# OPB 16550 UART (v1.00d)

**DS430 December 2, 2005** **Product Specification**

## Introduction

This document provides the specification for the OPB 16550 UART (Universal Asynchronous Receiver/Transmitter) Intellectual Property (IP).

The OPB 16550 UART described in this document has been designed incorporating features described in *National Semiconductor PC16550D UART with FIFOs data sheet* (June,1995), (http://www.national.com/pf/PC/PC16550D.html).

The National Semiconductor PC16550D data sheet is referenced throughout this document and should be used as the authoritative specification. Differences between the National Semiconductor implementation and the OPB 16550 UART Point Design implementation are highlighted and explained in Specification Exceptions.

## Features

- Hardware and software register compatible with all standard 16450 and 16550 UARTs
- Implements all standard serial interface protocols
  - 5, 6, 7, or 8 bits per character
  - Odd, Even, or no parity detection and generation
  - 1, 1.5, or 2 stop bit detection and generation
  - Internal baud rate generator and separate receiver clock input
  - Modem control functions
  - False start bit detection and recovery
  - Prioritized transmit, receive, line status, and modem control interrupts
  - Line break detection and generation
  - Internal loop back diagnostic functionality
  - Independent 16 word transmit and receive FIFOs

| LogiCORE™ Facts | | |
|---|---|---|
| **Core Specifics** | | |
| Supported Device Family | QPro™-R Virtex™-II, QPro Virtex-II, Spartan™-II, Spartan-IIE, Spartan-3, Spartan-3E, Virtex, Virtex-II, Virtex-II Pro, Virtex-4, Virtex-E | |
| Version of Core | opb_uart16550 | v1.00d |
| **Resources Used** | | |
| | Min | Max |
| Slices | 283 | 417 |
| LUTs | 328 | 545 |
| FFs | 275 | 338 |
| Block RAMs | | |
| **Provided with Core** | | |
| Documentation | Product Specification | |
| Design File Formats | VHDL | |
| Constraints File | N/A | |
| Verification | N/A | |
| Instantiation Template | N/A | |
| Reference Designs | None | |
| **Design Tool Requirements** | | |
| Xilinx Implementation Tools | ISE 6.1i or later | |
| Verification | ModelSim SE/EE 5.8e or later | |
| Simulation | ModelSim SE/EE 5.8e or later | |
| Synthesis | XST | |
| **Support** | | |
| Support provided by Xilinx, Inc. | | |

## Functional Description

The OPB 16550 UART implements the hardware and software functionality of the ubiquitous National Semiconductor 16550 UART. For complete details please refer to the National Semiconductor data sheet (http://www.national.com/pf/PC/PC16550D.html).

The OPB 16550 performs parallel to serial conversion on characters received from the CPU and serial to parallel conversion on characters received from a modem or microprocessor peripheral.

The OPB 16550 is capable of transmitting and receiving 8, 7, 6, or 5 bit characters, with 2, 1.5 or 1 stop bits and odd, even or no parity. The OPB 16550 can transmit and receive independently.

The device can be configured and it's status monitored via the internal register set. The OPB 16550 is capable of signaling receiver, transmitter and modem control interrupts. These interrupts can be masked, are prioritized and can be identified by reading an internal register.

The device contains a 16 bit, programmable, baud rate generator and independent 16 word transmit and receive FIFOs. The FIFOs can be enabled or disabled through software control.

The top-level block diagram for the OPB 16550 UART is shown in Figure 1

The OPB 16550 UART consists of the following top level modules

- OPB_IPIF
- IPIC_IF
- UART16550

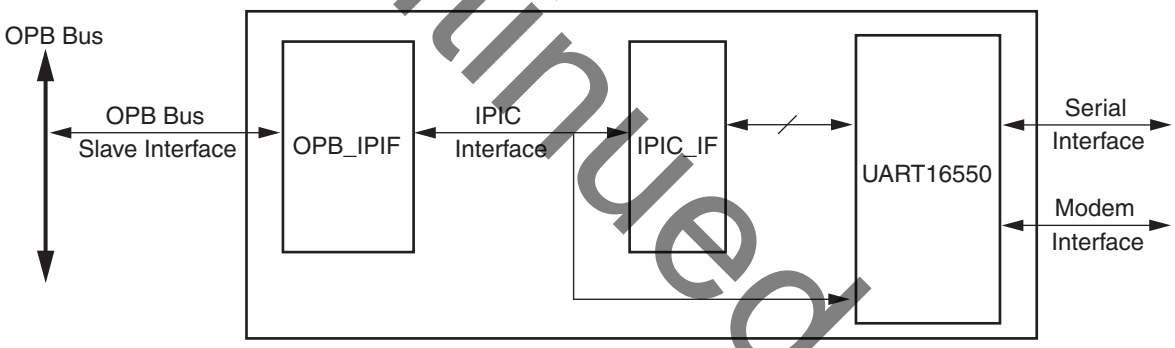


*Figure 1:* **OPB 16550 UART Top-level Block Diagram**

The detailed block diagram for the OPB 16550 UART is shown in Figure 2.

