Hardware co-simulation with communication server from MATLAB/Simulink

R. Bartosinski, J. Kadlec

Institute of Information Theory and Automation
Academy of Sciences of the Czech Republic

Abstract

This paper describes our solution for a hardware co-simulation of algorithms in programmable gate arrays (FPGAs) with a communication server out of Matlab. The implemented HW co-simulation with communication server has got many advantages and it allows uniform communication with different platforms via various communication interfaces (RS232, Ethernet, USB, etc.).

1 Introduction

When control and DSP algorithms are developed for FPGAs, at first they are often simulated in the MATLAB/Simulink environment. Verification of the implemented algorithms is also often done in the Simulink as a hardware co-simulation (for algorithms in micro-controllers it is called Processor-in-the-loop (PIL) co-simulation). During HW co-simulation, Simulink simulates the rest of the model without algorithm blocks for one sample interval and sends the algorithm input data to the hardware accelerator via a communication channel. When the hardware accelerator receives input data from the model, it executes the implemented algorithm for one sample step. Then the algorithm returns its output data computed during this step to Simulink, via a communication channel. At this point, Simulink finishes one sample cycle of the simulation and the model proceeds to the next sample interval. Algorithms are often developed for various hardware platforms with different communication interfaces (proprietary, serial, USB, ethernet, etc.) and for their using in a hardware co-simulation many of them need their own support in Matlab/Simulink. The proposed solution with universal Simulink block and a communication server eliminates many different blocks to one universal with unified access.

2 Proposed communication

One of possible solutions, how can be communication unified, is common Simulink block for communication with all development boards, but this block is rapidly extensible with difficulties. Better solution is one common Simulink block with communication through external server (Figure 1).

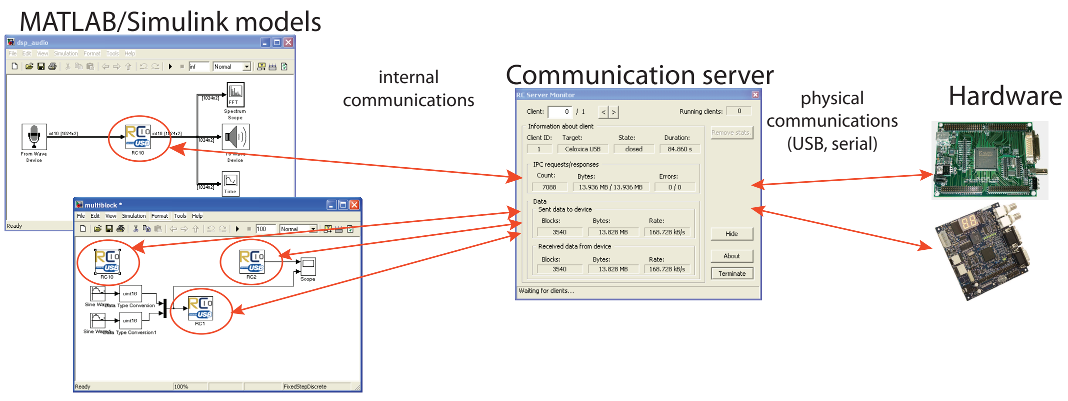


Figure 1: Data flow in a Simulink hardware co-simulation with the external communication server.

In the proposed solution input data are sent via inter-process communication to the multi-thread server, which sends data to the hardware platform. Thereafter the server reads data from the hardware and sends them back to Simulink as a block output data.
Some advantages of this solution are: simple extensibility with another platforms with different communication interface; sharing of a hardware board from several Simulink blocks or models; possibility to use board simultaneously from different applications and for example communication monitoring is also possible.

