Introduction (Ask a Question)

Core16550 is a standard Universal Asynchronous Receiver-Transmitter (UART) that ensures software compatibility with the widely used 16550 device. It handles serial-to-parallel data conversion for inputs from modems or other serial devices and performs parallel-to-serial conversion for data sent from the CPU to these devices.

During transmission, data is written in parallel into the UART's transmit First-In, First-Out (FIFO) buffer. The data is then serialized for output. When receiving, the UART converts incoming serial data into a parallel and enables easy access for the processor.

A typical application of the 16550 UART is illustrated in the following figure.

Figure 1. Typical 16550 Application

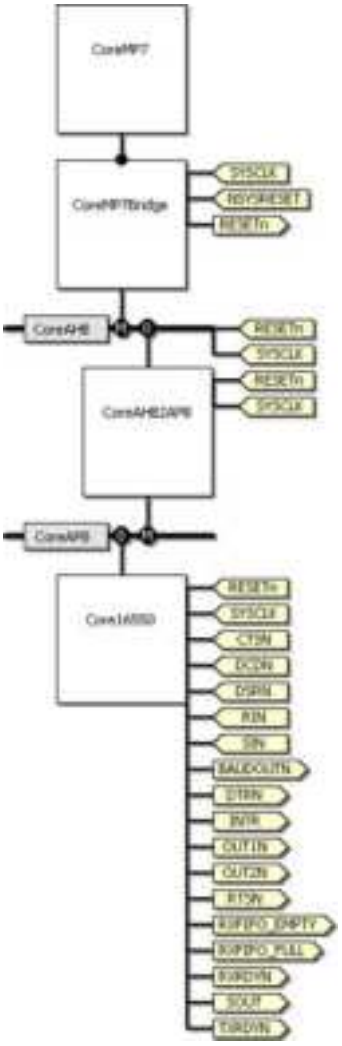


Table 1. Core16550 Summary

Core Version	This document applies to Core16550 v3.4
Supported Device Families	• RT PolarFire® • PolarFire® SoC • PolarFire • RTG4™ • IGLOO® 2 • SmartFusion® 2 • SmartFusion® • ProASIC3E • ProASIC3 • ProASIC® 3L • IGLOOe • IGLOO® • Fusion
Supported Tool Flow	Requires Libero® SoC v8.6 or later releases.
Licensing	No license is required for using this core. Complete verilog RTL source code is provided for the core and testbench.
Installation Instructions	Core16550 must be installed to the IP Catalog of Libero SoC automatically through the update function. Alternatively, Core16550 is manually downloaded from the catalog. Once the IP core is installed, it is configured, generated and instantiated within the SmartDesign for inclusion in the project.
Device Utilization and Performance	A summary of utilization and performance information for the Core16550 is listed in the Device Utilization and Performance section.

Key Features

Following are the key features of Core16550:

- Transmitter and receiver are each buffered with upto 16-byte FIFOs to reduce the number of interrupts presented to the CPU.
- Adds or strips standard asynchronous communication bits (Start, Stop and Parity).
- Independently controlled transmit, receive, line status and data set interrupts
- Programmable baud generator
- Modem control functions (CTSn, RTSn, DSRn, DTRn, RIn and DCDn).
- Advanced Peripheral Bus (APB) register interface

Discontinued Features

Very High Speed Integrated Circuit (VHSIC) Hardware Description Language (VHDL) support will be discontinued from this version.

Core16550 Change Log Information

This section provides a comprehensive overview of the newly incorporated features, beginning with the most recent release.

