Relative macro placement for faster physical design process

METRIC 2022

Mohit Sharma
Dr. Mohammed A. S. Khalid

Date: October 29th, 2022

University of Windsor
Department of Electrical and Computer Engineering

Outline

- Background and Motivation
- Main Contributions
- Methodology
- Experimental Results
- Conclusion and Future Work

Background and Motivation

- What are macro-cells? [1]
  - Modules or entities that represent certain design logic. Large blocks that can be viewed as black boxes.
  - Internal structural description may or may not be given although logic & electronic behavior is given. So, they have flexible geometries.
  - Much larger in size as compared to a standard cell.
- What are standard-cells?
  - Group of transistor and interconnect structures that provides a Boolean logic function or a storage function.
  - Standard cells in the circuit have the same height.

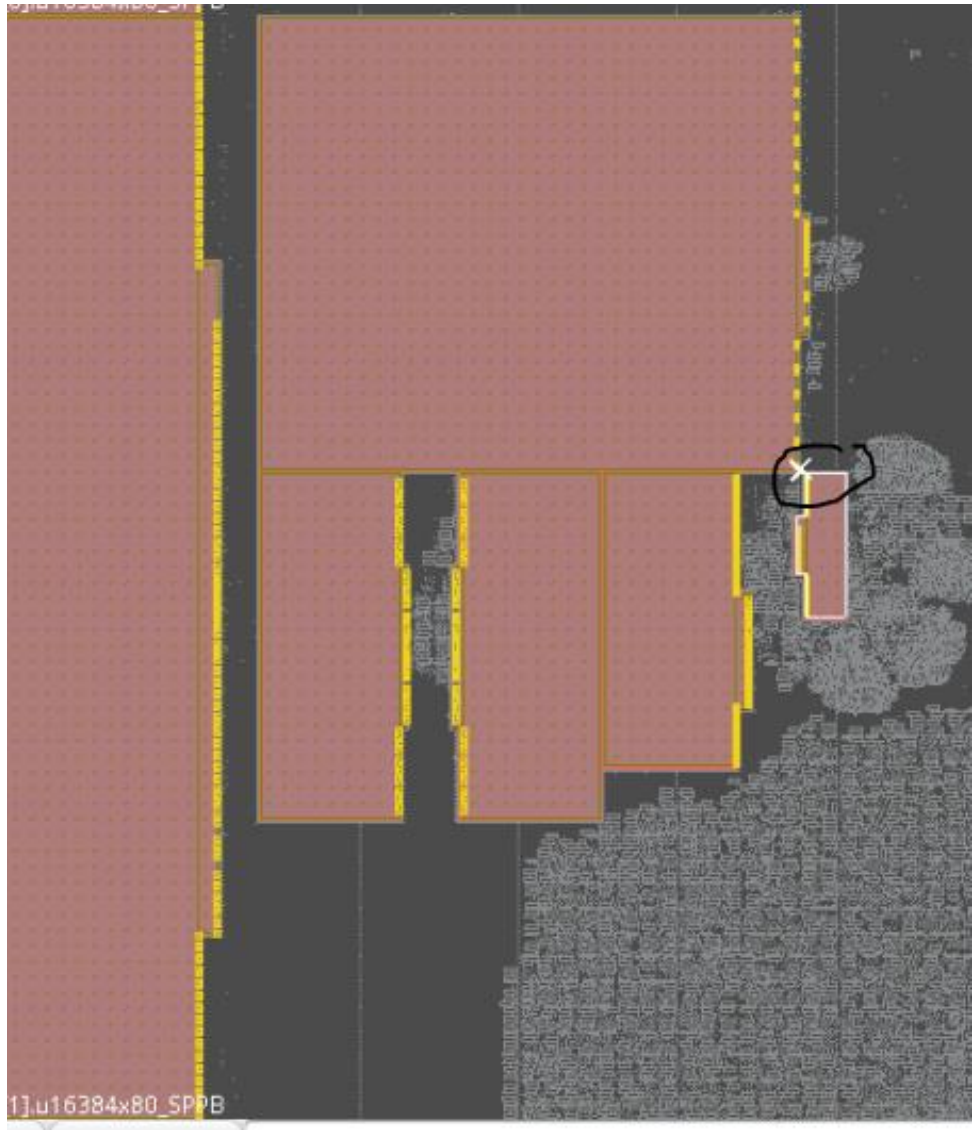


Figure 1: Macro Cells (Marron Rectangular blocks) and Standard Cells (Grey Clouds)