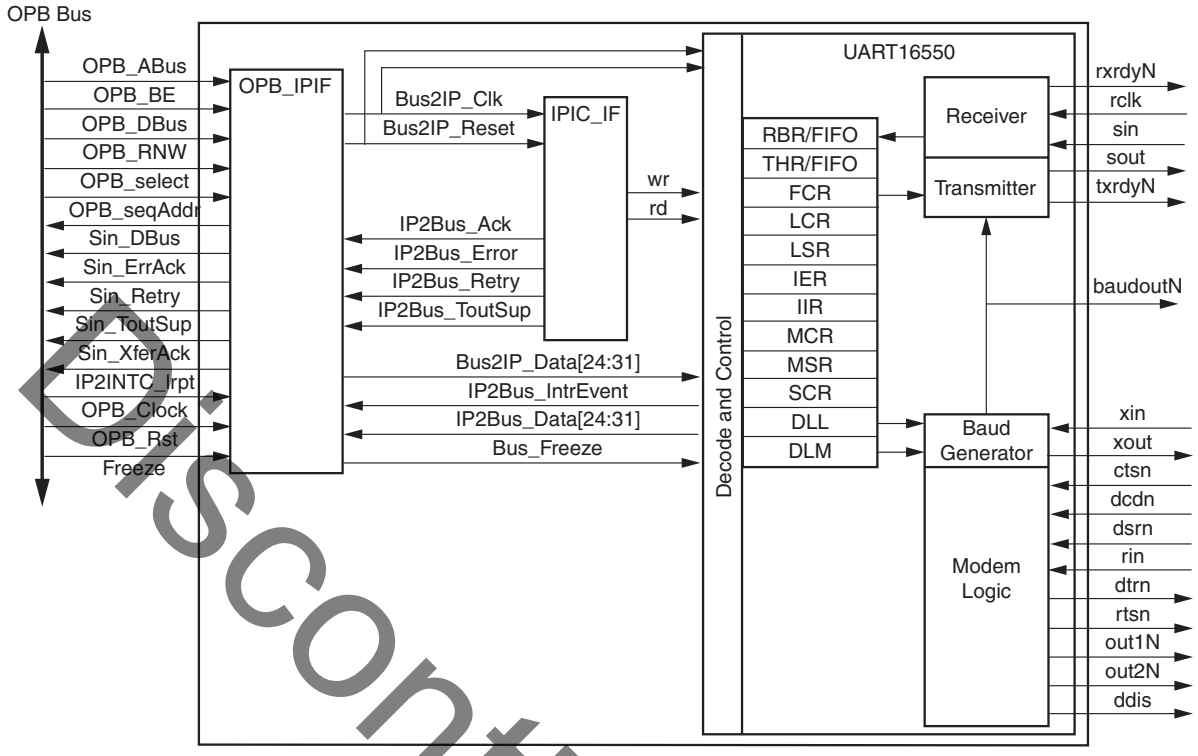


*Figure 2:* **OPB 16550 UART Detailed Block Diagram**



## OPB 16550 UART Design Parameters

To allow the user to obtain an OPB 16550 UART that is uniquely tailored system, certain features can be parameterized in the OPB 16550 UART design. This allows a design which only utilizes the resources required by the system and runs at the best possible performance. The features that can be parameterized in the OPB 16550 UART design are shown in Table 1.

*Table 1:* **Design Parameters**

| Generic | Feature / Description | Parameter Name | Allowable Values | Default Value | VHDL Type |
|---|---|---|---|---|---|
| **OPB Interface** | | | | | |
| G1 | OPB UART Base Address | C_BASEADDR | Valid Word Aligned Address(1) | None(2) | std_logic_vector |
| G2 | OPB Data Bus Width | C_OPB_DWIDTH | 32 | 32 | integer |
| G3 | OPB Address Bus Width | C_OPB_AWIDTH | 32 | 32 | integer |
| G4 | Device Block ID | C_DEV_BLK_ID | 0-255 | 0 | integer |
| G5 | Module Identification Register | C_DEV_MIR_ENABLE | 0,1 | 0 | integer |

*Table 1:* **Design Parameters *(Contd)***

| Generic | Feature / Description | Parameter Name | Allowable Values | Default Value | VHDL Type |
|---|---|---|---|---|---|
| G6 | OPB UART High Address | C_HIGHADDR | C_HIGHADDR -C_BASEADDR must be a power of 2 >= to C_BASEADDR+1FFF(1) | None(2) | std_logic_ vector |
| **UART Features** | | | | | |
| G7 | External XIN | C_HAS_EXTERNAL_ XIN | 0,1 | 0 | integer |
| G8 | External RCLK | C_HAS_EXTERNAL_ RCLK | 0,1 | 0 | integer |
| G9 | Select UART | C_IS_A_16550 | 0,1 | 1 | integer |

**Notes:**
1. Address range specified by C_BASEADDR and C_HIGHADDR must be at least 0x2000 and must e a power of 2.
2. No default value will be specified to insure that the actual value is set, i.e. if the value is not set, a compiler error will be generated.

## Allowable Parameter Combinations

The minimum address range specified by C_BASEADDR and C_HIGHADDR must be at least 0x0000_2000.

# OPB 16550 UART I/O Signals

The I/O signals for the OPB 16550 UART are listed in Table 2. The interfaces referenced in this table are shown in Figure 2 in the OPB 16550 UART block diagram.

*Table 2:* **OPB 16550 UART I/O Signals**

| Port | Signal Name | Interface | I/O | Initial State | Description |
|---|---|---|---|---|---|
| **OPB Slave Signals** | | | | | |
| P1 | OPB_ABus(0:C_OPB_AWIDTH-1) | OPB | I | | OPB Address Bus |
| P2 | OPB_BE(0:(C_OPB_AWIDTH/8)-1 ) | OPB | I | | OPB Byte Enable |
| P3 | OPB_DBus(0:C_OPB_DWIDTH-1) | OPB | I | | OPB Data Bus |
| P4 | OPB_RNW | OPB | I | | Read Not Write |
| P5 | OPB_Select | OPB | I | | OPB select |
| P6 | OPB_seqAddr | OPB | I | | OPB sequential address (unused) |
| P7 | Sln_DBus(0:C_OPB_DWIDTH-1) | OPB | O | 0 | Output Data Bus |
| P8 | Sln_ErrAck | OPB | O | 0 | Slave Error Acknowledge (always inactive) |
| P9 | Sln_Retry | OPB | O | 0 | Slave Bus Cycle Retry (always inactive) |
| P10 | Sln_ToutSup | OPB | O | 0 | Slave Time Out Suppress (always inactive) |

