Universal Asynchronous Receiver/Transmitter

June 2011 **Reference Design 1011**

Introduction

The Universal Asynchronous Receiver Transmitter (UART) is a popular and widely-used device for data communication in the field of telecommunication. There are different versions of UARTs in the industry. Some of them contain FIFOs for the receiver/transmitter data buffering and some of them have the 9 data bits mode (start bit + 9 data bits + parity + stop bits). This reference design describes a fully configurable UART optimized for and implemented in a variety of Lattice devices, which offer superior performance and architecture compared to existing semiconductor ASSPs (application-specific standard products).

Figure 1 shows the major blocks implemented in this reference design. The receiver performs serial-to-parallel conversion on the asynchronous data frame received from the serial data input SIN. The transmitter performs parallel-to-serial conversion on the 8-bit data received from the CPU. In order to synchronize the asynchronous serial data and to insure the data integrity, start, parity and stop bits are added to the serial data. An example of the UART frame format is shown in Figure 2.

Figure 1. UART Block Diagram

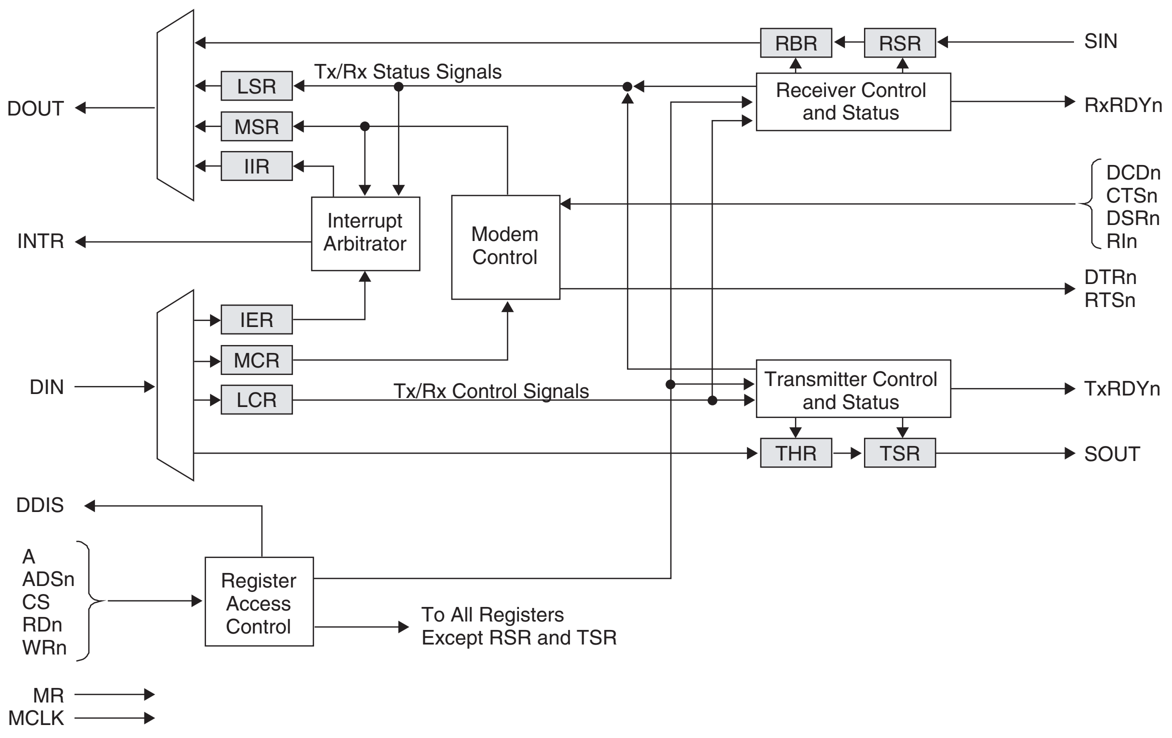
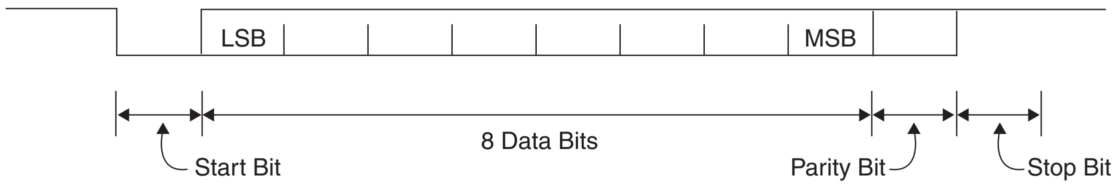