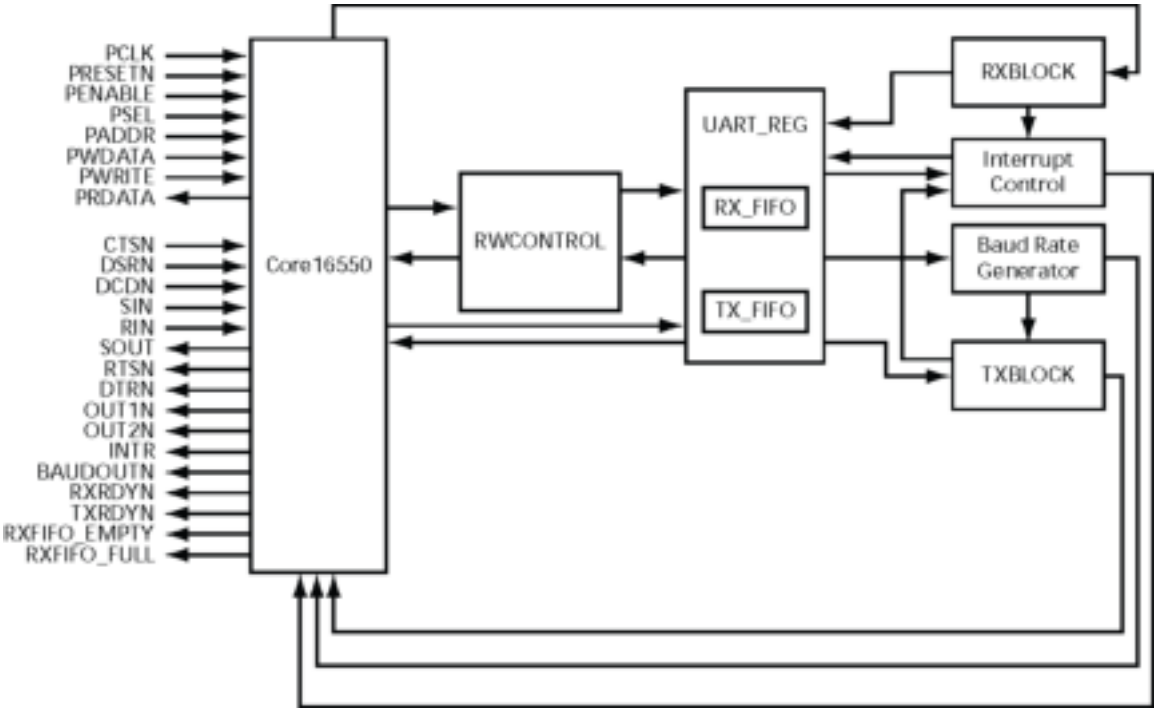
Version	What's New
Core16550 v3.4	• Core16550 uses system verilog keyword "break" as register name which was causing syntax error issue. The keyword is replaced with another name to resolve this issue. • Added PolarFire® family support
Core16550 v3.3	Added Radiation-tolerant FPGA (RTG4™) family support

Table of Contents

1. Functional Block Description (Ask a Question)

This section provides a short description for each element of the internal block diagram as displayed in the following figure.

Figure 1-1. Core16550 Block Diagram



1.1. Elements of the Internal Block Diagram (Ask a Question)

The following section provides information about elements of the the internal block diagram.

1.1.1. RWControl (Ask a Question)

The RWControl block is responsible for handling the communications with the processor (parallel) side of the system. All the writing and reading of Internal registers are accomplished through this block.

1.1.2. UART_Reg (Ask a Question)

The UART_Reg block holds all of the device Internal registers.

1.1.3. RXBlock (Ask a Question)

This is the receiver block. RXBlock receives the incoming serial word. It is programmable to recognize data widths, such as 5, 6, 7 or 8 bits; various parity settings, such as even, odd or no-parity; and different stop bits, such as 1, 1½ and 2 bits. RXBlock checks for errors in the input data stream, such as overrun errors, frame errors, parity errors and break errors. If the incoming word has no problems, it is placed in the receiver FIFO.

1.1.4. Interrupt Control (Ask a Question)

The Interrupt Control block sends an interrupt signal back to the processor, depending on the state of the FIFO and its received and transmitted data. The Interrupt Identification register provides the level of the interrupt. Interrupts are sent for empty transmission/receipt buffers (or FIFOs), an error in receiving a character, or other conditions requiring the attention of the processor.

1.1.5. Baud Rate Generator (Ask a Question)

This block takes the input PCLK and divides it by a programmed value (from 1 to 2^{16} – 1). The result is divided by 16 to create the transmission clock (BAUDOUT).

1.1.6. TXBlock (Ask a Question)

The Transmit block handles the transmission of data written to the Transmit FIFO. It adds the required Start, Parity and Stop bits to the data being transmitted so the receiving device can do the proper error handling and receiving.

1.2. Software Interface (Ask a Question)

The Core16550 register definitions and address mappings are described in this section. The following table shows the Core16550 register summary.

Table 1-1. Core16550 Register Summary

PADDR[4:0] (Address)	Divisor Latch Access Bit[1] (DLAB)	Name	Symbol	Default (reset) Value	No. of Bits	Read/Write
00	0	Receiver Buffer Register	RBR	XX	8	R
00	0	Transmitter Holding Register	THR	XX	8	W
00	1	Divisor Latch (LSB)	DLR	01h	8	R/W
04	1	Divisor Latch (MSB)	DMR	00h	8	R/W
04	0	Interrupt Enable Register	IER	00h	8	R/W
08	X	Interrupt Identification Register	IIR	C1h	8	R
08	X	FIFO Control Register	FCR	01h	8	W
0C	X	Line Control Register	LCR	00h	8	R/W
10	X	Modem Control Register	MCR	00h	8	R/W
14	X	Line Status Register	LSR	60h	8	R
18	X	Modem Status Register	MSR	00h	8	R
1C	X	Scratch Register	SR	00h	8	R/W

Important:

1. DLAB is the MSB of the Line Control Register (LCR bit 7).

1.2.1. Receiver Buffer Register (Ask a Question)

The Receiver Buffer register is defined in the following table.

Table 1-2. Receiver Buffer Register (Read Only)—Address 0 DLAB 0

Bits	Name	Default State	Valid States	Function
7..0	RBR	XX	0..FFh	Received data bits. Bit 0 is the LSB, and is the first received bit.

