M2M Recoms integration documentation

Révision	Date	Qui	Commentaires
0.1	10.01.2011	CBT	initial version

Contenu

PROJECT LOCATION:

The m2m_recoms project is store in two places: In the git recoms project and in th svn Math2Mat project.

The recoms_project is split in two gits repositories: recoms_hw contain all hardware structure. The two Matlab simulink design for m2m are store in this project. The current folder is \recoms_hw\dev\xilinx\Simulink\Basic_Design_m2m
Recoms_sw contain the user application, the current folder is \recoms_sw\app\m2m\m2m_test
The SVN Math2Mat project has a folder who contains the demonstration repositories: all m2m_recoms project is store in /wp4/recoms/dev repository. In the “hard” directory there are two zip of the matlabsimulink basic_m2m_design project, one zip for the toolsbox and the src vhdl file for the m2m_controller. In the “soft” directory there is the source of the user application, the doc folder contain this doc and others recoms / Math2Mat documentations. And the exe folder contain the execute environment for 1IN/1OUT and 2IN/2/OUT project: Bitstream (.xml and .bin files), generated Input and Output files, src_vhdl for the two examples functions.

1. HARDWARE SIDE: M2M INTEGRATION WITH MATLAB SIMULINK IN RECOMS PROJECT

1.1 Creation m2m files

The first step is creating the vhdl file with the m2m application tools. For this read the User_doc_Plugin.doc and dev_doc_plugin.doc files. When the Math2Mat Viewer is starting, open or create a project. Edit the matlab function, for example:

```
function s = test_base(a)

s = 2 * a;

endfunction;
```

Currently, only simple functions are available, the loop functions are not supported.

Before start the application you must configure the octave proprieties and simulation proprieties:

- Configure… → external tools : you must set the Path to Octave application
- Configure… → Simulations : in the filed “number of sample” you must set the number of data generating (the application below has tested with 500 data)
- for others fields, the application works with the default values

In “src_VHDL” folder there are the VHDL files generated by the application. The top of VHDL hierarchy is in the wrappers folders. For the Recoms project we use the wrapper_Recoms_xxx.vhd.

In “comp” folder there are the data files generated by Octave: there is one file per Input “file_input***x***.dat” and one file per output “file_output***x***.dat” (***x*** is the input/output number, incremental number started with 1)

1.2 Import m2m bloc in matlab simulink

After starting Matlab open Basic_Design_m2m project. In the “current directory” windows, open the Project_Base.mdl, and then go to the “USER_DESIGN” page. The project should look like the figure below:

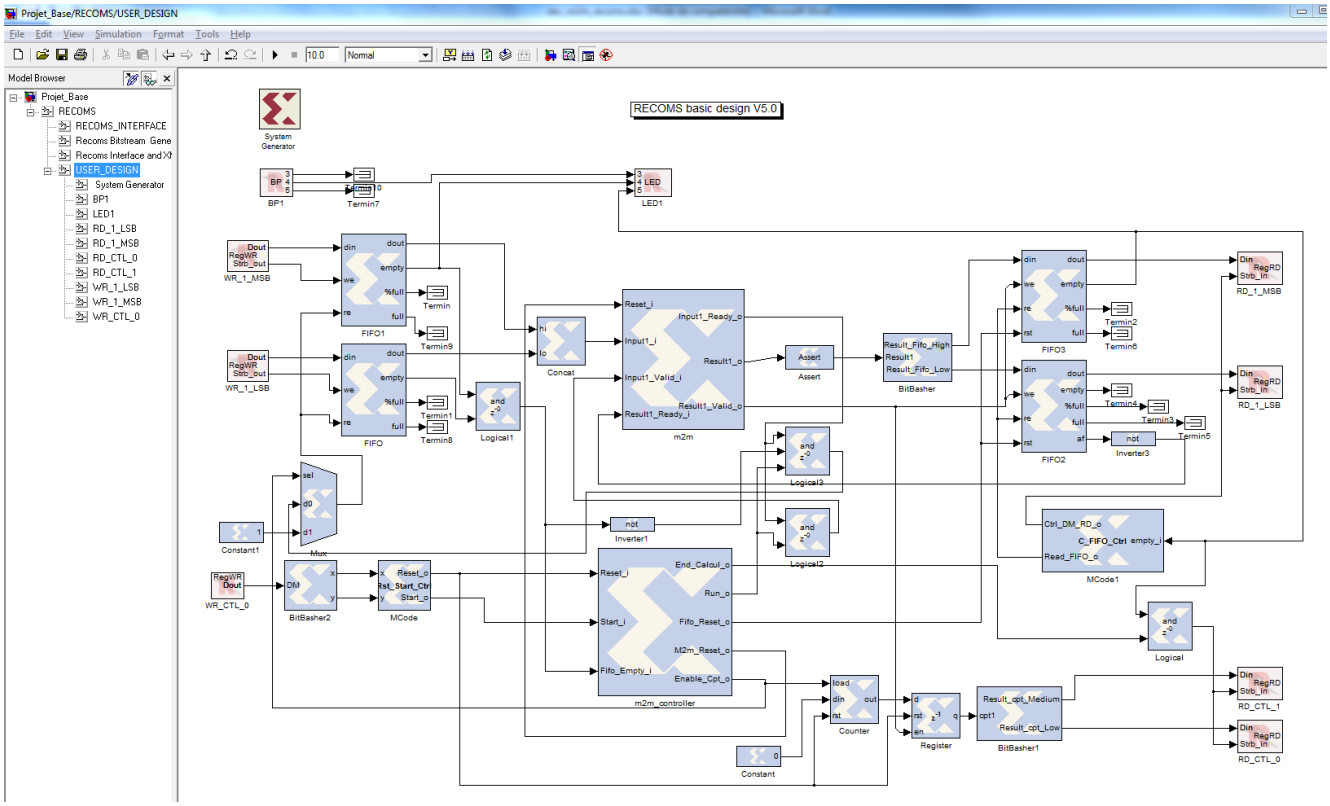


figure1: user_design project m2m_recoms (basic: 1 input 1 output)

Currently this design works for the s=2*a function. If you have another function you must do the following modification:

1) Delete the m2m bloc and open the library browser → then open the Xilinx Blockset libraries.

2) Create new black box (drag and drop black box element to the USER_DESIGN page) and import the top of VHDL file (wrapper_Recoms_xxx.vhd). Rename the black box. More information about the black box can be found with the help function. (right clic on the black box → help)

3) Modify the "black box" config file: On the main windows of Matlab, open the wrapper_Recoms_test_base_config.m file (this is the default file name if you don't change it when the black box is created). The m2m block is not combinational, you must comment the line 18:

```
% System Generator has to assume that your entity  has a combinational feed through;
%   if it  doesn't, then comment out the following line:

%this_block.tagAsCombinational;
```

4) Verify if the Input and Ouput generated are correct, for every output change "UFix_1_0" type to "Bool". In the "Add additional source files as needed" section, add all others VHDL files generated in folder "src_VHDL". You must add files in the order in which they should be compiled. See the example below.

```
  % Add addtional source files as needed.
  %  |-------------
  %  | Add files in the order in which they should be compiled.
  %  | If two files "a.vhd" and "b.vhd" contain the entities
  %  | entity_a and entity_b, and entity_a contains a
  %  | component of type entity_b, the correct sequence of
  %  | addFile() calls would be:
  %  |    this_block.addFile('b.vhd');
  %  |    this_block.addFile('a.vhd');
  %  |-------------

  %    this_block.addFile('');
  %    this_block.addFile('');
  this_block.addFile('E:/projets/math2mat/trunk/Release/1.0/test_proto/src_VHDL/pkg_definition.vhd');
  this_block.addFile('E:/projets/math2mat/trunk/Release/1.0/test_proto/src_VHDL/pkg_cellule.vhd');
  this_block.addFile('E:/projets/math2mat/trunk/Release/1.0/test_proto/src_VHDL/misc.vhd');
  this_block.addFile('E:/projets/math2mat/trunk/Release/1.0/test_proto/src_VHDL/mult.vhd');
  this_block.addFile('E:/projets/math2mat/trunk/Release/1.0/test_proto/src_VHDL/mult2.vhd');

this_block.addFile('E:/projets/math2mat/trunk/Release/1.0/test_proto/src_VHDL/wrapper_mult2_pipe_csa_32.vhd'
);
  this_block.addFile('E:/projets/math2mat/trunk/Release/1.0/test_proto/src_VHDL/test_base.vhd');

this_block.addFile('E:/projets/math2mat/trunk/Release/1.0/test_proto/src_VHDL/wrappers/wrapper_Recoms_test_b
ase.vhd');
```

