21 LPUART

21.1 Overview

LPUART is an enhanced asynchronous serial communication interface. Its working clock can be selected from bus clock (APBCLK), 32768Hz crystal clock (XTLF), 32KHz low power ring oscillator clock (LPOSC), high frequency ring oscillator clock (RCHF), system clock (SYSCLK), low power

When the working clock is XTLF or LPOSC, it can support up to 9600 wave.

When receiving data at high bit rates, the LPUART consumes very little power and can work in Sleep/DeepSleep mode.

Features:

ÿ Asynchronous data transmission and reception

ÿ 2 independent LPUARTs (LPUART0, LPUART1)

ÿ Standard UART frame format

- ÿ 1 bit start bit
- ÿ Configurable data length, supports 6, 7, 8, 9 bits
- ÿ Odd, even or no parity bit
- ÿ 1 or 2 stop bits

ÿ Programmable data polarity

ÿ When the working clock is XTLF or LPOSC, it supports data transmission and reception in Sleep/DeepSleep mode

ÿ Interrupt flag

- ÿ Receive Buffer is full
- ÿ Receive Buffer overflow
- ÿ Receive frame format error
- ÿ Receive check digit error
- ÿ START detection
- ÿ Data Matching
- ÿ Send Completed

ÿ Wake up the chip in sleep mode

- ÿ RXD falling edge wake-up
- ÿ Start bit detection wake-up
- ÿ 1 byte reception completed wake up
- ÿ 1-byte data match wakeup

ÿ Support DMA (not supported in Sleep/DeepSleep/RTCBKP mode)

