UART IP Core - Lattice Propel Builder

User Guide

FPGA-IPUG-02105-1.2

April 2021

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS and with all faults, and all risk associated with such information is entirely with Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
APB	Advanced Peripheral Bus
EIA	Electronic Industries Association
FIFO	First-In, First-Out
LMMI	Lattice Memory Mapped Interface
RTL	Register Transfer Language
UART	Universal Asynchronous Receiver/Transmitter

1. Introduction

The Lattice Semiconductor UART (Universal Asynchronous Receiver/Transmitter) IP Core is designed for use in serial communication, supporting the RS-232. The UART IP Core has many characteristics similar to those of the NS16450 UART. To preserve FPGA resources, the UART IP Core is not identical to the NS16450 UART. As such, it is not source code compatible. This means the existing driver code for the NS16450 UART does not work on the Lattice UART IP Core.

The design is implemented in Verilog HDL. The IP can be configured and generated based on Table 1.1.

Table 1.1. FPGA Software for IP Configuration, Generation, and Implementation

Supported FPGA Family	IP Configuration and Generation	IP Implementation (Synthesis, Map, Place and Route)
MachXO2™	Lattice Propel™ Builder software	Lattice Diamond® software
MachXO3™	Lattice Propel Builder software	Lattice Diamond software
MachXO3D™	Lattice Propel Builder software	Lattice Diamond software
Mach™-NX	Lattice Propel Builder software	Lattice Diamond software
CrossLink™-NX	Lattice Propel Builder software	Lattice Radiant™ software
Certus™-NX	Lattice Propel Builder software	Lattice Radiant software

1.1. Features

The key features of the UART IP include:

- APB 1.0 interface
- Similarity with the National Semiconductor NS16450 UART with different register addresses
- Optional 16-word-deep FIFO implemented in the UART transmit/receive path when FIFO mode is selected
- Insertion or extraction of standard asynchronous communication bits (start, stop, and parity) to or from the serial data
- Holding and shifting registers, which eliminate the need for precise synchronization between the host (APB interface) and serial data
- A common interrupt line for all internal UART data and error events. Interrupt conditions include receiver line errors, receiver buffer available, transmit buffer empty, and detection of status flag change.
- Fully prioritized interrupt system control
- Fully programmable serial interface characteristics:
  - Configurable data widths of 5, 6, 7, or 8 bits
  - Even-, odd-, or no-parity bit generation and detection
  - 1-, 1.5-, or 2-stop bit generation and detection
  - False start bit detection
  - Line break generation and detection
  - Interactive control signaling and status reporting capabilities
- Configurable Baud Rate support for Standard and Custom modes
  - In Standard mode, programmable divisor latch for baud rates provide fixed pre-defined values.
  - In Custom mode, Baud Rate accepts any value from range 1-999999.

1.2. Conventions

1.2.1. Nomenclature

The nomenclature used in this document is based on Verilog HDL.

1.2.2. Signal Names

Signal names that end with:

- *_n* are active low (asserted when value is logic 0)
- *_i* are input signals
- *_o* are output signals

1.2.3. Host

The logic unit inside the FPGA interacts with the UART IP Core through APB.

1.2.4. Attribute Names

Attribute names in this document are formatted in title case and italicized (*Attribute Name*).

2. Functional Descriptions

2.1. Overview

The UART IP Core performs two main functions:

- Serial-to-parallel conversion on data characters received from an external UART device; and
- Parallel-to-serial conversion on data characters received from the Host located in the FPGA.

The Host can read the complete status of the UART at any time during the functional operation. Status information reported includes the type and condition of the transfer operations being performed by the UART IP Core, as well as any error conditions (parity, overrun, framing, or break interrupt).

The UART IP has implemented a processor-interrupt system similar to UART 16450. Interrupts can be programmed to your requirements, minimizing the computing required to handle the communications link. The UART IP currently did not implement the MODEM-control feature of UART 16450.

The registers of UART IP Core are accessed by the host (FPGA internal components) through an AMBA APB interface. The functional block diagram of UART IP Core is shown in Figure 2.1. The dashed lines in the figure are optional components/signals, which means they may not be available in the IP when disabled in the attribute.