Figure 2. UART Frame Format: (1 Start Bit, 8 Data Bits, 1 Parity Bit, 1 Stop Bit)



This design can also be instantiated many times to get multiple UARTs in the same device. For easy instantiation of the design into a larger implementation, the bi-directional data bus is separated into two buses, DIN and DOUT, instead of using tri-state buffers. The transmitter and receiver both share a common internal Clk16X clock. This internal clock which needs to be 16 times of the desired baud rate clock frequency is obtained from the on-board clock through the MCLK input directly.

Features

- Functionally compatible with the NS16450 UART
- Raster performance than industry standard hardwired devices
- Inserts or extracts standard asynchronous communication bits (start, stop and parity) to or from the serial data
- Holding and shifting registers eliminate the need for precise synchronization between the CPU and serial data
- Standard CPU Interface
- Separate interrupt lines for data received (RxRdyn) and data transmitted (TxRdyn)
- 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 when a modem status flag change is detected.
- Fully-prioritized interrupt system control
- MODEM interface functions (CTS, RTS, DSR, DTR, RI and DCD)
- Fully programmable serial interface characteristics:
  - 5, 6, 7 or 8-bit characters
  - 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
- Separate input and output data buses for use as an embedded module in a larger design
- Transmitter enabled by new data write to transmit holding register
- Receiver synchronizes off the start bit
- Receiver samples all incoming bits at the center of each bit

Signal Descriptions

Table 1. Signal Descriptions

Signal	Type	Descriptions
Global Reset and Clock Interface		
MR	Input	Master reset
MCLK	Input	Master clock
Processor Interface		
A	Input	Address bus
DIN	Input	Data bus input
DOUT	Output	Data but output
ADSn	Input	Address strobe
CS	Input	Chip select
RDn	Input	Read
WRn	Input	Write
DDIS	Output	Driver disable
INTR	Output	Interrupt
Receiver Interface		
SIN	Input	Receiver serial input
RxRDYn	Output	Receiver ready
Transmitter Interface		
SOUT	Output	Transmitter serial output
TxRDYn	Output	Transmitter ready
Modem Interface		
DCDn	Input	Data carrier detect
CTSn	Input	Clear to send
DSRn	Input	Data set ready
RIn	Input	Ring indicator
DTRn	Output	Data terminal ready
RTSn	Output	Request to send

Register Descriptions

The UART contains two data buffering registers (RBR and THR), three status registers (IIR, LSR, and MSR), and three control registers (IER, LCR, and MCR). These registers and their addresses are shown below. Further information on the data buffering registers can be found in the Transmitter and Receiver sections of this document. Note that the programmable baud rate control register and the user accessible scratchpad register are not implemented in this design.

Table 2. Register Map

Register Name	Offset	31-16	15-8	7	6	5	4	3	2	1	0
RBR (Receive buffer Register)/ THR (Transmit Holding Register)	0x0	—	—	D7	D6	D5	D4	D3	D2	D1	D0
IER (Interrupt Enable Register)	0x1			0	0	0	0	MSI	RLSI	THRI	RBRI
IIR (Interrupt Identification Register)	0X2			0	0	0	0	0	ID1	ID0	STAT
LCR (Line Control Register)	0x3			0	SB	SP	EPS	PEN	STB	WLS1	WLS0
LSR (Line Status Register)	0x5			0	TEMT	THRE	BI	FE	PE	OE	DR

Interrupt Enable Register (IER, Address = 001)

7	6	5	4	3	2	1	0
0	0	0	0	MSI	RLSI	THRI	RBRI

RBRI: Receiver Buffer Register Interrupt (1 = Enable, 0 = Disable)
THRI: Transmitter Hold Register Interrupt (1 = Enable, 0 = Disable)
RLSI: Receiver Line Status Interrupt (1 = Enable, 0 = Disable)
MSI: MODEM Status Interrupt (1 = Enable, 0 = Disable)