21.2 Block Diagram

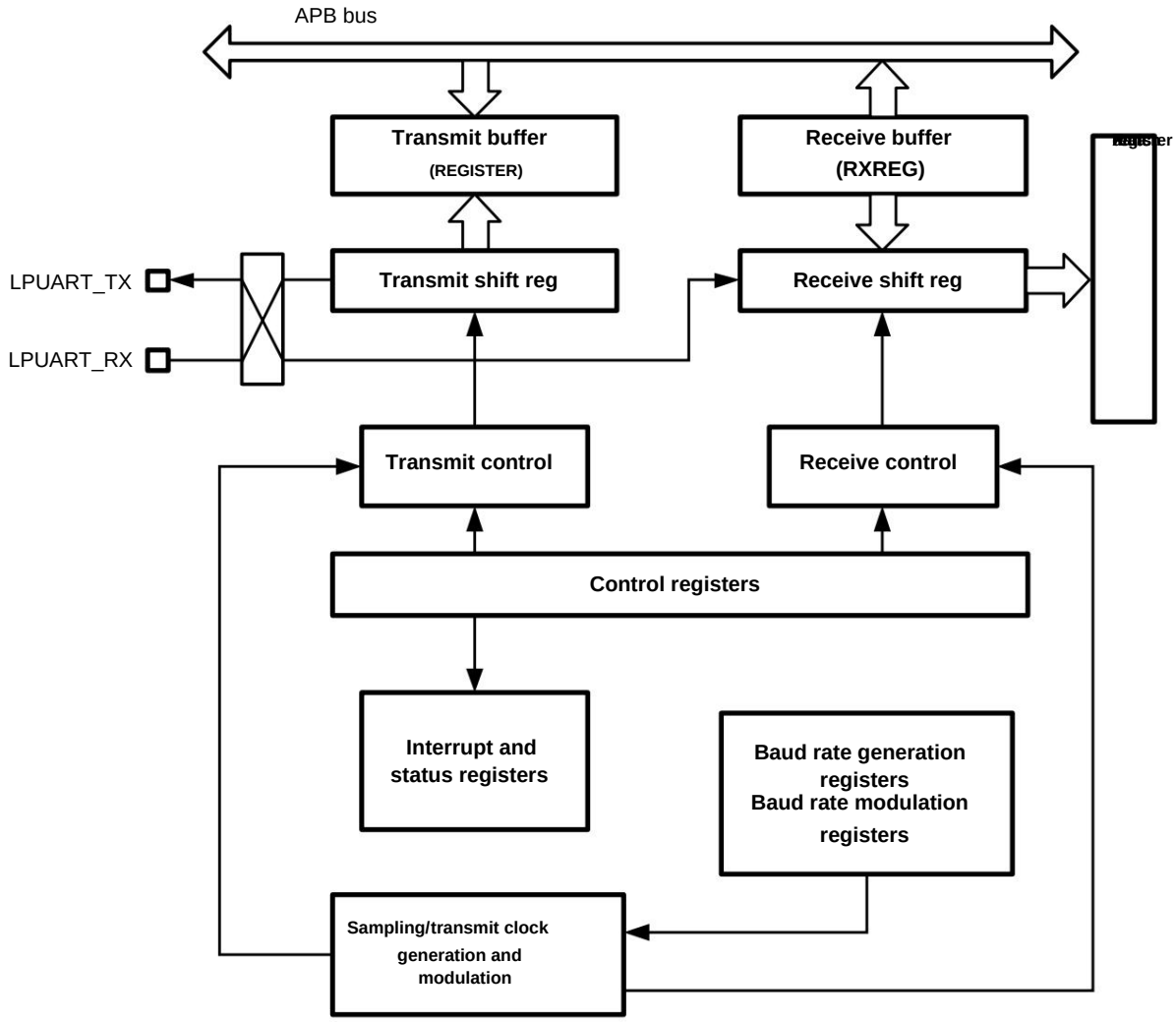


Figure **21-1LPUART** structure block diagram

21.3 Pin Definition

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

Pinout		UARTx	symbol	Function
PA13	PA2	LPUART0	LPUART0_RX	Data Reception
PA14	PA3		LPUART0_TX	Data transmission
PC2 PB13[1]		LPUART1	LPUART1_RX	Data Reception
PC3 PB14[1] [1]			LPUART1_TX	Data transmission

Available only on FM33LG0x6 models

When the LPUART 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 PC2 will be input into the module

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

21.4 Working Clock

LPUART uses a clock independent of APBCLK to send and receive data. Before working, you need to configure the relevant registers in the CMU module.

LPUART can work with XTLF or LPOSC. Because LPOSC has low precision, it is necessary to set the LPOSC to XTLF before using it for LPUART communication.

A clock calibration must be performed first to calibrate LPOSC to within +/-1%.

LPOSC calibration should not use XTLF, because XTLF can be used for communication when XTLF is available.

The circuit should be operated using RCHF, and the recommended reference input is 8MHz.

In ACTIVE mode, LPUART can also use RCHF to work. In this case, the clock accuracy will be higher than LPOSC to obtain better

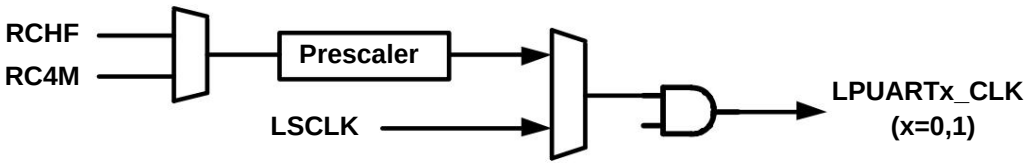
When using RCHF, the prescaler circuit pre-divides RCHF to obtain a frequency close to 32768Hz.

The clock frequency, for example, when RCHF is 8M/16M/24M, the prescaler division factor should be 244/488/732 respectively.

In ACTIVE and LP ACTIVE modes, the LPUART can use the RCMF clock to operate. The temperature coefficient of RCMF is relatively

RCHF is poor, so it is recommended to perform RCMF calibration before LPUART works.

The LPUART working clock structure is shown in the figure below. This part of the functions and registers are implemented in the CMU module.



21.5 Character Description

The basic timing sequence of LPUART 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.

