



wave_gen Design Overview

wave_gen Design Description

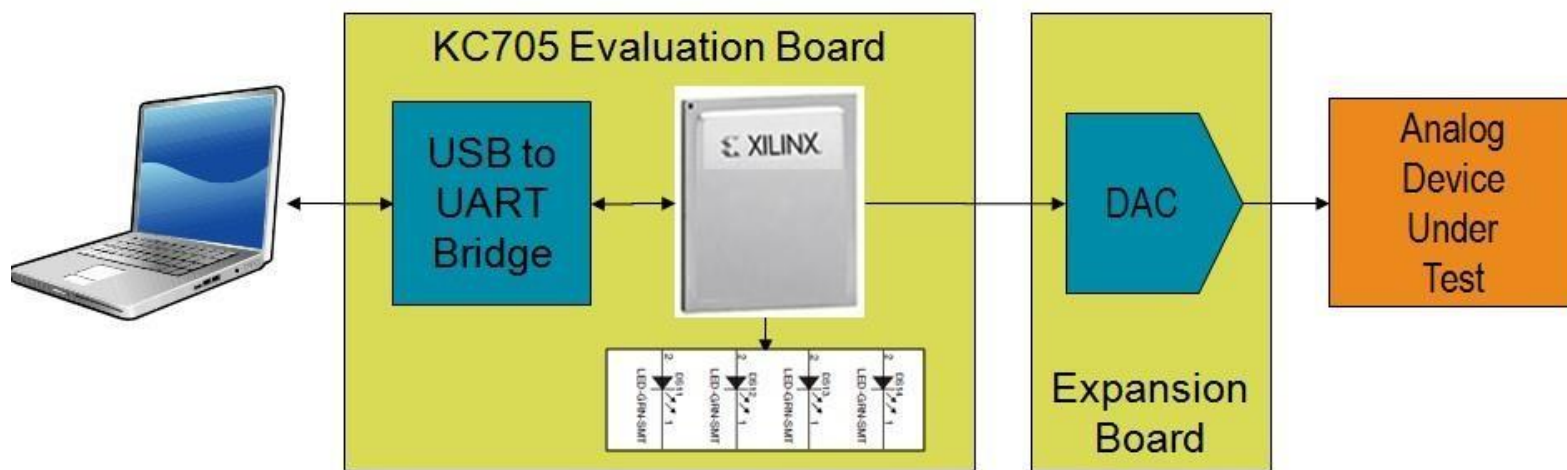


- **wave_gen Design Description**
- RS-232 Tutorial

wave_gen Lab Use

Overview

- PC sends and receives characters via USB cable
 - Uses a terminal program (such as HyperTerminal or Tera Term) and a Virtual Communication Port
- USB-to-UART bridge converts the USB protocol to RS-232 protocol
- DAC converts digital samples to analog using the Serial Peripheral Interface (SPI)
- *wave_gen* design in the FPGA does everything else
 - Processes commands and generates responses
 - Generates samples and sends them to the DAC and LEDs



