ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

# Design of UART Serial Communication Module and Simulation Using VHDL

Leena L Kanjani
M.Tech Student,
Avanthi Institute.

N.Ashok Kumar, M.Tech
Associate Prof & HOD,
ASRA, Hyderabad.

## Abstract:

This paper deals the design and implementation of SoC's UART-SPI interface. The UART-SPIinterfaceprovidesusage for the universal asynchronous receiver/transmitter (UART) to serial peripheralinterface (SPI). This interface can be used to communicate to SPI slave devices from a PC with UART port. Theconverter consists of three blocks: the UART interface, the UART -to SPI interfacing block and the SPI Masterinterface. The Interface of UART - SPI in SOC will prove very effective in many applications. Thecommunication in the SOC architecture makes easy as they have been connected with a bus. In future, moreapplications will add into the subsystem, by which the routing architecture plays a vital role in the system and itcan be implemented in SOC.

## Keywords:

UART; asynchronous serial communication; VHDL; Quartus II; simulation

## I. INTRODUCTION:

The aim of our project is to interface the UART with SPI master by using a controller called SPI UART controller. The Interface of UART - SPI in SOC will come very effective in many applications. The communication in the SOC architecture makes easy as they have been connected with a bus. In future as of more applications will add into the subsystem the routing architecture plays a vital role in the system and it can be implemented in SOC .The UART-SPI interface a special type of interface because the SPI is a synchronous bus and the UART is an asynchronous bus. UART can communicate with only one peripheral.This type of interface is necessary where PC wants to communicate with the SPI slaves through UART port of PC, in this the SPI-UART interface is useful and also like this we need this type of interface in many applications.

Using the UART-SPI Interface, UART can communicate with more number of devices.

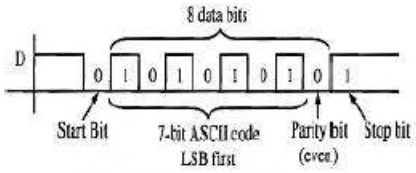


Figure 1: UART data format

The Universal Asynchronous Receiver/Transmitter (UART) takes bytes of data and transmits the individual bits in a sequential fashion. At the destination, a second UART re-assembles the bits into complete bytes. Each UART contains a shift register, which is the fundamental method of conversion between serial and parallel forms. Serial transmission of digital information (bits) through a single wire or other medium is less costly than parallel transmission through multiple wires. The UART usually does not directly generate or receive the external signals used between different items of equipment.

Separate interface devices are used to convert the logic level signals of the UART to and from the external signaling levels. External signals may be of many different forms Communication may be simplex (in one direction only, with no provision for the receiving device to send information back to the transmitting device), full duplex (both devices send and receive at the same time) or half duplex (devices take turns transmitting and receiving).During idle, no data state is high-voltage, or powered. This is a historic legacy from telegraphy, in which the line is held high to show that the line and transmitter are not damaged. Eachcharacter is sent as a logic low start bit, a configurable number of data bits (usually 8, but users can choose 5 to 8 or 9 bits dependingon which UART is in use), an optional parity bit if the number of bits per character chosen is not 9 bits, and one or more logic high stop bits .The start bit signals the receiver that a new character is coming.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

The next five to nine bits, depending on the code set employed, represent the character. If a parity bit is used, it would be placed after all of the data bits. The next one or two bits are always in the mark (logic high, i.e., '1') condition and called the stop bit(s).They signal the receiver that the character is completed. Since the start bit is logic low (0) and the stop bit is logic high (1) there are always at least two guaranteed signal changes between characters If the line is held in the logic low condition for longer than a character time, this is a break condition that can be detected by the UART.

## II. IMPLEMENTATION:

The structure of UART is as shown in figure 3.4, consists of Transmitter part and Receiver part, rather we can say, it consists of 3 units, transmitter circuit, receiver circuit and Control/Status Registers.

### A. Design of UART Transmitter:

The Block diagram of UART Transmitter is as shown in figure 3.5. The data is loaded from Data Bus Into TBR (Transmit Buffer Register) and from TBR to TSR (Transmit Shift Register), based on the control and status signals produced by the Control unit.

