CTM Creation in RedHawk-SC

Version: 2020.05.03

Table of Contents

1. Introduction

Technology scaling and architectural trends toward multi-die integration (PoP, Stacked-die SiP, and 3D-IC/TSV) call for increased thermal awareness in IC design. In deep sub-micron VLSI, temperature dependent leakage currents are a major source of power consumption. As power increases in the die, temperatures rise due to resistive loss and self-heating, which subsequently causes leakage power to increase further. This leakage thermal interaction can cause a thermal runaway resulting in chip failure. In addition to thermal runaway, temperature affects reliability. The current flowing through on-chip metal interconnect generates heat through resistive loss, making the design more susceptible to EM effects. Excessive temperature can significantly degrade the reliability of interconnect and devices, can even cause functional failures through electro-thermal coupling. Finally, in 3D stacked-die designs, the heat from neighboring chips also affects the final temperature distribution. Power-thermal integrity analysis becomes a critical step for multi-die SiP design.

To facilitate thermal analysis, RedHawk-SC can generate a thermal model of the chip called the CTM. The Chip Thermal Model (CTM) is essentially a temperature-dependent IC power consumption model.

To generate CTM, the power of the chip is calculated at the cell-level at multiple temperatures points. Then the design is partitioned into multiple regions. Each region is mapped with the power of the instances present in that region to form a “tile based” power-temperature table. This table captures the switching, self-heat and leakage power of instances as a function of temperature. In addition to the power-temperature data, the CTM also captures the layer-by-layer density of the interconnect geometries in the design for each region.

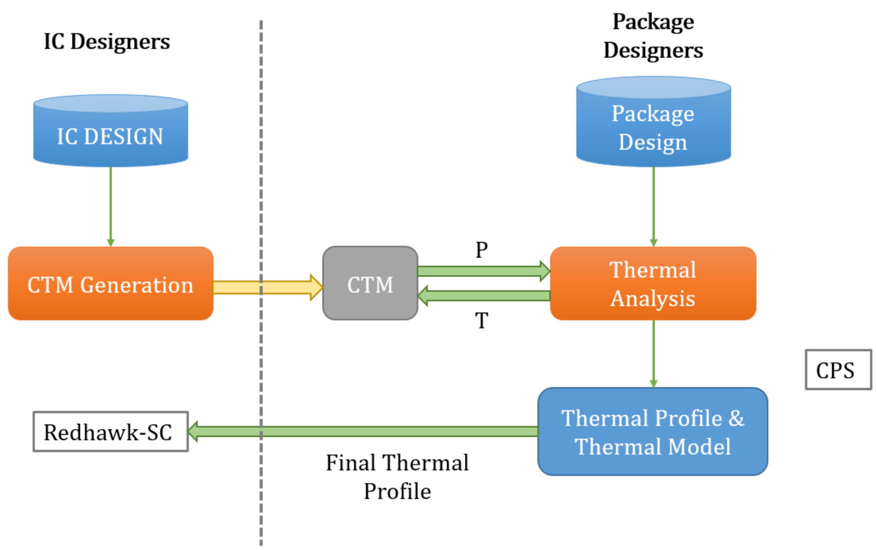


Figure 1 Chip-Package Thermal Analysis

2. CTM Creation Flows in RedHawk-SC:

Below Figure 2 shows the flow for creation analysis in RedHawk-SC.

