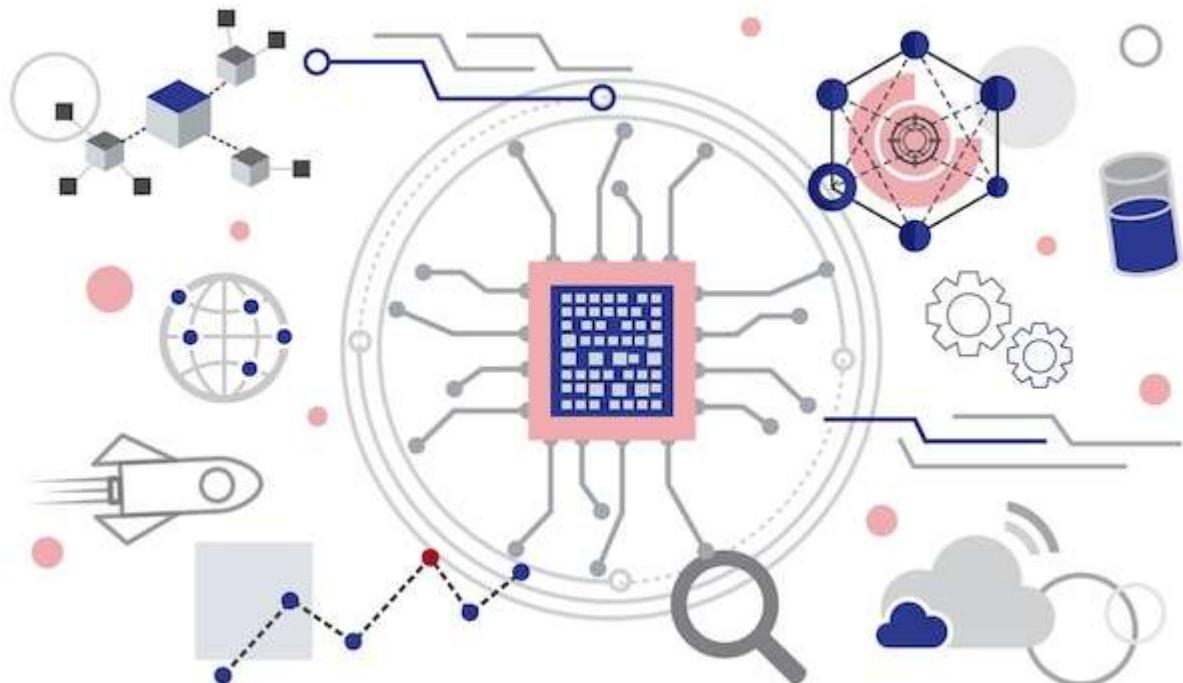




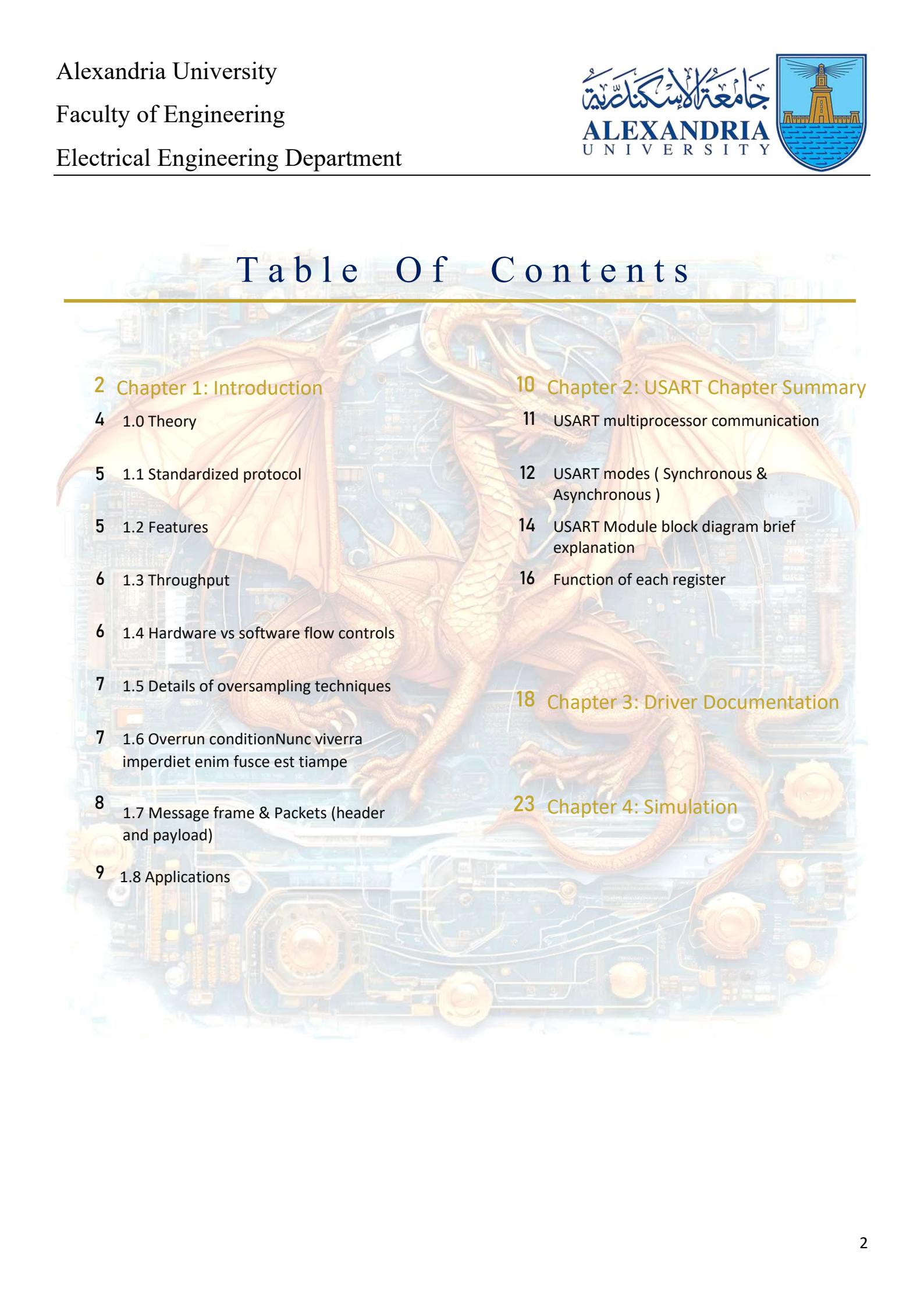
Communication Protocols

"Model_3_USART"



| ID | NAME | ID | NAME |
|----------|------------------------------|----------|-------------------------|
| 20010058 | احمد ايمن عبد الحميد قديل | 20010513 | خالد احمد زكي سعد |
| 20011101 | كريم احمد عبد المجيد | 20011582 | محمد سامر محمد |
| 20010852 | عبدالرحمن محمد محمود | 20011545 | محمد حسين مصطفى احمد |
| 20012254 | يوسف ايمن السيد عبد النبي | 20010732 | شهاب اشرف محمد |

Table Of Contents

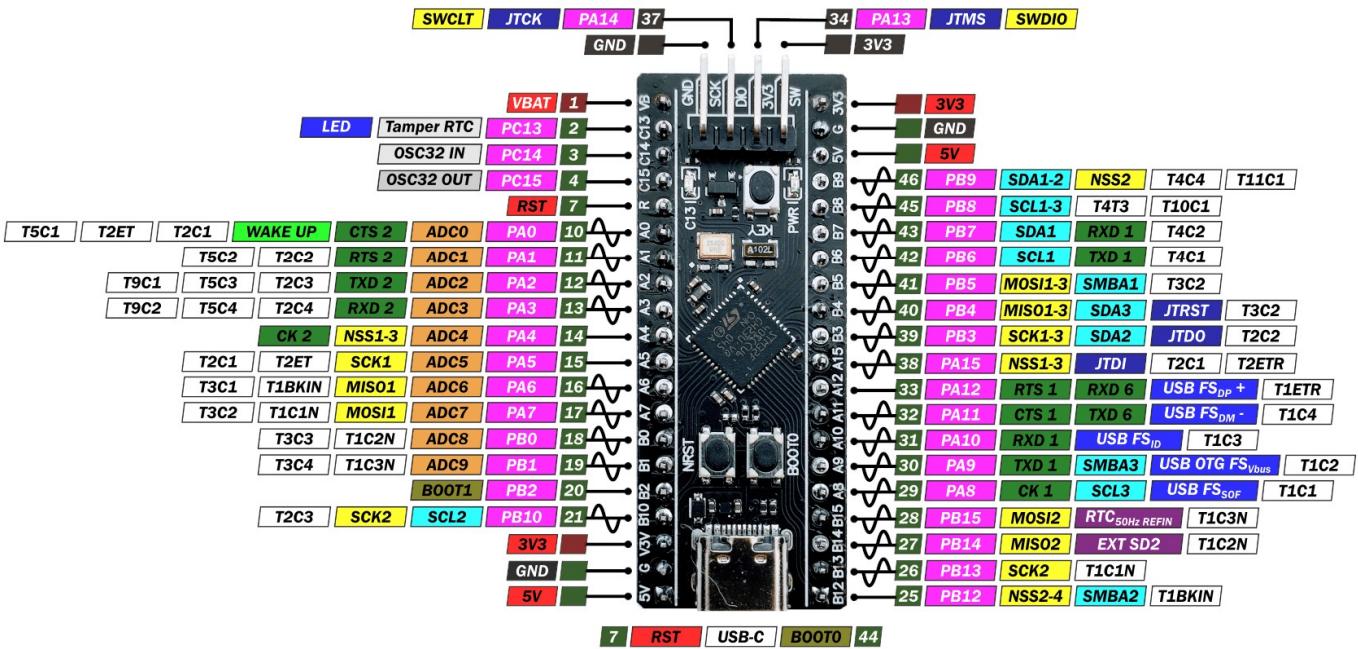
- 
- 2 Chapter 1: Introduction
 - 4 1.0 Theory
 - 5 1.1 Standardized protocol
 - 5 1.2 Features
 - 6 1.3 Throughput
 - 6 1.4 Hardware vs software flow controls
 - 7 1.5 Details of oversampling techniques
 - 7 1.6 Overrun conditionNunc viverra imperdiet enim fusce est tiamp
 - 8 1.7 Message frame & Packets (header and payload)
 - 9 1.8 Applications
 - 10 Chapter 2: USART Chapter Summary
 - 11 USART multiprocessor communication
 - 12 USART modes (Synchronous & Asynchronous)
 - 14 USART Module block diagram brief explanation
 - 16 Function of each register
 - 18 Chapter 3: Driver Documentation
 - 23 Chapter 4: Simulation

Chapter_1_Introduction

In our project we will discuss interfacing one of the most popular and modern microcontrollers into embedded systems STM32F401CC (Black Pill).

WeAct STM32F401

PINOUT



A USART (universal synchronous/asynchronous receiver/transmitter) is hardware that enables a device to communicate using serial protocols. It can function in a slower asynchronous mode, like a universal asynchronous receiver/transmitter (UART), or in a faster synchronous mode with a clock signal. USARTs are no longer common in consumer PCs but are still used in industrial equipment and embedded systems.

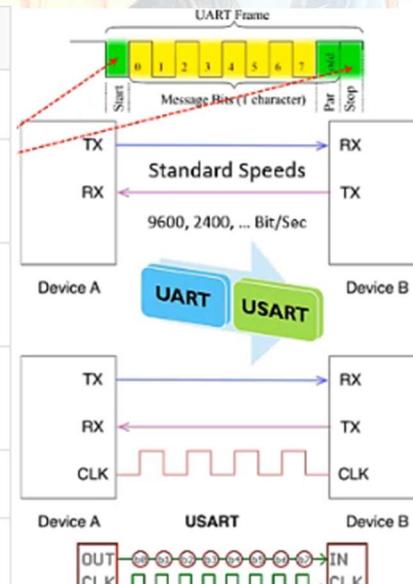
USART vs. UART

A UART device can use asynchronous communication protocols. A USART device can use both asynchronous and synchronous communication protocols. Therefore, a USART can do anything a UART can do and more. Because a USART requires more complex circuitry and more communication lines to fully implement, many devices may only implement a UART to save on cost, complexity or power usage.