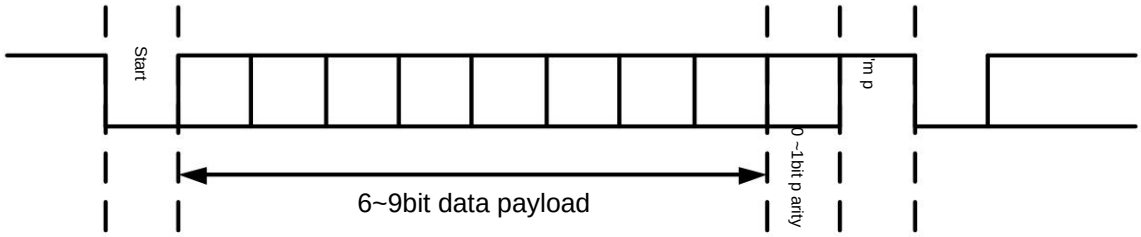


Figure **21-2** Character Description

LPUART supports multiple frame formats, which are controlled by the LPUARTxCSR.PDSEL register and the LPUARTxCSR.PARITY register.

See the table below:

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	[Start | 8 bits data | Parity | Stop]
10	00	[Start | 9 bits data | Stop]
	01, 10	[Start | 9 bits data | Parity | Stop]
11	00	[Start | 6 bits data | Stop]
	01, 10	[Start | 6 bits data | Parity | Stop]

Table **21-1LPUART** 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].

21.6 Functional Description

21.6.1 Bit Receive Sampling

In low-power serial mode, the working clock frequency is only about 32Khz. At this time, the standard serial port cannot support 9600bps communication, so it is necessary to

It is necessary to introduce bit modulation design.

The software needs to properly configure the modulation control register MCTL according to the different communication baud rates. The recommended configuration parameter table is as follows:

Baud	MCTL												
	Bit0 (start)	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7	Bit8	Bit9	Bit10	Bit11	Bit12
9600	0	1	0	0	1	0	1	0	1	0	1	0	0
4800	1	1	0	1	1	1	1	1	0	1	1	1	1
2400	1	1	0	1	1	0	1	1	0	1	1	0	1
1200	0	1	0	0	1	0	0	1	0	0	1	0	0
600	0	1	1	0	1	0	1	1	0	1	1	0	1
300	0	1	0	0	0	0	1	0	0	0	0	1	0

Table **21-2 LPUART** bit modulation parameter table

21.6.2 Receiving Process

ÿ Configure LPUBAUD register to determine baud rate

ÿ Select appropriate modulation parameters according to the baud rate and configure the MCTL register

ÿ Configure the LPUCON register to select frame format, polarity, interrupt parameters, etc.

ÿ Configure the LPUEN register to enable reception

ÿ Waiting for interrupt event

21.6.3 Sending Process

ÿ Configure LPUBAUD register to determine baud rate

ÿ Select appropriate modulation parameters according to the baud rate and configure the MCTL register

ÿ Configure the LPUCON register to select frame format, polarity, interrupt parameters, etc.

ÿ Configure the LPUEN register to enable transmission

ÿ Waiting for interrupt event

21.6.4 Using **DMA** for **LPUART** Transmission and Reception

When the LPUART module is enabled, the LPUART module will automatically generate a

The application software needs to configure the DMA channel connection in advance, point the specific channel to the LPUART peripheral, and set

The DMA channel will be enabled. After that, the DMA will automatically respond to the LPUART request and complete the RAM and

Data transfer between LPUARTs.

Application example: Using **DMA** for **LPUART1** reception

ÿ Configure DMA channel 0 or 3 as LPUART1_RX

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 LPUART1 module parameters

ÿ Enable LPUART1 module reception enable LPUEN.RXEN=1, wait for data reception

ÿ After receiving data, LPUART1 automatically generates DMA request

ÿ DMA responds to the request, reads the LPUART1 receive buffer register, and writes to the specified RAM address

21.6.5 Data reception wake-up in sleep mode

LPUART supports data reception and chip wake-up in Sleep and DeepSleep modes. At this time, the chip power consumption is extremely low and keeps

The RXD pin is monitored until a specific event arrives to wake up the chip and exit sleep mode.

