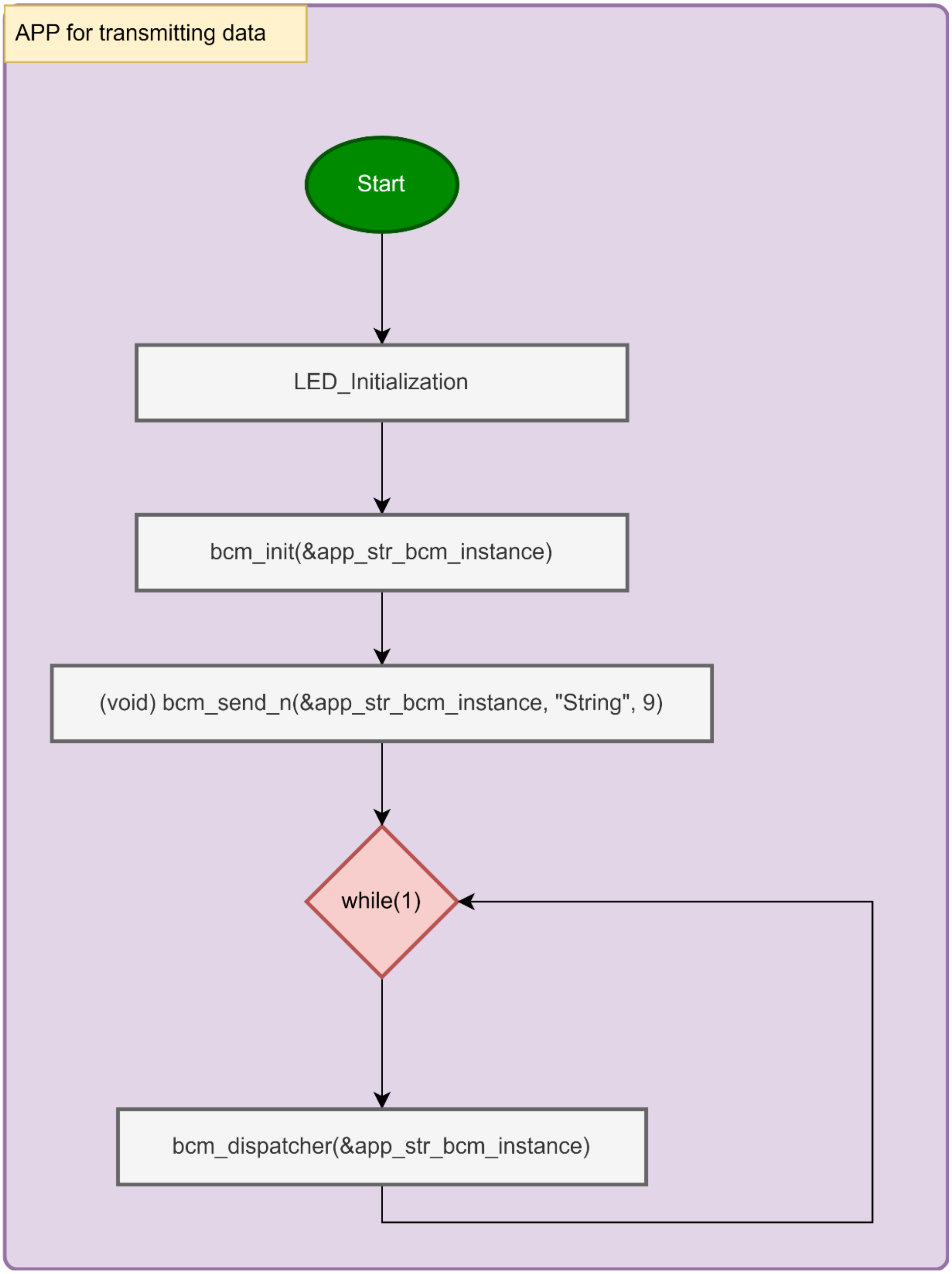
Flowchart

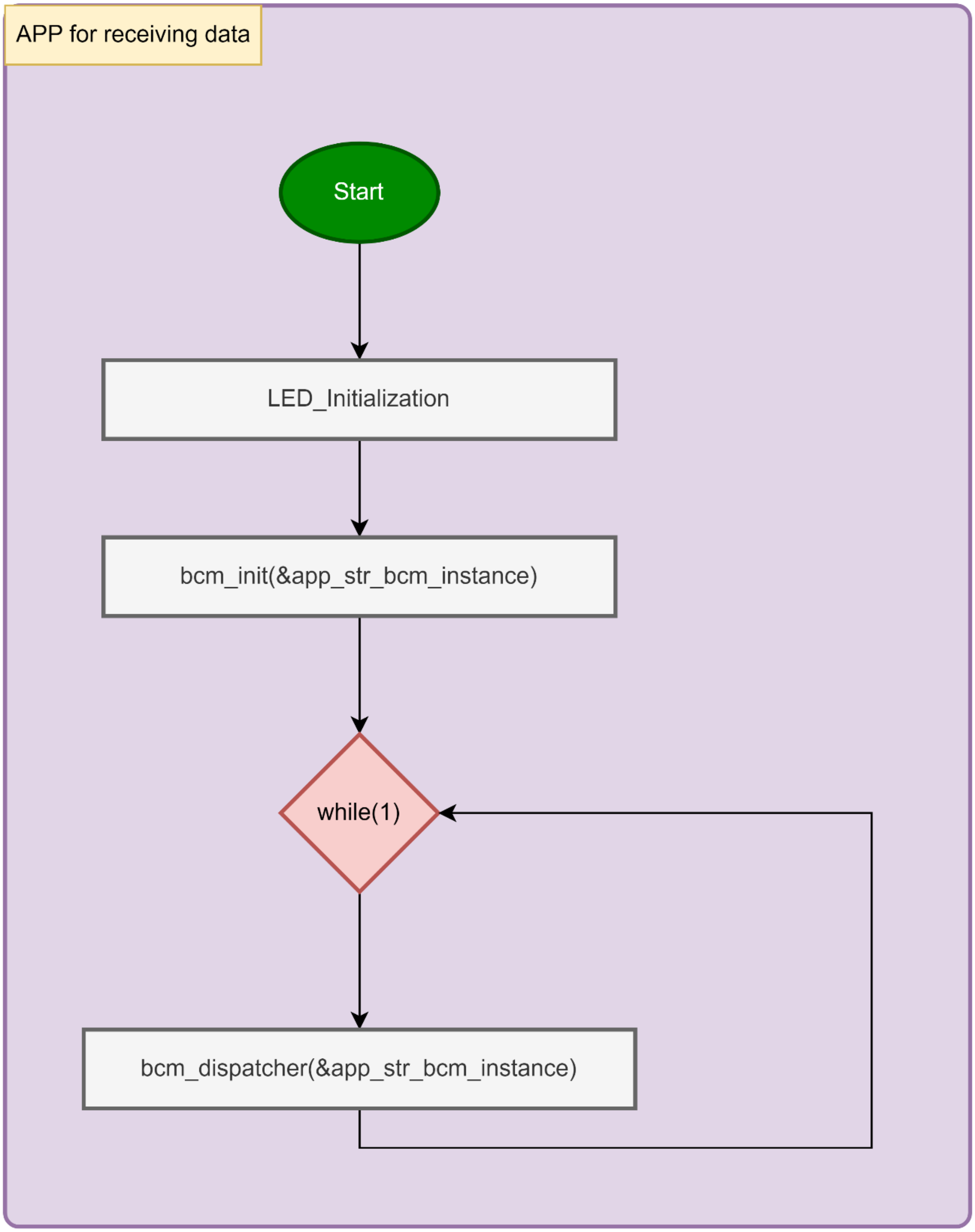
APP

APP_vidInit



APP_vidStart

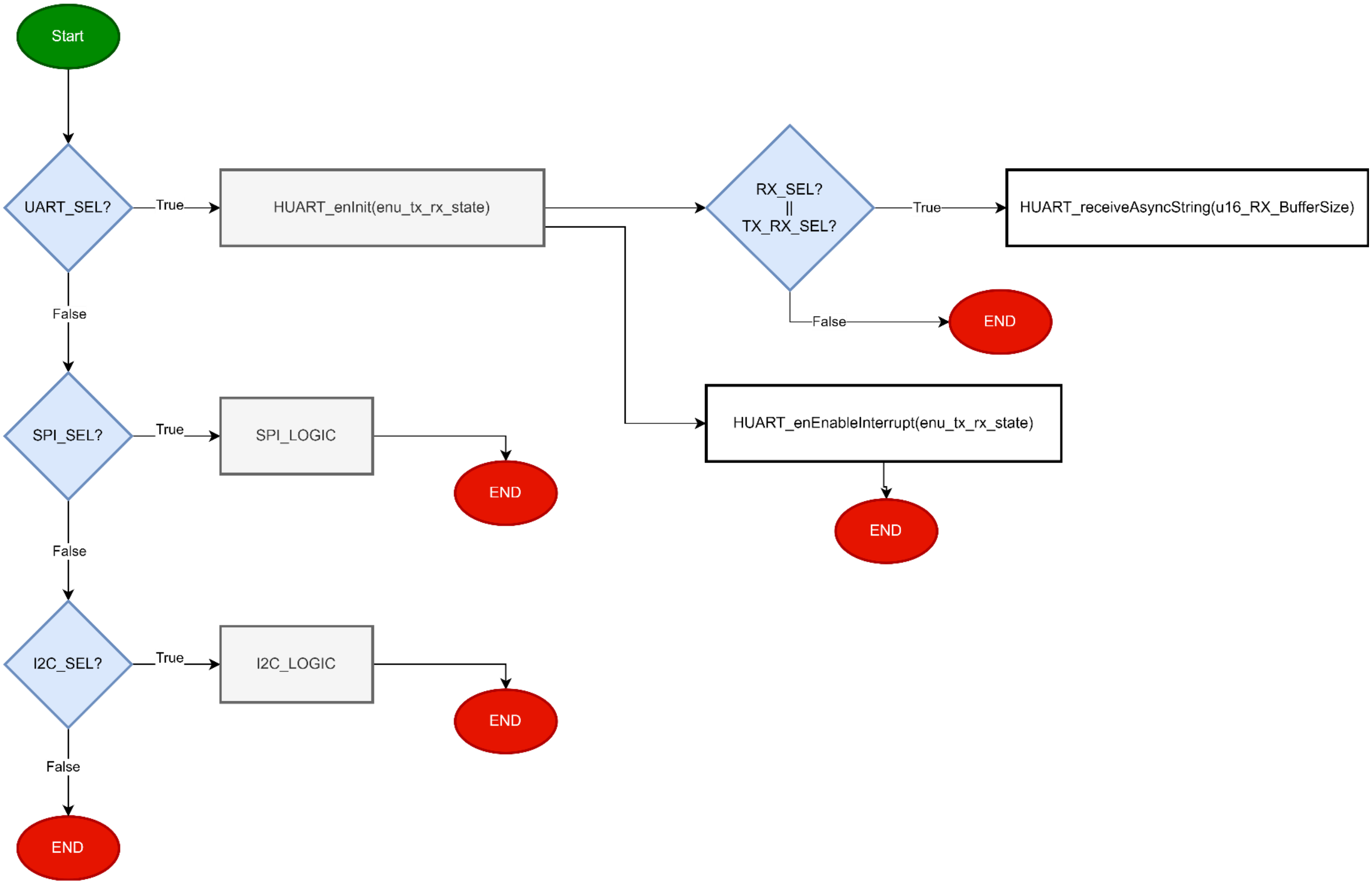

APP for transmitting data
Start
LED_Initialization
bcm_init(&app_str_bcm_instance)
(void) bcm_send_n(&app_str_bcm_instance, "String", 9)
while(1)
bcm_dispatcher(&app_str_bcm_instance)


APP for receiving data
Start
LED_Initialization
bcm_init(&app_str_bcm_instance)
while(1)
bcm_dispatcher(&app_str_bcm_instance)