Interrupt Identification Register (IIR, Address = 010)

7	6	5	4	3	2	1	0
0	0	0	0	0	ID1	ID0	STAT

Four prioritized interrupt levels and sources:

Level	ISR Bit [3:0]	Source of Interrupt	Interrupt Reset Control
None	Xxx1	None	None
Highest	0110	LSR error flags (OE/PE/FE/BI)	Reading LSR
Second	0010	LSR receiver data ready flag (DR)	Reading RBR
Third	0010	LSR flag THR empty (THRE)	Reading IIR or writing THR

Line Control Register

The Line Control Register (LCR) is used to specify the asynchronous data communication format.

7	6	5	4	3	2	1	0
0	SB	SP	EPS	PEN	STB	WLS1	WLS0

WLS1:0 – Word Length Select (Bit1- 0):
- 00 - 5 bit
- 01 - 6 bit
- 10 - 7 bit
- 11 - 8 bit

STB - Stop Bit Length (Bit 2):
- 0 - 1 stop
- 1 -
- WLS1:0 = 00: 1.5 stop
- WLS1:0 = 01: 2 stop
- WLS1:0 = 10: 2 stop
- WLS1:0 = 11: 2 stop

Parity enable:
- 0 – Parity bit disable
- 1 – Parity bit enable

Stick parity:
- 0 – Stick parity disable
- 1 – When PEN, EPS, or SP is set (that is, 1), parity is sent or checked for 0. When PEN or SP is set, EPS is clear, parity is sent or checked for 1.

Tx break:
- 0 – Disable break assertion
- 1 – Assert break. SOUT is driven low active (break character) as long as this bit is 1. It has no effect on transmitter logic.

MODEM Control Register (MCR, Address = 100)

7	6	5	4	3	2	1	0
0	0	0	0	0	0	RTS	DTR

DTR: This bit controls the Data Terminal Ready (DTRn) output.
DTR=0: force DTRn output to a logic 1 (normal default)
DTR=1: force DTRn output to a logic 0

RTS: This bit controls the Request to Send (RTSn) output
RTS=0: force RTSn output to a logic 1 (normal default)
RTS=1: force RTSn output to a logic 0

Line Status Register

7	6	5	4	3	2	1	0
0	TEMT	THRE	BI	FE	PE	OE	DR

DR: Receiver Data Ready
DR indicates the status of RBR. It will be set to logic 1 when RBR data is valid and will be reset to logic 0 when RBR is empty. When line errors (OE/PE/FE/BI) happen, DR will also be set to logic 1 and RBR will be updated to reflect the data bits portion of the frame. Pin RxRDYn is a complement of this bit.

OE: Overrun Error
This bit will be set when the next character is transferred into RBR before the previous RBR data is read by the CPU. Even though DR will still be 1 when OE is set to logic 1, the previous frame data stored in RBR which is not read by the CPU is removed and cannot be recovered.

PE: Parity Error
This bit will be set to logic 1 only when the parity is enabled and the parity bit is not at the correct logic state. For even parity, the parity bit should be 1 if an odd number of 1s in the data bits is received; otherwise, the parity bit should be 0. For odd parity, the parity bit should be 1 if an even number of 1s in the data bits is received; otherwise, the parity bit should be 0. For stick parity '1', the parity bit should be 1. For stick parity '0', the parity bit should be 0.

FE: Framing Error
FE will be reset to logic 0 whenever SIN is sampled high at the center of the first stop bit, regardless of how many stop bits the UART is configured to.

BI: Break Interrupt
BI will be set to logic 1 whenever SIN is low for longer than the whole frame (the time of start bit + data bits + parity bit + stop bits), not at the SIN rising edge where break is negated. If SIN is still low after BI is reset to logic 0 by reading LSR, BI will not be set to logic 1 again. Since break is also a framing error, FE will also be set to 1 when BI is set.