*Table 2:* **OPB 16550 UART I/O Signals *(Contd)***

| Port | Signal Name | Interface | I/O | Initial State | Description |
|---|---|---|---|---|---|
| P11 | Sln_XferAck | OPB | O | 0 | Slave Transfer Acknowledge |
| P12 | IP2INTC_Irpt | System | O | 0 | Device interrupt output to microprocessor interrupt input or system interrupt controller. Registered level type, asserted active high. |
| **UART Signals** | | | | | |
| P13 | baudoutN | Serial | O | 1 | Transmitter Clock |
| P14 | rclk | Serial | I | | Receiver 16x Clock (Optional. May be driven by baudoutN under control of the C_HAS_EXTERNAL_RCLK parameter) |
| P15 | sin | Serial | I | | Serial Data Input |
| P16 | sout | Serial | O | 1 | Serial Data Output |
| P17 | xin | Serial | I | | Baud Rate Generator reference clock. (Optional. May be driven by OPB_Clk under control of the C_HAS_EXTERNAL_XIN parameter) |
| P18 | xout | Serial | O | ~XIN | Inverted XIN |
| P19 | ctsN | Modem | I | | ClearToSend (active low) |
| P20 | dcdN | Modem | I | | Data Carrier Detect (active low) |
| P21 | dsrN | Modem | I | | Data Set Ready (active low) |
| P22 | dtrN | Modem | O | 1 | Data Terminal Ready (active low) |
| P23 | riN | Modem | I | | Ring Indicator (active low) |
| P24 | rtsN | Modem | O | 1 | Request To Send (active low) |
| P25 | ddis | User | O | 1 | Driver Disable. Low when CPU is reading OPB UART |
| P26 | out1N | User | O | 1 | User controlled output |
| P27 | our2N | User | O | 1 | User controlled output |
| P28 | rxrdyN | User | O | 1 | DMA control signal |
| P29 | txrdyN | User | O | 1 | DMA control signal |
| **System** | | | | | |
| P30 | OPB_Clk | System | I | | System clock |
| P31 | OPB_Rst | System | I | | System Reset (active high) |
| P32 | Freeze | System | I | | Freezes UART for software debug (active high) |

## Parameter - Port Dependencies

The width of many of the OPB 16550 UART signals depends on parameter. In addition, when certain features are parameterized away, the related input signals are unconnected. The dependencies between the OPB 16550 UART design parameters and I/O signals are shown in Table 3. parameters and I/O signals are shown in the following table.

*Table 3:* **Parameter-Port Dependencies**

| Generic or Port | Name | Affects | Depends | Relationship Description |
|---|---|---|---|---|
| **Design Parameters** | | | | |
| G1 | C_BASEADDR | | G3 | Bus width affects maximum allowable address. |
| G2 | C_OPB_DWIDTH | P3, P7 | | Affects number of bits in bus. |
| G3 | C_OPB_AWIDTH | P1, P2 | | Affects number of bits in bus. |
| G6 | C_HIGHADDR | | G3 | Bus width affects maximum allowable address. |
| G7 | C_HAS_EXTERNAL_XIN | P17 | | Connects XIN to OPB_CLK |
| G8 | C_HAS_EXTERNAL_RCLK | P14 | | Connects RCLK to baudoutN |
| **I/O Signals** | | | | |
| P1 | OPB_ABus(0:C_OPB_AWIDTH-1) | | G3 | Width varies with the size of the OPB Address bus. |
| P2 | OPB_BE(0:(C_OPB_AWIDTH/8)-1) | | G3 | Width varies with the size of the OPB Address bus. |
| P3 | OPB_DBus(0:C_OPB_DWIDTH-1) | | G2 | Width varies with the size of the OPB Data bus. |
| P7 | Sln_DBus(0:C_OPB_DWIDTH-1) | | G2 | Width varies with the size of the OPB Data bus. |
| P14 | rclk | | G8, P13 | This input is unconnected and UART receiver clock is connected to BAUDOUTn if C_HAS_EXTERNAL_RCLK=0. |
| P17 | xin | | G7, P30 | This input is unconnected and UART reference clock is connected to OPB_Clk if C_HAS_EXTERNAL_XIN=0. |



## OPB 16550 UART Register Definition

### OPB 16550 UART Interface

The OPB memory map location of the OPB 16550 UART is determined by setting the parameter C_BASEADDR in the IPIF interface module.

The internal registers of the OPB 16550 UART are offset from the C_BASEADDR base address. Additionally, some of the internal registers are accessible only when bit 7 of the Line Control Register (LCR) is set. The OPB 16550 UART internal register set is described in Table 4.

*Table 4:* **OPB 16550 UART Registers**

| Register Name | LCR(7)[1]+ C_BASEADDR + Address | Access |
|---|---|---|
| Receiver Buffer Register (RBR) | 0 + C_BASEADDR + 0x1000 | Read |
| Transmitter Holding Register (THR) | 0 + C_BASEADDR + 0x1000 | Write |
| Interrupt Enable Register (IER) | 0 + C_BASEADDR + 0x1004 | Read/Write |
| Interrupt Identification Register (IIR) | 0 + C_BASEADDR + 0x1008 | Read |
| FIFO Control Register (FCR) | 0 + C_BASEADDR + 0x1008 | Write |
| FIFO Control Register[2] | 1 + C_BASEADDR + 0x1008 | Read |
| Line Control Register (LCR) | X + C_BASEADDR + 0x100C | Read/Write |
| Modem Control Register (MCR) | X + C_BASEADDR + 0x1010 | Read/Write |
| Line Status Register (LSR) | X + C_BASEADDR + 0x1014 | Read/Write |
| Modem Status Register (MSR) | X + C_BASEADDR + 0x1018 | Read/Write |
| Scratch Register (SCR) | X + C_BASEADDR + 0x101C | Read/Write |
| Divisor Latch Register (DLL) | 1 + C_BASEADDR + 0x1000 | Read/Write |
| Divisor Latch Register (DLM) | 1 + C_BASEADDR + 0x1004 | Read/Write |

**Notes:**
1. X denotes a don't care
2. FIFO Control Register is Write Only in the National PC16550D



## OPB 16550 UART Register Logic

This section tabulates the internal OPB 16550 UART registers, including their reset values (if any).