wave_gen Design

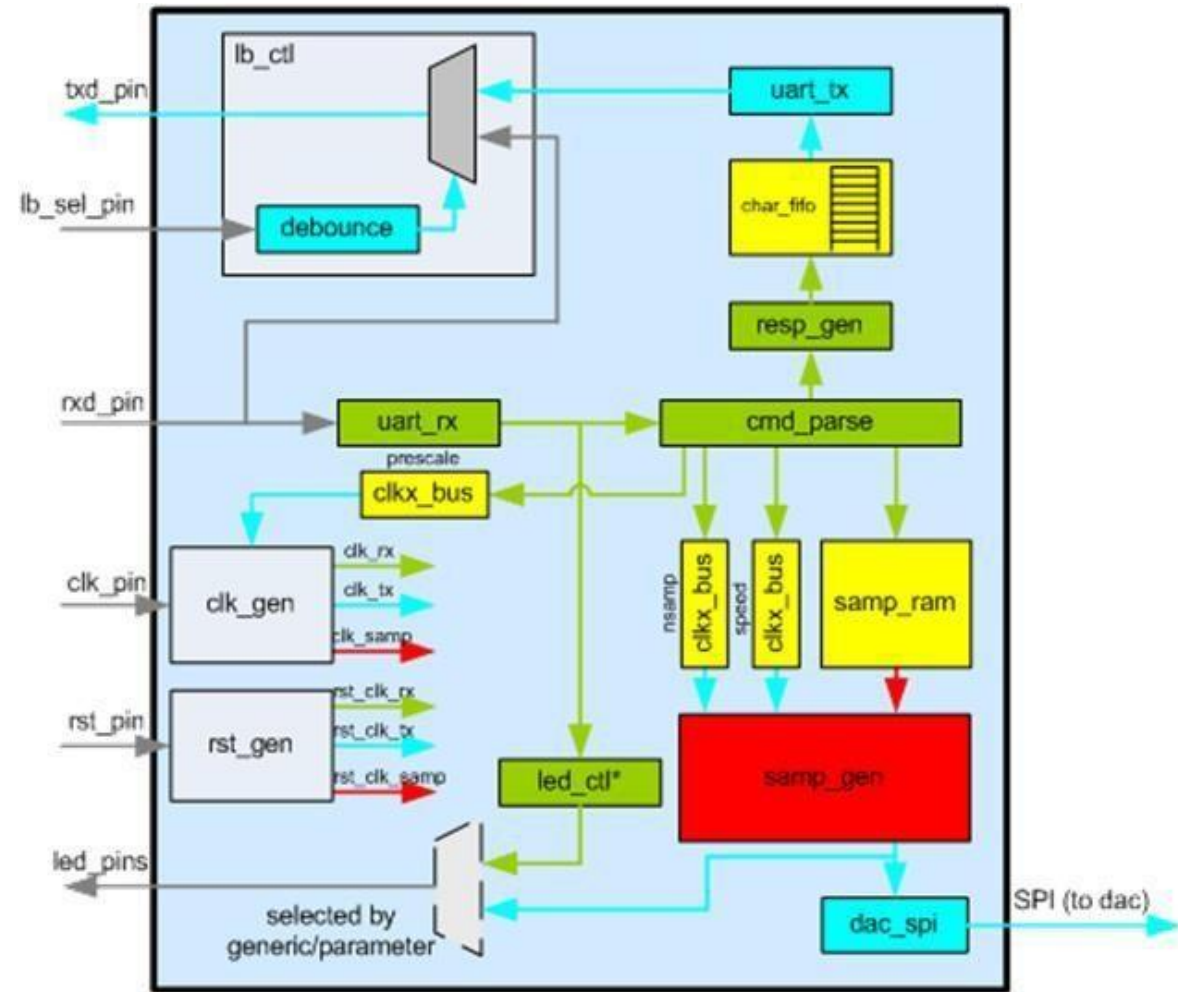
Basic Description

- This design *records* specific information via RS-232 serial communication and stores this data in an on-chip memory
- After data has been stored, it can be retrieved via the RS-232 communications channel, or *played* out via a bank of LEDs or a DAC
- Receives RS-232 serial data at 115200 baud (no parity, 8 data bits, no handshaking)
 - Handled in *uart_rx*, which is simple state machine and an over sampler
- Small command set controls how information is stored and played back

Full Project Flow Supported

- Simulation
 - A full self-checking testbench is provided in both Verilog and VHDL
- Synthesis and implementation
 - The design can be synthesized and implemented
 - Can be targeted to Artix™, Kintex™, and Virtex™ devices
- Download
 - The implemented design can be downloaded to the KC705 evaluation board
- Testing
 - The downloaded design can be verified via the USB port of a PC
 - A terminal program (such as HyperTerminal, Tera Term) can be used to send commands and receive responses from the design using a Virtual COM Port driver
 - A USB-to-UART bridge converts USB to RS-232
 - The samples being sent can be visually verified on the LEDs