Background and Motivation

- What is macro-cell placement?
  - Problem of placing a given macro cell circuit to optimize a certain design objective e.g., wirelength.
  - A macro cell circuit contains a small number of macro cells and a larger number of standard cells.
  - Standard Cells need to be placed in specified rows.
  - No overlap is allowed between any two cells being placed.

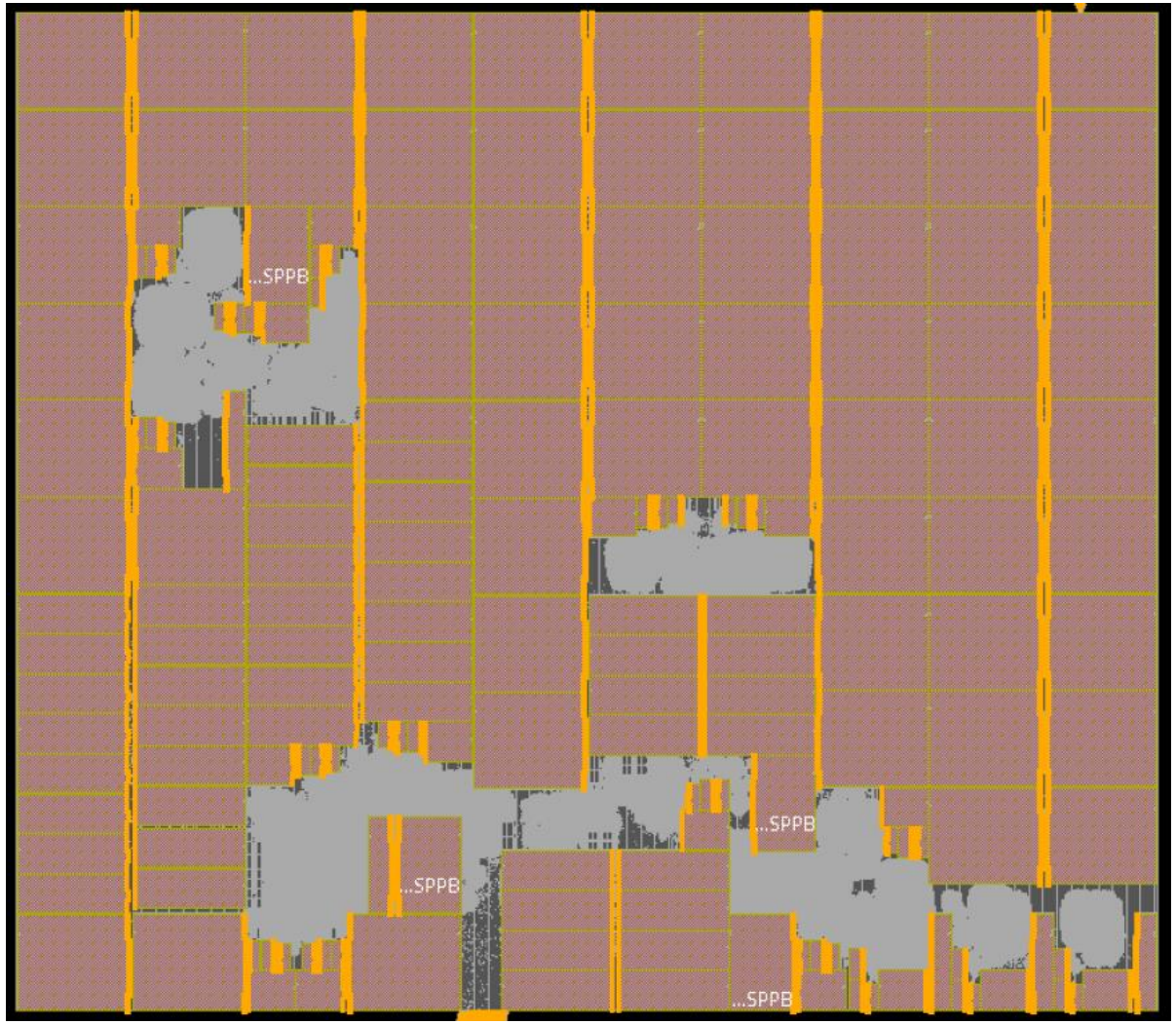


Figure 2: A placed macro-cell circuit

Background and Motivation

- Commercially available physical design CAD tool for automated macro placement give unoptimized placement [2]
- Placement done by automated macro placement tools have large Area, Wirelength and Power.

- Figure on the right show's placement achieved by a commercially available physical design CAD tool.
  - Communicating memories spread out leading to larger wirelength.
  - Underutilized core with lots of empty space resulting in larger area.

