Documentació
Estiu 2023

Jordi Cortadella Fortuny

Sílvia Fàbregas Salazar
Izan Beltran Ferreiro

Informació de Contacte - Contact Information

Per qualsevol dubte, posar-se en contacte amb qualsevol dels següents correus electrònics.

For any inquiries, please contact any of the following email addresses.

izan.beltran@estudiantat.upc.edu
silvia.fabregas@estudiantat.upc.edu

Requisits

- **KaHyPar**.
  Instal·lar KaHyPar. En el moment en què hem redactat aquest informe, no ha estat possible clonar el repositori seguint les indicacions.

  Alternativament, fem el següent:

```
>     git clone --depth=1 --recursive https://github.com/kahypar/kahypar.git
>     mkdir build && cd build
>     cmake .. -DCMAKE_BUILD_TYPE=RELEASE -DKAHYPAR_USE_MINIMAL_BOOST=ON
>     make
```

  Ofereixen una versió precompilada de la interfície de Python que podem instal·lar amb la següent comanda:

```
>     python3 -m pip install --index-url https://pypi.org/simple/ --no-deps
kahypar
```

- **OpenROAD-flow-scripts**
  Compilen utilitzant Binaris Precompilats.

  Instal·lar dependències::

  - Klayout
    - Descàrrega
  - Yosys
    - Descàrrega

```
>     sudo apt update
>     sudo apt upgrade

>     wget
https://www.klayout.org/downloads/Ubuntu-22/klayout_0.28.12-1_amd64.deb
>     sudo apt install ./klayout_0.28.12-1_amd64.deb

>     wget
https://github.com/YosysHQ/oss-cad-suite-build/releases/download/2023-09-30/oss
-cad-suite-linux-x64-20230930.tgz
>     tar zxvf oss-cad-suite-linux-x64-20230930.tgz
>     export PATH="$(pwd)/oss-cad-suite/bin:$PATH"
```

```
>      wget
https://github.com/Precision-Innovations/OpenROAD/releases/download/2023-09-28/
openroad_2.0_amd64-ubuntu22.04-2023-09-28.deb
>      sudo apt install ./openroad_2.0_amd64-ubuntu22.04-2023-09-28.deb
```

Una vegada instal·lat, verifiquem la instal·lació:

```
>      git clone
https://github.com/The-OpenROAD-Project/OpenROAD-flow-scripts.git
>      cd OpenROAD-flow-scripts/

>      export OPENROAD_EXE=$(command -v openroad)
>      export YOSYS_CMD=$(command -v yosys)

>      yosys -help
>      openroad -help
>      cd flow
>      make
>      make gui_final
```

Requirements

- **KaHyPar**.
  Install KaHyPar. At the time this report was written, it was not possible to clone the repo using the indications.

  Instead, we'll use the following:

```
>      git clone --depth=1 --recursive https://github.com/kahypar/kahypar.git
>      mkdir build && cd build
>      cmake .. -DCMAKE_BUILD_TYPE=RELEASE -DKAHYPAR_USE_MINIMAL_BOOST=ON
>      make
```

  They offer a precompiled version of the Python Interface which can be installed via:

```
>      python3 -m pip install --index-url https://pypi.org/simple/ --no-deps
kahypar
```

- **OpenROAD-flow-scripts**
  Build Using Pre-built Binaries.

  Install dependencies:

  - Klayout
    - Download
  - Yosys
    - Download

```
>      sudo apt update
>      sudo apt upgrade

>      wget
https://www.klayout.org/downloads/Ubuntu-22/klayout_0.28.12-1_amd64.deb
>      sudo apt install ./klayout_0.28.12-1_amd64.deb

>      wget
https://github.com/YosysHQ/oss-cad-suite-build/releases/download/2023-09-30/oss
-cad-suite-linux-x64-20230930.tgz
>      tar zxvf oss-cad-suite-linux-x64-20230930.tgz
>      export PATH="$(pwd)/oss-cad-suite/bin:$PATH"
```

```
>      wget
https://github.com/Precision-Innovations/OpenROAD/releases/download/2023-09-28/
openroad_2.0_amd64-ubuntu22.04-2023-09-28.deb
>      sudo apt install ./openroad_2.0_amd64-ubuntu22.04-2023-09-28.deb
```

Once installed, the installation shall be verified:

```
>      git clone
https://github.com/The-OpenROAD-Project/OpenROAD-flow-scripts.git
>      cd OpenROAD-flow-scripts/

>      export OPENROAD_EXE=$(command -v openroad)
>      export YOSYS_CMD=$(command -v yosys)

>      yosys -help
>      openroad -help
>      cd flow
>      make
>      make gui_final
```

Traductor

Com fer-ne ús

La missió del traductor és convertir de format OpenDB propi d'OpenROAD, a format FPEF+DIEF, propi de FRAME, per tal de poder comparar els resultats obtinguts amb FRAME amb els de la sortida de l'eina RTL-MP d'OpenROAD (Hierarchical Macro Placement). Amb la finalitat de simplificar el disseny, s'agruparan per una banda les terminals en clústers segons la seva posició fixada als marges del dau i, per l'altra, els diversos macros i stdcells.