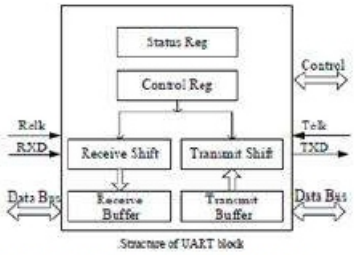


Figure 2. Structure of UART Block

The Size of TSR is taken in such a way that, it should accommodate the START and STOP bits alongwith the Data bits which are loaded from the Data Bus.The Data loaded into TSR has the format of STARTDATA-STOP bits which is as shown in figure of which, every time one bit will be sent, with reference to baudclock.Correspondingly, the data in TSR will keeps updating with o□s; will be completely filled with os, aftertransmission of the complete data packet.

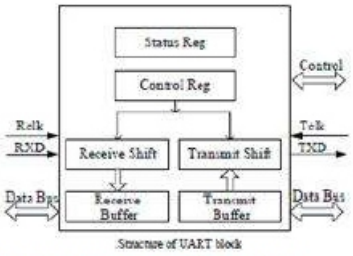


Figure 2. Structure of UART Block

The Size of TSR is taken in such a way that, it should accommodate the START and STOP bits alongwith the Databits which are loaded from the Data Bus.The Data loaded into TSR has the format of STARTDATA-STOP bits which is as shown in figure of which, every time one bit will be sent, with reference to baudclock.Correspondingly, the data in TSR will keeps updating with os; will be completely filled with os, aftertransmission of the complete data packet.

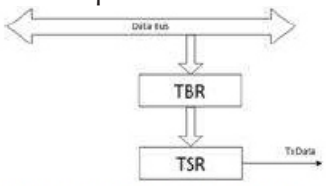


Figure 3. UART Transmitter Unit

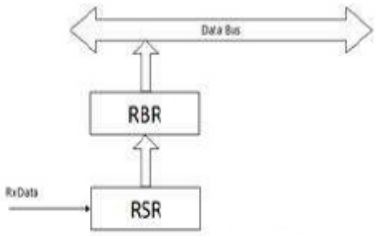


Figure 4. UART Receiver

### B. Design of UART Receiver:

The Block diagram of UART Receiver is as shown in figure 3.6. The data receiving will be capturedusing receiving baud clock and then loaded into RSR (Receive Shift Register) and from RSR to RBR (ReceiveBuffer Register), and then to Data Bus, based on the control and status signals produced by the Control unitThe Size of RSR is taken in such a way that, it should accommodate the START and STOP bits along with theData bits which are loaded from the Data Bus.

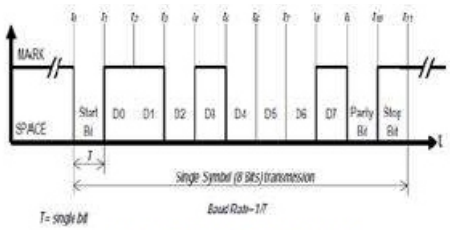


Figure 5. Serial Data Format

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

The start bit is always a 0 (logic low), which is also called a space. The start bit signals the receiving DTE that a character code is coming. The next five to eight bits, depending on the code set employed, represent the character. In the ASCII code set the eighth data bit may be a parity bit. The next one or two bits are always in the mark (logic high, i.e., '1') condition and called the stop bit(s). They provide a "rest" interval for the receiving DTE so that it may prepare for the next character which may be after the stop bit(s). The rest interval was required by mechanical Teletypes which used a motor driven camshaft to decode each character. At the end of each character the motor needed time to strike the character bail (print the character) and reset the cam shaft .All operations of the UART hardware are controlled by a clock signal which runs at a multiple (say,16) of the data rate - each data bit is as long as 16 clock pulses.