Design Block Diagram



Clock Domains in wave_gen

- The *wave_gen* design uses three clock domains
 - Receive clock (*clk_rx*)
 - Transmit clock (*clk_tx*)
 - Sample clock (*clk_samp*)
- All clocks are derived from a single clock input pin (*clk_pin*) using a single MMCM
 - The clock input depends on the board
 - For Kintex-7 development boards, the clock is differential and runs at 200 MHz
- The receive clock runs at the input clock frequency (200 MHz)
 - The receive clock is assumed to be asynchronous to the transmit and sample clocks
- The transmit clock runs either at the same frequency or at 31/32 of that frequency
- The sample clock is a decimated version of the transmit clock
 - Rate is determined by the *prescale* value

wave_gen Commands

Cmd	Input	Response	Description
*W	aaaavvvv	-OK or -ERR	03ff ≥ aaaa ≥ 0000. Value “vvvv” is written into RAM at location “aaaa” and “-OK” is return.
*R	aaaa	-hhhh dddd or -ERR	03ff ≥ aaaa ≥ 0000. If in range, then the value at “aaaa” is returned in hex and decimal.
*N	vvvv	-OK or -ERR	0400 ≥ vvvv ≥ 0001. Specifies the number of samples before recycling.
*P	vvvv	-OK or -ERR	ffff ≥ vvvv ≥ 0020. Specifies prescaling value to divide <i>clk_tx</i> by to produce <i>clk_samp</i> .
*S	vvvv	-OK or -ERR	ffff ≥ vvvv ≥ 0001. Specifies “speed” value to divide <i>clk_samp</i> by to produce the rate of read from RAM.
*n/*p/*s		-hhhh dddd	Returns current value of nsamp, prescale, and speed.
*G		-OK	Triggers a single pass through nsamp memory locations.
*C		-OK	Starts continuous triggering.
*H		-OK	Halts continuous loop at end of current cycle.

RS-232 Tutorial



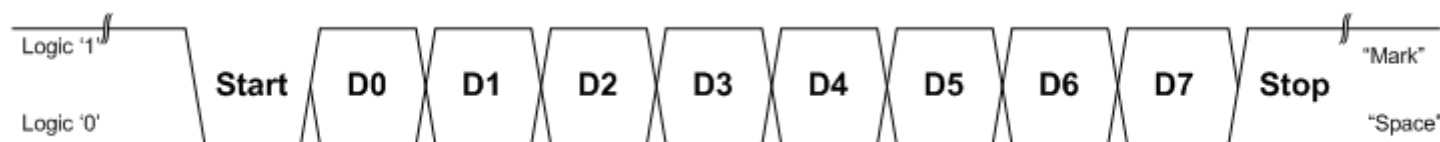
- wave_gen Design Description
- **RS-232 Tutorial**

Introduction to RS-232-C

- RS-232-C is a standard issued by the Electronics Industry Association (EIA)
 - Used since the late 1960s for communication between Data Terminal Equipment (DTE) and Data Communication Equipment (DCE)
- Allows communication between DTEs and DCEs over a short range at relatively high speeds*
- Uses very few wires
 - As little as a common ground, and one data wire in each direction
- Voltage levels, slew rates, cables, and connectors are all covered by the standard
- While not technically covered by the standard, commonly used transmission rates, framing, and character coding is usually considered part of RS-232
- Still available on most modern computers
 - Simply referred to as the *serial port* or *COM port*
 - USB to UART bridge for PCs that lack an RS-232 serial port