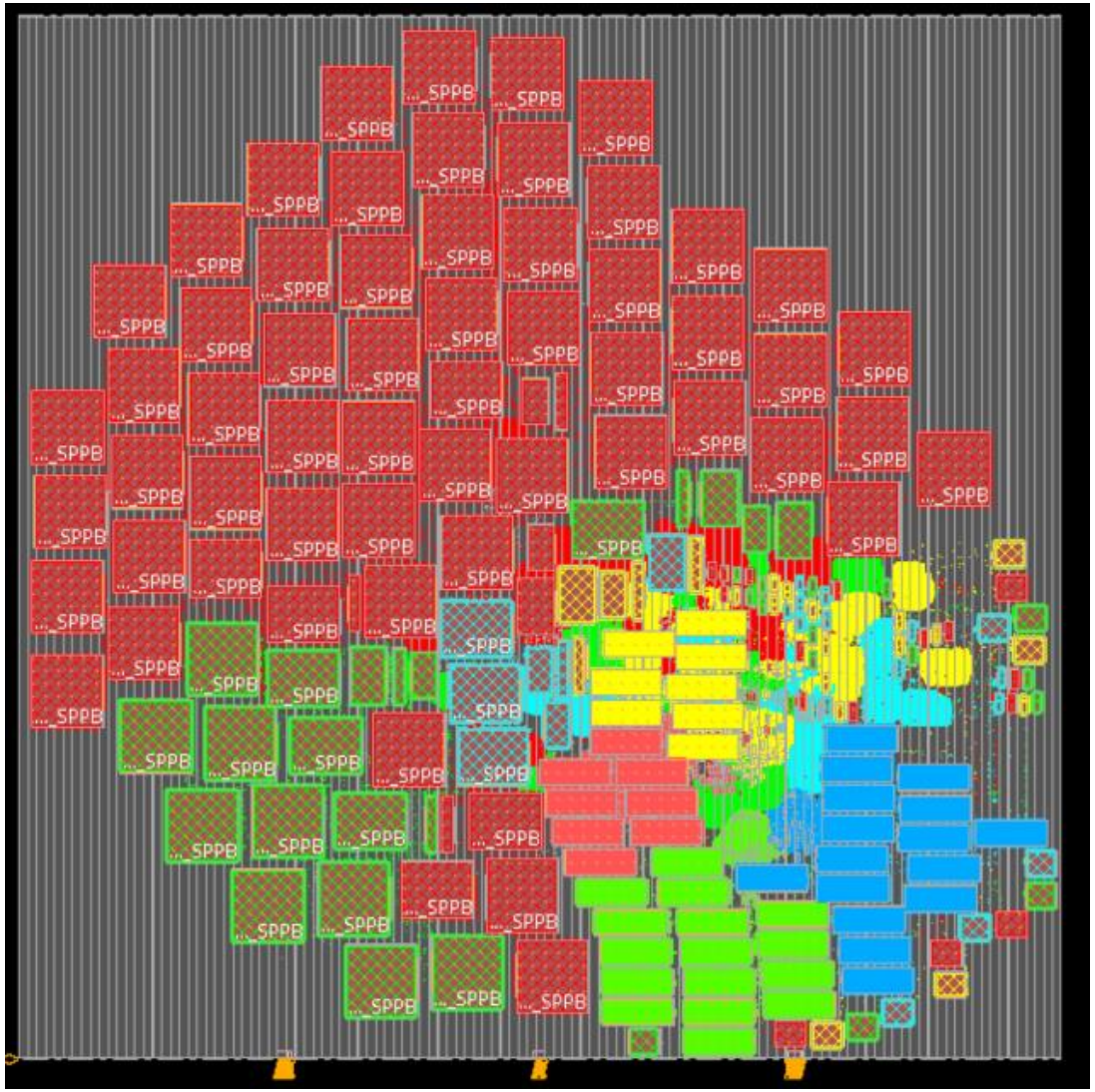


Figure 3: Placement achieved by a commercially available physical design CAD tool.

Background and Motivation

- Current research affirms that automatic macro placement doesn't achieve the most optimized metrics.
- Table shows results from a research on an automated macro-placement tool called Hierarchical Dataflow Placement (HiDaP) and compares the placement metrics achieved for eight circuits with Industrial EDA tool (IndEDA) and hand-crafted Floorplan (handFP)
- Hand-crafted Floorplan gave the best placement metrics, among the three methods utilized, for the greatest number of circuits out of the eight placed.

Circuit	Flow	Wirelength		Congestion	Timing	
		WL	Norm.	GRC%	WNS%	TNS
c1	IndEDA	13.19	1.029	6.51	0.0	0
520k cells	HiDaP	13.40	1.046	7.83	0.3	0
32 macros	**handFP**	12.81	1.000	7.36	-0.2	0
c2	IndEDA	46.01	1.180	12.99	-44.5	-931
3.95M cells	HiDaP	40.72	1.045	13.00	-19.0	-329
100 macros	**handFP**	38.97	1.000	9.33	-11.2	-213
c3	IndEDA	44.83	1.175	10.09	-75.5	-553
3.78M cells	HiDaP	35.02	0.918	8.29	-17.5	-260