1.2.2. Transmitter Holding Register (Ask a Question)

The Transmitter Holding register is defined in the following table.

Table 1-3. Transmitter Holding Register—Write Only

Bits	Name	Default State	Valid States	Function
7..0	THR	XX	0..FFh	To transmit data bits. Bit 0 is the LSB, and is transmitted first.

1.2.3. FIFO Control Register (Ask a Question)

The FIFO Control register is defined in the following table.

Table 1-4. FIFO Control Register—Write Only

Bits (7:0)	Default State	Valid States	Function
0	1	0, 1	Enables both the Transceiver (Tx) and Receiver (Rx) FIFOs. This bit must be set to 1 when other FCR bits are written to or they will not be programmed. 0: Disabled 1: Enabled
1	0	0, 1	Clears all bytes in the Rx FIFO and resets its counter logic. The Shift register is not cleared. 0: Disabled 1: Enabled
2	0	0, 1	Clears all bytes in the Tx FIFO and resets its counter logic. The Shift register is not cleared. 0: Disabled 1: Enabled
3	0	0, 1	0: Single transfer DMA: Transfer made between CPU bus cycles 1: Multi-transfer DMA: Transfers made until Rx FIFO empty or Transmission System Operator (TSO) Transmit (XMIT) FIFO filled. FCR[0] must be set to 1 to set FCR[3] to 1.
4, 5	0	0, 1	Reserved for future use.
6, 7	0	0, 1	These bits are used to set the trigger level for the Rx FIFO interrupt. 7 6 Rx FIFO Trigger Level (bytes) 0 0 01 0 1 04 1 0 08 1 1 14

1.2.4. The Divisor Control Registers (Ask a Question)

The Baud Rate (BR) clock is generated by dividing the input reference clock (PCLK) by 16 and the divisor value.

$$BR = \frac{PCLK}{16 \cdot DivisorValue}$$

The following table lists an example of divisor values for desired BR when using an 18.432 MHz reference clock.

Table 1-5. Divisor Latch (LS and MS)

Bits	Name	Default State	Valid States	Function
7..0	DLR	01h	01..FFh	The LSB of divisor value
7..0	DMR	00h	00..FFh	The MSB of divisor value

Table 1-6. Baud Rates and Divisor Values for 18.432 MHz Reference Clock

Baud Rate	Decimal Divisor (Divisor Value)	Percent Error
50	23040	0.0000%
75	15360	0.0000%
110	10473	–0.2865%
134.5	8565	0.0876%
150	7680	0.0000%
300	3840	0.0000%
600	1920	0.0000%
1,200	920	4.3478%
1,800	640	0.0000%

Table 1-6. Baud Rates and Divisor Values for 18.432 MHz Reference Clock (continued)

Baud Rate	Decimal Divisor (Divisor Value)	Percent Error
2,000	576	0.0000%
2,400	480	0.0000%
3,600	320	0.0000%
4,800	240	0.0000%
7,200	160	0.0000%
9,600	120	0.0000%
19,200	60	0.0000%
38,400	30	0.0000%
56,000	21	–2.0408%

1.2.5. Interrupt Enable Register (Ask a Question)

The Interrupt Enable register is defined in the following table.

Table 1-7. Interrupt Enable Register

Bits	Name	Default State	Valid State	Function
0	ERBFI	0	0, 1	Enables "Received Data Available Interrupt" 0: Disabled 1: Enabled
1	ETBEI	0	0, 1	Enables the "Transmitter Holding Register Empty Interrupt" 0: Disabled 1: Enabled
2	ELSI	0	0, 1	Enables the "Receiver Line Status Interrupt" 0: Disabled 1: Enabled
3	EDSSI	0	0, 1	Enables the "Modem Status Interrupt" 0: Disabled 1: Enabled
7..4	Reserved	0	0	Always 0

1.2.6. Interrupt Identification Register (Ask a Question)

The Interrupt Identification register is listed in the following table.

Table 1-8. Interrupt Identification Register

Bits	Name	Default State	Valid States	Function
3..0	IIR	1h	0..Ch	Interrupt identification bits.
5..4	Reserved	00	00	Always 00
7..6	Mode	11	11	11: FIFO mode

The Interrupt Identification register field is defined in the following table.

Table 1-9. Interrupt Identification Register Field (IIR)

IIR Value[3:0)]	Priority Level	Interrupt Type	Interrupt Source	Interrupt Reset Control
0110	Highest	Receiver line status	Overrun error, parity error, framing error or break interrupt	Reading the Line Status register
0100	Second	Received data available	Receiver data available	Reading the Receiver Buffer register or the FIFO drops below the trigger level

Table 1-9. Interrupt Identification Register Field (IIR) (continued)

IIR Value[3:0)]	Priority Level	Interrupt Type	Interrupt Source	Interrupt Reset Control
1100	Second	Character timeout indication	No characters are read from the Rx FIFO during the last four character times and there was at least one character in it during this time.	Reading the Receiver Buffer register
0010	Third	Transmitter Holding register empty	Transmitter Holding register empty	Reading the IIR or writing into the Transmitter Holding register
0000	Fourth	Modem status	Clear to Send, Data Set Ready, Ring Indicator or Data Carrier Detect	Reading the Modern Status register

1.2.7. Line Control Register (Ask a Question)

The Line Control register is listed in the following table.