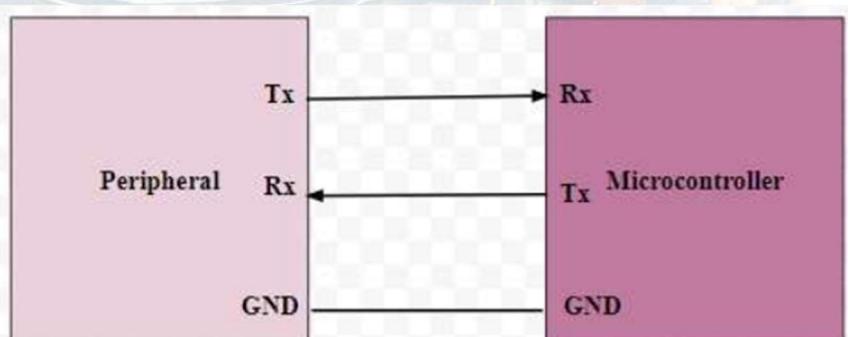
Asynchronous and synchronous serial communication

In serial communication, each bit of data is sent one at a time on a transmit wire. This is a serial communications interface. If the sender and the receiver don't agree on how the data is sent, such as the order and length of time of each bit, then the data becomes garbled, and they won't understand each other. Asynchronous and synchronous are two different ways to standardize how serial data is sent.

| S.NO | USART | UART |
|------|---|--|
| 1. | In USART, half duplex mode is used. | While in UART, full duplex mode is used. |
| 2. | The speed of USART is more than the speed of UART. | While the speed of UART is comparatively less. |
| 3. | USART uses both data signals and clock for its functioning. | While UART entails data signals only for its functioning. |
| 4. | In USART, data is transmitted in the form of blocks. | While in UART, data is transmitted in the form of bytes(one byte at a time). |
| 5. | USART can do its function like UART. | Whereas UART can't do its function like USART. |
| 6. | USART is more complex than UART in terms of complexity. | While UART is simple in terms of complexity. |



The main function of USART is to perform serial data communication. In USART, the communication between two devices can be done in two ways namely serial data communication and parallel data communication.



Standardization:

The Universal Synchronous Asynchronous Receiver Transmitter (USART) is a widely used communication protocol, and its specifications are often standardized.

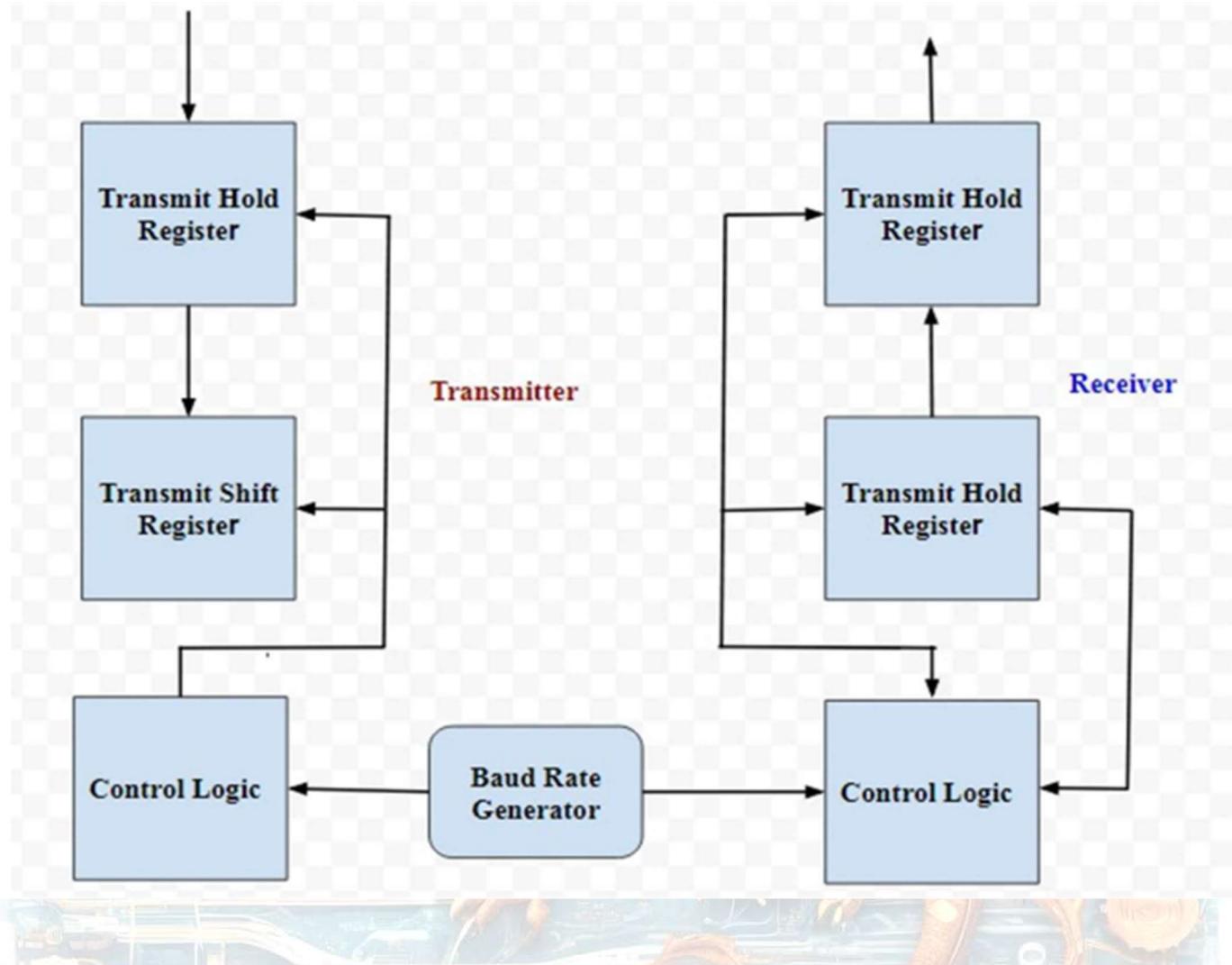
Standardization ensures compatibility across devices and allows for interoperability in various applications.

Features:

The USART (Universal Synchronous Asynchronous Receiver Transmitter) boasts a range of features that make it a versatile communication protocol. Here's a brief overview of its main features:

- **Full Duplex, Asynchronous Communications:** USART supports simultaneous transmission and reception of data, making it suitable for bidirectional communication.
- **NRZ Standard Format (Mark/Space):** Non-Return-to-Zero encoding is employed, utilizing Mark and Space levels for binary representation.
- **Configurable Oversampling:** The oversampling method is adjustable, offering flexibility between speed and clock tolerance. It can be configured by 16 or by 8.
- **Fractional Baud Rate Generator:** The system includes a fractional baud rate generator, allowing for common programmable transmit and receive baud rates.
- **Programmable Data Word Length:** Data word length can be programmed to 8 or 9 bits, adapting to the specific requirements of the communication.
- **Configurable Stop Bits:** Supports the use of 1 or 2 stop bits, providing options for framing data within the communication protocol.
- **LIN (Local Interconnect Network) Capabilities:** Features LIN Master Synchronous Break send capability and LIN slave break detection capability. It supports 13-bit break generation and 10/11 bit break detection when configured for LIN.
- **Transmitter Clock Output:** For synchronous transmission, the USART provides a transmitter clock output.
- **IrDA SIR Encoder Decoder:** Supports Infrared Data Association (IrDA) SIR encoding and decoding, including 3/16 bit duration for normal mode.
- **Smartcard Emulation Capability:** The USART can emulate Smartcard interfaces as per ISO 7816-3 standards, supporting 0.5, 1.5 stop bits for Smartcard operations.
- **Single-Wire Half-Duplex Communication:** Facilitates communication over a single wire in half-duplex mode.
- **Configurable Multibuffer Communication using DMA:** Allows for buffering of received/transmitted bytes in reserved SRAM using centralized Direct Memory Access (DMA).

- **Separate Enable Bits for Transmitter and Receiver:** The USART provides separate enable bits for transmitter and receiver, allowing independent control of these functions.
- **Baud Rate:** Determines the speed of data transmission, with common rates such as 9600, 19200, and 115200 bits per second (bps).



Throughput:

Throughput in USART communication is influenced by factors like baud rate, data frame size, and additional control bits. It represents the actual data transfer rate and is crucial for assessing the efficiency of communication.

Hardware vs. Software Flow Controls:

Hardware Flow Control: Involves dedicated hardware lines (e.g., RTS/CTS) to manage data flow between devices, enhancing reliability.

Software Flow Control: Uses special control characters within the data stream for flow management, offering a more flexible but potentially less robust solution.

Oversampling Techniques:

Oversampling is a technique employed to enhance the accuracy of sampled data. USART often uses oversampling to improve reliability. By sampling the received signal at a rate higher than the Nyquist rate, errors can be minimized, contributing to more accurate data reception.

$$\text{Tx/Rx baud} = \frac{f_{CK}}{8 \times \text{USARTDIV}}$$

if OVER8 = 1

Peripheral Clock

Divide factor to generate different baud rates

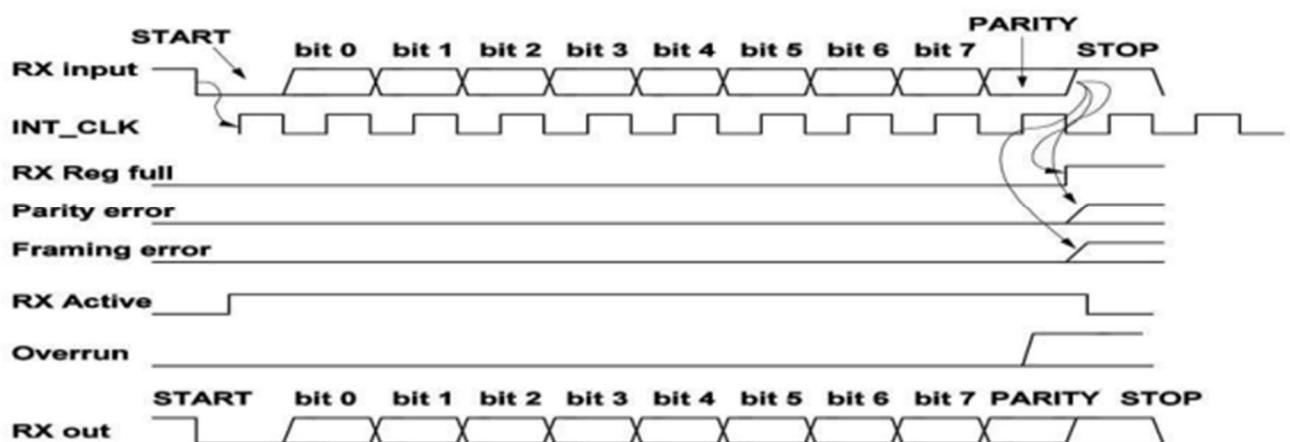
$$\text{Tx/Rx baud} = \frac{f_{CK}}{16 \times \text{USARTDIV}}$$

if OVER8 = 0

Overrun Condition:

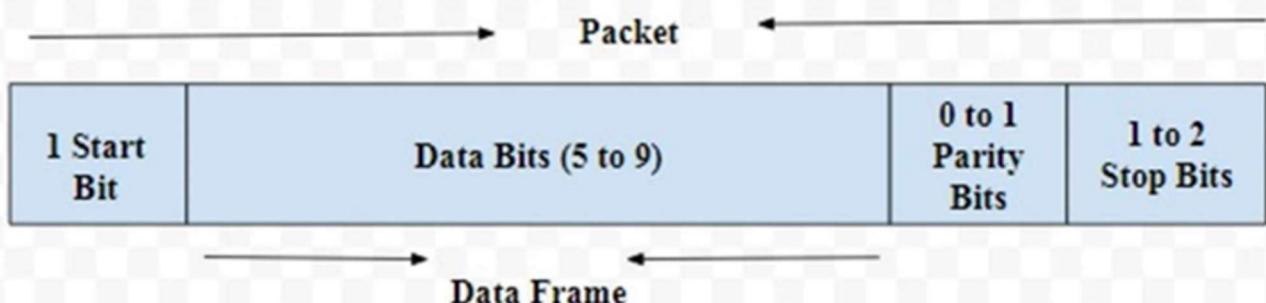
An overrun condition occurs when the USART receiver is unable to process incoming data at the rate it is being received. This can lead to data loss, and addressing this condition is crucial for maintaining data integrity.

