Document Title	Requirements & Specification of UART Driver
Document Owner	HEX CLAN
Document Responsibility	HEX CLAN

Table of Contents

1. Introduction

The USART Communication Software Requirements Specification (SRS) outlines the functionality and behavior of the USART (Universal Synchronous/Asynchronous Receiver/Transmitter) communication code. This code is designed for microcontroller systems and facilitates serial communication over USART, supporting both synchronous and asynchronous modes. It provides configuration options for interrupt-driven and polling-based communication.

2. Purpose

The purpose of this SRS is to describe the functionality of the USART communication code, including initialization, data transmission, and data reception. It also outlines the available options for configuring and customizing the USART communication.

3. Scope

The USART communication code includes functions for initializing USART, sending and receiving data, sending numbers, sending strings, and setting a callback function to handle received data. It is intended for microcontroller systems with an 8MHz system clock and can be adapted to different hardware configurations.

4. Definitions

- **USART**: Universal Synchronous/Asynchronous Receiver/Transmitter, a communication interface used for serial data transfer.

- **Baud Rate**: The data transmission rate in bits per second (bps).

- **Interrupt**: An event that interrupts the normal program flow to handle specific tasks or events.

- **Callback Function**: A user-defined function that is registered to be called in response to a specific event or condition.

5. Requirements

5.1. Initialization

Service Name	USART_voidInit
Syntax	void USART_voidInit(void);
Sync/Async	Configurable via USART_config.h
Reentrancy	Non-reentrant
Parameters (in)	none
Parameters (inout)	none
Parameters (out)	none
Return value	none
Description	The `USART_voidInit` function initializes USART communication. It configures various parameters, such as interrupt settings (if required), data frame format, and baud rate.
Available via	USART_interface.h

5.2. Receive Data

Service Name	USART_u8Receive
Syntax	u8 USART_u8Receive(void);
Sync/Async	Configurable via USART_config.h
Reentrancy	Non-reentrant
Parameters (in)	none
Parameters (inout)	none
Parameters (out)	none
Return value	UDR Value
Description	The `USART_u8Receive` function waits until data is received in the receive buffer and returns the received 8-bit data.
Available via	USART_interface.h

5.3. Send Data

Service Name	USART_voidSend
Syntax	void USART_voidSend(u8 Copy_u8data);
Sync/Async	Configurable via USART_config.h
Reentrancy	Non-reentrant
Parameters (in)	U8 data
Parameters (inout)	none
Parameters (out)	none
Return value	none
Description	The `USART_voidSend` function sends an 8-bit data character over USART. It waits until the transmit buffer is empty and then sends the data.
Available via	USART_interface.h

5.4. Send Numbers

Service Name	UART_voidSendSingleNumber
Syntax	void UART_voidSendSingleNumber(u8 Copy_u8number);
Sync/Async	Configurable via USART_config.h
Reentrancy	Non-reentrant
Parameters (in)	Copy_u8number
Parameters (inout)	none
Parameters (out)	none
Return value	none
Description	- `UART_voidSendSingleNumber` sends a single-digit number as a character.
Available via	USART_interface.h

Service Name	UART_voidSendNumber
Syntax	void UART_voidSendNumber(u32 Copy_u8number);
Sync/Async	Configurable via USART_config.h
Reentrancy	Non-reentrant
Parameters (in)	u32 Copy_number
Parameters (inout)	none
Parameters (out)	none
Return value	none
Description	- `UART_voidSendNumber` sends multi-digit numbers by breaking them down into individual digits and sending them as characters.
Available via	USART_interface.h

5.5. Send String

Service Name	USART_voidSendString
Syntax	void USART_voidSendString(u8 *str);
Sync/Async	Configurable via USART_config.h
Reentrancy	Non-reentrant
Parameters (in)	u8 *str
Parameters (inout)	none
Parameters (out)	none
Return value	none
Description	The `USART_voidSendString` function sends a string (character array) over USART. It iterates through the characters in the string and sends them one by one.
Available via	USART_interface.h

5.6. Receive String

Service Name	UARTGetString
Syntax	void UARTGetString(u8 *str);
Sync/Async	Configurable via USART_config.h
Reentrancy	Non-reentrant
Parameters (in)	u8 *str
Parameters (inout)	none
Parameters (out)	none
Return value	none
Description	The `UARTGetString` function receives a string until a newline character '\n' is encountered. It stores the received characters in the provided character array.
Available via	USART_interface.h