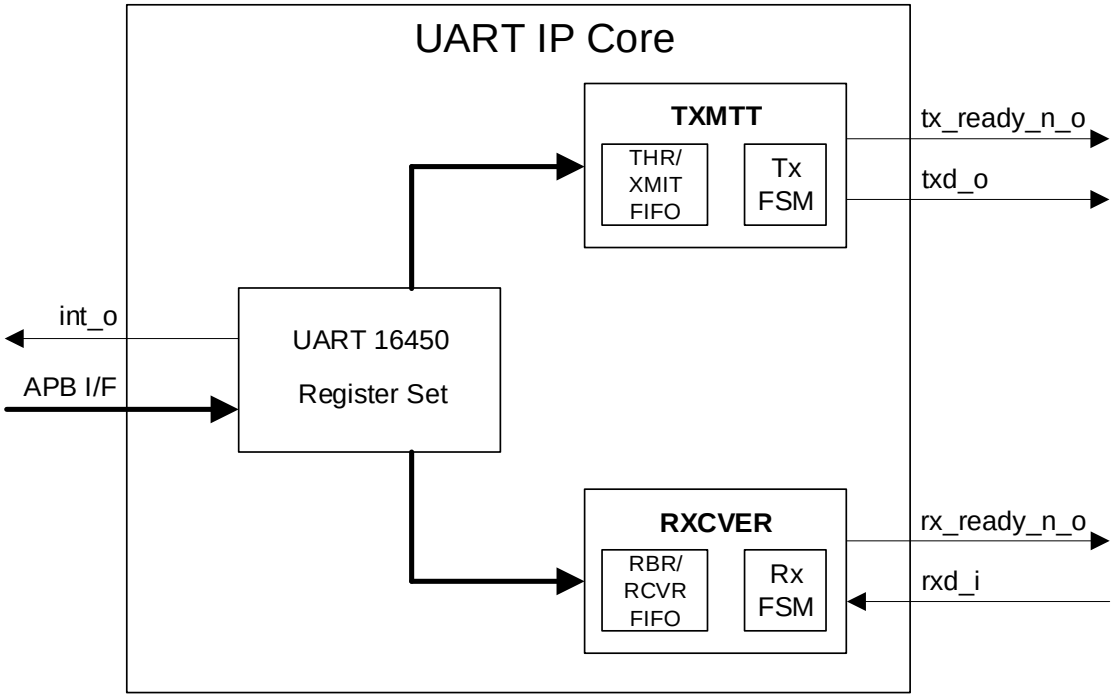

Figure 2.1. Functional Block Diagram

2.2. Signal Description

Table 2.1. UART IP Core Signal Description

Port Name	I/O	Width	Description
Clock and Reset			
clk_i	In	1	System clock
rst_n_i	In	1	Asynchronous active low reset The reset assertion can be asynchronous but reset negation should be synchronous. When asserted, output ports and registers are forced to their reset values.
Interrupt Port			
int_o	Out	1	Interrupt signal
Serial Interface			
txd_o	Out	1	Serial Output Serial data output to the communication link Reset value: 1'b1
rxd_i	In	1	Serial Input Serial data input from the communication link
APB Interface			
apb_psel_i	In	1	Select signal Indicates that the slave device is selected and a data transfer is required.
apb_paddr_i	In	6	Address signal
apb_pwdata_i	In	32	Write data signal Bits [31:8] are not used.
apb_pwrite_i	In	1	Direction signal Write = 1, Read = 0
apb_penable_i	In	1	Enable signal Indicates the second and subsequent cycles of an APB transfer.
apb_pready_o	Out	1	Ready signal Indicates transfer completion. Slave uses this signal to extend an APB transfer.
apb_pslverr_o	Out	1	Error signal Indicates a transfer failure. This signal is tied to 1'b0.
apb_prdata_o	Out	32	Read data signal Bits [31:8] are not used.
Auxiliary Signals			
rx_ready_n_o	Out	1	Indicates the UART receiver has received a data and is available in Receive Buffer Register. Reset value: 1'b1
tx_ready_n_o	Out	1	Indicates the UART transmitter is ready to receive new data for sending via txd_o signal. In FIFO mode (*FIFO Enable* is checked), this signal is asserted LOW when at least 1 data is available in XMIT FIFO. In non FIFO mode, this signal is asserted LOW when Transmitter Holding Register is empty. Reset value – 1'b1

2.3. Attribute Summary

The configurable attributes of the UART IP Core are shown in Table 2.2 and are described in Table 2.3.

The attributes can be configured through the IP Catalog's Module/IP wizard of the Lattice Propel Builder.

Table 2.2. Attributes Table

Attribute	Selectable Values	Default	Dependency on Other Attributes
General			
Enable APB	Checked	Checked	For information only
System Clock Frequency (MHz)	2–200	50	—
Serial Data Width	5, 6, 7, 8	8	—
Stop Bits	1, 2	1	—
Parity Enable	Checked, Unchecked	Unchecked	—
ODD Parity	Checked, Unchecked	Unchecked	Active if *Parity Enable* is Checked
Enable Stick Parity	Checked, Unchecked	Unchecked	Active if *Parity Enable* is Checked
Baud Rate			
Baud Rate Type	Standard, Custom	Standard	—
Standard Baud Rate	2400, 4800, 9600, 14400, 19200, 28800, 38400, 56000, 57600, 115200	115200	Active if *Baud Rate Type* == Standard
Custom Baud Rate	2400 – 1000000	115200	Active if *Baud Rate Type* == Custom
UART Feature Enables			
FIFO Enable	Checked, Unchecked	Unchecked	—
MODEM Enable	Unchecked	Unchecked	For information only
Rx Ready Enable	Checked, Unchecked	Unchecked	—
Tx Ready Enable	Checked, Unchecked	Unchecked	—