- **UART Receive Timing diagram:**



Message Frame and Packets:

The data transmission of a UART can be done by using a data bus in the form of parallel by other devices like a microcontroller, memory, CPU, etc. After receiving the parallel data from the bus, it forms a data packet by adding three bits like start, stop and parity. It reads the data packet bit by bit and converts the received data into the parallel form to eliminate the three bits of the data packet. In conclusion, the data packet received by the UART transfers in parallel toward the data bus at the receiving end.



| | |
|--------------------------------|--|
| Start Bit | Start-bit is also known as a synchronization bit that is placed before the actual data. Generally, an inactive data transmission line is controlled at a high-voltage level. To begin the data transmission, the UART transmission drags the data-line from a high voltage level (1) to a low voltage level (0). The obtaining UART notices this transform from the high level to low level over the data line as well as starts understanding the real data. Generally, there is just a single start bit. |
| Data Bits or Data Frame | The data bits include the real data being conveyed from the sender to receiver. The data frame length could be between 5 & 8. If the parity bit is not used when the data frame length could be 9-bit long. Generally, the LSB of the data to be transmitted first then it is very useful for transmitting. |
| Parity Bit | Parity bit lets the receiver to ensure whether the collected data is right or not. It is a lowlevel fault checking system & parity bit is available in two ranges such as Even Parity as well as Odd Parity. Actually, this bit is not widely used so it is not compulsory. |
| Stop Bit | The Stop Bit is placed at the ending of the data packet. Usually, this bit is 2-bits lengthy but frequently on bit only utilized. To stop the broadcast, the UART keeps the data-line on high voltage. |

Common USART Applications:

Embedded Systems:

USART is integral in microcontroller-based systems for communication with peripherals and external devices. It provides a reliable means for information exchange within embedded applications.



PC-Peripheral Communication:

Devices such as printers, scanners, and modems use USART for communication with computers. This enables seamless connectivity and interaction between various peripherals and the central processing unit.

Industrial Automation:

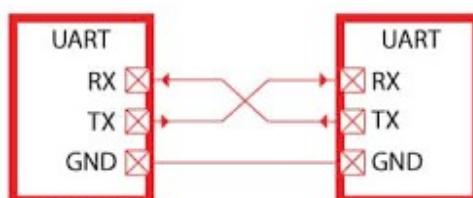
USART plays a vital role in industrial automation by facilitating communication between controllers, sensors, and other components. It forms the backbone of data exchange in automated systems, contributing to efficient and synchronized operations.



Serial Communication between Microcontrollers:

(Which we will explain a simple example about it)

Interconnecting microcontrollers in a network for information exchange is a common application of USART. This is prevalent in scenarios where multiple microcontrollers need to work together, sharing data and coordinating tasks.



Chapter_2_USART_Summary

USART : (Universal Synchronous/Asynchronous Receiver/Transmitter)

Types Of Communication (Serial – Parallel)

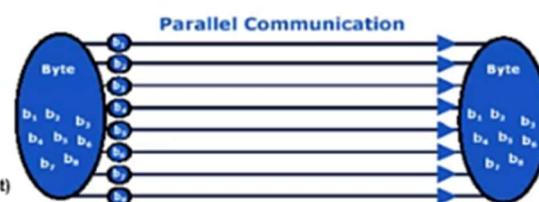
a) **Serial Communication**

- o The data will be sent (bit by bit), One bit at a time.
- o Sequentially transmission.
- o Needs one channel or wire for transmitting
- o **Protocol Examples :** UART – I2C – SPI – USB – Flex Ray – Controller Area Network "CAN" – MOST – LIN.



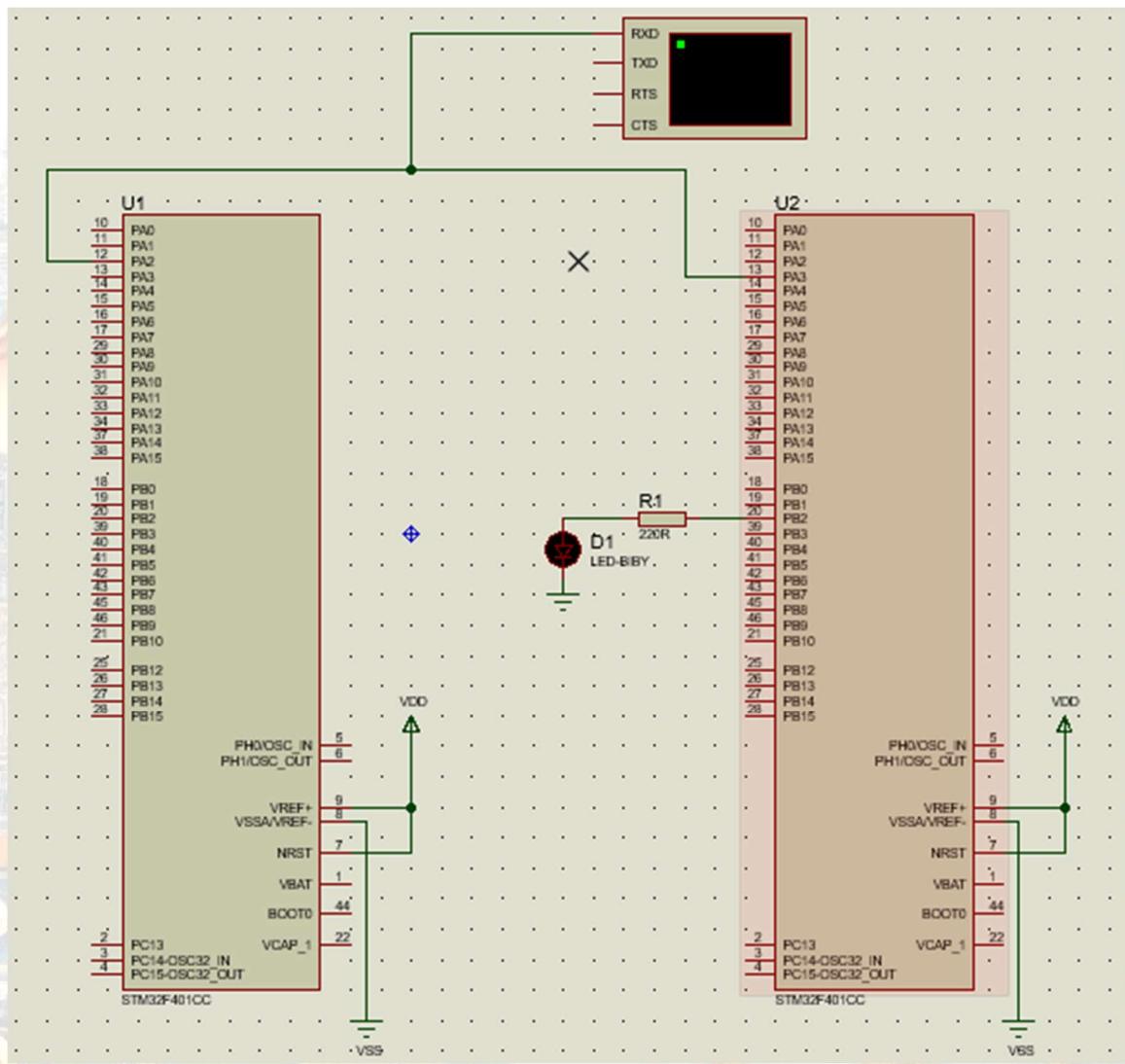
b) **Parallel Communication**

- o More than bit can be transmitted at the same time.
- o Needs more than channel or wire for transmitting.
- o **Protocol Examples :** PCI Bus (Peripheral Component Interconnect)
 - ✓ Connects the CPU and expansion boards such as modem cards, network cards and sound cards.



c) **Wireless Communication**

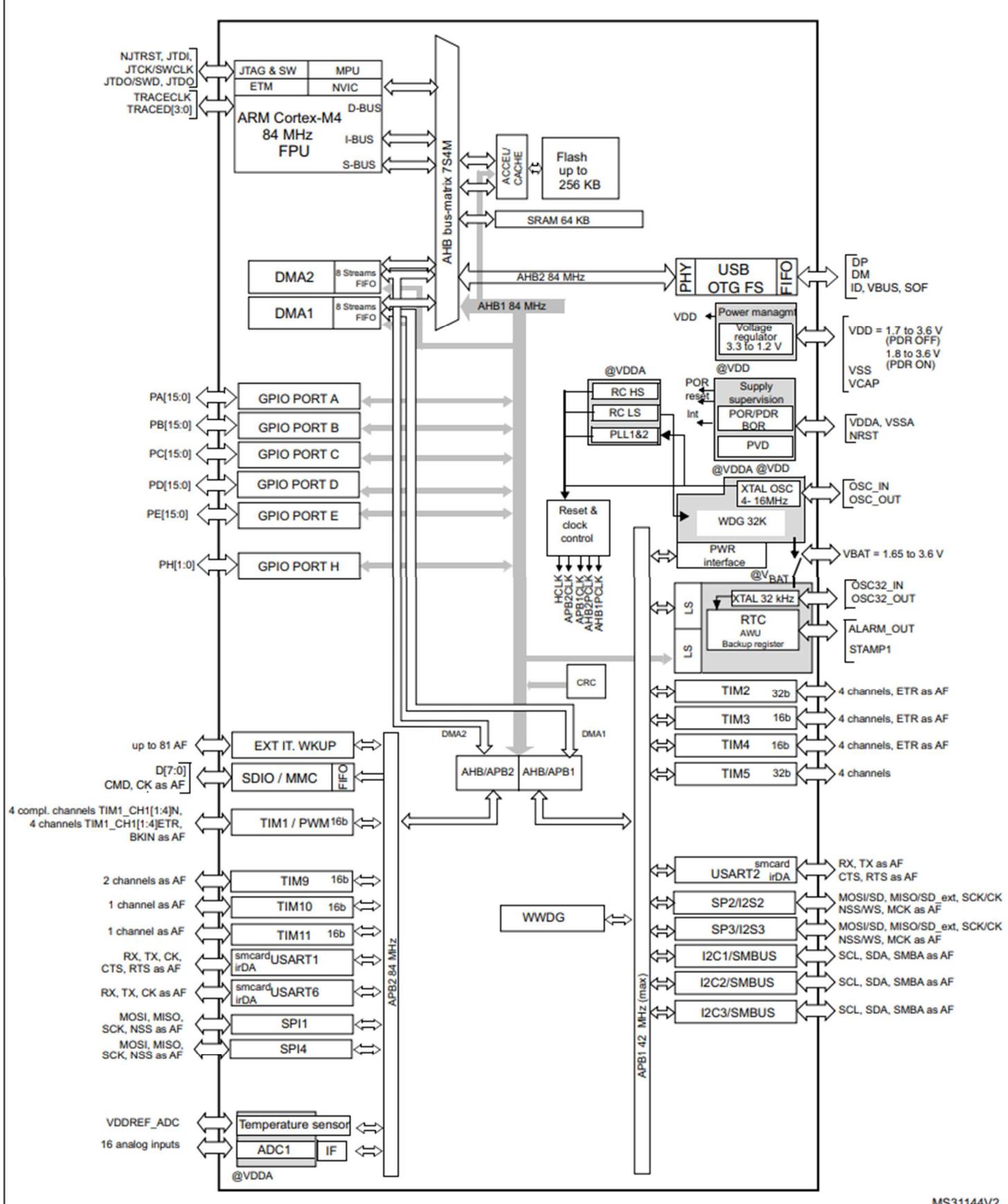
- o This done using (IR → Infrared / Radio Frequency → RF)
- o **Protocol Examples :** Bluetooth – IEEE 802.11 – ZigBee - WIFI - 2G (GSM) - 3G & 4G, ... etc.



