

Introduction

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

The XPS 16550 UART described in this document has been incorporating features described in *National Semiconductor PC16550D UART with FIFOs data sheet*.

The National Semiconductor PC16550D data sheet is referenced throughout this document and should be used as the authoritative specification. Differences between the National Semiconductor PC16550D and the XPS 16550 UART are highlighted in "[Specification Exceptions](#)" section.

Features

- Connects as a 32-bit Slave on PLB V4.6 bus of 32, 64 and 128 bits data width
- 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
 - Prioritized transmit, receive, line status and modem control interrupts
 - False start bit detection and recover
 - Line break detection and generation
 - Internal loop back diagnostic functionality
 - 16 byte transmit and receive FIFOs

| LogiCORE™ Facts | | |
|--|---|--------|
| Core Specifics | | |
| Supported Device Family | See EDK Supported Device Families | |
| Version of Core | xps_uart16550 | v3.00a |
| Resources Used | | |
| | Min | Max |
| Slices | Refer to the Table 17 , Table 18 and Table 19 , Table 20 , Table 21 | |
| LUTs | | |
| FFs | | |
| Block RAMs | N/A | |
| Special Features | N/A | |
| Provided with Core | | |
| Documentation | Product Specification | |
| Design File Formats | VHDL | |
| Constraints File | N/A | |
| Verification | N/A | |
| Instantiation Template | N/A | |
| Reference Designs & Application notes | N/A | |
| Design Tool Requirements | | |
| Xilinx Implementation Tools | See Tools for requirements. | |
| Verification | | |
| Simulation | | |
| Synthesis | | |
| Support | | |
| Provided by Xilinx, Inc. | | |

Functional Description

The XPS 16550 UART implements the hardware and software functionality of the ubiquitous National Semiconductor 16550 UART, that works in both 16450 and 16550 UART modes. For complete details please refer the National Semiconductor data sheet.

The XPS 16550 UART 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 XPS 16550 UART 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 XPS 16550 UART can transmit and receive independently.

The device can be configured and it's status monitored via the internal register set. The XPS 16550 UART 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 byte transmit and receive FIFOs. The FIFOs can be enabled or disabled through software control.

The top-level block diagram for the XPS 16550 UART is shown in [Figure 1](#).

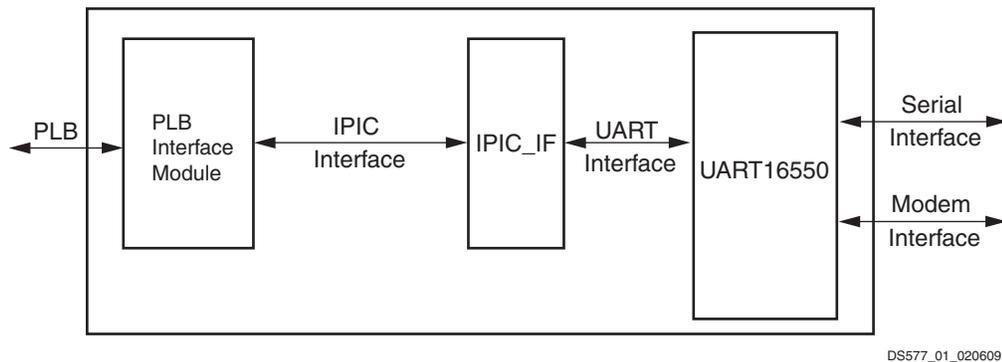
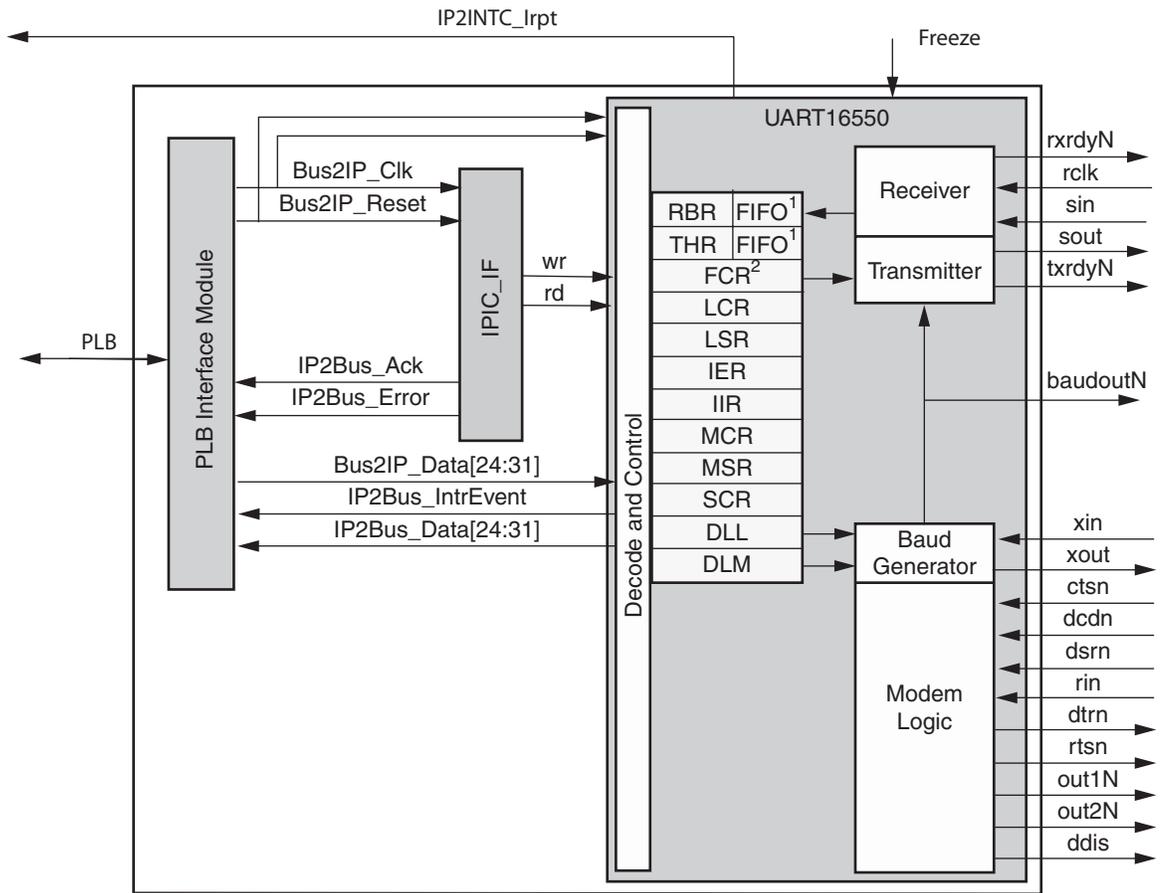


Figure 1: XPS 16550 UART Top-level Block Diagram

The top level modules of the XPS 16550 UART are:

- PLB Interface Module
- IPIC_IF
- UART16550

The detailed block diagram for the XPS 16550 UART is shown in [Figure 2](#).



