The Physical Logic Restructuring Project

Andy Fox

Summary

The work done on the physical restructuring project provides the infrastructure for remapping cuts. A cut is a collection of gates (informally called a blob in Open Road). A cut has a set of inputs (leaves) and outputs (roots).

Three pieces of work are provided:

1. Cut Signature Generation.
   - This takes a cut of logic (a cluster of gates) and generates an NPN signature (special form of truth-table) for each output of the cut. The typical application of this work is to generate a unique signature for a small (fewer than 10 inputs) cut of logic for look up in a library (or oracle) of alternative physical realizations. For example an AND/OR combination will have the same NPN signature as an NAND/NOR combination.
2. Oracle Generator
   - This takes in two netlists and produces a set of "difference" cuts The difference cuts correspond to the changes in the design. For each cut a verilog module is extracted and an NPN signature generated. The intended application of this work is to provide the data to learn what other tools have done with a design. The tool isolates the changes (finds common points between the designs and identifies the difference logic between the common points) and writes them out to an oracle file with an npn signature for each difference. The thinkig is that the differences are candidates for replacement in a design.
3. Timing driven physical synthesis remapping.
   - This is a source level integration of the abc package into the STA and its use to resynthesise and remap a cut of logic subject to real timing constraints. We build the abc network data structures from the timing analyzer data structures and invoke the abc mapper with timing constraints: inputs are constrained with arrival times and outputs with required times as needed by the abc mapper. The typical application of this work is to remap a portion of a logic design using more improved timing constraints. For example post floor planning more accurate timing constraints might be applied to a region of logic.

How to get the code

To get maple:
git clone git@github.com:andyfox-rushc/maple.git

To get the STA with NPN signature generation.

git clone git@github.com:andyfox-rushc/icbsta.git

Work is on dev branch.

How to build:
in icbsta directory:
mkdir build
cd build
cmake ../ -DABC_DIR=~/abc -DCMAKE_BUILD_TYPE=Debug

Set ABC_DIR to your abc release diretory. This should include libabc.a (do a make libabc.a in abc root dir)

How to run the code

npn signature generation:

Build and run the the google test in abclitetest/npnencode. This test shows how to invoke and decode the NPN encoding (truth table in, NPN out). The same code is used in the oracle generator.

```
g++ -g ../../abclite/extra/ExtraUtil.cc ../../abclite/kit/Kit.cc
../../abclite/npnencode/Npnencode.cc NpnEncodeTest.cc -I ../../abclite/extra
-I../../abclite/global -I../../abclite/kit -I../../abclite/npnencode
~/googletest/build/lib/libgtest.a -pthread
./a.out
```

oracle generation:

```
icb_verific -libread ../ua11lscef15bdrll_135c125_ss.lib
icb -cut_report -root aes_cipher_top -hier_vlog place.v -flattened_vlog place_opt.v
```

sta remapping: See file icbsta/examples/example2.tcl (need to copy nangate45_fast.lib to directory to run test)

```
read_liberty nangate45_fast.lib
read_verilog example2.v
link_design top
phys_remap nangate45_fast.lib

(optionally a script can be provided to run abc commands directly using:
-script “abc_commands” eg -script “strash; if -K 4; “
```

Directories and Code

Directory/File	Functionality
abclite	Subset of abc

abclite/extra	Utilities for abc
Abclite/global	
Abclite/kit	Truth table package for abc
Abclite/npnencode	NPN signature encoding
abclitetest	Google tests for abclite
restr	
restr/PhysRemap.cc	Physical remapping using abc “map” algorithm
restr/Cut2Ntk.cc	Build abc data structures from STA cut
Restr/Ntk2Cut.cc	Build Sta netlist from a cut
restr/Cutgen.cc	Sample of cut generation on sta
restr/Cut.cc	Cut representation
restr/FuncExpr2Kit.cc	Extract a truth table for an STA functional expression.
restr/LibRead.cc	Liberty reader (wrapper for abc reader)
restr/PhysTiming.cc/.hh	Physical timing interface: propagates timing from sta to abc. The arrival times for cut inputs and required times for cut outputs are annotated.

To Do (For Oct 17th).

Generate candidate cuts to optimize.
Run all oracle cases
Provide Cmake for google tests
Google tests for physical remapping

Proposed Timing Driven Generate Candidate Cut Algorithm:

Algorithm: ClusterCutGen :

```
1.  For each timing end point (eg memory, sequential element, output pad) {
2.  Search End Point List = Sort by negative slack
3.  while (Search End Point List != empty){
4.          end point = Search_End_Point_List. Head();
5.          Back_set = Walk back to timing end points;
6.          Front_set = Walk forward to timing end points;
7.          cutset = cutset + Form Cut (Back_set, Front_set);
8.          Delete cutset roots from Search End Point List;
9.  }
10.
11. Resultant cutset is set of non-overlapping regions of design to optimize.
12.
```