



2. APB_UART

Simplified Universal Asynchronous Receiver/Transmitter (UART)

Description:

apb_uart is a simplified UART APB controller that performs two main tasks:

- Converts serial data received from a peripheral device into parallel data sent to the host (CPU).
- Converts parallel data received from the host into serial data sent to a peripheral device.

The host can read the UART operation status at any time.

2.1. Requirements

Supports baud rate values: 2400, 4800, 9600, 19200, 38400, 57600, 76800, 115200, 230400 (configurable via parameters).

Supports clear-to-send (CTS) and request-to-send (RTS) handshaking signals (automatic flow control).

Programmable UART frame (configured via APB registers), including:

- Number of data bits: 5 to 8
- Number of parity bits: 0 or 1
- Parity type (if used): odd or even
- Number of stop bits: 1 or 2

Operation control via the APB protocol.

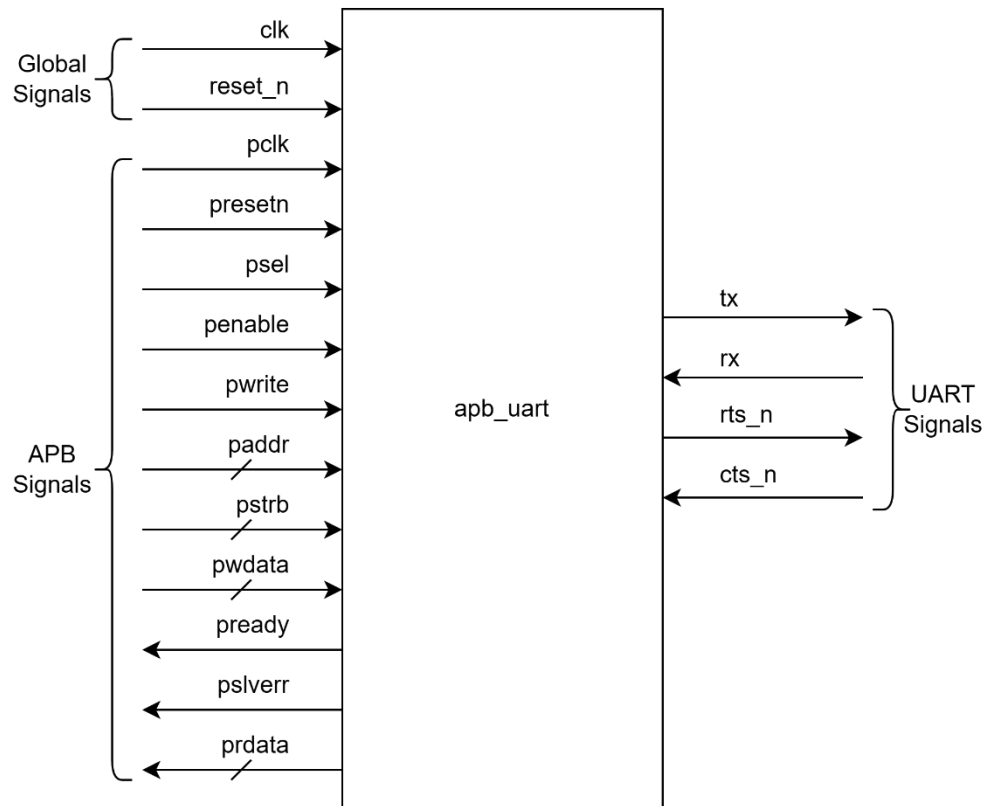


Figure 1. `apb_uart` block diagram

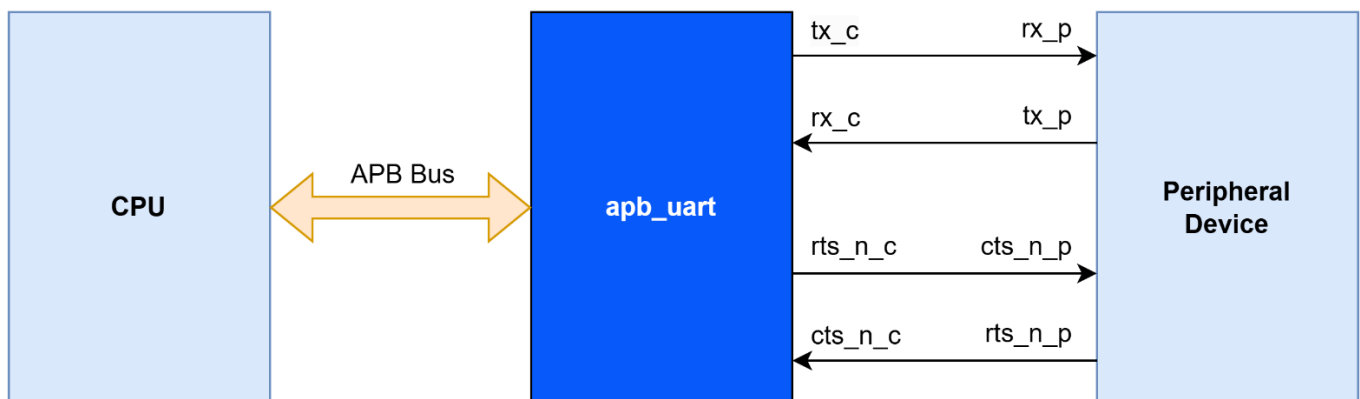


Figure 2. System using `apb_uart`

*Table 1. apb_uart parameter description*

Parameter	Value	Description
BAUDRATE	`CFG_BAUDRATE Default: 115200	Baud rate value

Table 2. apb_uart signals description

Signal Name	Width	I/O	Description
Global Signals			
clk	1	Input	System clock signal
reset_n	1	Input	System asynchronous reset, active LOW
APB Completer Signals			
pclk	1	Input	APB clock signal
presetn	1	Input	APB reset signal, active LOW, connected directly to the system reset
psel	1	Input	Select, indicates that the APB completer is selected and that a data transfer is required
penable	1	Input	Enable, indicates the second and subsequent cycles of an APB transfer
pwrite	1	Input	Direction, indicates an APB write access when HIGH and an APB read access when LOW
pstrb	4	Input	Write strobe, indicates which byte lanes to update during write transfer
paddr	12	Input	APB address bus
pwrdata	32	Input	APB write data
pready	1	Output	Ready, is used to extend an APB transfer by the completer
pslverr	1	Output	Transfer error
prdata	32	Output	APB read data
UART Signals			
rx	1	Input	Serial data receive
cts_n	1	Input	Clear-to-send handshaking signal
tx	1	Output	Serial data transmit
rts_n	1	Output	Request-to-send handshaking signal



2.2. Functional Description

apb_uart operates based on its register block, which provides information about the data to be processed, UART frame configuration, control signals, and status signals. For more details, refer to **Table 3**.

