MPACK 1 Scripts User Manual

Version 1.0

Xifan Tang

xifan.tang@epfl.ch

Supervisor:

Dr. Pierre-Emmanuel Gaillardon

pierre-emmanuel.gaillardon@epfl.ch

and Prof. Giovanni De Micheli

giovanni.demicheli@epfl.ch

Laboratory of Integrated Systems(LSI), EPFL

April 30, 2014

Contents

I. Motivation

MPACK1 scripts are developed to integrate MPACK1 into VPR7 (Versatile Place & Route) CAD flow. A Perl script, *mpack1_mix_flow.pl*, manages the MPACK1 flow shown in Fig. 1. The Perl script, connecting ABC (Logic Synthesis Tool), MPACK1 (Matrix-based BLE Packer), VPR 7 and another Perl script, *m2net.pl*, generates a table (CSV file) containing area, delay and power of a given matrix-based cluster architecture.

In Fig. 1, input circuit is first synthesized by ABC with logic optimization, then MPACK1 packed the circuits into matrix-based BLEs (*Basic Logic Elements*). AA-Pack clusters the matrix-based BLEs into CLBs (Configurable Logic Blocks).

Output matrix-based blif file of MPACK1 cannot be directly supported by VPR7 timing analysis engine because VPR7 requires more detailed architecture description and mapping information inside each matrix-based BLE. Perl script, *m2net.pl*, aims to fit the results from MPACK1 to VPR7 timing engine through (1) auto-generating the architecture description of matrix-based clusters with given settings, (2) inject the internal interconnections inside packed matrix-based clusters into the netlist (*.net) generated by AA-Pack. Works from m2net.pl are bold in Fig. 1.

MPACK1 outputs two blif files, one (called as matrix-blif in Fig. 1) without internal interconnection information inside matrix-based BLEs, the other (called as full-blif in Fig. 1) providing full internal interconnection information insides matrix-based BLEs. Besides, MPACK1 outputs a report file including the matrix architecture and internal connections of each matrix-based BLE. Perl script, *m2net.pl*, generates the architectural descriptions based on the matrix-blif. AA-Pack uses the matrix-blif to pack matrix-based BLEs into CLBs, and outputs results into a netlist (matrix netlist in Fig. 1). Perl script, *m2net.pl*, generates the detailed architectural descriptions based on the full-blif and MPACK1 report, and then inject the internal interconnections into the netlist from AA-Pack. To conduct power estimation, activity estimator (ACE 2.0) reads in the full-blif and outputs the activities file (*.act). VPR7 reads the full-blif, the netlist and detailed architecture description from *m2net.pl*. VPR7 produce area, timing and power of a hypothesis FPGA consisting of matrix-based clusters.