3 Simulink RC Library

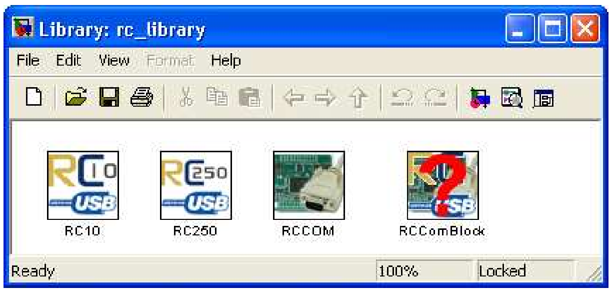
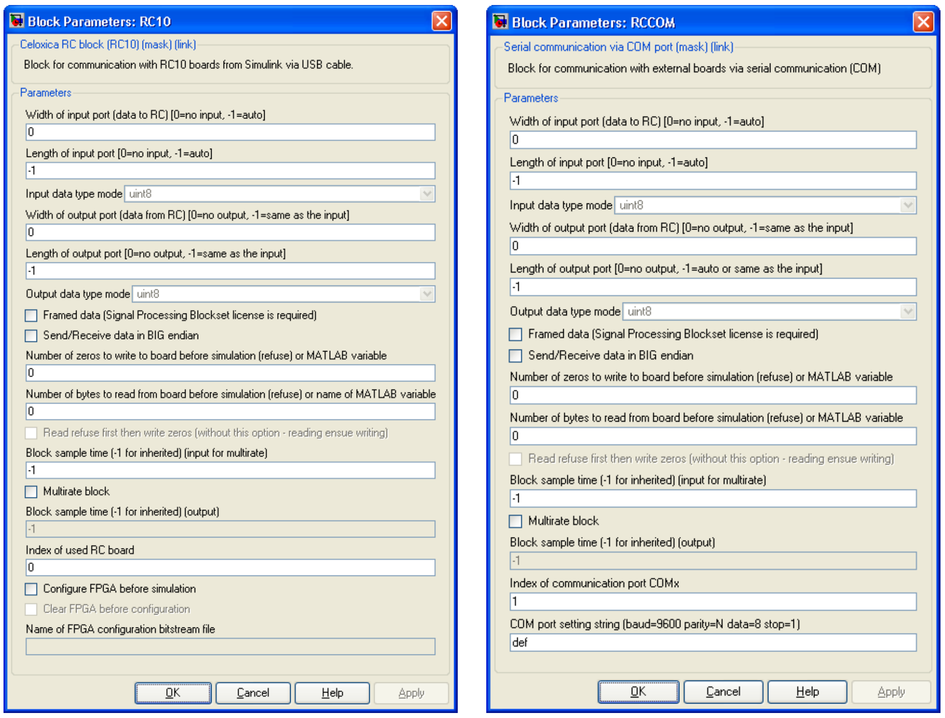


Figure 2: Simulink RC library for communication.

The solution with external server was originally created as a support for FPGA development board RC10 and RC250. Both communicate with PC through USB. But the function for communication with boards cannot be use from another process (Matlab crashes for memory access violation.). Therefore the communication server was a patch on this problem and the specific Simulink block was created for each board and for communication via serial port (see Figure 2).



Block Parameters: RC10

Celoxica RC block (RC10) (mask) (link)

Block for communication with RC10 boards from Simulink via USB cable.

Parameters

Width of input port (data to RC) [0=no input, -1=auto]: 0

Length of input port [0=no input, -1=auto]: -1

Input data type mode: uint8

Width of output port (data from RC) [0=no output, -1=same as the input]: 0

Length of output port [0=no output, -1=same as the input]: -1

Output data type mode: uint8

☐ Framed data (Signal Processing Blockset license is required)

☐ Send/Receive data in BIG endian

Number of zeros to write to board before simulation (refuse) or MATLAB variable: 0

Number of bytes to read from board before simulation (refuse) or name of MATLAB variable: 0

☐ Read refuse first then write zeros (without this option - reading ensue writing)

Block sample time (-1 for inherited) (input for multirate): -1

☐ Multirate block

Block sample time (-1 for inherited) (output): -1

Index of used RC board: 0

☐ Configure FPGA before simulation

☐ Clear FPGA before configuration

Name of FPGA configuration bitstream file:

OK Cancel Help Apply

Block Parameters: RCCOM

Serial communication via COM port (mask) (link)

Block for communication with external boards via serial communication (COM)

Parameters

Width of input port (data to RC) [0=no input, -1=auto]: 0

Length of input port [0=no input, -1=auto]: -1

Input data type mode: uint8

Width of output port (data from RC) [0=no output, -1=same as the input]: 0

Length of output port [0=no output, -1=auto or same as the input]: -1

Output data type mode: uint8

☐ Framed data (Signal Processing Blockset license is required)

☐ Send/Receive data in BIG endian

Number of zeros to write to board before simulation (refuse) or MATLAB variable: 0

Number of bytes to read from board before simulation (refuse) or MATLAB variable: 0

☐ Read refuse first then write zeros (without this option - reading ensue writing)

Block sample time (-1 for inherited) (input for multirate): -1

☐ Multirate block

Block sample time (-1 for inherited) (output): -1

Index of communication port COMx: 1

COM port setting string (baud=9600 parity=N data=8 stop=1): def

OK Cancel Help Apply

Figure 3: Setting dialogs of the RC10 and RCCOM blocks.

