# UART Datasheet

By Alex Clunan

ECE 4435 Computer Architecture & Design

May 3, 2025

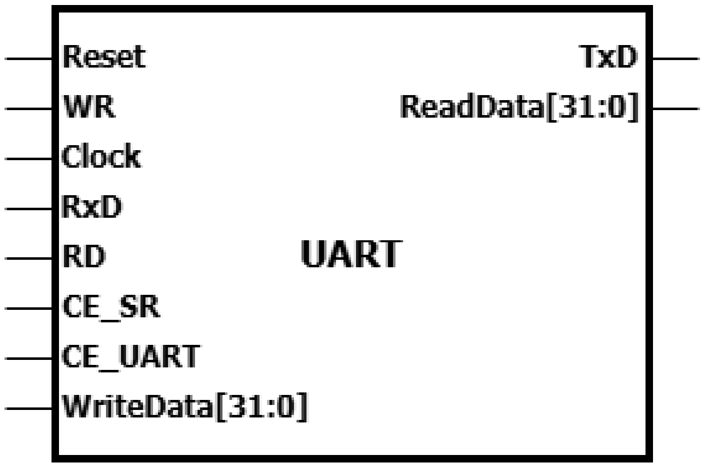
## Description

This module implements a basic Universal Asynchronous Receiver/Transmitter (UART) for serial communication. It includes a baud rate generator, a status register, and supports memory-mapped read and write operations. The UART handles both data transmission and reception, with separate chip enables for accessing the status register and the UART data interface. The baud rate is configurable by changing the frequency of the input clock, and the status register provides information on transmit and receive states, as well as error conditions.

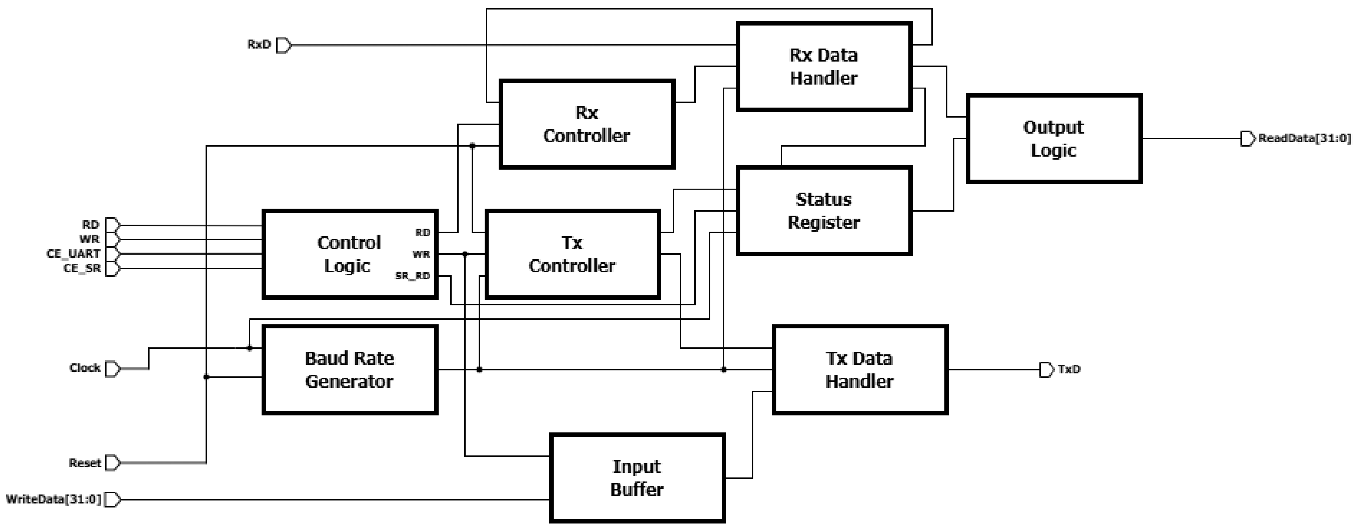
## Features

- **Single Module UART/BRG**
- **External Clock Input**
- **On Chip Baud Rate Generator**
- **32-bit Input/Output Buses**
- **Status Register for Error Identification and Transmit/Receive Status**
- **RISC-V Compatible**

