

# **UCD30xx UART Module Programmer's Manual**

# Table of contents

<b>OVERVIEW</b>	<b>3</b>
<b>UART FRAME FORMAT</b>	<b>4</b>
<b>ASYNCHRONOUS TIMING MODE</b>	<b>4</b>
<b>UART INTERRUPTS</b>	<b>6</b>
Transmit Interrupt	6
Receive Interrupt	7
Error Interrupts	7
<b>USEFUL C STATEMENTS</b>	<b>9</b>
<b>UART REGISTERS</b>	<b>10</b>
UART Control Register 0 (UARTCTRL0)	10
UART Receive Status Register (UARTRXST)	11
UART Transmit Status Register (UARTTXST)	11
UART Control Register 3 (UARTCTRL3)	12
UART Interrupt Status Register (UARTINTST)	13
UART Baud Divisor High Byte Register (UARTHBAUD)	13
UART Baud Divisor Middle Byte Register (UARTMBAUD)	14
UART Baud Divisor Low Byte Register (UARTLBAUD)	14
UART Receive Buffer (UARTRXBUF)	14
UART Transmit Buffer (UARTTXBUF)	14
UART I/O Control Register (UARTIOCTRLRX, UARTIOCTRLTX)	15

## Overview

This section provides an overview of the UART module.

Table 1 contains a brief description of the UART, lists its significant pins and described interrupts

Table 1. UART Module Overview

<b>Description</b>	The UART module is a universal asynchronous receiver-transmitter that implements the standard nonreturn to zero format. The UART module can be used to communicate, for example, through an RS-232 port or over a K-line.
<b>Pins</b>	SCI_Rx    UART receive pin SCI_Tx    UART transmit pin
<b>Interrupts</b>	The UART has three interrupts: transmit, receive, and error. Each interrupt can be individually enabled.
<b>Features</b>	Standard universal asynchronous receiver-transmitter (UART) communication Supports full- or half-duplex operation Standard nonreturn to zero (NRZ) format Double-buffered receive and transmit functions Configurable frame format of 3 to 13 bits per character based on the following: Data word length programmable from one to eight bits Parity programmable for zero or one parity bit, odd or even parity Stop programmable for one or two stop bits The 24-bit programmable baud rate supports 224 different baud rates provide high accuracy baud rate selection. Four error flags and six status flags provide detailed information regarding UART events.

## UART Frame Format

The UART uses a programmable frame format. All frames consist of the following:

- One start bit
- One to eight data bits
- Zero or one parity bit
- One or two stop bits

The frame format for both the transmitter and receiver is programmable through the bits in the UARTCTRL0 register.

Both receive and transmit data is in nonreturn to zero (NRZ) format, which means that the transmit and receive lines are at logic high when idle.

Each frame transmission begins with a start bit, in which the transmitter pulls the SCI line low (logic low).

Following the start bit, the frame data is sent and received least significant bit first (LSB).

A parity bit is present in every frame when the PARITY ENA bit (UARTCTRL0.5) is set. The value of the parity bit depends on the number of one bits in the frame and whether odd or even parity has been selected via the PARITY bit (UARTCTRL0.6).

All frames include one stop bit, which is always a high level. This high level at the end of each frame is used to indicate the end of a frame to ensure synchronization between communicating devices. Two stop bits are transmitted if the STOP bit (UARTCTRL0.7) is set. The example shown in Figure below use one stop bit per frame.



