MemecCore™ Product Line
3721 Valley Centre Drive
San Diego, CA 92130 USA
Americas: +1 800-752-3040
Europe: +41 (0) 32 374 32 00
Asia: +(852) 2410 2720
E-mail: actel.info@memecdesign.com
URL: www.memecdesign.com/actel

Product Summary

Intended Use

- Industry Standard Serial Interfaces
- System Peripherals
- Debug/Maintenance Interfaces

Key Features

- Baudrate Synthesizer for any baudrate up to 1/10 of clock speed
- No dedicated clock frequency
- 7 or 8 Bits Data
- No/Odd/Even Parity
- Error Detection
- 1 or 2 Stop Bits
- Format Check
- 3-Point Input Sampling
- Parallel Interface with Event Control

Targeted Devices

- SX-A Family
- Axcelerator Family
- ProASIC^{PLUS} Family

General Description

The MC-ACT-UARTF core is generally used as a data link layer with parallel interfaces and event communication. Microprocessor specific interfaces are built around the MC-ACT-UARTF, as well queues, interrupt controllers and status reporting circuits[1].

[1] For complete UART solutions, please contact Memec Design.

Core Deliverables

- Netlist Version
  - Netlist compatible with the Actel Designer place and route tool
- RTL Version
  - VHDL Source Code
  - Test Bench
- All
  - User Guide

Synthesis and Simulation Support

- Synthesis: Synplicity
- Simulation: ModelSim
- Other tools supported upon request

Verification

- Test Bench

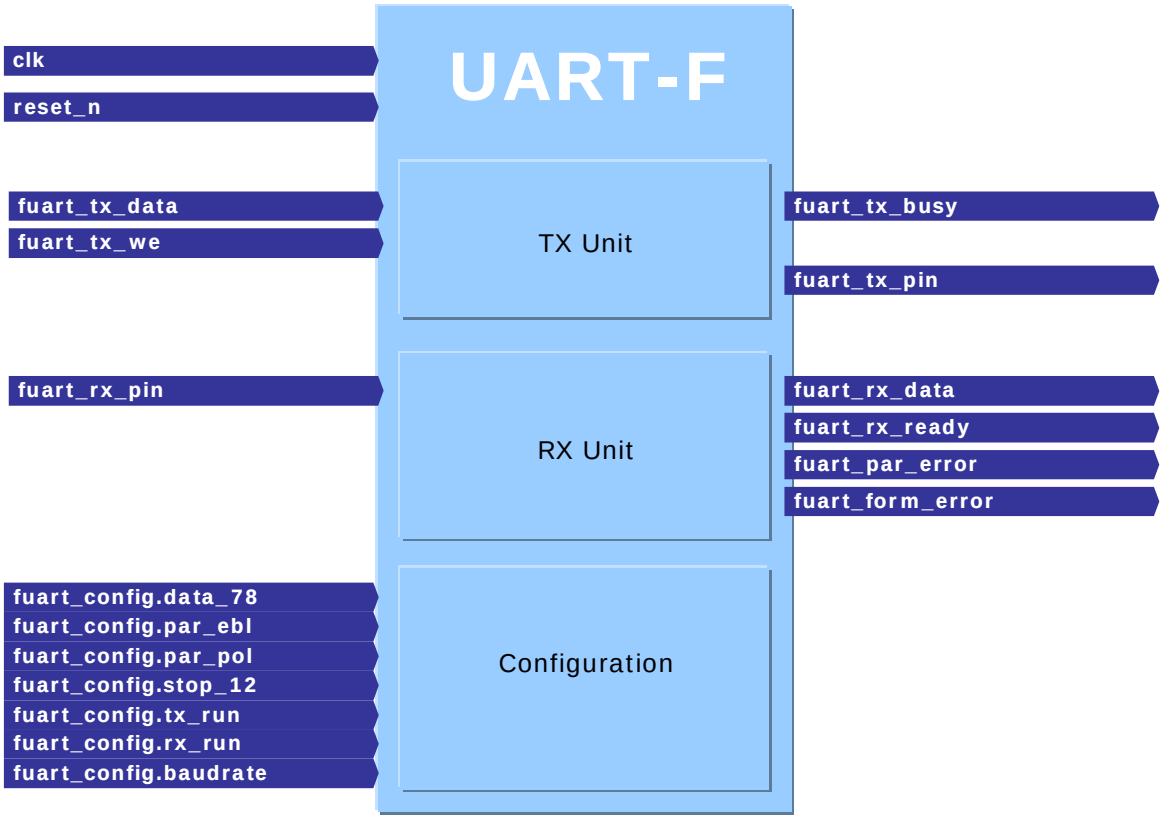

Figure 1:Block Diagram

Functional Description

Structure of UART-F

The structure of MC-ACT-UARTF consists of four principal block. A dual advanced high speed baudrate synthesizer is implemented to serve as receiver sampling source and transmitter clock source. Clock generation is configured through 16-bit register value.