Table 2.3. Attributes Descriptions

Attribute	Dependency on Other Attributes
General	
Enable APB	Selects memory-mapped interface for register access by the host. The interface is currently fixed to AMBA APB.
System Clock Frequency (MHz)	Specifies the target frequency of system clock. This is used for baud rate calculation.
Serial Data Width	Specifies the default data bit width of UART transactions by setting the reset value of char_len_sel field of Line Control Register. 5 – Reset value is 2'b00 6 – Reset value is 2'b01 7 – Reset value is 2'b10 8 – Reset value is 2'b11
Stop Bits	Specifies the default number of stop bits to be transmitted and received by specifying the reset value of stop_bit_ctrl bit of Line Control Register. 1 – Reset value is 1'b0 2 – Reset value is 1'b1
Parity Enable	Specifies the absence/presence of parity by setting the reset values of parity_en bits of Line Control Register. Checked – Reset value is 1'b1 Unchecked – Reset value is 1'b0
ODD Parity	Specifies the default parity type (odd/even) by setting the reset values of even_parity_sel of Line Control Register. Checked – Reset value is 1'b0 Unchecked – Reset value is 1'b1

Attribute	Dependency on Other Attributes
Enable Stick Parity	Enables the generation and checking of Stick parity by specifying the reset value of stick_parity_en bit of Line Control Register. Checked – Reset value is 1'b1 Unchecked – Reset value is 1'b0
Baud Rate	
Baud Rate Type	Selects between Standard Baud Rate and Custom Baud Rate for the reset value of Divisor Latch Register. The selected Baud rate is used to set the reset value of Divisor Latch Register as follows: {DLR_MSB, DLR_LSB} = *System Clock Frequency (MHz)* x 1000000 / Selected Baud Rate
Standard Baud Rate	Specifies the baud rate of UART transactions from the standard baud rate options.
Custom Baud Rate	Specifies the baud rate of UART transactions based on user input.
UART Feature Enables	
FIFO Enable	Enables the implementation of FIFO in both transmit and receive path.
MODEM Enable	Enables the presence of MODEM control signals. This is currently not supported.
Rx Ready Enable	Enables the presence of rx_ready_n_o signal on the generated IP.
Tx Ready Enable	Enables the presence of tx_ready_n_o signal on the generated IP.

2.4. Register Description

The register address map, shown in Table 2.4, specifies the available IP Core registers. This is based on register set of UART 16450 but the offset address is changed to simplify the access to each registers. The offset of each register increments by four to allow easy interfacing with the Processor and System Buses. In this case, each register is 32-bit wide wherein the lower 8 bits are used and the upper 24 bits are unused. The unused bits are treated as reserved – write access is ignored and read access returns 0.

Table 2.4. Registers Address Map

Offset	Register Name	Access Type	Description
0x00	RBR	RO	Receive Buffer Register
0x00	THR	WO	Transmitter Holding Register
0x04	IER	RW	Interrupt Enable Register
0x08	IIR	RO	Interrupt Identification Register
0x0C	LCR	RW	Line Control Register
0x10	Reserved	RSVD	Reserved
0x14	LSR	RO	Line Status Register
0x18–0x1C	Reserved	RSVD	Reserved
0x20	DLR_LSB	WO	Divisor Latch Register LSB
0x24	DLR_MSB	WO	Divisor Latch Register MSB
0x28–0x3C	Reserved	RSVD	Reserved

The behavior of registers to write and read access is defined by its access type, which is defined in Table 2.5.

Table 2.5. Access Type Definition

Access Type	Behavior on Read Access	Behavior on Write Access
RO	Returns register value	Ignores write access.
WO	Returns 0	Updates register value.
RW	Returns register value	Updates register value.
RSVD	Returns 0	Ignores write access.

2.4.1. Receive Buffer Register (RBR)

The Receive Buffer Register is the interface to the Receiver Buffer/FIFO (RCVR FIFO). In FIFO mode, reading from this register pops and returns the output data in the RCVR FIFO. If read is performed during RCVR FIFO empty, the last data in the FIFO is returned. In non-FIFO mode, this register can only hold one data. Thus, the receive data should be read before the next data is completely received.

The rx_ready_n_o signal is set to 1'b0 and LSR.data_rdy is set to 1'b1 when this register/FIFO has data.

Table 2.6. Receive Buffer Register

Field	Name	Access	Width	Reset
[7:0]	rx_data	RO	8	0x00

- rx_data
  Receive Data. Contains the received data from UART.