Note:
 1. 16450 UART mode does not support the FCR.
 2. 16450 UART mode does not support the FIFOs.

DS577_02_020609

Figure 2: XPS 16550 UART Detailed Block Diagram

PLB Interface Module

PLB Interface Module provides bidirectional interface between UART 16550 module and the PLB. The base element of the PLB Interface Module is slave attachment, which provides the basic functionality of PLB slave operation.

IPIC_IF

IPIC_IF module incorporates logic to acknowledge the write and read transactions initiated by the plbv46 slave single module to write into the UART 16550 module registers and read from UART 16550 module registers.

UART 16550

UART 16550 provides all the core features for transmission, reception of data and modem features of UART. The UART 16550 module of XPS 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 one, the UART 16550 module has FIFOs instantiated to support 16550 mode of operation. When C_IS_A_16550 is set to zero, the UART 16550 module works without FIFOs in 16450 mode. In 16550 mode, the FIFOs can be enabled by configuring FCR register.

XPS 16550 UART I/O Signals

The XPS 16550 UART I/O signals are listed and described in [Table 1](#).

Table 1: XPS 16550 UART I/O Signals

| Port | Signal Name | Interface | I/O | Initial State | Description |
|--|--|-----------|-----|---------------|--|
| System Signals | | | | | |
| P1 | Freeze | System | I | - | Freezes UART for software debug (active high) |
| P2 | IP2INTC_Irpt | System | O | 0 | Device interrupt output to microprocessor interrupt input or system interrupt controller (active high) |
| P3 | SPLB_Clk | System | I | - | PLB clock |
| P4 | SPLB_Rst | System | I | - | PLB reset (active high) |
| PLB Master Interface Signals | | | | | |
| P5 | PLB_ABus[0 : C_SPLB_AWIDTH-1] | PLB | I | - | PLB address bus |
| P6 | PLB_PAVAlid | PLB | I | - | PLB primary address valid indicator |
| P7 | PLB_masterID[0 : C_SPLB_MID_WIDTH - 1] | PLB | I | - | PLB current master identifier |
| P8 | PLB_RNW | PLB | I | - | PLB read not write |
| P9 | PLB_BE[0 : [C_SPLB_DWIDTH/8] - 1] | PLB | I | - | PLB byte enables |
| P10 | PLB_size[0 : 3] | PLB | I | - | PLB transfer size |
| P11 | PLB_type[0 : 2] | PLB | I | - | PLB transfer type |
| P12 | PLB_wrDBus[0 : C_SPLB_DWIDTH - 1] | PLB | I | - | PLB write data bus |
| Unused PLB Master Interface Signals | | | | | |
| P13 | PLB_UABus[0 : 31] | PLB | I | - | PLB upper address bits |
| P14 | PLB_SAVAlid | PLB | I | - | PLB secondary address valid |
| P15 | PLB_rdPrim | PLB | I | - | PLB secondary to primary read request indicator |
| P16 | PLB_wrPrim | PLB | I | - | PLB secondary to primary write request indicator |
| P17 | PLB_abort | PLB | I | - | PLB abort bus request |
| P18 | PLB_busLock | PLB | I | - | PLB bus lock |
| P19 | PLB_MSize[0 : 1] | PLB | I | - | PLB data bus width indicator |
| P20 | PLB_TAttribute[0 : 15] | PLB | I | - | PLB transfer attribute |
| P21 | PLB_lockerr | PLB | I | - | PLB lock error |
| P22 | PLB_wrBurst | PLB | I | - | PLB burst write transfer |
| P23 | PLB_rdBurst | PLB | I | - | PLB burst read transfer |
| P24 | PLB_wrPendReq | PLB | I | - | PLB pending bus write request |
| P25 | PLB_rdPendReq | PLB | I | - | PLB pending bus read request |
| P26 | PLB_rdPendPri[0 : 1] | PLB | I | - | PLB pending read request priority |
| P27 | PLB_wrPendPri[0 : 1] | PLB | I | - | PLB pending write request priority |
| P28 | PLB_reqPri[0 : 1] | PLB | I | - | PLB current request priority |

Table 1: XPS 16550 UART I/O Signals (Cont'd)

| Port | Signal Name | Interface | I/O | Initial State | Description |
|------------------------------------|---------------------------------------|-----------|-----|---------------|---|
| PLB Slave Interface Signals | | | | | |
| P29 | SI_addrAck | PLB | O | 0 | Slave address acknowledge |
| P30 | SI_SSize[0 : 1] | PLB | O | 0 | Slave data bus size |
| P31 | SI_wait | PLB | O | 0 | Slave wait indicator |
| P32 | SI_rearbitrate | PLB | O | 0 | Slave rearbitrate bus indicator |
| P33 | SI_wrDack | PLB | O | 0 | Slave write data acknowledge |
| P34 | SI_wrComp | PLB | O | 0 | Slave write transfer complete indicator |
| P35 | SI_rdBus[0 : C_SPLB_DWIDTH-1] | PLB | O | 0 | Slave read data bus |
| P36 | SI_rdDack | PLB | O | 0 | Slave read data acknowledge |
| P37 | SI_rdComp | PLB | O | 0 | Slave read transfer complete indicator |
| P38 | SI_MBusy[0 : C_SPLB_NUM_MASTERS - 1] | PLB | O | 0 | Slave busy indicator |
| P39 | SI_MWrErr[0 : C_SPLB_NUM_MASTERS - 1] | PLB | O | 0 | Slave write error indicator |
| P40 | SI_MRdErr[0 : C_SPLB_NUM_MASTERS - 1] | PLB | O | 0 | Slave read error indicator |
| Unused PLB Slave Interface Signals | | | | | |
| P41 | SI_wrBTerm | PLB | O | 0 | Slave terminate write burst transfer |
| P42 | SI_rdWdAddr[0 : 3] | PLB | O | 0 | Slave read word address |
| P43 | SI_rdBTerm | PLB | O | 0 | Slave terminate read burst transfer |
| P44 | SI_MIRQ[0 : C_SPLB_NUM_MASTERS - 1] | PLB | O | 0 | Master interrupt request |
| UART Signals | | | | | |
| P45 | baudoutN | Serial | O | 1 | 16 x clock signal from the transmitter section of the UART |
| P46 | rclk | Serial | I | - | Receiver 16x clock (Optional, may be driven externally under control of the C_HAS_EXTERNAL_RCLK parameter) |
| P47 | sin | Serial | I | - | Serial data input |
| P48 | sout | Serial | O | 1 | Serial data output |
| P49 | xin | Serial | I | - | Baud rate generator reference clock (Optional, may be driven externally under control of the C_HAS_EXTERNAL_XIN parameter) |
| P50 | xout | Serial | O | - | If C_HAS_EXTERNAL_XIN = 0, xout will be 0, if C_HAS_EXTERNAL_XIN = 1 xout can be used as Baud rate generator reference feedback clock |
| P51 | ctsN | Modem | I | - | Clear to send (active low). When low, this indicates that the MODEM or data set is ready to exchange data. |
| P52 | dcdN | Modem | I | - | Data carrier detect (active low). When low, indicates that the data carrier has been detected by the MODEM or data set. |
| P53 | dsrN | Modem | I | - | Data set ready (active low). When low, this indicates that the MODEM or data set is ready to establish the communications link with the UART. |