5) If you have more than one Input or one Ouput you must modify the number of DM (data mover): for one Input (32 bits) there is two DM 16 bits

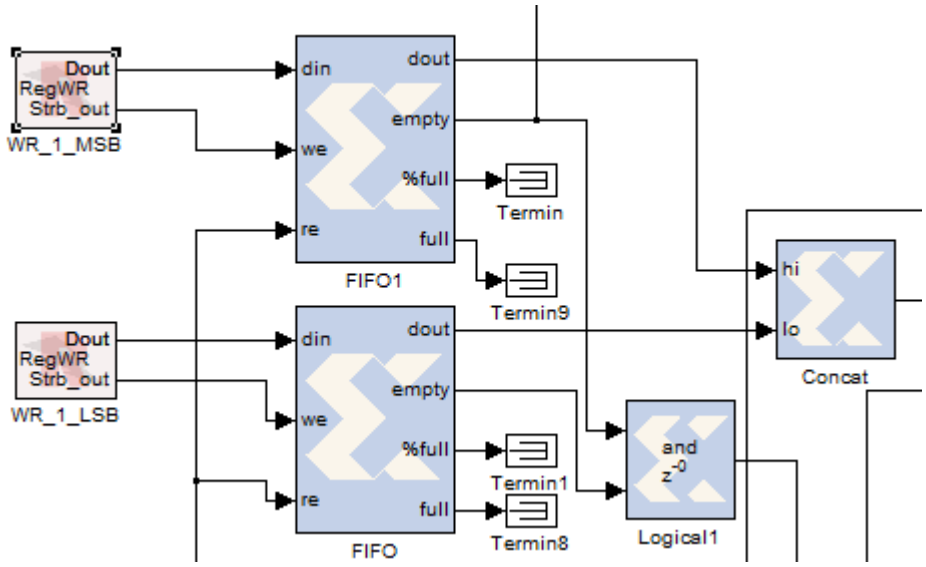


Figure 2: DM and Fifo for one Input

The DM name convention is defined in the next chapter (1.3). You must respect the name of the DM, if not the application in the linux side doesn't work correctly.

If you have a second entry or more: copy and paste the bloc in the figure above (2 DM 2 Fifos, 1 concat element) all fifo empty signal are connected to an “And” gate. For an additional output is the same principle.

6) The figure below shows the fifo configuration. Currently the size of the fifo (Depth 512) are always higher than the number of data sending to the fifo. The Depth can be set from 16 to 64K. If you have more data than 64K, you must add a controller bloc to check the “full” signal of the fifo.