THRE: THR Empty
THRE will be set to logic 1 whenever THR is empty which indicates that the transmitter is ready to accept new data to transmit. Pin TxRDYn is a complement of this bit.

TEMT: Both THR and TSR are Empty
This bit will be set to logic 1 when THRE is set to 1 and the last data bit in the TSR is shifted out through SOUT.

The four error flags (OE, PE, FE and BI) of LSR will be reset to logic 0 after a LSR read.

MODEM Status Register (MSR, Address =110)

7	6	5	4	3	2	1	0
DCD	RI	DSR	CTS	DDCD	TERI	DDSR	DCTS

DCD (Data Carrier Detect) is the complement of DCDn input.
RI (Ring Indicator) is the complement of RIn input.
DSR (Data Set Ready) is the complement of DSRn input.
CTS (Clear to Sent) is the complement of CTSn input.
DDCD (Delta DCD) indicates that the DCDn input has changed state.
TERI (Trailing Edge of RI) indicates that the RIn input has changed from a low to high state.
DDSR (Delta DSR) indicates that the DSRn has changed state since the last time it was read by the CPU.
DCTS (Delta CTS) indicates that the CTSn has changed state since the last time it was read by the CPU.

When bits 0-3 are set to logic 1, a MODEM status interrupt is generated if the interrupt is enabled. These four bits will be reset to logic 0 when the CPU reads MSR.

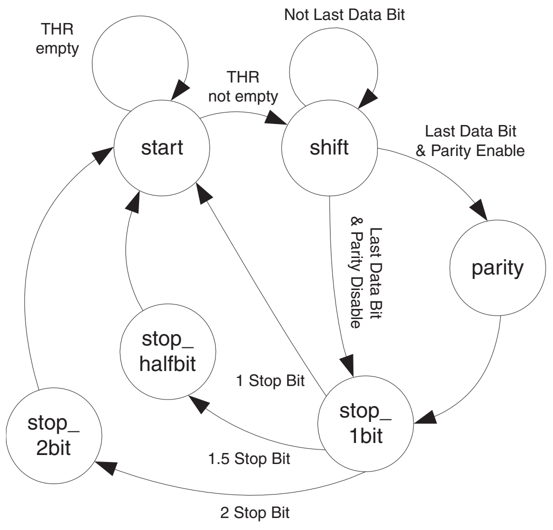
Transmitter

The serial transmitter section consists of an 8-bit Transmitter Hold Register (THR) and Transmitter Shift Register (TSR). There are two ways to indicate the status of THR: an independent TxRDY output pin or the THRE flag in the Line Status Register (LSR). When the THR is empty, pin TxRDYn will be low active, and the THR empty flag in LSR will be set to a logic 1. Only when the THR is empty, can a write operation be performed to transfer the data from CPU to THR without trashing the previous data. After the data is loaded in THR, the THR empty flag in LSR will be reset to logic 0, and pin TxRDYn will go inactive high.

The serial data transmission will be automatically enabled after the data is loaded into THR. First a start bit (logic 0) is transmitted and the data in THR is parallel loaded to TSR automatically. Then the data bits will be shifted out of TSR with certain word lengths defined in Line Control Register (LCR) followed by the parity bit if parity is enabled. Finally, the stop bit (logic 1) is generated to indicate the end of the frame. This serial data frame (start bit + data bits + parity bit + stop bit) will be transmitted at the rate of 1/16 of Clk16X frequency. After a frame is fully transmitted, another frame will be transmitted immediately if THR is not empty (due to a THR write occurring during the first frame of transmission). This automatic sequencing causes the frames to be transmitted back-to-back which increases the transmission bandwidth. When no transmission is taking place, the SOUT pin is held in the high state.

The behavior of the transmitter is controlled by the FSM (Finite State Machine) shown in Figure 3:

Figure 3. Transmitter State Machine



<start>: When the UART is reset by the MR pin, the transmitter FSM will be reset to this state. When in this state, the transmitter is waiting to assert the start bit. A start bit will be asserted as soon as the THR is not empty. Once a low SOUT (start bit) is asserted, the FSM will switch to <shift> state.