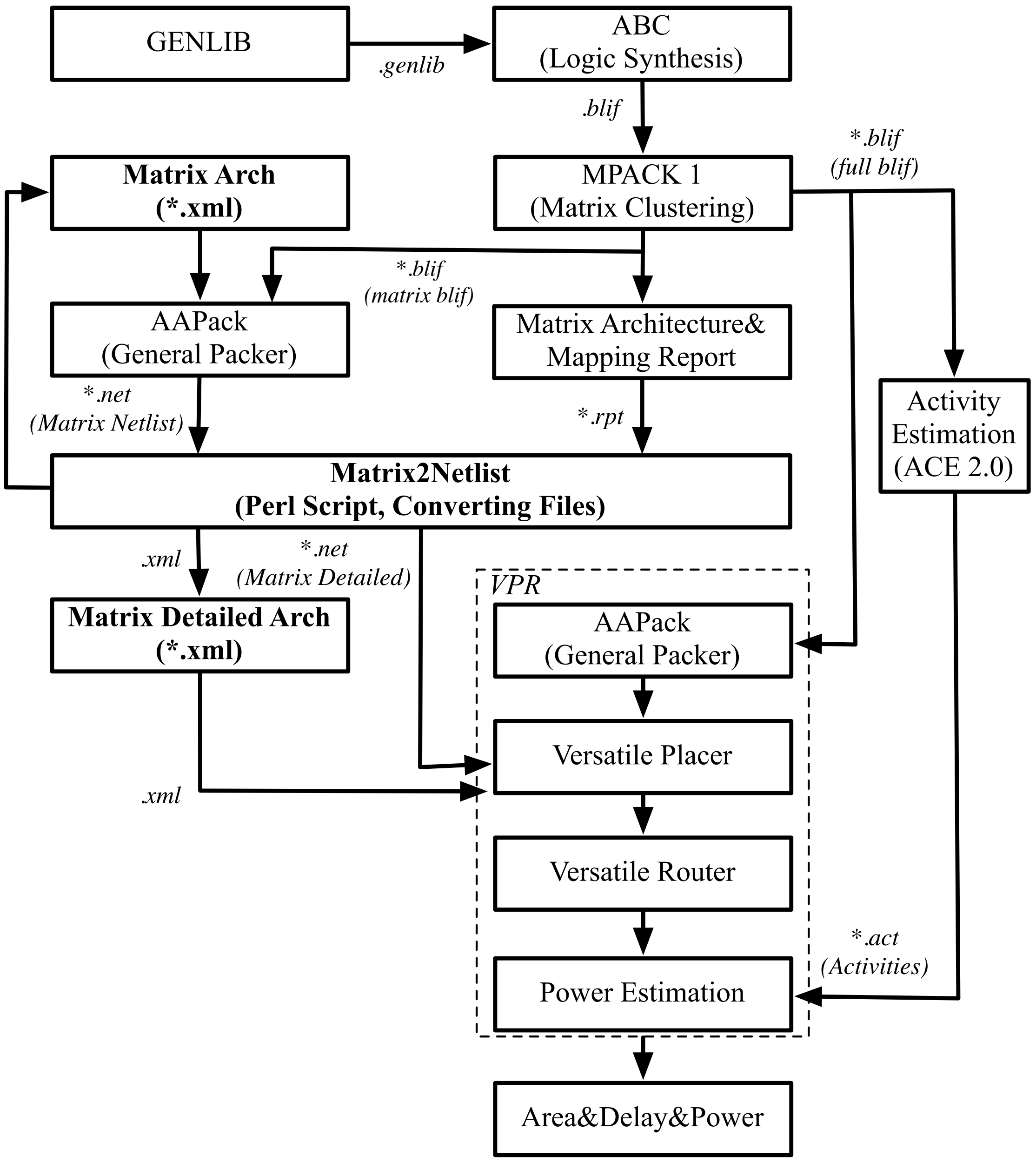


Fig. 1. MPACK1 flow

II. Perl Script m2net.pl

m2net.pl is configured by command-line options and a configuration file. The first part introduce the command-line options and the second part introduce the configuration file. In the thrid part, we introduce the file format of MPACK1 report. The final part describe the internal flow of m2net.pl.

A. *Command-Line Options*

Perl script, m2net.pl, are called twice in the MPACK1 flow. In the first call, it generates matrix architecture file, configured in the mode of *pack_arch*. In the second call, it generates the detailed architectural description and the netlist including internal interconnections of matrix-based clusters, configured in the mode of *m2net*.

The help desk of m2net.pl is called by "perl m2net.pl -help".

```
Help Desk:
  perl m2net.pl [-options <value>]
    Mandatory options:
      -conf : specify the basic configuration files for m2net
      -mpack1_rpt : MPACK1 report file
      -mode <pack_arch|m2net> : select mode
        1. pack_arch : only output XML architecture file for AApack
        2. m2net : output XML architecture file for VPR, convert *.net file for VPR
      -N : Number of MCluster inside a CLB
      -I : Number of input of a CLB
    Mandatory options for -mode pack_arch:
      -arch_file_pack : filename of output XML format architecture file for AAPack
    Mandatory options for -mode m2net:
      -net_file_in : filename of input *.net file from AAPack
      -net_file_out : filename of output *.net file for VPR
      -arch_file_vpr : filename of output XML format architecture file for VPR
    Other Options:
      -power : run power estimations in MPACK1 flow
      -remove_designs : clear the previous results and ensure a clean start
      -debug : debug mode
      -help : print usage
```

