Floor Planner and Top Level Router

Phase 2

By: Engy Raafat, Habiba Gamal, Maha Moussa

Components

1) Floor Planner

2) Pin Assignment

3) Router

Abstract Steps

1) Macros are placed on the die to optimize their placement

2) Place the pins of the soft macros based on the interconnect to avoid congestion

3) Route the required nets

Inputs

- DEF for the hard macros
- LEF for the soft macros
- Technology LEF
- Verilog gatelevel netlist

SECOND PHASE

Floor Planner

- DEF Files parsing
- DEF File components section output
- Margins
- Optimization by Connection trials
- More Legalization
- Testing

Router Progress

- Technology LEF parsing
- Supports having n metal layers with direction that is extracted from TECH LEF during runtime
- Supports metal layers of different pitches hat are extracted from TECH LEF during runtime
- Supports multi-pin nets (was done last phase on 3 metal layers with known directions)
- Generates nets segment in the DEF

Pin Assignment

- Implemented this phase
- An important step between floorplanning and routing
- Connects pins of hard macros and then soft macros
- Input: gate level netlist and output of the floor planning and Tech Lef
- Output: x,y,z coordinates of each pin

What’s missing

- Parse gatelevel netlist to get interconnect information and I/Os
- Connect the floorplanner with the router (macros act as obstacles in all metal layers)
- Enhance format of nets section in DEF (e.g. place asterisk in place of repeated coordinates)
- Complete the DEF output
- Connecting data from the technology LEF to the floor-planner
- Output the I/Os in the grid

THE FINAL PHASE

DEF Parser

It gets a directory of DEF files and parses each file to get needed data for each hard maco:

- Name
- Die Area
- Pins and their info

Floor Planner Enhancements

Optimizes according to the following parameters:

- Total die area
- Interconnect between modules
- Connections between i/o pads and modules

The weights that each of these variables affect can be changed

Floor Planner

How modules are entered:

- Hard Macros: Directory of DEF files: (a text file with names and the files), parsed by the DEF parser
- Soft Macros: Specified file format for soft macros that contains name and area

```
.soft_module A 100
.soft_module B 200
.soft_module C 500
.soft_module D 400
.soft_module E 120
.soft_module F 600
```

Outputs

DEF file:

It outputs the Die Area, Components and pins Sections of the DEF file

Information file: It contains the utilization factor and the location of all modules

To the Router:

A grid that contains i/os and macros

Outputs

- To the pin assignment:

  A file that contains: Macros names, dimensions, locations and pin data in the following format:

```
arbiter         12800           20640           1               47684
pins:
                vdd             metal4          5760            120
                gnd             metal4          14240           120
                clk             metal3          20480           1600
                rst             metal2          14240           200
                request<0>              metal2          14560           12800
                request<1>              metal2          15040           12800
                request<2>              metal2          8960            12800
                request<3>              metal2          13440           12800
                request<4>              metal2          11200           12800
                request<5>              metal2          12000           12800
                grant<0>                metal2          4480            12800
                grant<1>                metal3          320             7600
                grant<2>                metal3          320             1600
                grant<3>                metal3          320             11600
                grant<4>                metal2          1920            12800
                grant<5>                metal3          320             5600
                select<0>               metal3          320             9600
                select<1>               metal2          1600            200
                select<2>               metal2          4480            200
                active          metal3          320             3600
```

Legalizations

- Adjustments with the grids
- Margins
- Soft Macros area

Pin Assignment

- Testing and debugging
- In previous phases, Assignment worked between soft macros or hard and soft macros only
- Used the floor planner output instead of user input
- Parsed a file that contains information needed from gate level netlist

Router

- Given 4 files, the router finds a path from a source pin to target pin(s) to connect them together.
- Our implementation does more DFS than BFS, so the program's runtime speed for big grids is good (until BFS must be done and DFS won't find a path)
- The input files are for example "test_macros", "test_pins", test_nets"
- In "test_macros", the coordinates and size of the macros is given (to be obstacles in the grid). The fourth file is the technology LEF
- In "test_pins", the length and width of the grid are given, as well as, the location of the pins from the pin assignment (to be obstacles in the grid).
- In "test_nets", the nets are given in terms of the pin names and their coordinates.

What was already done?

- DFS and BFS and finding a route.
- Support for multi-pin nets
- Support for n metal layers, without prior knowledge of the direction and pitch of the layer.
- Metal layers of different pitches
- Parsing the technology lef
- Outputting in a similar structure to the DEF file although was not entirely in the same format (was missing *)

What was done in this phase?

- The format outputted in the DEF exactly matches the standards.
- A log is outputted on the command line to report the routed nets and the unrouted nets. For the unrouted nets, there are two options:
  1) “Could not route the net” means that there was no route found between the source and any target
  2) “Could not route pin” means that no route was found to this pin, but other pins in the same net are routed.
- The macros are placed as obstacles on the grid
- Minor changes to support large grids
- Two major performance enhancements were done for large grids.

Router performance enhancement #1

- For multi-pin nets only.
- The program used to find a route to the nearest cell in the already formed route. (usually reached through both DFS and BFS)
- In this enhancement, the program tries to route to the nearest cell that can be reached by DFS only.
- The DFS now starts from a target pin till it reaches the nearest cell in this routed net <- ROUTE FOUND
- If this cell is found (which is usually the case), we have saved the runtime of flooding which is long for large grids.

Router performance enhancement #2

- For multi-pin nets ONLY
- When doing BFS, flooding is stopped if a routed cell that is a part of this net is reached.
- For this reason, when routing later pins in multi-pin nets, flooding starts from target

Assumptions & Future Work

- In the file “test_nets”, the nets are inputted in a similar structure to that outputted in the DEF. This is because we did not implement a verilog gatelevel netlist parser, so we could not obtain the interconnect information
- If we had more time, we would have implemented the Verilog parser.
- If we had more time, we would have researched and implemented more performance enhancements to reduce the runtime of BFS

Future work for the whole project

- Parsing gatelevel netlist to get interconnect information and area of soft macros
- Integrating the project’s 3 programs as one unified program