Because all blocks make the same functions and they have many setting identical (Figure 3) one common block 'RCComBlock' has been created. This block replaces all previously mentioned blocks and it can be used for all new interfaces supported in the communication server.

The setting dialog of the common block has two parts The first part is data flow settings which is the same for all communication interfaces, i.e. settings of block's input and output ports, their sizes and types. Here is also settings of block's sample time(s).
The second part is specific settings for each interfaces and boards. Here is mainly hardware initialization, i.e. choosing of FPGA configuration bitstrem file, hardware clock's settings. These properties are provided by communication server (more in section 'Communication server').

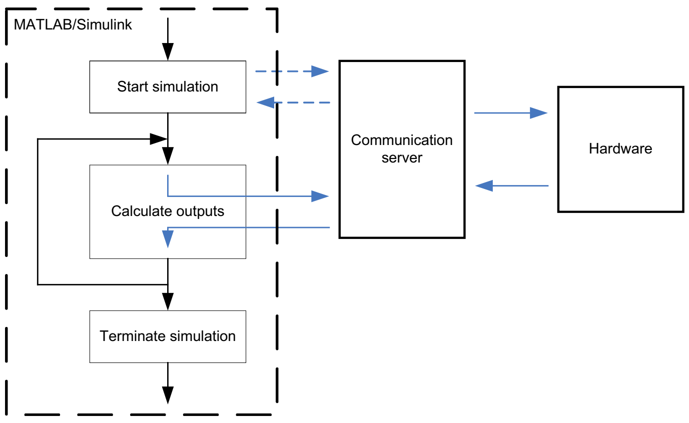


Figure 4: Interactions with external server during simulation.

Simulink interacts with server and hardware as shown in figure 4.

When a simulation starts communication block starts inter-process communication with the server and then the server opens communication with the board. In this phase, the server initiates hardware board, i.e. configures FPGA with a bitstream. Then the server can send or recieve one-shot Matlab data to/from hardware - it can be used for algorithm initialization. During simulation, the communication block re-types data to required format and sends them to hardware through the server if the block has an input port. If the block has an output port, the server reads required data from hardware and sends them to the block where they are re-typed to the block's output data type.

The block is primarily intended for simulation of discrete systems and it can be connected with all basic data types with various width and length for block data transfers. Because we need use the hardware accelerator for DSP applications with Signal Processing blockset the block's input/output data can be framed. For more universal use, the block can be set as a multirate block, i.e. input and output ports can have different sample time.

4 MATLAB RCCOM object

Table 1: MATLAB RCCOM COMMANDS.

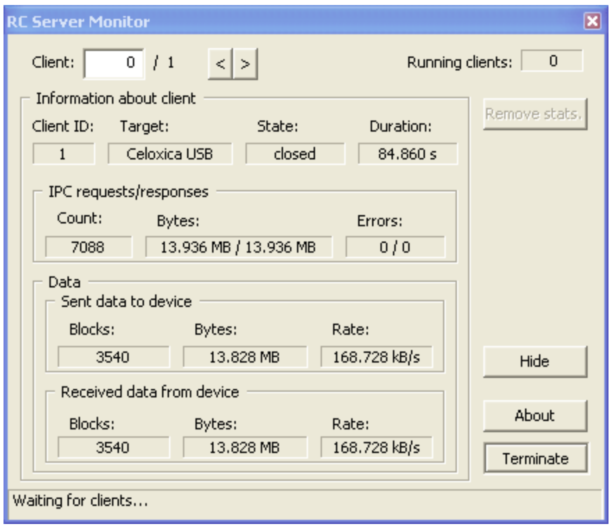
Function	**Description**
rccom_getversion	returns version of the communication server
rccom_enumifaces	enumerates all supported communication interfaces
rccom_enumdevices	enumerates all devices for the specific interface
rccom_enumobjects	enumerates all RCCOM objects
rccom	creates new RCCOM object for specific interface and device
fopen	opens communication between Matlab and hardware board
fclose	closes communication with hardware board
get	gets all device's getable properties or their values
set	gets all device's adjustable properties or sets their values
fread	reads data from the device described by RCCOM object
fwrite	sends data to the device described by RCCOM object

The special Matlab RCCOM object was created as another possible access to all supported hardware platforms via the communication server. This object is similar to the 'serial' object in Matlab serial port interface. The RCCOM library contains basic commands (Table 1) for interface, platform and object enumerations; open/close communication with hardware board; getting/setting properties of the connected board and read/write data from/to the hardware. Syntax of all commands is the same as the commands for the 'serial' object and it can be easily replaced.