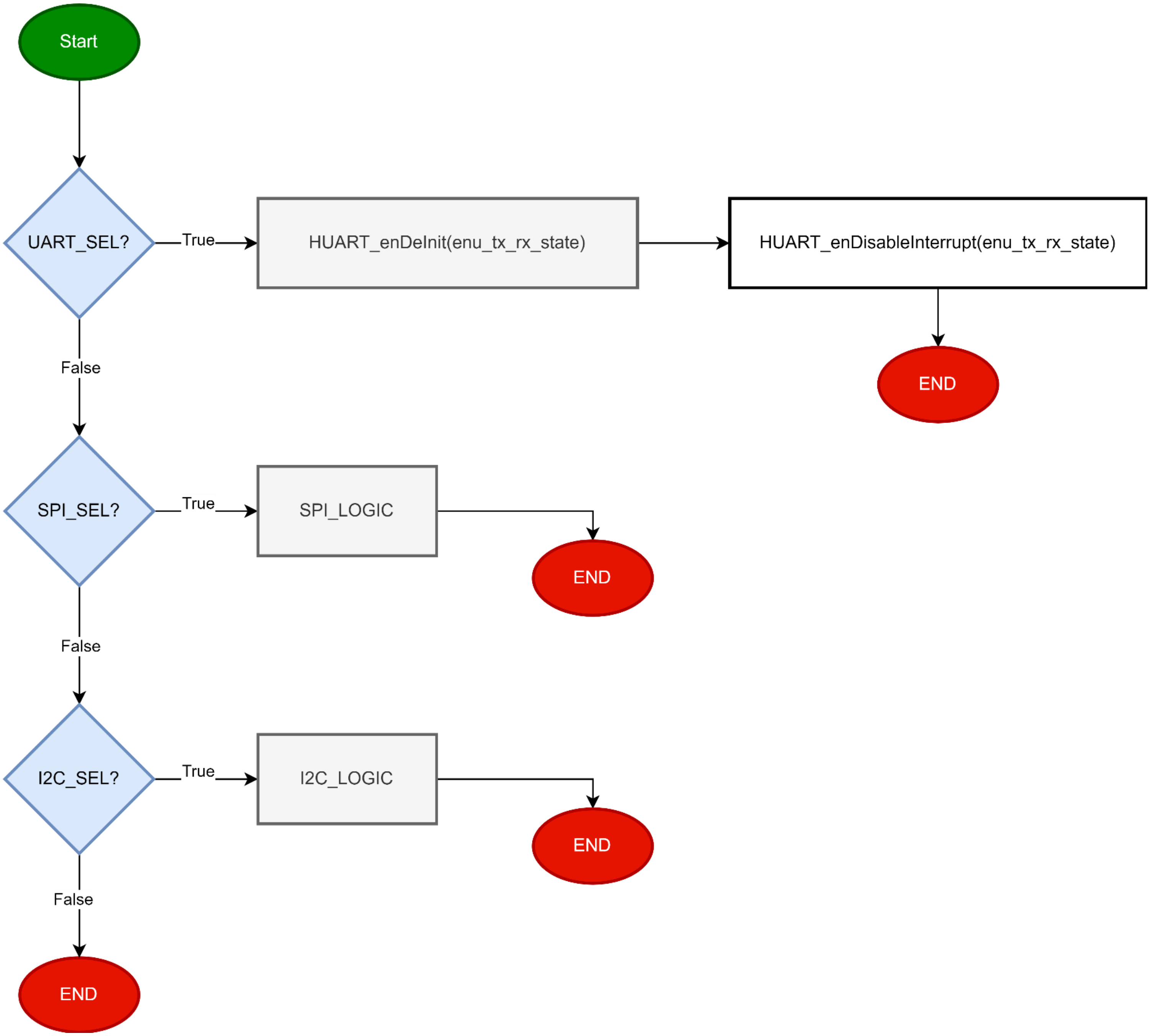
SERVICE

BCM module

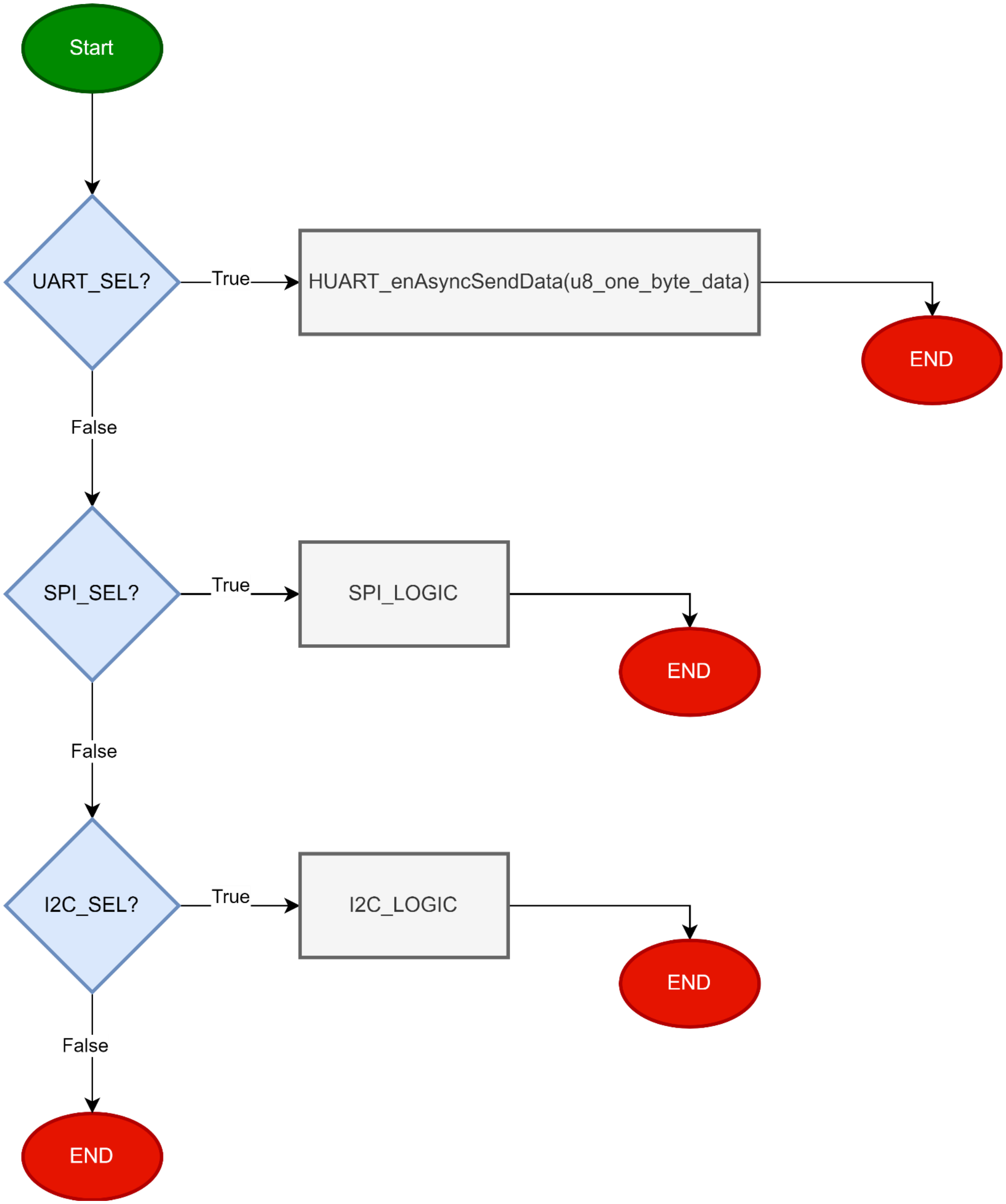
bcm_init



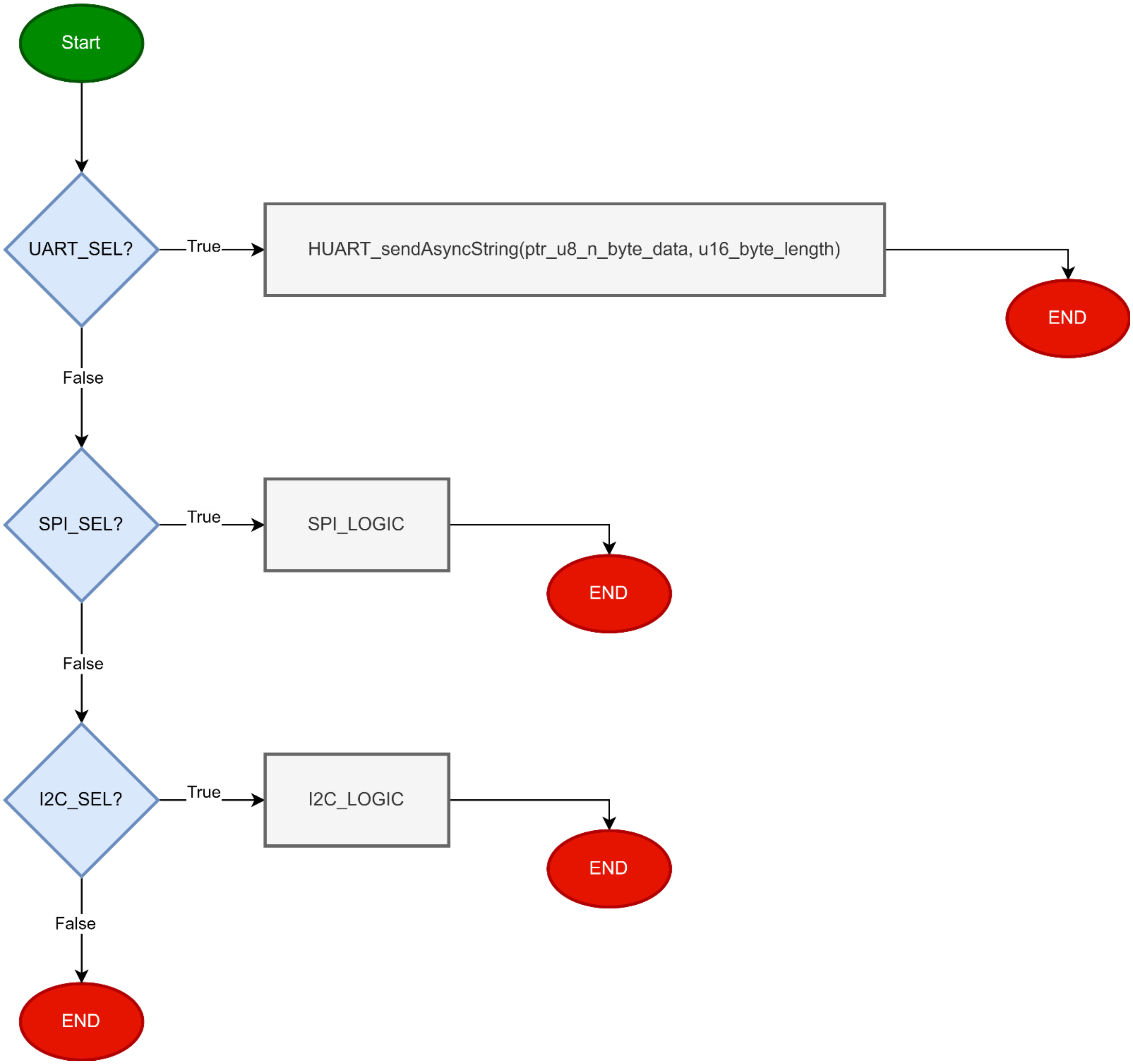
bcm_deinit


Start
UART_SEL?
True
HUART_enDeInit(enu_tx_rx_state)
HUART_enDisableInterrupt(enu_tx_rx_state)
END
False
SPI_SEL?
True
SPI_LOGIC
END
False
I2C_SEL?
True
I2C_LOGIC
END
False
END