Table 1: XPS 16550 UART I/O Signals (Cont'd)

| Port | Signal Name | Interface | I/O | Initial State | Description |
|------|-------------|-----------|-----|---------------|---|
| P54 | dtrN | Modem | O | 1 | Data terminal ready (active low). When low, this informs the MODEM or data set that the UART is ready to establish a communications link. |
| P55 | riN | Modem | I | | Ring indicator (active low). When low, this indicates that a telephone ringing signal has been received by the MODEM or data set. |
| P56 | rtsN | Modem | O | 1 | Request to send (active low). When low, this informs the MODEM or data set that the UART is ready to exchange data. |
| P57 | ddis | User | O | 1 | Driver disable. This goes low when CPU is reading data from UART. |
| P58 | out1N | User | O | 1 | User controlled output |
| P59 | our2N | User | O | 1 | User controlled output |
| P60 | rxrdyN | User | O | 1 | DMA control signal |
| P61 | txrdyN | User | O | 0 | DMA control signal |

XPS 16550 UART Design Parameters

To allow the user to create a XPS 16550 UART that is uniquely tailored for the user's system, certain features are parameterizable in the XPS 16550 UART design. This allows the user to have a design that utilizes only the resources required by the system and runs at the best possible performance. The features that are parameterizable in the XPS 16550 UART core are as shown in Table 2.

Table 2: Design Parameters

| Generic | Parameter Description | Parameter Name | Allowable Values | Default Value | VHDL Type |
|-------------------|---------------------------------------|--------------------|---|---------------------|------------------|
| System Parameters | | | | | |
| G1 | XILINX FPGA Family | C_FAMILY | See C_FAMILY parameter values | | string |
| PLB Parameters | | | | | |
| G2 | XPS 16550 UART Base Address | C_BASEADDR | Valid Word Aligned Address ⁽¹⁾ | None ⁽²⁾ | std_logic_vector |
| G3 | XPS 16550 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 |
| G4 | PLB Data Bus Width | C_SPLB_DWIDTH | 32, 64, 128 | 32 | integer |
| G5 | PLB Address Bus Width | C_SPLB_AWIDTH | 32 | 32 | integer |
| G6 | PLB Point-to-Point or shared topology | C_SPLB_P2P | 0 : PLB shared topology 1 : Reserved | 0 | integer |
| G7 | PLB master ID bus width | C_SPLB_MID_WIDTH | $\log_2(C_SPLB_NUM_MASTERS)$ with a minimum value of 1 | 1 | integer |
| G8 | Number of PLB masters | C_SPLB_NUM_MASTERS | 1 - 16 | 1 | integer |

Table 2: Design Parameters (Cont'd)

| Generic | Parameter Description | Parameter Name | Allowable Values | Default Value | VHDL Type |
|----------------------|-------------------------------------|--------------------------------------|---|---------------|-----------|
| G9 | Width of slave data bus | C_SPLB_NATIVE_DWIDTH | 32 | 32 | integer |
| G10 | Burst support | C_SPLB_SUPPORTS_BURST | 0 | 0 | integer |
| 16550 UART Interface | | | | | |
| G11 | External xin | C_HAS_EXTERNAL_XIN | 0 : xin is open ⁽³⁾ 1 : xin is externally driven ⁽⁴⁾ | 0 | integer |
| G12 | External rclk | C_HAS_EXTERNAL_RCLK | 0 : rclk is open 1 : rclk is externally driven | 0 | integer |
| G13 | Select 16450/16550 UART | C_IS_A_16550 | 0 : 16450 mode 1 : 16550 mode | 1 | integer |
| G14 | External xin clock frequency in Hz. | C_EXTERNAL_XIN_CLK_HZ ⁽⁵⁾ | Valid xin clock frequency in Hz. | 25000000 | integer |

Notes:

1. Address range specified by C_BASEADDR and C_HIGHADDR must be at least 0x2000 and must be 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.
3. When C_HAS_EXTERNAL_XIN=0, this core uses SPLB_Clk as an reference clock for the baud calculation. User must use SPLB_Clk frequency to calculate baud divisor value for DLL and DLM register configuration.
4. The external xin input clock must be less than half of SPLB_Clk.
5. External xin clock frequency. User must configure this parameter when external xin is used. (C_HAS_EXTERNAL_XIN is '1').

Parameter - Port Dependencies

The dependencies between the XPS 16550 UART core design parameters and I/O signals are described in Table 3. In addition, when certain features are parameterized out of the design, the related logic will no longer be a part of the design. The unused input signals and related output signals are set to a specified value.