5 Communication server

The communication server is multi-thread process, which inter-connects each client (Simulink communication block, Matlab RCCOM object, access from other applications) with the selected hardware board. This server allows hardware sharing from several clients (Simulink blocks). It also provides information about each supported communication interface and about their and the connected device's adjustable and getable properties.

Only the one visible part of server is a monitoring window, which shows information about data flow between each client and connected hardware and possibly it shows the last hardware or communication error, too.


RC Server Monitor

Client: 0 / 1
Running clients: 0

Information about client
Client ID: 1
Target: Celoxica USB
State: closed
Duration: 84.860 s

IPC requests/responses
Count: 7088
Bytes: 13.936 MB / 13.936 MB
Errors: 0 / 0

Data
Sent data to device
Blocks: 3540
Bytes: 13.828 MB
Rate: 168.728 kB/s

Received data from device
Blocks: 3540
Bytes: 13.828 MB
Rate: 168.728 kB/s

Waiting for clients...

Figure 5: Communication server's monitoring window.

The server can support whichever communication interface, which must be described with the following functions (Table 2). These functions are called during communication with hardware.

Table 2: Functions of each communication interface.

Function	**Description**
Init	Initialization of the interface
Done	Stop interface
EnumInst	Enumerate all connected boards
Open	Open communication with the board
Close	Close communication with the bodard
GetInfo	Get board text information
ReadData	Read data from the board
WriteData	Write data to the board
EnumProps	Enumerate all properties and prepare their description

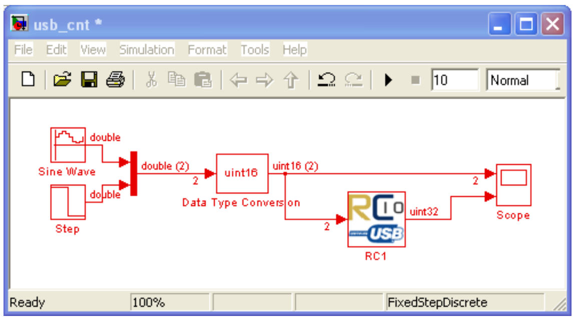
There is implemented communication via serial port and USB in this version of the server.

Another communication interface can be easily added to the server and in the next version of the server it will be able to be added as a dynamically linked library.

6 Examples

This section shows several simple examples of hardware co-simulation with the communication server. Size of block's input and output data must correspond to algorithm in the connected hardware. The communication block can be also used as an input interface from hardware (ADC, microphone) to Matlab/Simulink and as an output interface from Matlab/Simulink to hardware (DAC, speaker, ports).

The first example (Figure 6) shows simple 16-bit multiplier in hardware. It returns 32-bit result from two 16-bit number each sample time.



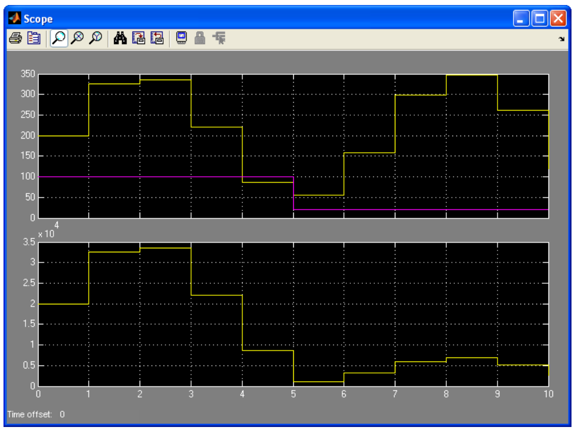


Figure 6: Example 1: Multiplication on RC10 board.

The second example (Figure 7) shows DSP audio filter in hardware connected between PC microphone and speaker. This example uses Signal Processing Blockset and its framed data. The hardware is RC10 board and it is connected via USB - data sended as framed blocks increase data throughput.