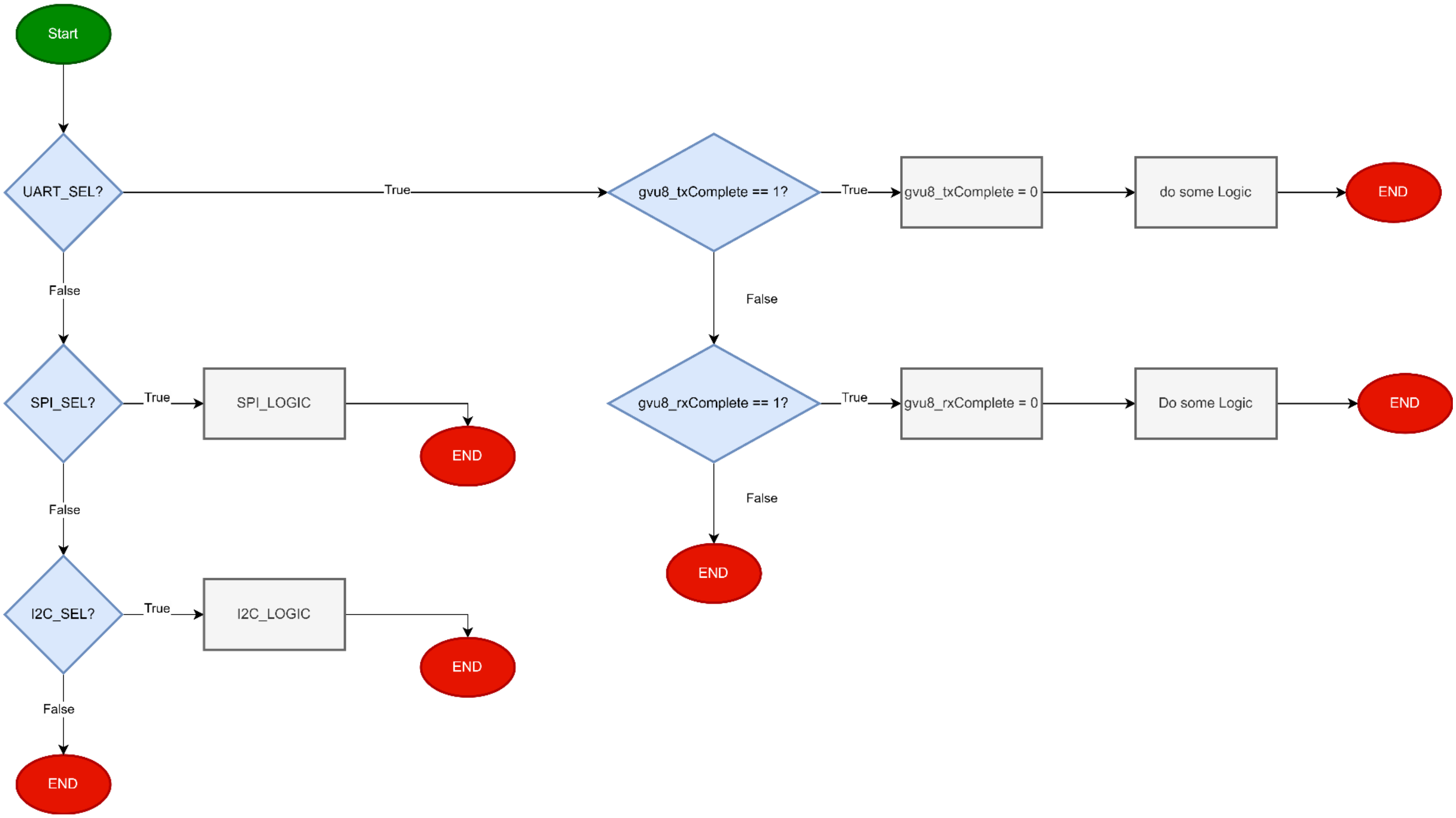
bcm_send


Start
UART_SEL?
True
HUART_enAsyncSendData(u8_one_byte_data)
END
False
SPI_SEL?
True
SPI_LOGIC
END
False
I2C_SEL?
True
I2C_LOGIC
END
False
END

bcm_send_n


Start
UART_SEL?
True
HUART_sendAsyncString(ptr_u8_n_byte_data, u16_byte_length)
END
False
SPI_SEL?
True
SPI_LOGIC
END
False
I2C_SEL?
True
I2C_LOGIC
END
False
END