## Symbol Layout



## Block Diagram



## Pin Description

| PIN | TYPE (I/O) | DESCRIPTION |
|---|---|---|
| Reset | I | Asynchronous active-high reset. Initializes all internal state and logic. |
| WR | I | Active high write enable signal. Used in conjunction with CE_UART to transmit data. |
| Clock | I | System clock input. Used for timing and generating the baud rate clocks. |
| RxD | I | Serial data input for receiving asynchronous communication. |
| RD | I | Active high read enable signal. Used with CE_UART or CE_SR to read data or status. |
| CE_SR | I | Active high chip enable for the status register. Enables access to status register's output. |
| CE_UART | I | Active high chip enable for the UART data interface. Used with WR or RD for access. |
| WriteData[31:0] | I | 32-bit data input bus. Provides transmit data to the UART. |

| | | |
|---|---|---|
| TxD | O | Serial data output for asynchronous transmission. |
| ReadData[31:0] | O | 32-bit output bus. Provides received data or status register output depending on access. |

## Functional Elements

### Transmitter

The UART transmitter serializes parallel data from the Din input and drives it on the TxD output line using asynchronous serial protocol formatting. This includes appending a start bit, transmitting the data bits LSB-first, optionally inserting a parity bit, and concluding with a stop bit.

Transmission begins when a write (WR) pulse is received. The input data is latched into an internal buffer and a control state machine sequences the transmission process. A start bit (0) is sent first, followed by each bit of the data shifted out through a shift register on the system clock.

A parity bit is computed in parallel with transmission and inserted into the output stream if enabled. A stop bit (1) follows the parity or final data bit to complete the frame.

The TxRDY signal is asserted when the transmitter is idle and ready to accept new data. It is cleared when a transmission is in progress. Internal timing and sequencing are managed by a finite state machine that ensures correct output timing and framing.

The transmitter resets its state when Reset is asserted, halting any ongoing transmission and returning to an idle state.

### Receiver

The UART receiver accepts serial input data from the RxD line, decodes it according to the asynchronous protocol, and outputs the resulting parallel data on the Dout bus. It uses a finite state machine to manage the reception process, including start bit detection, data sampling, optional parity checking, and readiness signaling.

The reception begins with the detection of a start bit, identified as a falling edge on RxD while idle. Once detected, a baud-rate clock is internally generated to time the sampling of each bit in the frame. As data bits are received, they are shifted into a register on each baud clock pulse.

After all data bits are collected, a parity bit is optionally checked for errors. If the parity does not match, the RxParityErr flag is asserted. Once the full frame is received, the assembled data is presented on Dout, and RxRDY is asserted to indicate valid data is available. This signal is cleared upon a read (RD) operation.

The receiver resets all internal logic when Reset is asserted, ensuring predictable startup and recovery behavior.

### Status Register

The status register provides real-time flags indicating the operational state of the UART transmitter and receiver. It is accessed via the ReadData output when RD and CE_SR are asserted.

Bit Definitions:
Bit [0] – TxRDY (Transmit Ready):
Set to 1 when the transmitter is ready to accept new data. Cleared automatically when data is written to the transmitter.

Bit [1] – RxRDY (Receive Ready):
Set to 1 when new data has been received and is available in the receive buffer. Cleared when data is read from the receiver.

Bit [2] – RxParityErr (Receive Parity Error):
Set to 1 if a parity error is detected in the received data. Indicates that the received byte may be corrupted.

Bits [31:3] – Reserved:
These bits are reserved and always read as 0. They have no functional impact.

The status register is updated automatically by the internal UART logic. Software can poll or read this register to determine when to read or write data, or to detect receive errors.

## Application

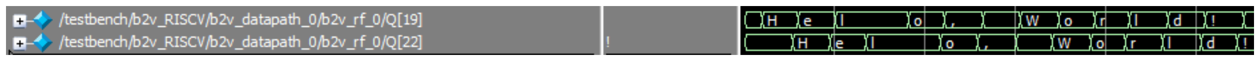
### Memory Mapped IO

This system was verified using and RISCV CPU connected to RAM, instruction memory, and address decoder and UART. Using the attached assembly program and testbench, the UART system was able to transmit and receive text.

### Verification

To test the UART transmitter using the provided assembly code, one can use a RISC-V simulator or a hardware setup that supports memory-mapped input/output. The program is designed to transmit a test string—specifically "Hello, World!"—one character at a time through the UART interface. It begins by storing the string into a designated area in RAM. The program then enters a loop that checks whether the UART transmitter is ready to send a character by polling the status register. Once the transmitter is ready, it reads the next character from memory and writes it to the UART data register, initiating its transmission. This process repeats until all characters in the string have been sent.

The following images show the correct send and receive of the UART system. The transmitted data is shown in register 19, and the received data is shown in register 22.


/testbench/b2v_RISCV/b2v_datapath_0/b2v_rf_0/Q[19]
/testbench/b2v_RISCV/b2v_datapath_0/b2v_rf_0/Q[22]
!
H e l o , W o r l d !
H e l o , W o r l d !