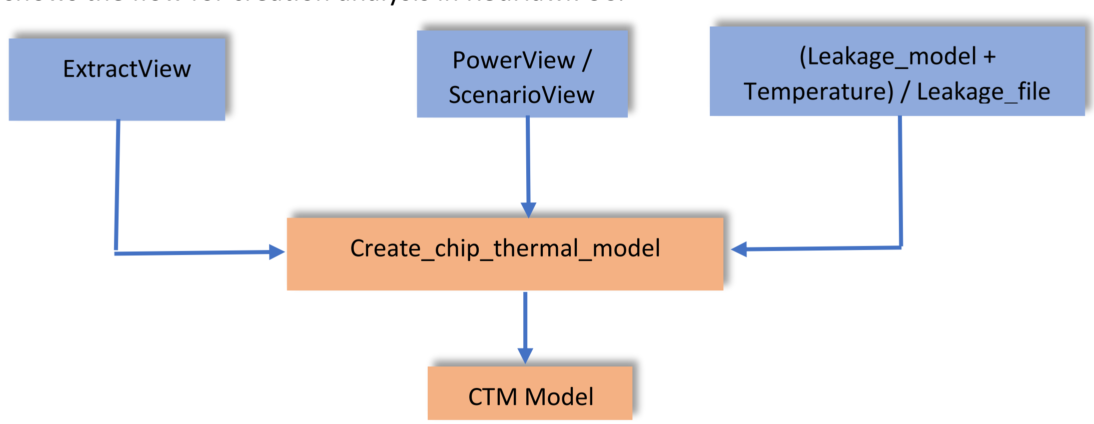


Figure 2 CTM Creation Flow in RedHawk-SC

2.1 Command for CTM Creation:

```
db.create_chip_thermal_model(ev, scn=None, pwr=None,
leakage_model='', leakage_file='', leakage_factor={},
temperatures=[], resolution=10.0, missing_cell_report='',
ctm_output_directory='', chip_id=-1, tag='ctm_files')
```

ARGUMENTS:

ev	Input ExtractView (type=ExtractView, required=True)
scn	Input ScenarioView (type=object, default_value=None)
pwr	Input PowerView (type=object, default_value=None)
leakage_model	Leakage model file name. The leakage model file is HJSON format binary file generated using apldi characterization (type=str, default_value='')
leakage_file	Leakage power file name (type=str, default_value='')
temperatures	temperature list to generate ctm power files as leakage_factor is given (type=list, default_value=[])
resolution	Size of the tile, the unit is um, the default is 10.0 (type=float, default_value=10.0, constraint="float > 0.0")
missing_cell_report	Missing cell report file name (type=str, default_value='')
ctm_output_directory	ctm_output_directory : The directory to output chip thermal model files (type=str, default_value='')
chip_id	The chip id for multichip cases, the default is -1 for singlechip cases (type=int, default_value=-1)
tag	Tag of this chip thermal model creation, to distinguish from other ctm creation in the same script (type=str, default_value='ctm_files')

Notes:

- Either scn or pwr must be provided to generate the power density files. If both inputs are provided, pwr has higher priority than scn.
- Either leakage_file or leakage_model must be provided to get the leakage scaling parameter information. leakage_model has higher priority than leakage_file if both given.
- The default of ctm_output_directory is cm_get_work_dir_name(). The zipped tar ball of ctm files will be created in the directory.
- If Instance Power File(IPF) is used to create PowerView, make sure that IPF contain leakage power information otherwise power result would not scale.
- CTM considers reference temperature from PowerView/ScenarioView. If temperature is not provided as input to PowerView/ScenarioView, RedHawk-SC considers default temperature=25C (from PowerView/ScenarioView) for CTM creation.

- Users are suggested to use below help function for latest/updated syntax.

  help([SeaScapeDB.create_chip_thermal_model], detailed=True)

3. Example Usage for creating CTM model:

3.1 Use APL Leakage_model as input:

```
leakage_model='./cells.leakage'
temperatures = [100.0, 115.0, 130.0, 145.0, 160.0]
db.create_chip_thermal_model(ev, pwr=pwr, resolution=50.0,
leakage_model=leakage_model, temperatures=temperatures, tag='ctm',
missing_cell_report='missing_cells_leakage_model')
or:
db.create_chip_thermal_model(ev, scn=scn, resolution=50.0,
leakage_model=leakage_model, temperatures=temperatures, tag='ctm',
missing_cell_report='missing_cells_leakage_model')
```

- leakage power at different temperature is computed based on exponential equation.
- The leakage model is in "HJSON" binary format created by apldi characterization.
- Necessary keywords part of apldi config file are:
  - *SWEEP_TEMPERATURE 125 115 130 145 160*
  - *OUTPUT_HJSON 1*

  OUTPUT_HJSON is required only for leakage_model creation, while SWEEP_TEMPERATURE is necessary for both flows.
- Default leakage scaling parameter defined inside leakage_model will be used for all the missing cells. All these cells can be reported using "missing_cell_report" argument.