Table 3: Parameter-Port Dependencies

| Generic or Port | Name | Affects | Depends | Relationship Description |
|--------------------------|---------------------------------------|---------------|---------|--|
| Design Parameters | | | | |
| G4 | C_SPLB_DWIDTH | P9, P12, P35 | - | Affects the size of the PLB data bus |
| G5 | C_SPLB_AWIDTH | P5 | - | Affects the size of the PLB address bus |
| G7 | C_SPLB_MID_WIDTH | P7 | G8 | Affects the width of the PLB master ID |
| G8 | C_SPLB_NUM_MASTERS | P38, P39, P40 | - | Identify the specific master on the PLB |
| G11 | C_HAS_EXTERNAL_XIN | P49, P50 | - | Affects the generation of baud rate |
| G12 | C_HAS_EXTERNAL_RCLK | P46 | - | Affects the usage of 16x receiver clock |
| I/O Signals | | | | |
| P5 | PLB_ABus[0:C_SPLB_AWIDTH - 1] | - | G5 | Width varies with the size of the PLB address bus |
| P7 | PLB_masterID[0: C_SPLB_MID_WIDTH - 1] | - | G7 | Width varies with the size of the number of masters on the PLB |
| P9 | PLB_BE[0:(C_SPLB_DWIDTH/8) - 1] | - | G4 | Width varies with the size of the PLB data bus |
| P12 | PLB_wrDBus[0:C_SPLB_DWIDTH - 1] | - | G4 | Width varies with the size of the PLB data bus |
| P35 | SI_rdBus[0:C_SPLB_DWIDTH - 1] | - | G4 | Width varies with the size of the PLB data bus |

Table 3: Parameter-Port Dependencies (Cont'd)

| Generic or Port | Name | Affects | Depends | Relationship Description |
|-----------------|-------------------------------------|---------|----------|--|
| P38 | SI_MBusy[0:C_SPLB_NUM_MASTERS - 1] | - | G8 | Width varies with the number of masters on the PLB |
| P39 | SI_MWrErr[0:C_SPLB_NUM_MASTERS - 1] | - | G8 | Width varies with the number of masters on the PLB |
| P40 | SI_MRdErr[0:C_SPLB_NUM_MASTERS - 1] | - | G8 | Width varies with the number of masters on the PLB |
| P46 | rclk | - | G12, P45 | If C_HAS_EXTERNAL_RCLK = 0 baudoutN is used as 16x receiver clock, C_HAS_EXTERNAL_RCLK = 1, rclk is used as 16x receiver clk |
| P49 | xin | - | G11 | When C_HAS_EXTERNAL_XIN = 0, xin is unconnected, C_HAS_EXTERNAL_XIN = 1, xin is driven externally |

XPS 16550 UART Register Definition

XPS 16550 UART Interface

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

Table 4: XPS 16550 UART Registers

| Register Name | LCR(7)+ 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) ⁽³⁾ | X + C_BASEADDR + 0x1008 | Write |
| FIFO Control Register ^{(2), (3)} | 1 + C_BASEADDR + 0x1008 | Read |
| Line Control Register (LCR) | X ⁽¹⁾ + C_BASEADDR + 0x100C | Read/Write |
| Modem Control Register (MCR) | X ⁽¹⁾ + C_BASEADDR + 0x1010 | Read/Write |
| Line Status Register (LSR) | X ⁽¹⁾ + C_BASEADDR + 0x1014 | Read/Write |
| Modem Status Register (MSR) | X ⁽¹⁾ + C_BASEADDR + 0x1018 | Read/Write |
| Scratch Register (SCR) | X ⁽¹⁾ + 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
3. 16450 UART mode implementation does not include this register

XPS 16550 UART Register Logic

This section tabulates the internal XPS 16550 UART registers, including their reset values (if any). Please refer to the National Semiconductor PC16550D UART with FIFOs data sheet (June, 1995) 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: Receiver Buffer Register (RBR)

Table 5: Receiver Buffer Register Bit Definitions

| Bit | Name | Access | Reset Value | Description |
|-------|----------|--------|-------------|-------------------------|
| 0-23 | Reserved | N/A | N/A | Reserved |
| 24-31 | RBR | Read | "00000000" | 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 | Reserved | N/A | N/A | Reserved |
| 24-31 | THR | Write | "11111111" | 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

| Bit | Name | Access | Reset Value | Description |
|-------|----------|------------|-----------------------|--|
| 0-23 | Reserved | N/A | N/A | Reserved |
| 24-27 | N/A | Read/Write | "0000" ⁽¹⁾ | Always returns "0000" |
| 28 | EDSSI | Read/Write | '0' | Enable Modem Status Interrupt '0' = Disables Modem Status Interrupts '1' = Enables Modem Status Interrupts |
| 29 | ELSI | Read/Write | '0' | Enable Receiver Line Status Interrupt '0' = Disables Receiver Line Status Interrupts '1' = Enables Receiver Line Status Interrupts |
| 30 | ETBEI | Read/Write | '0' | Enable Transmitter Holding Register Empty Interrupt '0' = Disables Transmitter Holding Register Empty Interrupts '1' = Enables Transmitter Holding Register Interrupts |
| 31 | ERBFI | Read/Write | '0' | Enable Received Data Available Interrupt '0' = Disables Received Data Available Interrupts '1' = Enables Received Data Available Interrupts |

1. Reading these bits always return "0000"

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

| Bit | Name | Access | Reset Value | Description |
|------------------------------|------------------------|--------|-----------------------------|---|
| 0-23 | Reserved | N/A | N/A | Reserved |
| 24-25 | FIFOEN ⁽³⁾ | Read | "00" | FIFOs Enabled. Always zero if not in FIFO mode |
| 26-27 | N/A | Read | "00"<RD Red> ⁽¹⁾ | Always returns "00" |
| 28<RD Red> ⁽³⁾ 30 | INTID2 | Read | "000" | Interrupt ID "011" = Receiver Line Status (Highest) "010" = Received Data Available (Second) "110" = Character Timeout (Second) "001" = Transmitter Holding Register Empty (Third) "000" = Modem Status (Fourth) |
| 31 | INTPEND ⁽²⁾ | Read | '1' | Interrupt Pending. Interrupt is pending when cleared |

Notes:

1. Reading these bits always return "00"
2. If INTPEND = '0', interrupt is pending. See National Semiconductor PC16550D data sheet for more details
3. Bits are always zero in 16450 UART mode

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.

