20 UART

20.1 Overview

The features of UART serial communication module are as follows

ÿ Baud rate software configurable

ÿ 4 independent channels (UART0, UART1, UART4, UART5)

ÿ Full double-carrier communication port

ÿ UART has data reception completion/reception error interrupt and displays the error type

ÿ Configurable data length, supports 6, 7, 8, 9 bits

ÿ Configurable stop bits - supports 1 stop bit or 2 stop bits

ÿ Can be configured as infrared modulation output function, and the carrier frequency and carrier duty cycle can be set

ÿ Support DMA

ÿ Support receiving timeout mechanism

20.2 Block Diagram

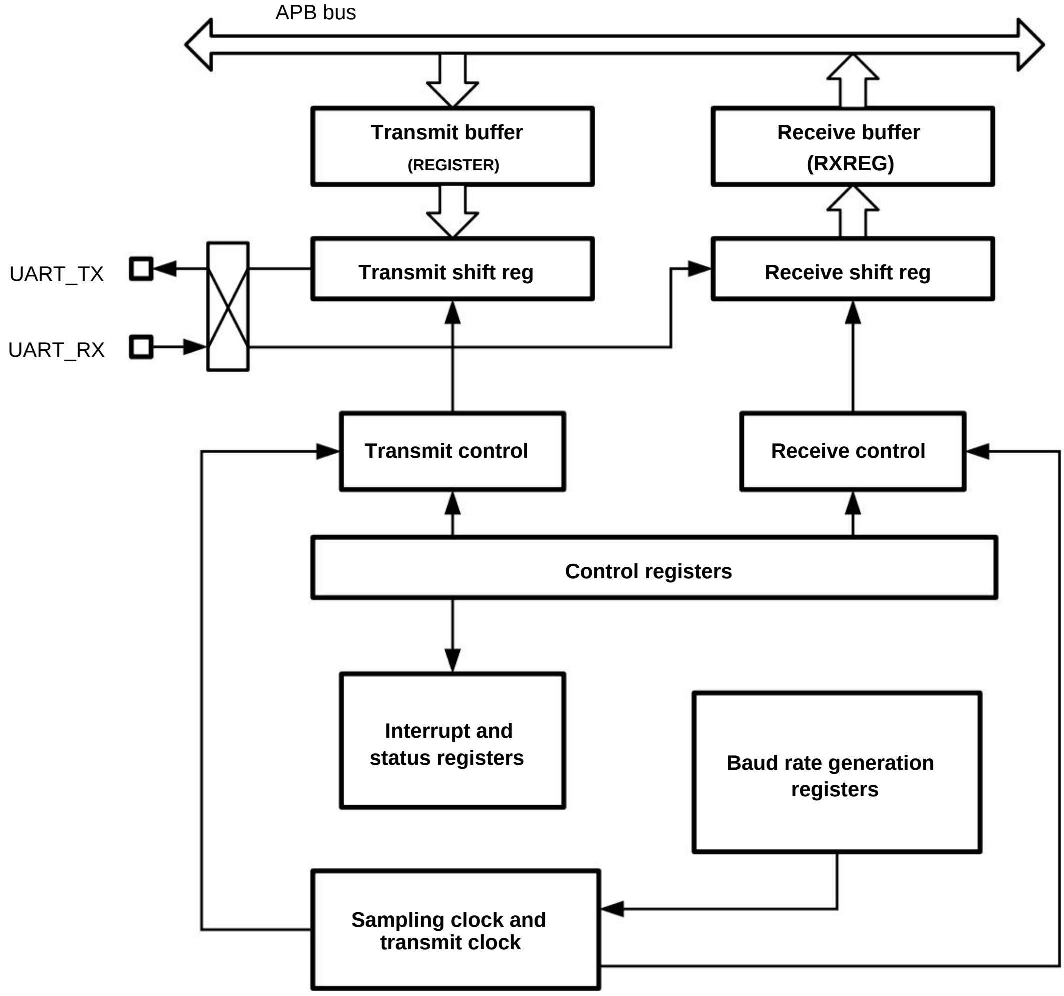

Figure **20-1 UART** interface timing

20.3 Pin Definition

The UART module uses 2 pins to communicate with external devices. The transmit and receive signals of each UART may be mapped to different GPIOs.

The following table shows the pin mapping relationship of FM33LC0x6

Pinout		UARTx	symbol	Function
PA2	PA13	UART0	UART0_RX	Data Reception
PA3	PA14		UART0_TX	Data transmission
PC2	PB13	UART1	UART1_RX	Data Reception
PC3	PB14		UART1_TX	Data transmission
PA0	PB2	UART4	UART4_RX	Data Reception
PA1	PB3		UART4_TX	Data transmission
PC4	PD0	UART5	UART5_RX	Data Reception
PC5	PD1		UART5_TX	Data transmission

The following table shows the pin mapping relationship of FM33LG0x6

Pinout		UARTx	symbol	Function
PA2	PA13	UART0	UART0_RX	Data Reception
PA3	PA14		UART0_TX	Data transmission
PC2	PB13	UART1	UART1_RX	Data Reception
PC3	PB14		UART1_TX	Data transmission
PA0	PB2	UART4	UART4_RX	Data Reception
PA1	PB3		UART4_TX	Data transmission
PC4	PD0	UART5	UART5_RX	Data Reception
PC5	PD1		UART5_TX	Data transmission

Table **20-1 UART** pin list

When the UART function is mapped to multiple pins simultaneously:

ÿ PA2 and PA13 are configured as digital peripheral functions at the same time

ÿ Only the RX signal on PA2 will be input into the module

ÿ PC2 and PB13 are configured as digital peripheral functions at the same time

ÿ Only the RX signal on PA2 will be input into the module

ÿ PC4 and PD0 are configured as digital peripheral functions at the same time

ÿ Only the RX signal on PC4 will be input into the module

ÿ PA0 and PB2 are configured as digital peripheral functions at the same time

ÿ Only the RX signal on PA0 will be input into the module

ÿ When the UART transmit function is mapped to multiple GPIO pins at the same time, these pins will send data at the same time

20.4 **UART** Type Distinction

FM33L0xx integrates several different types of UART (LPUART), the differences are shown in the following table:

UART Features	**UART0/1**	**UART4/5**	**LPUART0/1**
DMA supports half-	AND	AND	AND
duplex/full-duplex infrared	AND	AND	AND
transmission	AND	AND	-
dual clock domain (working clock is independent of bus)	AND	-	AND
sleep wake-up	AND	-	AND
receive timeout	AND	-	-
send delay data	AND	-	-
length	6ÿ7ÿ8ÿ9bits		

Table **20-2 UART** type list

20.5 **UART** Character Description

The basic timing sequence of UART character transmission is shown in the figure below. Each character frame contains at least 1 bit START bit and at least 1 bit STOP bit.

The data length can be configured to be 6~9 bits, and you can choose whether to have a check bit or not.

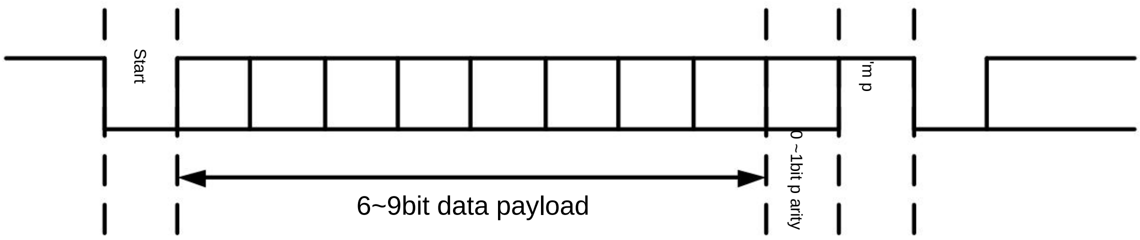

Figure **20-2 UART** character description

The UART supports multiple frame formats, which are controlled by the UARTxCSR.PDSEL register and the UARTxCSR.PARITY register.

surface:

PDSEL	**PARITY**	**Frame format[1]**
00	00	[Start | 7 bits data | Stop]
	01, 10	[Start | 7 bits data | Parity | Stop]
01	00	[Start | 8 bits data | Stop]
	01, 10 00	[Start | 8 bits data | Parity | Stop]
10		[Start | 9 bits data | Stop]
	01, 10 00	[Start | 9 bits data | Parity | Stop]
11		[Start | 6 bits data | Stop]
	01, 10	[Start | 6 bits data | Parity | Stop]

Table **20-3 UART** data frame format

[1]: The Stop bit can be 1 bit or 2 bits, determined by the STOPCFG register.

Note that the PDSEL register is used to configure the data length of the frame. The communication frame length is [start bit + data bit + check bit + stop bit].

20.6 Functional Description

20.6.1 Clock Structure

UART0 and UART1 use a dual clock structure:

ÿ The bus register clock is represented by pclk, which is derived from APBCLK. When the CPU or DMA needs to access the UART internal register

When the device is used, pclk must be enabled

ÿ The data receiving and transmitting clock is represented by uclk. In addition to APBCLK, it can also come from RCHF, SYSCLK, and RCMF.

Can work independently of APBCLK. uclk must be enabled to send and receive data.

The control of Pclk and uclk is completed in the CMU module. The corresponding CMU control registers must be correctly configured before UART communication.

device.

The dual clock structure can make the operation of UART0 and UART1 not limited by the configuration of APBCLK.

When the APBCLK frequency is very high, the UART can still work at a reduced frequency; or vice versa, the CPU can work at a lower frequency.

The lower the frequency, the higher the baud rate of UART data communication will not be affected.

Theoretically, there is no relative constraint between pclk and uclk. uclk can be faster or slower than pclk. However, applications need to pay attention to the

When the frequency difference is large, whether the CPU or DMA can move the data in time.

Unlike UART0 and UART1, UART4 and UART5 use a single clock structure. In this case, uclk=pclk.

The data receiving and transmitting clock also comes from APBCLK.

20.6.2 Bit Receive Sampling

UART oversamples the received data 16 times the baud rate and performs a two-out-of-three majority decision at the middle of each bit.

With high signal noise suppression capability.

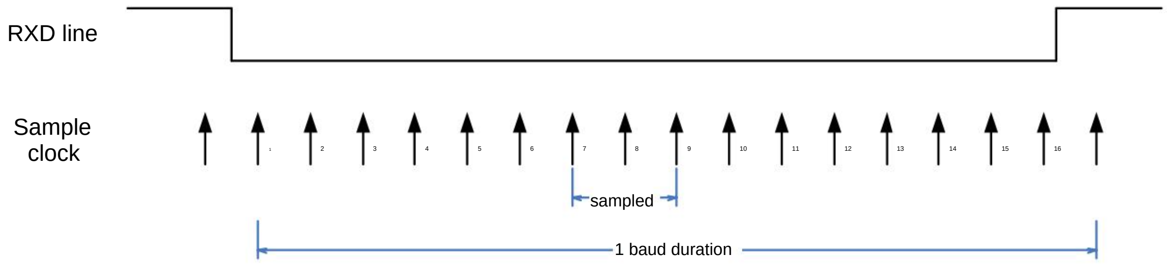

Figure **20-3** -bit receive sampling

The bit received by the receiving shift register is the result of majority judgment. For example, if the result of three samplings is 001, the judgment is 0; if it is 011,

The judgment is 1.