Format analyzer detects the incoming data stream to sequence bits stored and interface by receive unit. The detection and derivation of data stream enables a bit sampling mechanism to reduce necessary clock cycle in receiving unit.

Transmit unit is starting serial out after a parallel event controlled interface is used.

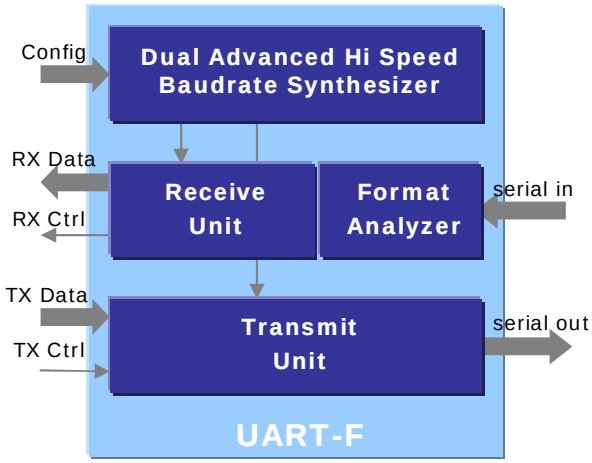

Figure 2: UART-F Structure

Event Communication

For communicating events, the MC-ACT-UARTF core uses or produces active '1' pulses, which are activated for only one clk cycle. In the inactive state, they remain low with respect to the rising clk edge, so glitches may occur. For communicating over clock domains, these events must be synchronized first!

Configuration

The configuration pins are used to set the bitrate, bit timing and output format. They're static inputs and used for both receiver and transmitter in common.

Baudrate

The baudrate generator is not a simple prescaler, but allows generating all baudrates from the system clock within a certain range. There is no special clock frequency needed for that purpose so that you're free to choose the system clock for the MC-ACT-UARTF, which simplifies considerably the clock structure.

To configurate the baudrate a 16bit configuration register value is used.

Examples:

For 1Mhz clock and 115'200bps, n is 30198(dec), accuracy better than 33ppm,

For 20Mhz clock and 1Mbps, n is 13107(dec), accuracy better than 76ppm.

Limitations:

Values for n lower than 100(dec) should not be used, otherwise the accuracy may be below 1%.

e.g. for 1MHz clock, the possible baudrates with accuracy better than 1% range from 381bps to 250kbps.

Accuracy:

The worst case accuracy can be calculated by simply inversing the value n. Therefore, a value larger than 100 will guarantee accuracy better than 1%, values larger than 1000 produce results better than 0.1%.

Jitter:

The faster the baudrate, the better the accuracy, but more relative jitter is added. Maximum absolute jitter is always equal 1/fclk.

Serial Interface

The serial interface includes the receive and transmit path separately. It is full a duplex solution, receive and transmit is possible at the same time.

Transmitter Interface

For transmitting data, a parallel event controlled interface is used. It is an efficient way to embed the MC-ACT-UARTF in systems as well as connecting simple or complex specific interfaces, including queues etc., to it.

The transmitter path stores the incoming byte in the shift register by means of the fuart_tx_we signal and starts the transmitting activity. fuart_tx_busy goes high also and remains high until the data is sent.

Receiver Interface

For receiving data, a similar type of interface is used as in the transmitter path

The receiver path contains several checks and special features. First, the level at the fuart_rx_pin is watched. When a falling edge is detected, the receiver is started. A reception is started only when the start bit after a falling edge is detected low. If parity is enabled, it is checked and event failures are reported on fuart_par_error. Missing stop bits (level not zero) are reported as format checks. In all error cases, the data byte is aborted and the error reason is reported. Please note that fuart_rx_ready is not asserted when error reporting is done.

Device Requirements

Family	Device	Utilization			Performance
		COMB	SEQ	Total	
SX-A	SX08A-3	274 (54%)	104 (41%)	378 (50%)	142 MHz
ProASIC^{PLUS}	APA075-STD	n/a	n/a	659 (22%)	70 MHz
Axcelerator	AX500-3	270 (5%)	104 (4%)	374 (5%)	153 MHz

Table 1: Device Utilization and Performance

Verification and Compliance

Complete functional and timing simulation has been performed on the UART-F using ModelSim 5.5d. The UART-F core has been used successfully in customer designs.

