



AXI4-Lite UART 16550 (Beta Release)

Version 1.0

April 18, 2023

Copyright

Copyright © 2021 Rapid Silicon. All rights reserved. This document may not, in whole or part, be reproduced, modified, distributed, or publicly displayed without prior written consent from Rapid Silicon ("Rapid Silicon").

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	3
Introduction	3
Features	3
Overview	4
AXIL UART	4
IP Specification	5
Standards	5
IP Support Details	5
Resource Utilization	8
Registers and Address Space	8
Operation	8
Design Flow	10
IP Customization and Generation	10
Parameters Customization	11
Example Design	12
Overview	12
Simulating the Example Design	12
Synthesis and PR	12
Test Bench	13
Release	14
Release History	14

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 an integrated circuit 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.

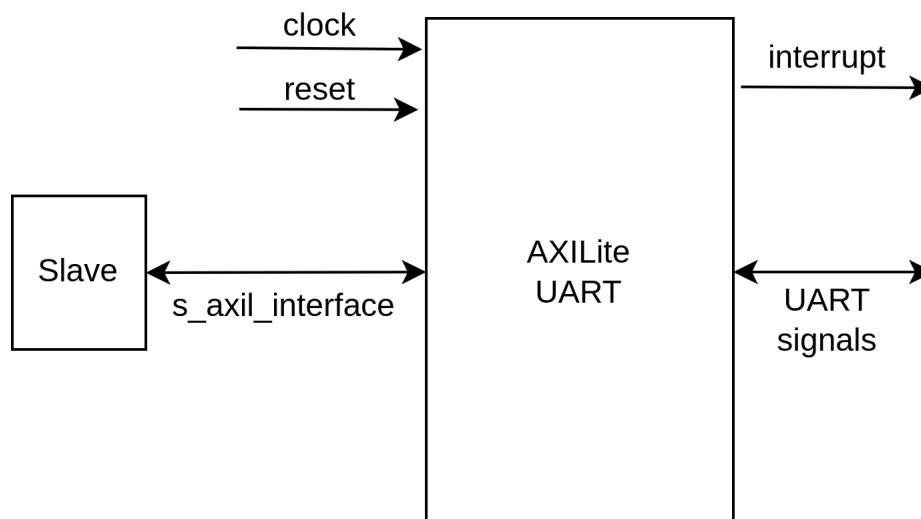


Figure 1: AXIL UART Block Diagram

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 File	Testbench	Simulation Model	Software Driver	Analyze and Elaboration	Simulation	Synthesis
GEMINI	AXI4 Lite	Verilog	SDC	Python / C	SoC / LiteX	Icarus	Raptor	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	Address Width for UART
DATA WIDTH	8 / 16 / 32 / 64	32	Data Width for UART

Table 2: Parameters

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_aclk	I	Active Low Reset
Write Address Channel		
s_axi_awvalid	I	AXI4-Lite data
s_axi_awaddr	I	AXI4-Lite keep data qualifier
s_axi_awprot	I	AXI4-Lite valid transfer
s_axi_awready	O	AXI4-Lite transfer ready
Write Data Channel		
s_axi_wvalid	I	AXI4-Lite boundary of transfer packet
s_axi_wdata	I	AXI4-Lite data stream identifier
s_axi_wstrb	I	AXI4-Lite data routing information
s_axi_wready	O	AXI4-Lite user defined sideband information
Write Response Channel		
s_axi_bvalid	O	AXI4-Lite data
s_axi_bresp	O	AXI4-Lite keep data qualifier
s_axi_bready	I	AXI4-Lite valid transfer
Read Address Channel		
s_axi_arravlid	I	AXI4-Lite transfer ready
s_axi_araddr	I	AXI4-Lite boundary of transfer packet
s_axi_arprot	I	AXI4-Lite data stream identifier
s_axi_arready	O	AXI4-Lite data routing information
Read Data Channel		
s_axi_rvalid	O	AXI4-Lite user defined sideband information
s_axi_rdata	O	AXI4-Lite user defined sideband information
s_axi_rresp	O	AXI4-Lite user defined sideband information
s_axi_rready	I	AXI4-Lite user defined sideband information
UART Signals		
int_o	O	Interrupt output
srx_pad_i	I	Serial output signal
stx_pad_o	O	Serial input signal
rts_pad_o	O	Request To Send
cts_pad_i	I	Data Terminal Ready
dtr_pad_o	O	Clear To Send
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