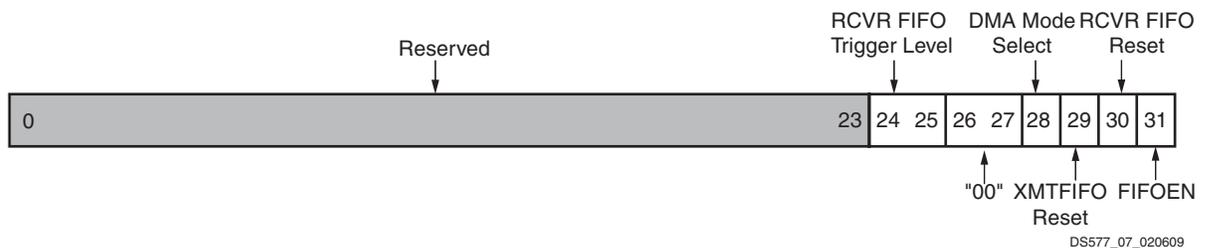


Figure 7: FIFO Control Register (FCR)

Table 9: FIFO Control Register Bit Definitions⁽¹⁾

| Bit | Name | Access | Reset Value | Description |
|-------|-------------------------|------------|---------------------|--|
| 0-23 | Reserved | N/A | N/A | Reserved |
| 24-25 | RCVR FIFO Trigger Level | Read/Write | "00" | RCVR FIFO Trigger Level. "00" = 1 byte "01" = 4 bytes "10" = 8 bytes "11" = 14 bytes |
| 26-27 | Reserved | Read/Write | "00" ⁽²⁾ | Always returns "00" |
| 28 | DMA Mode Select | Read/Write | '0' | DMA Mode Select '0' = Mode 0 '1' = Mode 1 |

Table 9: FIFO Control Register Bit Definitions⁽¹⁾ (Cont'd)

| Bit | Name | Access | Reset Value | Description |
|-----|-----------------|------------|-------------|--|
| 29 | XMIT FIFO Reset | Read/Write | '0' | Transmitter FIFO Reset '1' = Resets XMIT FIFO |
| 30 | RCVR FIFO Reset | Read/Write | '0' | Receiver FIFO Reset '1' = Resets RCVR FIFO |
| 31 | FIFOEN | Read/Write | '0' | FIFO Enable '1' = Enables FIFOs |

Notes:

1. FCR is not included in 16450 UART mode
2. Reading these bits always return "00"

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.

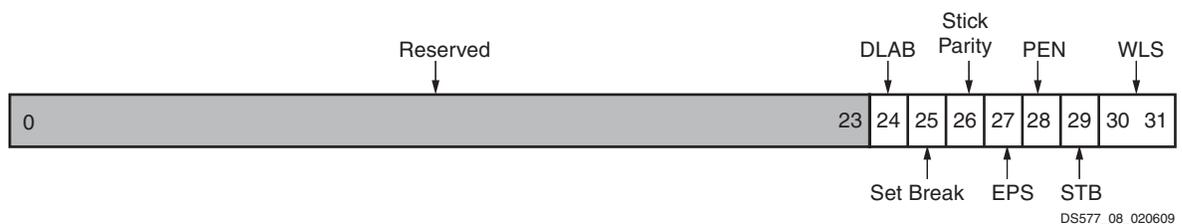


Figure 8: Line Control Register (LCR)

Table 10: Line Control Register Bit Definitions

| Bit | Name | Access | Reset Value | Description |
|------|--------------|------------|-------------|---|
| 0-23 | Reserved | N/A | N/A | Reserved |
| 24 | DLAB | Read/Write | '0' | Divisor Latch Access Bit. '1' = Allows access to the Divisor Latch Registers and reading of the FIFO Control Register '0' = Allows access to RBR, THR, IER and IIR registers |
| 25 | Set Break | Read/Write | '0' | Set Break '1' = Enables break condition. Sets SOUT to '0' '0' = Disables break condition. |
| 26 | Stick Parity | Read/Write | '0' | Stick Parity '1' = When bits 28, 27 are logic1 the Parity bit is transmitted and checked as a logic 0. If bit 27 is a logic 0 and bit 28 is logic 1 then the Parity bit is transmitted and checked as a logic 1. '0' = Stick Parity is disabled |
| 27 | EPS | Read/Write | '0' | Even Parity Select '1' = Selects Even parity '0' = Selects Odd parity |

Table 10: Line Control Register Bit Definitions (Cont'd)

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

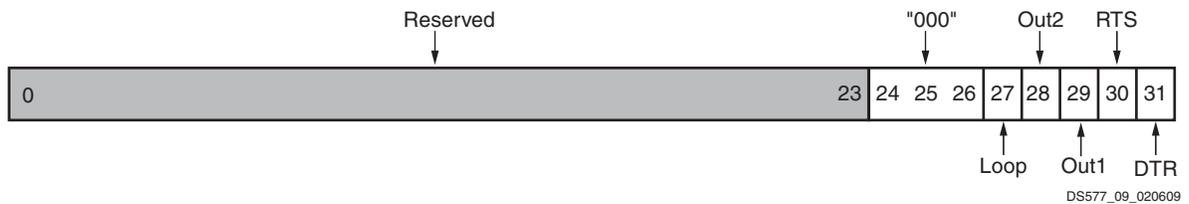


Figure 9: Modem Control Register (MCR)

Table 11: Modem Control Register Bit Definitions

| Bit | Name | Access | Reset Value | Description |
|-------|----------|------------|------------------|--|
| 0-23 | Reserved | N/A | N/A | Reserved |
| 24-26 | N/A | Read/Write | "000"<RD Red>(1) | Always "000" |
| 27 | Loop | Read/Write | '0' | Loop Back '1' = Enables loop back |
| 28 | Out2 | Read/Write | '0' | User Output 2 '1' = Drives OUT2N low '0' = Drives OUT2N high |
| 29 | Out1 | Read/Write | '0' | User Output 1 '1' = Drives OUT1N low '0' = Drives OUT1N high |

Table 11: Modem Control Register Bit Definitions (Cont'd)

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

Notes:

1. Reading these bits always return "000"

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.

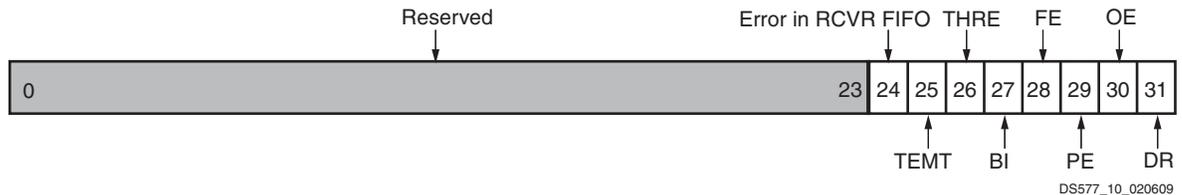


Figure 10: Line Status Register (LSR)