3.2 Use APL LEAK file as input:

```
leakage_file = './cells.leakage'
db.create_chip_thermal_model(ev, pwr=pwr, leakage_file=leakage_file,
resolution=50.0,
tag='ctm',missing_cell_report='missing_cells_leakage_file')
or:
db.create_chip_thermal_model(ev, scn=scn, leakage_file=leakage_file,
resolution=50.0, tag='ctm',
missing_cell_report='missing_cells_leakage_file')
```

- In this flow Leakage scaling parameter are honored from the leakage file and power values from PowerView/ScenarioView.

3.3 Comparison of two input behaviors

Data Coverage	Leakage_file	Leakage_model
Cells covered	LIB is used to get the leakage power at the nom_temperature of LIB. Leakage power at nom_temperature is scaled for different temperatures depending on the scaling parameters derived from the APL leakage_file data corresponding to the cell under consideration	LIB is used to get the leakage power at the nom_temperature of LIB. Leakage power at nom_temperature is scaled for different temperatures depending on the scaling parameters derived from APL leakage_model file corresponding to the cell under consideration.
Cells not covered	LIB is used to get the leakage power at the nom_temperature of LIB. Leakage power at nom_temperature is scaled for different temperatures depending on the average scaling parameter derived from the APL leakage_file data of the cells covered in APL leakage file	LIB is used to get the leakage power at the nom_temperature of LIB and default leakage scaling parameter from leakage_model will be used for missing cells.

4. Output:

The output file is ctm_files.tar.gz. The default location is in gp folder.

Sample files in ctm_files.tar.gz:

CTM_header.txt

metal_density.ctm

power_T[1].ctm

power_T[2].ctm

power_T[3].ctm

power_T[4].ctm

power_T[5].ctm

CTM Information inside run.log:

INFO<CTM.116> Reference temperature used for CTM creation is 125.0 C.

INFO<CTM.117> CTM power (Watt) at different temperatures in 'ctm_leak_model'

```
+---------------+--------------+--------------+--------------+--------------+--------------+--------------+

Temperature  |   125.0    |   135.0    |   145.0    |   155.0    |   165.0    |   175.0    |

+---------------+--------------+--------------+--------------+--------------+--------------+--------------+
```

Total Power | 2.544550e+00 | 2.564152e+00 | 2.588496e+00 | 2.618730e+00 | 2.656277e+00 | 2.702908e+00 |

Leakage Power | 1.160229e-02 | 1.960229e-02 | 4.394665e-02 | 7.418024e-02 | 1.117277e-01 | 1.583585e-01 |

+---------------+--------------+--------------+--------------+--------------+--------------+--------------+

5. Thermal back annotation flow inside RedHawk-SC:

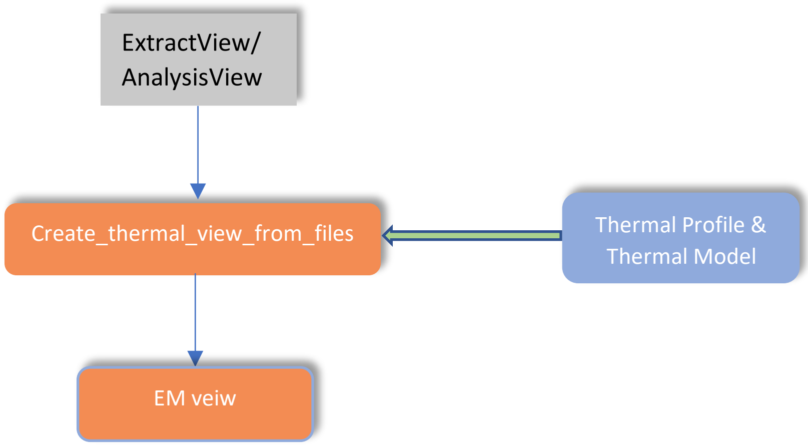
Thermal Analysis with CPS & RedHawk-SC comprises of three-steps

1) CTM Generation
2) Chip Package Thermal Analysis using CPS
3) RedHawk-SC EM signoff using back annotated Thermal profile from CPS.