In the left MCU we define PA2 as transmitter and, in the right MCU we define PA3 as receiver.

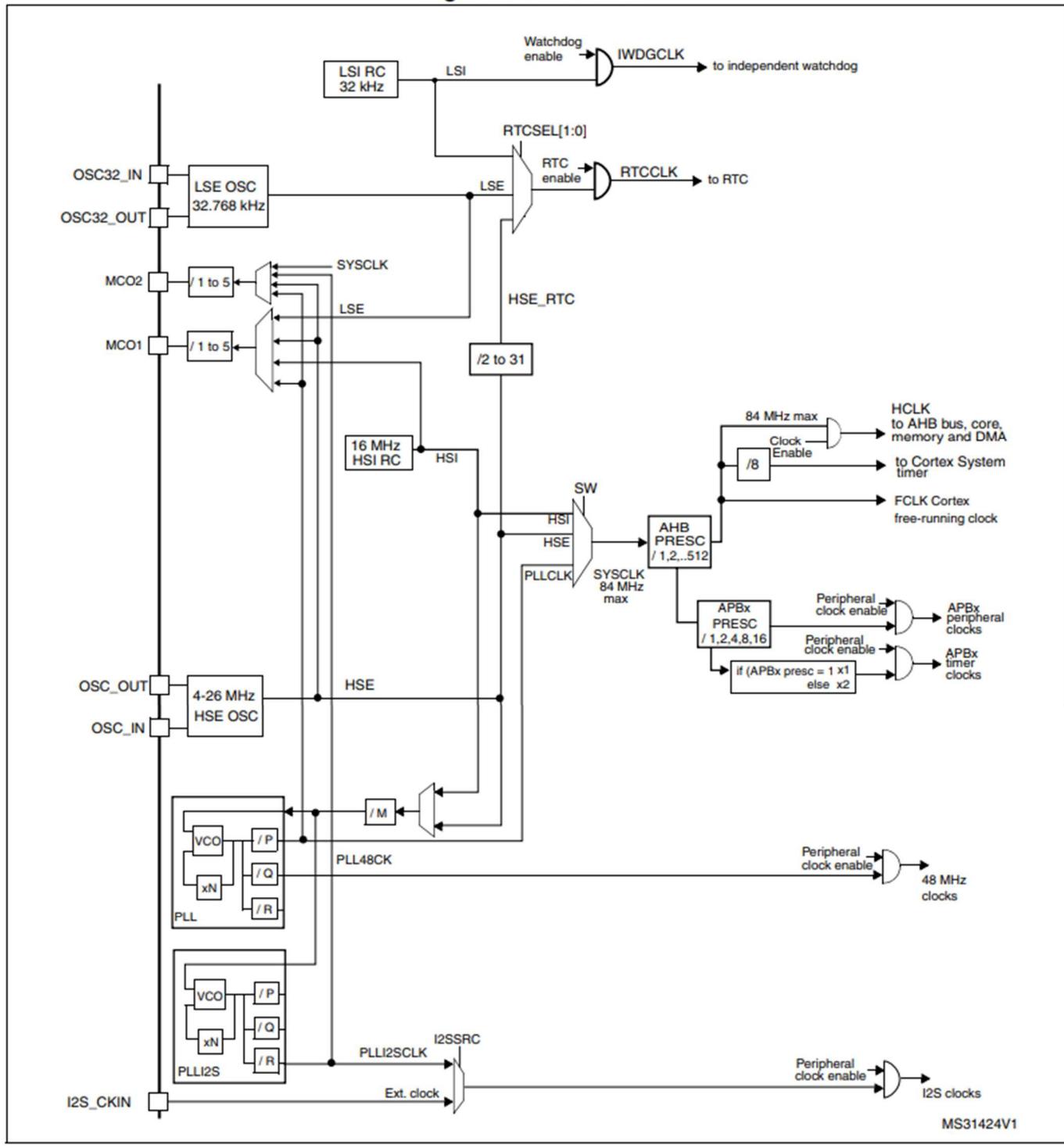
And in the virtual terminal we can see what is inside the data bus which is transmitted by lift MCU and received by the another one .

Figure 3. STM32F401xB/STM32F401xC block diagram



MS31144V2

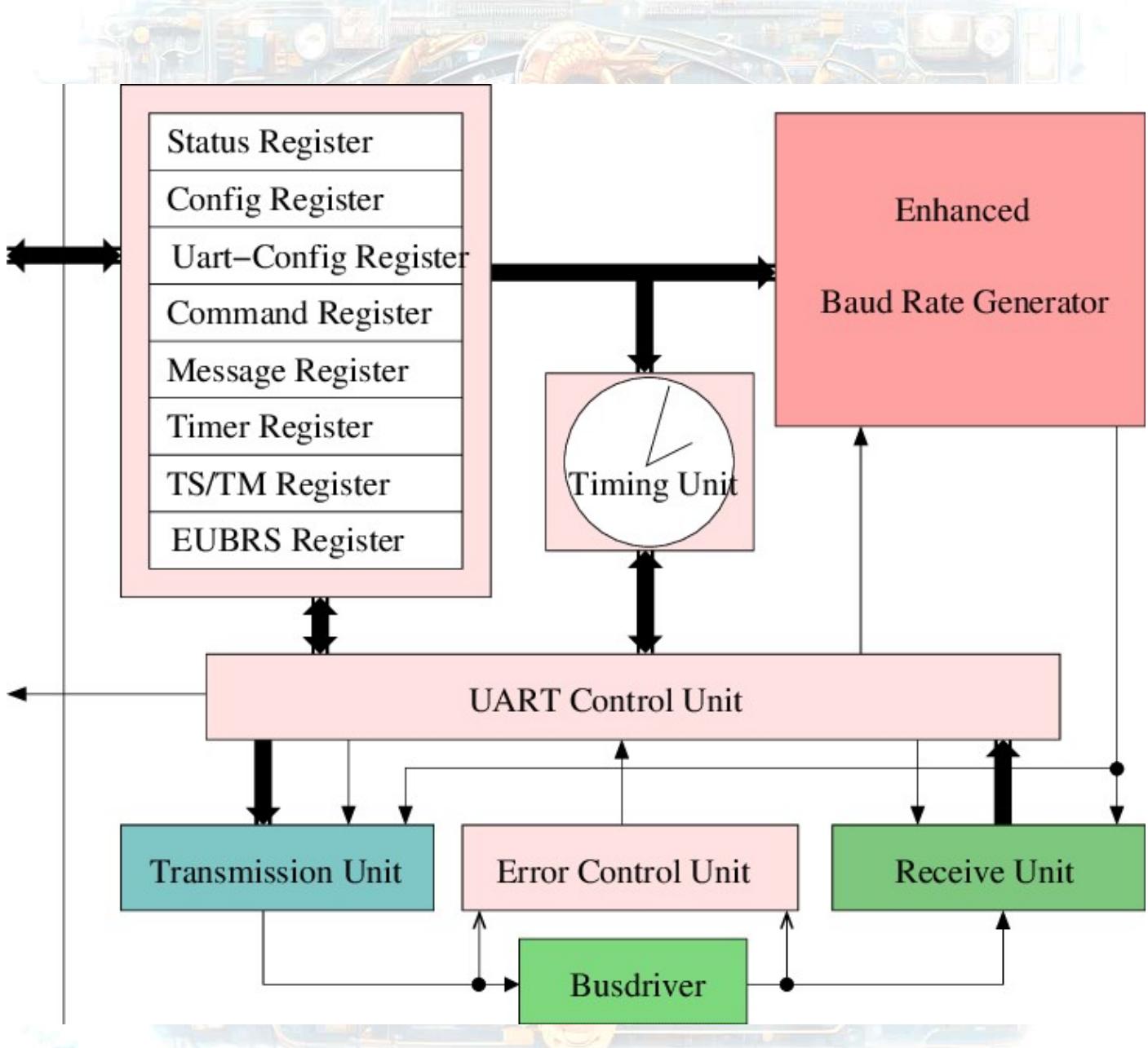
Figure 12. Clock tree



In our code we enable HSI therefore our Clock will be 16MHz

“HSI clock The HSI clock signal is generated from an internal 16 MHz RC oscillator and can be used directly as a system clock, or used as PLL input. The HSI RC oscillator has the advantage of providing a clock source at low cost (no external components). It also has a faster startup time than the HSE crystal oscillator however, even with calibration the frequency is less accurate than an external crystal oscillator or ceramic resonator.”

The Universal Synchronous Asynchronous Receiver Transmitter (USART) module is a communication peripheral commonly found in microcontrollers and other embedded systems. Its primary function is to facilitate serial communication between the microcontroller and other devices, such as sensors, displays, or other microcontrollers. Below is a brief explanation of the USART module block diagram:



1. Transmitter Section:

- **Data Register (TXD):** This register holds the data to be transmitted. When the microcontroller is ready to send data, it loads the TXD register with the data.
- **Transmitter Shift Register:** The data from the TXD register is shifted bit by bit into this register. The shift register is responsible for serializing the parallel data into a stream of bits for transmission.

2. Baud Rate Generator:

- **Baud Rate Control:** The baud rate is the speed at which data is transmitted over the serial communication link. The baud rate generator sets the timing for bit transmission and reception. The microcontroller's clock frequency and the desired baud rate are used to configure the baud rate generator.

3. Control Logic:

- **Control Register:** This register contains various control bits that configure the operation of the USART module. It includes settings such as the number of data bits, parity mode, and stop bits.
- **Mode Control:** The USART module can operate in either synchronous or asynchronous mode. In asynchronous mode, the start bit is used to synchronize communication, while synchronous mode relies on a separate clock signal.

4. Receiver Section:

- **Data Register (RXD):** When data is received, it is stored in the RXD register. The microcontroller reads this register to retrieve the received data.
- **Receiver Shift Register:** The incoming serial data is shifted into this register, where it is deserialized back into parallel data.

5. Interrupt Control:

- **Interrupt Enable/Disable:** USART modules often support interrupts to notify the microcontroller when certain events occur, such as the completion of data transmission or the reception of new data. The interrupt control logic allows the microcontroller to enable or disable these interrupts.

