AXI4-Lite UART 16550
(Beta Release)

Version 0.1

November 14, 2023

Trademarks

All Rapid Silicon trademarks are as listed at www.rapidsilicon.com. Synopsys and Synplify Pro are trademarks of Synopsys, Inc. Aldec and Active-HDL are trademarks of Aldec, Inc. Modelsim and Questa are trademarks or registered trademarks of Siemens Industry Software Inc. or its subsidiaries in the United States or other countries. All other trademarks are the property of their respective owners.

Disclaimers

NO WARRANTIES: THE INFORMATION PROVIDED IN THIS DOCUMENT IS "AS IS" WITHOUT ANY EXPRESS OR IMPLIED WARRANTY OF ANY KIND INCLUDING WARRANTIES OF ACCURACY, COMPLETENESS, MERCHANTABILITY, NONINFRINGEMENT OF INTELLECTUAL PROPERTY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL RAPID SILICON OR ITS SUPPLIERS BE LIABLE FOR ANY DAMAGES WHATSOEVER (WHETHER DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL, INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OF OR INABILITY TO USE THE INFORMATION PROVIDED IN THIS DOCUMENT, EVEN IF RAPID SILICON HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. BECAUSE SOME JURISDICTIONS PROHIBIT THE EXCLUSION OR LIMITATION OF CERTAIN LIABILITY, SOME OF THE ABOVE LIMITATIONS MAY NOT APPLY TO YOU.

Rapid Silicon may make changes to these materials, specifications, or information, or to the products described herein, at any time without notice. Rapid Silicon makes no commitment to update this documentation. Rapid Silicon reserves the right to discontinue any product or service without notice and assumes no obligation to correct any errors contained herein or to advise any user of this document of any correction if such be made. Rapid Silicon recommends its customers obtain the latest version of the relevant information to establish that the information being relied upon is current and before ordering any products.

Contents

IP Summary

Introduction

The UART (Universal Asynchronous Receiver/Transmitter) core offers the ability to communicate serially, enabling interaction with external devices such as modems or other computers using an RS232 protocol and a serial cable. The core has been developed to be highly compatible with the National Semiconductors' 16550A device, which is an industry standard. This is an AXI-Lite compliant UART IP for easy integration with other AXI based systems.

Features

- FIFO only operation.
- Register level and functionality compatibility with NS16550A.
- Debug Interface in 32-bit data bus mode.
- AXI4-Lite interface in 32-bit or 8-bit data bus modes.

Overview

AXIL UART

The UART 16550 is a device that provides asynchronous serial communication capabilities for interacting with external devices, such as modems, printers, and other computers. It is a widely used standard in the industry for serial communication and has been around since the early 1980s.
The Universal Asynchronous Receiver/Transmitter (UART) is a component within the device that provides the ability to transmit and receive data over a serial connection. It is a soft IP core that converts parallel data into a serial stream of bits that can be transmitted over a single communication line, and then reconverts the received serial data back into parallel data.The UART 16550 is designed to be compatible with the National Semiconductors' 16550A device, which is an industry standard. This compatibility allows for easy integration of the device into existing systems and makes it a popular choice for many applications.
Overall, the UART 16550 is a reliable and efficient device that provides serial communication capabilities for a variety of applications. Its features and compatibility with industry standards make it a popular choice in many different industries, including telecommunications, industrial control, and computing. A block diagram for the AXI-Lite UART IP is shown in Figure 1.