5.7. Set Receive Callback

Service Name	USART_voidRecieveSetCallBack
Syntax	void USART_voidRecieveSetCallBack(void (*Copy_pvRecieveFunc)(u8 copy_u8received));
Sync/Async	Configurable via USART_config.h
Reentrancy	Non-reentrant
Parameters (in)	(*Copy_pvRecieveFunc)(u8 copy_u8received)
Parameters (inout)	none
Parameters (out)	none
Return value	none
Description	The `USART_voidRecieveSetCallBack` function allows external code to set a callback function to be executed when data is received. The provided function pointer will be called when data is received via USART.
Available via	USART_interface.h

6. Interrupt Service Routine

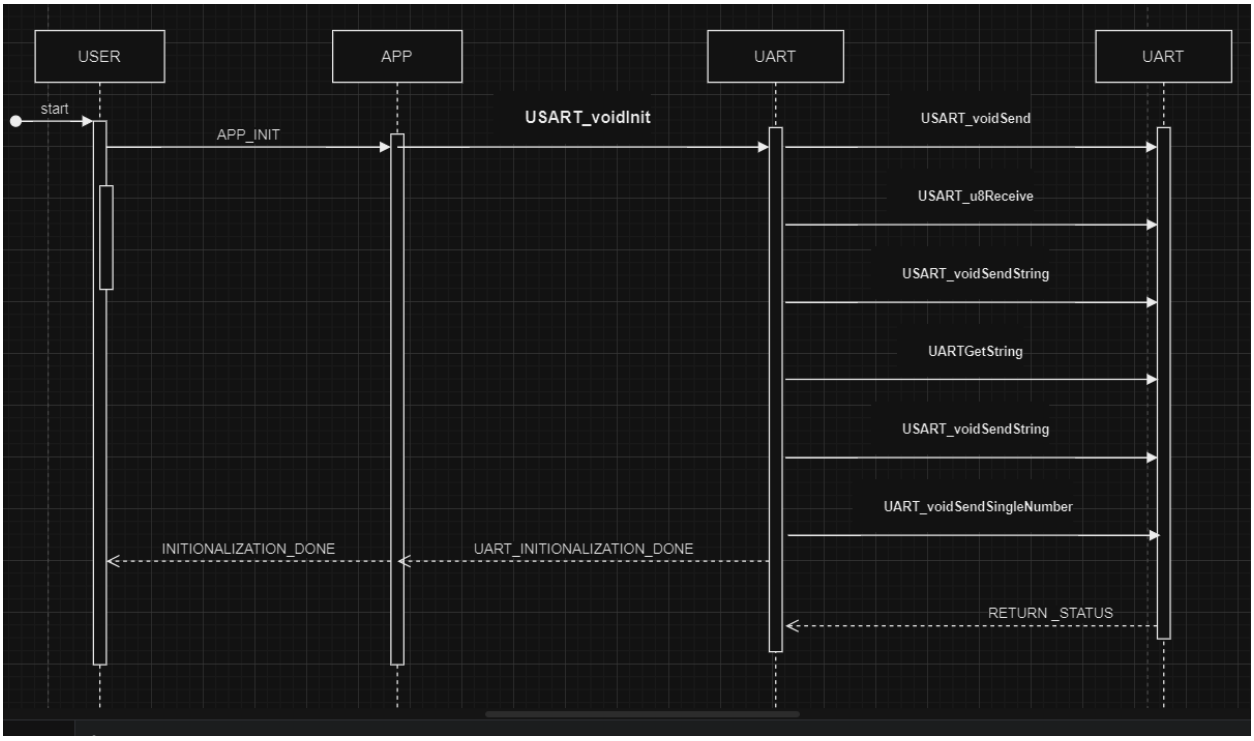
6.1. ISR Function

void __vector_13(void) __attribute__((signal));