Please refer to the National Semiconductor PC16550D UART with FIFOs data sheet (June, 1995),

(http://www.national.com/pf/PC/PC16550D.html) for a more detailed description of the register behavior.

### Receiver Buffer Register

This is an 8 bit read register as shown in Figure 3. The Receiver Buffer Register contains the last received character. The bit definitions for the register are shown in Table 5. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

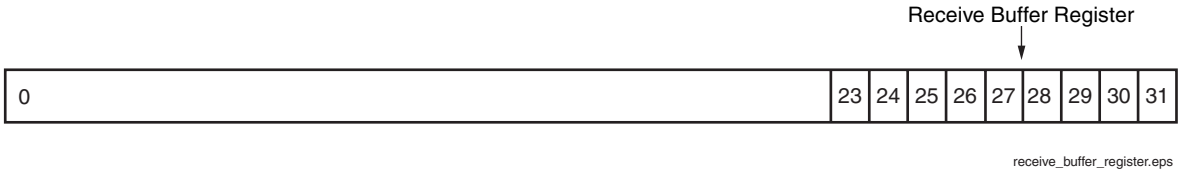


*Figure 3:* **Recever Buffer Register (RBR)**

*Table 5:* **Receiver Buffer Register Bit Definitions**

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 0-23 | N/A | N/A | N/A | Unused. Set to zeroes on read |
| 24-31 | RBR | Read | “00000000" | RBR. Last received character |

### Transmitter Holding Register

This is an 8 bit write register as shown in Figure 4. The Transmitter Holding Register contains the character to be transmitted next. The bit definitions for the register are shown in Table 6. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

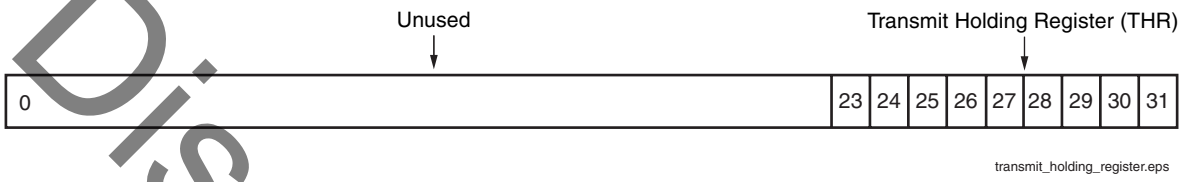


*Figure 4:* **Transmit Holding Register (THR)**

*Table 6:* **Transmitter Holding Register Bit Definitions**

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 0-23 | N/A | N/A | N/A | N/A |
| 24-31 | THR | Write | “11111111” | THR. Holds the character to be transmitted next |

### Interrupt Enable Register

This is an 8 bit read/write register as shown in Figure 5. The Interrupt Enable Register contains the bits which enable interrupts. The bit definitions for the register are shown in Table 7. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

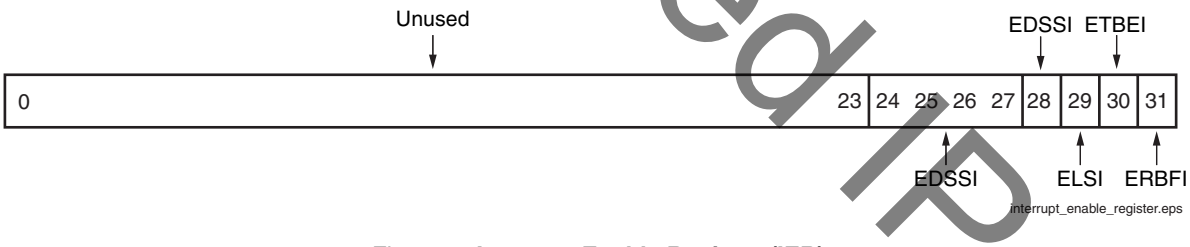


*Figure 5:* **Interrupt Enable Register (IER)**

*Table 7:* **Interrupt Enable Register Bit Definitions(1)**

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 0-23 | N/A | N/A | N/A | N/A |
| 24-27 | | Read/Write | “0000" | |
| 28 | EDSSI | Read/Write | "0" | Enable Modem Status Interrupt.<br>"0" -> Disables Modem Status Interrupts.<br>"1" -> Enables Modem Status Interrupts. |

*Table 7:* **Interrupt Enable Register Bit Definitions(1) *(Contd)***

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 29 | ELSI | Read/Write | "0" | Enable Receiver Line Status Interrupt.<br>"0" -> Disables Receiver Line Status Interrupts.<br>"1" -> Enables Receiver Line Status Interrupts. |
| 30 | ETBEI | Read/Write | "0" | Enable Transmitter Holding Register Empty Interrupt.<br>"0" -> Disables Transmitter Holding Register Empty Interrupts.<br>"1" -> Enables Transmitter Holding Register Interrupts. |
| 31 | ERBFI | Read/Write | "0" | Enable Received Data Available Interrupt.<br>"0" -> Disables Received Data Available Interrupts.<br>"1" -> Enables Received Data Available Interrupts. |

**Notes:**
1. Bold faced bits are permanently low. Writing to these bits is allowed. Reading always returns "0".

## Interrupt Identification Register

This is an 8 bit read register as shown in Figure 6. The Interrupt Identification Register contains the priority interrupt identification. The bit definitions for the register are shown in Table 8. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