RS-232-C Protocol

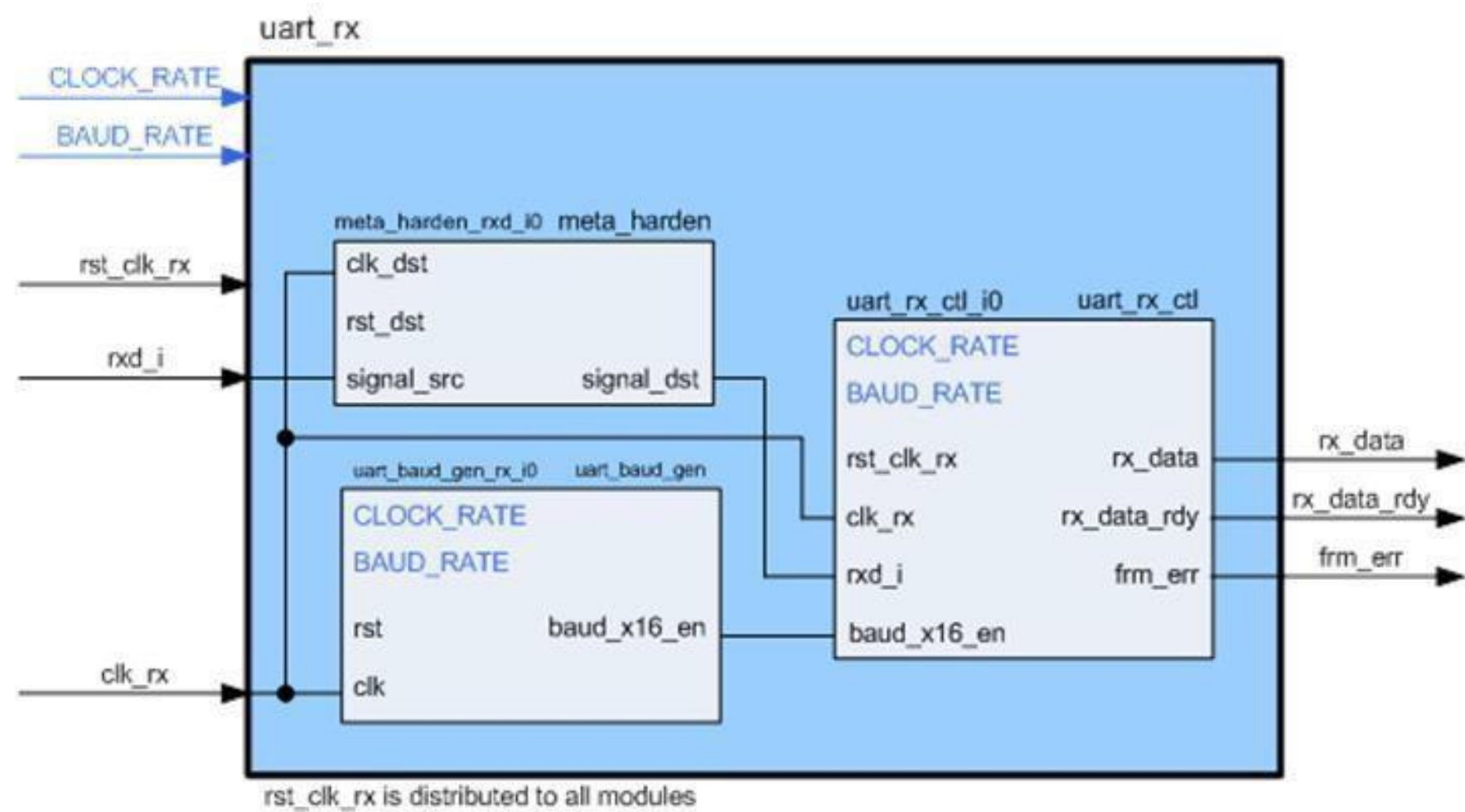
- Data communication is character oriented and serially transmitted
 - A character is typically an 8-bit byte
- Characters are usually encoded, using the American Standard Code for Information Interchange (ASCII)
- Framing is usually accomplished, using a single START bit before the character and a single STOP bit after the character
 - The character is sent with the least-significant bit first
- The line is driven high (or *Mark*) when idle
 - A START bit drives the line low (or *Space*) and the STOP bit is a Mark



RS-232-C Timing

- RS-232-C communication is asynchronous
 - No clock is sent along with the data
- The equipment on both ends must be set to the same baud rate
 - The baud rate is the number of symbols per second sent on the line
 - Each symbol is one bit; hence, the baud rate is the number of bits per second
- Each bit (START, DATA, STOP) is sent on the line for the same amount of time
 - One bit period is $1 / \text{BAUD_RATE}$
- The transmitter and receiver have independent time references
 - The standard allows for some difference between the local references

uart_rx Module



GENERAL DISCLOSURE AND ATTRIBUTION STATEMENT

The information contained herein is for informational purposes only and is subject to change without notice. While every precaution has been taken in the preparation of this material, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this material, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD's products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale. GD-18.

©2023 Advanced Micro Devices, Inc. All rights reserved. AMD, the AMD Arrow logo, Artix, Kintex, and Virtex, and combinations thereof are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective owners.