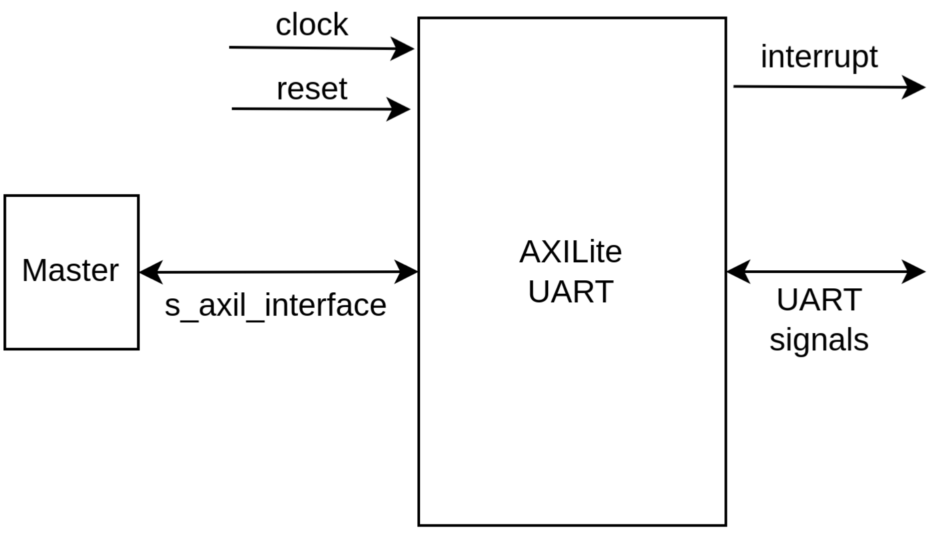

Figure 1: AXIL UART Block Diagram

IP Specification

AXI Lite UART is a type of Universal Asynchronous Receiver-Transmitter (UART) that uses the AXI Lite protocol to interface with other devices in an embedded system. UARTs are commonly used to transmit and receive data between a microcontroller or processor and other devices, such as sensors, actuators, or other microcontrollers. The AXI Lite UART provides a standard UART interface using the AXI Lite protocol, which allows it to easily integrate with other AXI Lite devices in a system. This can help simplify the overall system design and reduce the number of components required. The operation of UART can be defined by utilizing the various registers in its address space, the details of which are given below. The internal block diagram can be seen in Figure 2.

Standards

The AXI4-Lite interface is compliant with the AMBA® AXI Protocol Specification.

IP Support Details

The Table 1 gives the support details for AXIL UART.

Compliance		IP Resources				Tool Flow		
Device	**Interface**	**Source Files**	**Constraint Files**	**Testbench**	**Simulation Model**	**Analyze and Elaboration**	**Simualtion**	**Synthesis**
Gemini	AXI4-Lite	Verilog	-	-	-	Verific (Raptor)	Icarus (Raptor)	Raptor

Table 1: IP Details

Parameters

Table 2 lists the parameters of the AXIL UART.

Parameter	Values	Default Value	Description
ADDRESS WIDTH	8, 16, 32	16	AXIL Read/Write Address Width for UART
DATA WIDTH	32 / 64	32	AXIL Read/Write Data Width for UART

Table 2: Parameters

- **Note:** Data Width for the UART registers is 8 bit but due to AXI-Lite limitation, the data width for the top level interface is either kept 32 or 64 bits. Keeping in mind that the above MSBs out of the remaining 8 LSBs will be truncated or ignored when in use because of the internal 8 bit registers of UART.

Port List

Table 3 lists the top interface ports of the AXIL UART.

Signal Name	I/O	Description
AXI Clock and Reset		
s_axi_aclk	I	System Clock
s_axi_aresetn	I	Active Low Reset
Write Address Channel		
s_axi_awvalid	I	AXI4-Lite Write Address Valid
s_axi_awaddr	I	AXI4-Lite Write Address
s_axi_awprot	I	AXI4-Lite Write Address Protection type
s_axi_awready	O	AXI4-Lite Write Address Ready
Write Data Channel		
s_axi_wvalid	I	AXI4-Lite Write Data Valid
s_axi_wdata	I	AXI4-Lite Write Data
s_axi_wstrb	I	AXI4-Lite Write Data Strobe
s_axi_wready	O	AXI4-Lite Write Data Ready
Write Response Channel		
s_axi_bvalid	O	AXI4-Lite Write Response Valid
s_axi_bresp	O	AXI4-Lite Write Response
s_axi_bready	I	AXI4-Lite Write Response Ready
Read Address Channel		
s_axi_arravlid	I	AXI4-Lite Read Address Valid
s_axi_araddr	I	AXI4-Lite Read Address
s_axi_arprot	I	AXI4-Lite Read Address Protection type
s_axi_arready	O	AXI4-Lite Read Address Ready
Read Data Channel		
s_axi_rvalid	O	AXI4-Lite Read Data Valid
s_axi_rdata	O	AXI4-Lite Data
s_axi_rresp	O	AXI4-Lite Read Data Response
s_axi_rready	I	AXI4-Lite Read Data Ready
UART Signals		
int_o	O	Interrupt output
srx_pad_i	I	Serial RX input signal
stx_pad_o	O	Serial TX output signal
rts_pad_o	O	Request To Send
cts_pad_i	I	Clear To Send
dtr_pad_o	O	Data Terminal Ready
dsr_pad_i	I	Data Set Ready
ri_pad_i	I	Ring Indicator
dcd_pad_i	I	Data Carrier Detect

