BMPR: B*-tree based Modular Placer and PR-aware Router

Install (Linux):

1. Download file to a folder and go to the folder.
2. "tar –xvf bmpr.tar" (if it has a bmpr.tar file)
3. Go to main folder of BMPR. (the folder includes m_fpga.pl and m_vpr.pl)
4. "make"
5. go to folder "vpr_reconfig/placer_m"
6. "make"
7. Finish install.
8. (Back to main folder of BMPR.)

Quick Test:

1. Go to main folder of BMPR.
2. "perl m_fpga.pl mkSMAdapter4B 6k6_n8_10.xml 1"
3. "perl m_vpr.pl mkSMAdapter4B 6k6_n8_10.xml"

Note: mkSMAdapter4B is the case name, 6k6_n8_10.xml is the architecture, and 1 is a parameter to prepare data regarded as a library.
If the user needs to test other cases, he/she can replace the case mkSMAdapter4B with another case name, eg.
perl m_fpga.pl newcasename 6k6_n8_10.xml 1
perl m_vpr.pl newcasename 6k6_n8_10.xml

We give more detailed information about the tool below.

Prepare files:

BMPR is modified from VTR and B*-tree (open source). Thus, arch file, .v file, etc. should follow folder and path of the VTR. Here is some extra setting for using BMPR.

1. Verilog files: User should decompose logic function into modules. Meanwhile, the connection port should have the same name.

Here is an example, ch_intrinsics in VTR benchmarks:

```
module memset
(
clk, reset, start, finish, return_val, m, c, n,
memory_controller_write_enable, memory_controller_address,
memory_controller_in, memory_controller_out
);

module memory_controller
(
clk, memory_controller_address, memory_controller_write_enable,
memory_controller_in, memory_controller_out, str_address,
str_write_enable, str_in, str_out
);
```

Ports of the same name indicate there is a connection.
In the example, memory_controller_address of *memset* is output while memory_controller_address of *memory_controller* is input. There is a singal memory_controller_address sent from *memset* to *memory_controller*. If you are used to design a logic function with sub-modules, now you do not need a top module anymore with signal mapping anymore. You should only focus on giving the exactly same name for modules with connection. Here you need to notice that there should be no duplicate name for modules of o conntection. ***vrename*** is a useful tool for changing ports' name. you can find it here http://www.veripool.org/wiki/verilog-perl.

2. Architecture files: Please provide of architecture files with varies aspect ratio. You can modify it from the *main architecture* (m-arch) by setting "<layout auto="0.8"/>". Here the aspect ratio is 0.8. With more aspect ratio, it may help optimization of BMPR. We highly recommend you set ratio and 1/ratio, *e.g*. 0.5 and 2(1/0.5).
3. Files in the task folder: There should be **5** files in the *config* folder, ***architecture file***, ***list.txt***, ***xconfig.txt***, ***yconfig.txt***, and ***zconfig.txt***. The ***architecture file*** is the m-arch of the FPGA. You can use any m-arch name here. Modules of the logic function should be listed in the ***list.txt***. Please list all Verilog files of modules and architecture files with various aspect ratios for ***xconfig.txt***. In ***yconfig.txt***, you can use only m-arch for simulation. Meanwhile, please change filename extension ".v" to ".pre-vpr.blif" for all modules and start running stage from vpr. The ***zconfig.txt*** is a normal config with only m-arch and top-module's Verilog file. It is possible to skip this stage if you do not have top module's Verilog file.
4. Do not forget add clock information of modules to file "benchmark_clocks.clock" as the same manner as VTR(VPR).

Read to Go:

1. To prepare the useful information for BMPR, you should run a trial round:
   "perl m_fpga.pl ***logic m-arch n***"
   Here we call a perl file, m_fpga.pl. ***logic*** is your name of the logic function without filename extension. ***m-arch*** is your name of the main architecture with filename extension. ***n*** is round number. If there is no run1, run2, run3, etc. in your task folder, you can just set it as "1" for running trial round from run1 to run3. Or, you can set it to number of next round. For example, if there are run1 to run12 in the task folder, you should set ***n*** to 13.
2. With information, running BMPR is very simple.
   "perl m_vpr.pl ***logic m-arch***"
   Definitions of ***logic*** and ***m-arch*** are the same as above.