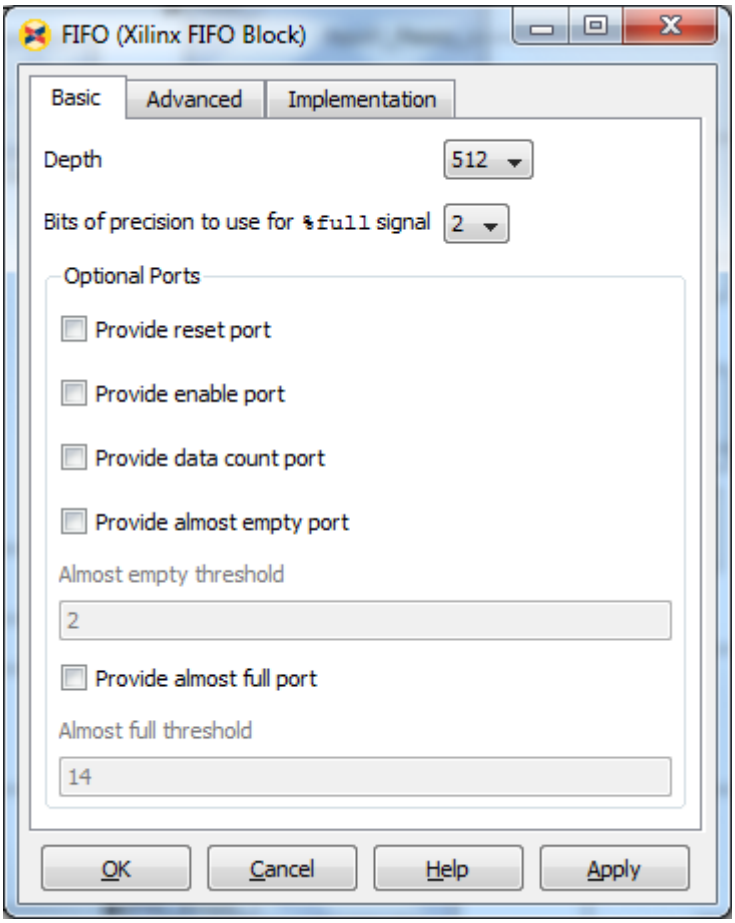


Figure 3: fifo block configuration

1.2.1 Second example with two Input / Output

The project name is Basic_Design_m2m_2in_2out.
Example of Matlab function:

```
function [s,t] = test_base(a,b)

  s =  a + b;
  t= a * b;

endfunction;
```

User design of Matlab:

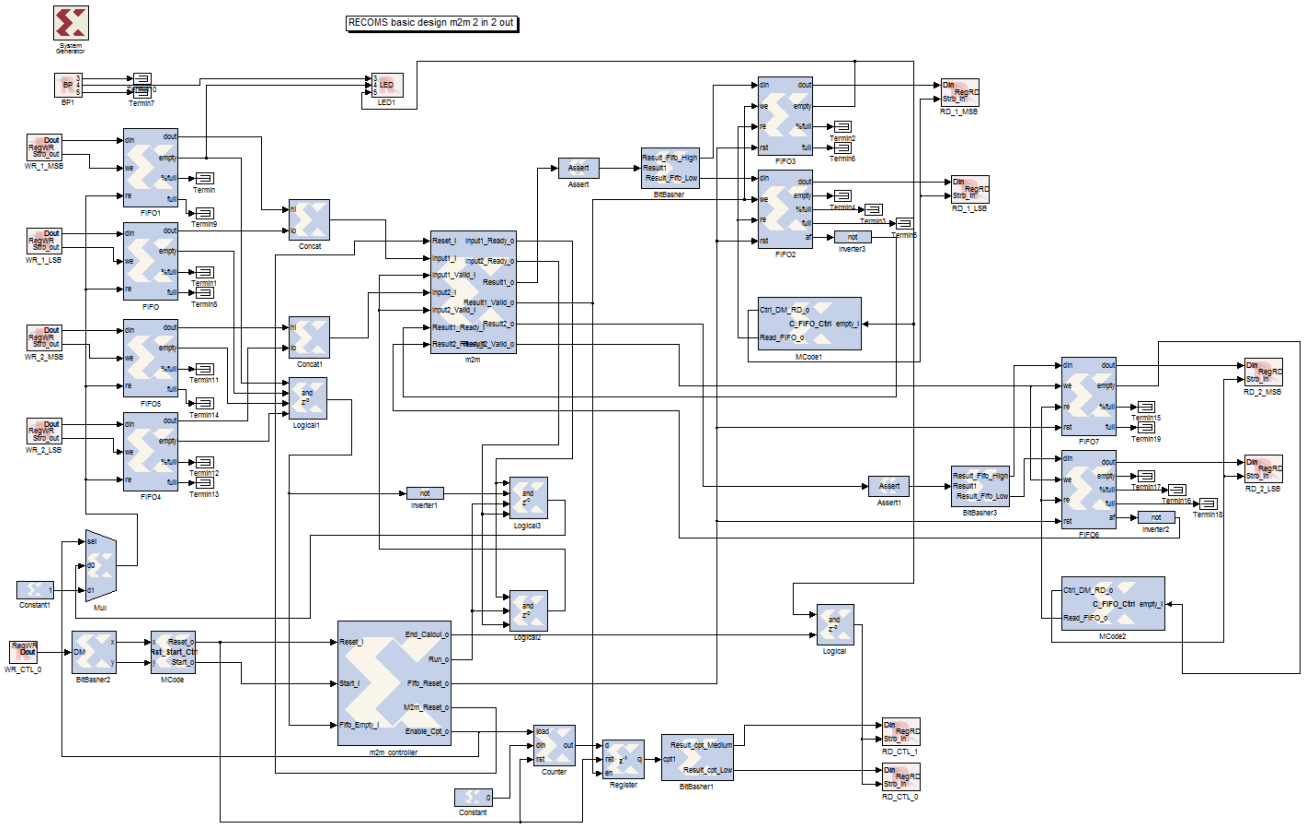


figure4: user_design project Basic_design_m2m_2in_2out

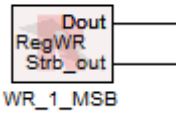
1.3 DM nomenclature

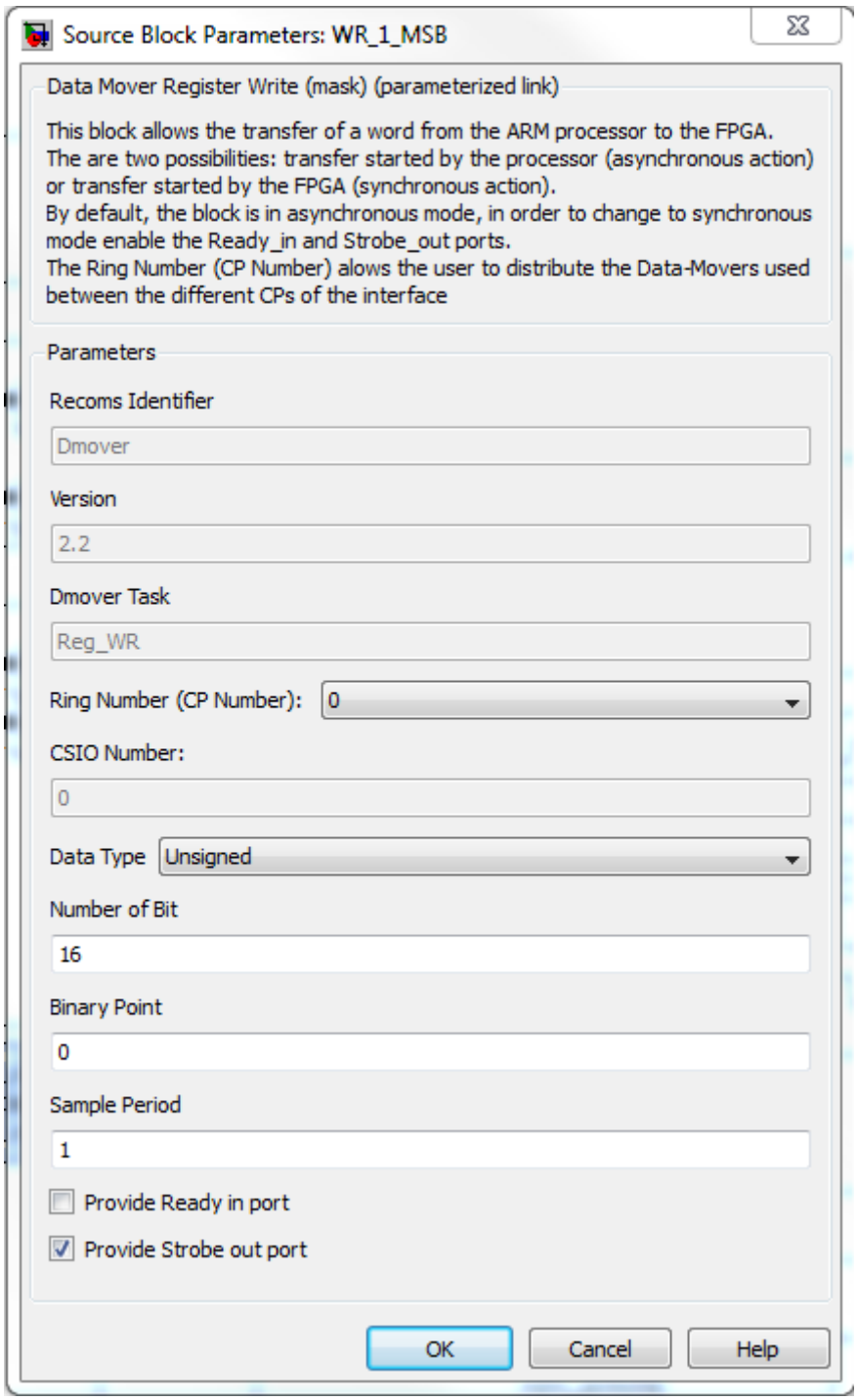
Each DM is connected to a ring buffer (communication port) and a csio number. We have defined five ring buffers that are linked with the user space application: two ring buffer for input data, two for output data and one for all control signals. The constraint is that the CP (communication port) is only available for 16bits but data are on 32bits that is the reason why we have two DM per data input/output. If we want that all data sent by the user application are going to the corresponding DM, we must respect the following nomenclature:

All DM for data Input are connected to ring 0 for MSB data, ring 1 for LSB data. The csio value is an incremental default value. All data output are connected to ring 2 for MSB data and ring 3 for LSB data. All signals control are connected on ring 4

All inputs begin with “WR”, this corresponding with a write function from the user application. All outputs begin with “RD”. (Read for the user application)

Example: DM and configuration windows for the first input (16bits MSB)





The next example table contains all names, ring, Csio for two Inputs and two Outputs.

DM name	Ring number	CSIO number Automatic value	comment
WR_1_MSB	0	0	First Input data 16 bits MSB
WR_1_LSB	1	0	First Input data 16 bits LSB
WR_2_MSB	0	1	Second Input data 16 bits MSB
WR_2_LSB	1	1	Second Input data 16 bits LSB
RD_1_MSB	2	0	First Output data 16 bits MSB
RD_1_LSB	3	0	First Output data 16 bits LSB
RD_2_MSB	2	1	Second Output data 16 bits MSB
RD_2_LSB	3	1	Second Output data 16 bits LSB
WR_CTL_0	4	0	First write control signals
RD_CTL_0	4	1	First read control signals
RD_CTL_1	4	2	Second read control signals
…	…	…	…

You can find more information about feature of DM, ring buffer, Csio… in the main recoms board specification doc: Recoms_Board_Spec.pdf

2. SOFTWARE SIDE: APPLICATION IN USERSPACE RECOMS

2.1 Start with recoms board

The board boot automatically when is powered. There are two possibilities to connect with the recoms board: with serial terminal or with a ssh connection. After booting, we can see the IP address on the screen. If you don't have an IP address you must connect with the serial terminal, for example with picocom:

```
# picocom –b 9600 /dev/ttyUSB0
```

Then you must configure the network on the card.

When you have an IP Address and if you are in the same network:

```
# ssh root@ip_address
```

2.2 Working directory

The application works direcly in /home/root. The bitstream directory is in /home/root/rvp/test_m2m_fifo.