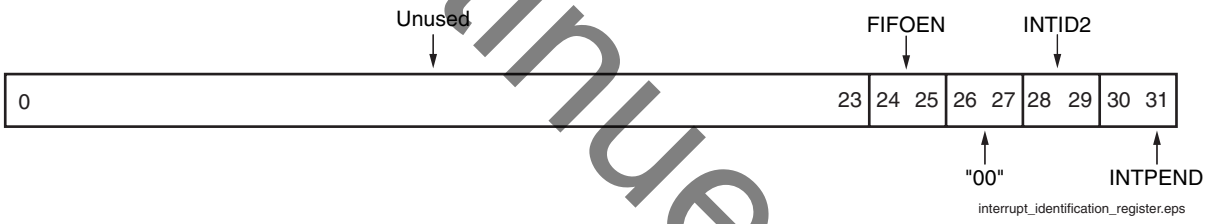


*Figure 6:* **Interrupt Identification Register (IIR)**

*Table 8:* **Interrupt Identification Register Bit Definitions (1)**

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 0-23 | N/A | N/A | N/A | N/A |
| 24-25 | FIFOEN | Read | “00" | FIFOs Enabled. Always zero if not in FIFO mode. |
| 26-27 | | Read | “**00**" | Always returns "00" |

*Table 8:* **Interrupt Identification Register Bit Definitions (1) *(Contd)***

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 28-30 | INTID2 | Read | “000" | Interrupt ID.(2)<br>"011" -> Receiver Line Status (Highest).<br>"010" -> Received Data Available (Second).<br>"110" -> Character Timeout (Second).<br>"001" -> Transmitter Holding Register Empty (Third).<br>"000" -> Modem Status (Fourth). |
| 31 | INTPEND(2) | Read | “1" | Interrupt Pending. Interrupt is pending when cleared. |

**Notes:**
1. Bold faced bits are permanently low. Reading these bits always return "0"
2. If bit 0 is cleared. See National Semiconductor PC16550D data sheet for more detail.

## FIFO Control Register

This is an 8 bit write/read register as shown in Figure 7. The FIFO Control Register contains the FIFO configuration bits. The bit definitions for the register are shown in Table 9. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

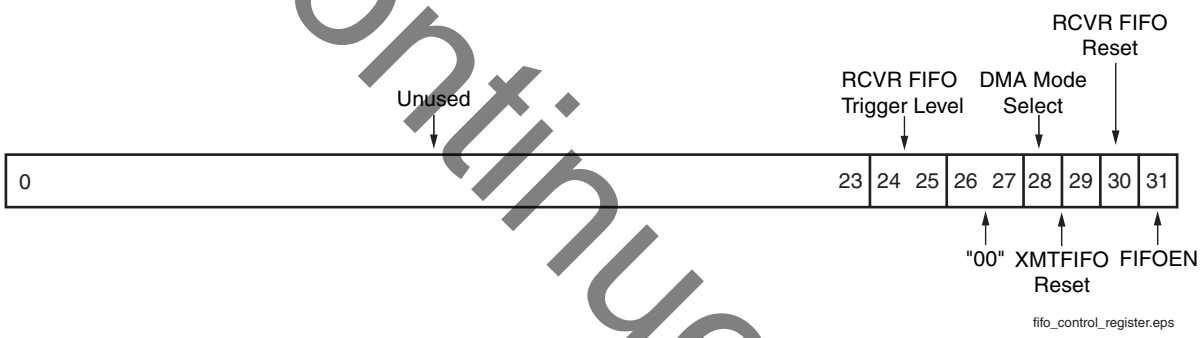


*Figure 7:* **FIFO Control Register (FCR)**

*Table 9:* **FIFO Control Register Bit Definitions (1)**

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 0-23 | N/A | N/A | N/A | N/A |
| 24-25 | RCVR FIFO Trigger Level | Read/Write | “00” | RCVR FIFO Trigger Level.<br>"00" -> 1 byte.<br>"01" -> 4 bytes.<br>"10" -> 8 bytes.<br>"11" -> 14 bytes. |
| 26-27 | Reserved | Read/Write | “**00**” | Always returns "00" |
| 28 | DMA Mode Select | Read/Write | “0” | DMA Mode Select.<br>"0" -> Mode 0.<br>"1" -> Mode 1. |
| 29 | XMIT FIFO Reset | Read/Write | “0” | Transmitter FIFO Reset.<br>"1" -> Resets XMIT FIFO. |

*Table 9:* **FIFO Control Register Bit Definitions** (1) *(Contd)*

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 30 | RCVR FIFO Reset | Read/Write | “0” | Receiver FIFO Reset.<br>"1" -> Resets RCVR FIFO. |
| 31 | FIFOEN | Read/Write | “0” | FIFO Enable.<br>"1" -> Enables FIFOs. |

**Notes:**
1. Bold faced bits are permanently low. Reading these bits always return "0"

## Line Control Register

This is an 8 bit write/read register as shown in Figure 8. The Line Control Register contains the serial communication configuration bits. The bit definitions for the register are shown in Table 10. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

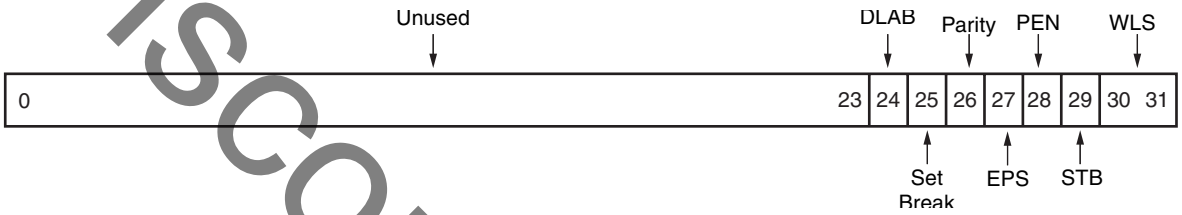


*Figure 8:* **Line Control Register (LCR)**

*Table 10:* **Line Control Register Bit Definitions**

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 0-23 | N/A | N/A | N/A | N/A |
| 24 | DLAB | Read/Write | "0" | Divisor Latch Access Bit. "1" - > Allows access to the Divisor Latch Registers and reading of the FIFO Control Register. |
| 25 | Set Break | Read/Write | "0" | Set Break.<br>"1" -> Sets SOUT to ’0". |
| 26 | Stick Parity | Read/Write | "0" | Stick Parity.<br>"1" -> Sets SOUT to ’0". |
| 27 | EPS | Read/Write | "0" | Even Parity Select.<br>"1" -> Selects Even parity.<br>"0" -> Selects Odd parity. |