<shift>: When the FSM is in this state, it is waiting for the last (most significant) data bit to be shifted out. After the last data bit is shifted out, the FSM will switch to <parity> state if parity is enabled. Otherwise, it will switch to <stop_1bit> state.

<parity>: When the FSM is in this state, the last data bit is still in transmission. When the transmission is complete, the FSM will assert the parity bit. Once the parity bit is asserted, the FSM switch to the <stop_1bit> state.

<stop_1bit>: No matter if the stop bit is configured to be 1, 1.5 or 2 bits long, the FSM will always switch to this state, wait for a baud clock cycle, and then assert the stop bit(s). For one stop bit, the FSM switches back to the <start> state and waits to assert the start bit of another frame. For 1.5 stop bits, it switches to <stop_halfbit> state and stays there for just half a baud clock cycle before switching to <start> state. For two stop bits, it switches to <stop_2bit> state then switches back to <start> state. Note that the stop bit(s) is asserted at the time when the FSM is leaving the <stop_1_bit> state.

<stop_halfbit>: This state is for 5-bit data bits with a 1.5 stop bit. The FSM will stay in this state for only half baud clock cycle and then switch to <start> state.

<stop_2bit>: When the FSM is in this state, the first stop bit is in transmission. It waits for a baud clock cycle, then asserts the second stop bit and switches to the <start> state.

Receiver

The serial receiver section also contains an 8-bit Receiver Buffer Register (RBR) and Receiver Shift Register (RSR). The status of RBR can be provided by either independent pin RxRDYn or the Receiver Data available flag (DR) in LSR.

Since the serial frame is asynchronous to the receiving clock, a high to low transition of SIN pin will be treated as the Start bit of a frame. However, in order to avoid receiving a incorrect data due to SIN signal noise, the False Start Bit Detection feature is implemented in the design which requires the start bit to be low at least 50% of the receiving baud rate clock cycle. Since the internal clock Clk16X is 16 times the receiving/transmitting baud rate clock frequency, the start bit needs to be low at least eight Clk16X clocks to be considered as a valid start bit.

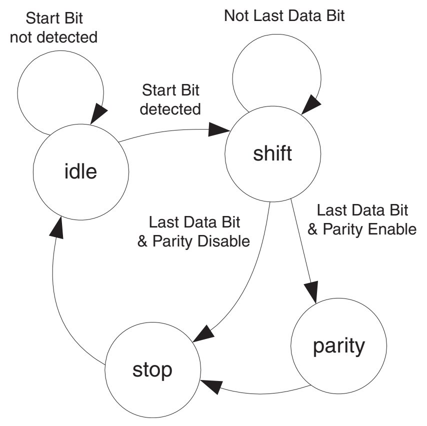
Once a valid eight Clk16X clocks start bit is received, the data bits and parity bit will be sampled every 16 Clk16X clocks (the receiving baud rate). If the start bit is exactly 16 Clk16X clocks long, each of the following bits will be sampled at the center of the bit itself. LSR will be updated to show the received frame status when any of the line errors (overrun error, parity error, framing error, break) is detected.

Whenever the 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 based on the Clk16X clock. If both samples of the SIN are low, the UART will take in the following frame's data bits after the eight Clk16X clocks start bit is sampled. The resynchronization will not occur for the framing error caused by the break.

When the data is available in RBR, the RxRDYn will be low active, and the Receiver Data available flag (DR) in LSR will be set to logic 1 to inform the CPU that the data is ready to be read.

The behavior of the receiver is controlled by the FSM shown in Figure 4.

Figure 4. Receiver State Machine



<idle>: When the MR pin resets the UART, the receiver FSM will be reset to this state. When in this state, it is waiting for SIN to be changed from high to low and stay low for eight Clk16X clocks to be considered a valid start bit. Once a valid start bit is detected, the FSM will switch to <shift> state.

<shift>: When the FSM is in this state, it waits 16 Clk16X clocks for each data bit to shift into RSR. After the last data bit is shifted in, the FSM will switch to <parity> state if parity is enabled. Otherwise, it will switch to <stop> state.

<parity>: When the FSM is in this state, it waits for 16 Clk16X clocks and then samples the parity bit. Once the parity bit is sampled, the FSM switches to the <stop> state.

