mprfgen user manual

Title	mprfgen
Description	Multi-port register file generator
Author	Nikolaos Kavvadias 2007-2013 Courtesy of Ajax Compilers
Copyright	Ajax Compilers 2012, 2013
Contact	nkavvadias@ajaxcompilers.com
Website	http://www.ajaxcompilers.com
Release Date	12 January 2013
Version	1.0.0
Rev. history	
v1.0.0	12-01-2013 First commercial release.
v0.0.2	07-10-2010 The first real update since the initial version. Should work with ISE XST 12.3. Read-through mode added.
v0.0.1	27-12-2007 Initial release.

1. Introduction

`mprfgen` is a multi-port register file generator. It can either infer storage or instantiate block-RAM embedded memories for the Xilinx Spartan-3, Virtex-4, Virtex-5 FPGAs or newer.

The generated memories are initialized to ZEROs. Also, when using RAM inference, the following compilation/elaboration order should be used:

1. `util_functions_pkg.vhd`
2. `regfile_core.vhd`
3. `<your-generated-file>.vhd`

Please, note that the default read mode option (read-async) will generate distributed RAM. For block RAM, use read-first, write-first or read-through, instead.

2. File listing

The mprfgen distribution includes the files listed in the following table. Files and/or directories denoted by a capital S are only available in source releases of mprfgen.

/mprfgen	Top-level directory
S build.sh env.sh	Automated build script. Script for setting up the environment.
/bin/lin	Linux executable directory
mprfgen.exe	Static executable for Linux platforms.
/bin/win	Windows executable directory
mprfgen.exe	Static executable for Windows platforms.
/doc	Documentation directory
AUTHORS LICENSE README README.html README.pdf VERSION	List of authors. End-user license agreement. This file. HTML version of README. PDF version of README. Current version of mprfgen.
/sim	Directory for simulation files
change-generics.pl ghdl.mk	Perl script for adapting the testbench template. Makefile for running a GHDL (http://ghdl.free.fr) simulation.
ghdl.sh regfile_tb_tmpl.vhd	Bash script for running a GHDL simulation. Generic testbench for simulative testing of generated multi-port register files.
S /src	Source code directory
S Makefile **S** mprfgen.c	Makefile for generating the mprfgen executable. The source code for the application.
/test	Directory for testing mprfgen
regfile.vhd	Generic model of a multi-port register file (for reference).
regfile_core.vhd	Single-read, single-write port memory model used by the generated memories (mandatory for inferred memory).
test[1-5].vhd test.sh util_functions_pkg.vhd	Sample files generated as shown in section 5. Generate the samples discussed in section 5. A VHDL package providing some utility functions.

3. mprfgen setup

For using mprfgen, this distribution comes with ready-made native executables for Windows and Linux, which can be found in the corresponding `/bin/exedir` directory, where `exedir` is `lin` for Linux or `win` for Windows.

In order to build mprfgen from source on your platform, just run the `build.sh` bash script from the command prompt. First, change to the appropriate directory, for instance:

```
cd /home/user/mprfgen/
```

assuming `mprfgen` was installed inside `/home/user` and finally run the script:

```
$ ./build.sh
```

4. mprfgen usage

The `mprfgen` program can be invoked with several options (see complete options listing below). The usual tasks that can be accomplished with `mprfgen` are:

- infer a multi-port register file for NWP write ports and NRP read ports
- instantiate a multi-port register file for Xilinx FPGAs

`mprfgen` can be invoked as:

```
$./mprfgen [options] <out.vhd>
```

options' is one or more of the following:

-h Print this help.

-infer Use generic RAM storage that can be inferred as block RAM(s).

-<read-mode> Read mode supported by the generated RAM. Valid options: (read-async, read-first, write-first, read-through). `read-through` cannot be used for block RAM instantiation. Default is `read-async`.

-nwp <num> Number of write ports for the register file.

-nrp <num> Number of read ports for the register file.

-bw <num> Bitwidth for each memory entry.

-nregs <num> Memory size (number of words).

5. Running some tests

Here follow some simple usage examples of `mprfgen` assuming the user is at the `/mprfgen/test` directory and is running Linux.

1. Generate a 3-read, 2-write port generic register file.

```
$ ../bin/lin/mprfgen.exe -infer -nwp 2 -nrp 3 test1.vhd
```

2. Generate a 1-read, 1-write port 32x2048 memory.

```
$ ../bin/lin/mprfgen.exe -infer -read-first -nwp 1 -nrp 1
-bw 32 -nregs 2048 test2.vhd
```

3. Generate a 2-read, 1-write port LUT-based register file.

```
$ ../bin/lin/mprfgen.exe -infer -read-async test3.vhd
```

4. Generate a 2-read, 1-write port block RAM register file with direct instantiation.

```
$ ../bin/lin/mprfgen.exe -read-first test4.vhd
```

5. Generate a 2-read, 2-write port block RAM register file with direct instantiation.

```
$ ../bin/lin/mprfgen.exe -infer -read-first -nwp 2 -nrp 2
test5.vhd
```

It is better to set up the `MPRFGEN_BIN_PATH` environmental variable using

```
$ source env.sh WINDOWS
```

or

```
$ source env.sh LINUX
```

The user can add the mprfgen binaries directory to the PATH, e.g.:

```
export PATH=$MPRFGEN_BIN_PATH:$PATH
```

in order to directly use `mprfgen` invocations.

Windows users are prompted to set the `MPRFGEN_BIN_PATH` environmental variable to an appropriate value. An example is `C:\mprfgen\bin\win`, assuming that `mprfgen` has been installed in `C:`.

6. Simulation

The generated multi-port register files can be simulated with the supplied files found in the `/sim` subdirectory, given that they have been generated with the `-infer` option enabled. This means that the provided `test4.vhd` cannot be simulated, while `test1.vhd`, `test2.vhd`, `test3.vhd`, and `test5.vhd` that don't require proprietary Xilinx libraries can.

For running the GHDL simulation of a generated file, e.g. `test1.vhd`, change directory to the `/sim` subdirectory:

```
$ cd $MPRFGEN_HOME/sim/rtl_sim/run
```

assuming `MPRFGEN_HOME` is the directory where the top-level `/mprfgen` is found.

Then, the corresponding shell script is executed:

```
$ ./ghdl.sh test1
```

The simulation produces a VCD (waveform) dump named `test1.vcd` which can be inspected for simulation correctness.

For the stimulus data used in the testbench, it is assumed that the following inequations hold:

- `NWP * AW >= 4`
- `NRP * AW >= 4`
- `DW >= 8`

7. Prerequisities

- Standard UNIX-based tools (tested with gcc-3.4.4 on cygwin/x86, gcc-3.4.5 on mingw/x86, gcc-4.1.2 on Fedora 8 and gcc-4.6.1 on Ubuntu 11.10).
  - make
  - bash (shell)
  - perl

  For rebuilding on Windows, the Cygwin (http://sources.redhat.com/cygwin) is a popular solution, since it provides a near-complete POSIX environment.
- GHDL simulator (http://ghdl.free.fr) for Windows or Linux.

  The latest GHDL distribution (0.29.1, Windows version) also installs GTKwave on Windows.