Since UART performs 16 times oversampling on the input signal, the SPBRG configuration must not be less than 16, that is, the UART working clock must be At least 16 times the baud rate.

20.6.3 Data Transmission

In the transmit mode, the serial data transmission circuit of UART mainly includes a transmit shift register (TSR). The function of TSR is to The data to be sent must be written to the send buffer first. When the software sets the TXEN register, if the send buffer If the area is not empty, UART loads the buffer data into TSR and starts shifting output.

Note: Since the register operation clock and the baud rate clock are asynchronous, when sending starts, it is necessary to wait for the baud rate clock to arrive.
TXEN Set to *UART* Start sending *Start* Between the two positions, there is a maximum 1 individual *baud* delay

TXBE and TXSE are the transmit interrupt flags, indicating transmit buffer empty and TSR empty respectively. Software can choose to interrupt at the appropriate time. The send completion interrupt is generated at this point.

Generally, the TSR register is empty at the beginning. To send data, you need to set the baud rate SPBRG and enable the sending module (set TXEN is 1), then write to the TXBUF register to start sending. You can also write to the TXBUF register after setting the baud rate SPBRG. Register, and then set TXEN to enable the transmit module to start data transmission.

If the TXEN bit is cleared to 0, the data transmission will be interrupted and the transmission module will be reset.

The following figure shows an example of UART asynchronous transmission. In this example, the software first writes data to TXBUF and then starts transmission by setting TXEN. deliver.

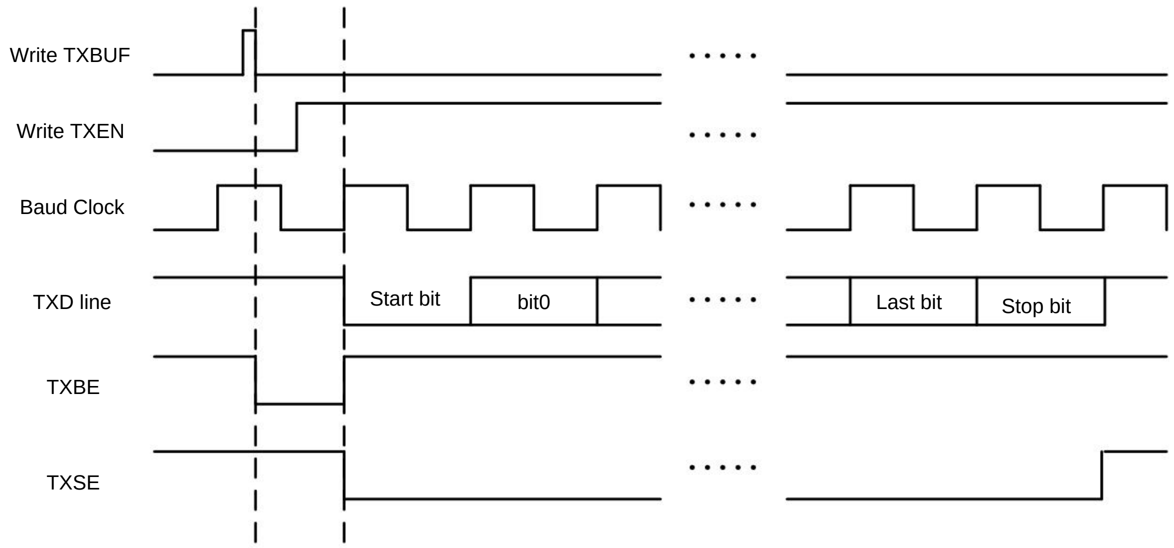

Figure **20-4 UART** asynchronous transmission waveform **1**

The recommended steps in the above figure are as follows:

ÿ Select the appropriate baud rate and initialize SPBRG

ÿ If an interrupt is required, set TXSE_IE or TXBE_IE

ÿ Determine the format of data transmission: set the PDSEL register to determine the length of the data to be sent; set the PARITY register to select Whether to send the parity bit and the parity type, set the STOPSEL register to decide whether to send 1 bit or 2 bits of stop bit

ÿ If you want to send infrared modulated serial data, write the appropriate value to the IRCON register to obtain the corresponding modulation frequency and duty cycle, and set TXIREN

ÿ Write the data to be sent into the TXBUF register (automatically start sending)

ÿ Enable the transmit module: set TXEN

The software can also set TXEN first and then write to TXBUF. In this case, the UART will start the transmission process immediately after the data is written to TXBUF.

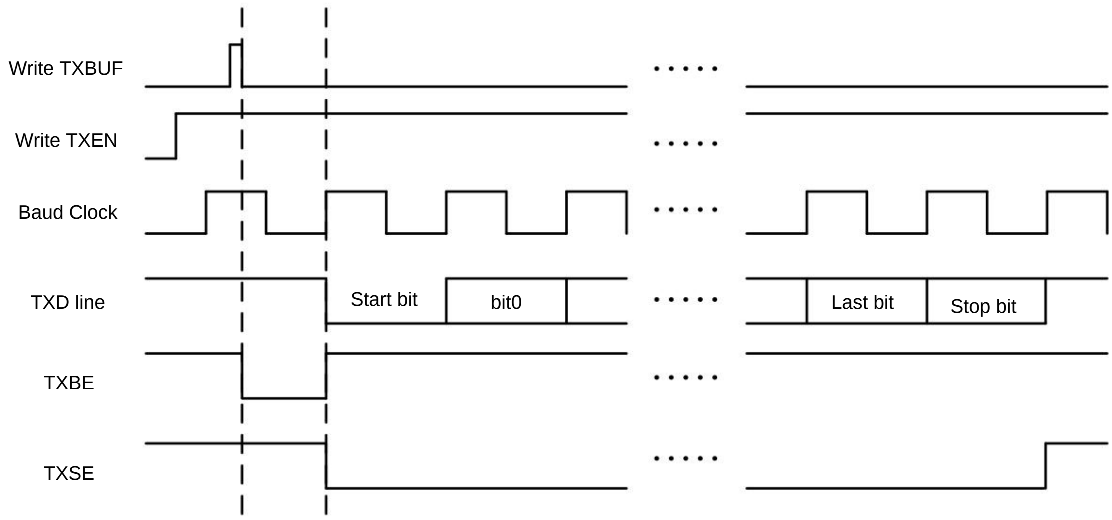

Figure **20-5 UART** asynchronous transmission waveform **2**

When TXBUF is empty, the software can immediately write the next data to be sent to achieve continuous data transmission without intervals.