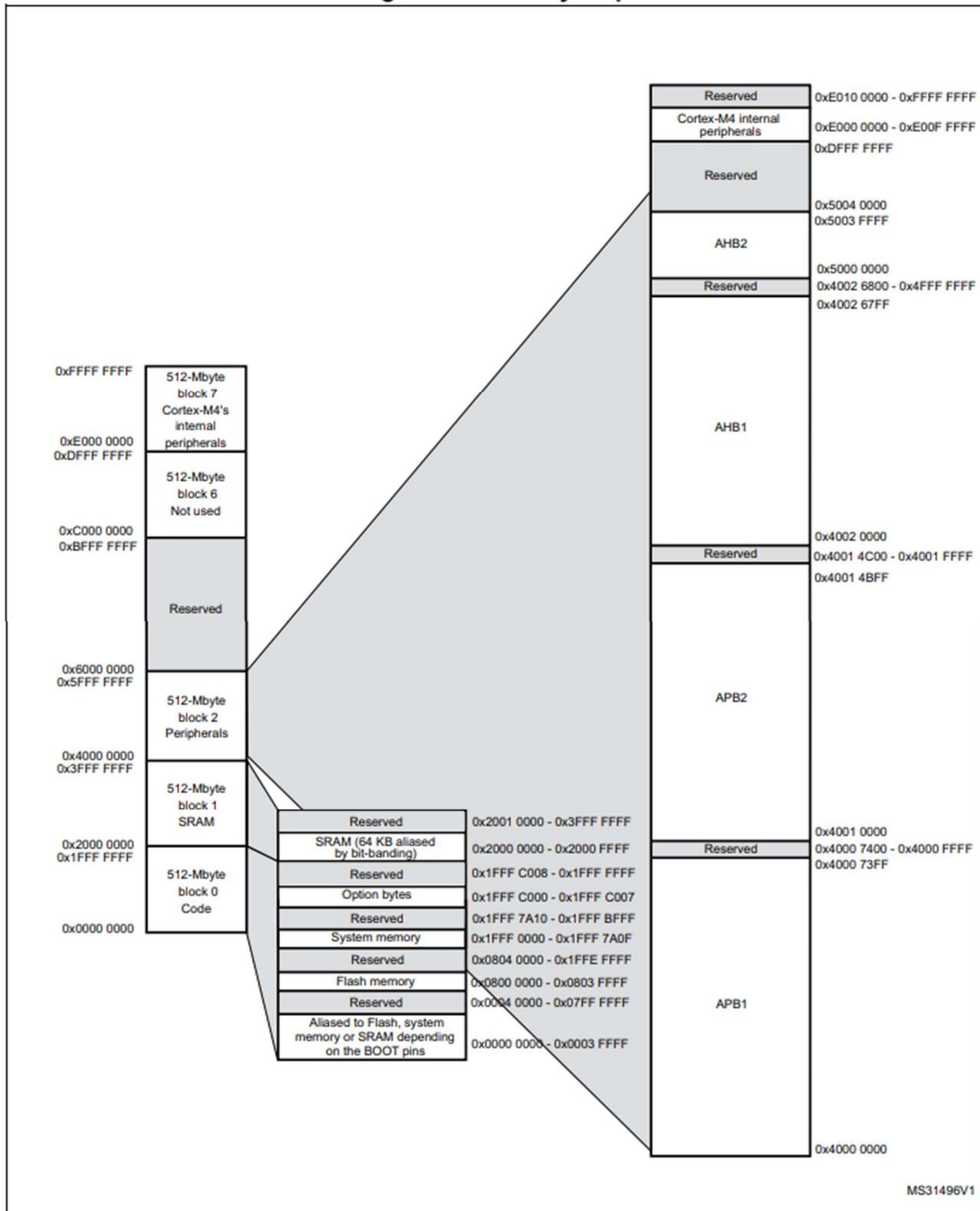
6. Error Detection and Control:

- **Parity Check:** Some USART modules support parity checking for error detection. Parity bits are transmitted or checked based on the chosen parity mode (even, odd, or none).
- **Error Flags:** Flags are provided to indicate the occurrence of errors during communication, such as framing errors or overrun errors.