*Table 10:* **Line Control Register Bit Definitions**

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 28 | PEN | Read/Write | "0" | Parity Enable<br>"1" -> Enables parity. |
| 29 | STB | Read/Write | "0" | Number of Stop Bits.<br>"1" -> 1 Stop bit.<br>"0" -> 2 Stop bits or 1.5 if 5 bits/character selected). |
| 30-31 | WLS | Read/Write | "00" | Word Length Select.<br>"00" -> 5 bits/character.<br>"01" -> 6 bits/character.<br>"10" -> 7bits/character.<br>"11" -> 8bits/character. |

## Modem Control Register

This is an 8 bit write/read register as shown in Figure 9. The Modem Control Register contains the modem signalling configuration bits. The bit definitions for the register are shown in Table 11. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.


*Figure 9:* **Modem Control Register (MCR)**

*Table 11:* **Modem Control Register Bit Definitions** (1)

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 0-23 | N/A | N/A | N/A | N/A |
| 24-26 | | Read/Write | "**000**" | |
| 27 | Loop | Read/Write | "0" | Loop Back.<br>"1" -> Enables loop back. |
| 28 | Out2 | Read/Write | "0" | User Output 2.<br>"1" -> Drives OUT2N low.<br>"0" -> Drives OUT2N high. |
| 29 | Out1 | Read/Write | "0" | User Output 1.<br>"1" -> Drives OUT1N low.<br>"0" -> Drives OUT1N high. |

*Table 11:* **Modem Control Register Bit Definitions (1)**

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 30 | RTS | Read/Write | "0" | Request To Send.<br>"1" -> Drives RTSN low.<br>"0" -> Drives RTSN high. |
| 31 | DTR | Read/Write | "0" | Data Terminal Ready.<br>"1" -> Drives DTRN low.<br>"0" -> Drives DTRN high. |

**Notes:**
1. Bold faced bits permanently low

## Line Status Register

This is an 8 bit write/read register as shown in Figure 10. The Line Status Register contains the current status of receiver and transmitter. The bit definitions for the register are shown in Table 12. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

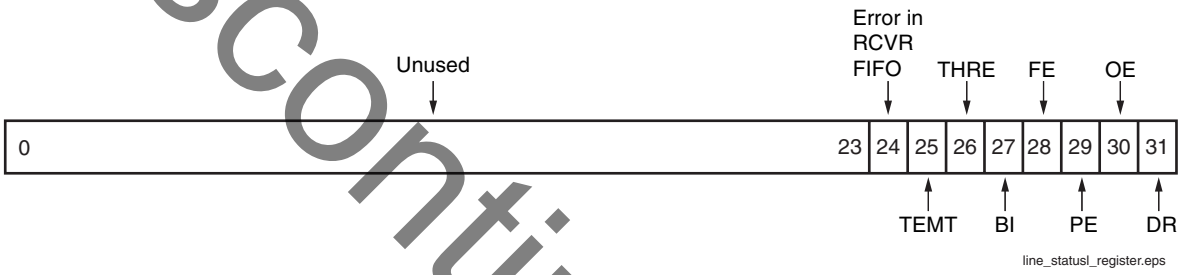


*Figure 10:* **Line Status Register (LSR)**

*Table 12:* **Line Status Register Bit Definitions**

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 0-23 | N/A | N/A | N/A | N/A |
| 24 | Error in RCVR FIFO | Read/Write | “0” | Error in RCVR FIFO.<br>RCVR FIFO contains at least one receiver error. |
| 25 | TEMT | Read/Write | “1” | Transmitter Empty. |
| 26 | THRE | Read/Write | “1” | Transmitter Holding Register Empty. |
| 27 | BI | Read/Write | “0” | Break Interrupt.<br>Set when SIN is held low for an entire character time. |
| 28 | FE | Read/Write | “0” | Framing Error.<br>Character missing a stop bit. Receiver resynchs with next character, if possible. |
| 29 | PE | Read/Write | “0” | Parity Error. |
| 30 | OE | Read/Write | “0” | Overrun Error.<br>RBR not read before next character is received. |
| 31 | DR | Read/Write | “0” | Data Ready. |

### Modem Status Register

This is an 8 bit write/read register as shown in Figure 11. The Modem Status Register contains the current state of the Modem Interface. The bit definitions for the register are shown in Table 13. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