2.2.1 Procedure to program the FPGA

First copy the Project_Base.xml and recoms_top.bin files to /home/root/rvp/test_m2m_fifo (you can use the "scp" command).
Refresh the environments with the refresh button on the screen, navigate with the touchscreen and go to the project folder (Basic_design_m2m) → clic and a window appear and ask: "Do you want to load the "Basic_design_m2m" virtual peripheral?" → yes. A new window indicates when the program and configuration are done. The LED sw4 and sw5 are turn on.

2.2.2 Procedure to run application

First copy all file_inputx.dat and file_ouputx.dat in /home/root/ copy the application (m2m_test) in the same folder. Verify if the bitstream has loaded. You can check this, if they are some DM names in /sys/kernel/design/top folder. Delete the older resultx.dat files
Start the application with ./m2m_test

2.3 m2m_test application

This application check the number of data Input/Ouput, read and send all the data to the FPGA, send the start signal, read all result from FPGA and compare them with the generated data reference file (file generated by Octave). The application returns also the number of cycle for calculate all data. To find the calculate time; multiply the number of cycle with the clock frequency. By default the clock system in FPGA are set to 15.6 [ns].

2.3.1 Examples

Example with 1Input and 1Output with 500 data: (the function is 2*a)

```
root@colibri:~$ ./m2m_test
test read write fifo m2m
nb_input  1 , nb_output 1
number of bytes read from /home/root/file_input1.dat : 2000
number data write to fifo fpga 500
number of bytes read from fifo fpga 2000
number of bytes write in /home/root/result1.dat : 2000
result 506.000000 , 0x1fa

start compare result between /home/root/file_output1.dat and /home/root/result1.
dat
500 data compared:
No error found
exit application
root@colibri:~$
```

The “result” line corresponds to the number of cycle for calculate all data. (in this case we have 506 clock cycle). The time duration is number of cycle * clock_fpga_user_design: 506*15.6[ns]=7.89[us]
We also note that there is no error between the calculated file and the reference file

Example with 2In/2Out with 500 data: (function are a+b, a*b)

```
root@colibri:~$ ./m2m_test
test read write fifo m2m
nb_input  2 , nb_output 2
number of bytes read from /home/root/file_input1.dat : 2000
number data write to fifo fpga 500
number of bytes read from /home/root/file_input2.dat : 2000
number data write to fifo fpga 500
number of bytes read from fifo fpga 2000
number of bytes write in /home/root/result1.dat : 2000
number of bytes read from fifo fpga 2000
number of bytes write in /home/root/result2.dat : 2000
result 507.000000 , 0x1fb

start compare result between /home/root/file_output1.dat and /home/root/result1.
dat
500 data compared:
No error found

start compare result between /home/root/file_output2.dat and /home/root/result2.
dat
500 data compared:
No error found
exit application
root@colibri:~$
```

In this case the time duration is 507*15.6[ns]= 7.91[us]
We also note that there is no error between the calculated files and the reference files.

2.3.2 Example with errors

If the bitstream isn't loaded, this message appears:

```
root@colibri:~$ ./m2m_test
test read write fifo m2m
ERROR bitstream in FPGA NOT LOAD or bad bitstream
error read info driver
root@colibri:~$
```

Example with two data errors in the reference file: (file_output1.dat)

```
root@colibri:~$ ./m2m_test
test read write fifo m2m
nb_input  2 , nb_output 2
number of bytes read from /home/root/file_input1.dat : 2000
number data write to fifo fpga 500
number of bytes read from /home/root/file_input2.dat : 2000
number data write to fifo fpga 500
number of bytes read from fifo fpga 2000
number of bytes write in /home/root/result1.dat : 2000
number of bytes read from fifo fpga 2000
number of bytes write in /home/root/result2.dat : 2000
result 507.000000 , 0x1fb

start compare result between /home/root/file_output1.dat and /home/root/result1.
dat
ERROR not same value on data 38
ERROR not same value on data 95
500 data compared:
WARNING: 2 error found

start compare result between /home/root/file_output2.dat and /home/root/result2.
dat
500 data compared:
No error found
exit application
root@colibri:~$
```

Sources of application are in git project: \recoms_sw\app\m2m\m2m_test
Svn project: /wp4/recoms/dev/soft/