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|-------------|-------------|---------------|--------------|--------------|--------------|---------------|-------------|-------------|-------------|------------------|----------|----------|----------------------------------|-------------|--------------|-------------|-------------|-------------|---------------|--------------|--------------|--------------|--------------|-------------|-------------|-------------|------------------|---------|---------|----------------------------------|-------------|-------------|-------------|-------------|-------------|-------------|----|----|----|----|----|----|----|----|----|----|--|-------------|---------|----|----|----------|----|---|----|---------------|----|----|---------|---------|---------|---------|----|----------|-------|-------|-------|-------|----------|-----|----------------------------------|-------|-------|-------|-------|-----|-------------|-------------|-------------|---|---|-------------|-------------|-------------|----|----|----|-------|-------|-------|-------|-------|----|----|----|----|---|----------|----|----|----|----|----|----|----------|---------------|----|----|----------|----------|----|----|----|------------|----|----|----|----|----|----|---|------------|---|---|---|------------|---|---|----------|------------|----------|----------|---------|----------|---------|----------|-----------|-----------|----------|----------|----------|----------|----------|----------|----------|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|--|--|--|--|------------|------------|------------|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| <p>6.3.1 RCC clock control register (RCC_CR)</p> <p>Address offset: 0x00 Reset value: 0x0000 XX81 where X is undefined. Access: no wait state, word, half-word and byte access</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td> </tr> <tr> <td colspan="4">Reserved</td><td>PLL12S RDY ON</td><td>PLL12S ON</td><td>PLLRDY</td><td>POLLON</td><td colspan="4">Reserved</td><td>CSS ON</td><td>HSE BYP</td><td>HSE RDY</td><td>HSE ON</td><td></td> </tr> <tr> <td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td> </tr> <tr> <td colspan="8">HSICAL[7:0]</td><td colspan="4">HSISTRIM[4:0]</td><td>Res.</td><td>HSI RDY</td><td>HSION</td><td></td><td></td> </tr> <tr> <td>r</td><td>r</td><td>r</td><td>r</td><td>r</td><td>r</td><td>r</td><td>r</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>r</td><td>rw</td><td></td> </tr> </table> <p>Bits 31:28 Reserved, must be kept at reset value.</p> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | Reserved | | | | PLL12S RDY ON | PLL12S ON | PLLRDY | POLLON | Reserved | | | | CSS ON | HSE BYP | HSE RDY | HSE ON | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | HSICAL[7:0] | | | | | | | | HSISTRIM[4:0] | | | | Res. | HSI RDY | HSION | | | r | r | r | r | r | r | r | r | rw | rw | rw | rw | rw | r | rw | | <p>6.3.9 RCC AHB1 peripheral clock enable register (RCC_AHB1ENR)</p> <p>Address offset: 0x30 Reset value: 0x0000 0000 Access: no wait state, word, half-word and byte access.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td> </tr> <tr> <td colspan="8">Reserved</td><td colspan="4">DMA2EN DMA1EN</td><td colspan="4">Reserved</td><td></td> </tr> <tr> <td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td> </tr> <tr> <td colspan="4">Reserved</td><td colspan="2">CRcen</td><td colspan="4">Reserved</td><td>GPIOH EN</td><td>Reserved</td><td>GPIOEEN</td><td>GPIOF EN</td><td>GPIOG EN</td><td>GPIOH EN</td> </tr> <tr> <td>r</td><td>rw</td><td></td><td></td><td></td><td></td><td>rw</td><td></td><td></td><td></td><td>rw</td><td></td><td>rw</td><td>rw</td><td>rw</td><td></td> </tr> </table> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | Reserved | | | | | | | | DMA2EN DMA1EN | | | | Reserved | | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Reserved | | | | CRcen | | Reserved | | | | GPIOH EN | Reserved | GPIOEEN | GPIOF EN | GPIOG EN | GPIOH EN | r | rw | | | | | rw | | | | rw | | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | | | | PLL12S RDY ON | PLL12S ON | PLLRDY | POLLON | Reserved | | | | CSS ON | HSE BYP | HSE RDY | HSE ON | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| HSICAL[7:0] | | | | | | | | HSISTRIM[4:0] | | | | Res. | HSI RDY | HSION | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| r | r | r | r | r | r | r | r | rw | rw | rw | rw | rw | r | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | | | | | | | | DMA2EN DMA1EN | | | | Reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | | | | CRcen | | Reserved | | | | GPIOH EN | Reserved | GPIOEEN | GPIOF EN | GPIOG EN | GPIOH EN | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| r | rw | | | | | rw | | | | rw | | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| To Enable HIS Clock | To Enable Port (A, B) Clock | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| <p>6.3.11 RCC APB1 peripheral clock enable register (RCC_APB1ENR)</p> <p>Address offset: 0x40 Reset value: 0x0000 0000 Access: no wait state, word, half-word and byte access.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td> </tr> <tr> <td colspan="4">Reserved</td><td>PWR EN</td><td colspan="4">Reserved</td><td>I2C3 EN</td><td>I2C2 EN</td><td>I2C1 EN</td><td colspan="4">Reserved</td> </tr> <tr> <td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td> </tr> <tr> <td>SPI3 EN</td><td>SPI2 EN</td><td colspan="2">WWDG EN</td><td>rw</td><td colspan="4">Reserved</td><td>rw</td><td>rw</td><td>rw</td><td>TIM5 EN</td><td>TIM4 EN</td><td>TIM3 EN</td><td>TIM2 EN</td><td></td> </tr> <tr> <td>rw</td><td>rw</td><td></td><td></td><td></td><td>rw</td><td></td><td></td><td></td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td></td> </tr> </table> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | Reserved | | | | PWR EN | Reserved | | | | I2C3 EN | I2C2 EN | I2C1 EN | Reserved | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | SPI3 EN | SPI2 EN | WWDG EN | | rw | Reserved | | | | rw | rw | rw | TIM5 EN | TIM4 EN | TIM3 EN | TIM2 EN | | rw | rw | | | | rw | | | | rw | rw | rw | rw | rw | rw | | <p>6.3.12 RCC APB2 peripheral clock enable register (RCC_APB2ENR)</p> <p>Address offset: 0x44 Reset value: 0x0000 0000 Access: no wait state, word, half-word and byte access.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td> </tr> <tr> <td colspan="8">Reserved</td><td colspan="4">TIM11 EN</td><td colspan="4">TIM10 EN</td> </tr> <tr> <td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td> </tr> <tr> <td>Reserved</td><td>SYSCF GEN</td><td>SPI4EN</td><td>SPI11 EN</td><td>SDIO EN</td><td>Reserved</td><td>ADC1 EN</td><td>Reserved</td><td>USART6 EN</td><td>USART1 EN</td><td>Reserved</td><td>Reserved</td><td>Reserved</td><td>Reserved</td><td>Reserved</td><td>TI1 EN</td> </tr> <tr> <td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td></td><td>rw</td><td></td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td> </tr> </table> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | Reserved | | | | | | | | TIM11 EN | | | | TIM10 EN | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Reserved | SYSCF GEN | SPI4EN | SPI11 EN | SDIO EN | Reserved | ADC1 EN | Reserved | USART6 EN | USART1 EN | Reserved | Reserved | Reserved | Reserved | Reserved | TI1 EN | rw | rw | rw | rw | rw | | rw | | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | | | | PWR EN | Reserved | | | | I2C3 EN | I2C2 EN | I2C1 EN | Reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| SPI3 EN | SPI2 EN | WWDG EN | | rw | Reserved | | | | rw | rw | rw | TIM5 EN | TIM4 EN | TIM3 EN | TIM2 EN | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rw | rw | | | | rw | | | | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | | | | | | | | TIM11 EN | | | | TIM10 EN | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | SYSCF GEN | SPI4EN | SPI11 EN | SDIO EN | Reserved | ADC1 EN | Reserved | USART6 EN | USART1 EN | Reserved | Reserved | Reserved | Reserved | Reserved | TI1 EN | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | | rw | | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| To Enable USART2 Clock | To Enable TIMER2 Clock | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| <p>8.4.1 GPIO port mode register (GPIOx_MODER) (x = A..E and H)</p> <p>Address offset: 0x00 Reset values:<ul style="list-style-type: none">• 0x0C00 0000 for port A• 0x0000 0280 for port B• 0x0000 0000 for other ports</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td> </tr> <tr> <td colspan="4">MODER15[1:0]</td><td>MODER14[1:0]</td><td>MODER13[1:0]</td><td>MODER12[1:0]</td><td>MODER11[1:0]</td><td>MODER10[1:0]</td><td>MODER9[1:0]</td><td>MODER8[1:0]</td><td colspan="4">MODER7[1:0]</td><td>MODER6[1:0]</td><td>MODER5[1:0]</td><td>MODER4[1:0]</td><td>MODER3[1:0]</td><td>MODER2[1:0]</td><td>MODER1[1:0]</td><td>MODER0[1:0]</td> </tr> <tr> <td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td> </tr> <tr> <td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td><td></td><td></td><td></td> </tr> <tr> <td>MODER7[1:0]</td><td>MODER6[1:0]</td><td>MODER5[1:0]</td><td>MODER4[1:0]</td><td>MODER3[1:0]</td><td>MODER2[1:0]</td><td>MODER1[1:0]</td><td>MODER0[1:0]</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td> </tr> </table> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | MODER15[1:0] | | | | MODER14[1:0] | MODER13[1:0] | MODER12[1:0] | MODER11[1:0] | MODER10[1:0] | MODER9[1:0] | MODER8[1:0] | MODER7[1:0] | | | | MODER6[1:0] | MODER5[1:0] | MODER4[1:0] | MODER3[1:0] | MODER2[1:0] | MODER1[1:0] | MODER0[1:0] | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | MODER7[1:0] | MODER6[1:0] | MODER5[1:0] | MODER4[1:0] | MODER3[1:0] | MODER2[1:0] | MODER1[1:0] | MODER0[1:0] | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | <p>8.4.9 GPIO alternate function low register (GPIOx_AFRL) (x = A..E and H)</p> <p>Address offset: 0x20 Reset value: 0x0000 0000</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td> </tr> <tr> <td colspan="8">AFRL7[3:0]</td><td colspan="4">AFRL6[3:0]</td><td colspan="4">AFRL5[3:0]</td><td colspan="4">AFRL4[3:0]</td> </tr> <tr> <td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td> </tr> <tr> <td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td><td></td><td></td><td></td> </tr> <tr> <td>AFRL3[3:0]</td><td>AFRL2[3:0]</td><td>AFRL1[3:0]</td><td>AFRL0[3:0]</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td> </tr> </table> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | AFRL7[3:0] | | | | | | | | AFRL6[3:0] | | | | AFRL5[3:0] | | | | AFRL4[3:0] | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | AFRL3[3:0] | AFRL2[3:0] | AFRL1[3:0] | AFRL0[3:0] | rw |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| MODER15[1:0] | | | | MODER14[1:0] | MODER13[1:0] | MODER12[1:0] | MODER11[1:0] | MODER10[1:0] | MODER9[1:0] | MODER8[1:0] | MODER7[1:0] | | | | MODER6[1:0] | MODER5[1:0] | MODER4[1:0] | MODER3[1:0] | MODER2[1:0] | MODER1[1:0] | MODER0[1:0] | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| MODER7[1:0] | MODER6[1:0] | MODER5[1:0] | MODER4[1:0] | MODER3[1:0] | MODER2[1:0] | MODER1[1:0] | MODER0[1:0] | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| AFRL7[3:0] | | | | | | | | AFRL6[3:0] | | | | AFRL5[3:0] | | | | AFRL4[3:0] | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| AFRL3[3:0] | AFRL2[3:0] | AFRL1[3:0] | AFRL0[3:0] | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| To Configure the pins (output or alternative function) | To Select which signal that will be mux. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| <p>13.4.1 TIMx control register 1 (TIMx_CR1)</p> <p>Address offset: 0x0000 Reset value: 0x0000</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td> </tr> <tr> <td colspan="4">Reserved</td><td>CKD[1:0]</td><td>ARPE</td><td>CMS</td><td>DIR</td><td>OPM</td><td>URS</td><td>UDIS</td><td>CEN</td><td colspan="4">Pin x (x = 0..7)</td> </tr> <tr> <td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td> </tr> </table> | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Reserved | | | | CKD[1:0] | ARPE | CMS | DIR | OPM | URS | UDIS | CEN | Pin x (x = 0..7) | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | <p>13.4.5 TIMx status register (TIMx_SR)</p> <p>Address offset: 0x0010 Reset value: 0x0000</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td> </tr> <tr> <td>Reserved</td><td>CC4OF</td><td>CC3OF</td><td>CC2OF</td><td>CC1OF</td><td>Reserved</td><td>TIF</td><td>Res</td><td>CC4IF</td><td>CC3IF</td><td>CC2IF</td><td>CC1IF</td><td>UIF</td><td></td><td></td><td></td> </tr> <tr> <td>rc_w0</td><td>rc_w0</td><td>rc_w0</td><td>rc_w0</td><td>rc_w0</td><td></td><td>rw</td><td>rw</td><td>rc_w0</td><td>rc_w0</td><td>rc_w0</td><td>rc_w0</td><td>rc_w0</td><td></td><td></td><td></td> </tr> </table> | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Reserved | CC4OF | CC3OF | CC2OF | CC1OF | Reserved | TIF | Res | CC4IF | CC3IF | CC2IF | CC1IF | UIF | | | | rc_w0 | rc_w0 | rc_w0 | rc_w0 | rc_w0 | | rw | rw | rc_w0 | rc_w0 | rc_w0 | rc_w0 | rc_w0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | | | | CKD[1:0] | ARPE | CMS | DIR | OPM | URS | UDIS | CEN | Pin x (x = 0..7) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | CC4OF | CC3OF | CC2OF | CC1OF | Reserved | TIF | Res | CC4IF | CC3IF | CC2IF | CC1IF | UIF | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rc_w0 | rc_w0 | rc_w0 | rc_w0 | rc_w0 | | rw | rw | rc_w0 | rc_w0 | rc_w0 | rc_w0 | rc_w0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| To make counter: AUTO RELOAD, EDGE ALIGNED MODE (UP OR DOWN ONLY) SET THE DIRECTION TO "UP" | Update interrupt flag | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| <p>13.4.6 TIMx event generation register (TIMx_EGR)</p> <p>Address offset: 0x14 Reset value: 0x0000</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td> </tr> <tr> <td colspan="8">Reserved</td><td>TG</td><td>Res.</td><td>CC4G</td><td>CC3G</td><td>CC2G</td><td>CC1G</td><td>UG</td><td colspan="4">CNT[31:16] (depending on timers)</td> </tr> <tr> <td>w</td><td>w</td><td>w</td><td>w</td><td>w</td><td>w</td><td>w</td><td>w</td><td>w</td><td>w</td><td>w</td><td>w</td><td>w</td><td>w</td><td>w</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td> </tr> </table> | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Reserved | | | | | | | | TG | Res. | CC4G | CC3G | CC2G | CC1G | UG | CNT[31:16] (depending on timers) | | | | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | rw | rw | rw | rw | rw | <p>13.4.10 TIMx counter (TIMx_CNT)</p> <p>Address offset: 0x24 Reset value: 0x0000 0000</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td> </tr> <tr> <td colspan="8">CNT[31:16] (depending on timers)</td><td colspan="4">CNT[15:0]</td><td colspan="4">CNT[15:0]</td> </tr> <tr> <td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td> </tr> <tr> <td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td><td></td><td></td><td></td><td></td> </tr> </table> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | CNT[31:16] (depending on timers) | | | | | | | | CNT[15:0] | | | | CNT[15:0] | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | | | | | | | | TG | Res. | CC4G | CC3G | CC2G | CC1G | UG | CNT[31:16] (depending on timers) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| CNT[31:16] (depending on timers) | | | | | | | | CNT[15:0] | | | | CNT[15:0] | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--|---|-----------|-------|------|------|------|------|-----------------------------|------|--------|----------|----|----|-----|-----|----|-----------|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|-------|-----------|-------|------|------|------|------|-----------------------------|------|------|----------|----|----|----|----|----------|----------------------------------|----|----|----|----|----|----|-----------------------------|----|----|----|----|----|----|----|--|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----------|----------|----|----|----|----|---|---|---|---|---|---|---|---|---|---|-----------|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|-------|----------|-------|------|------|------|------|------|------|--------|--------|--------|-----|----|-----|-----|----|------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| <p>13.4.11 TIMx prescaler (TIMx_PSC) Address offset: 0x28 Reset value: 0x0000</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr> <tr><td colspan="16" style="text-align: center;">PSC[15:0]</td></tr> <tr><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td></tr> </table> <p>Bits 15:0 PSC[15:0]: Prescaler value The counter clock frequency CK_CNT is equal to $f_{CK_PSC} / (PSC[15:0] + 1)$.</p> | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | PSC[15:0] | | | | | | | | | | | | | | | | rw | <p>13.4.12 TIMx auto-reload register (TIMx_ARR) Address offset: 0x2C Reset value: 0xFFFF FFFF</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr> <tr><td colspan="16" style="text-align: center;">ARR[31:16] (depending on timers)</td></tr> <tr><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td></tr> <tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr> <tr><td colspan="16" style="text-align: center;">ARR[15:0]</td></tr> <tr><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td></tr> </table> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | ARR[31:16] (depending on timers) | | | | | | | | | | | | | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | ARR[15:0] | | | | | | | | | | | | | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| PSC[15:0] | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ARR[31:16] (depending on timers) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ARR[15:0] | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| FOR USART | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| <p>19.6.1 Status register (USART_SR) Address offset: 0x00 Reset value: 0x00C0 0000</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr> <tr><td colspan="16" style="text-align: center;">Reserved</td></tr> <tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr> <tr><td colspan="8" style="text-align: center;">CTS LBD TXE TC RXNE IDLE ORE NF FE PE</td><td colspan="8" style="text-align: center;">rc_w0 rc_w0 r rc_w0 r r r r</td></tr> <tr><td colspan="8" style="text-align: center;">Reserved</td><td colspan="8" style="text-align: center;">rc_w0 rc_w0 r rc_w0 r r r r</td></tr> </table> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | Reserved | | | | | | | | | | | | | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | CTS LBD TXE TC RXNE IDLE ORE NF FE PE | | | | | | | | rc_w0 rc_w0 r rc_w0 r r r r | | | | | | | | Reserved | | | | | | | | rc_w0 rc_w0 r rc_w0 r r r r | | | | | | | | <p>19.6.2 Data register (USART_DR) Address offset: 0x04 Reset value: 0XXXXX XXXXX</p> <p>Bits 31:9 Reserved, must be kept at reset value Bits 8:0 DR[8:0]: Data value Contains the Received or Transmitted data character, depending on whether it is read from or written to.</p> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| CTS LBD TXE TC RXNE IDLE ORE NF FE PE | | | | | | | | rc_w0 rc_w0 r rc_w0 r r r r | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | | | | | | | | rc_w0 rc_w0 r rc_w0 r r r r | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| States register | Date register | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| <p>19.6.3 Baud rate register (USART_BRR) Note: The baud counters stop counting if the TE or RE bits are disabled respectively. Address offset: 0x08 Reset value: 0x0000 0000</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr> <tr><td colspan="16" style="text-align: center;">Reserved</td></tr> <tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr> <tr><td colspan="8" style="text-align: center;">DIV_Mantissa[11:0]</td><td colspan="8" style="text-align: center;">DIV_Fraction[3:0]</td></tr> <tr><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td></tr> </table> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | Reserved | | | | | | | | | | | | | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | DIV_Mantissa[11:0] | | | | | | | | DIV_Fraction[3:0] | | | | | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | <p>19.6.4 Control register 1 (USART_CR1) Address offset: 0x0C Reset value: 0x0000 0000</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr> <tr><td colspan="16" style="text-align: center;">Reserved</td></tr> <tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr> <tr><td>OVER8</td><td>Reserved</td><td>UE</td><td>M</td><td>WAKE</td><td>PCE</td><td>PS</td><td>PEIE</td><td>TXIE</td><td>TCIE</td><td>RXNEIE</td><td>IDLEIE</td><td>TE</td><td>RE</td><td>RWU</td><td>SBK</td></tr> <tr><td>rw</td><td>Res.</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td></tr> </table> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | Reserved | | | | | | | | | | | | | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | OVER8 | Reserved | UE | M | WAKE | PCE | PS | PEIE | TXIE | TCIE | RXNEIE | IDLEIE | TE | RE | RWU | SBK | rw | Res. | rw | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DIV_Mantissa[11:0] | | | | | | | | DIV_Fraction[3:0] | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| OVER8 | Reserved | UE | M | WAKE | PCE | PS | PEIE | TXIE | TCIE | RXNEIE | IDLEIE | TE | RE | RWU | SBK | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rw | Res. | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BaudRate (Mantissa & Fraction) | TRANSMITTER ENABLE RECIEVER ENABLE Disable parity Set word length to 1 start bit and 8 data bits Over sample by 8 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| <p>19.6.5 Control register 2 (USART_CR2) Address offset: 0x10 Reset value: 0x0000 0000</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr> <tr><td colspan="16" style="text-align: center;">Reserved</td></tr> <tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr> <tr><td>Res.</td><td>LINEN</td><td>STOP[1:0]</td><td>CLKEN</td><td>CPOL</td><td>CPHA</td><td>LBCL</td><td>Res.</td><td>LBDIE</td><td>LBOL</td><td>Res.</td><td colspan="6">ADD[3:0]</td></tr> <tr><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td></tr> </table> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | Reserved | | | | | | | | | | | | | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Res. | LINEN | STOP[1:0] | CLKEN | CPOL | CPHA | LBCL | Res. | LBDIE | LBOL | Res. | ADD[3:0] | | | | | | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | <p>19.6.6 Control register 3 (USART_CR3) Address offset: 0x14 Reset value: 0x0000 0000</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr> <tr><td colspan="16" style="text-align: center;">Reserved</td></tr> <tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr> <tr><td>ONEBIT</td><td>CTSIE</td><td>CTSE</td><td>RTSE</td><td>DMAT</td><td>DMAR</td><td>SCEN</td><td>NACK</td><td>HDSSEL</td><td>IRLP</td><td>IREN</td><td>EIE</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td></tr> <tr><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td><td>rw</td></tr> </table> | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | Reserved | | | | | | | | | | | | | | | | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | ONEBIT | CTSIE | CTSE | RTSE | DMAT | DMAR | SCEN | NACK | HDSSEL | IRLP | IREN | EIE | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Res. | LINEN | STOP[1:0] | CLKEN | CPOL | CPHA | LBCL | Res. | LBDIE | LBOL | Res. | ADD[3:0] | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ONEBIT | CTSIE | CTSE | RTSE | DMAT | DMAR | SCEN | NACK | HDSSEL | IRLP | IREN | EIE | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Disable clock 1 stop bit | Half-duplex mode is disabled One sample per bit | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Figure 15. Memory map