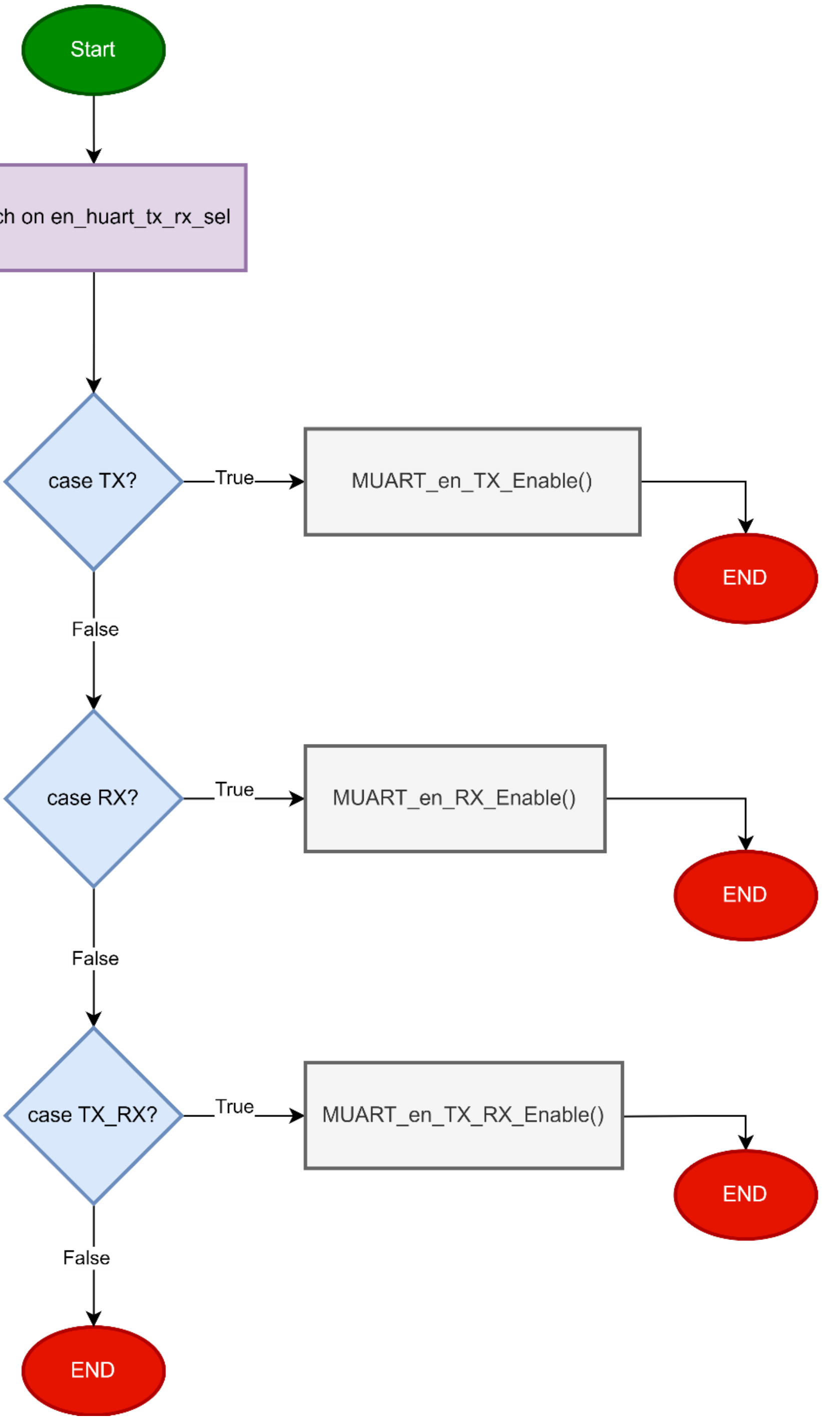
bcm_dispatcher



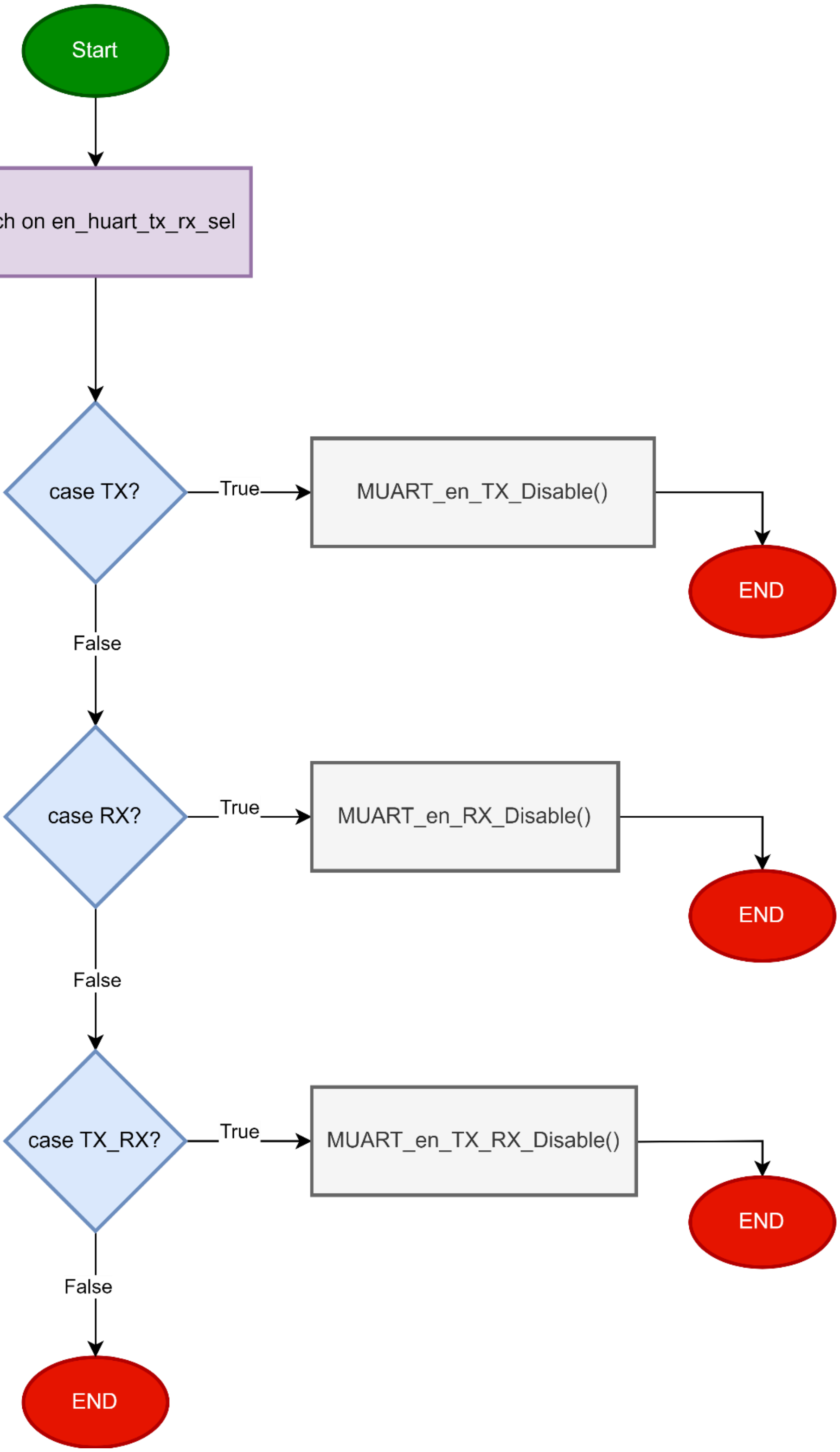
HAL

HUART module

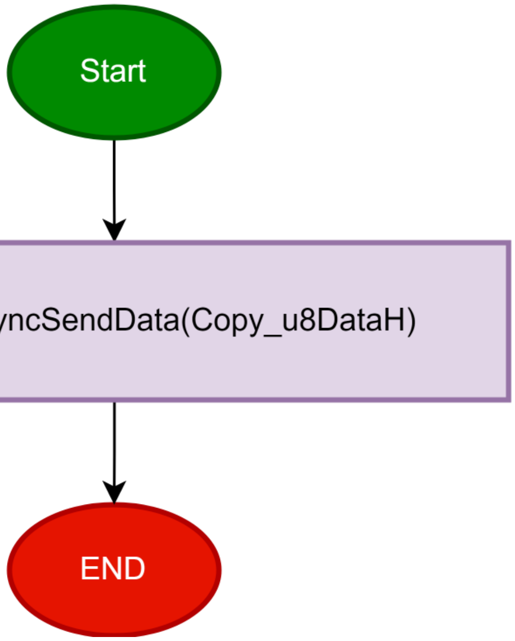
HUART_enInit



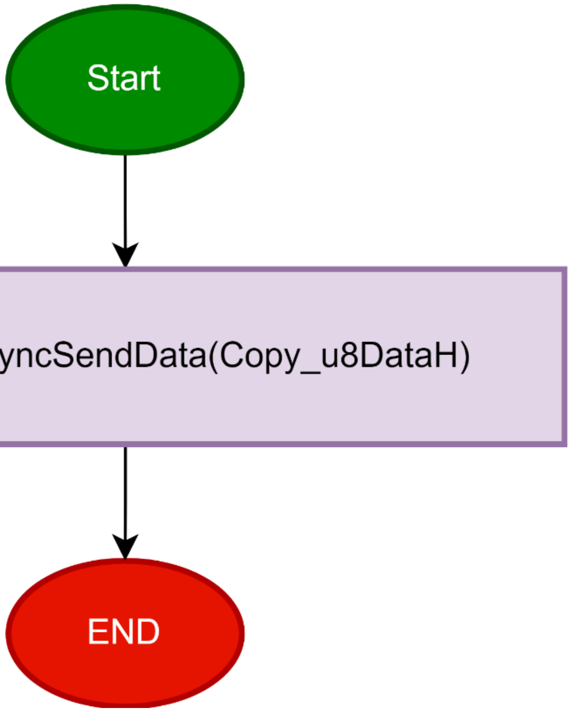
HUART_enDeInit



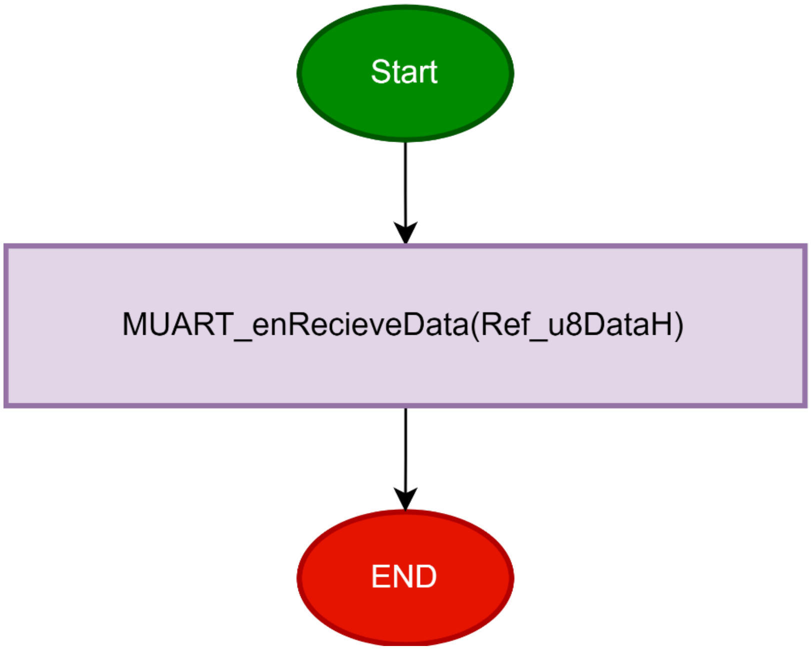
HUART_enSyncSendData



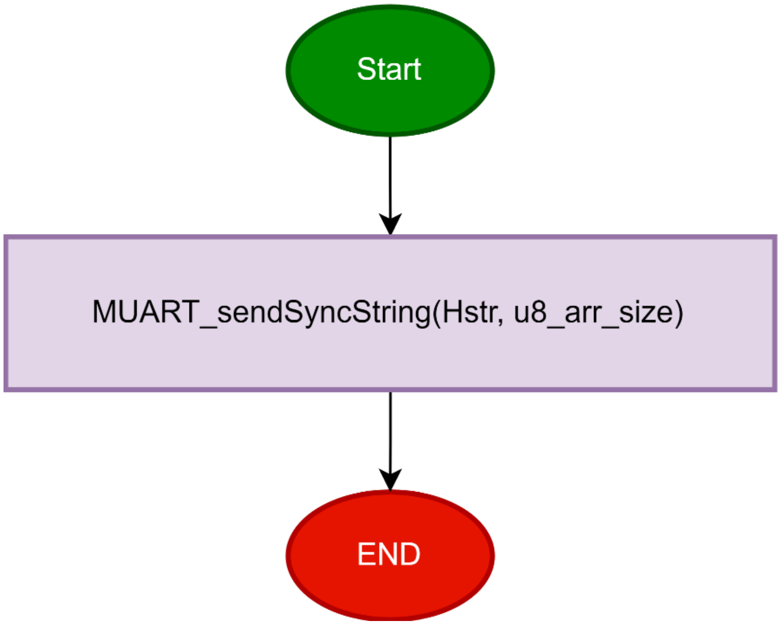
HUART_enAsyncSendData



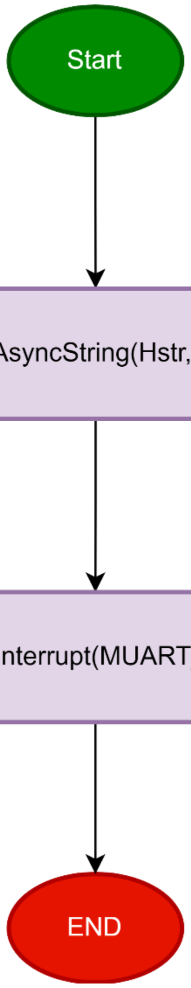
HUART_enRecieveData



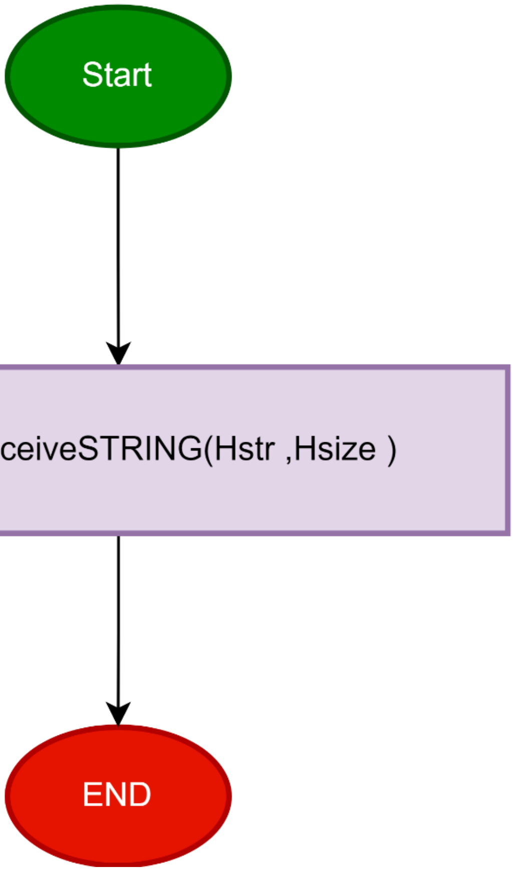
HUART_sendSyncString



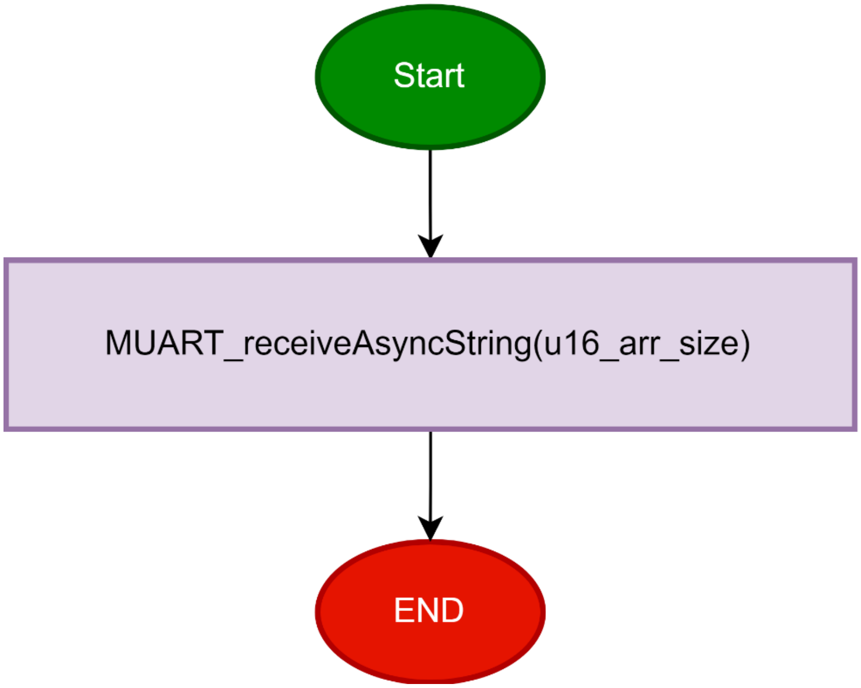
HUART_sendAsyncString


Start
MUART_sendAsyncString(Hstr, u16_arr_size)
MUART_enEnableInterrupt(MUART_TX_INTERRUPT)
END