<stop>: No matter if the stop bit length is configured to be 1, 1.5 or 2 bits long, the FSM will always wait for 16 Clk16X clocks and then samples the stop bit. As long as a logic 1 is sampled at the stop bit, the framing error flag (FE) in LSR will not be set. The receiver does not check whether the stop bit is in the right length as configured. The FSM switches back to <idle> state after the stop bit sampling.

Interrupt

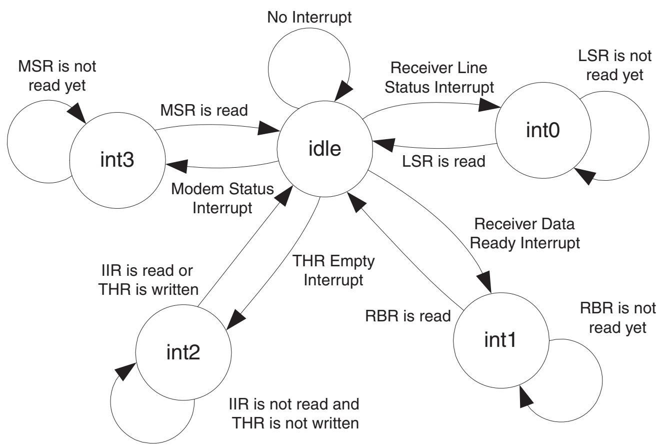
The UART common interrupt request pin INTR will go high active when any of the interrupt conditions is matched and enabled by Interrupt Enable Register (IER).

The UART prioritizes interrupts into four levels to minimize external software interaction, and records these in the Interrupt Identification Register (IIR). The four levels of interrupt conditions in order of priority are Receiver Line Status; Received Data Ready; Transmitter Holding Register Empty and MODEM Status.

Performing a read cycle on IIR freezes all interrupts and indicates the highest priority pending interrupt to the CPU. No other interrupts are acknowledged until the pending interrupt is serviced. When the IIR is read, the current pending interrupt is cleared. A lower level interrupt may be seen at next IIR reading.

The behavior of the interrupt is controlled by the FSM shown in Figure 5.

Figure 5. Interrupt State Machine



<idle>: When the MR pin resets the UART, the interrupt FSM will be reset to this state. When in this state, it is waiting for the enabled interrupt conditions to be true and then switches to the interrupt state with highest priority.

<int0>: The FSM switches to this state when the highest priority level interrupt occurs. It stays at this state until the LSR is read.

<int1>: The FSM switches to this state when the second priority level interrupt occurs. It stays at this state until the RBR is read.

<int2>: The FSM switches to this state when the third priority level interrupt occurs. It stays at this state until the IIR is read or after THR is written.

<int3>: The FSM switches to this state when the fourth priority level interrupt occurs. It stays at this state until the MSR is read.

The interrupts continue to be generated as long as the corresponding enable bit in IER is set and the corresponding interrupt condition is matched. Since the interrupt state machine is running at Clk16X, the INTR pin might be low for a Clk16X clock and then return to high again.

Testbench Description

The following files are needed for simulation:

1. uart_tx_tb.vhd – Transmit testbench
2. uart_rx_tb.vhd – Receiver testbench
3. uart_int_tb.vhd – Interrupt testbench

Timing Specifications

The timing diagrams below show an 8-bit pattern “55” written to the transmit hold register. This in turn initiates a UART transmit on the SOUT line after TEMT is read and once the transmit empty flag is set in the line status register.