The receiver tests the state of the incomingsignal on each clock pulse, looking for the beginning of the start bit. If the apparent start bit lasts at least onehalfof the bit time, it is valid and signals the start of a new character. If not, the spurious pulse is ignored .Afterwaiting a further bit time, the state of the line is again sampled and the resulting level clocked into a shift register. After the required number of bit periods for the character length (5 to 8 bits, typically) have elapsed, thecontents of the shift register is made available (in parallel fashion) to the receiving system. The UART will set aflag indicating new data is available, and may also generate a processor interrupt to request that the hostprocessortransfers the received data. In some common types of UART, a small first-in, first-out (FIFO) buffermemory is inserted between the receiver shift register and the host system interface.This allows thehostprocessor more time to handle an interrupt from the UART and prevents loss of received data at high rates.Transmission operation is simpler since it is under the control of the transmitting system. As soon asdata is deposited in the shift register, the UART hardware generates a start bit, shifts the required number ofdata bits out to the line, generates and appends the parity bit (if used), and appends the stop bits. Sincetransmission of a single character may take a long time relative to CPU speeds, the UART will maintain a flagshowing busy status so that the host system does not deposit a new character for transmission until the previousone has been completed; this may also be done with an interrupt.

Since full-duplex operation requires charactersto be sent and received at the same time, practical UARTs use two different shift registers for transmittedcharacters and received characters.Transmitting and receiving UARTs must be set for the same bit speed, character length, parity, andstop bits for proper operation. The receiving UART may detect some mismatched settings and set a "framingerror" flagbit for the host system; in exceptional cases the receiving UART will produce an erratic stream ofmutilatedcharacters and transfer them to the host system.Typical serial ports used with personal computers connected to modems use eight data bits, no parity,and one stop bit; for this configuration the number of ASCII character per seconds equals the bit rate divided by10.

## B.1 Special Receiver Conditions:

### Over-run Error:

An "overrun error" occurs when the UART receiver cannot process the character that just came inbefore the next one arrives. Various UART devices have differing amounts of buffer space to hold receivedcharacters. The CPU must service the UART in order to remove characters from the input buffer. If the CPUdoes not service the UART quickly enough and the buffer becomes full, an Overrun Error will occur.

### Under run Error:

An "under-run error" occurs when the UART transmitter has completed sending a character and the transmit buffer is empty. In asynchronous modes this is treated as an indication that no data remains to be transmitted, rather than an error, since additional stop bits can be appended. This error indication is commonly found in USARTs, since an under run is more serious in synchronous systems.

### Framing Error:

A "framing error" occurs when the designated "start" and "stop" bits are not valid. As the "start" bit is used to identify the beginning of an incoming character, it acts as a reference for the remaining bits. If the data line is not in the expected idle state when the "stop" bit is expected, a Framing Error will occur.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

## Parity Error:

A "parity error" occurs when the number of "active" bits does not agree with the specified parity configuration of the UART, producing a Parity Error. Because the "parity" bit is optional, this error will not occur if parity has been disabled. Parity error is set when the parity of an incoming data character does not match the expected value.

## C. SPI Design:

SPI stands for Serial Peripheral Interface is a synchronous protocol that allows a master device to initiate communication with a slave device. Data is exchanged between these devices. SPI is implemented by a hardware module called the Synchronous Serial Port or the Master Synchronous Serial Port. It allows serial communication between two or more devices at a high speed and is reasonably easy to implement .SPI is a Synchronous protocol. Only the master device can control the clock line, SCK. Often a slave select signal will control when a device isaccessed. This signal must be used for when more than one slaveexists in a system, but can be optional when only one slave exists in the circuit. As a general rule, it should be used. This signal is known as the SS signal and stands for "Slave Select." It indicates to a slave that the master wishes to start an SPI data exchange between that slave device and itself. The signal is most often active low, so a low on this line will indicate the SPI is active, while a high will signal inactivity. It is often used to improve noise immunity of the system. Its function is to reset the SPI slave so that it is ready to receive the next byte .

SPI is a Serial Interface and uses the following signals to serially exchange data with another device:

» SS - This signal is known as Slave Select. When it goes low, the slave device will listen for SPI clock and data signals.

» SCK - This is the serial clock signal. It is generated by the master device and controls when data is sent and when it is read.

» MOSI - The signal is generated by Master, recipient is the Slave.

» MISO -The signals are generated by Slaves, recipient is the master.

» SI - Serial Data Input (used to transfer data into the SPI device).

» SO - Serial Data Output (used to transfer data out of the SPI device).

» CS - Chip Select Input (for enabling device operation).

» W- Write Protect Input (used to guard against program/erase instructions).

## C.1 Data transmission:

A typical hardware setup using two shift registers to form an inter-chipcircular buffer To begin a communication, the bus master first configures the clock, using a frequency less than or equal to themaximum frequency the slave device supports. Such frequencies are commonly in the range of 1–100 MHz.