## Asynchronous Timing Mode

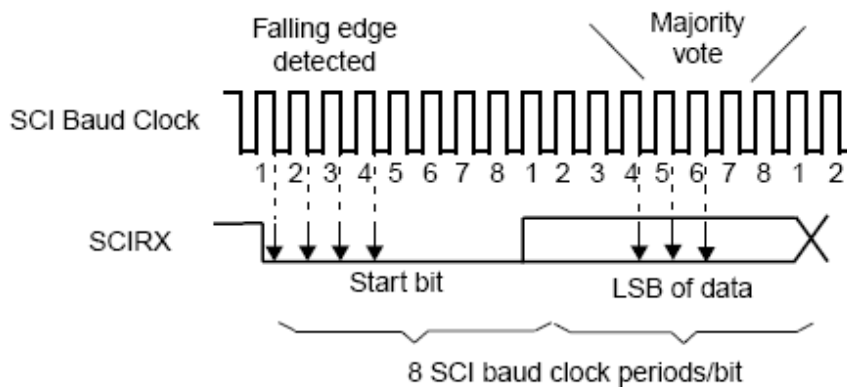
In the asynchronous timing mode, each bit in a frame has a duration of 8 UART baud clock periods. Each bit therefore consists of 8 samples (one for each clock period). When the UART is using asynchronous mode, the baud rates of all communicating devices must match as closely as possible. Receive errors result from devices communicating at different baud rates.

With the receiver in the asynchronous timing mode, the UART detects a valid

start bit if the first four samples after a falling edge on the SCI\_RX pin are of logic level 0. As soon as a falling edge is detected on SCI\_RX, the UART assumes that a frame is being received and synchronizes itself to the bus. The UART module has been designed to provide some protection from noise causing unintended start bits or incorrect data. Without protection, a noise spike that brings an idle receive line low may be interpreted as a start bit. The UART prevents this by requiring a start bit to bring the SCI\_RX line low for at least four contiguous UART baud clock periods. If any of the receive samples during the first four UART baud clock periods is not a logic low, then the UART does not consider this a start bit and considers the receive line idle. When another falling edge is detected, the UART checks for a valid, noise-free start bit.

When a valid start bit is detected, the UART determines the value of each bit by sampling the SCI\_RX line value during the fourth, fifth, and sixth UART baud clock periods. A majority vote of these samples is used to determine the value stored in the UART receiver shift register. By sampling in the middle of the bit, the UART reduces errors caused by propagation delays and rise and fall times. By taking a majority vote, the UART reduces the likelihood of data corruption caused by data line noise. Figure x illustrates how the receiver samples a start bit and a data bit in asynchronous timing mode.

The transmitter transmits each bit for a duration of 8 UART baud clock periods. During the first clock period for a bit, the transmitter shifts the value of that bit onto the SCI\_TX pin. The transmitter then holds the current bit value on SCI\_TX for 8 UART baud clock periods.



$$\text{Asynchronous baud value} = \left( \frac{\text{ICLK Frequency}}{8(\text{BAUD} + 1)} \right)$$

## UART Interrupts

The UART receiver and transmitter can be controlled by interrupts. The receive and transmit interrupts allow efficient operation of the UART by reading and writing character information to and from the UART as new data arrives and when old data has just been sent. The RXRDY flag (UARTRXST.2) used by the receiver indicates that new data is available to be read. The receiver also has various error interrupts that indicate when a particular error condition is active. An active error interrupt condition is indicated by the RXERR flag in UARTRXST. However, the exact source of an error interrupt can be determined by checking the parity error (PE), frame error (FE), overrun error (OE), break-detect (BRKDT), and wake-up (WAKEUP) flags also located in UARTRXST. Additionally, the transmitter uses the TXRDY flag (UARTTXST.2) to indicate that the transmitter is ready for new data to be written that will be sent to the bus.

Transmit, receive, and error interrupts are enabled or disabled through separate interrupt-enable bits. When not enabled, the interrupts are not asserted; however, polled operation of the UART is still possible because the interrupt flags continue to indicate module events.

The UART module generates three interrupt requests to the UCD30xx system module: one each for transmitter, receiver, and error interrupts. Each of these interrupts must also be configured in the UCD30xx system module before operation. Normally, the error interrupt has the highest priority, the receiver interrupt has the next highest priority, and the transmitter generally has the lowest priority. This prioritizing scheme reduces the possibility of missed error conditions and receiver overrun. For interrupt priority levels on a specific device, consult the device data sheet.

### *Transmit Interrupt*

The following paragraphs describe how a CPU interrupt can be initiated by a transmit ready condition.