Table 1-10. Line Control Register

Bits	Name	Default State	Valid States	Function
1..0	WLS	0	0..3h	Word Length Select 00: 5 bits 01: 6 bits 10: 7 bits 11: 8 bits
2	STB	0	0, 1	Number of Stop Bits 0: 1 Stop bit 1: 1½ Stop bits when WLS = 00 2: Stop bits in other cases
3	PEN	0	0, 1	Parity Enable 0: Disabled 1: Enabled. Parity is added in transmission and checked in receiving.
4	EPS	0	0, 1	Even Parity Select 0: Odd parity 1 : Even parity
5	SP	0	0, 1	Stick Parity 0: Disabled 1: Enabled Following are the parity details, when stick parity is enabled: Bits 4..3 11: 0 will be sent as a Parity bit, and checked in receiving. 01: 1 will be sent as a Parity bit, and checked in receiving.
6	SB	0	0, 1	Set Break 0: Disabled 1: Set break. SOUT is forced to 0. This does not have any effect on transmitter logic. The break is disabled by setting the bit to 0.
7	DLAB	0	0, 1	Divisor Latch Access Bit 0: Disabled. Normal Addressing mode in use. 1: Enabled. Enables access to the Divisor Latch registers during read or write operation to address 0 and 1.

1.2.8. Modem Control Register (Ask a Question)

The Modem Control register is listed in the following table.

Table 1-11. Modem Control Register

Bits	Name	Default State	Valid States	Function
0	DTR	0	0, 1	Controls the Data Terminal Ready (DTRn) output. 0: DTRn <= 1 1: DTRn <= 0
1	RTS	0	0, 1	Controls the Request to Send (RTSn) output. 0: RTSn <= 1 1: RTSn <= 0
2	Out1	0	0, 1	Controls the Output1 (OUT1n) signal. 0: OUT1n <= 1 1: OUT1n <= 0
3	Out2	0	0, 1	Controls the Output2 (OUT2n) signal. 0: OUT2n <= 1 1: OUT2n <= 0
4	Loop	0	0, 1	Loop enable bit 0: Disabled 1: Enabled. The following occurs in Loop mode: SOUT is set to 1. The SIN, DSRn, CTSn, RIn and DCDn inputs are disconnected. The output of the Transmitter Shift register is looped back into the Receiver Shift register. The modem control outputs (DTRn, RTSn, OUT1n and OUT2n) are connected internally to the modem control inputs, and the modem control output pins are set at 1. In Loopback mode, the transmitted data is immediately received, allowing the CPU to check the operation of the UART. The interrupts are operating in Loop mode.
7..4	Reserved	0h	0	Reserved

1.2.9. Line Status Register (Ask a Question)

The Line Status register is defined in the following table.

Table 1-12. Line Status Register—Read Only

Bits	Name	Default State	Valid States	Function
0	DR	0	0, 1	Data Ready indicator 1 when a data byte has been received and stored in the receive buffer or the FIFO. DR is cleared to 0 when the CPU reads the data from the receive buffer or the FIFO.
1	OE	0	0, 1	Overrun Error indicator Indicates that the new byte was received before the CPU read the byte from the receive buffer, and that the earlier data byte is destroyed. OE is cleared when the CPU reads the Line Status register. If the data continues to fill the FIFO beyond the trigger level, an overrun error occurs once the FIFO is full and the next character has been completely received in the Shift register. The character in the Shift register is overwritten, but it is not transferred to the FIFO.
2	PE	0	0, 1	Parity Error indicator Indicates that the received byte had a parity error. PE is cleared when the CPU reads the Line Status register. This error is revealed to the CPU when its associated character is at the top of the FIFO.
3	FE	0	0, 1	Framing Error indicator Indicates that the received byte did not have a valid Stop bit. FE is cleared when the CPU reads the Line Status register. The UART will try to resynchronize after a framing error. To do this, it assumes that the framing error was due to the next Start bit, so it samples this Start bit twice, and then starts receiving the data. This error is revealed to the CPU when its associated character is at the top of the FIFO.