Table 3: AXIL UART Interface

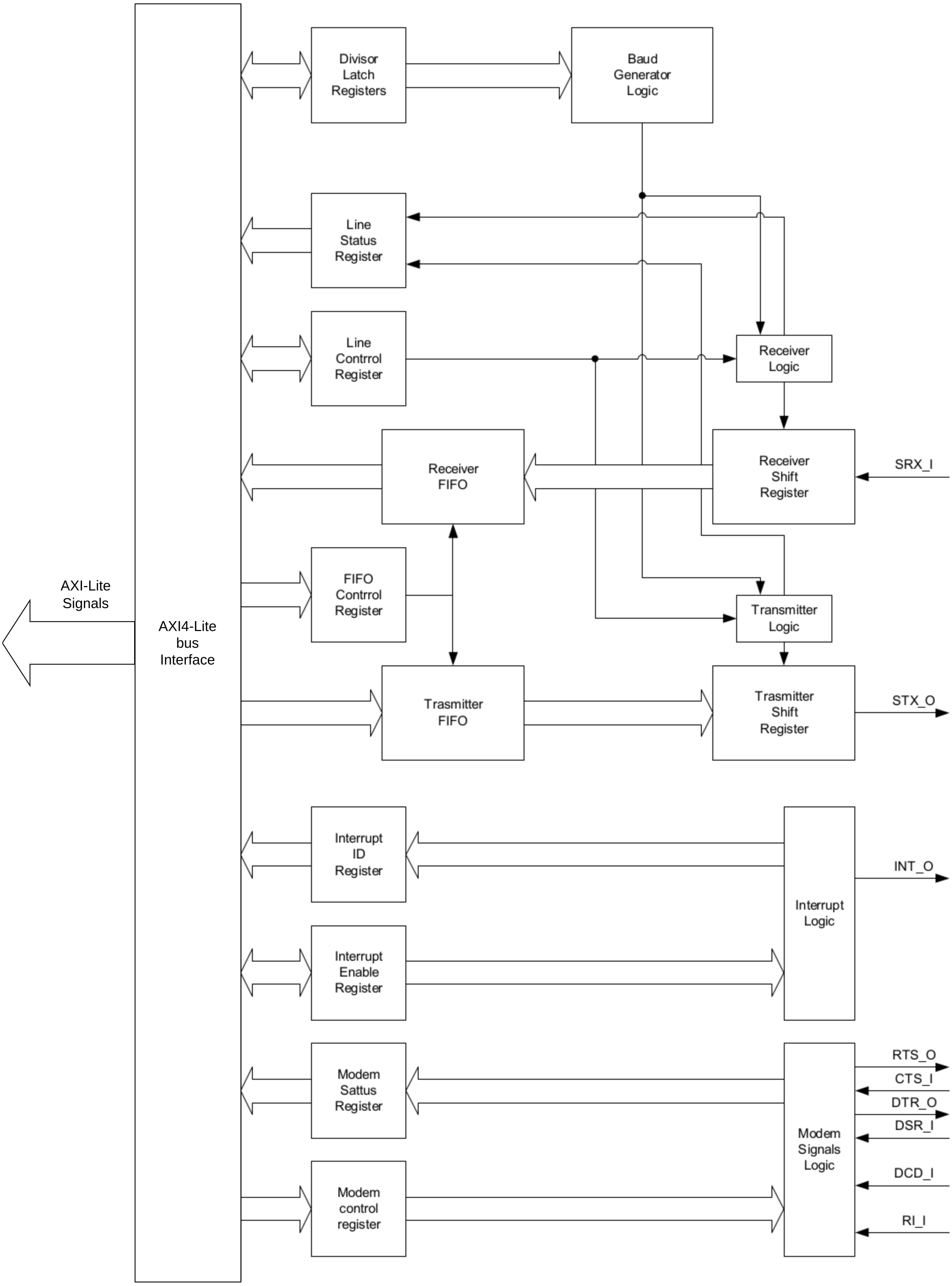

Figure 2: AXIL UART Internal Diagram

Resource Utilization

The parameters for computing the maximum and minimum resource utilization are given in Table 4, remaining parameters have been kept at their default values.

Tool	Raptor Design Suite			
FPGA Device	GEMINI			
Configuration			**Resource Utilization**	
Minumum Resource	**Options**	**Configuration**	**Resource**	**Utilized**
	ADDR WIDTH	8	LUTs	677
	DATA WIDTH	8	Registers	557
Maximum Resource	**Options**	**Configuration**	**Resource**	**Utilized**
	ADDR WIDTH	32	LUTs	814
	DATA WIDTH	64	Registers	636

Table 4: Resource Utilization

Registers and Address Space

The Table 5 shows the address space of this UART IP that can be accessed and modified according to the user requirement.

Name	Address Offset	Width	Access	Description
Receiver Buffer	0	8	R	Receiver FIFO output
Transmitter Holding Register (THR)	0	8	W	Transmit FIFO input
Interrupt Enable	1	8	RW	Enable/Mask interrupts generated by UART
Interrupt Identification	2	8	R	Get interrupt information
FIFO Control	2	8	W	Control FIFO options
Line Control Register	3	8	RW	Control connection
Modem Control	4	8	W	Controls Modem
Line Status	5	8	R	Status information
Modem Status	6	8	R	Modem Status

Table 5: Register Address Space

In addition, there are 2 Clock Divisor registers that together form one 16-bit. The registers can be accessed when the 7th (DLAB) bit of the Line Control Register is set to '1'. At this time the above registers at addresses 0-1 can't be accessed as detailed in Table 6. The list of abbreviations may be seen in Table 15 at the end of this document.