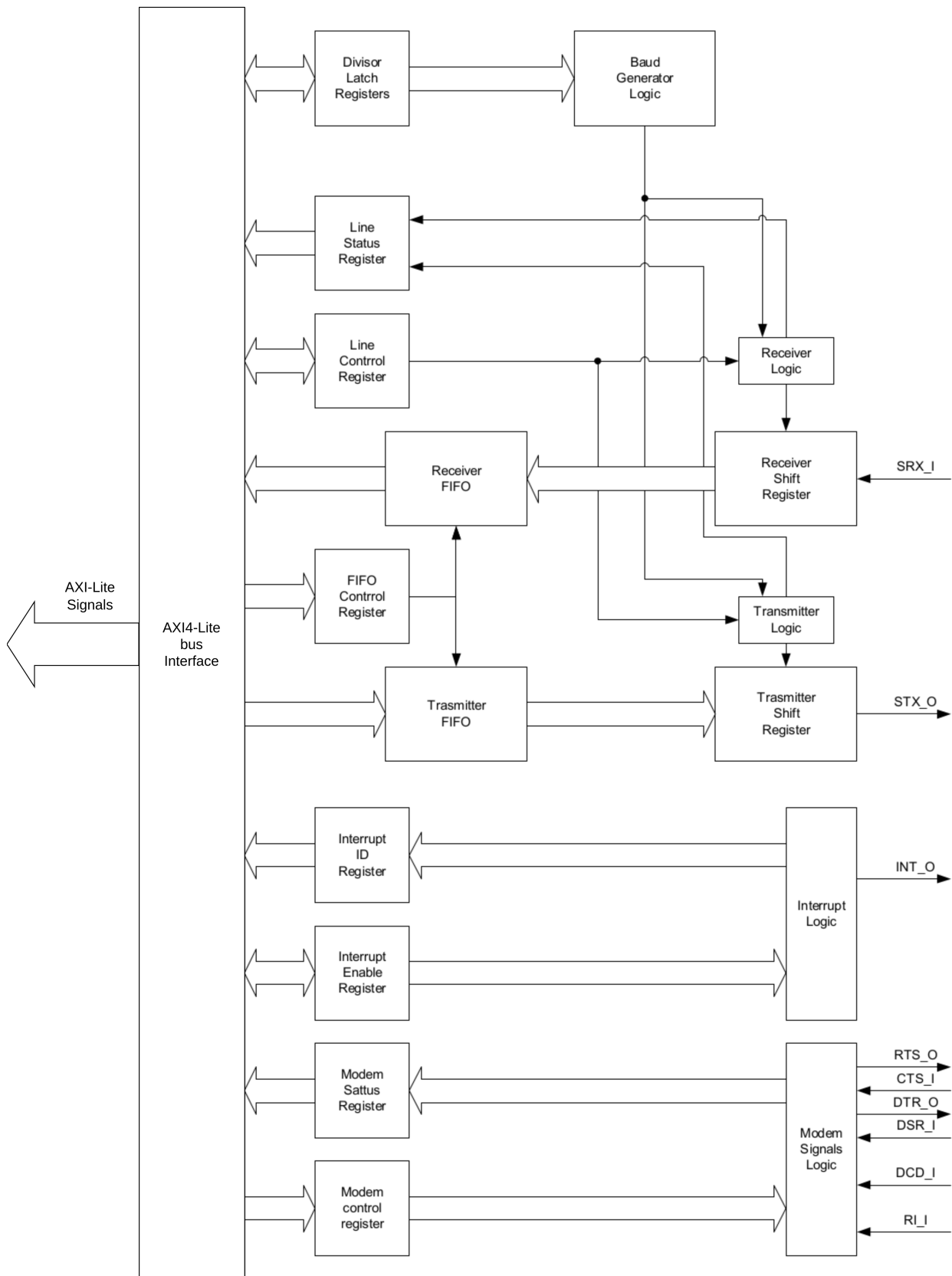


Figure 2: AXIL UART Internal Diagram

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	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

More information on the individual registers and how to use them can be accessed [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.



Configure IP [X]

Configuring axil_uart16550 (v1.0) from rapidsilicon's ip library

Parameters

ADDR_WIDTH 16 ▼

DATA_WIDTH 32 ▼

Output

Module Name axil_uart16550_wrapper

Output Dir axil_uart16550_wrapper

Generate IP

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.

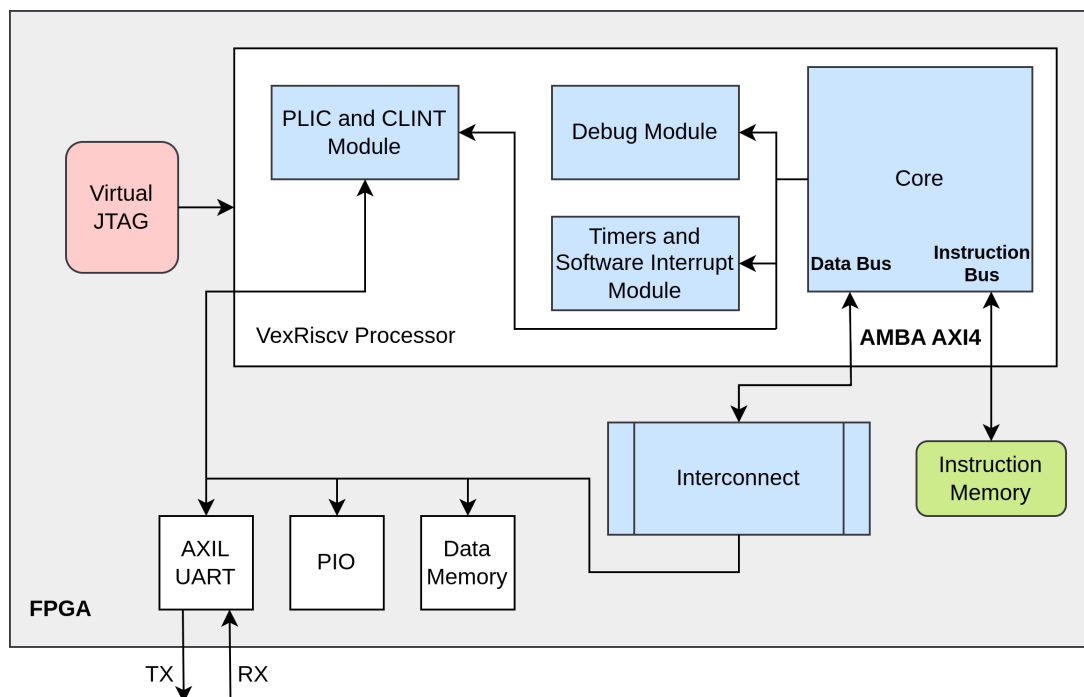


Figure 5: AXIL UART inside and SoC

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 a Coco-tb based environment that iteratively stimulates all the master/slave pairs while also stress testing the data routing between them.

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
April 18, 2023	0.01	Initial version AXI4-Lite UART User Guide Document