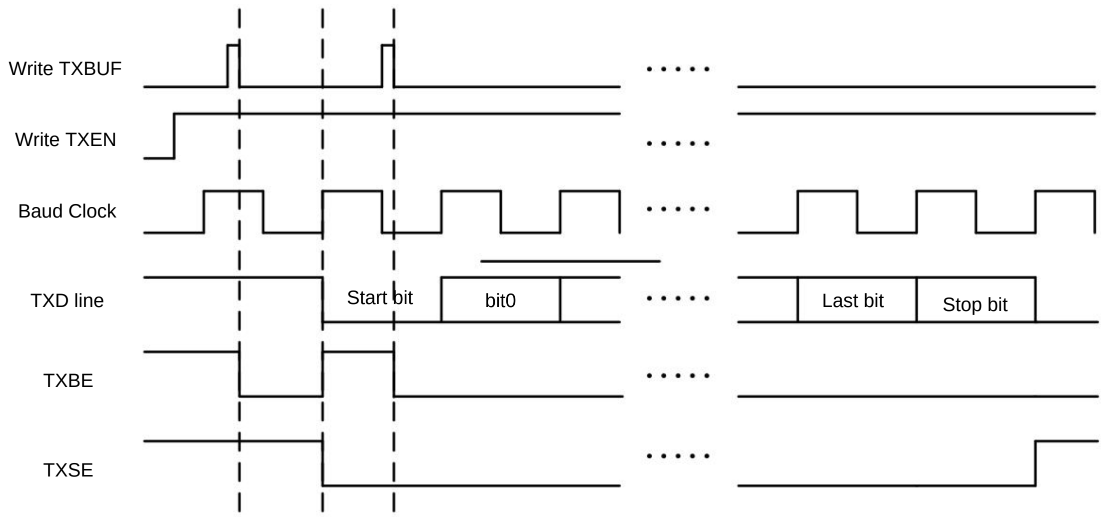

Figure **20-6 UART** asynchronous transmission waveform **3**

20.6.4 Data Reception

The serial data receiving circuit of UART mainly includes a receiving shift register (RSR). When the stop bit is received, RSR will receive the The received data is sent to the receive buffer (RXBUFFER). After the transmission is completed, an interrupt will be triggered each time the received data is sent to the receive buffer. The flag RXBF is set to 1. When the receive buffer is full, RSR will still write a frame of data into the receive buffer after receiving it, which means overwriting There is original data in the buffer, and RXBF is set again. At the same time, a receive overflow error occurs, and OERR is set to 1; software writes 1 or Reading RXBUF can clear the OERR flag.

During the reception process, if the correct stop bit is not detected, a frame format error occurs and FERR is set to 1; if a parity check occurs Wrong, the flag PERR is set to 1.

The recommended asynchronous receive operations are as follows:

ÿ Select the appropriate baud rate and initialize SPBRG

ÿ If an interrupt is required, set RXBF_IE

ÿ Set the format of data reception: Set the PDSEL register to determine the length of the data to be sent; set the PARITY register to select

Whether to send the parity bit and the parity type, set the STOPSEL register to decide whether to send 1 bit or 2 bits of stop bit

ÿ Enable the receiving module: set RXEN

ÿ When a frame is received, the RXBF bit is set to 1. If the RXBF_IE bit is 1, an interrupt will be generated.

ÿ Read the PERR, FERR, and OERR registers to determine if there is a data error or overflow

ÿ Read the received data in the RXBUF register

20.6.5 Using **DMA** for **UART** Transmission and Reception

When the UART module is enabled, the UART module will automatically generate corresponding The application software needs to configure the DMA channel connection in advance, point the specific channel to the UART peripheral, set the RAM access The DMA channel will be enabled. After that, the DMA will automatically respond to the UART request and complete the communication between RAM and UART. data handling.

Application example: using **DMA** for **UART0** reception

ÿ Configure DMA channel 1 or 3 as RXD0

ÿ Set the corresponding channel parameters: RAM pointer address, address increment and decrement, channel priority, transfer length and interrupt settings, etc.

ÿ Enable the corresponding DMA channel

ÿ Configure UART module parameters

ÿ Enable UART module receiving and wait for data reception

ÿ UART automatically generates DMA request after receiving data

ÿ DMA responds to the request, reads the UART receive buffer register, and writes to the specified RAM address

20.6.6 Transmit completion interrupt in **DMA** mode

When UART sends data via DMA, DMA will generate a DMA channel interrupt after the data transfer of the specified length is completed.

When the channel interrupt occurs, the last frame of data has just been written into the UART send buffer and has not yet been sent out.

By configuring the DMATXIFCFG register, it is possible to generate a DMA signal when the DMA transfer is completed and the last frame of data is sent.

Generate a send completion interrupt (buffer empty or shift register empty) to achieve the interrupt after all data are sent out.

Application scenarios of CPU.