The host can control and monitor the module's operation by accessing its register block via the APB protocol. **apb_uart** uses the values programmed into its registers by the host to execute the appropriate data transmission/reception process with a peripheral device through the UART protocol.

Each UART transaction transmits or receives one character, where the bit width is equal to the configured number of data bits. **Figure 2** illustrates how **apb_uart**, the peripheral device, and the host are connected. The control flows of the module are shown in **Figures 3 and 4**.

Note:

- System clock signal and APB clock signal would use the same frequency.
- In case the number of data bits is smaller than **tx_data/rx_data** bitwidth, most significant bits of **tx_data/rx_data** would be masked. For example, when the number of data bits is 5, only **tx_data[4:0]/rx_data[4:0]** are valid.



2.3. Registers Block

Table 3. *apb_uart's register block description*

Register Name	Register Description	Address	Register SW Access	Field Name	Field Reset Value	Position	Field Description
tx_data_reg	TX data register	0x0	RW	tx_data	0	[7:0]	Parallel data from the host which is converted to serial data sent to peripheral device
				rfu	0	[31:8]	Reserved for future use
rx_data_reg	RX data register	0x4	RO	rx_data	0	[7:0]	Parallel data archived after serial-to-parallel conversion on data received on peripheral device
				rfu	0	[31:8]	Reserved for future use
cfg_reg	UART frame configuration register	0x8	RW	data_bit_num	0	[1:0]	The number of data bits 2'b00: 5 bits 2'b01: 6 bits 2'b10: 7 bits 2'b11: 8 bits
				stop_bit_num	0	[2]	The number of stop bits 1'b1: 1 bits 1'b0: disable
				parity_en	0	[3]	Parity enable 1'b1: enable 1'b0: disable
				parity_type	0	[4]	Parity type select 1'b1: Even 1'b0: Odd
				rfu	0	[31:5]	Reserved for future use
ctrl_reg	Operation control Register	0xC	RW	start_tx	0	[0]	Set to start converting parallel data received from the host to serial data sent to peripheral device
				rfu	0	[31:1]	Reserved for future use
stt_reg	Operation status register	0x10	RO	tx_done	1'b1	[0]	Set to indicate that previous parallel-to-serial conversion on data received from the host is completed
				rx_done	0	[1]	Set to indicate that previous serial-to-parallel conversion on data received on peripheral device is completed
				parity_error	0	[2]	A parity error occurs when the parity of the received