Name	Address	Width	Access	Description
Divisor Latch Byte 1 (LSB)	0	8	RW	The LSB of the divisor latch
Divisor Latch Byte 2	1	8	RW	The MSB of the divisor latch

Table 6: Divisor Latches

When using 32-bit data bus interface, additional read-only registers are available for debug purposes as detailed in Table 7:

Name	Address	Width	Access	Description
Debug 1	8	32	R	First debug register
Debug 2	12	32	R	Second debug register

Table 7: Debug Registers

- **Interrupt Enable Register (IER)**

  This register allows enabling and disabling interrupt generation by the UART detailed in Table 8.

Bit	Access	Description
0	RW	Received Data available interrupt ‘0’ – disabled ‘1’ – enabled
1	RW	Transmitter Holding Register empty interrupt ‘0’ – disabled ‘1’ – enabled
2	RW	Receiver Line Status Interrupt ‘0’ – disabled ‘1’ – enabled
3	RW	Modem Status Interrupt ‘0’ – disabled ‘1’ – enabled
7 - 4	RW	Reserved. Should be logic ‘0’.

Table 8: IER Register

  Reset Value: 00h

- **Interrupt Identification Register (IIR)**

  The IIR enables the programmer to retrieve what is the current highest priority pending interrupt. Bit 0 indicates that an interrupt is pending when it’s logic ‘0’. When it’s ‘1’ – no interrupt is pending. The Table 9 displays the list of possible interrupts along with the bits they enable, priority, and their source and reset control.

Bit 3	Bit 2	Bit 1	Priority	Interrupt Type	Interrupt Source	Interrupt Reset Control
0	1	1	1st	Receiver Line Status	Parity, Overrun or Framing errors or Break Interrupt	Reading the Line Status Register
0	1	0	2nd	Receiver Data Available	FIFO trigger level reached	FIFO drops below trigger level
1	1	0	2nd	Timeout Indication	There’s at least 1 character in the FIFO but no character has been input to the FIFO or read from it for the last 4 Char times.	Reading from the FIFO (Receiver Buffer Register)
0	0	1	3rd	Transmitter Holding Register Empty	Transmitter Holding Register Empty	Writing to the Transmitter Holding Register or reading IIR.
0	0	0	4th	Modem Status	CTS, DSR, RI or DCD.	Reading the Modem status register.

Table 9: IIR Register

Bits 4 and 5: Logic ‘0’.
Bits 6 and 7: Logic ‘1’ for compatibility reason.
Reset Value: C1h

- **FIFO Control Register (FCR)**

The FCR allows selection of the FIFO trigger level (the number of bytes in FIFO required to enable the Received Data Available interrupt). In addition, the FIFOs can be cleared using this register as detailed in Table 10.

Bit	Access	Description
0	W	Ignored (Used to enable FIFOs in NS16550D). Since this UART only supports FIFO mode, this bit is ignored.
1	W	Writing a ‘1’ to bit 1 clears the Receiver FIFO and resets its logic. But it doesn’t clear the shift register, i.e. receiving of the current character continues.
2	W	Writing a ‘1’ to bit 2 clears the Transmitter FIFO and resets its logic. The shift register is not cleared, i.e. transmitting of the current character continues.
5 - 3	W	Ignored
7 - 6	W	Define the Receiver FIFO Interrupt trigger level ‘00’ – 1 byte ‘01’ – 4 bytes ‘10’ – 8 bytes ‘11’ – 14 bytes

Table 10: FCR Register

Reset Value : 11000000b

- **Line Control Register (LCR)**

The line control register allows the specification of the format of the asynchronous data communication used. A bit in the register also allows access to the Divisor Latches, which define the baud rate. Reading from the register is allowed to check the current settings of the communication as detailed in Table 11.