Table 12: Line Status Register Bit Definitions

| Bit | Name | Access | Reset Value | Description |
|------|--------------------|------------|-------------|--|
| 0-23 | Reserved | N/A | N/A | Reserved |
| 24 | Error in RCVR FIFO | Read/Write | '0' | Error in RCVR FIFO ⁽¹⁾ 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 Set when SIN is held low for an entire character time |
| 28 | FE | Read/Write | '0' | Framing Error Character missing a stop bit. Receiver re-synchronizes with next character, if possible |
| 29 | PE | Read/Write | '0' | Parity Error |
| 30 | OE | Read/Write | '0' | Overrun Error RBR not read before next character is received |
| 31 | DR | Read/Write | '0' | Data Ready |

1. The error will be reported until the last character containing error in the FIFO gets read out of the FIFO.

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

| Bit | Name | Access | Reset Value | Description |
|------|----------|------------|-------------|--|
| 0-23 | Reserved | N/A | N/A | Reserved |
| 24 | DCD | Read/Write | 'X' | Data Carrier Detect Complement of DCDN input |
| 25 | RI | Read/Write | 'X' | Ring Indicator Complement of RIN input |
| 26 | DSR | Read/Write | 'X' | Data Set Ready Complement of DSRN input |
| 27 | CTS | Read/Write | 'X' | Clear To Send Complement of CTSN input |
| 28 | DDCD | Read/Write | '0' | Delta Data Carrier Detect Change in DCDN since last MSR read |
| 29 | TERI | Read/Write | '0' | Trailing Edge Ring Indicator RIN has changed from a low to a high |
| 30 | DDSR | Read/Write | '0' | Delta Data Set Ready Change in DSRN since last MSR read |
| 31 | DCTS | Read/Write | '0' | Delta Clear To Send 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 12. 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 | Reserved | N/A | N/A | Reserved |
| 24-31 | Scratch | Read/Write | "00000000" | Hold the data temporarily |

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

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

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

Divisor Latch (Most Significant Byte) Register

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 | Reserved | N/A | N/A | Reserved |
| 24-31 | DLM | Read/Write | "00000000" | Divisor Latch Most Significant Byte |

User Application Hints

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

1. 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.
2. Write Interrupt Enable Register to activate the individual interrupts
3. Write to the FIFO Control Register to enable the FIFO's, clear the FIFO's, set the RCVR FIFO trigger level.
4. 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.
5. Service the interrupts when ever an interrupt is triggered by the XPS 16550 UART.

An example use of the XPS 16550 UART with the operating mode set to the following parameters in 16550 mode explained below.

- Baud rate: 56Kbps
 - System clock : 100 Mhz (C_HAS_EXTERNAL_XIN = 0)
 - 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_006F 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.

An example use of the XPS 16550 UART when external xin clock is used (C_HAS_EXTERNAL_XIN = 1) with the operating mode set to the following parameters in 16550 mode explained below.

- Baud rate: 56Kbps
 - System clock : 100 Mhz
 - External xin clock : 1.8432 Mhz
 - 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_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. Other steps remains same as show in the example above.

Design Implementation

Target Technology

The target technology is an FPGA listed in [EDK Supported Device Families](#).

Device Utilization and Performance Benchmarks

Core Performance

Because the XPS 16550 UART core will be used with other design modules in the FPGA, the utilization and timing numbers reported in this section are estimates only. When the XPS 16550 UART core is combined with other designs in the system, the utilization of FPGA resources and timing of the XPS 16550 UART design will vary from the results reported here.

The XPS 16550 UART resource utilization for various parameter combinations measured with Virtex[®]-4 as the target device are detailed [Table 17](#).

Table 17: Performance and Resource Utilization Benchmarks on Virtex-4 (xc4vlx40-ff1148-10)

| Parameter Values | | | | | | Device Resources | | | Performance |
|------------------|--------------------|------------|---------------|--------------------|---------------------|------------------|------------------|------|------------------------|
| C_IS_A_16550 | C_SPLB_NUM_MASTERS | C_SPLB_P2P | C_SPLB_DWIDTH | C_HAS_EXTERNAL_XIN | C_HAS_EXTERNAL_RCLK | Slices | Slice Flip-Flops | LUTs | F _{MAX} (MHz) |
| 0 | 1 | 1 | 32 | 0 | 0 | 307 | 263 | 419 | 125 |
| 0 | 4 | 0 | 64 | 0 | 1 | 362 | 282 | 413 | 125 |
| 0 | 8 | 0 | 128 | 1 | 0 | 373 | 283 | 436 | 125 |
| 0 | 8 | 0 | 128 | 1 | 1 | 454 | 334 | 603 | 125 |
| 1 | 8 | 0 | 64 | 0 | 0 | 470 | 332 | 617 | 125 |
| 1 | 1 | 0 | 128 | 0 | 1 | 452 | 331 | 593 | 125 |
| 1 | 1 | 0 | 128 | 1 | 0 | 459 | 331 | 612 | 125 |
| 1 | 16 | 0 | 128 | 1 | 1 | 447 | 335 | 617 | 125 |

The XPS 16550 UART resource utilization for various parameter combinations measured with Virtex-5 as the target device are detailed [Table 18](#).

Table 18: Performance and Resource Utilization Benchmarks on Virtex-5 (xc5vlx50-ff1153-1)

| Parameter Values | | | | | | Device Resources | | Performance |
|------------------|--------------------|------------|---------------|--------------------|---------------------|------------------|------|------------------------|
| C_IS_A_16550 | C_SPLB_NUM_MASTERS | C_SPLB_P2P | C_SPLB_DWIDTH | C_HAS_EXTERNAL_XIN | C_HAS_EXTERNAL_RCLK | Slice Flip-Flops | LUTs | F _{MAX} (MHz) |
| 0 | 1 | 1 | 32 | 0 | 0 | 324 | 411 | 150 |
| 0 | 4 | 0 | 64 | 0 | 1 | 286 | 300 | 150 |
| 0 | 8 | 0 | 128 | 1 | 0 | 282 | 307 | 150 |
| 0 | 8 | 0 | 128 | 1 | 1 | 335 | 420 | 150 |
| 1 | 8 | 0 | 64 | 0 | 0 | 332 | 412 | 150 |
| 1 | 1 | 0 | 128 | 0 | 1 | 332 | 408 | 150 |
| 1 | 1 | 0 | 128 | 1 | 0 | 331 | 406 | 150 |
| 1 | 16 | 0 | 128 | 1 | 1 | 336 | 424 | 150 |

The XPS 16550 UART resource utilization for various parameter combinations measured with Spartan®-3E as the target device are detailed [Table 19](#).