2.4.2. Transmitter Holding Register (THR)

The Transmitter Holding Register is the interface to the Transmitter Buffer/FIFO (XMIT FIFO). In FIFO mode, writing to this register pushes the write data to the XMIT FIFO. If write is performed during XMIT FIFO full, data is lost. The host can only know if the XMIT FIFO is empty or not (via LSR.thr_empty); it cannot know how many characters are currently in the XMIT FIFO. Thus, it is recommended to initiate the write sequence to this register only when XMIT FIFO is empty wherein the host can write up to 16-character data. In non-FIFO mode, this register can store only one data while the previously written data is being sent. For example, you can write two data in which the first data immediately goes to the Transmitter Shift Register for transmission and the second data is stored in this register until the previous one is fully transmitted.

Table 2.7. Transmitter Holding Register

Field	Name	Access	Width	Reset
[7:0]	tx_data	WO	8	0x00

- tx_data
  Transmit Data. Contains the transmit data to UART.

2.4.3. Interrupt Enable Register (IER)

The Interrupt Enable Register enables the three types of UART interrupts. When enabled, each interrupt can individually activate the interrupt (int_o) output signal. Each interrupt can individually be enabled/disabled by writing to the corresponding bit in the IER. Disabling an interrupt prevents it from being indicated as active in the IIR and from activating the int_o output signal. On the other hand, the UART status including Line Status Registers are not affected by IER setting.

Table 2.8. Interrupt Enable Register

Field	Name	Access	Width	Reset
[7:3]	reserved	RSVD	3	—
[2]	rls_int_en	RW	1	Refer to attribute: *Receiver Line Status Interrupt*
[1]	thre_int_en	RW	1	Refer to attribute: *Transmitter Holding Register Empty Interrupt*
[0]	rda_int_en	RW	1	Refer to attribute: *Received Data Available Interrupt*

- rls_int_en
  Receiver Line Status Interrupt Enable. This bit controls the enabling of Receiver Line Status Interrupt.
  1'b0 – Interrupt is disabled
  1'b1 – Interrupt is enabled
- thre_int_en

Transmitter Holding Register Empty Interrupt Enable. This bit controls the enabling of Transmitter Holding Register Empty Interrupt.

1'b0 – Interrupt is disabled

1'b1 – Interrupt is enabled

- rda_int_en
  Received Data Available Interrupt Enable. This bit controls the enabling of Received Data Available Interrupt and Timeout Interrupt (timeout_int).

  1'b0 – Interrupt is disabled

  1'b1 – Interrupt is enabled

2.4.4. Interrupt Identification Register (IIR)

To minimize software overhead during data character transfers, the UART IP Core prioritizes interrupts into four levels and records these in the Interrupt Identification Register. When the host reads the IIR, the IP Core only indicates the highest priority pending interrupt. After clearing the source of the highest priority pending interrupt, IIR updates to the next priority pending interrupt. The summary of controlling (asserting and clearing) the interrupt is described in the Interrupt section.

Table 2.9. Interrupt Identification Register

Field	Name	Access	Width	Reset
[7:6]	fifos_en	RO	2	See description
[5:3]	reserved	RSVD	2	—
[2:1]	int_prio	RO	2	2'b00
[0]	int_pending	RO	1	1'b1

- fifos_en
  FIFOs Enabled. Fixed value based on *FIFO Enable* attribute. If FIFO is enabled, fifos_en is set to logic 2'b11 to indicate that both write and read FIFOs are enabled.

  FIFO Enable is Unchecked – Fixed to 2'b00

  FIFO Enable is Checked – Fixed to 2'b11
- int_prio
  Interrupt Priority. This bit indicates the highest priority pending interrupt. For example, if all interrupts are asserted, int_prio returns 2'b11 when read. After Receiver Line Status Interrupt is cleared, int_prio becomes 2'b10 to indicate the next high priority pending interrupt.

  2'b11 – Receiver Line Status Interrupt

  2'b10 – Received Data Available Interrupt

  2'b01 – Transmitter Holding Register Empty Interrupt

  2'b00 – MODEM Status Interrupt (Not yet supported)
- int_pending
  Interrupt Pending. This bit indicates that an interrupt is pended.

  1'b0 – Interrupt is pending according to int_prio and timeout_int.

  1'b1 – No pending interrupt

2.4.5. Line Control Register (LCR)

The Line Control Register configures character length, number stop bits and parity bit.