Bit	Access	Description
1 - 0	RW	Select number of bits in each character ‘00’ – 5 bits ‘01’ – 6 bits ‘10’ – 7 bits ‘11’ – 8 bits

2	RW	Specify the number of generated stop bits ‘0’ – 1 stop bit ‘1’ – 1.5 stop bits when 5-bit character length selected and 2 bits otherwise Note that the receiver always checks the first stop bit only.
3	RW	Parity Enable ‘0’– No parity ‘1’– Parity bit is generated on each outgoing character and is checked on each incoming one.
4	RW	Even Parity select ‘0’– Odd number of ‘1’ is transmitted and checked in each word (data and parity combined). In other words, if the data has an even number of ‘1’ in it, then the parity bit is ‘1’. ‘1’ – Even number of ‘1’ is transmitted in each word.
5	RW	Stick Parity bit. ‘0’ – Stick Parity disabled ‘1’ – If bits 3 and 4 are logic ‘1’, the parity bit is transmitted and checked as logic ‘0’. If bit 3 is ‘1’ and bit 4 is ‘0’ then the parity bit is transmitted and checked as ‘1’.
6	RW	Break Control bit ‘1’ – the serial out is forced into logic ‘0’ (break state). ‘0’ – break is disabled
7	RW	Divisor Latch Access bit. ‘1’– The divisor latches can be accessed ‘0’– The normal registers are accessed

Table 11: LCR Register

Reset Value: 00000011b

- **Modem Control Register (MCR)**

The modem control register allows transferring control signals to a modem connected to the UART detailed in Table 12.

Bit	Access	Description
0	W	Data Terminal Ready (DTR) signal control ‘0’ – DTR is ‘1’ ‘1’ – DTR is ‘0’
1	W	Request To Send (RTS) signal control ‘0’ – RTS is ‘1’ ‘1’ – RTS is ‘0’
2	W	Out1. In loopback mode, connected Ring Indicator (RI) signal input

3	W	Out2. In loopback mode, connected to Data Carrier Detect (DCD) input.
4	W	Loopback mode ‘0’ – normal operation ‘1’ – loopback mode. When in loopback mode, the Serial Output Signal (STX$_PAD_O$)*issettologic*‘1’*.Thesignalofthe* transmitter shift register is internally connected to the input of the receiver shift register. The followingconnections are made: DTR -> DSR RTS -> CTS Out1 -> RI Out2 -> DCD
7 - 5	W	Ignored

Table 12: MCR Register

Reset Value: 0

- **Line Status Register (LSR)**

Table 13 details the usage of Line Status Register.

Bit	Access	Description
0	R	Data Ready (DR) indicator. ‘0’ – No characters in the FIFO ‘1’ – At least one character has been received and is in the FIFO.
1	R	Overrun Error (OE) indicator ‘1’– If the FIFO is full and another character has been received in the receiver shift register. If another character is starting to arrive, it will overwrite the data in the shift register but the FIFO will remain intact. The bit is cleared upon reading from the register. Generates Receiver Line Status interrupt. ‘0’– No overrun state
2	R	Parity Error (PE) indicator ‘1’– The character that is currently at the top of the FIFO has been received with parity error. The bit is cleared upon reading from the register. Generates Receiver Line Status interrupt. ‘0’– No parity error in the current character

3	R	Framing Error (FE) indicator '1'– The received character at the top of the FIFO did not have a valid stop bit. Of course, generally, it might be that all the Following data is corrupt. The bit is cleared upon reading from the register. Generates Receiver Line Status interrupt. '0' – No framing error in the current character
4	R	Break Interrupt (BI) indicator '1'–A break condition has been reached in the current character. The break occurs when the line is held in logic 0 for a time of one character (start bit + data + parity + stop bit). In that case, one zero character enters the FIFO and the UART waits for a valid start bit to receive next character. The bit is cleared upon Reading from the register. Generates Receiver Line Status interrupt. '0'– No break condition in the current character
5	R	Transmit FIFO is empty. '1'– The transmitter FIFO is empty. Generates Transmitter Holding Register Empty interrupt. The bit is cleared when data is being been written to the transmitter FIFO. '0'– Otherwise
6	R	Transmitter Empty indicator. '1' – Both the transmitter FIFO and transmitter shift register are empty. The bit is cleared when data is being been written to the transmitter FIFO. '0' – Otherwise
7	R	'1'– At least one parity error, framing error or break indications have been received and are inside the FIFO. The bit is cleared upon reading from the register. '0' – Otherwise.

Table 13: LSR Register

- **Modem Status Register**

  The register displays the current state of the modem control lines. Also, four bits also provide an indication in the state of one of the modem status lines. These bits are set to '1' when a change in corresponding line has been detected and they are reset when the register is being read as detailed in Table 14.

Bit	Access	Description
0	R	Delta Clear To Send (DCTS) indicator '1' – The CTS line has changed its state.
1	R	Delta Data Set Ready (DDSR) indicator '1'– The DSR line has changed its state.

2	R	Trailing Edge of Ring Indicator (TERI) detector. The RI line has changed its state from low to high state.
3	R	Delta Data Carrier Detect (DDCD) indicator ‘1’– The DCD line has changed its state.
4	R	Complement of the CTS input or equals to RTS in loopback mode.
5	R	Complement of the DSR input or equals to DTR in loopback mode.
6	R	Complement of the RI input or equals to Out1 in loopback mode.
7	R	Complement of the DCD input or equals to Out2 in loopback mode.