ÿ Configure LPUBAUD register to determine baud rate

ÿ Select appropriate modulation parameters according to the baud rate and configure the MCTL register

ÿ Configure the LPUCON register, select the frame format and polarity, and select the wake-up event as the START bit, a

Frame reception completed, one frame data matched or RXD falling edge detected

ÿ Configure the LPUEN register to enable reception

ÿ The software enters Sleep/DeepSleep

21.6.6 Data **DMA** Transmission and Reception in **LPRUN** Mode

Through LPUART and DMA, the software can realize the automatic transmission and reception of a certain amount of data by LPUART in LPRUN mode without CPU intervention.

At the same time, the power consumption of the whole chip is guaranteed to be less than 10uA under typical conditions.

ÿ Configure LPUBAUD register to determine baud rate

ÿ Select appropriate modulation parameters according to the baud rate and configure the MCTL register

ÿ Configure the LPUCON register to select frame format, polarity, interrupt parameters, etc.

ÿ Configure DMA channel control register and select LPUART transceiver

ÿ If data needs to be sent, write the data to be sent to the specified location in RAM

ÿ Configure DMA data transmission and reception length and RAM pointer

ÿ Select the system main clock as LSCLK

ÿ Software enters LPRUN

ÿ Configure the LPUEN register to enable sending and receiving

ÿ If the CPU has no additional work, it can actively enter WFI/WFE and wait for interrupt wake-up

21.6.7 Transmit Completion Interrupt in **DMA** Mode

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

But when the channel interrupt occurs, the last frame of data has just been written into the LPUART 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 to send LPUART

ÿ Disable DMA channel interrupt enable

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

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

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

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

TXBE_IE **TXSE_IE**	**DMATXIFCFG Frame No.**		**TXBE** **TXSE** is	**LPUART 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 not generate	
	1	1~N-1	After each frame is sent, it is set to not generate	
		N	Set after each frame is sent	

Table **21-3 LPUART DMA** interrupt description

21.7 Registers

offset address	name	symbol
LPUART0 register (module start address: 0x40010400)		
0x00000000	LPUART0 Control Status Register ÿLPUART0 Control Status Registerÿ	LPUART0_CSR
0x00000004	LPUART0 interrupt enable register ÿLPUART0 Interrupt Enable Registerÿ	LPUART0_IER
0x00000008	LPUART0 interrupt flag register ÿLPUART0 Interrupt Status Registerÿ	LPUART0_ISR
0x0000000C	LPUART0 Baud Rate Modulation Register ÿLPUART0 Baud rate Modulation Registerÿ	LPUART0_BMR
0x00000010	LPUART0 Receive Buffer Register ÿLPUART0 Receive Buffer Registerÿ	LPUART0_RXBUF
0x00000014	LPUART0 transmit buffer register ÿLPUART0 Transmit Buffer Registerÿ	LPUART0_TXBUF
0x00000018	LPUART0 Data Match Register ÿLPUART0 data Matching Registerÿ	LPUART0_DMR
LPUART1 register (module start address: 0x40018400)		
0x00000000	LPUART1 Control Status Register ÿLPUART1 Control Status Registerÿ	LPUART1_CSR
0x00000004	LPUART1 interrupt enable register ÿLPUART1 Interrupt Enable Registerÿ	LPUART1_IER
0x00000008	LPUART1 Interrupt Flag Register ÿLPUART1 Interrupt Status Registerÿ	LPUART1_ISR
0x0000000C	LPUART1 Baud Rate Modulation Register ÿLPUART1 Baud rate Modulation Registerÿ	LPUART1_BMR
0x00000010	LPUART1 Receive Buffer Register ÿLPUART1 Receive Data Registerÿ	LPUART1_RXBUF
0x00000014	LPUART1 transmit buffer register ÿLPUART1 Transmit Data Registerÿ	LPUART1_TXBUF
0x00000018	LPUART1 Data Match Register ÿLPUART1 data Matching Registerÿ	LPUART1_DMR