Table 2.10. Line Control Register

Field	Name	Access	Width	Reset
[7]	reserved	RSVD	1	1'b0
[6]	break_ctrl_en	RW	1	1'b0
[5]	stick_parity_en	RW	1	Refer to attribute: *Enable Stick Parity*
[4]	even_parity_sel	RW	1	Refer to attribute: *Parity Type*
[3]	parity_en	RW	1	Refer to attribute: *Parity Type*
[2]	stop_bit_ctrl	RW	1	Refer to attribute: *Stop Bits*
[1:0]	char_len_sel	RW	1	Refer to attribute: Serial Data Width

- break_ctrl_en
  Break Control Enable. When break control enabled, it causes serial output (txd_o) to go and stay at logic 0, which is interpreted as long stream of 0 bits by the receiving UART - the *break condition*. The Break Control Bit acts only on txd_o and does not control the transmitter logic.
  1'b0 – Disables break control.
  1'b1 – Enables break control.
- stick_parity_en
  Stick Parity Bit. When Bit 3, Bit 4, and Bit 5 of this register are logic 1, the parity bit is transmitted and checked as a logic 0. When Bit 3 and Bit 5 are logic 1 and Bit 4 is logic 0, then the parity bit is transmitted and checked as a logic 1. Stick parity is usually used for checking the parity check behavior of the external UART.
  1'b0 – Disables stick parity.
  1'b1 – Enables stick parity.
- even_parity_sel
  Even Parity Select bit. This bit is enabled when parity_en is 1'b1.
  1'b0 – Odd parity. An odd number of logic 1s are transmitted and checked in data character bits.
  1'b1 – Even parity. An even number of logic 1s are transmitted and checked in data character bits.
- parity_en
  Parity Enable bit. Enables transmission and checking of parity bit.
  1'b0 – Disables parity.
  1'b1 – Enables parity.
- stop_bit_ctrl
  Stop Bit Control. Specifies number of stop bits. The receiver only checks the first stop bit regardless of the setting.
  1'b0 – One stop bit is generated in the transmitted data.
  1'b1 – If char_len_sel==2'b00, a single one and a half stop bits are generated. Otherwise, two stop bits are generated.
- char_len_sel
  Character length select. Selects a character length.
  2'b00 – 5 bits
  2'b01 – 6 bits
  2'b10 – 7 bits
  2'b11 – 8 bits

2.4.6. Line Status Register (LSR)

The Line Status Register provides status information to the Host concerning data transfer. This register is not affected by the Interrupt Enable Register. For example, if *Receiver Line Status Interrupt* is disabled, interrupt is not generated but the host can still read the actual status of the transfer from this register.

Table 2.11. Line Status Control Register

Field	Name	Access	Width	Reset
[7]	reserved	RSVD	1	—
[6]	xmitr_empty	RO	1	1'b1
[5]	thr_empty	RO	1	1'b1
[4]	break_cond	RO	1	1'b0
[3]	framing_err	RO	1	1'b0
[2]	parity_err	RO	1	1'b0
[1]	overrun_err	RO	1	1'b0
[0]	data_rdy	RO	1	1'b0

- xmitr_empty
  Transmitter Empty indicator
  1'b0 – Indicates if XMIT FIFO or Transmitter Shift Register has data.
  1'b1 – Indicates if XMIT FIFO and Transmitter Shift Register are both empty.
- thr_empty
  Transmitter Holding Register empty indicator. In FIFO mode, the thr_empty asserts after the last data in FIFO is sent to the Transmitter Shift Register. This bit negates when at least one byte is written to the XMIT FIFO. In non-FIFO mode, the thr_empty asserts when Transmitter Holding Register is empty. The thr_empty bit generates interrupt when IER.thre_int_en is set to logic 1.
  1'b0 – Indicates if XMIT FIFO contains at least one data.
  1'b1 – Indicates if XMIT FIFO is empty.
- break_cond
  Break condition indicator. Break condition occurs when the received data input is held in the Spacing state (logic 0) for longer than a full word transmission time (that is, the total time of Start bit a data bits a Parity a Stop bits). The break_cond bit is reset whenever the Host reads the contents of the Line Status Register. This error is associated with the particular character in the FIFO to which it applies. This error is asserted when its associated character is at the output end of the FIFO. When break occurs, only one zero character is loaded into the FIFO. The next character transfer is enabled after rxd_i goes to the marking state and receives the next valid start bit. The break_cond generates interrupt when asserted if Receiver Line Status Interrupt is enabled (IER.rls_int_en == 1).
  1'b0 – No break condition
  1'b1 – Break condition occurs
- framing_err
  Framing Error Indicator. Framing Error occurs when a received character did not have a valid stop bit. This bit asserts whenever the Stop bit following the last data bit or parity bit is detected as a logic 0 bit (Spacing level). The framing_err bit is reset whenever the Host reads the contents of the Line Status Register. This error is associated with the particular character in the FIFO it applies to. This error is asserted when its associated character is at the output end of the FIFO. The framing_err generates interrupt when asserted if Receiver Line Status Interrupt is enabled (IER.rls_int_en == 1).
  1'b0 – No framing error.
  1'b1 – Framing error occurs
- parity_err
  Parity Error Indicator. Parity error occurs when a received character does not have a correct even or odd parity as selected by LCR.even_parity_sel. The parity_err bit is reset whenever the Host reads the contents of the Line Status Register. This error is associated with the particular character in the FIFO to which it applies. This error is

asserted when its associated character is at the output end of the FIFO. The parity_err generates interrupt when asserted if Receiver Line Status Interrupt is enabled (IER.rls_int_en == 1).