The software workflow is described as follows:

ÿ Configure DMA channel for UART transmission

ÿ Disable DMA channel interrupt enable

ÿ Set the DMATXIFCFG register to allow only the last frame of data to generate an interrupt output

ÿ Prepare data to be sent and enable DMA

ÿ Set the UART TXBE_IE or TXSE_IE register to enable interrupt generation

ÿ UART sends continuously until the last frame, and no TXBE or TXSE interrupts are generated during the transmission

ÿ After the last frame is sent, UART generates a TXBE or TXSE interrupt

The following table assumes that the UART sends N frames via DMA:

TXBE_IE **TXSE_IE**	**DMATXIFCFG Frame No.**		**TXBE** **TXSE** is	**UART interrupt**
0	x	1~N	set after each frame is sent and does	not generate
1	0	1~N	After each frame is sent, it is set to no	t generate
	1	1~N-1	After each frame is sent, it is set to no	t generate
		N	Set after each frame is sent	

Table **20-4 DMA** send interrupt

20.7 Baud Rate Generation

20.7.1 Baud Rate Generation

The baud rate factor register is a 16-bit readable and writable register, and its value X is any integer between 16 and 65535.

Baud rate calculation formula:

Baud = FCLK / (SPBRG+1)ÿ

Note: *FCLK*In different *UART* can be different clocks. *UART4* and *UART5* ÿ *FCLK*that is *APBCLK* ;right

At *UART0* and *UART1* ÿ *FCLK* Is independent and *APBCLK* working clock.

To support full-duplex communication, receive and transmit baud rates are generated separately;

The following table shows the baud rates for commonly used system clock frequencies:

Baud	FCLK=16MHz			FCLK=8MHz		
bps	Actual (bps)	Error%	X+1	Actual (bps)	Error%	X+1
300	300.0019	0.000625	53333	299.9963	-0.00125	26667
1200	1200.03	0.0025	13333 -0.005	1199.94	-0.005	6667
2400	2399.88	6667		2400.24	0.010001	3333
4800	4800.48	0.010001	3333	4799.04	-0.02	1667
9600	9598.08	-0.02	1667	9603.842	0.040016	833
19200	19207.68	0.040016	833	19184.65	-0.07994	417
38400	38369.3	-0.07994	417	38461.54	0.160256	208
57600	57553.96	-0.07994	278	57553.96	-0.07994	139
115200	115107.9	-0.07994	139	115942	0.644122	69
230400	231884.1	0.644122	69	228571.4	-0.79365	35
460800	457142.9	-0.79365	35	470588.2	2.124183	17

Baud	FCLK=24MHz			FCLK=32MHz		
bps	Actual (bps)	Error%	X+1	Actual (bps)	Error%	X+1
300	300	0	80000	299.9991	-0.00031	106667
1200	1200	0	20000 0	1199.985	-0.00125	26667
2400	2400		10000	2400.06	0.0025	13333
4800	4800	0	5000	4799.76	-0.005	6667
9600	9600	0	2500	9600.96	0.010001	3333
19200	19200	0	1250	19196.16	-0.02	1667
38400	38400		625	38415.37	0.040016	833
57600	57553.96	0 -0.07994	417	57553.96	-0.07994	556
115200	115384.6	0.160256	208	115107.9	-0.07994	278
230400	230769.2	0.160256	104	230215.8	-0.07994	139
460800	461538.5	0.160256	52	463768.1	0.644122	69

Table **20-5** Baud rate calculation under common clock frequencies

20.7.1 Baud Rate Adaptation

The baud rate adaptation function can be realized by using the Capture function of the Timer. One method to achieve this is to use an external UART device

A frame is sent according to the agreed data content (such as 0xF8), and the Timer counts the high-level pulse width of the frame data, and the MCU reads

The baud rate factor is calculated from the Timer capture result and written into the baud rate generator register as the clock frequency divider for the baud rate generator.

The value X is used. At this time, the receiving state is reset, and the start bit is waited for again, and data is received at the baud rate generated by the written baud rate factor.

Refer to the Timer section.

20.8 Infrared Modulation

The TZBRG register stores an 11-bit frequency division factor X, which can be any integer between 0 and 2047. All UARTs share the same

An infrared modulation frequency generator.

Infrared modulation frequency calculation formula:

FIR = FAPBCLK/ (TZBRG+ 1)

The infrared modulation method is: modulate the infrared frequency when sending data 0, and do not modulate when sending data 1.

The IRFLAG bit in the register controls the polarity of the infrared modulation output. When IRFLAG=0, it is a positive polarity output, suitable for

Suitable for PNP tube driving; when IRFLAG=1, it is negative polarity output, suitable for NPN tube driving.

The TH register is used to configure the infrared modulation duty cycle.

Duty cycle: Y = (TZBRG[10:4] * TH) /(TZBRG+1)

When TH=4'b0000, the duty cycle is Y = (TZBRG[10:1]+1)/(X+1);

When TZBRG[10:4]=7'h00, the duty cycle is Y = TH/(TZBRG[3:0]+ 1); if TH>TZBRG [3:0], the infrared

The modulation clock IRCLK is a fixed high level.