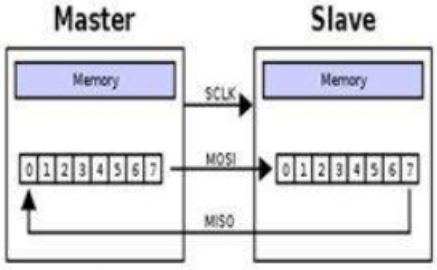


Figure 6. Data Transmission from master to slave

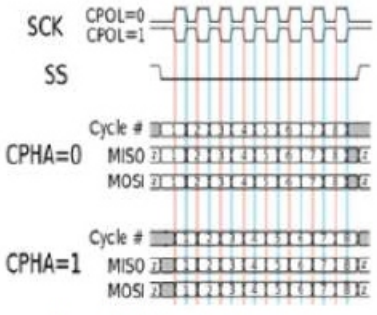


Figure 7. Clock polarity and phase

The master then transmits the appropriate chip select bit for the desired chip to logic 0. Logic 0 is transmitted because the chip select line is active low, meaning its off state is a logic 1; on is asserted with a logic0. If a waiting period is required (such as for analog-to-digital conversion), then the master must wait for at least that period of time before starting to issue clock cycles.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

During each SPI clock cycle, a full duplex data transmission occurs:

» the master sends a bit on the MOSI line; the slave reads it from that same line

» the slave sends a bit on the MISO line; the master reads it from that same line

Not all transmissions require all four of these operations to be meaningful but they do happen. Transmissions normally involve two shift registers of some given word size, such as eight bits, one in the master and one in the slave; they are connected in a ring. Data is usually shifted out with the most significant bit first, while shifting a new least significant bit into the same register. After that register has been shifted out, the master and slave have exchanged register values. Then each device takes that value and does something with it, such as writing it to memory. If there is more data to exchange, the shift registers are loaded with new data and the process repeats .Transmissions may involve any number of clock cycles. When there is no more data to be transmitted,the master stops toggling its clock. Normally, it then deselects the slave.

Transmissions often consist of 8-bitwords, and a master can initiate multiple such transmissions if it wishes/needs. However, other word sizes are also common, such as 16-bit words for touchscreen controllers or audio codecs, like the TSC2101 from Texas Instruments; or 12-bit words for many digital-to-analogs or analog-to-digital converters. Every slave on the busthat hasn't been activated using its chip select line must disregard the input clock and MOSI signals, and must not drive MISO. The master must select only one slave at a time. A timing diagram showing clock polarity and phase in addition to setting the clock frequency, themaster must also configure the clock polarity and phase with respect to the data. Freescale's SPI Block Guide names these two options as CPOL and CPHA respectively, and most vendors have adopted that convention.

The timing diagram is shown to the right. The timing is further described below and applies to both the master and the slave device

» At CPOL=0 the base value of the clock is zero

» For CPHA=0, data is captured on the clock's rising edge (lowhigh transition) and data is propagated on a falling edge(highlow clock transition).

» For CPHA=1, data is captured on the clock's falling edge and data is propagated on a rising edge.

» At CPOL=1 the base value of the clock is one (inversion of CPOL=0)

» For CPHA=0, data is captured on clock's falling edge and data is propagated on a rising edge.

» For CPHA=1, data is captured on clock's rising edge and data is propagated on a falling edge.

## III. SYNTHESIS:

The developed UART is simulated and verified according to their functionality. Once the functional verification is done, the RTL model is taken to the synthesis process using the Xilinx ISE tool. In synthesis process, the RTL model will be converted to the gate level netlist mapped to a specific technology library. Herein this Spartan 3E family, many different devices were available in the Xilinx ISE tool. In order to synthesis thisDWT and IDWT design the device named as "XC3S500E" has been chosen and the package as "FG320" with the device speed such as "-4".

### A. Timing Summary:

In timing summary, details regarding time period and frequency is shown are approximate while synthesize. After place and routing is over, we get the exact timing summery. Hence the maximum operating frequency of this synthesized design is given as 167.983MHz and the minimum period as 5.953ns. Here ,OFFSET IN is the minimum input arrival time before clock and OFFSET OUT is maximum output required time after clock.