1'b0 – No parity error

1'b1 – Parity error occurs

- overrun_err

  Overrun Error Indicator. Overrun error occurs when RCVR FIFO is full and the next character has been fully received. In this case, the new character is lost. The overrun_err generates interrupt when asserted if Receiver Line Status Interrupt is enabled (IER.rls_int_en == 1).

  1'b0 – No overrun error

  1'b1 – Overrun error occurs

- data_rdy

  Data Ready Indicator. This bit asserts when a complete incoming character is received and transferred into the Receiver Buffer Register or the FIFO. This bit is reset to a logic 0 by reading all of the data in the Receiver Buffer Register or the FIFO.

  1'b0 – Receive Buffer Register has no data

  1'b1 – Receive Buffer Register has data

2.4.7. Divisor Latch Register (DLR_MSB, DLR_LSB)

The UART IP Core contains a programmable baud generator that is capable of dividing the reference clock input (clk_i) by divisors of 1 to (2^{16}–1), and producing a 16 x clock for driving the internal transmitter and receiver logic. Two 8-bit registers (DLR_MSB, DLR_LSB) store the divisor in a 16-bit binary format. These Divisor Latch Registers must be loaded during initialization to ensure proper operation of the baud generator. Upon loading either of the Divisor Latch Registers, a 16-bit Baud Counter is immediately loaded.

These register can be accessed when LCR.dlab=1.

Table 2.12. Line Control Register

DLR_MSB Field	Name	Access	Width	Reset
[7:0]	divisor_msb	RW	8	Refer to attribute: *Baud Rate*

Table 2.13. Line Control Register

DLR_MSB Field	Name	Access	Width	Reset
[7:0]	divisor_lsb	RW	8	Refer to attribute: *Baud Rate*

- divisor_msb

  Upper byte of the Divisor Latch Register

- divisor_lsb

  Lower byte of the Divisor Latch Register

  The divisor ({divisor_msb,divisor_lsb}) is calculated as follows: ({divisor_msb,divisor_lsb}= (frequency input)/ (baud rate). For the example, Table 2.14 shows the values of the Divisor Latch Register for the UART IP supported standard baud rates grid for a system clock of 55.296 MHz.

Table 2.14. Standard Baud Rates Grid with DLR Values for 55.296 MHz System Clock

Supported Baud Rate Grid	Divisor Latch (divisor_msb, divisor_lsb)
2400	23040
4800	11520
9600	5760
14400	3840
19200	2880
28800	1920
38400	1440
56000	987
57600	960
115200	480

2.5. Operation Details

2.5.1. UART Clock Frequency

The UART only has a single clock input, clk_i. This clock is used for the entire IP, including the APB interface for transferring information with the Host internal to the FPGA and the UART's txd_o and rxd_i for sending/receiving serial data with external UART device.

2.5.2. Receiver

In non-FIFO mode, the serial receiver (RXCVER) section contains an 8-bit receiver buffer register (RBR) and receiver shift register (RSR). In FIFO mode, the RBR is a 16-word-deep FIFO.

Since the serial frame is asynchronous to the receiving clock, a high-to-low transition of the rxd_i pin is treated as the start bit of a frame. However, to avoid receiving incorrect data because of rxd_i signal noise, false-start bit detection is implemented. The UART requires the start bit to be low or at least 50% of the baud rate clock. The UART samples rxd_i for half the bit duration, and if a sample is low, a start bit is detected.

Once a valid start bit is received, the data bits, the parity bit, and the stop bit are sampled at the programmed baud rate. The start bit is expected to be exactly equal to the bit duration. Thus, each of the following bits are sampled at the center of the bit period itself.

The receive logic monitors the incoming data stream and reports the state of the incoming line in the line status register (LSR). The LSR is used to indicate overrun, parity, and framing errors. It also indicates when a line break has been received.

Whenever a framing error is detected, the UART assumes that the error was due to the start bit of the following frame and tries to resynchronize it. To do this, it samples the start bit twice. If both samples of the rxd_i are low, the UART resynchronizes and accepts the data following the second start bit. The resynchronization does not occur for a framing error caused by a break character.

Once the entire data bit is received, the data in the Receive Shift Register (RSR) is transferred to the Receive Buffer Register (RBR). The LSR.data_rdy bit is set to 1'b1 to indicate to the CPU that a byte of data has been received. The external

rx_ready_n_o output is asserted (that is, logic 0) simultaneously with the LSR.data_rdy bit. You can also configure the UART to generate an interrupt upon successful transfer from the RSR to the RBR by writing 1'b1 to IER.rda_int_en.

