www.ijecs.in
International Journal Of Engineering And Computer Science
Volume 13 Issue 10 October 2024, Page No. 26492-26497
ISSN: 2319-7242 DOI: 10.18535/ijecs/v13i10.4906

UART IP core Design and Verification using WISHBONE Interface

Tejaswi Chandupatla[1], Prof. Sotoudeh Hamedi-Hagh [1]

[1] San Jose State University Department of Electrical Engineering

Abstract
The organization of communication between the devices is greatly aided by the communication protocol. These protocols have a clear set of guidelines that have been agreed upon by the tools needed for effective communication. A popular protocol for serial communication is UART. The hardware description language for the SV/Verilog UART functional module is designed in this paper. The serial connection capabilities offered by this UART IP CORE enable interaction with modems and other external devices. The Universal Verification Technique is used to undertake functional verification of the UART. By comparing the collected response to the intended response via the scoreboard mechanism, the reusable UVM test bench architecture is meant to push the randomized stimuli to the unit under test to assess the functional correctness. This core is made to be as compatible as possible with commercially accepted designs. The WISHBONE INTERFACE WITH 8-BIT OR 32-BIT SELECTABLE DATA BUS MODES is one of this design's primary characteristics. Interface debugging in 32-bit data bus mode. Functionality and register-level compatibility. FIFO procedure. The UVM methodology is used to validate the design. For optimal functional coverage, the test bench is written with regression test cases. The overall execution time will be shortened if the stimulus may be reused.

1. Introduction

1.1 Design

Verification of the design is very important in the VLSI chip design process. The plan should be tried for various experiments to check its usefulness. Verifying integrated circuits and determining their efficiency requires a lot of time and effort. The intricacy of confirmation increments with the intricacy of the gadget's plan. The conventional method of directed testing is ineffective when it comes to verifying intricate designs, which takes anywhere from 60 to 70 percent of the product development life cycle. Using Verilog or System Verilog to verify ICs takes longer and doesn't allow the test bench to be reused in the environment. The most common method of verification is the Universal Verification Methodology. The numerous class libraries at the UVM make it possible to reuse the environment across projects. We used communication protocol designs in this paper to verify using the UVM, which is UART and Intersystem protocols. Through the inter-bus system, the inter-system protocol is used to establish communication between two distinct devices, such as a computer and a microcontroller kit. One of the inter-system communication protocols is known as UART. Universal Asynchronous Receiver Transmitter is abbreviated as UART. [4] UART is a high-reliability hardware communication protocol used primarily for long-distance and reliable data transmission between devices. It can be utilized in both transmissions and getting information sequentially in a non-concurrent way. UART is an asynchronous communication protocol that does not use a clock signal to communicate between devices. One of its pins serves as a transmitter, and the other is a receiver. After receiving data from the system bus and transmitting it, the UART transmitter converts

parallel data into serial format. UART receivers convert serial data back into parallel format before sending it to the system bus. There are only two wires required for data transmission and reception between two UART modules.

1.2 Verification

Both within and between the chips, as well as between the systems, data is transferred. There won't be a way to distribute the clock because it is an asynchronous clock. There are three modes in UART. 1) THE MODE OF A HALF-DUPLEX: Both the transmission and the reception can take place simultaneously in the half-duplex mode. 2) FULLDUPLEX MODE: In the full-duplex mode, both gathering and transmission occur simultaneously. 3) LOOP BACK MODE: It is used for testing. We will be connected to both the transmitter and the receiver of the same UART, which makes it easier to really check its accuracy.

1.3 UART

One of the most popular hardware communication protocols is UART. Data transmission in both directions, asynchronously, and serially are all supported. UART can convey in the simplex, half-duplex, and full-duplex modes. One-way transmission is permitted in simplex mode. Either transmitter or recipient is permitted transmission at a time in half-duplex mode; if the transmitter is sending data, the receiver will be idle, and vice versa. In full duplex mode, transmissions in either direction are permitted; both the transmitter and the receiver operate simultaneously. There are typically two wires in UART: a Tx transmitter and an Rx receiver, as depicted in fig.