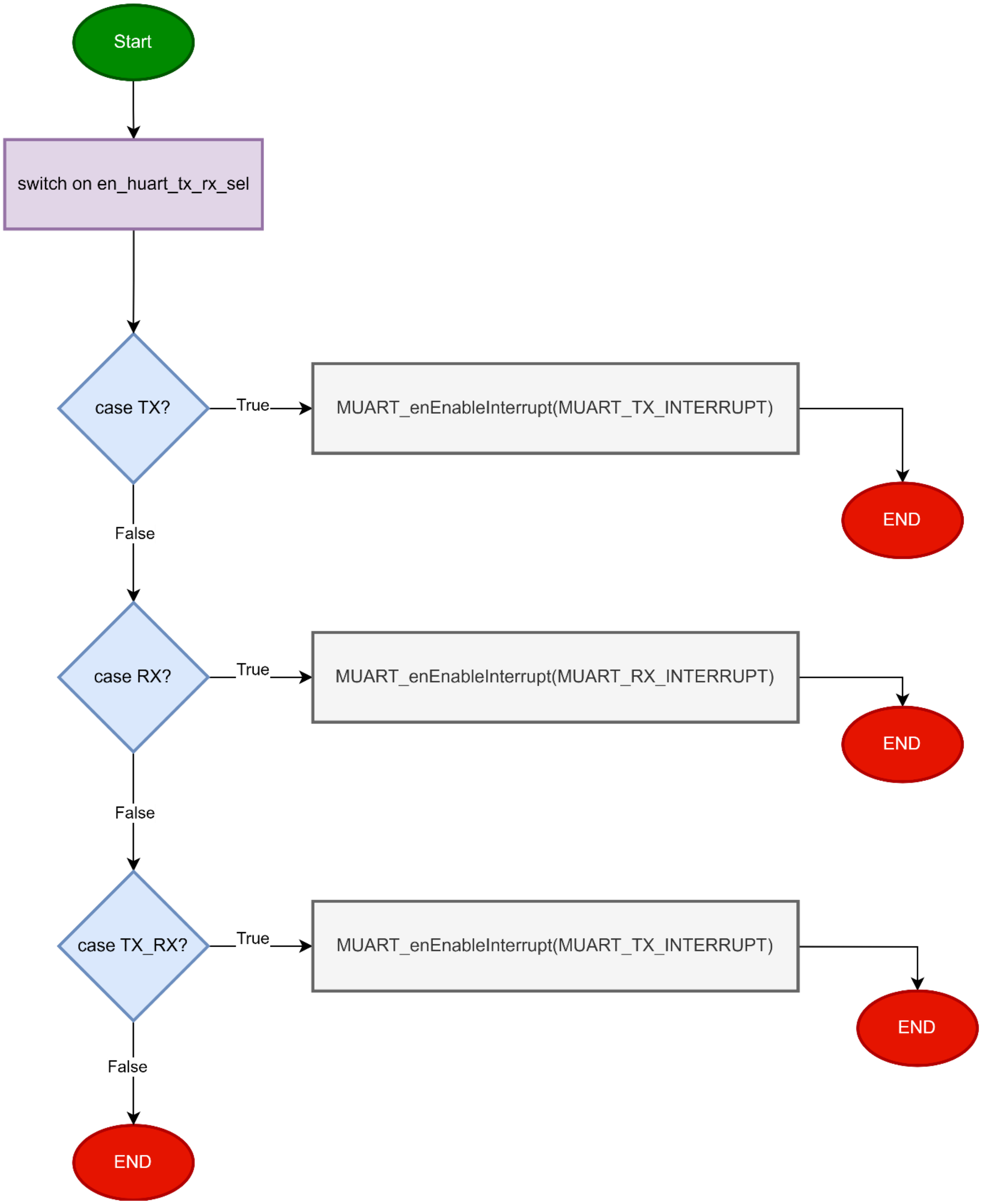
HUART_receiveSTRING



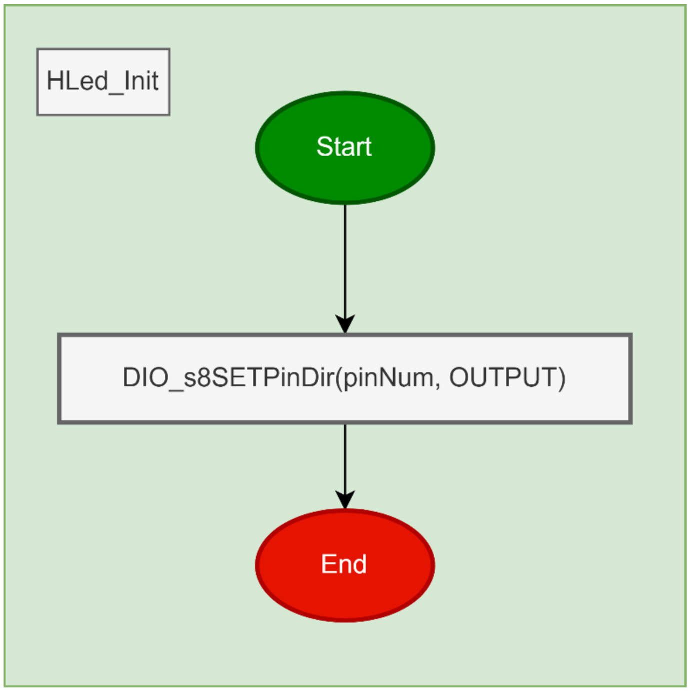
HUART_receiveAsyncString

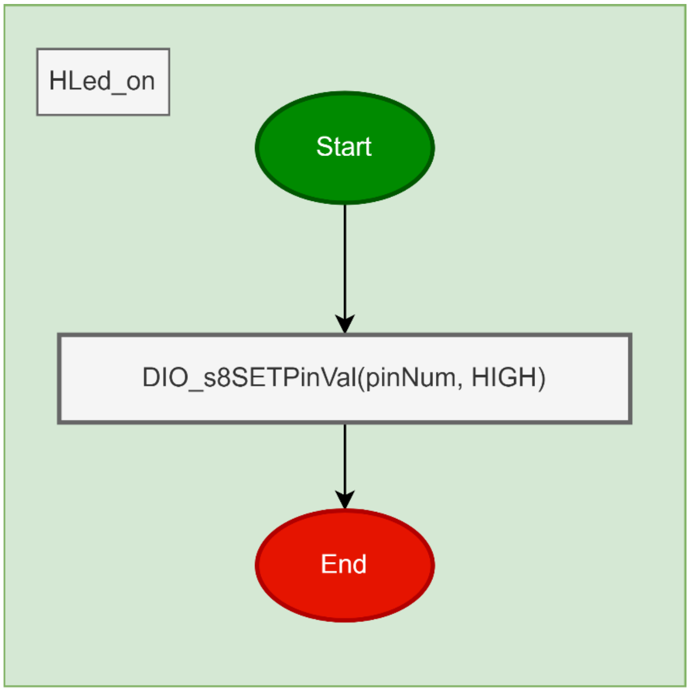


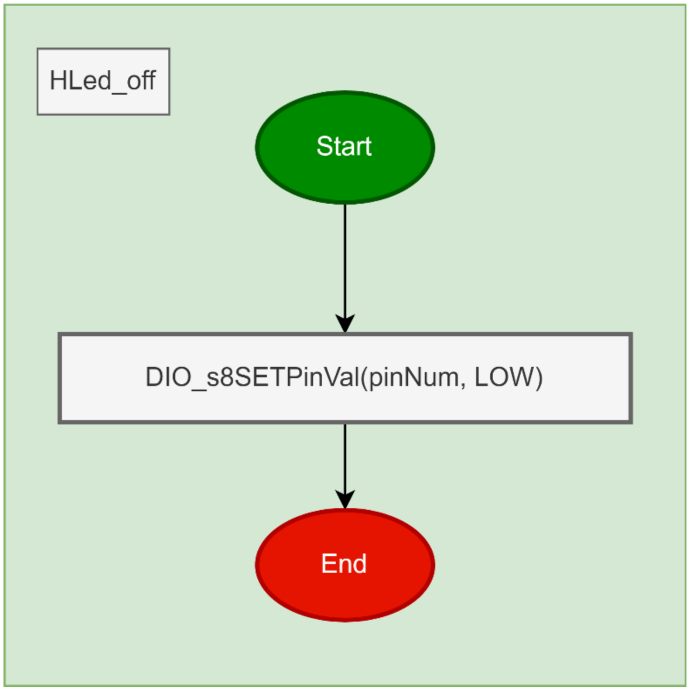
HUART_enEnableInterrupt

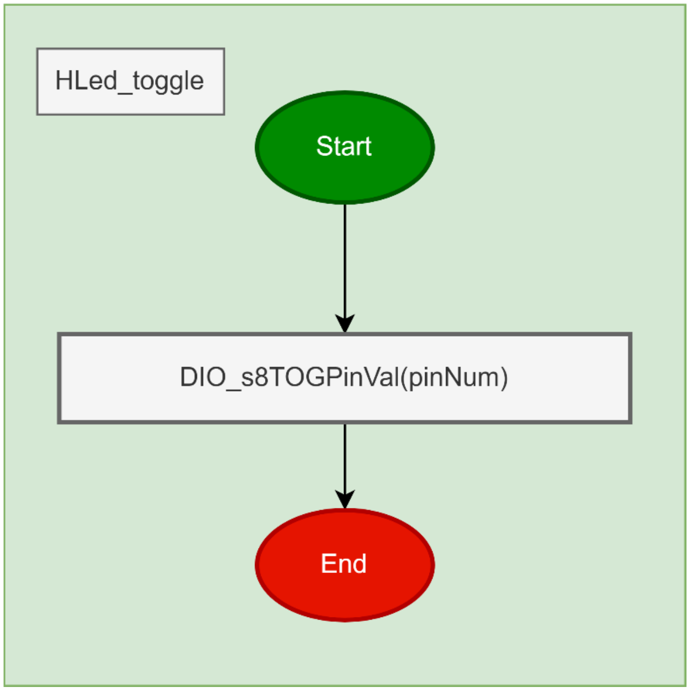

Start
switch on en_huart_tx_rx_sel
case TX?
True
MUART_enEnableInterrupt(MUART_TX_INTERRUPT)
END
False
case RX?
True
MUART_enEnableInterrupt(MUART_RX_INTERRUPT)
END
False
case TX_RX?
True
MUART_enEnableInterrupt(MUART_TX_INTERRUPT)
END
False
END