When the infrared modulation polarity is reversed (IRFLAG=1), the duty cycle is also 1-Y

The infrared modulation waveform is shown in the figure below:

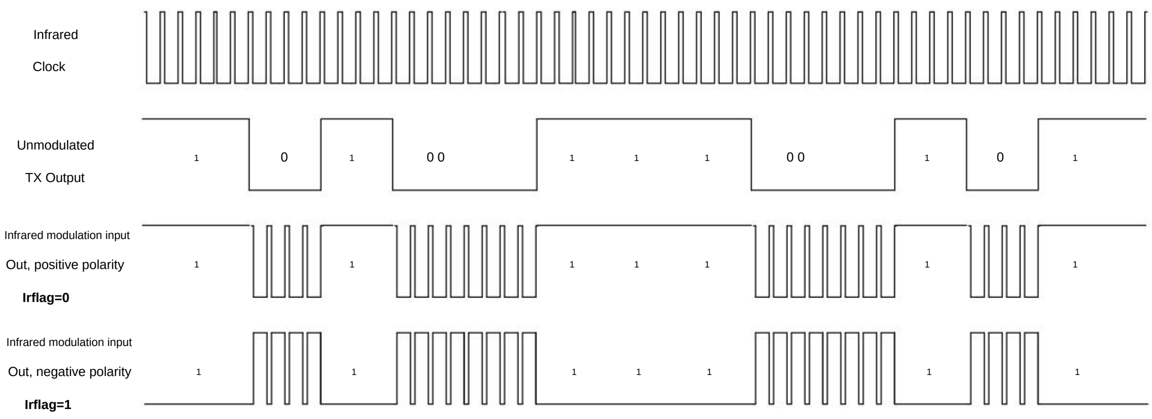

Figure **20-7** Infrared modulation waveform

Regardless of whether the active level is 0 or 1, the duty cycle is defined as the high level length/period.

20.9 Receive Timeout

For time-sensitive applications such as MODBUS, a timeout mechanism is designed. When the RXTOEN register is enabled, the timeout counter is set to

The baud rate clock counts, and each time a complete data frame is received, the timeout counter is cleared and restarted.

The upper limit is software configurable, up to 255 baud.

Note: *UART4* and *UART5* The receive timeout feature is not supported.

20.10 Sending Delay

The TXDLY_LEN register can be used to control the interval between two data frame transmissions in baud.

The interval between the end of the last STOP bit of the previous frame and the start bit of the next frame.

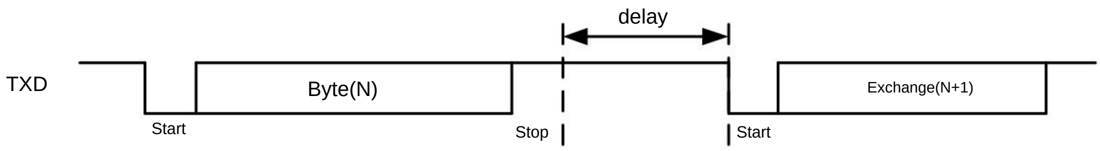

Figure **20-8 UART** transmission delay

Note: *UART4* and *UART5* The send delay feature is not supported.

20.11 Registers

offset address	name	symbol
	UART common register (module starting address: 0x40019C00)	
0x00000000	Infrared modulation configuration register ÿInfrared modulation Control Registerÿ	UART_IRCR
	UART0 register (module start address: 0x40011C00)	
0x00000000	UART0 Control Status Register ÿUART0 Control Status Registerÿ	UART0_CSR
0x00000004	UART0 Interrupt Enable Register ÿUART0 Interrupt Enable Registerÿ	UART0_IER
0x00000008	UART0 Interrupt Flag Register ÿUART0 Interrupt Status Registerÿ	UART0_ISR
0x0000000C	UART0 Timeout and Delay Registers ÿUART0 Time-Out and Delay Registerÿ	UART0_TODR
0x00000010	UART0 Receive Buffer Register ÿUART0 Receive Bufferÿ	UART0_RXBUF
0x00000014	UART0 Transmit Buffer Register ÿUART0 Transmit Bufferÿ	UART0_TXBUF
0x00000018	UATR0 Baud rate generation register ÿUART0 Baud rate Generator Registerÿ	UART0_BGR
	UART1 register (module start address: 0x40012000)	
0x00000000	UART1 Control Status Register	UART1_CSR

offset address	name	symbol
	ÿUART1 Control Status Registerÿ	
0x00000004	UART1 Interrupt Enable Register ÿUART1 Interrupt Enable Registerÿ	UART1_IER
0x00000008	UART1 Interrupt Flag Register ÿUART1 Interrupt Status Registerÿ	UART1_ISR
0x0000000C	UART1 Timeout and Delay Registers ÿUART1 Time-Out and Delay Registerÿ	UART1_TODR
0x00000010	UART1 Receive Buffer Register ÿUART1 Receive Bufferÿ	UART1_RXBUF
0x00000014	UART1 transmit buffer register ÿUART1 Transmit Bufferÿ	UART1_TXBUF
0x00000018	UATR1 Baud rate generation register ÿUART1 Baud rate Generator Registerÿ	UART1_BGR
UART4 register (module start address: 0x4001A000)		
0x00000000	UART4 Control Status Register ÿUART4 Control Status Registerÿ	UART4_CSR
0x00000004	UART4 interrupt enable register ÿUART4 Interrupt Enable Registerÿ	UART4_IER
0x00000008	UART4 interrupt flag register ÿUART4 Interrupt Status Registerÿ	UART4_ISR
0x00000010	UART4 Receive Buffer Register ÿUART4 Receive Bufferÿ	UART4_RXBUF
0x00000014	UART4 transmit buffer register ÿUART4 Transmit Bufferÿ	UART4_TXBUF
0x00000018	UATR4 Baud rate generation register ÿUART4 Baud rate Generator Registerÿ	UART4_BGR
UART5 register (module start address: 0x4001A400)		
0x00000000	UART5 Control Status Register ÿUART5 Control Status Registerÿ	UART5_CSR
0x00000004	UART5 interrupt enable register ÿUART5 Interrupt Enable Registerÿ	UART5_IER
0x00000008	UART5 interrupt flag register ÿUART5 Interrupt Status Registerÿ	UART5_ISR
0x00000010	UART5 Receive Buffer Register ÿUART5 Receive Bufferÿ	UART5_RXBUF
0x00000014	UART5 transmit buffer register ÿUART5 Transmit Bufferÿ	UART5_TXBUF
0x00000018	UATR5 Baud rate generation register ÿUART5 Baud rate Generator Registerÿ	UART5_BGR