B. *Configuration file*

In this section, an example configuration file is shown. The configuration file consists of three parts, *arch_model*, *arch_device*, and *arch_complexblocks*, which determine critical parameters in FPGA architecture description file. Use "#" for comments.

```
# Technology PTM 22nm CMOS
[arch_model]
matrix_model_name = MATRIX # Model name of matrix in matrix-blif file.
matrix_inport_name = I # Input port name of matrix in matrix-blif file.
matrix_outport_name = O # Output port name of matrix in matrix-blif file.
cell_model_name = cell # Model name of cell inside matrix in full-blif file.
cell_inport_name = I # Input port name of cell inside matrix in full-blif file.
cell_outport_name = O # Output port name of cell inside matrix in full-blif file.

[arch_device] # For more information, please refer to VPR7 User Manual.
R_minW_nmos = 8926
R_minW_pmos = 16067
ipin_mux_trans_size = 1.22226
C_ipin_cblock = 1.47e-15
T_ipin_cblock = 7.247e-11
grid_logic_tile_area = 1747.20
mux_R = 551
mux_Cin = 7.7e-16
mux_Cout = 3.488e-15
mux_Tdel = 5.8e-11
mux_trans_size = 2.63074
mux_buf_size = 27.645901
segment_length = 4
segment_Rmetal = 18.18
```

segment_Cmetal = 3.54e-15

[arch_complexblocks]

CLB_logic_equivalent = true

matrix_delay = 1.30e-10

cell_delay = 1.30e-10

dff_tsetup = 6.6e-11

dff_tclk2q = 1.24e-10

mux2to1_delay = 1.5e-11

C. *MPACK1 Report Format*

MPACK1 report format describes detailed information about matrix-based BLE architecture and internal interconnections inside each matrix-based BLE.