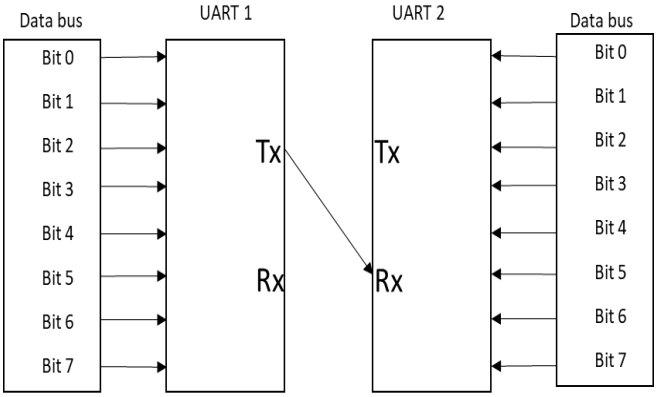

Figure (1): Universal Asynchronous Receiver Transmitter

After receiving data from the data bus, the transmitting UART converts parallel data into serial data. After receiving serial data, the receiving UART performs a bit-by-bit conversion to parallel data. There is no clock synchronization between the transmitter (Tx) and the receiver (Rx), so they must agree on a clock frequency beforehand. By setting the baud rate for UART Tx and UART Rx, this can be accomplished. The speed of data transmission is measured in baud rates, which are expressed as bits per second (bps). UART is able to function at a variety of baud rates, including 1200, 2400, 4800, 9600, and 115200 bps. In both UARTs, Tx, and Rx must function at the same baud rate [1] [3]. Packages are used to send data.

START	D0	D1	D2	D3	D4	D5	D6	D7	PARITY	STOP

Figure (2): UART frame format

The outline organization of a UART is displayed in fig - 2. There are start, stop, and parity bits in it. The start and stop bits of a message represent the beginning and end of the message. The parity bit serves as the check bit at the receiving end. The parity bit can be turned on or off based on what is needed.

In UART, frame communication is initiated by the transmitter. A data frame is built by adding start, stop, and parity bits. After that, the data frame is sent serially to the Tx pin. The start and stop bits are removed when the Rx pin reads the data frame. The data actually has a length of 5 to 8 bits [4].

With nine bits, it is data bus mode. The start bit is a low signal, while the stop bit is an active high signal. A parity bit and six data bits make up the remaining bits. Before sending data, UART converts CPU parallel data to serial data. Correspondence between something like two UARTS relies upon the non-concurrent successive strategy for transmission [1]. The UARTs shake hands using synchronizing bits. A starting piece, a number of configurable information bits, a free equality spot, and at least one stop bit is sent with each character.

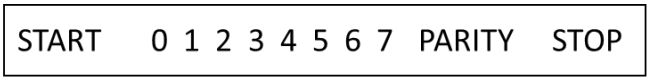

Figure (3): Bit Details explaining parity, start and stop bits

The architecture of UART comprises Wishbone Interface, Interrupt Registers, Control Registers, Baud generator, and Divisor Latches.

1.4 Wishbone

The wishbone interface is the standard computer bus interface that enables integrated circuit communication. On the wishbone bus, data can be transferred in both 8-bit and 32-bit formats.

Table -1: Wishbone Interface Signal

PORT	WIDTH	DIRECTION
CLK	1	INPUT
WB_RST_I	1	INPUT
WB_ADDR_I	5 or 3	INPUT
WB_SEL_I	4	INPUT
WB_DAT_I	32 or 8	INPUT
WB_WE_I	1	INPUT
WB_STB_I	1	INPUT
WB_CYC_I	1	INPUT
WB_DAT_O	32 or 8	OUTPUT
WB_ACK_O	1	OUTPUT

2.5 Interrupt Registers

The interrupts are enabled and identified by this register. Interrupt registers come in two varieties. The Intrude on Distinguishing proof Register and the Hinder Empower Register are both of them. The interfering empowering register controls whether the UART creates intrudes. It has a width of 8 pieces. The interrupt identification register enables the programmer to identify the pending interrupt that currently has the highest priority. [6] BIT-0 indicates that an interrupt is pending when logic is zero. When it is one, the obstacle does not appear. The width of the register is 8 bits.

2.6 Control Registers

There are two registers for control. The FIFO trigger level can be selected in the FIFO control register, which is also known as the LINE control register. The data width of the FIFO register is 8 bits. The asynchronous data communication format can be specified using the line control register. Always set the 0th bit to 0. [7] Additionally, access to the baud rate's divisor locks is made possible by a registered piece. The ongoing correspondence setting can be checked by perusing the register.