20.11.1 Infrared Modulation Configuration Register (UART_IRCR)

name	UART_IRCR							
Offset	0x00000000							
Bit31 Bit	name Bit	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	-							
	U-0							
permission	Bit23 Bit	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
Bit permission	U-0							

Bit 15 Bit Name		Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
IRFLAG Bit Permission		TH				TZBRG[10:8]		
R/W-0		R/W-0000				R/W-000		
	Bit7 Bit	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TZBRG[7:0]							
Bit permission	R/W-1101 0010							

Position No.	Mnemonics	Functional Description
31:16	-	Unimplemented: Read as 0
15	**IRFLAG**	Controls the default output polarity when sending data using infrared modulation (Infra Red) 0: Positive polarity 1: Negative polarity
14:11	**TH**	Infrared duty cycle modulation parameters (Tranmission High Duty)
10:0	**TZBRG**	infrared modulation frequency (Transmission Baud Rate)

20.11.2 UARTx Control Status Register (UARTx_CSR)

name	**UARTx_CSR(x=0,1,4,5)**							
Offset	0x00000000							
bit Bit31 Bit name		Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
bit	-							BUSY
	U-0							R-0
permission bit Bit23		Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
Bit	-						CHICKENS'	FOOD
name	U-0						R/W-0	R/W-0
bit permission bit Bit15		Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Position Name	-			IOSWAP	NEWUP	DMATXIFCFG	BITORD	STOPCFG
Bit permissions	U-0			R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
Bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	PDSEL		PARITY		RXPOL	TXPOL	RXEN	TXEN
Bit permissions	R/W-00		R/W-00		R/W-0	R/W-0	R/W-0	R/W-0

Position No.	Mnemonics	Functional Description
31:25	-	Unimplemented: Read as 0
24	**BUSY**	UART communication flag, read-only (Busy) 1: UART is communicating 0: UART idle
23:18	-	Unimplemented: Read as 0
17	**CHILDREN**	Transmit Infra-red modulation Enable 1: Enable infrared modulation transmission 0: Disable infrared modulation transmission
16	**RXTOEN**	Receive Time-Out Enable 1: Enable the receive timeout function 0: Disable the receive timeout function
15:13	-	Unimplemented: Read as 0

Position No.	Mnemonics	Functional Description
12	**IOSWAP**	RX and TX pins swapped 0: Default pin order (same as package diagram) 1: Swap the pin order
11	**NEWUP**	UART RX falling edge wake-up function enable register (only for UART0 and UART1) ÿÿ(Negedge Wakeupenable) 1: Enable RX falling edge wake-up 0: Disable RX falling edge wake-up
10	**DMATXIFCFG**	DMA transmission completion interrupt enable, only valid when UART transmits via DMA (DMA transmit interrupt enable) 1: When IE=1, after the last frame is sent in DMA mode, the interrupt signal is enabled. After the data frame before the last frame is sent, the interrupt signal output is not allowed 0: Whether to allow interrupt signal output is determined only by IE
9	**BITORD**	Bit Order when sending/receiving data 0ÿLSB first 1ÿMSB first
8	**STOPCFG**	The stop bit width configuration is only valid for the sending frame format. The stop bit width is not determined when receiving. Number (Stop bit config) 0: 1 stop bit 1: 2 stop bits
7:6	**PDSEL**	The data length of each frame is selected; this register is valid for both data sending and receiving (Payload data length Select) 00: 7-bit data 01: 8-bit data 10: 9-bit data 11: 6-bit data
5:4	**PARITY**	Check bit configuration; this register is valid for both data sending and receiving 00: No check digit 01: Even parity 10: Odd parity 11ÿRFU
3	**RXPOL**	Receive data polarity configuration (Receive Polarity) 0: Forward 1: Negate
2	**TXPOL**	Transmit Polarity Configuration 0: Forward 1: Negate
	RXEN	Receive Enable, 1 is valid (Receive Enable)
1 0	**TXEN**	Transmit Enable, 1 is valid (Transmit Enable)

20.11.3 UARTx Interrupt Enable Register (UARTx_IER)

name	**UARTx_IER(x=0,1,4,5)**							
Offset	0x00000004							
Bit31 Bit	Name Bit	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	-							
	U-0							
Permission	Bit23 Bit	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
Name	-							
Bit Permission	U-0							

Bit 15		Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Position Name	-				RXTO_I AND	RXERR_ IE	-	RXBF_I AND
Permission	U-0				R/W-0	R/W-0	U-0	R/W-0
bits	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Position Name	NEWUP_ IE	-					TXBE_IE	TXSE_IE
bit permission	R/W-0	U-0					R/W-0	R/W-0

Position No.	Mnemonics	Functional Description
31:12	-	Unimplemented: Read as 0
11	**RXTO_IE**	Receive time-out interrupt enable, 1 is valid (Receive Time-Out Interrupt Enable) (Only UART0 and UART1 are valid)
10	**RXERR_IE**	receive error interrupt enable, 1 is valid (Receive Error Interrupt Enable)
9	-	Unimplemented: Read as 0
8	**RXBF_IE**	Receive buffer full interrupt enable, 1 is valid (Receive Buffer Full Interrupt Enable)
7	**NEWUP_IE**	RX falling edge asynchronous detection interrupt enable, 1 is valid (Negedge Wakeup Interrupt Enable)
6:2	-	Unimplemented: Read as **0**
1	**TXBE_IE**	Transmit buffer empty interrupt enable, 1 is valid (Transmit Buffer Empty Interrupt Enable)
0	**TXSE_IE**	The transmit buffer is empty and the transmit shift register is empty interrupt is enabled, 1 is valid (Transmit Shift register Empty Interrupt Enable)