The transmit ready (TXRDY) flag is set when the UART transfers the contents of UARTTXBUF to the shift register, UARTTXSHF. The TXRDY flag indicates that UARTTXBUF is ready to be loaded with more data. In addition, the UART sets the TX EMPTY bit if both the UARTTXBUF and UARTTXSHF registers are empty.

Transmit interrupts are enabled by the **TX\_INT\_ENA** (UARTCTRL3.3) bit. If the **TX\_INT\_ENA** bit (UARTCTRL3.3) is set, then a transmit interrupt is generated when the TXRDY flag goes high.

Writing data to the UARTTXBUF register clears the TXRDY bit. When this data has been moved to the UARTTXSHF register, the TXRDY bit is set again. The

interrupt request can be suspended by clearing the **TX\_INT\_ENA** bit; however, when the **TX\_INT\_ENA** bit is again set to 1, the TXRDY interrupt is asserted again. The transmit interrupt request can be eliminated until the next series of values is written to UARTRXBUF by disabling the transmitter via the TXENA bit (UARTTXST.0 = 0), an UART software reset, or by a device hardware reset.

### ***Receive Interrupt***

The receive ready (RXRDY) flag is set when the UART transfers newly received data from SCIRXSHF to UARTRXBUF. The RXRDY flag therefore indicates that the UART has new data to be read. Receive interrupts are enabled by the **RX\_INT\_ENA** bit. If the **RX\_INT\_ENA** bit (UARTCTRL3.4) is set when the UART sets the RXRDY flag, then a receive interrupt is generated.

### ***Error Interrupts***

The UCD30xx's UART module provides hardware indication of error conditions to provide you with information about the status of module operation. According to the data being assembled by the receiver, the UART monitors received data for errors and sets the parity error (PE), framing error (FE), and/or the break-detect (BRKDT) flag when these conditions are detected. In addition, the UART sets the overrun error (OE) flag if a transfer of new data from UARTRXSHF to UARTRXBUF overwrites unread data in UARTRXBUF. (If both overrun and parity errors occur, only the overrun error flag is set.) The UART sets the wake-up flag (WAKEUP) if bus activity on the RX line either prevents power-down mode from being entered, or RX line activity causes an exit from power-down mode. Each of these flags is located in the receiver status (UARTRXST) register.

The RXERR flag, also in the UARTRXST register, is the logical OR of the parity error, framing error, overrun error, wake-up, and break-detect flags. This UART Data Transfer feature allows software that is polling for receiver errors to check only one bit, which will indicate if any or all of the five error conditions are active.

Error interrupts are controlled by three separate enable bits: RXERR INT ENA, BRKDT INT ENA, and WAKEUP INT ENA.

If RXERR INT ENA (UARTCTRL3.0) is set, an error interrupt is generated when the receiver detects either a parity, framing, or overrun error. The break-detect interrupt is enabled separately.

If BRKDT INT ENA (UARTCTRL3.1) is set, an error interrupt is generated if the receiver detects a break condition. A break condition occurs when the SCI\_RX line remains continuously low (active) for at least 10 bits immediately following a missed stop bit.

If WAKEUP INT ENA (UARTCTRL3.2) is set, an error interrupt is generated when bus activity on the RX line either prevents power-down mode from being entered or RX line activity causes an exit from power-down mode.