							character does not match the calculated parity
				rfu	0	[31:3]	Reserved for future use



2.4. Control Flow

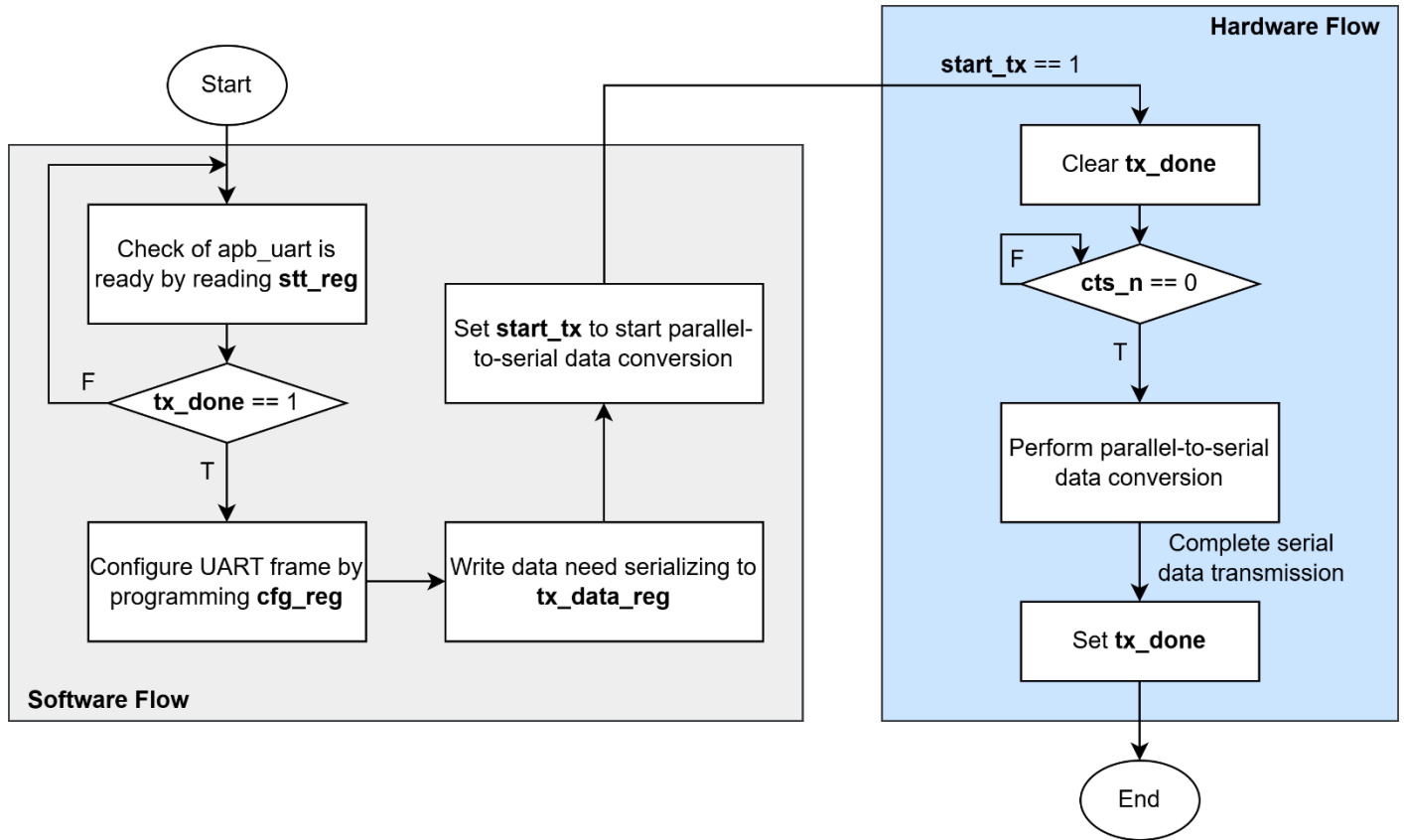