# Report sample of MPACK1

matrix_width = 3

matrix_depth = 3 # can also be regarded as layer number + 1

cell_size = 2

# Interconnection scheme for each layer

# Detailled interconnection scheme for each layer valid only for 2 input cells

layer_detailed

X[3].I[0] = 0

X[3].I[1] = 1

X[4].I[0] = 0

X[4].I[1] = 2

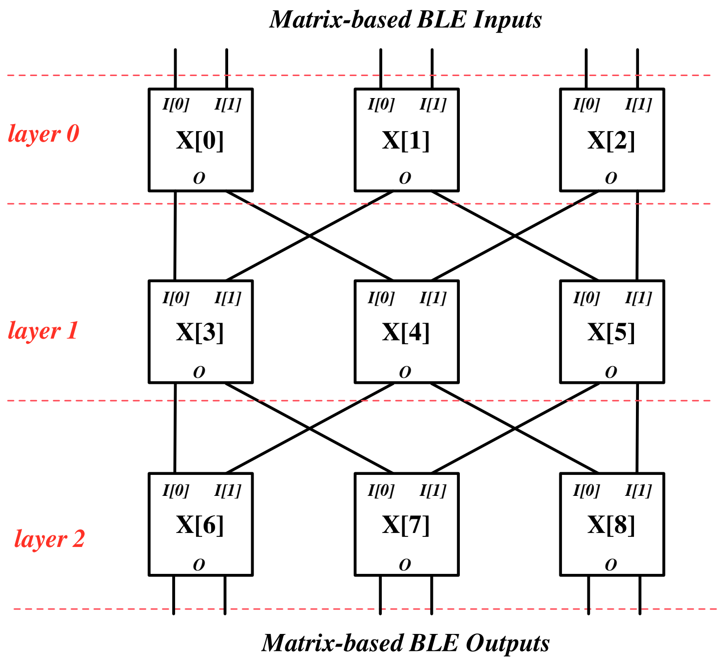
X[5].I[0] = 1

X[5].I[1] = 2

X[6].I[0] = 3

X[6].I[1] = 4

X[7].I[0] = 3



```
X[7].I[1] = 5
X[8].I[0] = 4
X[8].I[1] = 5
End

# MCluster number
mclusters_number =59 # Number of MCluster in the circuit

# BLE connectivity, this is an example for describing ONLY ONE BLE
# cell[i][j] = <unconn|conn>*cell_size,<net_name(output)>
# Both input are unconn implies this cell is un-used
# Require output nets to match corresponding matrix in *.net file generated by AAPack
# For unused cell, we don't care the net name, instead use keyword "open"
MCluster
cell[0][0] = conn,unconn,buf580
cell[0][1] = conn,conn,n50
cell[0][2] = conn,unconn,buf581
cell[1][0] = conn,conn,c&lt;0&gt;[0]
cell[1][1] = unconn,unconn,open
cell[1][2] = conn,conn,n53
cell[2][0] = conn,unconn,buf582
cell[2][1] = unconn,conn,buf583
cell[2][2] = unconn,unconn,open
end
```

D. *Perl Script m2net.pl Flow*

Recall that m2net.pl has two modes, one (called “pack_arch”) and the other (called “m2net”).

Mode “m2net” is the core of m2net perl script. In this part, additional information about internal interconnections inside matrix-based BLEs (In MPACK1 report, it is called

MCluster) is imported into the net file from AA-PACK, which enables VPR to conduct timing analysis. Each MCluster in MPACK1 report has unique net names. For example, in the sample file of MPACK1 report, net names of MCluster are "buf582 buf583 open". In the net file from AAPack, the net names of MCluster is presented as "<port name="O">buf582 buf583 open</port>". Each time we find matched net names, we add information about internal interconnections inside this MCluster.

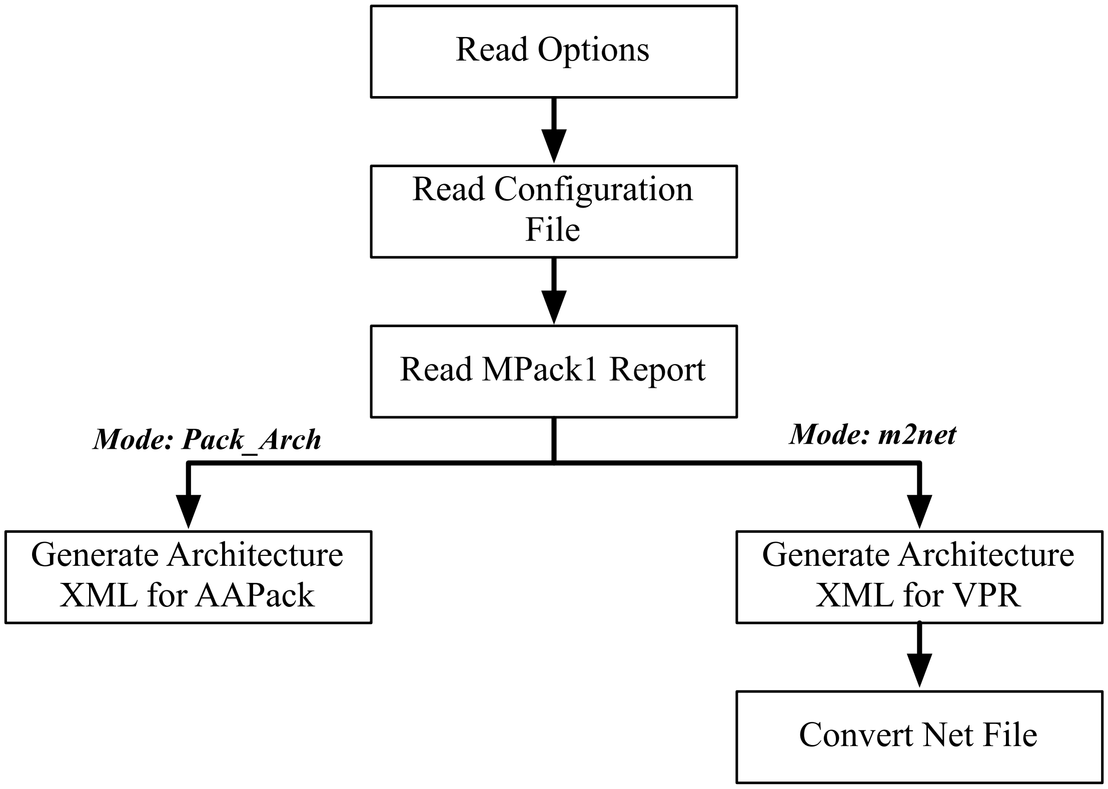


Fig. 2. M2net Perl Script flow chart

III. Perl Script mpack1_mix_flow.pl

Perl script mpack1_mix_flow.pl also are configured by command-line options and a configuration file. In the first part, we introduce the command-line options. In the second part, we introduce the configuration file. The third part reveals the outputs of this perl script.