Figure 6. UART Transmit Timing Diagram

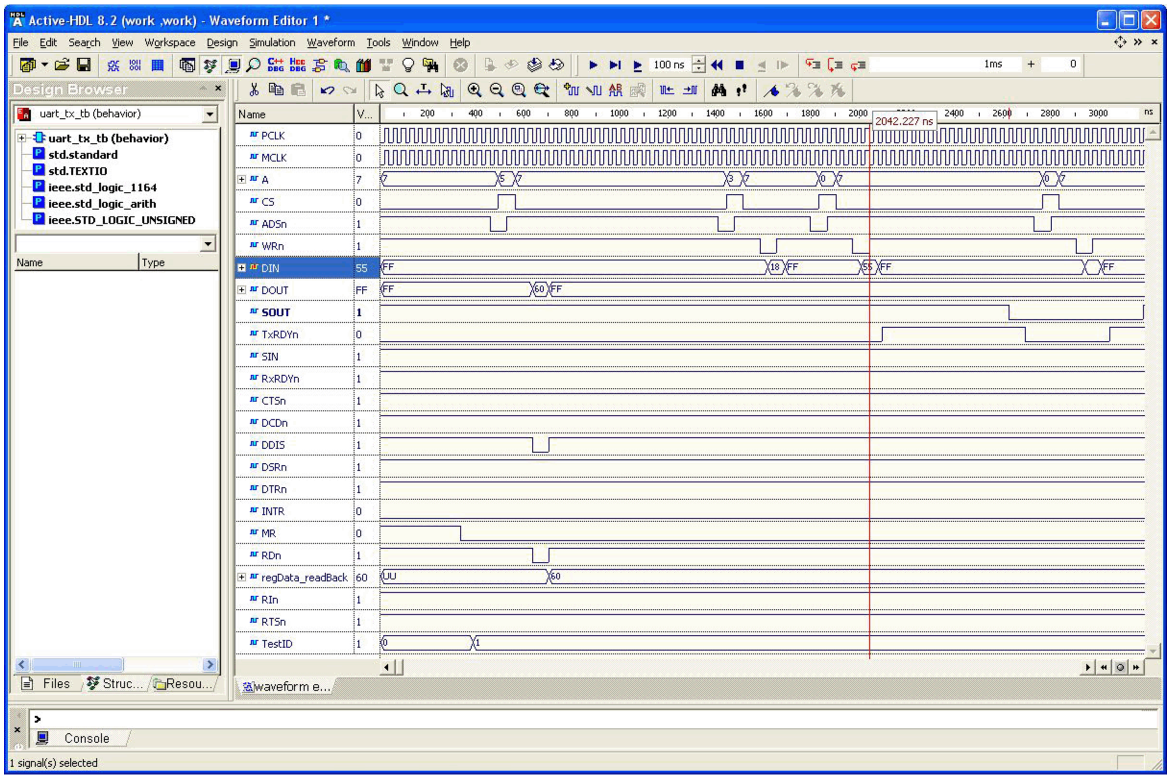
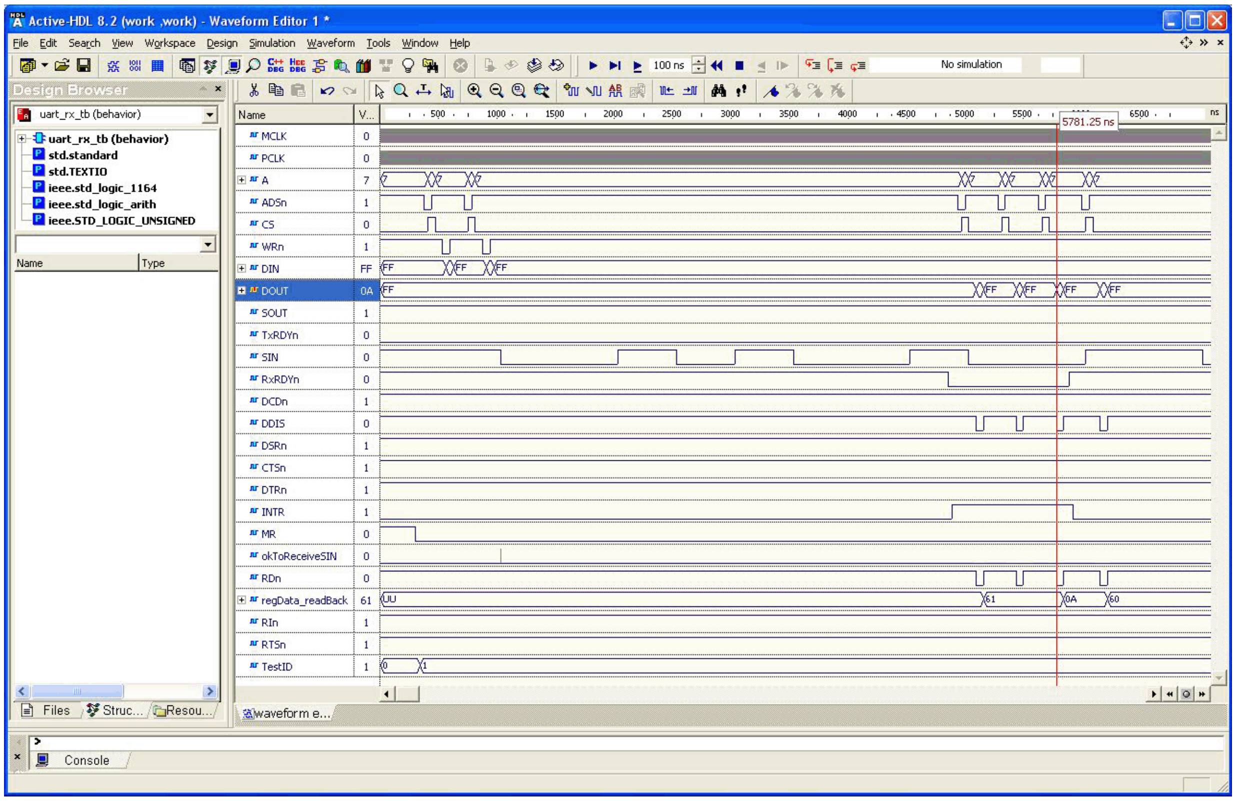