HLED module


HLed_Init
Start
DIO_s8SETPinDir(pinNum, OUTPUT)
End


HLed_on
Start
DIO_s8SETPinVal(pinNum, HIGH)
End

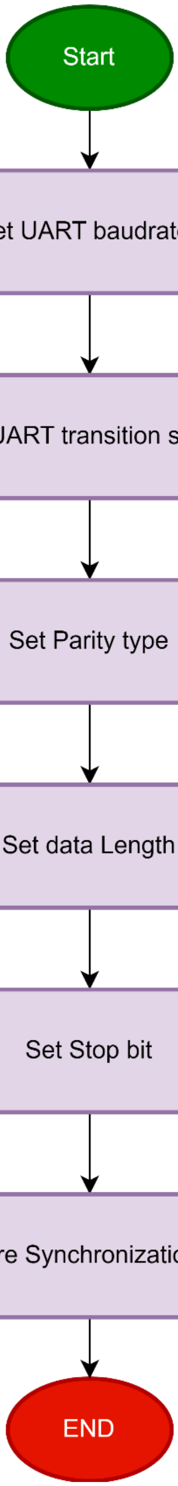

HLed_off
Start
DIO_s8SETPinVal(pinNum, LOW)
End


HLed_toggle
Start
DIO_s8TOGPinVal(pinNum)
End

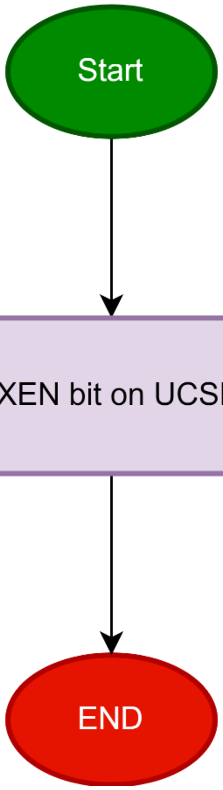
MCAL

MUART module

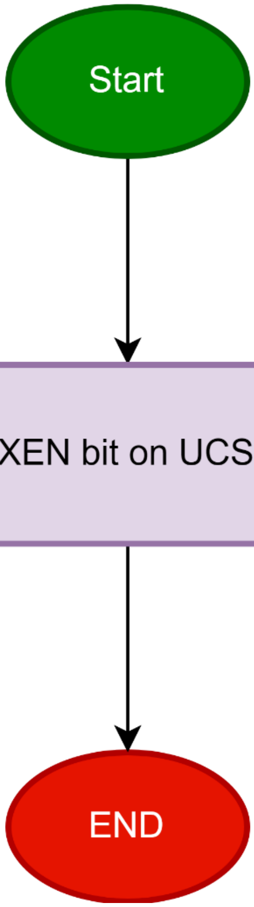
MUART_enInit



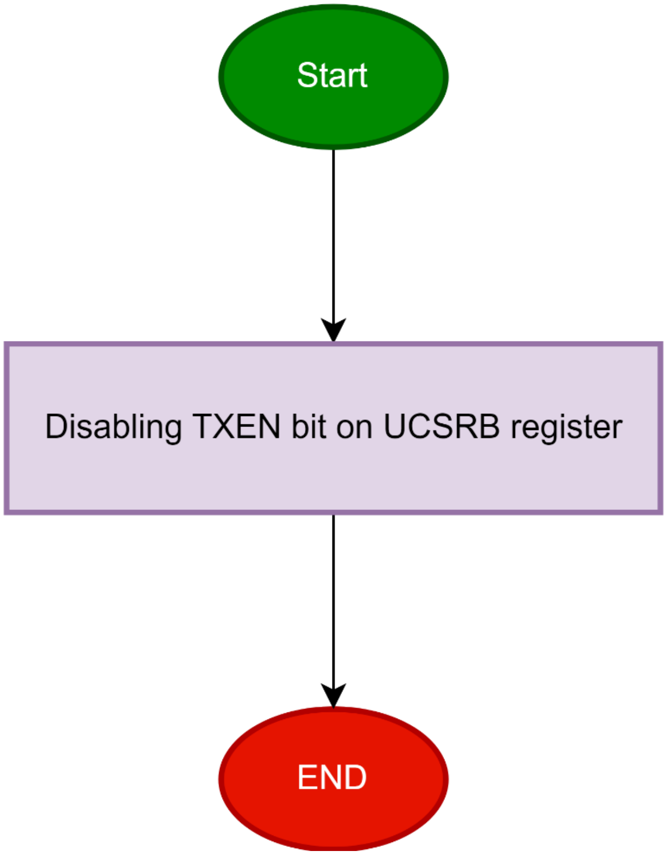
MUART_en_TX_Enable



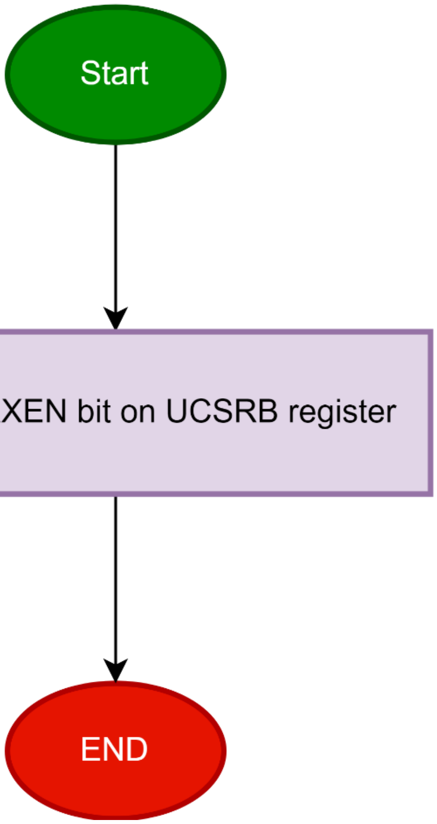
MUART_en_RX_Enable



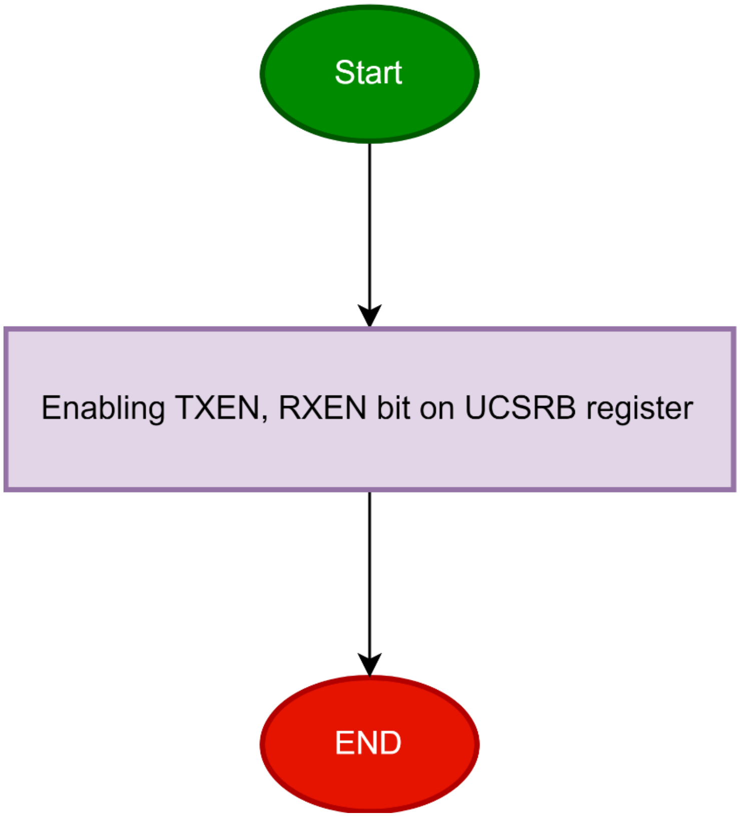
MUART_en_TX_Disable



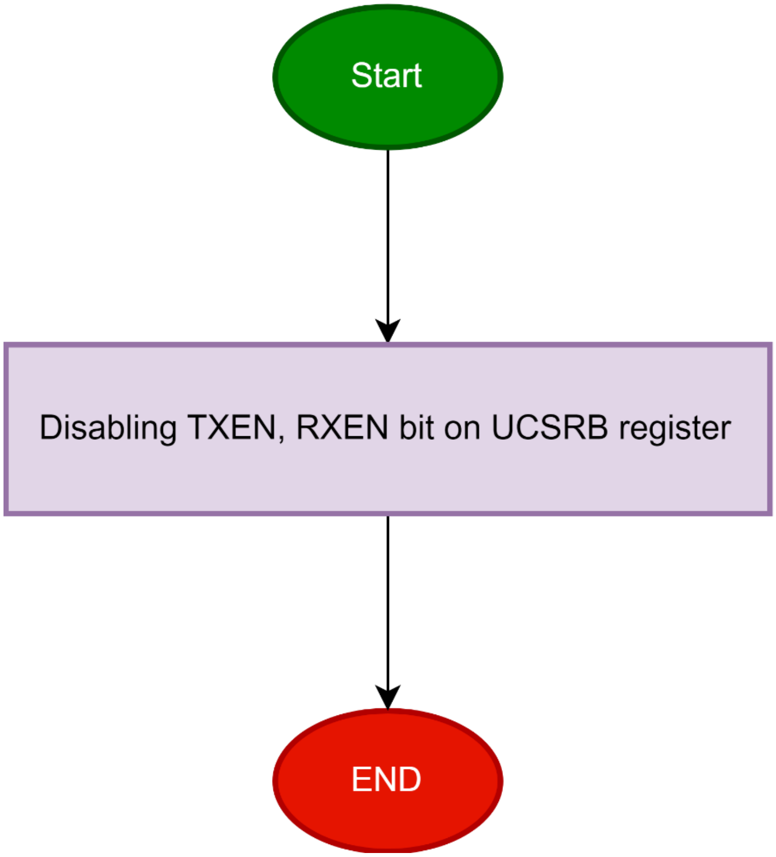
MUART_en_RX_Disable



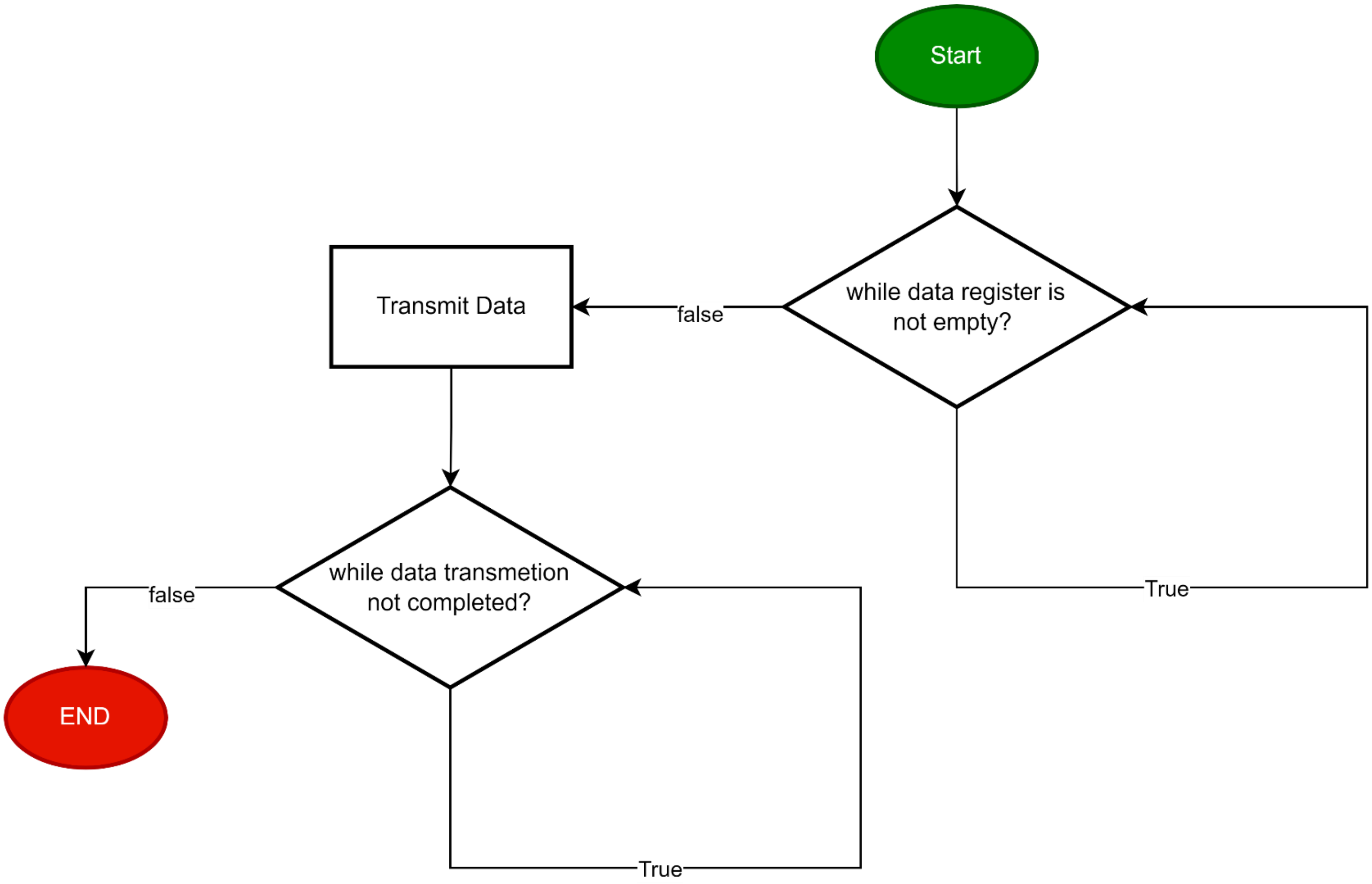
MUART_en_TX_RX_Enable



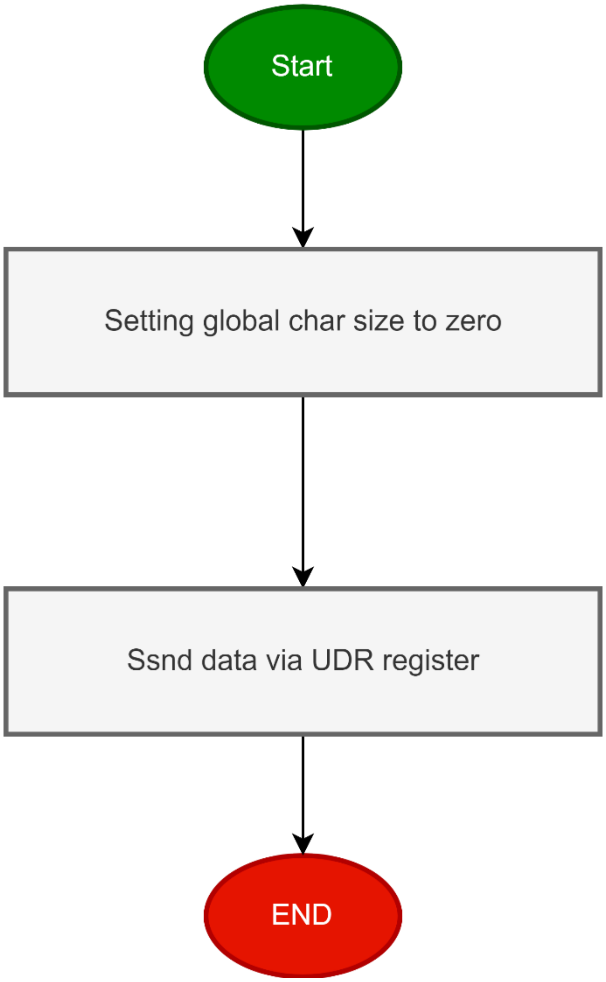
MUART_en_TX_RX_Disable



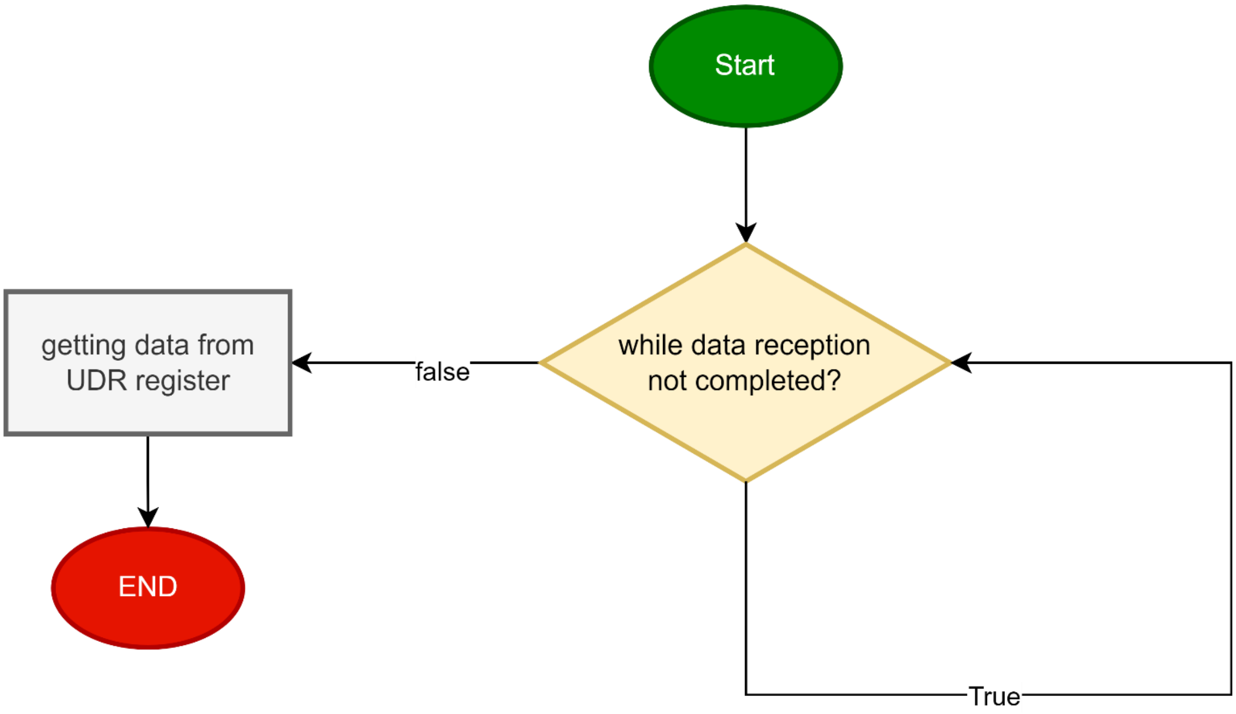
MUART_enSyncSendData


Start
while data register is not empty?
false
True
Transmit Data
while data transmetion not completed?
false
True
END