A. *Command-line Options*

Help desk of mpack1_mix_flow.pl can be called by “perl mpack1_mix_flow.pl -help“.

Help desk:

```
mpack1_mix_flow [-options <value>]
Mandatory options:
-conf : specify the basic configuration files for fpga_flow
-benchmark : the configuration file contains benchmark file names
-rpt : CSV file consists of data
-N : N-LUT/Matrix
-I : Number of inputs of a CLB
Other Options:
-K : K-LUT, mandatory when standard flow is chosen
-M : M-Matrix, mandatory when mpack flow is chosen
-power : run power estimation oriented flow
-remove_designs : clear the results folder before start a new job
-debug : debug mode
-help : print usage
```

B. *Configuration File*

A configuration file consists three part, *dir_path*, *flow_conf* and *csv_tags*. In *dir_path*, we define the paths of all the tools in MPACK1 flow. In *flow_conf*, we define the paths of input files required in MPACK1 flow. In *csv_tags*, we define the results we want from log files of MPACK1, VPR and VPR Power.

```
# Sample Configuration Example, use “#” for comments
[dir_path] # Highly recommend absolute path
benchmark_dir = <path of your benchmark directory >
abc_path = <path of ABC logic synthesis tool> # should include execute file, i.e., abc
mpack_path = <path of MPACK1> # should include execute file, i.e., mpack
```

vpr_path = <path of VPR> ***# should include execute file, i.e., vpr***

rpt_dir = <path of directory storing results> ***# i.e., MPACK1_FLOW/RESULTS***

m2net_path = <path of m2net.pl> ***#i.e., ./m2net.pl***

ace_path = <path of activity estimation tool> ***# should include execute file, i.e., ace***

[flow_conf]

flow_type = mpack ***# FIXED TO MPACK***

std_vpr_arch = <standard VPR architecture> ***# Use relative path under VPR folder is OK, OUT OF DATE***

mpack_stdlib = <GenLib for ABC in MPACK1 flow> ***# Use relative path under ABC folder is OK***

m2net_conf = <M2NET_CONF/sample_m2net.conf>

power_tech_xml = <POWER_TECH_XML/sample.xml>

[csv_tags] ***# Results to be extracted from MPACK, VPR results. Use | as split***

mpack_tags = Global mapping efficiency:|efficiency:|occupancy wo buf:|efficiency wo buf: ***# Results to be extracted from MPACK1 log***

vpr_tags = Netlist clb blocks:|Final critical path:|Total logic delay:|total net delay:|Total routing area:|Total used logic block area: ***# Results to be extracted from VPR log***

vpr_power_tags = PB Types|Routing ***# Results to be extracted from VPR Power log***

C. *Benchmark configuration file*

To build a benchmark configuration file, you list all the circuit names, one per line. Keep in mind that all the blif files should locate in the ***benchmark_dir*** defined in the configuration file.

# Sample benchmark configuration file, use "#" as comments

add16_new.blif

...

spla.blif

##t481.blif

D. *Outputs*

mpack1_mix_flow.pl stores all the intermediate files in the ***result_dir*** defined in the configuration file. A independent folder is created under ***result_dir*** for each benchmark, for example, for benchmark spla, folder "spla" is created. Inside the independent folders, you can find log files and outputs of ABC, MPACK1, AA-PACK, and VPR.

mpack1_mix_flow.pl collects the results in a CSV file, which can be opened by Excel. This CSV file lists the results whose tags have been defined in ***csv_tags*** defined in the configuration file for all benchmarks defined in benchmark configuration file.