MS31496V1

Table 9. Alternate function mapping

| Port: | AF00 | AF01 | AF02 | AF03 | AF04 | AF05 | AF06 | AF07 | AF08 | AF09 | AF10 | AF11 | AF12 | AF13 | AF14 | AF15 |
|--------|------------|-----------------------|--------------------------|--------------------|---------------------------------------|-------------------------|---------------------------------|----------------|---------------|---------|-----------------|------|------|------|------|--------------|
| SYS_AF | TIM1/TIM2 | TIM3/ TIM4/TIM5 | TIM9/ TIM10/ TIM11 | I2C1/I2C2/ I2C3 | SPI1/SPI2/ I2S2/SPI3/ I2S3/SPI4 | SPI2/I2S2/ SPI3/I2S3 | SPI3/I2S3/ USART1/ USART2 | USART6 | I2C2/ I2C3 | OTG1_FS | SDIO | | | | | |
| PA0 | - | TIM2_CH1/ TIM2_ETR | TIM5_CH1 | - | - | - | - | USART2_ CTS | - | - | - | - | - | - | - | EVENT OUT |
| PA1 | - | TIM2_CH2 | TIM5_CH2 | - | - | - | - | USART2_ RTS | - | - | - | - | - | - | - | EVENT OUT |
| PA2 | - | TIM2_CH3 | TIM5_CH3 | TIM9_CH1 | - | - | - | USART2_ TX | - | - | - | - | - | - | - | EVENT OUT |
| PA3 | - | TIM2_CH4 | TIM5_CH4 | TIM9_CH2 | - | - | - | USART2_ RX | - | - | - | - | - | - | - | EVENT OUT |
| PA4 | - | - | - | - | - | SPI1_NSS | SPI3_NSS/ I2S3_WS | USART2_ CK | - | - | - | - | - | - | - | EVENT OUT |
| PA5 | - | TIM2_CH1/ TIM2_ETR | - | - | - | SPI1_SCK | - | - | - | - | - | - | - | - | - | EVENT OUT |
| PA6 | - | TIM1_BKIN | TIM3_CH1 | - | - | SPI1_MISO | - | - | - | - | - | - | - | - | - | EVENT OUT |
| PA7 | - | TIM1_CH1N | TIM3_CH2 | - | - | SPI1_MOSI | - | - | - | - | - | - | - | - | - | EVENT OUT |
| PA8 | MCO_1 | TIM1_CH1 | - | - | I2C3_SCL | - | - | USART1_ OK | - | - | OTG_FS_ SOF | - | - | - | - | EVENT OUT |
| PA9 | - | TIM1_CH2 | - | - | I2C3_SMBA | - | - | USART1_ TX | - | - | OTG_FS_ VBUS | - | - | - | - | EVENT OUT |
| PA10 | - | TIM1_CH3 | - | - | - | - | - | USART1_ RX | - | - | OTG_FS_J D | - | - | - | - | EVENT OUT |
| PA11 | - | TIM1_CH4 | - | - | - | - | - | USART1_CTS | USART6_ TX | - | OTG_FS_DM | - | - | - | - | EVENT OUT |
| PA12 | - | TIM1_ETR | - | - | - | - | - | USART1_RTS | USART6_RX | - | OTG_FS_DP | - | - | - | - | EVENT OUT |
| PA13 | JTMS_SWDIO | - | - | - | - | - | - | - | - | - | - | - | - | - | - | EVENT OUT |
| PA14 | JTCK_SWCLK | - | - | - | - | - | - | - | - | - | - | - | - | - | - | EVENT OUT |
| PA15 | JTDI | TIM2_CH1/ TIM2_ETR | - | - | - | SPI1_NSS | SPI3_NSS/ I2S3_WS | - | - | - | - | - | - | - | - | EVENT OUT |



Chapter_3_Function_Documentation

```
bool create_driver(unsigned int id)
```

Description:

This function creates a device driver and signs it with the provided unique ID.

Parameters:

id: a unique non-zero device ID.

Return value:

if the function succeeded a true is returned, otherwise, false.

Source:

```
bool create_driver(unsigned int id)
{
    //The function code
}
```

```
/**If the function is split into sub-private
functions, include their code here.**/
```

| | |
|--------------|--|
| | void begin(uint32_t baudRate) |
| Description | <ul style="list-style-type: none"> • Initialize the GPIO • Set the given baud rate: 9600, 115200, ... • Set the USART to asynchronous mode. • Set the data word length to 8 bit. • Set to only 1 stop bit. • Set the parity to none. • Set the USART mode. • Set to full-duplex. • Set the oversampling mode. |
| Parameters | The given baudrate |
| Return Value | null |
| | <p style="color: red; font-weight: bold;">Source</p> <pre> void begin(uint32_t baudRate) { /*----- Initialize the GPIO -----*/ RCC_CR = (1<<0); //Enable HSI Clock while(!(RCC_CR & (1<<1))) { } //waits until HSI is ready RCC_AHB1ENR = (1 << 0); // ENABLE PORT A CLOCK GPIOA_MODER = (1<<5); // CONFIGURE PA2 IN ALTERNATE FUNCTION MODE GPIOA_AFRL = (7<<8); // ENABLE ALTERNATE FUNCTION TO USE USART2 (ON PA2) /*----- Initialize USART -----*/ RCC_APB1ENR = (1<<17); // USART2 clock enable USART2_CR1 = (1<<15); //Set the over sampling bit to over sample by 8 USART2_CR1 = (1<<13); // USART ENABLE USART2_CR1 &= ~(1<<12); // 0>To Set word length to 1 start bit and 8 data bits USART2_CR1 &= ~(1<<10); // Set parity enable to zero to disable parity USART2_CR1 = (1<<3); // TRANSMITTER ENABLE USART2_CR2 &= ~(1<<13); // Set bit 12 and 13 as zero for 1 stop bit USART2_CR2 &= ~(1<<12); // Set bit 12 and 13 as zero for 1 stop bit USART2_CR2 &= ~(1<<11); // Disable clock USART2_CR3 = (1<<11); // One sample per bit USART2_CR3 &= ~(1<<3); // Half-duplex mode is disabled /*----- SET THE GIVEN BAUD RATE -----*/ double USART_DIV = (double)System_Core_Clock / (8.0 * baudRate); uint32_t DIV_Mantissa = (uint32_t)USART_DIV; uint32_t DIV_Fraction = (uint32_t)((USART_DIV - (double)DIV_Mantissa) * 8.0); USART2_BRR = (DIV_Mantissa << 4) (DIV_Fraction & 0x0F); } </pre> |

| | |
|-----------------------|---|
| | STATE available(void) |
| Description | Check if the received data buffer is empty or not. |
| Parameters | null |
| Return Value | If state == 1 → there is data available If state == 0 → there is no data available |
| | Source |
| STATE available(void) | |
| { | |
| | return (USART2_SR & (1 << 5)) != 0; // If RXNE is set, there is data available |
| } | |

| | |
|---------------------------------|--|
| | DATA RECEIVED read(void) |
| Description | Receives the data, if the data buffer is not empty. |
| Parameters | |
| Return Value | |
| | Source |
| DATA_RECEIVED read(void) | |
| { | |
| | while (!(USART2_SR & (1 << 5))){} // WAIT UNTILL THE RECIEVING |
| DATA REGISTER IS EMPTY | |
| | return USART2_DR & 0xFF; // RETURN THE |
| DATA WHICH IN THE DATA REGISTER | |
| } | |

| | |
|---|---|
| | DATA RECEIVED STATE readBytes(uint8_t* BUFFER, uint32_t LENGTH) |
| Description | Read the number of bytes specified by len and store them in buffer. |
| Parameters | Data and the number of it's characters |
| Return Value | 1 that mean all the data has been read |
| | Source |
| DATA_RECEIVED_STATE readBytes(uint8_t* BUFFER, uint32_t LENGTH) | |
| { | |
| | for (uint32_t i = 0; i < LENGTH; ++i) |
| | { |
| | while (!(USART2_SR & (1 << 5))){} // WAIT UNTILL THE RECIEVING |
| DATA REGISTER IS EMPTY | |
| | BUFFER[i] = USART2_DR & 0xFF; // buffer[i] = USART1->DR; // |
| Read received data | |
| } | |
| | return 1 ; |
| | } |

| | |
|--------------|---|
| | void write(uint8_t BYTE) |
| Description | Send the data present in the input parameter. |
| Parameters | The char you will send |
| Return Value | null |

Source

```
void write(uint8_t BYTE)
{
    while (!(USART2_SR & (1 << 7))){}           // WAIT UNTILL THE TRANSMIT
DATA REGISTER IS EMPTY
    USART2_DR = BYTE & 0xFF;                      // WRITE THE
CHARACTER TO THE DATA REGISTER
}
```

| | |
|--------------|---|
| | Send the number of bytes depicted by len and stored in buffer. An overloaded function based on write. |
| Description | |
| Parameters | The array of string you will send and it's lenght |
| Return Value | null |

Source

```
void writeBytes(uint8_t *BUFFER, uint32_t LENGTH) {
    for (uint32_t i = 0; i < LENGTH; ++i) {
        while (!(USART2_SR & (1 << 7))){}           // WAIT UNTILL THE TRANSMIT
DATA REGISTER IS EMPTY
        USART2_DR = BUFFER[i];                        // WRITE
THE CHARACTER TO THE DATA REGISTER
    }
}
```

| | |
|--------------|--|
| | void print(const char *str) |
| Description | Send the whole string in the input parameter |
| Parameters | Input a string |
| Return Value | Null |

Source

```
void print(const char *str)
{
    while (*str != '\0')
    {
        while (!(USART2_SR & (1 << 7))){}           // WAIT UNTILL THE TRANSMIT DATA
REGISTER IS EMPTY

        USART2_DR = (uint32_t)(*str++);
    }
}
```