Table 1-12. Line Status Register—Read Only (continued)

Bits	Name	Default State	Valid States	Function
4	BI	0	0, 1	Break Interrupt indicator Indicates that the received data is at 0, longer than a full word transmission time (Start bit + Data bits + Parity + Stop bits). BI is cleared when the CPU reads the Line Status register. This error is revealed to the CPU when its associated character is at the top of the FIFO. When break occurs, only one zero character is loaded into the FIFO.
5	THRE	1	0, 1	Transmitter Holding Register Empty (THRE) indicator Indicates that the UART is ready to transmit a new data byte. THRE causes an interrupt to the CPU when bit 1 (ETBEI) in the Interrupt Enable register is 1. This bit is set when the TX FIFO is empty. It is cleared when at least one byte is written to the TX FIFO.
6	TEMT	1	0, 1	Transmitter Empty indicator This bit is set to 1 when both the transmitter FIFO and Shift registers are empty.
7	FIER	0	1	This bit is set when there is at least one parity error, framing error or break indication in the FIFO. FIER is cleared when the CPU reads the LSR if there are no subsequent errors in the FIFO.

1.2.10. Modem Status Register (Ask a Question)

The Modem Status register is listed in the following table.

Table 1-13. Modem Status Register—Read Only

Bits	Name	Default State	Valid States	Function
0	DCTS	0	0, 1	Delta Clear to Send indicator. Indicates that the CTSn input has changed state since the last time it was read by the CPU.
1	DDSR	0	0, 1	Delta Data Set Ready indicator Indicates that the DSRn input has changed state since the last time it was read by the CPU.
2	TERI	0	0, 1	Trailing Edge of Ring Indicator detector. Indicates that RI input has changed from 0 to 1.
3	DDCD	0	0, 1	Delta Data Carrier Detect indicator Indicates that DCD input has changed state. **Note:** Whenever bit 0, 1, 2 or 3 is set to 1, a Modem Status interrupt is generated.
4	CTS	0	0, 1	Clear to Send The complement of the CTSn input. When bit 4 of the Modem Control Register (MCR) is set to 1 (loop), this bit is equivalent to DTR in the MCR.
5	DSR	0	0, 1	Data Set Ready The complement of the DSR input. When bit 4 of the MCR is set to 1 (loop), this bit is equivalent to RTSn in the MCR.
6	RI	0	0, 1	Ring Indicator The complement of the RIn input. When bit 4 of the MCR is set to 1 (loop), this bit is equivalent to OUT1 in the MCR.
7	DCD	0	0, 1	Data Carrier Detect The complement of DCDn input. When bit 4 of the MCR is set to 1 (loop), this bit is equivalent to OUT2 in the MCR.

1.2.11. Scratch Register (Ask a Question)

The Scratch register is defined in the following table.

Table 1-14. Scratch Register

Bits	Name	Default State	Function
7..0	SCR	00h	Read/Write register for CPU. No effects on UART operation.

2. Tool Flows (Ask a Question)

This section provides the details about the tool flows.

2.1. SmartDesign (Ask a Question)

Core16550 is available for download in the SmartDesign IP deployment design environment. The core is configured using the configuration GUI within SmartDesign, see the following figure.

For information about how to use SmartDesign to instantiate, configure, connect and generate cores, see the SmartDesign User Guide.

Figure 2-1. Core16550 Configuration

2.2. Simulation Flows (Ask a Question)

The user testbench for Core16550 is included in all releases.

To run simulations, select the **User Testbench Flow** option within SmartDesign and click Generate Design under the SmartDesign menu. The user testbench is selected through the Core Testbench Configuration GUI.

When the SmartDesign generates the Libero SoC project, it installs the user testbench files.

To run the user testbench, set the design root to the Core16550 instantiation in the **Libero SoC Design Hierarchy** pane and click the **Simulation** icon in the **SoC Design Flow** window. This invokes ModelSim® and automatically runs the simulation.

2.2.1. Synthesis in Libero SoC (Ask a Question)

Click the **Synthesis** icon in Libero SoC. The **Synthesis** window appears. The Synplify® project. Set Synplify to use the Verilog 2001 standard if Verilog is being used. To run Synthesis, click the **Run** icon.

2.2.2. Place-and-Route in Libero SoC (Ask a Question)

To set the design route appropriately and run Synthesis, click the **Layout** icon in Libero SoC and invoke Designer. Core16550 does not require any special place-and-route settings.

3. Core16550 (Ask a Question)

This section provides information about parameters used in this core.

3.1. Parameters (Ask a Question)

Core16550 does not support any top-level parameters.

4. Core Interfaces (Ask a Question)

This section provides input and output summary.

4.1. I/O Signal Description (Ask a Question)

The following lists the Core16550 I/O definitions.