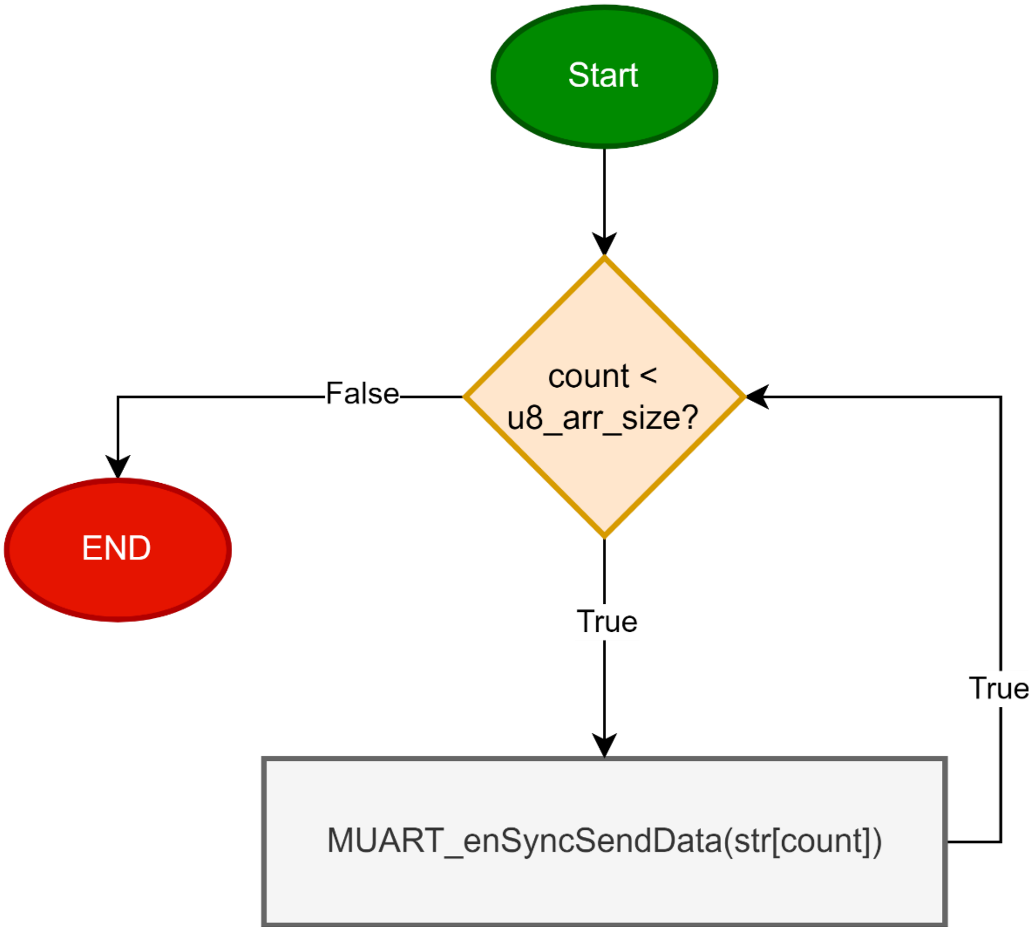
MUART_enAsyncSendData



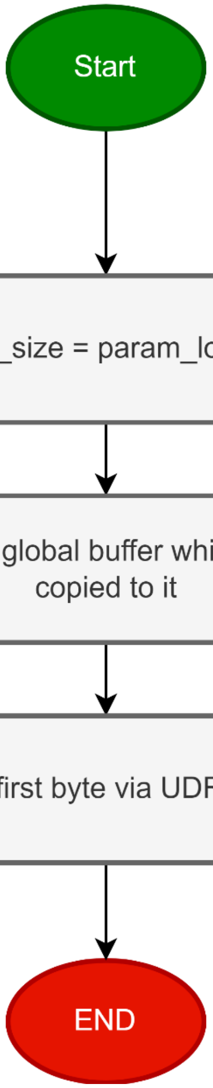
MUART_enRecieveData


Start
while data reception not completed?
false
getting data from UDR register
END
True

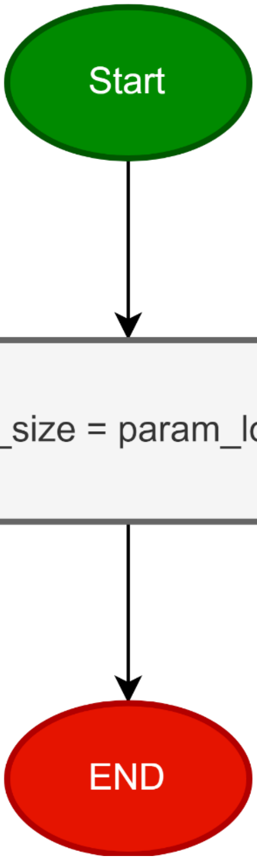
MUART_sendSyncString



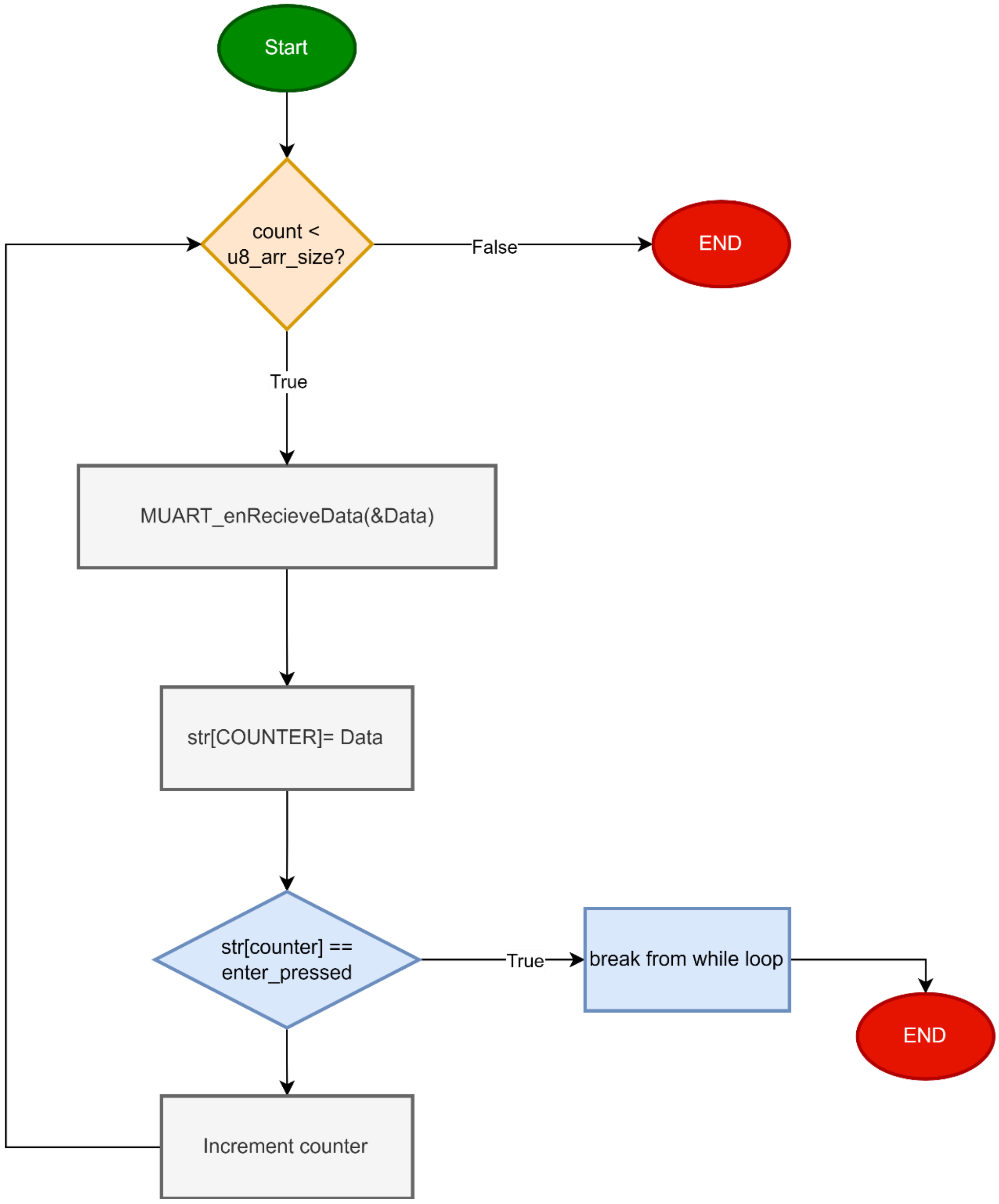
MUART_sendAsyncString


Start
global_array_size = param_local_arr_size
creating new global buffer which data to be copied to it
Transmit first byte via UDR register
END