Figure 7 shows the data received on the receiver serial data input line SIN.

Figure 7. UART Receiver Timing Diagram



Implementation

This design is implemented in VHDL. When using this design in a different device, density, speed, or grade, performance and utilization may vary. Default settings are used during the fitting of the design.

Table 3. Performance and Resource Utilization

Device Family	Language	Speed Grade	Utilization	f_{MAX} (MHz)	I/Os	Architecture Resources
MachXO2™ [1]	VHDL	-5	206 LUTs	>120	37	N/A
MachXO™ [2]	VHDL	-4	205 LUTs	>120	37	N/A
LatticeECP3™ [3]	VHDL	-6	203 LUTs	>120	37	N/A
LatticeXP2™ [4]	VHDL	-5	211 LUTs	>120	37	N/A
ispMACH® 4000ZE[5]	VHDL	-5 (ns)	139 Macrocells	>120	37	N/A
Platform Manager™ [6]	VHDL	-3	208 LUTs	>120	37	N/A

1. Performance and utilization characteristics are generated using LCMXO2-256HC-5TG100C, with Lattice Diamond® 1.3 design software.
2. Performance and utilization characteristics are generated using LCMXO256C-4T100C with Lattice Diamond 1.3 design software.
3. Performance and utilization characteristics are generated usingLFE3-17EA-6FTN256C with Lattice Diamond 1.3 design software.
4. Performance and utilization characteristics are generated using LFXP2-5E-5M132C with Lattice Diamond 1.3 design software.
5. Performance and utilization characteristics are generated using LC4128ZE-5TN100C with ispLEVER® Classic 1.4 design software.
6. Performance and utilization characteristics are generated using LPTM10-1247-3TG128CES with ispLEVER 8.1 SP1 design software.

Technical Support Assistance

Hotline:	1-800-LATTICE (North America)
	+1-503-268-8001 (Outside North America)
e-mail:	techsupport@latticesemi.com
Internet:	www.latticesemi.com

Revision History

Date	Version	Change Summary
—	—	Previous Lattice releases.
February 2009	01.1	Updated for MachXO support.
July 2009	01.2	Updated for ispMACH 4000ZE support.
December 2009	01.3	Added support for LatticeXP2 device family.
December 2010	01.4	Added support for Platform Manager device family.
		Added support for Lattice Diamond 1.1 and ispLEVER 8.1 SP1 design software.
April 2011	01.5	Added support for MachXO2 device family.
		Added support for Lattice Diamond 1.2 design software.
June 2011	01.6	Added support for LatticeECP3 device family.
		Added support for Lattice Diamond 1.3 design software.