20.11.4 UARTx Interrupt Flag Register (UARTx_ISR)

name	**UARTx_ISR(x=0,1,4,5)**							
Offset	0x00000008							
bit	Bit31 Bit name	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
bit	-							
permission	U-0							
bit	Bit23 Bit name bit	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
	-				PERR	FERR	OERR	
permission	U-0				R/W-0	R/W-0	R/W-0	
bit	Bit15 Bit name bit	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
	-			RXTO	-		RXBF	
permission	U-0			R/W-0	U-0		R/W-0	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Bit Name	NEWKF Bit	-					TXBE	TXSE
Permission	R/W-0	U-0					R-0	R/W-0

Position No.	Mnemonics	Functional Description
31:19	-	Unimplemented: Read as 0
18	**Perr**	Parity error interrupt flag, hardware set, software write 1 to clear (Parity Error,write 1 to clear)
17	**FERR**	Frame format error interrupt flag, hardware set, software write 1 to clear (Frame Error flag,write 1 to clear)

Position No.	Mnemonics	Functional Description
16	**OERR**	Receive buffer overflow error interrupt flag. When the receive buffer is full, a new Set when data is received; set by hardware, cleared when software writes 1 or reads RXBUF When receiving overflow, the original data in the receiving buffer is overwritten by the new data. (RX buffer Overflow Error flag,write 1 to clear)
15:12	-	Unimplemented: Read as 0
11	**RXTO**	Receive timeout interrupt flag, hardware set, software write 1 to clear (Receive Time-Out flag,write 1 to clear) (Only UART0 and UART1 are valid)
10:9	-	Unimplemented: Read as 0
8	**RXBF**	Receive buffer full interrupt flag, hardware set, software write 1 or read RXBUF ÿÿ (Receive Buffer Full flag write 1 to clear)
7	**NEWKF**	RX falling edge asynchronous detection interrupt flag, hardware set, software write 1 to clear (Negedge Wakeup Flag write 1 to clear) (Only UART0 and UART1 are valid)
6:2	-	Unimplemented: Read as 0
1	**TXBE**	Transmit buffer empty interrupt flag, set by hardware, cleared when software writes to TXBUF (Transmit Buffer Empty flag)
0	**TXSE**	The transmit buffer is empty and the shift register transmits the complete interrupt flag, which is set by hardware and written by software. 1 or cleared when software writes to the send buffer (Transmit Shift register Empty flag,write 1 to clear)

20.11.5 UARTx Timeout and Delay Register (UARTx_TODR)

name	**UARTx_TODR(x=0,1)**							
Offset	0x0000000C							
Bit31 Bit	name bit	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	-							
	U-0							
permission	Bit23 Bit	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
bit	U-0							
permission Bit15		Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Bit	TXDLY_LEN							
name	R/W-0000 0000							
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	RXTO_LEN							
permission	Bit name bit permission R/W-1111 1111							

Position No.	Mnemonics	Functional Description
31:16	-	Unimplemented: Read as 0
15:8	**TXDLY_LEN**	transmit delay, maximum 255baud (Transmit Delay Length)
7:0	**RXTO_LEN**	Receive time-out overflow length, maximum 255baud (Receive Time-Out Length)

20.11.6 UARTx Receive Buffer Register (UARTx_RXBUF)

name	UARTx_RXBUF(x=0,1,4,5)
Offset	0x00000010

Bit31 Bit	name bit	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	-							
	U-0							
permission	bit23 Bit	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
name	-							
bit	U-0							
permission	bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Position Name	-							RXBUF[8]
Bit permissions	U-0							R-0
Bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	RXBUF[7:0]							
Bit permissions	R-0000 0000							

Position No.	Mnemonics	Functional Description
31:9	-	Unimplemented: Read as 0
8:0	**RXBUF**	receives data buffer register data (Receive buffer) When 7 bits are sent and received, the received 7 bits of data are stored in RXBUF[6:0]

20.11.7 UARTx Transmit Buffer Register (UARTx_TXBUF)

name	UARTx_TXBUF(x=0,1,4,5)							
Offset	0x00000014							
bit Bit31	Bit name	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
bit	-							
	U-0							
permission	bit Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
Bit	-							
name	U-0							
bit permission	bit Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Position Name	-							TXBUF[8]
Bit permissions	U-0							R/W-0
Bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name	TXBUF[7:0]							
Bit permissions	R/W-0000 0000							

Position No.	Mnemonics	Functional Description
31:9	-	Unimplemented: Read as 0
8:0	**TXBUF**	transmit data buffer register data (Transmit Buffer) When 7 bits are sent and received, the 7 bits of data sent are written to TXBUF[6:0]

20.11.8 UATRx Baud Rate Generation Register (UARTx_BGR)

name	UARTx_BGR(x=0,1,4,5)							
Offset	0x00000018							
Bit31 Bit	Name	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
	-							

bit	U-0							
permission	bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
bit	-							
name	U-0							
permission	bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
bit	SPBRG[15:8]							
name	R/W-0000 0011							
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	SPBRG[7:0]							
permission	bit name permission R/W-0100 0001							

Position No.	Mnemonics	Functional Description
31:16	-	Unimplemented: Read as 0
15:0	**SPBRG**	Baud Rate Generator Register Value (Serial Port Baud Rate Generation)

For details on baud rate calculation, see chapter 36.5 Baud rate generation

Note: When SPBRG <= 0x000F, UARTDIV=16'H000F;

When SPBRG > 0x000F, UARTDIV = SPBRG;