Figure 3. Parallel-to-serial data conversion control flow

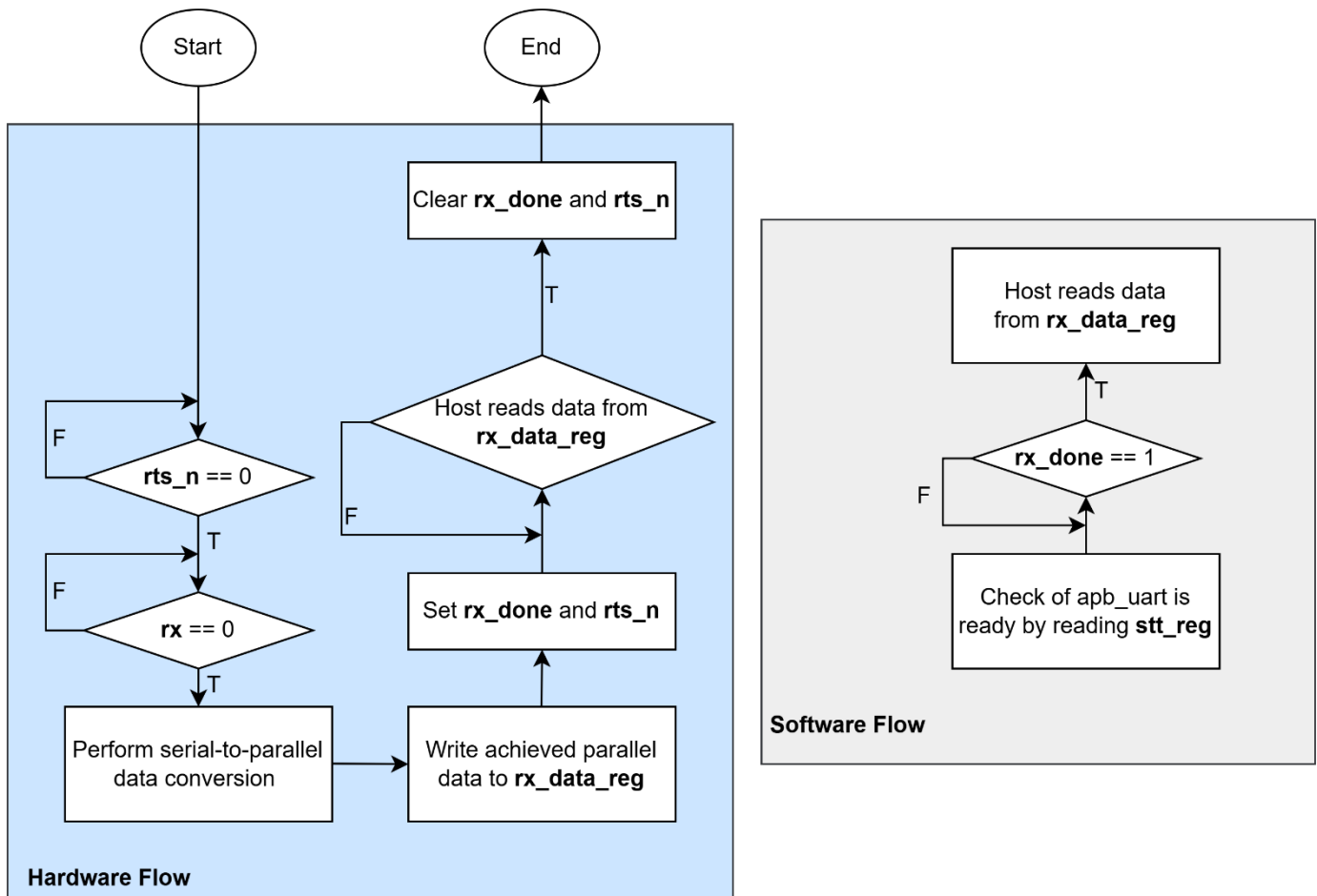


Figure 4. Serial-to-parallel data conversion control flow