El primer pas consisteix a generar el fitxer .odb a tractar. Això es duu a terme fent la següent crida un cop dins el directori `OpenROAD-flow-scripts/flow`. Per fer-ho, hem d'escollir un dels dissenys que trobem a `OpenROAD-flow-scripts/flow/designs`. (O qualsevol altre que estigui en el format adequat i que sigui d'interès).

```
>     make DESIGN_CONFIG=<DESIGN_PATH/config.mk>
```

Un cop executat, els fitxers .odb generats es troben a `OpenROAD-flow-scripts/flow/results`.

Procedim a cridar el traductor (`odb_to_frame.py`) amb la següent comanda.

```
>     openroad -python <path a traductor>/odb_to_frame.py --design
path_to_odb
```

on el path_to_odb representa el path al qual es troba el fitxer .odb a traduir.

També es poden afegir els següents paràmetres: imbalance per kahypar (--imbalance), tipus d'optimització per kahypar (--optimization), path al kahypar (--kahypar_path), directori de sortida (--output_dir) i verbosa (--verbose).

La sortida del traductor consisteix en tres fitxers. Els dos primers representen el fitxer de la netlist amb els mòduls de tipus soft i les seves connexions i el fitxer del dau que conté les mesures del xip. El tercer recull els diversos clusters i els elements que hi pertanyen, per tal de fer possible la traducció inversa. Més específicament, un diccionari que relaciona cada element amb el cluster corresponent.

Estructura del flux de traducció

El flux de traducció s'organitza en un conjunt de tasques ordenades.

1. **Generació de clusters de terminals** - Bundled IOs. Inicialment, s'agrupen les terminals (*bterms*) segons la seva posició a les bores del dau. Per defecte se'n generen 12, però aquest valor pot ser alterat modificant el valor de la variable *num_bundled_IOs*. Aquesta funció està basada en la homònima d'OpenROAD.

2. **Obtenció de mesures del dau** - amplada (width) i alçada (height).

3. **Obtenció d'instàncies** - *stdcells* i *macros* - **i de connexions** - nets. Per tal de realitzar la partició d'elements, ens guardem tant la informació relativa als *macros* com un diccionari que relaciona el nom de cada instància amb el seu *node_id*. El primer és usat per tal de fixar cada *macro* a un clúster diferent. D'aquesta manera, cada clúster consistirà en un *macro* i múltiples *stdcells*. El segon té la funció de descodificar el resultat obtingut per la partició.

4. **Particionat de l'hipergraf**. Es genera la partició usant la interfície de Python de KaHyPar. La partició es genera si i només si hi ha *stdcells* en el disseny. En cas negatiu, cada macro representarà un mòdul de tipus *hard*; és a dir, de forma no modificable.

5. **Generació mòduls de tipus soft**. Obtinguda la partició, generem el diccionari de mòduls. Per cada mòdul, en calculem:
   - Àrea: suma d'àrees.
   - Centre: centroide de tots els elements.
   - Ratio d'aspecte: calculat per tal que, donada qualsevol deformació del mòdul de tipus *soft*, sempre hi càpiga el macro donades les seves mesures fixes d'amplada i alçada.

6. **Afegit de clústers de terminals**.

7. **Càlcul de connexions (nets) entre mòduls**. Es col·loca una aresta entre dos mòduls de tipus soft si existeix com a mínim una aresta entre algun element del primer clúster amb algun element del segon clúster. El pes de l'aresta és la suma de pesos de totes les arestes que existeixen entre els dos clústers. S'utilitza la mateixa metodologia de càlcul de noves connexions que a OpenROAD (veure mètode).

8. **Escriptura de sortida a fitxers**.

Informació més detallada del traductor i de la seva implementació es pot trobar a la documentació del codi. Per qualsevol dubte, no dubtar en posar-se en contacte.

Translator

How to employ it

The translator's aim is to convert from the OpenDB format used in OpenROAD to the FPEF+DIEF format employed by FRAME. This conversion is necessary so as to
compare the results generated by FRAME and those produced by the RTL-MP tool in OpenROAD (Hierarchical Macro Placement). In order to simplify the design, on the one hand terminals are clusterized based on their position relative to the die margins, and on the other hand, the macros and stdcells.

The first step consists in generating the .odb file. The following call is done once inside the OpenROAD-flow-scripts/flow directory. In order to perform such action, we ought to choose between one of the designs in `OpenROAD-flow-scripts/flow/designs`.

```
>     make DESIGN_CONFIG=<DESIGN_PATH/config.mk>
```

Once run, the generated .odb files are located inside OpenROAD-flow-scripts/flow/results.