## Useful C statements

```
Uart1Regs.UARTCTRL3.bit.SW_RESET = 0;  
// means, software reset UART before initializing UART
```

```
Uart1Regs.UARTCTRL0.bit.DATA_SIZE = 7;  
// means, data is set to 8 bits size
```

```
Uart1Regs.UARTCTRL0.bit.STOP = 1;  
// means, expect/generate 2 stop bits
```

```
Uart1Regs.UARTCTRL0.bit.SYNC_MODE = 1;  
// means, work in asynchronous mode (always set this bit)
```

```
Uart1Regs.UARTCTRL3.bit.CLOCK = 1;  
// means, internal clock selected
```

```
Uart1Regs.UARTHBAUD.byte.BAUD_DIV_H = 0;  
Uart1Regs.UARTMBAUD.byte.BAUD_DIV_M = 0;  
Uart1Regs.UARTLBAUD.byte.BAUD_DIV_L = 50;  
// means, use a divider of 50 for generation of 38400 bps baud rate, assuming the  
// Iclk = 15.625 MHz
```

```
Uart1Regs.UARTIOCTRLTX.bit.IO_FUNC = 1;  
// means, enable transmit pin
```

```
Uart1Regs.UARTIOCTRLTX.bit.IO_DIR = 1;  
// means, enable transmit pin as output
```

```
while(Uart1Regs.UARTTXST.bit.TX_RDY == 0);  
// means, wait till the transmission is ended and the output buffer is empty
```

```
Uart1Regs.UARTTXBUF.all = data;  
// means, put out a data byte for transmission
```

```
if(Uart1Regs.UARTRXST.bit.RX_RDY == 1)  
{  
    rx_byte = Uart1Regs.UARTRXBUF.byte.RXDAT; //clears RXRDY flag  
}  
// means, if a new data byte received read this byte into a variable
```

## UART Registers

### UART Control Register 0 (UARTCTRL0)

Address FFF7D800

Bit Number	7	6	5	4
Bit Name	STOP	PARITY	PARITY_ENA	SYNC_MODE
Access	R/W	R/W	R/W	R/W
Default	0	0	0	0

Bit Number	3	2:0
Bit Name	ADDR_MODE	DATA_SIZE
Access	R/W	R/W
Default	0	000

**Bit 7: STOP** – Configures stop bits for each frame

0 = One STOP bit included in each frame (Default)

1 = Two STOP bits included in each frame

**Bit 6: PARITY** – Sets odd or even parity

0 = Even parity (Default)

1 = Odd parity

**Bit 5: PARITY\_ENA** - Enables parity transmission

0 = No parity bit included in each frame (Default)

1 = One parity bit included in each frame

**Bit 4: SYNC\_MODE** – Selects between Synchronous mode and Asynchronous mode

0 = Synchronous (Default)

1 = Asynchronous ( Needs to be set for UART operation)

**Bit 3: ADDR\_MODE** – Selects between Idle and Address Bit Mode

0 = IDLE Line mode with no Address bit (Default)

1 = Address Bit mode with one Address bit

**Bits 2-0: DATA\_SIZE** – Determines the TX and RX byte size

000 = No Data (Default)

001 = 1 bit of data

010 = 2 bits of data

011 = 3 bits of data

100 = 4 bits of data

101 = 5 bits of data

110 = 6 bits of data

111 = 7 bits of data

### ***UART Receive Status Register (UARTRXST)***

**Address FFF7D804**

Bit Number	4	3	2	1	0
Bit Name	RX_IDLE	SLEEP	RX_RDY	RX_WAKE	RX_ENA
Access	R	R/W	R	R	R/W
Default	-	0	-	-	0

**Bit 4: RX\_IDLE** –RX Idle status bit

0 = Not in Rx Idle State

1 = Rx Idle detected

**Bit 3: SLEEP** – Sleep Mode Configuration

0 = Sleep Mode disabled (Default)

1 = Sleep Mode enabled

**Bit 2: RX\_RDY** – UART Receiver ready status bit

0 = UART Receiver not ready

1 = UART Receiver ready

**Bit 1: RX\_WAKE** – UART Receiver wake status bit

0 = UART Receiver has not entered wakeup state

1 = UART Receiver has entered wakeup state

**Bit 0: RX\_ENA** – Turns on UART Receiver

0 = UART Receiver disabled (Default)

1 = UART Receiver enabled

### ***UART Transmit Status Register (UARTTXST)***

**Address FFF7D808**

Bit Number	7	6	5:4
Bit Name	CONTINUE	LOOPBACK	RESERVED
Access	R/W	R/W	-
Default	0	0	00