Parameters and Customization (Advanced):

1. Stop showing display by editing **m_vpr.pl** line **24**:
change
system("../../../vpr/vpr $arch $mainname -route -route_chan_width 200");
to
system("../../../vpr/vpr $arch $mainname -route -nodisp -route_chan_width 200");
2. Changing the channel width 200 to ***n*** by editing ***m_vpr.pl*** line **24**:
change
system("../../../vpr/vpr $arch $mainname -route -route_chan_width 200");
to
system("../../../vpr/vpr $arch $mainname -route -route_chan_width ***n***");
n should be a reasonable number based on your architecture.
3. Accelerating execution time of the trial round is possible. VTR supports multicore processing, thus, we set ***n*** cores by editing ***m_fpga.pl*** line 26:
change
system("perl vtr_flow/scripts/run_vtr_task.pl $mainname -p6");
to
system("perl vtr_flow/scripts/run_vtr_task.pl $mainname -p***n***");
n should be a reasonable number based on your core numbers. We preset it as 6 cores system.
4. It is possible to skip physical synthesis step of ***zconfig.txt***, which is known as top-module synthesis, for accelerating operation time. However, the user should provide a netlist-like file or pre-synthesized netlist. Here is one example of boundtop:

```
1  <block name="boundtop.net" instance="FPGA_packed_netlist[0]">
2      <inputs>
3          top^tm3_clk_v0 top^raygroup01~0 top^raygroup01~1 top^raygroupvalid01
           top^raygroup10~0 top^raygroup10~1 top^raygroupvalid10 ...
4      </inputs>
5
6      <outputs>
7          out:top^we_vblockramcontroller out:top^saddr~0 out:top^saddr~1 out:top^saddr~2
           out:top^saddr~3 out:top^saddr~4 out:top^saddr~5 out:top^saddr~6 out:top^saddr~7
           out:top^saddr~8 out:top^saddr~9 ...
8      </outputs>
9
10     <globals>
11         top^tm3_clk_v0
12     </globals>
13 </block>
```

The file should be the name corresponding to your ***logic.*** Here, it is ***boundtop.net***. It should contain information of inputs, outputs, and globals. You have to list all IOs' information here. Meanwhile, please follow the format and use ***tab*** or ***\t***. In line 2, it is only one ***\t***. In line 3, it is ***\t\t.*** There is no need to use enter/return while listing IOs in inputs/outputs/globals. A single line, even it is a very long line, would be good. Once you finish the file, please save it in folder ***module*** under your logic task folder, which contains run1,run2, etc., as file name ***logic.net***.
After all above setting, you can then skip top-module synthesis by editing ***m_fpga.pl*** line ***43, 44, 46***:
change
system("cp vtr_flow/tasks/$mainname/config/zconfig.txt
vtr_flow/tasks/$mainname/config/config.txt");
to
#system("cp vtr_flow/tasks/$mainname/config/zconfig.txt
vtr_flow/tasks/$mainname/config/config.txt");

change
system("vtr_flow/scripts/run_vtr_task.pl $mainname");
to
#system("vtr_flow/scripts/run_vtr_task.pl $mainname");

change
system("cp
vtr_flow/tasks/$mainname/run$run_num_top/$arch/$mainname.v/$mainn
ame.net vtr_flow/tasks/$mainname/module/$mainname.net");
to
#system("cp
vtr_flow/tasks/$mainname/run$run_num_top/$arch/$mainname.v/$mainn
ame.net vtr_flow/tasks/$mainname/module/$mainname.net");

Thanks.
This manual is written by Yi-ChungChen and Fubing Mao.
If you have any problem, please contact:

FubingMao, FMAO001@e.ntu.edu.sg
Yi-Chung Chen, YIC63@pitt.edu

Project member:
Fubing Mao, Wei Zhang, Yi-chung Chen, Hai (Helen) Li, Bingsheng He.

Please cite the following paper if it is used.

Fubing Mao, Yi-Chung Chen, Wei Zhang, Hai (Helen) Li, and Bingsheng He. Library-Based Placement and Routing in FPGAs with Support of Partial Reconfiguration. ACM Trans. Des. Autom. Electron. Syst (TODAES). May 2016.

04/10/2016 Version 0.3