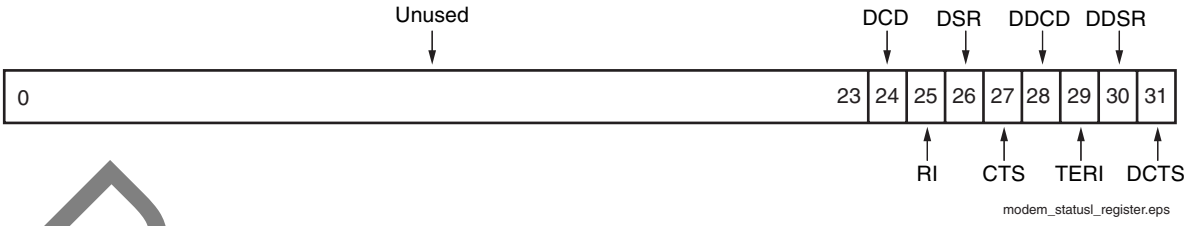


*Figure 11:* **Modem Status Register (MSR)**

*Table 13:* **Modem Status Register Bit Definitions (1)**

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 0-23 | N/A | N/A | N/A | N/A |
| 24 | DCD | Read/Write | “X” | Data Carrier Detect.<br>Complement of DCDN input. |
| 25 | RI | Read/Write | "X" | Ring Indicator.<br>Complement of RIN input. |
| 26 | DSR | Read/Write | "X" | Data Set Ready.<br>Complement of DSRN input. |
| 27 | CTS | Read/Write | "X" | Clear To Send.<br>Complement of CTSN input. |
| 28 | DDCD | Read/Write | "0" | Delta Data Carrier Detect.<br>Change in DCDN since last MSR read. |
| 29 | TERI | Read/Write | "0" | Trailing Edge Ring Indicator.<br>RIN has changed from a low to a high. |
| 30 | DDSR | Read/Write | "0" | Delta Data Set Ready.<br>Change in DSRN since last MSR read. |
| 31 | DCTS | Read/Write | "0" | Delta Clear To Send.<br>Change in CTSN since last MSR read. |

**Notes:**
1. X represents bit driven by external input.

### Scratch Register

This is an 8 bit write/read register as shown in Figure 11. The Scratch Register can be used to hold user data. The bit definitions for the register are shown in Table 14. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

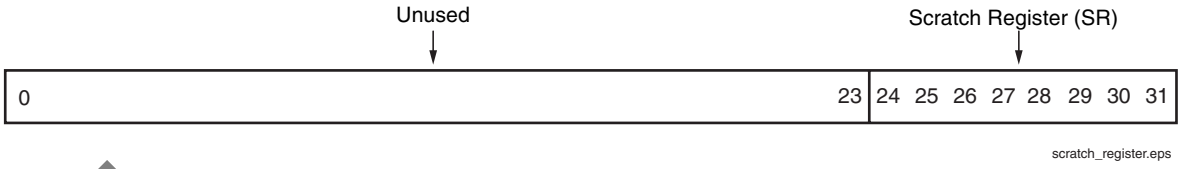


*Figure 12:* **Scratch Register (SCR)**

*Table 14:* **Scratch Register Bit Definitions**

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 0-23 | N/A | N/A | N/A | N/A |
| 24-31 | Scratch | Read/Write | “00000000” | Scratch. |

### Divisor Latch (Least Significant Byte) Register

This is an 8 bit write/read register as shown in Figure 13. The Divisor Latch (Least Significant Byte) Register holds the least significant byte of the baud rate generator counter. The bit definitions for the register are shown in Table 15. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

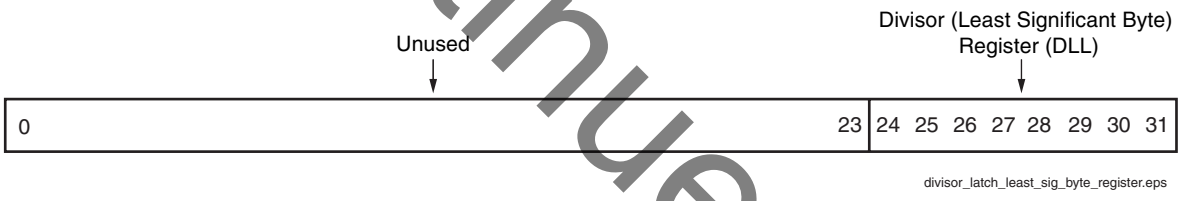


*Figure 13:* **Divisor Latch (Least Significant Byte) Register**

### Divisor Latch (Most Significant Byte) Register

*Table 15:* **Divisor Latch (Least Significant Byte) Register Bit Definitions**

| Bit Location | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 0-23 | N/A | N/A | N/A | N/A |
| 24-31 | DLL | Read/Write | “00000000” | Divisor Latch Least Significant Byte. |

This is an 8 bit write/read register as shown in Figure 14. The Divisor Latch (Most Significant Byte) Register holds the most significant byte of the baud rate generator counter. The bit definitions for the register are shown in Table 16. The offset and accessibility of this register from C_BASEADDR value is as shown in Table 4.

.

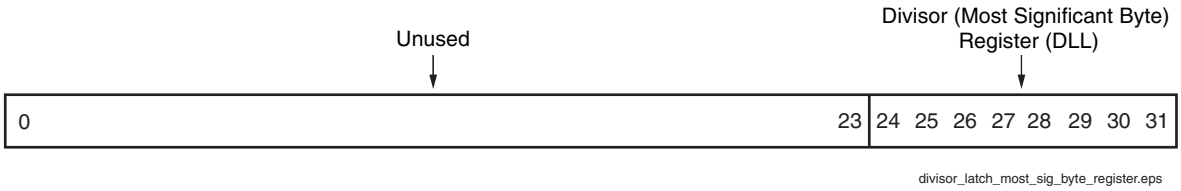


*Figure 14:* **Divisor Latch (Most Significant Byte) Register**

*Table 16:* **Divisor (Most Significant Byte) Register Bit Definitions**

| Bit | Name | Access | Reset Value | Description |
|---|---|---|---|---|
| 0-23 | N/A | N/A | N/A | N/A |
| 24-31 | DLM | Read/Write | “00000000” | Divisor Latch Most Significant Byte. |

## OPB_IPIF

OPB_IPIF provides bidirectional interface between UART16550 IP core and the OPB 32-bit bus standard.The base element of the OPB_IPIF is slave attachment, which provides the basic functionality of OPB slave operation.The OPB_IPIF is configured to include the following services.

The services are

- OPB slave attachment
- Module Identification Register

## IPIC_IF

IPIC_IF module incorporates logic to acknowledge the write and read transactions initiated by the OPB_IPIF module to write into the UART16550 module registers, and read from UART16550 module registers.