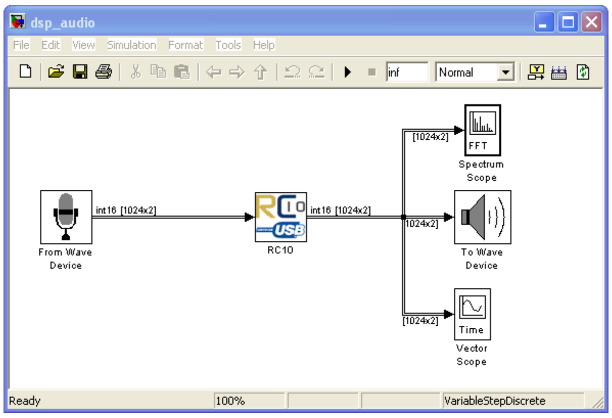
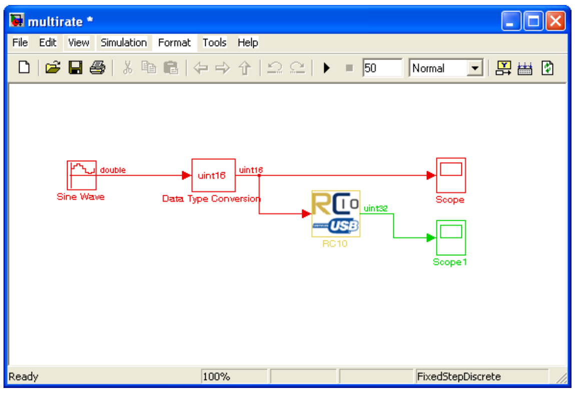
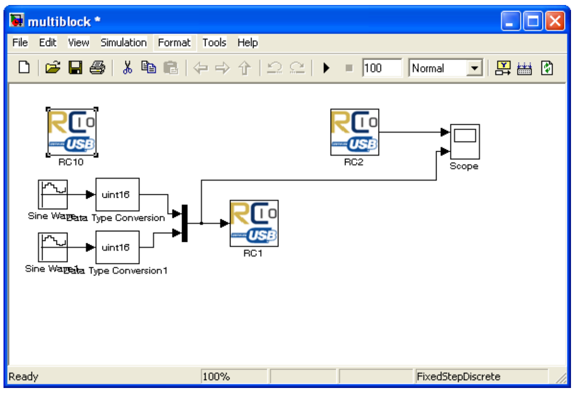


Figure 7: Example 2: DSP filter

The third example (Figure 8a) shows the communication block set as a multirate block, i.e. with different sample times on input and output ports. In this example, hardware provides simple 16-bit multiplication which reads each simulation sample time one 16-bit number and it returns one 32-bit number each second sample time. Simulink has one specific behaviour here - it waits for block's output data at time=0s even if simulation has started at time=0s and the simulation stops with error, because the hardware receives only one number and it waits for



(a) Multirate block



(b) Multiblock model

Figure 8: Example 3: Multirate block, Example 4: Multiblock model.

second one while Simulink waits for hardware's result.

The fourth example (Figure 8b) shows hardware sharing from several blocks. All three blocks are connected to the same hardware. The RC10 block configures hardware only and it doesn't participate on simulation, because it hasn't got any inputs or outputs. The RC1 block has got input port only and it displays input on LED. The RC2 block has gor one output port and it provides data measured by hardware ADC. Right priorities of all hardware blocks must be set in this example.

7 Conclusions

This paper describes hardware co-simulation with external communication server. The server provides interconnection between Matlab/Simulink and each supported hardware. Such interconnection can be realized with special Simulink blocks for each communication interface (and protocol), but the external server has got several advantages. For example the hardware platforms (with different interfaces) can be changed without influence on simulation model. The hardware can be changed by Matlab script and it can speed up co-simulation with several different hardware platforms. With external server, the hardware platform can be shared from several blocks, models or other programs. The server provides also connection with virtual platform and data monitoring.

8 Acknowledge

This work was partially supported by Academy of Sciences of the Czech Republic under project No. 1ET400750406.

References

[1] Celoxica. *RC host library and FTU3 manual*, 2005.

[2] The MathWorks. *MATLAB:External Interfaces*. 3 Apple Hill Drive, Natick, MA 01760-2098, USA, 2006.

[3] The MathWorks. *Simulink - Writing S-Functions*. 3 Apple Hill Drive, Natick, MA 01760-2098, USA, 2006.

Roman Bartosinski
Academy of Sciences of the Czech Republic,
Institute of Information Theory and Automation,
Department of Signal Processing,
Pod vodárenskou věží 4, 18208 Praha 8, Czech Republic
e-mail: bartosr@utia.cas.cz
www: sp.utia.cz

Jiří Kadlec
Academy of Sciences of the Czech Republic,
Institute of Information Theory and Automation,
Department of Signal Processing,
Pod vodárenskou věží 4, 18208 Praha 8, Czech Republic
e-mail: kadlec@utia.cas.cz
www: sp.utia.cz