In this section we will discuss about the RedHawk-SC thermal back annotation flow. As shown in figure1 the Thermal profile generated out of CPS (chip package system) Analysis can be read by RHSC for creating thermal view.

There are two different back annotation flows available in RedHawk-SC

Flow 1) EM analysis based on CTA (thermal_profile)

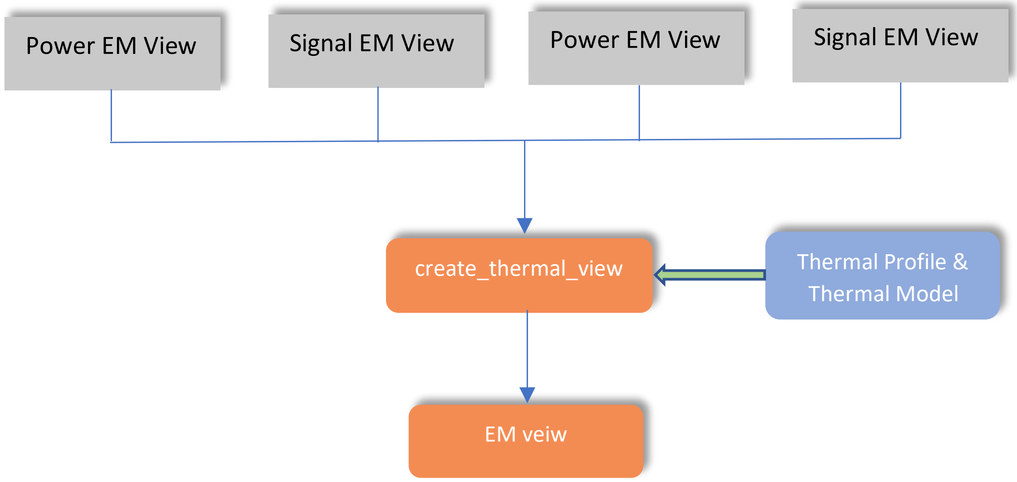



```
thmv_cta_ev = db.create_thermal_view_from_files(ev=[ev,ev_signal],
thermal_profile = thermal_profile, options=options,
tag='thmv_cta_ev')
```

```
thmv_cta_av = db.create_thermal_view_from_files(av_dynamic,av_sigem,
thermal_profile = thermal_profile, options=options,
tag='thmv_cta_av')
```

Flow 2) EM analysis based on CTA (thermal_profile) and SelfHeat flow

In this flow, RHSC considers the ambient temperature from chip thermal profile instead of temperature argument.

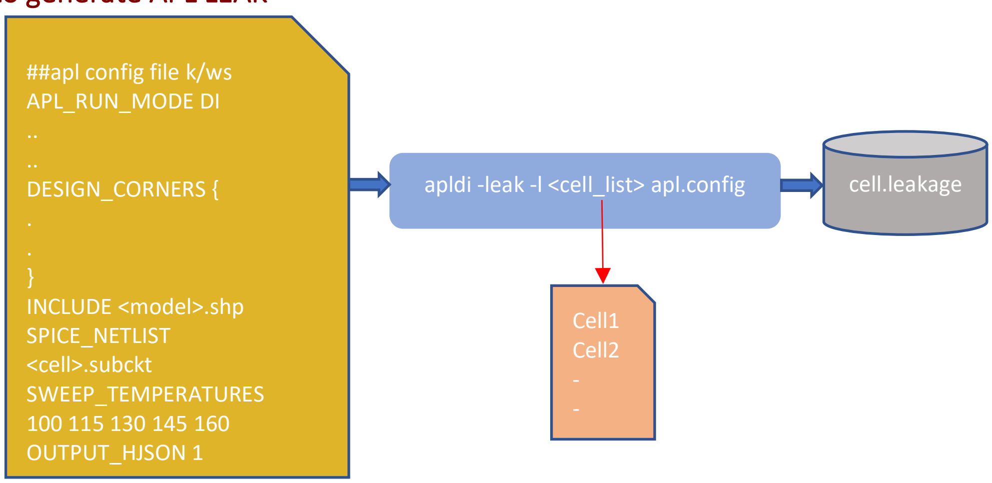



```
thermal_calculation_settings={'thermal_flow':1,
                              'thermal_profile': thermal_profile}

thmv = db.create_thermal_view(pem_rms, em_rms,sh_file, trf_file,
temperature_environment=125,thermal_calculation_settings=thermal_calc
ulation_settings, options=options, tag='thmv_flow1_cta')
```

APPENDIX

How to generate APL LEAK



- Add keyword SWEEP_TEMPERATURE to basic **apldi** configuration file, specify a list of characterization temperature samples, in degrees centigrade.