Table 14: MSR Register

- **Divisor Latches**

  The divisor latches can be accessed by setting the 7th bit of LCR to ‘1’. You should restore this bit to ‘0’ after setting the divisor latches in order to restore access to the other registers that occupy the same addresses. The 2 bytes form one 16-bit register, which is internally accessed as a single number. You should therefore set all 2 bytes of the register to ensure normal operation. The register is set to the default value of 0 on reset, which disables all serial I/O operations in order to ensure explicit setup of the register in the software. The value set should be equal to (system clock speed) / (16 x desired baud rate). The internal counter starts to work when the LSB of DL is written, so when setting the divisor, write the MSB first and the LSB last.

More details on the UART 16550 IP may be accessed from here.

Operation

Upon reset the core performs the following tasks:

- The receiver and transmitter FIFOs are cleared.
- The receiver and transmitter shift registers are cleared.
- The Divisor Latch register is set to 0.
- The Line Control Register is set to communication of 8 bits of data, no parity, 1 stop bit.
- All interrupts are disabled in the Interrupt Enable Register.

For proper operation, perform the following:

- Set the Line Control Register to the desired line control parameters. Set bit 7 to ‘1’ to allow access to the Divisor Latches.
- Set the Divisor Latches, MSB first, LSB next.
- Set bit 7 of LCR to ‘0’ to disable access to Divisor Latches. At this time the transmission engine starts working and data can be sent and received.

- Set the FIFO trigger level. Generally, higher trigger level values produce less interrupt to the system, so setting it to 14 bytes is recommended if the system responds fast enough.
- Enable desired interrupts by setting appropriate bits in the Interrupt Enable register.

Design Flow

IP Customization and Generation

AXIL UART IP core is a part of the Raptor Design Suite Software. A customized AXIL UART can be generated from the Raptor's IP configurator window as shown in Figure 3.