Table 4-1. I/O Signal Summary

Name	Type	Polarity	Description
PRESETN	Input	Low	Master reset
PCLK	Input	—	Master clock PCLK is divided by the value of the Divisor registers. The result is then divided by 16 to produce the baud rate. The resultant signal is the BAUDOUT signal. The rising edge of this pin is used to strobe all input and output signals.
PWRITE	Input	High	APB write/read enable, active-high. When HIGH, data is written to the specified address location. When LOW, data is read from the specified address location.
PADDR[4:0]	Input	—	APB Address This bus provides the link for the CPU to the address of the register of Core16550 to be read from or written to.
PSEL	Input	High	APB select When this is HIGH along with PENABLE, reading and writing to Core16550 is enabled.
PWDATA[7:0]	Input	—	Data input bus Data on this bus will be written into the addressed register during a write cycle.
PENABLE	Input	High	APB enable When this is HIGH along with PSEL, reading and writing to Core16550 is enabled.
PRDATA[7:0]	Output	—	Data output bus This bus holds the value of the addressed register during a read cycle.
CTSn	Input	Low	Clear to Send This active-low signal is an input showing when the attached device (modem) is ready to accept data. Core16550 passes this information to the CPU through the Modem Status register. This register also indicates that if the CTSn signal has changed since the last time, the register was read.
DSRn	Input	Low	Data Set Ready This active-low signal is an input indicating when the attached device (modem) is ready to set up a link with Core16550. Core16550 passes this information to the CPU through the Modem Status register. This register also indicates if the DSRn signal has changed since the last time the register was read.
DCDn	Input	Low	Data Carrier Detect This active-low signal is an input indicating when the attached device (modem) has detected a carrier. Core16550 passes this information to the CPU though the Modem Status register. This register also indicates if the DCDn signal has changed since the last time the register is read.
SIN	Input	—	Serial Input Data This data is transmitted into Core16550. It is synchronized with the PCLK input pin.
RIn	Input	Low	Ring Indicator This active-low signal is an input showing when the attached device (modem) has sensed a ring signal on the telephone line. Core16550 passes this information to the CPU through the Modem Status register. This register also indicates when the RIn trailing edge was sensed.
SOUT	Output	—	Serial output data This data is transmitted from Core16550. It is synchronized with the BAUDOUT output pin.
RTSn	Output	Low	Request to Send This active-low output signal is used to inform the attached device (modem) that Core16550 is ready to send data. It is programmed by the CPU through the Modem Control register.

Table 4-1. I/O Signal Summary (continued)

Name	Type	Polarity	Description
DTRn	Output	Low	Data Terminal Ready This active-low output signal informs the attached device (modem) that Core16550 is ready to establish a communications link. It is programmed by the CPU through the Modem Control register.
OUT1n	Output	Low	Output 1 This active-low output is a user-defined signal. CPU programs this signal through the Modem Control register and is set to the opposite value.
OUT2n	Output	Low	Output 2 This active-low output signal is a user-defined signal. It is programmed by the CPU through the Modem Control register and is set to the opposite value. programmed.
INTR	Output	High	Interrupt Pending This active-high output signal is the interrupt output signal from Core16550. It is programmed to become active on certain events, informing the CPU that such an event has occurred, (for more details, see Interrupt Identification Register). The CPU then takes appropriate action.
BAUDOUTn	Output	Low	Baud out This is an output clock signal derived from the input clock for synchronizing the data output stream from SOUT.
RXRDYN	Output	Low	Receiver ready to receive transmissions. The CPU is indicated by this active-low output signal that the receiver section of Core16550 is available for data to be read.
TXRDYN	Output	Low	Transmitter ready to transmit data. This active-low signal indicates to the CPU that the transmitter section of Core16550 has space to write data for transmission.
rxfifo_empty	Output	High	Receive FIFO empty. This signal goes HIGH when the receive FIFO is empty.
rxfifo_full	Output	High	Receive FIFO full. This signal goes High when the receive FIFO is full.

5. Timing Diagrams (Ask a Question)

This section provides timing diagrams of this core.

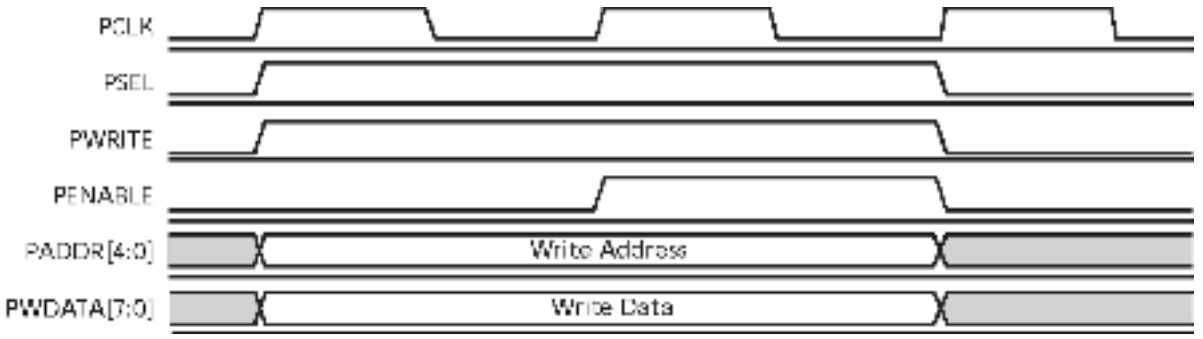
5.1. Data Write Cycle and Data Read Cycle (Ask a Question)

Figure 5-1 and Figure 5-2 depict write cycle and read cycle timing relationships relative to the APB system clock, PCLK.