2.5.3. Transmitter

The serial transmitter (TXMITT) section consists of an 8-bit Transmitter Holding Register (THR) and Transmitter Shift Register (TSR) when the UART is in non-FIFO mode. When the UART is in FIFO mode, THR is a 16-word-deep FIFO. The UART provides two methods of indicating the status of THR: a tx_ready_n_o output signal or a Transmitter Holding Register empty (thr_empty) flag in the line status register (LSR). If UART is in non-FIFO mode when THR is empty, the tx_ready_n_o signal becomes low active, and the LSR.thr_empty flag is set to a logic 1'b1. THR should be written only when LSR.thr_empty is 1'b1 or tx_ready_n_o is 1'b0. After the data is loaded in THR, it is loaded to TSR for transmission. When another data is written to THR the LSR.thr_empty is reset to logic 0 and the tx_ready_n_o pin goes inactive high.

If the UART is in FIFO mode, when THR FIFO is not full, the tx_ready_n_o signal becomes low active to signal that the transmitter logic can still receive new data for transition. For this mode, the LSR.thr_empty is set to logic 1'b1 only when FIFO is already empty. This is because the Host in the FPGA fabric cannot know the actual amount of data in FIFO. Thus, it is recommended that the host writes up to 16 data to RBR when LSR.thr_empty is 1'b1. The serial data transmission is automatically enabled after the data is loaded into THR. First, a start bit (logic 0) is transmitted and the data in THR is automatically parallel-loaded to TSR. The data bits are shifted out of TSR, followed by the parity bit, if parity is enabled. Finally, the stop bit (logic 1) is generated to indicate the end of the frame. After a frame is fully transmitted, another frame is transmitted immediately if THR is not empty. This automatic sequencing causes the frames to be transmitted back to back, which increases the transmission bandwidth. The txd_o signal is held high when no transmission is in progress.

2.5.4. Interrupt

The common interrupt request signal (int_o) asserts when any interrupt conditions occurs and the interrupt is enabled in the Interrupt Enable Register (IER). The UART prioritizes interrupts into three levels to minimize external software interaction and records these in the interrupt identification register (IIR). Table 2.15 shows the four levels of interrupt conditions/types, the condition for asserting the interrupts, and the corresponding clearing control.

Performing a read cycle on IIR freezes all interrupts and indicates the highest priority pending interrupt to the Host. The IIR does not acknowledge/record new interrupts until the host services the pending interrupt. Whenever the IIR is read, the current pending interrupt is cleared, meaning the IIR can record the new interrupt again. Any pending lower-priority interrupt becomes visible in the IIR after the clearing the previous interrupt and reading the IIR.

Table 2.15. Interrupt Control Function

IIR[2:0]	Priority	Interrupt Type*	Interrupt Assertion Cause	Interrupt Clearing Control
3'b001	—	None	None	—
3'b110	Highest	Receiver Line Status	Overrun Error or Parity Error or Framing Error or Break Interrupt	Reading the Line Status Register
3'b100	Second	Received Data Available	RCVR FIFO reached the trigger level (FCR.trig_lvl)	Reading the Receiver Buffer Register until RCVR FIFO drops below trigger level
3'b010	Third	Transmitter Holding Register Empty	XMIT FIFO is empty	Writing to XMIT FIFO.
3'b000	Fourth	MODEM Status	Not yet supported	—

***Note:** If the interrupt type is disabled in IER, the interrupt control function treats the interrupt as not asserted even if the source of interrupt is asserted. Hence, the IIR value is affected by the IER.

2.6. Programming Flow

2.6.1. Initialization

The following UART register fields should be set properly before performing UART transaction:

- Line Control Register – even_parity_sel, parity_en, stop_bit_ctrl, char_len_sel
- Divisor Latch Registers – divisor_msb, divisor_lsb

These should match the corresponding setting in the communicating UART for the serial transaction to be successful. Note that reset values of these register fields are configurable during IP generation. Thus in some applications, initialization step is not necessary when attributes are properly set.

2.6.2. Transmit Operation

The steps for transmitting character data through the UART IP Core is shown in the succeeding steps. This assumes that the IP core is not performing transmit operation or at least the XMIT FIFO is empty.

Transmit Operation – Interrupt Mode

1. Write a data to THR. In FIFO mode, you can write up to 16-character data.
2. Set IER.thre_int_en=1'b1 to enable Transmit Holding Register Empty interrupt
3. Wait for Transmit Holding Register Empty interrupt to assert:
   a. Wait for interrupt assertion and check that IIR[3:0]= 4'b0010.
4. If you need to send more characters, repeat Steps 1-3 until all characters are sent.
5. When using interrupt, set IER.thre_int_en=1'b0 to disable the interrupt.

Transmit Operation – Polling Mode

1. Write a data to THR. It is recommended not to enable FIFO for polling mode to save resource.
2. Read LSR until the thr_empty bit asserts.
3. If you need to send more characters, repeat Steps 1 and 2 until all characters are sent.