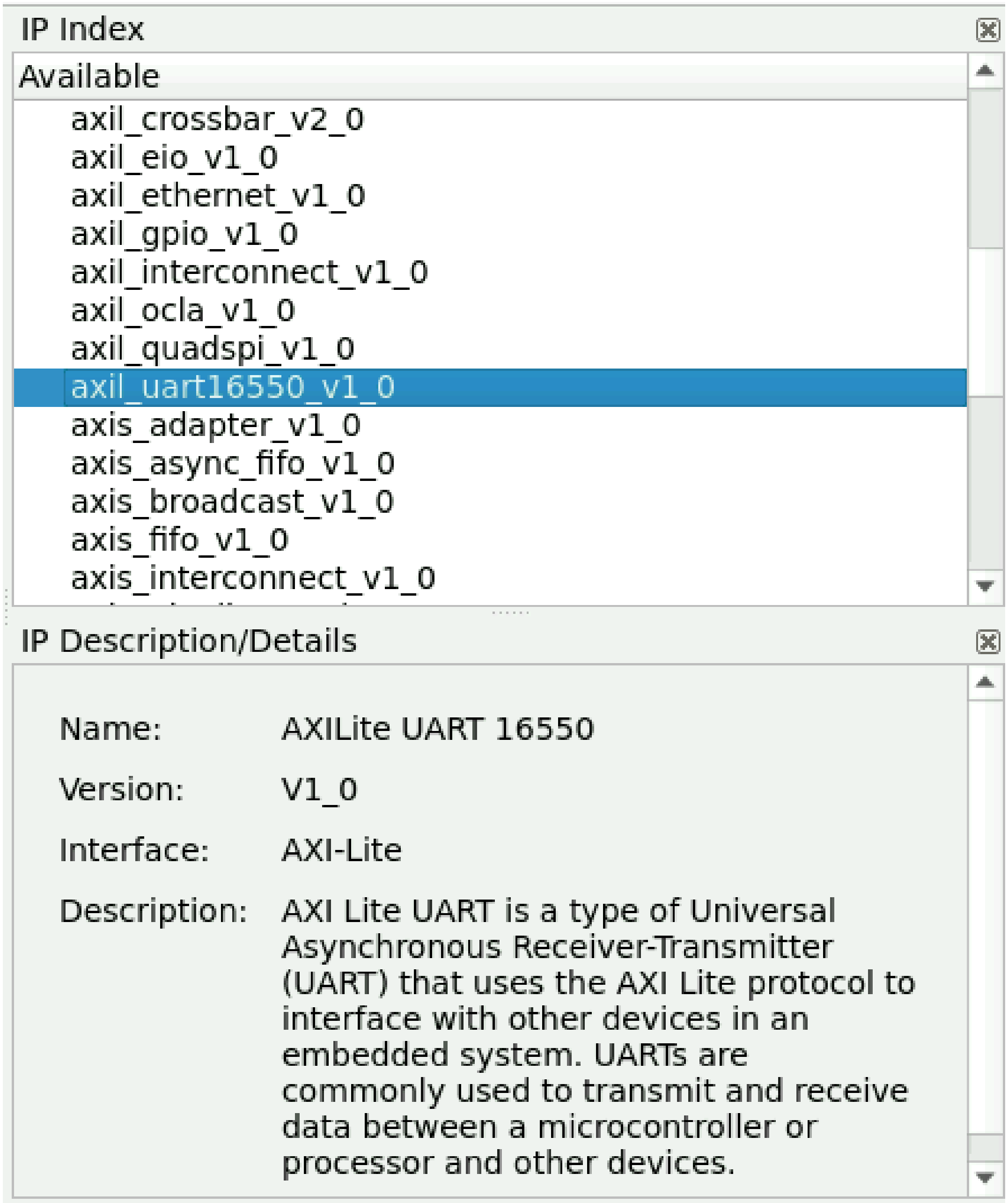

Figure 3: IP list

Parameters Customization

From the IP configuration window, the parameters of the UART can be configured and UART features can be enabled for generating a customized UART IP core that suits the user application requirement as shown in Figure 4. After IP Customization, all the source files are made available to the user with a top wrapper that instantiates a parameterized instance of the AXIL UART.