Bit Number	3	2	1	0
Bit Name	TX_EMPTY	TX_RDY	TX_WAKE	TX_ENA
Access	R	R	R/W	R/W
Default	-	-	0	0

**Bit 7: CONTINUE** – Configure operation in suspend mode

0 = Stop transmitting on suspend (Default)

1 = Continue transmitting after initiation of suspend

**Bit 6: LOOPBACK** – Loopback Mode Configuration

0 = Normal mode (Default)

1 = Loopback Mode

**Bit 5-4: RESERVED** – Unused bits – Default to 00

**Bit 3: TX\_EMPTY** – Transmit buffer status

0 = Transmit buffer is not empty

- 1 = Transmit buffer is empty
- Bit 2: TX\_RDY** – Transmitter Ready  
0 = UART Transmitter is not ready  
1 = UART Transmitter is ready to transmit data
- Bit 1: TX\_WAKE** – TX wake control bit  
0 = UART Transmitter Wakeup disabled (Default)  
1 = UART Transmitter Wakeup enabled
- Bit 0: TX\_ENA** – Turns on TX module  
0 = UART Transmitter Disabled (Default)  
1 = UART Transmitter Enabled

### ***UART Control Register 3 (UARTCTRL3)***

**Address FFF7D80C**

Bit Number	7	6	5	4
Bit Name	SW_RESET	POWERDOWN	CLOCK	RX_INT_ENA
Access	R/W	R/W	R/W	R/W
Default	0	0	0	0

Bit Number	3	2	1	0
Bit Name	TX_INT_ENA	WAKEUP_INT_ENA	BRKDT_INT_ENA	ERR_INT_ENA
Access	R/W	R/W	R/W	R/W
Default	0	0	0	0

- Bit 7: SW\_RESET** – Software reset for UART Transmitter/Receiver  
0 = Disables Software Reset (Default)  
1 = Enables Software Reset
- Bit 6: POWERDOWN** – Power-down Transmitter/Receiver Control  
0 = Disables Power-down mode (Default)  
1 = Enables Power-down mode
- Bit 5: CLOCK** – UART Clock Select  
0 = Selects external clock (Default)  
1 = Selects internal clock
- Bit 4: RX\_INT\_ENA** – Enables the interrupts from UART Receiver  
0 = Disables interrupts from UART Receiver (Default)  
1 = Enables interrupts from UART Receiver
- Bit 3: TX\_INT\_ENA** – Enables the interrupts from UART Transmitter  
0 = Disables interrupts from UART Transmitter (Default)  
1 = Enables interrupts from UART Transmitter
- Bit 2: WAKEUP\_INT\_ENA** – Enables the wakeup interrupt from UART  
0 = Disables Wakeup Interrupt (Default)  
1 = Enables Wakeup Interrupt
- Bit 1: BRKDT\_INT\_ENA** – Enables the Broken Circuit interrupt from UART Receiver  
0 = Disables Broken Circuit Interrupt (Default)  
1 = Enables Broken Circuit Interrupt
- Bit 0: ERR\_INT\_ENA** – Enables UART Receiver Error Interrupt

0 = Disables UART Receiver Error Interrupt (Default)  
 1 = Enables UART Receiver Error Interrupt

### ***UART Interrupt Status Register (UARTINTST)***

***Address FFF7D810***

Bit Number	7	6	5	4
Bit Name	BUS_BUSY	RESERVED	FRAME_ERR	OVERRUN_ERR
Access	R	-	R	R
Default	-	0	-	-

Bit Number	3	2	1	0
Bit Name	PARITY_ERR	WAKEUP_INT	BRKDT_INT	RX_ERR
Access	R	R	R	R
Default	-	-	-	-

**Bit 7: BUS\_BUSY** – UART Receiver Busy Indicator

0 = UART Receiver ready to accept new frame  
 1 = UART Receiver currently processing message

**Bit 6: RESERVED** – Unused bit – Default to 0

**Bit 5: FRAME\_ERR** – UART Receiver Framing Error

0 = No framing error found within incoming data message  
 1 = Indicates the incoming data message had a framing error