5.1.1. Register Write (Ask a Question)

The following figure shows the Address, Select and Enable signals are latched and must be valid prior to the rising edge of PCLK. Writing occurs at the rising edge of the PCLK signal.

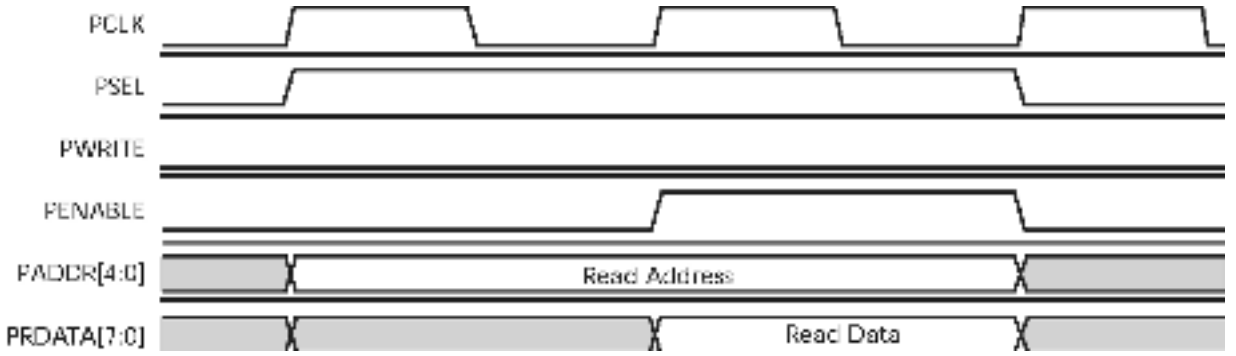
Figure 5-1. Data Write Cycle



5.1.2. Register Read (Ask a Question)

The following figure shows the Address, Select and Enable signals are latched and must be valid prior to the rising edge of PCLK. Reading occurs at the rising edge of the PCLK signal.

Figure 5-2. Data Read Cycle

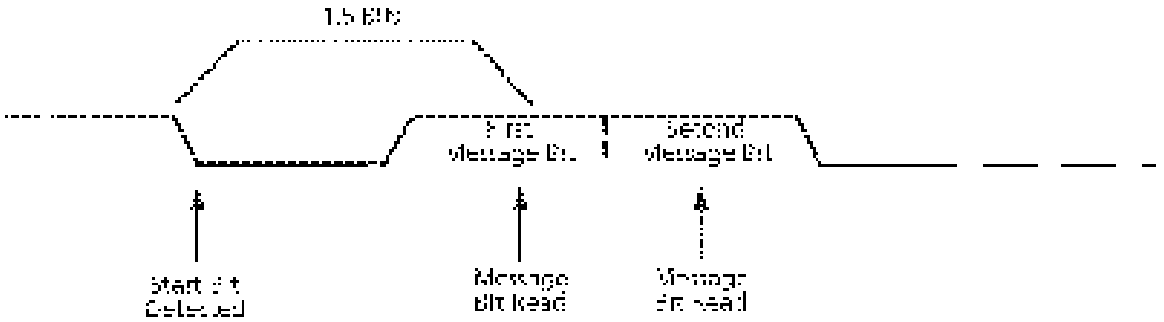


For more details on the descriptions and timing waveforms, see AMBA specification.

5.2. Receiver Synchronization (Ask a Question)

When the receiver detects a Low state in the incoming data stream, it synchronizes to it. After the start edge, the UART waits 1.5 × (the normal bit length). This causes each subsequent bit to be read at the middle of its width. The following figure depicts this synchronization process.

Figure 5-3. Receiver Synchronization



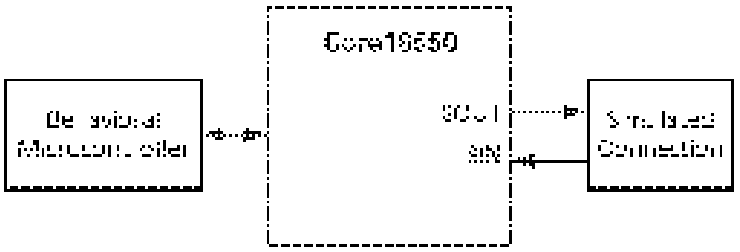
6. Testbench Operation (Ask a Question)

Only one testbench is provided with Core16550: Verilog user testbench. This is a simple-to-use testbench written in Verilog. This testbench is intended for customer modification.

6.1. User Testbench (Ask a Question)

The following figure shows the block diagram of the example user design and testbench.

Figure 6-1. Core16550 User Testbench



The user testbench includes a simple example design that serves as a reference for users who want to implement their own designs.