Obtained the .odb file, you can invoke the translator from the terminal by entering the following command.

```
>     openroad -python <path to translator>/odb_to_frame.py --design
path_to_odb
```

where 'path_to_odb' stands for the path where the file .odb you wish to translate is stored.

The following parameters can also be added: imbalance for kahypar (--imbalance), optimization type for kahypar (--optimization), path to kahypar (--kahypar_path), output directory (--output_dir) and verbose (- - verbose).

The output generated by the translator comprises three files. The first two files consist of the netlist file, containing the soft modules and their connections, and the die file, which includes the die measurements - height and width. The third file collects the cluster information and the elements they contain, so as to allow the inverse translation. More specifically, a dictionary that associates each element with its corresponding cluster.

Translating Flow Structure

The translation flow is structured as a series of ordered tasks.

1. **Cluster Generation of Terminals** - Bundled IOs. Initially, terminals (bterms) are grouped according to their position on the die edges. By default, 12 of them are generated, but this value may be modified by changing the value of the variable *num_bundled_IOs*. This function is based on the OpenROAD counterpart.

9. **Obtaining die measurements** - width and height.

10. **Obtaining instances** - *stdcells* and *macros* - **and connections** - nets. In order to perform the partition, we shall store all the *macro*-related information and a dictionary that maps the name of each instance to its *node_id*. The first one is to assign each *macro* to a different cluster. Thus, each cluster shall consist of one *macro* and multiple *stdcells*. The second one serves to decode the results produced by the partitioning..

11. **Hypergraph partitioning**. The partition is generated using the Python Interface of KaHyPar. The partition is generated if and only if there are *stdcells* in the design. Otherwise, each macro will represent a *hard* module; in other words, they will be non-modifiable in shape.

12. **Generation of soft modules**. Once obtained the partition, we create the module dictionary. For each module, the following properties are computed:
    - Area: sum of areas.
    - Center: centroid of all elements.
    - Aspect ratio: calculated to ensure that, given any deformation of the soft module, the macro always fits within its fixed width and height measurements.

13. **Addition of terminal clusters**.

14. **Computation of new connections (nets) between modules**. An edge is places between two soft modules if there is at least one edge between any element of the first cluster and any element of the second one. The weight of the edge is the result of summing the weights of all the edges that exist between the two clusters. The same methodology for calculating new connections as in OpenROAD is used (see method).

15. **Output writing to Files.**.

More in detail information about the translator and its implementation can be found in the code documentation. For any questions or inquiries, please feel free to reach out for assistance.

Altres - FRAME

- S'han afegit a FRAME certes modificacions per tal de permetre l'ús de terminals. Tanmateix, queda realitzar canvis a les últimes etapes de FRAME (rect, legalfloor), on es produeixen errors si hi ha elements d'àrea zero, com és el cas de les terminals.

- Encara s'ha de considerar com es durà a terme la comparativa entre FRAME i RTL-MP. Obtinguda la sortida, s'han de desfer els clústers i possiblements ubicar cada element per tal de fer un càlcul acurat de la llargada de cable.

- Hi ha un segment del codi de glbfloor que de tant en tant no funciona. Fitxer: optimization.py. Funció: get_a(allocation, Module, cell_index).

  La següent divisió hauria de retornar sempre valors menors a 1, però degut a problemes de precisió, pot retornar valors molt lleugerament superiors a 1 (i.e. 1.0000000000000042):

```
        return alloc_rect.area_overlap(module_rect) / alloc_rect.area
```

- Per aquests models, és de vital importància realitzar un bon ajust dels hiperparàmetres. Tenir en ment les unitats amb què es tracta.

- S'hauria de canviar com s'escullen les possibles k a l'algorisme de forces, ja que els resultats depenen de les magnituds amb les quals es tracti.

- Afegir a draw.py possibilitat de no dibuixar les arestes.

Other - FRAME

- Certain modifications have been added to FRAME to enable the use of terminals. Nonetheless, changes still need to be made in the final stages of FRAME (rect, legalfloor), where errors might occur if there are elements with zero area, as is the case with terminals.

- The method for comparing FRAME and RTL-MP is yet to be determined. Once the output is obtained, the clusters need to be undone, and each element may need to be placed in order to perform an accurate cable length calculation.

- There is a segment of the glbfloor code that occasionally does not work. File: optimization.py. Function: get_a(allocation, Module, cell_index).
  The following division should always return values less than 1, but due to precision issues, may return values very slightly greater than 1 (i.e. 1.0000000000000042):

```
        return alloc_rect.area_overlap(module_rect) / alloc_rect.area
```

- For these models, it is vitally important to choose appropriate values for the hyperparameters.

- How the possible k's are chosen in the forces algorithm should be changed, since the results depend on the magnitudes being dealt with.

- Add to draw.py option to not draw edges.