2.6.3. Receive Operation

The steps for the receiving character data through the UART IP Core is shown in the succeeding steps. This assumes that the IP core is not performing receive operation.

Receive Operation – Interrupt Mode

1. Enable the following interrupts:
   a. Received Data Available Interrupt (IER.rda_int_en=1'b1) – to notify the host that a data is received.
   b. Receiver Line Status interrupt (IER.rls_int_en=1'b1) – to notify the host of receive status such as error and break condition.
2. Wait for interrupt assertion and check that IIR[2:0]= 3'b100 (Receive Data Available). If Receiver Line Status Interrupt asserts (IIR[2:0]=3'b110), read the LSR to determine the cause.
3. If Receiver Line Status Interrupt does not occur, read the character data from RBR:
   a. If Receive Data Available Interrupt occurs, read a data from RBR.
   b. If Character Timeout Interrupt occurs, read LSR. If LSR.data_rdy=1'b1, read RBR.
4. Repeat Steps 2-3 until all expected data are received.

Receive Operation – Polling Mode

1. Read LSR until the thr_empty bit asserts. Also, check that no error status bits are asserted.
2. Read RBR if there is no error.
3. If you need to receive more characters, repeat Steps 1 and 2 until all characters are received.

2.7. Data Format

The character data is written to THR and read from RBR in little endian format as shown in Figure 2.2.

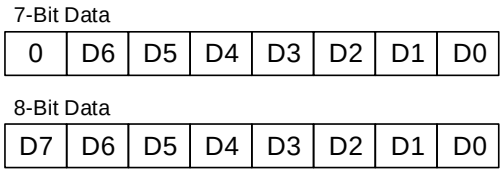

Figure 2.2. Tx/Rx Data Format

2.8. Timing Diagrams

During transmission, data is shifted out by the transmitter logic in LSB first format as shown in Figure 2.3. This timing diagram shows the transmission of 1 character without flow control. The UART IP Core then sends the Start Bit, character data, and Stop Bit on the txd_o line.

Figure 2.3. Transmit Operation Timing Diagram without Flow Control Signals (1 byte)

When transmitting multiple data bytes, the next byte is transmitted immediately after the stop bit for the first byte. This is shown in Figure 2.4. In this diagram, one stop bit is generated.

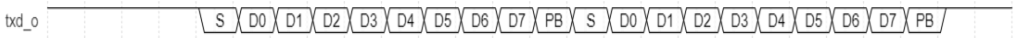

Figure 2.4. Transmit Operation Timing Diagram without Flow Control Signals (2 bytes)

Figure 2.5 shows the receive operation wherein the external UART transmit the data being received. The data format is similar to Figure 2.3 but the transfer occurs in rxd_i signal and transfer direction is opposite.

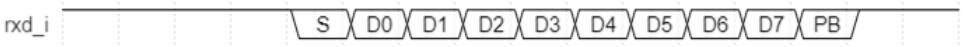

Figure 2.5. Receive Operation Timing Diagram with Flow Control Signals (1 byte)

Figure 2.6 shows the timing diagram for receiving multiple data bytes. Similar to Figure 2.4, The next transaction is started immediately after the previous stop bit.

Figure 2.6. Transmit Operation Timing Diagram with Flow Control Signals (2 bytes)

The APB interface is not described in this document. Refer to AMBA 3 APB Protocol v1.0 Specification for information and the timing diagram of the APB interface. The UART's APB interface has no wait state for both write and read access.

Appendix A. Resource Utilization

Table A.1 shows resource utilization of UART for the MachXO3D using Lattice Synthesis Engine of Lattice Diamond Software.

Table A.1. Resource Utilization

Configuration	Clk Fmax (MHz)*	PFU Registers	LUTs	EBRs
Default	107.898	150	232	0
“FIFO Enable” is Checked, Others = Default	102.323	267	334	0

***Note:** Fmax is generated when the FPGA design only contains I^2C Slave IP Core and the target Frequency is 50 MHz. These values may be reduced when user logic is added to the FPGA design.

References

- MachXO2 Web Page at www.latticesemi.com
- MachXO3 Web Page at www.latticesemi.com
- MachXO3D FPGA Web Page in latticesemi.com
- Mach-NX FPGA Web Page in latticesemi.com
- CrossLink-NX Web Page in latticesemi.com
- Certus-NX Web Page in latticesemi.com

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

Revision History

Revision 1.2, April 2021

Section	Change Summary
Introduction	Updated Table 1.1 to add support for MachXO2 and MachXO3.
References	Added references to MachXO2 and MachXO3.

Revision 1.1, December 2020

Section	Change Summary
Introduction	Modified second paragraph and added Table 1.1. FPGA Software for IP Configuration, Generation, and Implementation.
References	Updated this section.

Revision 1.0, May 2020

Section	Change Summary
All	Initial release.

www.latticesemi.com