2.7 Baudgenerator

Based on the divisor latch value, the baud generator is in charge of generating a periodic baud pulse that is used to set the serial transmission baud rate. Both the transmitter and the receiver use this periodic pulse to generate sampling pulses for sampling the data that has been received and the data that will be transmitted. [8] 16 clock cycles result in one baud out. The data will be sent for one bit for sixteen clock cycles. There are two debug registers, and 32-bit data bus mode is supported by them. It has an address mode with 5 bits. It is provided for chip testing debugging purposes and is read-only. Each has a 256-byte FIFO to stop the progression of data [3]. The use of FIFO buffers increases the overall transmission rate and reduces the amount of time spent on context switching by making it possible for slower processors to respond. They check for various parity options, facilitate start/stop frames, and identify transmission errors in addition to data transfer [7].

2.8 Divisor Latches

Set the seventh piece of LCR to 1 to get to the divisor latches. To regain access to the other registers at the same address, reset the bit to zero after setting the divisor latches. The two bytes consolidate to shape a solitary sixteen-cycle register that is gotten inside as a solitary number. On reset, two bytes are pushed to zero to ensure normal operation. In order to guarantee that the register is explicitly set up in the software, the reset turns off all serial I/O operations. [9] The worth set ought to be equivalent to (framework clock speed)/16*desired baud rate. While setting the divisor, compose the MSB first, then the LSB last in light of the fact that the interior counter

beginnings working when the LSB of DL is returned.

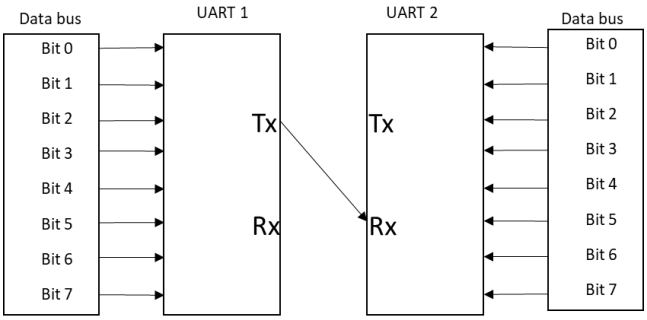

Figure (4): Diagram of the UART IP Core's Blocks

2. Universal Verification Methodology

The verification environment of the test bench consists of many components.

2.1 UVM Agent

Agents make up the UVM methodology test bench environment. Top TB has two Agent tops, one for UART A and one for UART B, both of which are active, therefore it has the classes MONITOR, DRIVER, SEQUENCER, and CONFIG. [10] During the run phase of the test case, sequences will be attached to the sequencer outside of TB using the start method. With the sequencer, the sequences will generate stimuli and transfer them to the driver. With the aid of the top static interface, each agent is connected to the DUT.

2.2 UVM Sequencer

The sequencer's output signals are driven into the DUT's pins by the driver. Create a virtual interface declaration in the driver to link it to the DUT. And for printing, copying, and comparison, we use macros. There is a connection between static and dynamic components, hence a virtual interface is offered.

2.3 DUV

The device that needs to be verified goes by the name of DUV. The scoreboard compares the output to the reference model when the DUV is operated with a variety of inputs. The functioning is verified using functional coverage.

2.4 Display

The monitor takes the bus's signal data and converts it into transactions. Scoreboard and monitor are linked using TLM interfaces. Examine exports and ports.

2.5 Sequence/Virtual Sequence

To enable a single sequence to regulate activity across several agents, a virtual sequencer is used throughout the stimulus generating process. [6]It does not process items on its own, is not tied to any driver, and simply refers to virtual sequencers in an agent environment.

2.6 Config Database

It is a flexible, effective system for arranging setup.

2.7 Scoreboard

The output of the UART2 receiver is compared to the inputs of the UART1 monitor on the scoreboard. Test cases are contained in a virtual test class that extends the uvm base test.

b) The environment is created, tested, and the package is imported in this section.

b) The logic inputs are randomized using a type of transaction that was developed from uvm sequence item.

d) To validate the half-duplex and full-duplex modes, a number of test cases were created.

This keeps track of how many times the code has been executed.

- This verification has made use of the full code.

3. Coverage Report

3.1 Functional Coverage:

This measures how widely the design standard has been applied.

- Putting our verification% specification to use.

3.2 Code Coverage:

This keeps track of how many times the code has been executed.

-This verification has made use of the full code.