Signal Descriptions

The following signal descriptions define the IO signals.

Signal	Direction	Description
clk	in	**System clock**, rising edge used only, must be at least 64 times higher than maximum baudrate
reset_n	in	**Asynchronous system reset**, active low, goes to all flip flops
fuart_config.baudrate[15:0]	in	**Baudrate** configuration value
fuart_config.data_78	in	**Transmit and receive data size**: '0': use 7 bit data '1': use 8 bit data
fuart_config.par_ebl	in	**Parity enable**: '0': no parity check, no parity bit transmitted and received '1': use parity check, parity bit inserted and checked
fuart_config.par_pol	in	**Parity polarity**: '0': use even parity[2] '1': use odd parity This parameter is ignored when par_ebl is inactive!
fuart_config.stop_12	in	Transmit and receive **stop bit number**: '0': use and check 1 stop bit '1': use and check 2 stop bits
fuart_config.tx_run	in	**Transmit control**: '0': transmitter off, ignores all inputs, outputs inactive '1': transmitter is working
fuart_config.rx_run	in	**Receive control**: '0': receiver off, ignores all inputs, outputs are inactive '1': receiver is working
rx_pin	in	Pin for the **incoming bit stream**. The inactive state is logic '1'
tx_pin	out	Pin for the **outgoing bit stream**. The inactive state is logic '1'
fuart_tx_data[7:0]	in	8bit **data to be transmitted**. For 7bit configuration, bit[7] is ignored. Data must be valid and stable when fuart_tx_we is active.
fuart_tx_we	in	**Event** for storing the **tx_data** in the transmit shift register and start of transmission. It's up to the system to not activate this input when the MC-ACT-UARTF is busy.
fuart_tx_busy	out	When the transmitter is sending a byte, this **status output** remains active (logic '1') until it is ready to send a new byte. While fuart_tx_busy is '1', fuart_tx_we mustn't be activated.
fuart_rx_data[7:0]	out	8bit **data that has been received**. For 7bit configuration, bit[7] is ignored. The data will be stable only during the active phase of fuart_rx_ready. Add a buffer register if data should remain stable until reception of next character.
fuart_rx_ready	out	**Event** (active '1') for signalling, that a **new byte has arrived** and the fuart_rx_data is valid now.
fuart_par_error	out	**Event** (active '1') for signalling, that a byte with **wrong parity** has been received and aborted (it's not visible at rx_ready) This signal is always inactive when par_ebl is deactivated.
fuart_form_error	out	**Event** (active '1') for signalling, that a byte with **wrong format** has been received and aborted (it's not visible at rx_ready)

Table 2: Core I/O Signals

[2] number of ones in a byte, including parity bit is even

Recommended Design Experience

For the source version, users should be familiar with HDL entry and Actel design flows. Users should be familiar with Actel Libero v2.2 Integrated Design Environment (IDE) and preferably with Synplify and ModelSim.

Ordering Information

Part Number	Description
MC-ACT-UARTF-NET	Core Netlist
MC-ACT-UARTF-VHD	Core VHDL

Table 3: Core Part Numbers

The CORE is provided under license from Memec Design for use in Actel programmable logic devices. Please contact Memec Design for pricing and more information.

Information furnished by Memec Design is believed to be accurate and reliable. Memec Design reserves the right to change specifications detailed in this data sheet at any time without notice, in order to improve reliability, function or design, and assumes no responsibility for any errors within this document. Memec Design does not make any commitment to update this information.

Memec Design assumes no obligation to correct any errors contained herein or to advise any user of this text of any correction, if such be made, nor does the Company assume responsibility for the functioning of undescribed features or parameters. Memec Design will not assume any liability for the accuracy or correctness of any support or assistance provided to a user.

Memec Design does not represent that products described herein are free from patent infringement or from any other third-party right. No license is granted by implication or otherwise under any patent or patent rights of Memec Design.

MemecCore products are not intended for use in life support appliances, devices, or systems. Use of a MemecCore product in such application without the written consent of the appropriate Memec Design officer is prohibited.

All trademarks, registered trademarks, or service marks are property of their respective owners.

Datasheet Revision History

Version	Date	Description
Datasheet 1.0	November 27, 2002	Initial Release
Datasheet 1.1	January 07, 2003	Performance information modified
Datasheet 1.2	January 23, 2003	Modification done in section core deliverables; Added logo to footer
Datasheet 1.3	February 25, 2003	Modification done in section device requirements, new URL and address inserted