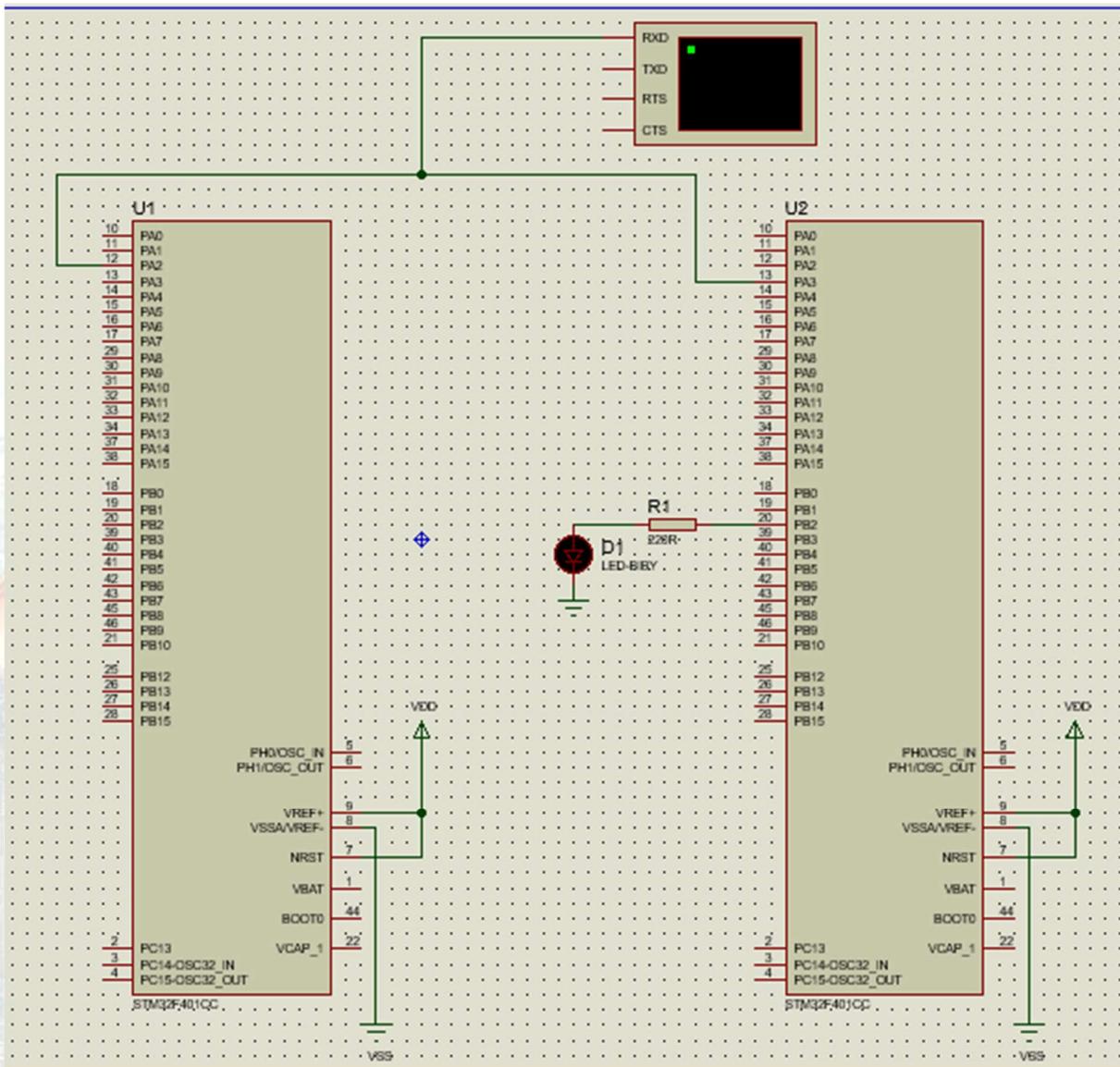
| | |
|---------------------|--|
| | void println(const char *str) |
| Description | Send the whole string in the input parameter with an additional null-terminator |
| Parameters | Send the string with \0 at the end of it |
| Return Value | null |
| | Source |
| | <pre>void println(const char *str) { while (*str != '\0') { while (!(USART2_SR & (1 << 7))){} // WAIT UNTILL THE TRANSMIT DATA REGISTER IS EMPTY USART2_DR = (uint32_t)(*str++); // WRITE THE CHARACTER TO THE DATA REGISTER } while (!(USART2_SR & (1 << 7))); USART2_DR = '\0'; NULL }</pre> |

| | |
|---------------------|---|
| | void delay_ms(unsigned int delay) |
| Description | If required delay in our system |
| Parameters | The desired delay in millesconds |
| Return Value | Null |
| | Source |
| | <pre>void delay_ms(unsigned int delay) { RCC_APB1ENR = (1<<0); // ENABLE TIMER_2 TIM2_CR1 = (1<<7); // SET AUTO RELOAD REGISTER TIM2_CR1 &=~ ((1<<5) (1<<6)); // EDGE ALIGNED MODE (UP OR DOWN ONLY) TIM2_CR1 &=~ (1<<4); // SET THE DIRECTION TO "UP" //COUNTER FREQUENCY = Fclk /PSC (16MHz / (15999+1)) = 1 kHz TIM2_PSC = 15999; TIM2_ARR = 10000; //THE MAX. COUNTER NUMBER (MAX. SECONDS 10s) TIM2_EGR = (1<<0); // INITIALIZE THE COUNTER TO START FROM ZERO TIM2_CR1 = (1<<0); // START THE COUNTER //while (!(TIM2_SR & (1<<0))); // UIF: Update interrupt flag TIM2_CNT = 0; unsigned int target = TIM2_CNT + delay; while(TIM2_CNT < target); // WAIT TO THE COUNTER IN THIS WHILE LOOP TIM2_CR1 &=~ (1<<0); // STOP THE COUNTER }</pre> |

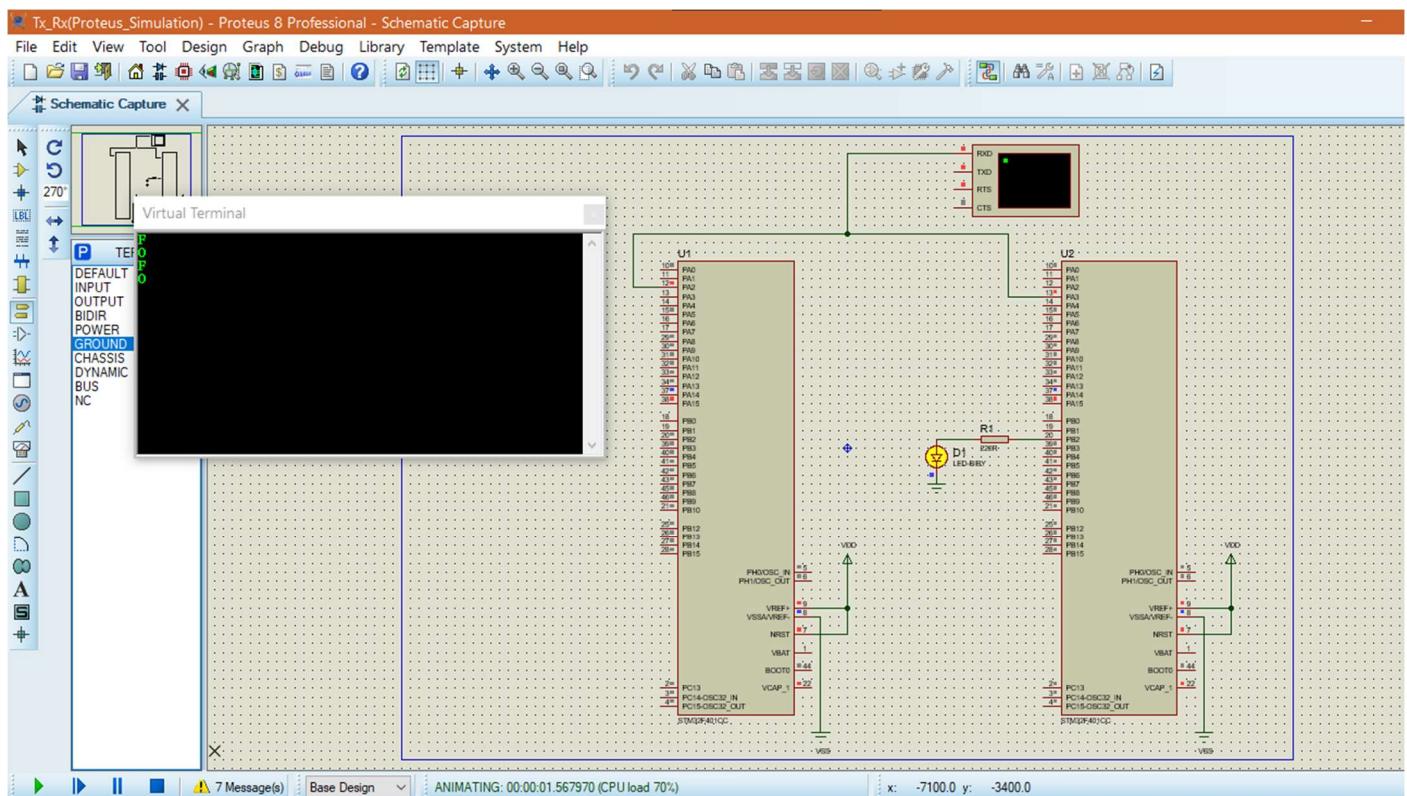


Chapter_4_Proteus Simulation

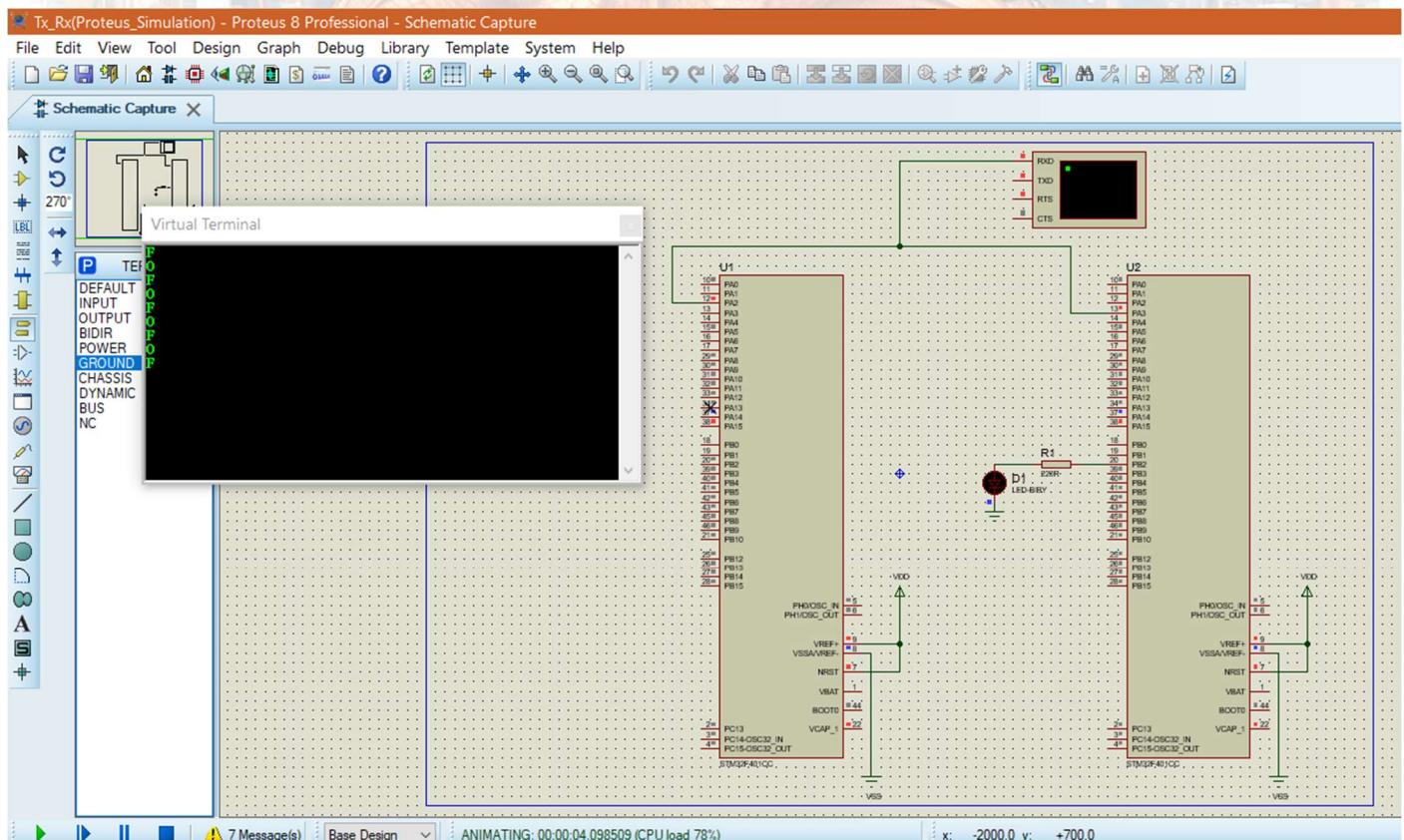
PROTEUS



When the right MCU receive ‘O’ the led is On



When the right MCU receive ‘F’ the led is Off





Thank You