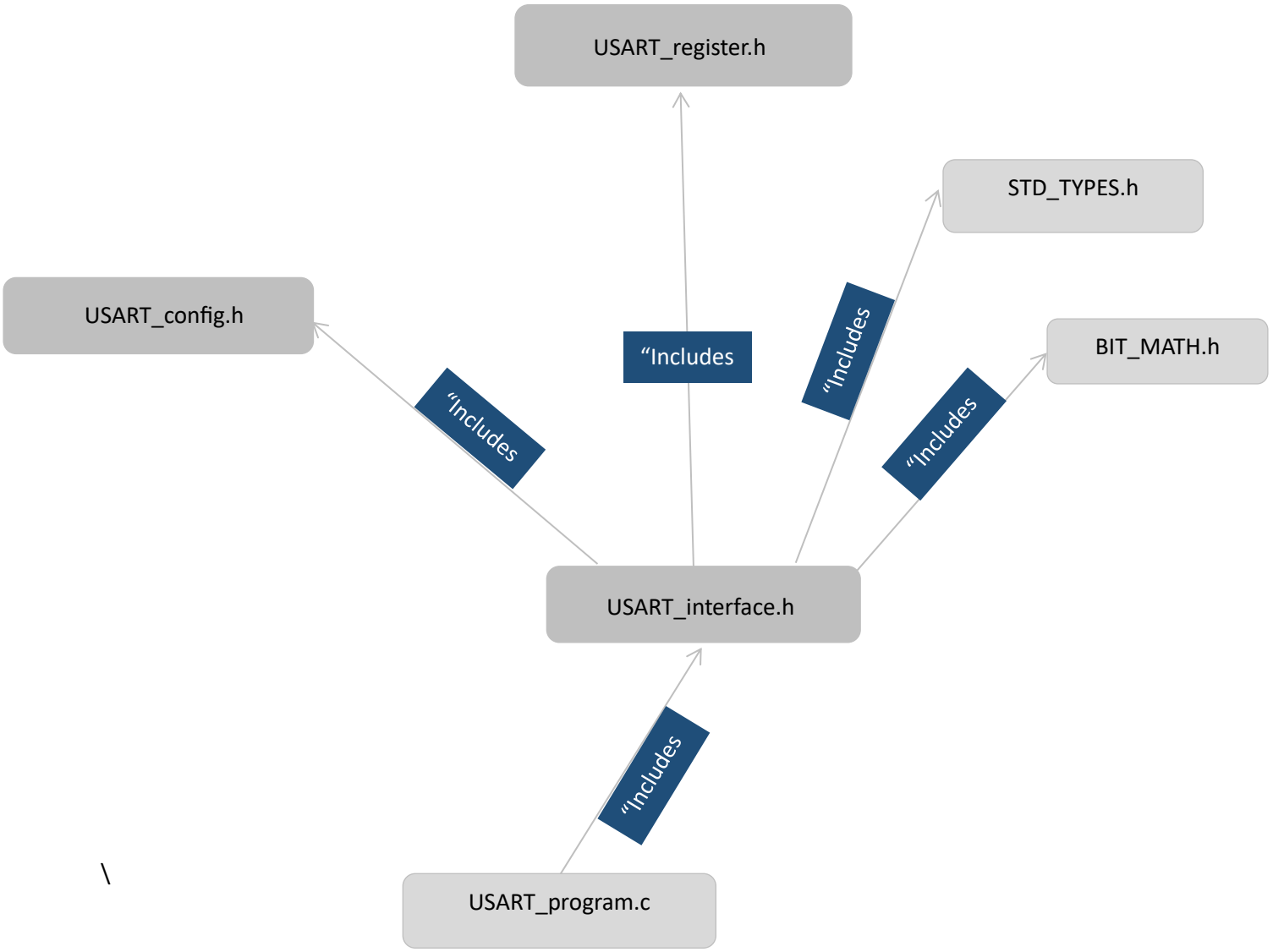
6.2. Description

This is the USART Receive Interrupt Service Routine (ISR). It is called when data is received via USART. If a callback function is registered, it calls the callback function and passes the received data.

7. Sequence Diagram



File Structure



\

9. Conclusion

The USART Communication code outlined in this Software Requirements Specification provides essential functionality for serial communication in microcontroller systems. It supports both interrupt-driven and polling-based communication, making it a versatile solution for transmitting and receiving data over USART. Users can customize and configure the code to suit their specific hardware requirements.