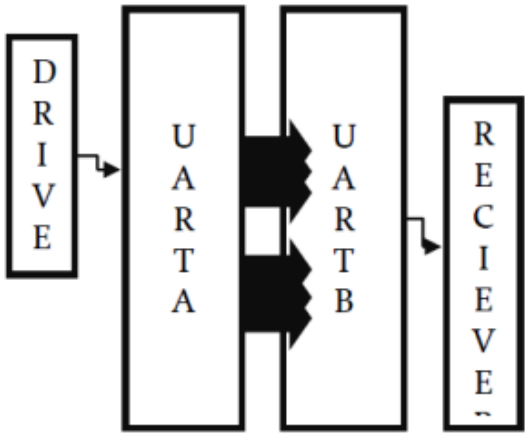

Figure (5): Half Duplex Mode

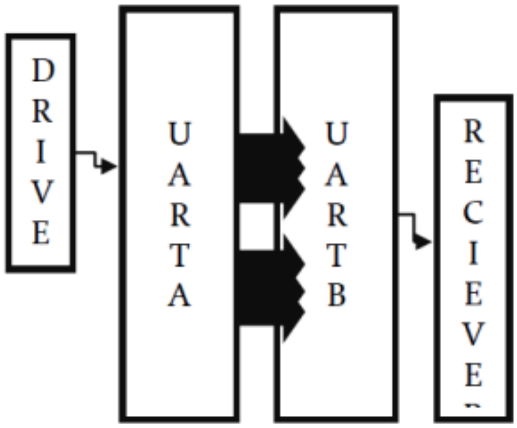

Figure (6): Full Duplex Mode

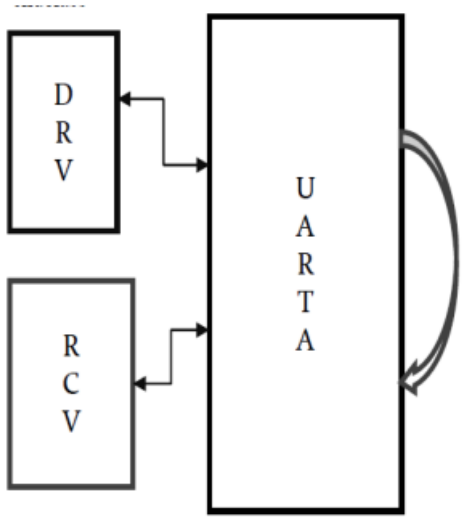

Figure (7): Back Loop Mode

Only one UART will function in this mode, and we will transmit and receive data using the same UART. By raising MCR 4th a little bit. Hence, the internal connection will take place and function in [5][6].

4. Conclusion

The UART IP core has been tested in the operation modes of HALF DUPLEX MODE, FULL DUPLEX MODE, and LOOP BACK MODE. I am currently working on covering all functions. Additionally, corner cases were verified and register functionality was checked. The UART transmission from parallel to serial and back again has been confirmed.

References:

1. Priyanka B., Gokul M.R., Nigitha A., & Poomica., J. (2021). “Design of UART Using Verilog And Verifying Using UVM”. 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS), 1, 1270-1273.
2. B. Vineeth; B. Bala Tripura Sundari, "UVM Based Testbench Architecture for Coverage Driven Functional Verification of SPI Protocol", 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2018 sep19
3. "Universal Asynchronous Receiver and Transmitter (UART)", pp. 1-12, 2018, [online] Available: https://www.researchgate.net/publication/308988751_Universal_Asynchronous_Receiver_and_Transmitter_UART.
4. "Design and Verification of UART using System", pp. 1-12, 2020, [online] Available: https://www.ijeat.org/wp content/uploads/papers/v9i5/E1135069520 .pdf.
5. Dr. Punith Kumar M B, Sreekantesha H N, “Design and Verification of SPI Core Using UVM” Journal of Emerging Technologies and Innovative Research (JETIR) JETIR May 2019, Volume 6, Issue 5
6. Frank Durda, Serial and UART Tutorial, uhc1em@FreeBSD.org.
7. M.S. Harvey, “Generic UART Manual” Silicon Valley. December 1999.
8. PCI6550DUniversalAsynchronousReceiver/Transmitter.

9. Ni W, Wang X. “Functional coverage-driven UVM-based UART IP verification” In2015 IEEE 11th International Conference on ASIC (ASICON) 2015 Nov 3 (pp. 1-4). IEEE.
10. Pallavi Polsani, V. Priyanka B., Y. Padma Sai, “Design and Verification of Serial Peripheral Interface (SPI) Protocol” International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878 (Online), Volume-8 Issue-6, March 2020.