Table 19: Performance and Resource Utilization Benchmarks on Spartan-3E (xc3s1600e-fg484-4)

| Parameter Values | | | | | | Device Resources | | | Performance |
|------------------|--------------------|------------|---------------|--------------------|---------------------|------------------|------------------|------|------------------------|
| C_IS_A_16550 | C_SPLB_NUM_MASTERS | C_SPLB_P2P | C_SPLB_DWIDTH | C_HAS_EXTERNAL_XIN | C_HAS_EXTERNAL_RCLK | Slices | Slice Flip-Flops | LUTs | F _{MAX} (MHz) |
| 0 | 1 | 1 | 32 | 0 | 0 | 287 | 257 | 367 | 100 |
| 0 | 4 | 0 | 64 | 0 | 1 | 340 | 281 | 379 | 100 |
| 0 | 8 | 0 | 128 | 1 | 0 | 368 | 282 | 387 | 100 |
| 0 | 8 | 0 | 128 | 1 | 1 | 466 | 333 | 549 | 100 |
| 1 | 8 | 0 | 64 | 0 | 0 | 445 | 331 | 547 | 100 |
| 1 | 1 | 0 | 128 | 0 | 1 | 470 | 330 | 539 | 100 |
| 1 | 1 | 0 | 128 | 1 | 0 | 447 | 330 | 543 | 100 |
| 1 | 16 | 0 | 128 | 1 | 1 | 454 | 334 | 560 | 100 |

The XPS 16550 UART resource utilization for various parameter combinations measured with Spartan-6 as the target device are detailed [Table 20](#).

Table 20: Performance and Resource Utilization Benchmarks on Spartan-6 (xc6slx16-csg324-2)

| Parameter Values | | | | | | Device Resources | | | Performance |
|------------------|--------------------|------------|---------------|--------------------|---------------------|------------------|------------------|------|------------------------|
| C_IS_A_16550 | C_SPLB_NUM_MASTERS | C_SPLB_P2P | C_SPLB_DWIDTH | C_HAS_EXTERNAL_XIN | C_HAS_EXTERNAL_RCLK | Slices | Slice Flip-Flops | LUTs | F _{MAX} (MHz) |
| 0 | 1 | 1 | 32 | 0 | 0 | 204 | 314 | 442 | 160 |
| 0 | 4 | 0 | 64 | 0 | 1 | 127 | 268 | 335 | 160 |
| 0 | 8 | 0 | 128 | 1 | 0 | 171 | 256 | 359 | 160 |
| 0 | 8 | 0 | 128 | 1 | 1 | 156 | 307 | 424 | 160 |
| 1 | 8 | 0 | 64 | 0 | 0 | 209 | 313 | 428 | 160 |
| 1 | 1 | 0 | 128 | 0 | 1 | 173 | 306 | 425 | 160 |
| 1 | 1 | 0 | 128 | 1 | 0 | 156 | 306 | 438 | 160 |
| 1 | 16 | 0 | 128 | 1 | 1 | 192 | 325 | 459 | 160 |

The XPS 16550 UART resource utilization for various parameter combinations measured with Virtex-6 as the target device are detailed [Table 21](#).

Table 21: Performance and Resource Utilization Benchmarks on Virtex-6 (xc6vlx75t-ff784-1)

| Parameter Values | | | | | | Device Resources | | | Performance |
|------------------|--------------------|------------|---------------|--------------------|---------------------|------------------|------------------|------|------------------------|
| C_IS_A_16550 | C_SPLB_NUM_MASTERS | C_SPLB_P2P | C_SPLB_DWIDTH | C_HAS_EXTERNAL_XIN | C_HAS_EXTERNAL_RCLK | Slices | Slice Flip-Flops | LUTs | F _{MAX} (MHz) |
| 0 | 1 | 1 | 32 | 0 | 0 | 186 | 384 | 475 | 160 |
| 0 | 4 | 0 | 64 | 0 | 1 | 156 | 268 | 353 | 160 |
| 0 | 8 | 0 | 128 | 1 | 0 | 156 | 268 | 353 | 160 |
| 0 | 8 | 0 | 128 | 1 | 1 | 194 | 324 | 496 | 160 |
| 1 | 8 | 0 | 64 | 0 | 0 | 196 | 322 | 479 | 160 |

Table 21: Performance and Resource Utilization Benchmarks on Virtex-6 (xc6vlx75t-ff784-1)

| | | | | | | | | | |
|---|----|---|-----|---|---|-----|-----|-----|-----|
| 1 | 1 | 0 | 128 | 0 | 1 | 221 | 314 | 470 | 160 |
| 1 | 1 | 0 | 128 | 1 | 0 | 198 | 314 | 485 | 160 |
| 1 | 16 | 0 | 128 | 1 | 1 | 193 | 333 | 508 | 160 |

System Performance

To measure the system performance (F_{MAX}) of this core, this core was added to a Virtex-4 system, a Virtex-5 system, and a Spartan-3A system as the Device Under Test (DUT) as shown in Figure 15, Figure 16, and Figure 17.

Because the XPS 16550 UART core is used with other design modules in the FPGA, the utilization and timing numbers reported in this section are estimates only. When this core is combined with other designs in the system, the utilization of FPGA resources and timing of the design will vary from the results reported here.

