STM32L475 Embedded Driver Development

Yaoctzin Serrato

September 17, 2020

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Contents

USART driver **iii**

USART driver

UART stands for Universal Asynchronous Receiver Transmitter, whereas USART stands for Universal Synchronous Asynchronous Receiver Transmitter. They are basically a piece of hardware that converts parallel data to serial data. The only difference is that UART supports only Asynchronous mode, whereas USART supports both synchronous and asynchronous modes.
Unlike Ethernet or USB, there is no specific port for UART/USART communication. They are commonly used in conjunction with protocols such as RS232, RS434, etc.
In synchronous transmission the clock signal is sent separately from the data stream and no start/stop bits are used. If it is asynchronous mode, then the clock signal is not sent but instead, synchronization bits will be used: start and stop bits, besides the data stream.

USART hardware components

Typically, USART hardware will have the following components:

- Baudrate generator
- TX/RX shift registers
- Transmit/Receive control blocks
- Transmit/Receive buffers
- First-in First-out (FIFO) buffer memory

USART pins

USART bidirectional communication requires the following pins:

- **TX - Transmit (required)**: USART module transmits data over TX pin. If nothing is beaing transmitted, then the TX line will remain HIGH, which is the idle state of TX line.

- **RX - Receive (required)**: USART module receives data over RX pin. The module continuously samples RX line to detect the start bit of an incoming frame.
- **RTS - Request to Send (optional)**: RTS pin is optional for basic communication set up, but required if hardware flow control is used. RTS is an active-LOW pin. This pin is used by the USART module to inform an external device that new data is needed (RTS pin is then pulled to LOW).
- **CTS - Clear to Send (optional)**: CTS pin is optional for basic communication set up, but required if hardware flow control is used. CTS is an active-LOW pin. When hardware flow control is used, the data transmision on the TX line happens only if the CTS pin is pulled LOW. Otherwise, the data transmision will remain inactive. CTS pin has to be pulled LOW by another device in order to enable the data transmision over TX pin.

RX pin of one device is connected to TX pin of another device. The same happens with CTS and RTS pins: RTS pin of one device is connected to CTS pin of another device. If device A wants data from device B, and hardware flow control is being used, device A will pull LOW its RTS pin which in turn will pull LOW the CTS pin of device B and the data frame will be transmitted over TX line of device B and received over RX line of device A.

Image of USART pins (Microsoft VISIO)

USART frame format

A frame refers to the entire data packet which is being sent or received during the communication. The formats of the data packet vary from protocol to protocol. The following image shows the frame format of USART packets.

Image of USART frame formats for 9-bit and 8-bit word lengths (Microsoft VISIO)

USART frame commences with a LOW start bit of 1-bit duration. Then follows data bits from LSB to MSB (although this can be configured to start with MSB and finish with LSB) and this bit stream has a length of 5 to 9 bits. The data bits are followed by the parity bit, which is an optional bit. If parity bit is used, it will occupy one bit from the data stream, which means, for a 9-bit packet 8 bits will be of data plus the parity bit; for an 8-bit packet 7 bits will be of data plus the parity bit. Parity can be either odd or even. Finally, a frame ends with an stop bit. Stop bit is always HIGH and can be configured for 1, 1.5 or 2 bit duration.

USART baudrate

The significance of baudrate is how fast the data is sent over a serial line. It is usually expressed in units of bits-per-second [bps]. If the baudrate value is inverted, the result is the time a single bit takes to be transmitted. For example, a bit takes 104[us] to be transmitted with a baudrate of 9600[bps]. Both transmitting and receiving devices should

operate at the same rate to have a proper communication. The higher the baudrate goes, the faster the data is sent or received. The baudrates are usually depending on the peripheral clock frequency of the USART peripheral.

USART synchronization bits

The synchronization bits are 2 to 3 bits that are transfered in each frame of data. These bits are the start bits and stop bits and they mark the begining and end of a packet, respectively. There is always 1 start bit but stop bits are configurable to 1, 1.5 or 2 bits. Typically, 1 stop bit is used, however, if the baud rate is high (in MHz), then it is recommended to use 2 stop bits.
The start bit is always indicated by an idle data line going from HIGH to LOW, while the stop bits will transition back to the idle state (HIGH value). The receive engine of the USART module is capable enough for catching these transitions using over sampling techniques.

Image of USART frame formats showing start and stop bits (Microsoft VISIO)

USART parity bit

Adding a parity bit is the simplest method of error detection. Parity is simply the number of ones (1) appearing in the binary form of a number. For example, the binary representation of the decimal value 55 is 0b00110111, which has 5 ones (an odd number). There are two options in parity selection:

- **Odd parity**: Parity bit is set to 1 in case the number of ones in the data stream is an even number. Parity bit is left as 0 if the number of ones in the data stream is an odd number.
- **Even parity**: Parity bit is set to 1 in case the number of ones in the data stream is an odd number. Parity bit is left as 0 if the number of ones in the data stream is an even number.

USART transmitter

The heart of the transmitter is the Transmit Shift Register where parallel data is converted to serial data. The shift register obtains the data to be transmitted from the Transmit Data Register (TDR). The new data is not loaded to the shift register until the stop bit has been transmitted from the previous packet. As soon as the stop bit is transmitted, the shift register is loaded with new data from TDR.

Image of the USART block diagram.

Typically, a USART transmission would require the following configuration (irrespective of the MCU used):

- Specify the word length.

- Specify the number of stop bits.
- Select the desired baudrate (before this selection the USART peripheral clock value must be known because such frequency puts a limit on the maximum baudrate that can be generated).
- Enable the Transmit Block.
- Enable the USART module.
- If the corresponding flag is ready, write the data to send in the TDR. This step must be repeated for each data packet to be transmitted.
- After writing the last data packet, wait until the Transmission Complete flag is ready in order to disable the USART module.

USART receiver

During a USART reception the data shifts in into the Receive Shift Register. After the USART receive engine detects the stop bit of the packet frame, the data within the shift register is transferred to the Receive Data Register (RDR). The heart of the receive engine of the USART module is the Receive Shift Register, where the serial data is converted to parallel data (arranged within a register).

Image of the USART block diagram.

Typically, a USART reception would require the following configuration (irrespective of the MCU used):

- Specify the word length.
- Specify the number of stop bits.
- Select the desired baudrate (before this selection the USART peripheral clock value must be known because such frequency puts a limit on the maximum baudrate that can be generated).
- Enable the Receiver Block.
- Enable the USART module.
- Once the Receiver Block is enebaled, it will start searching for the start bit of an incoming data frame. When a character is received, the data can be read from the RDR only after the corresponding indicative flag is ready. This flag would indicate that a complete frame has been received and can now be read.

There are 2 oversampling options given in the STM32 MCUs: 1) Oversampling by 8 and 2) oversampling by 16. When oversampling by 8 is used, it samples the RX line 8 times the UART Peripheral clock frequency; and when the oversampling by 16 is used, it samples the RX line 16 times the periphral clock frequency to find out the start bit.