21.7.1LPUARTx Control Status Register (LPUARTx_CSR)

name	LPUARTx_CSR(x=0,1)							
Offset	0x00000000							
Bit31 Bit	name	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24
								BUSY
	U-0							R-0
Permission	bit Bit23	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16
Position Name					WKBYTE_CFG		RXEV	
	U-0				R/W-0	U-0	R/W-00	
Permission	bit Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8
Position Name					IOSWAP	DMATXIFCFG	BITORD	STOPCFG
Permission	U-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**	LPUART communication flag, read-only (Busy) 1: LPUART is communicating 0: LPUART idle
23:20	-	Unimplemented: Read as 0
19	**WKBYTE_CFG**	Data reception wake-up condition configuration (Wakeup Byte Config) 1: Wake-up is triggered only after receiving 1 byte and the parity check and STOP bit are correct Interrupt 0: After receiving 1 byte, do not check the parity bit and STOP bit, and directly trigger the wake-up interrupt
18	-	Unimplemented: Read as 0
17:16	**RXEV**	Wake-up interrupt event configuration, used to control the event under which the CPU is provided with a wake-up interrupt (Receive Wakeup Event) 00: START bit detection wake-up 01: 1 byte data received 10: Received data matches successfully 11: RXD falling edge detection
15:12	-	Unimplemented: Read as 0
11	**IOSWAP**	RX and TX pin swapping (IO swapping) 0: Default pin order (same as package diagram) 1: Swap the pin order
10	**DMATXIFCFG**	DMA transmission completion interrupt is enabled, which is only available when LPUART transmits via DMA. ÿ (DMA Transmit Interrupt Config) 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**	Select the data length for each frame; 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 (Parity) 00: No check digit 01: Even parity 10: Odd parity 11ÿRFU
3	**RXPOL**	Receive data polarity configuration (Receive Polarity) 0: Forward 1: Negate

Position No.	Mnemonics	Functional Description
2	**TXPOL**	Transmit Polarity Configuration 0: Forward 1: Negate
1	**RXEN**	Receive Enable, 1 is valid (Receive Enable)
0	**TXEN**	Transmit Enable, 1 is valid (Transmit Enable)

21.7.2 **LPUARTx** Interrupt Enable Register (LPUARTx _**IER)**

name	LPUARTx_IER(x=0,1)							
Offset	0x00000004							
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	
permission	U-0							
bit Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
Position Name			RXEV_I AND		RXERR_ IE		RXBF_I AND	
Bit permissions	U-0		R/W-0	U-0	R/W-0	U-0	R/W-0	
Bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
name							TXBE_IE	TXSE_IE
Bit permissions	U-0						R/W-0	R/W-0

Position No.	Mnemonics	Functional Description
31:13	-	Unimplemented: Read as 0
12	**RXEV_IE**	Receive wake-up event interrupt enable, 1 is valid (Receive Event Interrupt Enable)
11	-	Unimplemented: Read as 0
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)
7:2	-	Unimplemented: Read as 0
1	**TXBE_IE**	Transmit Buffer Empty Interrupt Enable, 1 is valid (Transmit Buffer Empty Interrupt Enable)
0	**TXSE_IE**	Transmit buffer empty and transmit shift register empty interrupt enable, 1 valid (Transmit Shift register Interrupt Enable)

21.7.3 **LPUARTx** Interrupt Flag Register (LPUARTx _**ISR)**

name	LPUARTx_ISR(x=0,1)							
Offset	0x00000008							
Bit31 Bit Name Bit	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	
							RXEVF	
	U-0						R/W-0	
Permission Bit23 Bit	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
Name				FERRY OIL	OIL			