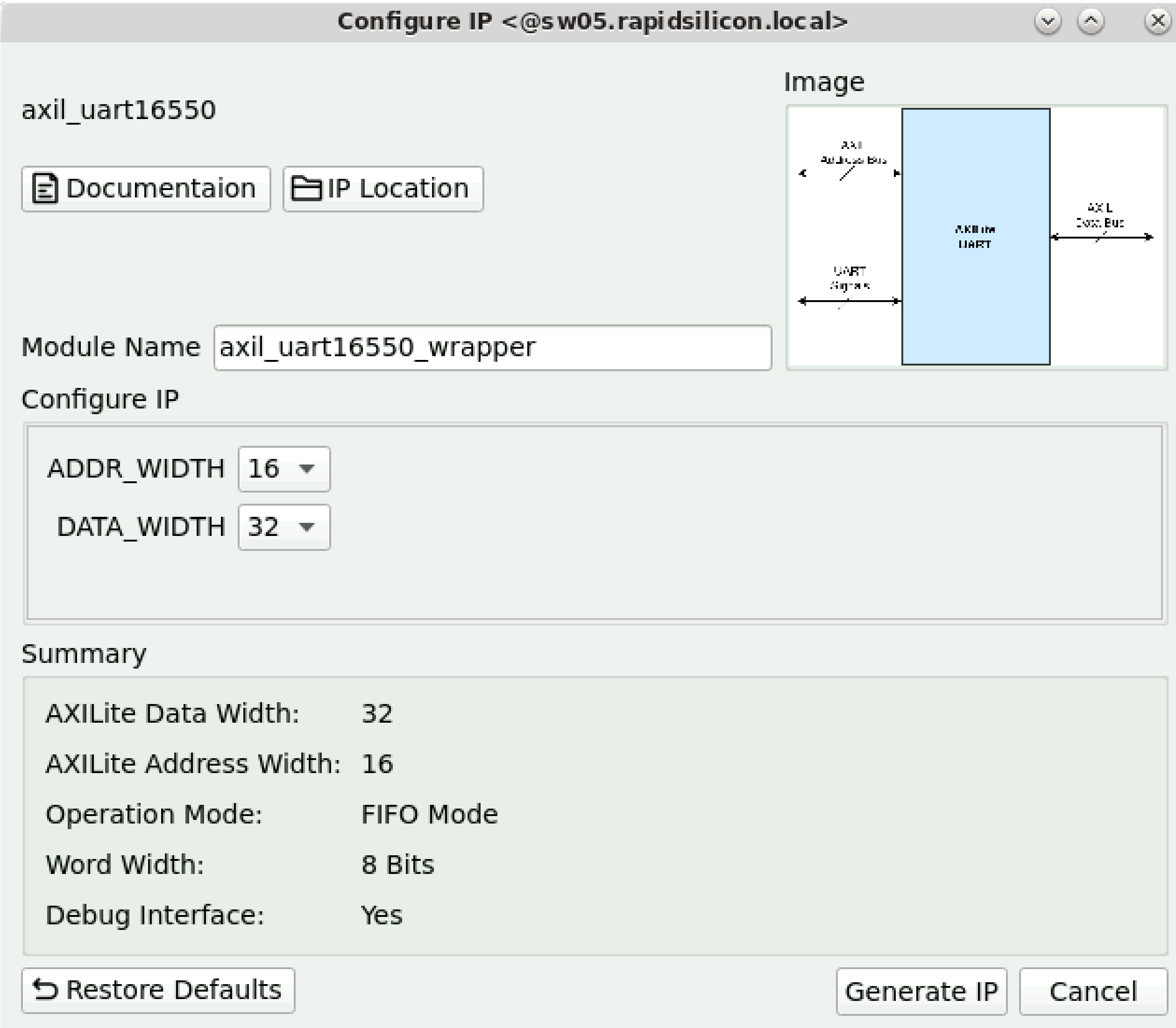

Figure 4: IP Configuration

Example Design

Overview

This AXIL UART IP can be utilized in a system that requires sequential transmission and reception of data from the outside world. UART is a crucial component in many electronic systems, enabling communication between the system and external devices through a serial interface. It can be embedded inside SoCs to enable two-way communication via the SoC. One such example design of this AXIL UART can be visualized in Figure 5.

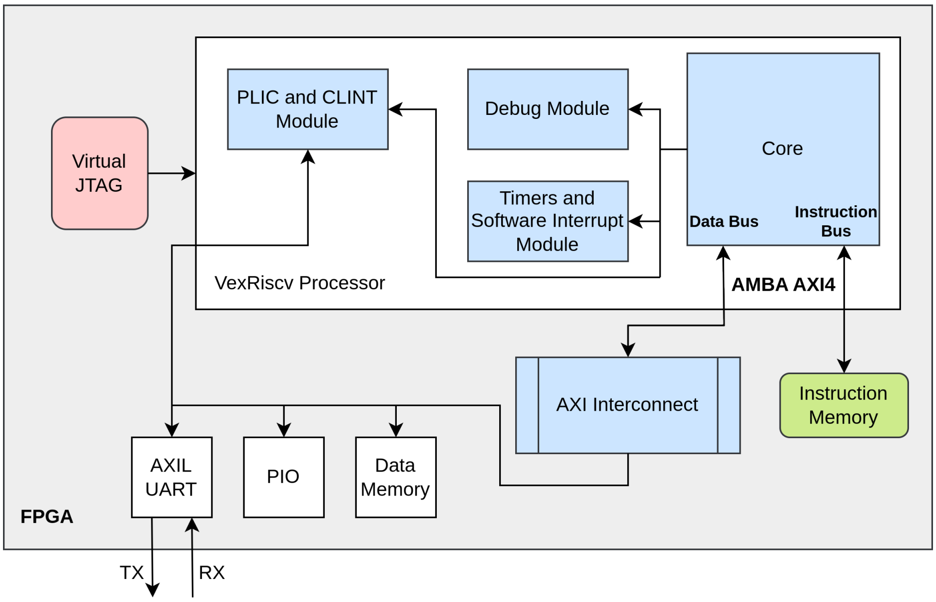

Figure 5: AXIL UART inside and SoC

Simulating the Example Design

The IP being Verilog HDL, can be simulated via a bunch of industry standard stimulus. For instance, it could be simulated via writing a Verilog Test-bench, or incorporating a soft processor that can stimulate this UART. The bundled example design is stimulated via an SoC based design as pictured above via any simulator of choice. The testplan is written in C or Assembly and fed to the SoC in a hex format, which then stimulates the UART IP connected within the SoC in a loopback fashion so both the RX and TX streams can be compared and verified based on the generated interrupts that are handled by the PLIC controller on the CPU tile. In this way, most of the UART IP is covered in testing.

Synthesis and PR

Raptor Suite is armed with tools for Synthesis along with Post and Route capabilities and the generated post-synthesis and post-route and place net-lists can be viewed and analyzed from within the Raptor. The generated bit-stream can then be uploaded on an FPGA device to be utilized in hardware applications.

Test Bench

The AXIL UART is simulated via incorporating it in an SoC. The SoC is booted up via writing a bare-metal firmware in C / Assembly. The testbench for this UART AXIL is incorporating inside this bare-metal firmware in a loopback fashion to make sure that the received data is the same as the one that was transmitted. The AXIL UART also generates different types of interrupts that are handled by writing an ISR in Assembly that handles the interrupts. This firmware is then loaded onto the SoC and the UART starts its operation. The clock and reset is given externally via a Verilog testbench file. The bare metal testbench can be enhanced to cover different types of UART operations making sure all the UART registers are getting hit by the test, ensuring complete coverage and the usability of the UART by integration with other AXI based systems and peripherals.

Release

Release History

Date	Version	Revisions
November 14, 2023	0.1	Initial version AXI4-Lite UART User Guide Document

List of Abbreviations

Abbreviation	Definition
R	Read Only
W	Write Only
RW	Read and Write
UART	Universal Asynchronous Receiver / Transmitter

Table 15: List of Abbreviations