The testbench for example, user design implements a subset of the functionality tested in the verification testbench, for more details, see User Testbench. Conceptually, as shown in Figure 6-1, instantiation of Core16550 is simulated using a behavioral microcontroller and a simulated loopback connection. For example, user testbench demonstrates the transmit and receive by the same Core16550 unit, so you can gain a basic understanding of how to use this core.

The user testbench demonstrates the basic setup, transmit and receive operations of Core16550. The user testbench performs the following steps:

1. Write to the control registers.
2. Check received data.
3. Turn on transmit and receive.
4. Read the control registers.
5. Transmit and receive one byte.

7. Device Utilization and Performance (Ask a Question)

The following table lists the Core16550 utilization and performance data.

Table 7-1. Core16550 Utilization and Performance PolarFire and PolarFire SoC

Device Details		Resources			RAM
Family	Device	4LUT	DFF	Logic Elements	µSRAM
PolarFire®	MPF100T- FCSG325I	752	284	753	2
PolarFire® SoC	MPFS250TS- FCSG536I	716	284	720	2
RTG4™	RT4G150- 1CG1657M	871	351	874	2
IGLOO® 2	M2GL050TFB GA896STD	754	271	1021	2
SmartFusion® 2	M2S050TFBG A896STD	754	271	1021	2
SmartFusion®	A2F500M3G- STD	1163	243	1406	2
IGLOO®/IGLOOE	AGL600- STD/AGLE600 V2	1010	237	1247	2
Fusion	AFS600-STD	1010	237	1247	2
ProASIC® 3/E	A3P600-STD	1010	237	1247	2
ProASIC Plus®	APA075-STD	1209	233	1442	2
RTAX-S	RTAX250S- STD	608	229	837	2
Axcelerator®	AX125-STD	608	229	837	2

8. Resolved Issues (Ask a Question)

The following table lists all the resolved issues for the various Core16550 releases.

Table 8-1. Resolved Issues

Version	Changes
v3.4	• Core16550 uses System Verilog Keyword “break” as register name which was causing syntax error issue. This has been fixed by replacing the keyword with another name. • Added PolarFire® family support

9. Revision History (Ask a Question)

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

Revision	Date	Description
A	12/2024	The following is the list of changes in revision A of the document: • This document is migrated to Microchip standards. • The document number is changed to "DS50003798" from "UG0110". • Added PolarFire®, PolarFire Soc, and RT PolarFire to the Introduction and Device Utilization and Performance.
4.0	02/2015	Added “IGLOO2” and “RTG4” to the Introduction and to Table 2 in the Device Utilization and Performance.
3.0	12/2012	Added "SmartFusion2" to the Introduction and to Table 2 in the Device Utilization and Performance (SAR 43254).
2.0	04/2012	• Added "SmartFusion" to the Introduction and to Table 2 in the Device Utilization and Performance (SAR 36238). • Added "AGLE600V2" to Table 2 in the Device Utilization and Performance, as also modified the "Cell or Tiles" values for SmartFusion (SAR 11944). • Updated Default (reset) Value for IIR and FCR in Table 1-1, Core16550 Register Summary (SAR 37805). • Updated FCR[3] definition in Table 1-4 (SAR 22657). • Updated Default State for 7..6 Bit definition in Table 1-8 (SAR 36238). • Replaced Figure 2-1 with a new image (SAR 36238). • Deleted the "Evaluation" section and modified the Obfuscated section (SAR 36238). • Added references to Libero SoC, and the SmartDesign and Simulation Flows sections (SAR 36238). • Removed all references to CoreConsole and Libero IDE, deleted the "CoreConsole" and "Importing into Libero IDE" sections (SAR 36238). • Renamed the Place-and-Route in Libero IDE section to Place-and-Route in Libero SoC (SAR 36238). • Added "SmartFusion" Core Parameter (SAR 36238). • Deleted the "Verification Testbench" section (SAR 37805). Evaluation license/RTL no longer relevant as Obfuscated licenses are available free.
1.0	01/2008	The Introduction was added in v2.1.

Microchip FPGA Support

Microchip FPGA products group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, and worldwide sales offices. Customers are suggested to visit Microchip online resources prior to contacting support as it is very likely that their queries have been already answered.

Contact Technical Support Center through the website at www.microchip.com/support. Mention the FPGA Device Part number, select appropriate case category, and upload design files while creating a technical support case.

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

- From North America, call **800.262.1060**
- From the rest of the world, call **650.318.4460**
- Fax, from anywhere in the world, **650.318.8044**

Microchip Information

Trademarks

The “Microchip” name and logo, the “M” logo, and other names, logos, and brands are registered and unregistered trademarks of Microchip Technology Incorporated or its affiliates and/or subsidiaries in the United States and/or other countries (“Microchip Trademarks”). Information regarding Microchip Trademarks can be found at https://www.microchip.com/en-us/about/legal-information/microchip-trademarks.

ISBN:

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip products are strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.