## UART16550

UART16550 provides all the core features for transmission, reception of data and modem features of UART.The UART16550 module of opb 16550 UART can be configured for 16450 or 16550 mode of operation. This is accomplished by the usage of generic C_IS_A_16550. If C_IS_A_16550 set to TRUE, the uart16550 module has FIFOs instantiated to support 16550 mode of operation. When C_IS_A_16550 is set to FALSE, the UART16550 module works without a FIFOs in its transmitting and receiving path.

## User Application Hints

The use of the OPB 16550 UART is outlined in the steps below.

1. Issue a software reset to the device by writing the data word 0x0000_000A to the MIR/Reset register address.This causes the reset signal to the IPIF and the OPB 16550 UART core.
2. The system programmer specifies the format of the asynchronous data communications exchange i.e Data bits (5,6,7 or 8), setting of parity ON and selecting on the even or odd parity, setting of the number stop bits for the transmission and set the Divisor latch access bit by programming the Line Control Register.
3. Write Interrupt Enable Register to activate the individual interrupts
4. Write to the FIFO Control Register to enable the FIFO's, clear the FIFO's, set the RCVR FIFO trigger level.
5. Write to Divisor Latch least significant byte first and Divisor Latch most significant byte second for proper setting of the baud rate of the UART.
6. Service the interrupts when ever an interrupt is triggered by the OPB 16550 UART.

An example use of the OPB 16550 UART with the operating mode set to the following parameters

- baud rate: 56Kbps
- Enabled and Threshold settings for the FIFO receive buffer
- Format of asynchronous data exchange 8 data bits, Even parity and 2 stop bits

1. Write 0x0000_0080 to Line Control Register.This configures DLAB bit which allows the writing into the Divisor Latch's Least significant and Most significant bytes.
2. Write 0x0000_0002 to Divisor Latch's Least significant byte and write 0x0000_0000 to Divisor Latch's Most significant byte in that order. This configures the baud rate setup of UART to 56Kbps operation.
3. Write 0x0000_001F to Line Control Register. This configures word length to 8 bits, Number of stop bits to 2, Parity is enabled and set to Even parity and DLAB bit is set to value 0 to enable the use of Transmit Holding register and Receive buffer register data for transmitting and reception of data.
4. Write 0x0000_0011 to Interrupt Enable Register. This enables the Transmitter holding register empty interrupt and Receive data available interrupt.
5. Write the buffer to Transmit Holding register and read the data received from Receive Holding register by servicing the interrupts generated.

## Design Implementation

### Target Technology

The intended target technology is Virtex-II Family.

### Device Utilization and Timing

OPB_Clk is capable of running at 100 MHz. XIN and RCLK must be less than 1/2 OPB_Clk frequency.

## Performance Benchmarks

*Table 17:* **Performance and Resource Utilization for Benchmarks OPB 16550 UART Benchmarks (Virtex-II Pro -6)**

| Parameter Values | | | | Device Resources | | | $f_{MAX}$ (MHz) |
|---|---|---|---|---|---|---|---|
| C_IS_A_16550 | C_DEV_MIR_ENABLE | C_HAS_EXTERNAL_XIN | C_HAS_EXTERNAL_RCLK | Slices | Slice Flip-Flops | 4-input LUTs | $f_{MAX}$ |
| 0 | 0 | 0 | 0 | 283 | 275 | 328 | 175.35 |
| 0 | 0 | 0 | 1 | 286 | 277 | 329 | 155.95 |
| 0 | 0 | 1 | 0 | 288 | 278 | 331 | 141.90 |
| 0 | 0 | 1 | 1 | 290 | 280 | 332 | 156.42 |
| 0 | 1 | 0 | 0 | 292 | 280 | 344 | 183.68 |
| 0 | 1 | 0 | 1 | 294 | 282 | 345 | 171.35 |
| 0 | 1 | 1 | 0 | 295 | 283 | 346 | 172.56 |
| 0 | 1 | 1 | 1 | 296 | 285 | 347 | 163.18 |
| 1 | 0 | 0 | 0 | 396 | 325 | 522 | 114.67 |
| 1 | 0 | 0 | 1 | 398 | 327 | 523 | 117.99 |
| 1 | 0 | 1 | 0 | 407 | 331 | 528 | 118.27 |
| 1 | 0 | 1 | 1 | 410 | 333 | 529 | 150.58 |
| 1 | 1 | 0 | 0 | 405 | 330 | 540 | 148.74 |
| 1 | 1 | 0 | 1 | 407 | 332 | 541 | 146.84 |
| 1 | 1 | 1 | 0 | 415 | 336 | 544 | 162.47 |
| 1 | 1 | 1 | 1 | 417 | 338 | 545 | 144.19 |



# Specification Exceptions

## FIFO Control Register

The FIFO control register has been made read/write. Read access is controlled by setting Line Control Register bit 7.

## System Clock

The asynchronous microprocessor interface of the National Semiconductor PC16550D is synchronized to the system clock input of the UART.

### Register Addresses

All internal registers reside on a 32 bit word boundary not on 8 bit byte boundaries

## Reference Documents

The following documents contain reference information important to understand the UART design:

*National Semiconductor PC16550D UART with FIFOs* data sheet (June, 1995)
(http://www.national.com/pf/PC/PC16550D.html)

## Revision History

| Date | Version | Revision |
|---|---|---|
| 11/10/04 | 1.0 | Initial Xilinx Release. |
| 4/6/05 | 1.1 | Updated for EDK 7.1.1 SP1 release. |
| 5/2/05 | 1.2 | Converted to new DS template; incorporated CR202609; updated figures to Xilinx Graphic Standards; reformatted tables; added User Application Hints section. |
| 12/2/05 | 1.3 | Added Spartan-3E to supported device families listing. |