Bit	U-0				R/W-0	R/W-0	R/W-0	R/W-0
permission Bit15 Bit	Bit14	Bit13	Bit12		Bit11	Bit10	Bit9	Bit8
name								RXBF
Bit	U-0							R/W-0
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
							TXBE	TXSE
permission Bit name Bit permission	U-0						R/W-0	R/W-0

Position No.	Mnemonics	Functional Description
31:25	-	Unimplemented: Read as 0
24	**RXEVF**	Receive wake-up event interrupt flag, hardware set, software write 1 to clear (Receive Event Interrupt Flag, write 1 to clear) The interrupt flag trigger source is configured by the LPUxCR.RXEV register.
23:20	-	Unimplemented: Read as 0
19	**FIXED**	Transmit buffer overflow error, hardware set, software write 1 to clear (Transmit Overflow Error flag, write 1 to clear) When the data in the transmit buffer has not yet entered the shift register for transmission, the software sends the data to the transmit buffer. Storing or writing new data will trigger the TXOV flag to be set.
18	**Perr**	Parity Error interrupt flag, hardware set, software write 1 to clear (Parity Error flag,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)
16	**OERR**	Receive buffer overflow error interrupt flag. When the receive buffer is full, a new Set when data is received; hardware sets, software writes 1 to clear (Receive Buffer Overflow Error flag, write 1 to clear)
15:9	-	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)
7:2	-	Unimplemented: Read as 0
1	**TXBE**	Transmit Buffer Empty interrupt flag, hardware set, cleared when writing to TXBUF (Transmit Buffer Empty flag, write 1 to clear)
0	**TXSE**	The transmit buffer is empty and the transmit shift register is empty interrupt flag, set by hardware and written by software Or the transmit data is cleared when it is loaded into the shift register (Transmit Shift register Empty flag, write 1 to clear)

21.7.4LPUARTx Baud Rate Modulation Register (LPUARTx _**BMR)**

name	LPUARTx_BMR(x=0,1)							
Offset	0x0000000C							
Bit31 Bit name bit	Bit30	Bit29	Bit28	Bit27	Bit26	Bit25	Bit24	
			MCTL[12:8]					
	U-0		R/W-00000					
permission Bit23 Bit	Bit22	Bit21	Bit20	Bit19	Bit18	Bit17	Bit16	
name	MCTL[7:0]							
bit	R/W-0000 0000							
permission Bit15	Bit14	Bit13	Bit12	Bit11	Bit10	Bit9	Bit8	
Bit								
name	U-0							
bit permission	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0

Bit name		BAUD
and permission	U-0	R/W-000

Position No.	Mnemonics	Functional Description
31:29	-	Unimplemented: Read as 0
28:16	**MCTL**	LPUART Bit Width Modulation Control Signal for Each Bit
15:3	-	Unimplemented: Read as 0
2:0	**BAUD**	Baud rate control (bps) 000ÿ9600 001ÿ4800 010ÿ2400 011ÿ1200 100ÿ600 101/110/111ÿ300

21.7.5 **LPUARTx** Receive Buffer Register (LPUARTx _**RXBUF)**

name	LPUARTx_RXBUF(x=0,1)							
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** Receive Data Buffer Register (Receive Buffer)	

21.7.6**LPUARTx** Transmit Buffer Register (LPUARTx _**TXBUF)**

name	LPUARTx_TXBUF(x=0,1)							
Offset	0x00000014							
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								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 (Transmit Buffer)	

21.7.7**LPUARTx** Data Match Register (LPUARTx _**DMR)**

name	LPUARTx_DMR(x=0,1)							
Offset	0x00000018							
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							MATD[8]	
name	U-0						R/W-0	
bit	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
	MATD[7:0]							
permission Bit name bit permission	R/W-0000 0000							

Position No.	Mnemonics	Functional Description
31:9	-	Unimplemented: Read as 0
8:0	**MATD**	The first frame receives comparison data. If RXEV=10, when the first frame data is received Same as MATD, triggers RXEVF interrupt, which can be used for data in sleep mode. Wake up according to the received data. (Matched Data)