**Speed Grade : -4**
**Minimum period: 5.953ns (Maximum Frequency : 167.983MHz)**
**Minimum input arrival time before clock : 4.537ns**
**Maximum output required time after clock : 5.919ns**

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

## B. RTL Schematic:

The RTL (Register Transfer Logic) can be viewed as black box after synthesize of design is made. ItShows the inputs and outputs of the system. By double-clicking on the diagram we can see gates, flip flops and MUX.


Figure 8. UART Schematic with Basic Inputs and Output

## IV. SIMULATION RESULTS:

A. UART Transmitter

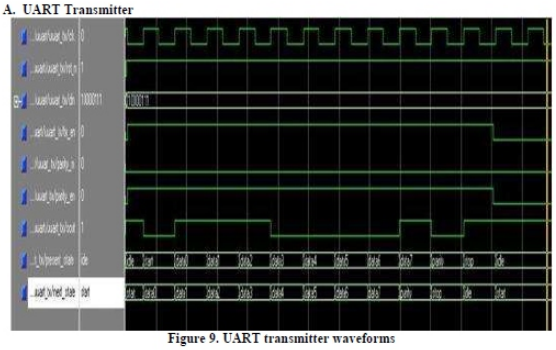

Figure 9. UART transmitter waveforms

Above fig shows the waveforms of the UART transmitter. The present_state indicates the current stateof thestatemachine. It traverses from IDLE to STOP state. The data input can be seen on din and corresponding-serial output is given on sout. Since parity_en = 1, parity bit is appended to data.

B. UART Receiver

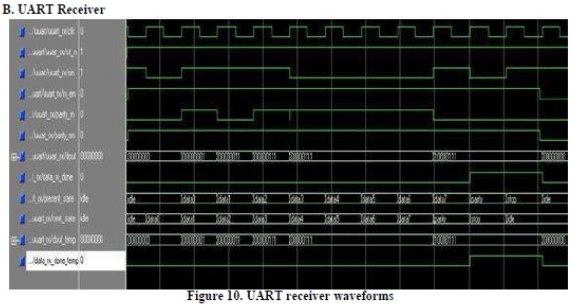

Figure 10. UART receiver waveforms

Above fig shows the receiver waveforms. The state transitions are similar to transmitter. The serial datainput comes on sin and output data is dout. The data is sampled when data_rx_done = ,,1.

C. Parity Generator

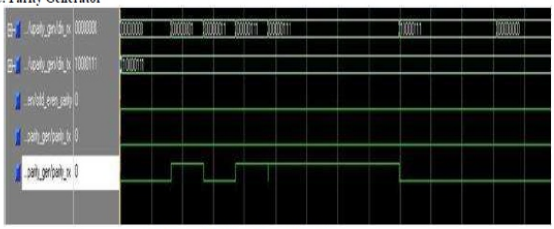

Figure 11. Waveforms of parity generator

Above fig shows the waveforms of parity generator. Depending on the odd_ even _parity, odd (if odd_ even_ parity = ,,1) or even (if odd_ even _parity=0) parity is generated. The parity _rx bit corresponds to receiver data (din _rx) and parity_ tx corresponds to transmitter data (din _tx).

## V. CONCLUSION:

This design uses VHDL as design language to achieve the modules of UART. Using Quartus II software, Altera's Cyclone series FPGA chip EP2C5F256C6 to complete simulation and test. The results are stable and reliable. The design has great flexibility, high integration, with some reference value. Especially in the field of electronic design, where SOC technology has recently become increasingly mature, this design shows great significance.

## REFERENCES:

[1]. Zou,Jie Yang,Jianning Design and Realization of UART Controller Bas ed on FPGALiakot Ali Roslina Sidekshak Aris Alauddin Mohd. Ali Bambang Sunaryo Suparjo.Design of a micro - UART for SoC application [J].In: Computers and Electrical Engineering 30 (2004)257– 268.

[2]. HU Hua, BAI Feng-e. Design and Simulation of UART Serial Communication Module Based onVerilog -HDL[J]. J ISUANJ I YU XIANDA IHUA 2008 Vol. 8

[3]. Frank Durda Serial and UART Tutorial. uhclem@ FreeBSD.org