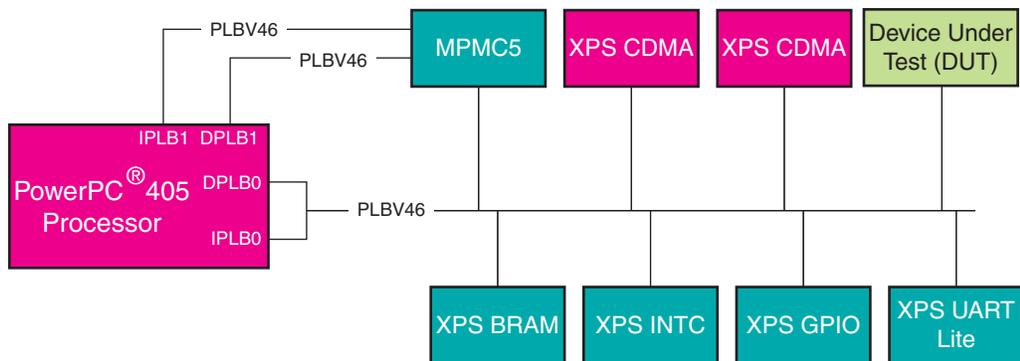


Figure 15: Virtex-4 FX System

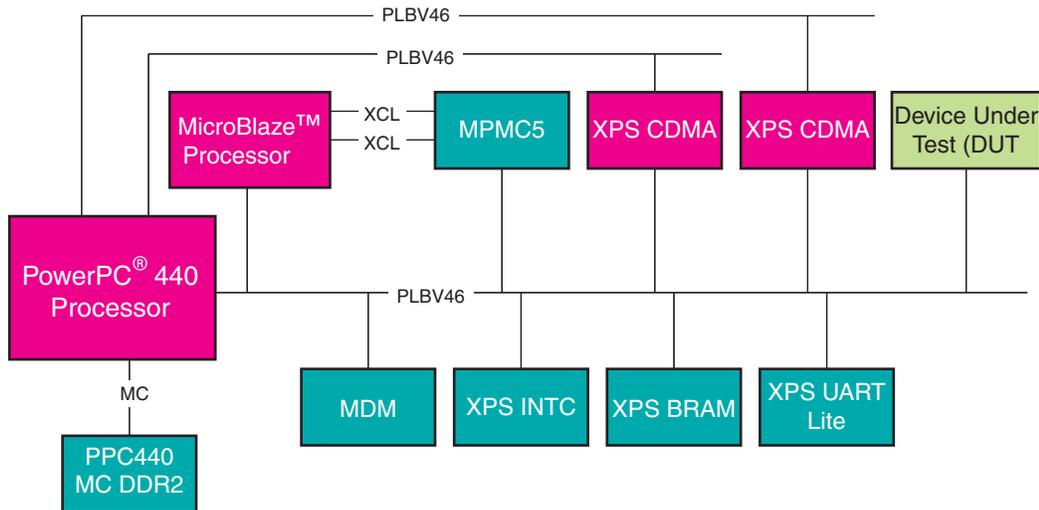


Figure 16: Virtex-5 FXT System

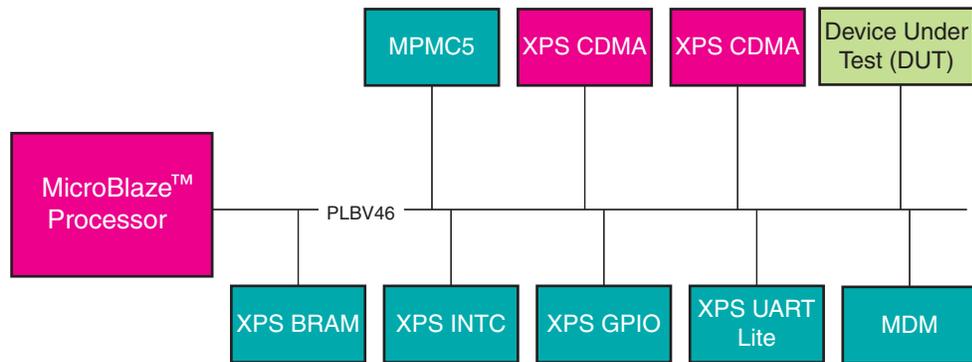


Figure 17: Spartan 3ADSP System

The target FPGA was then filled with logic to drive the LUT and BRAM utilization to approximately 70% and the I/O utilization to approximately 80%. Using the default tool options and the slowest speed grade for the target FPGA, the resulting target F_{MAX} numbers are shown in [Table 22](#).

Table 22: XPS 16550 UART System Performance

| Target FPGA | Target F_{MAX} (MHz) |
|---------------|------------------------|
| S3ADSP3400 -4 | 90 |
| V4FX60 -10 | 100 |
| V5FXT70 -1 | 120 |

The target F_{MAX} is influenced by the exact system and is provided for guidance. It is not a guaranteed value across all systems.

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.

XIN Clock

If the xin input is driven externally, then the xin clock must be less than or equal to half of the system clock. (i.e. $xin \leq (SPLB_Clk/2)$). This is mandatory for the proper functioning of the core.

Register Addresses

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

Support

Xilinx provides technical support for this LogiCORE product when used as described in the product documentation. Xilinx cannot guarantee timing, functionality, or support of product if implemented in devices that are not defined in the documentation, if customized beyond that allowed in the product documentation, or if changes are made to any section of the design labeled *DO NOT MODIFY*.

Reference Documents

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

1. National Semiconductor PC16550D UART with FIFOs data sheet (June, 1995)
(<http://www.national.com/pf/PC/PC16550D.html>)
2. IBM CoreConnect 128-Bit Processor Local Bus: Architecture Specifications version 4.6

Revision History

| Date | Version | Revision |
|------------|---------|---|
| 10/10/2006 | 1.0 | Initial Xilinx Release. |
| 4/20/2007 | 1.1 | Added Spartan-3 support. |
| 9/26/2007 | 1.2 | Added F _{MAX} Margin "System Performance" section. |
| 11/27/2007 | 1.3 | Added SP-3A DSP support. |
| 12/14/2007 | 1.4 | Modified the version from v1.00a to v2.00a to make the core license free. |
| 4/21/2008 | 1.5 | Added Automotive Spartan-3E, Automotive Spartan-3A, Automotive Spartan-3 and Automotive Spartan-3A DSP support. |
| 7/16/2008 | 1.6 | Added QPro Virtex-4 Hi-Rel and QPro Virtex-4 Rad Tolerant FPGA support. |
| 9/25/2008 | 1.7 | Updated to version v2.01.a. Removed Virtex2p support. |
| 10/08/2008 | 1.8 | Updated resource utilization and LSR(7) bit description. |
| 4/24/09 | 1.9 | Replaced references to supported device families and tool name(s) with hyperlink to PDF file. |
| 7/15/09 | 1.10 | Updated to version v3.00.a. Updated utilization and Fmax numbers for S6 and V6. |

Notice of Disclaimer

Xilinx is providing this design, code, or information (collectively, the "Information") to you "AS-IS" with no warranty of any kind, express or implied. Xilinx makes no representation that the Information, or any particular implementation thereof, is free from any claims of infringement. You are responsible for obtaining any rights you may require for any implementation based on the Information. All specifications are subject to change without notice. XILINX EXPRESSLY DISCLAIMS ANY WARRANTY WHATSOEVER WITH RESPECT TO THE ADEQUACY OF THE INFORMATION OR ANY IMPLEMENTATION BASED THEREON, INCLUDING BUT NOT LIMITED TO ANY WARRANTIES OR REPRESENTATIONS THAT THIS IMPLEMENTATION IS FREE FROM CLAIMS OF INFRINGEMENT AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Except as stated herein, none of the Information may be copied, reproduced, distributed, republished, downloaded, displayed, posted, or transmitted in any form or by any

means including, but not limited to, electronic, mechanical, photocopying, recording, or otherwise, without the prior written consent of Xilinx.