  Example:

  SWEEP_TEMPERATURE 100 115 130 145 160

- Run **apldi** as below:

  apldi -s 2 -l cell.list -leak -o cell.leakage apldi.conf

- OUTPUT_HJSON 1 is only required for generating leakage_model file.

Sample APL Leakage file

Below is the sample APL LEAK file for a cell DFQD1 with pins VDD, VPP, VSS and VBB

```
data_version cell.leakage 10v1
tool_name apl
version 2021 R1.RD  RHEL6
Released Date: 04/15/2020
data_tag asc 1586999450 Wed Apr 15 18:10:50 2020
file_signature TOP_FFGNP_LocalMC 125 1.15 2919760742 000000 FF
cell_temp 3 5
25.000000 50.000000 75.000000 100.000000 125.000000
DFQD1 1 4 VDD 25.000000 4.6555e-09 7.8063e-09 50.000000 1.13813e-08 1.94572e-08 75.000000
2.53732e-08 4.38237e-08 100.000000 5.19222e-08 9.01572e-08 125.000000 9.84649e-08 1.71311e-
07 VPP 25.000000 1.09791e-10 2.00089e-10 50.000000 1.62033e-10 2.98395e-10 75.000000
2.38718e-10 4.44693e-10 100.000000 3.52169e-10 6.63437e-10 125.000000 5.21452e-10 9.92085e-
10 VSS 25.000000 4.6921e-09 7.9301e-09 50.000000 1.14518e-08 1.96609e-08 75.000000
2.54975e-08 4.4151e-08 100.000000 5.21324e-08 9.06758e-08 125.000000 9.88123e-08 1.72126e-
07 VBB 25.000000 5.02748e-11 5.02728e-11 50.000000 6.85571e-11 6.85514e-11 75.000000
9.13421e-11 9.13296e-11 100.000000 1.18703e-10 1.18679e-10 125.000000 1.50654e-10 1.50614e-
10
AIOI21 1  4 VDD 25.000000 7.1601e-09 5.9335e-09 50.000000 1.82221e-08 1.44297e-08 75.000000
4.17855e-08 3.19564e-08 100.000000 8.71253e-08 6.50788e-08 125.000000 1.66945e-07 1.23174e-
07 VPP 25.000000 2.3672e-10 1.09982e-10 50.000000 3.45216e-10 1.62334e-10 75.000000
5.00852e-10 2.39182e-10 100.000000 7.26812e-10 3.52897e-10 125.000000 1.0592e-09 5.22669e-
10 VSS 25.000000 7.3797e-09 5.8772e-09 50.000000 1.855e-08 1.43754e-08 75.000000 4.22688e-
08 3.19166e-08 100.000000 8.78343e-08 6.50782e-08 125.000000 1.67986e-07 1.23257e-07 VBB
25.000000 1e-15 1.23029e-10 50.000000 1e-15 1.73211e-10 75.000000 1e-15 2.35444e-10
100.000000 1e-15 3.09704e-10 125.000000 1e-15 3.9569e-10
```

The file is generated by performing APL leakage characterization on the cell using the 'aplleak' utility. The first 8 lines of this file contribute to the header section. 7th and 8th lines specify the number of temperature values (5 in this case) and the actual temperature values used for generating APL LEAK (25 50 75 100 and 125 in this case). The file has a cell named DFQD1 in the first column. Second column has the code for the cell type. Third column defines the number of power/ground pins. Then a pattern repeats which consists of the pin name and the corresponding temperature value followed by 2 leakage current values. The first leakage current value is captured when the output is at 0 and the second when the output is 1 for each temperature.