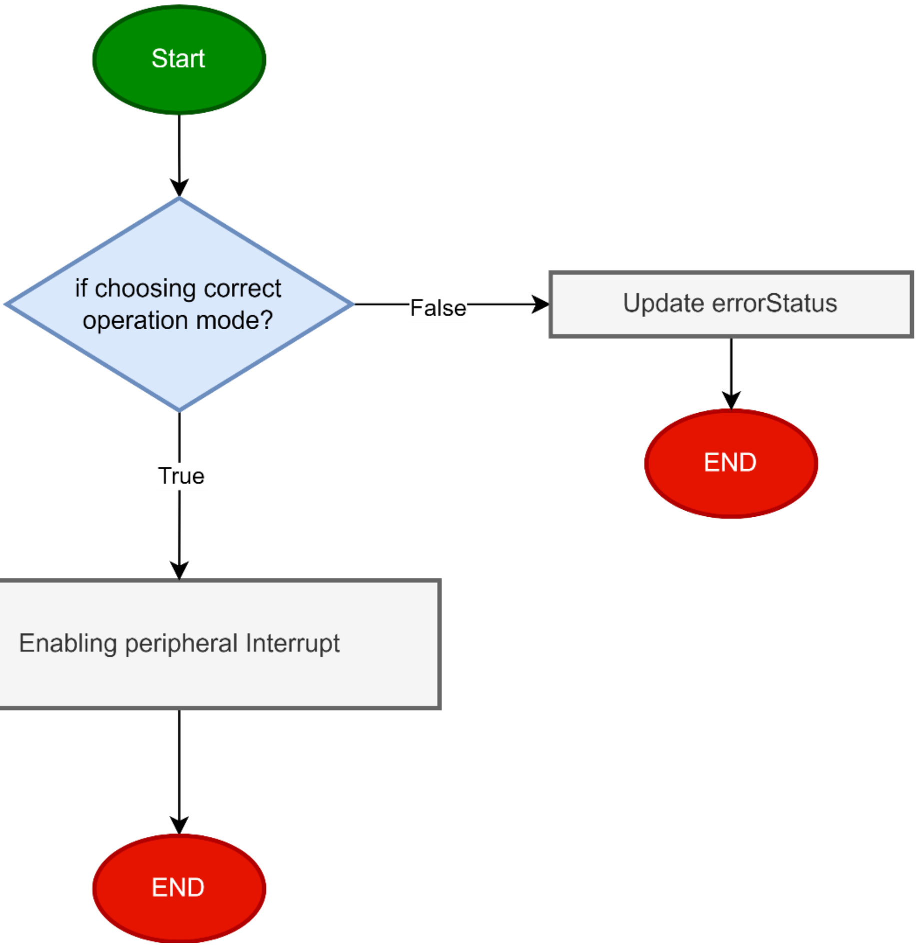
MUART_receiveAsyncString



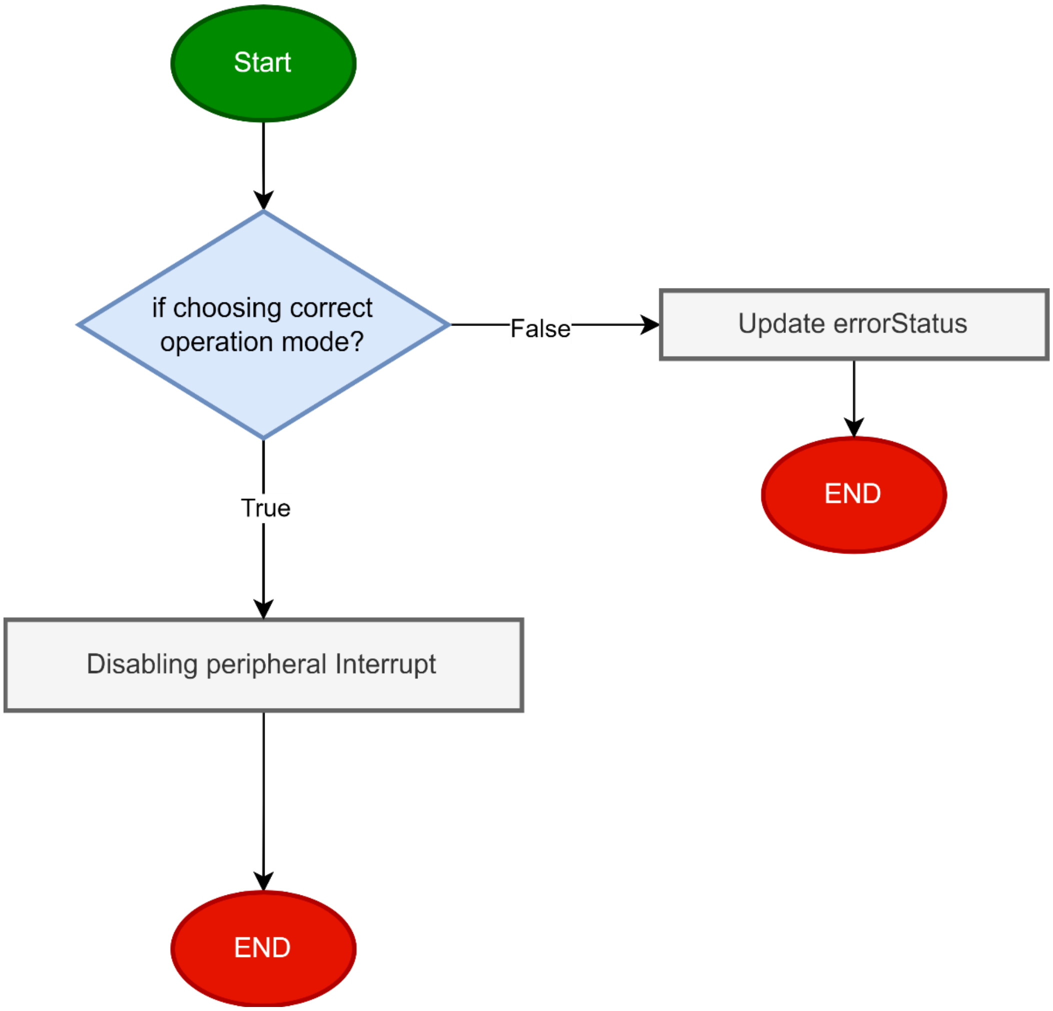
MUART_receiveSTRING



MUART_enEnableInterrupt

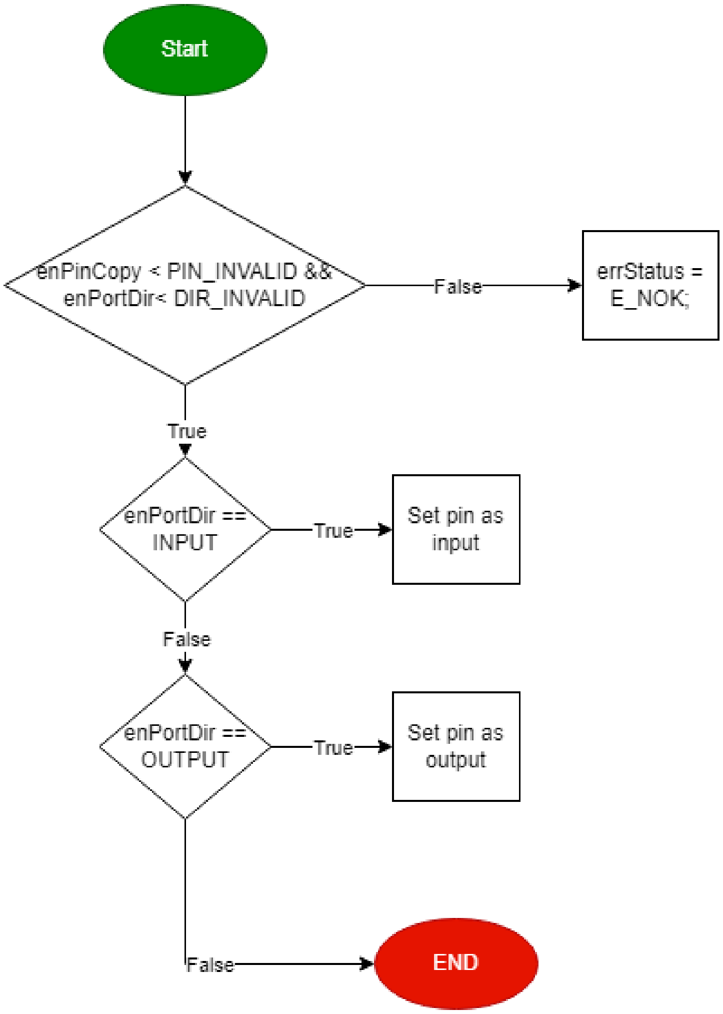

Start
if choosing correct operation mode?
False
Update errorStatus
END
True
Enabling peripheral Interrupt
END

MUART_enDisableInterrupt

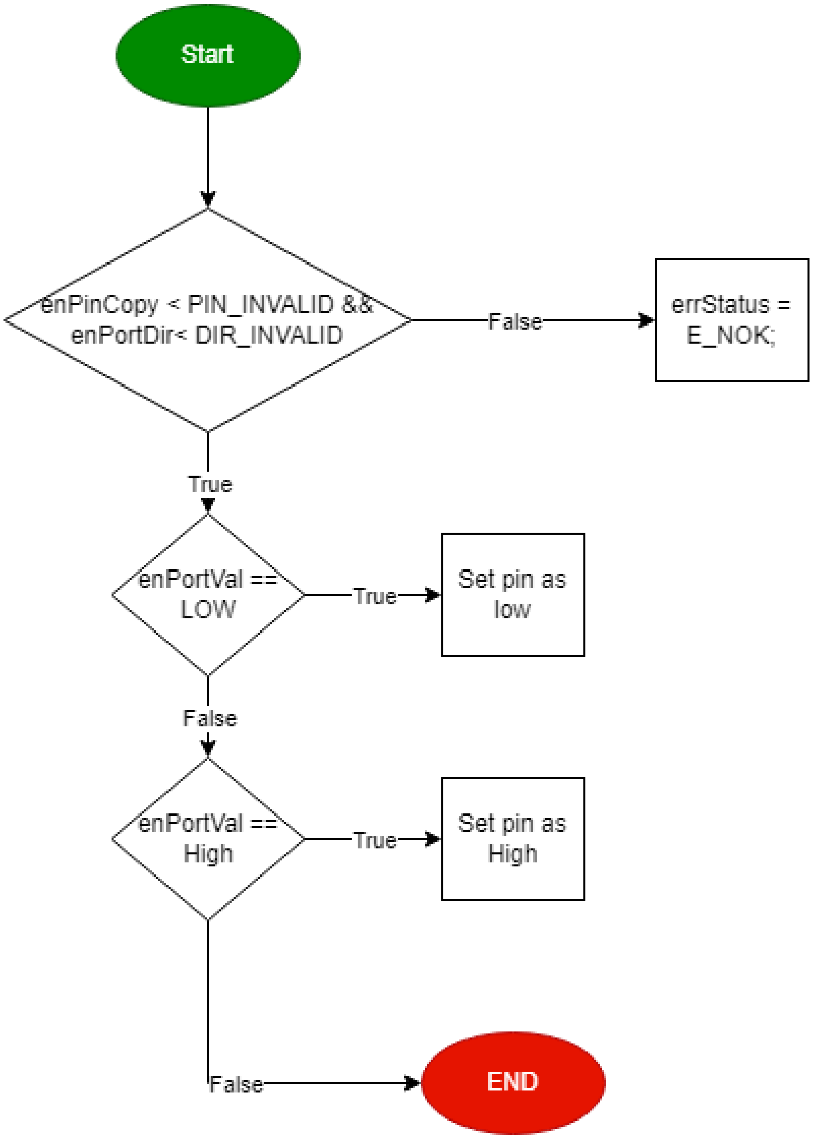

Start
if choosing correct operation mode?
False
Update errorStatus
END
True
Disabling peripheral Interrupt
END

DIO module

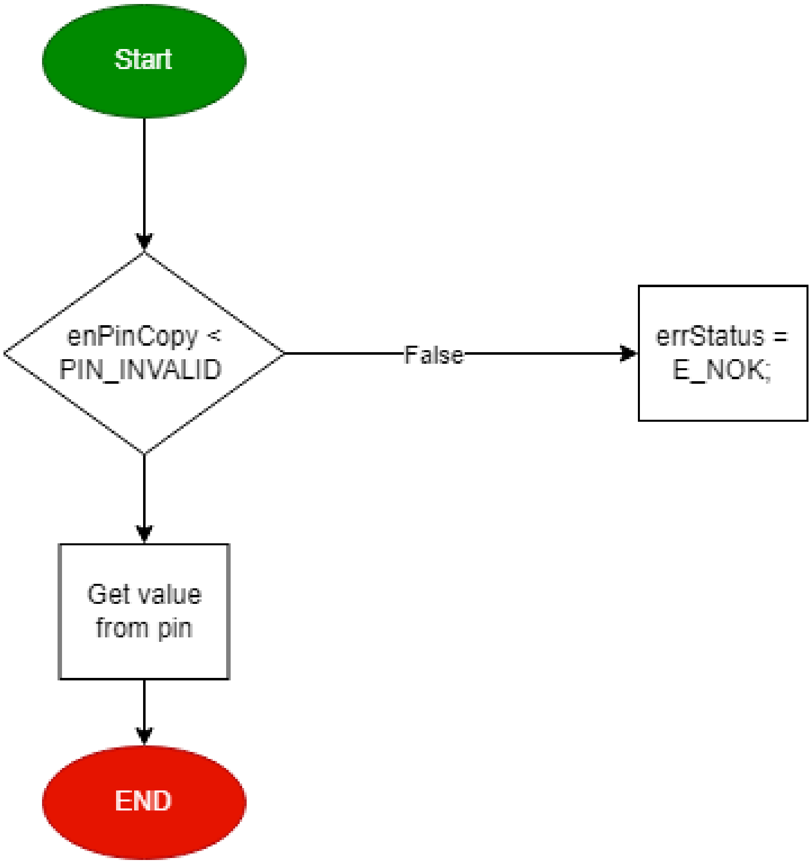
DIO_s8SETPinDir



DIO_s8SETPinVal



DIO_s8GETPinVal



Pre-compiling configuration

SERVICE

BCM module

BCM_UART_BAUDRATE

Name	BCM_UART_BAUDRATE
Type	MACRO
Range	Baudrate configuration for UART
Found in	Bcm_config.h

MCAL

MUART module

MUART_SPEED_TYPE

Name	MUART_SPEED_TYPE
Type	MACRO
Description	Configure UART Speed
Configuration	MUART_SINGLE_SPEED
	MUART_DOUBLE_SPEED
Found in	muart_config.h

MUART_TX_RX

Name	MUART_TX_RX
Type	MACRO
Description	Configure UART operation
Configuration	MUART_TX_ENABLE
	MUART_RX_ENABLE
	MUART_TX_RX_ENABLE
Found in	muart_config.h

MUART_STOP_BIT

Name	MUART_STOP_BIT
Type	MACRO
Description	Configure UART stop bit
Configuration	`MUART_1_STOP_BIT`
	`MUART_2_STOP_BIT`
Found in	muart_config.h

Linking Configuration

SERVICE

BCM module

str_bcm_instance_t

Name	str_bcm_instance_t
Type	Struct
Description	Configure BCM struct Instance
Configuration	`enu_communication_sel_t`
	`enu_tx_rx_state_t`
Found in	bcm_Interface.h

enu_communication_sel_t

Name	enu_communication_sel_t
Type	enum
Description	Choosing communication protocol
Configuration	*BCM_UART*
	BCM_SPI
	BCM_I2C
Found in	bcm_Interface.h

enu_tx_rx_state_t

Name	enu_tx_rx_state_t
Type	enum
Description	Choosing communication operation
Configuration	*BCM_TX*
	BCM_RX
	BCM_TX_RX
Found in	bcm_Interface.h

MCAL

MUART module

en_muartParity_t

Name	en_muartParity_t
Type	enum
Description	Choosing MUART parity type
Configuration	*MUART_NO_PARITY*
	MUART_PR_RESERVED
	MUART_EVEN_PARITY
	MUART_ODD_PARITY
Found in	muart_Config.h

en_muartDataLength_t

Name	en_muartDataLength_t
Type	enum
Description	Choosing data length in byte
Configuration	*MUART_FIVE_BIT_DATA*
	MUART_SIX_BIT_DATA
	MUART_SEVEN_BIT_DATA
	MUART_EIGHT_BIT_DATA
	MUART_NINE_BIT_DATA
Found in	muart_Config.h

st_muart_t

Name	st_muart_t
Type	struct
Description	Configure MUART struct Instance
Configuration	*en_muartParity_t*
	en_muartDataLength_t
Found in	muart_Config.h