**Bit 4: OVERRUN\_ERR** – UART Receiver Buffer Overflow

0 = No overflow condition found in receive buffer  
 1 = Indicates the receive buffer has overflowed

**Bit 3: PARITY\_ERR** – UART Receiver Parity Error

0 = No parity error found on the incoming data message  
 1 = Indicates a parity error found on the incoming data message

**Bit 2: WAKEUP\_INT** – UART Receiver Wakeup Interrupt

0 = No Wakeup Interrupt received from UART Receiver  
 1 = Wakeup Interrupt received from UART Receiver

**Bit 1: BRKDT\_INT** – UART Receiver Broken Circuit Interrupt

0 = No Broken Circuit interrupt received from UART Receiver  
 1 = Indicates a Broken Circuit interrupt received from UART Receiver

**Bit 0: RX\_ERR** – UART Receiver Error

0 = No UART Receiver Errors detected  
 1 = Frame Error or Overrun error or Parity Error or Broken Circuit error received from UART Receiver

### ***UART Baud Divisor High Byte Register (UARTBHAUD)***

***Address FFF7D814***

<b>Bit Number</b>	<b>7:0</b>
<b>Bit Name</b>	BAUD_DIV_H
<b>Access</b>	R/W
<b>Default</b>	0000_0000

**Bits 7-0: BAUD\_DIV\_H** - Sets the high byte of the 24 bit baud rate selector

### ***UART Baud Divisor Middle Byte Register (UARTMBAUD)***

**Address FFF7D818**

<b>Bit Number</b>	<b>7:0</b>
<b>Bit Name</b>	BAUD_DIV_M
<b>Access</b>	R/W
<b>Default</b>	0000_0000

**Bits 7-0: BAUD\_DIV\_M** - Sets the middle byte of the 24 bit baud rate selector

### ***UART Baud Divisor Low Byte Register (UARTLBAUD)***

**Address FFF7D81C**

<b>Bit Number</b>	<b>7:0</b>
<b>Bit Name</b>	BAUD_DIV_L
<b>Access</b>	R/W
<b>Default</b>	0000_0000

**Bits 7-0: BAUD\_DIV\_L** - Sets the low byte of the 24 bit baud rate selector

### ***UART Receive Buffer (UARTRXBUF)***

**Address FFF7D824**

<b>Bit Number</b>	<b>7:0</b>
<b>Bit Name</b>	RXDAT
<b>Access</b>	R
<b>Default</b>	-

**Bits 7-0: RXDAT** – Contains the last data byte received from the UART Receiver

### ***UART Transmit Buffer (UARTTXBUF)***

**Address FFF7D828**

<b>Bit Number</b>	<b>7:0</b>
<b>Bit Name</b>	TXDAT
<b>Access</b>	R/W
<b>Default</b>	0000_0000

**Bits 7-0: TXDAT** – Contains the data byte to be transmitted by the UART Transmitter

### ***UART I/O Control Register (UARTIOCTRLRX, UARTIOCTRLTX)***

**Address FFF7D830 – UART I/O (RX) Control Register**

**Address FFF7D834 – UART I/O (TX) Control Register**

<b>Bit Number</b>	<b>3</b>	<b>2</b>	<b>1</b>	<b>0</b>
<b>Bit Name</b>	DATA_IN	DATA_OUT	IO_FUNC	IO_DIR
<b>Access</b>	R	R/W	R/W	R/W
<b>Default</b>	-	0	0	0

**Bit 3: DATA\_IN** – Data received from pin when configured as GPIO

**Bit 2: DATA\_OUT** – Data transmitted to pin when configured as GPIO

**Bit 1: IO\_FUNC** – Selects the function for UART pins

0 = GPIO mode (Default)

1 = Normal operation for SCI\_RX/SCI\_TX

**Bit 0: IO\_DIR** – Pin direction when configured as GPIO

0 = Input (Default)

1 = Output