94 macros	**handFP**	38.16	1.000	9.15	-17.8	-317
c4	IndEDA	45.03	1.174	7.24	-54.4	-2167
4.81M cells	HiDaP	40.43	1.054	4.94	-31.2	-2686
122 macros	**handFP**	38.35	1.000	3.33	-22.8	-1736
c5	IndEDA	44.25	1.162	2.02	-30.8	-1940
1.39M cells	HiDaP	39.51	1.038	4.72	-25.1	-1149
133 macros	**handFP**	38.06	1.000	3.42	-39.8	-1017
c6	IndEDA	96.42	1.288	9.95	-70.0	-15341
2.87M cells	HiDaP	79.20	1.058	2.22	-37.0	-5051
90 macros	**handFP**	74.87	1.000	1.63	-27.3	-3688
c7	IndEDA	41.44	1.174	38.56	-34.9	-1060
1.67M cells	HiDaP	35.52	1.007	6.47	-29.9	-1059
108 macros	**handFP**	35.29	1.000	4.61	-20.4	-774
C8	IndEDA	24.85	0.987	1.02	-3.4	-44
2.20M cells	HiDaP	23.75	0.944	1.37	0.0	0
37 macros	**handFP**	25.17	1.000	0.93	-3.9	-24

Table I: Table referenced from paper on RTL Aware Dataflow Driven macro placement [3].

Background and Motivation

- Hand placement gives best results most of the times, but it is slow process.
- 200 macros – 2 weeks
- What if this can be improved?
- Develop methods and techniques that reduce hand crafted placement time.

Main Contribution

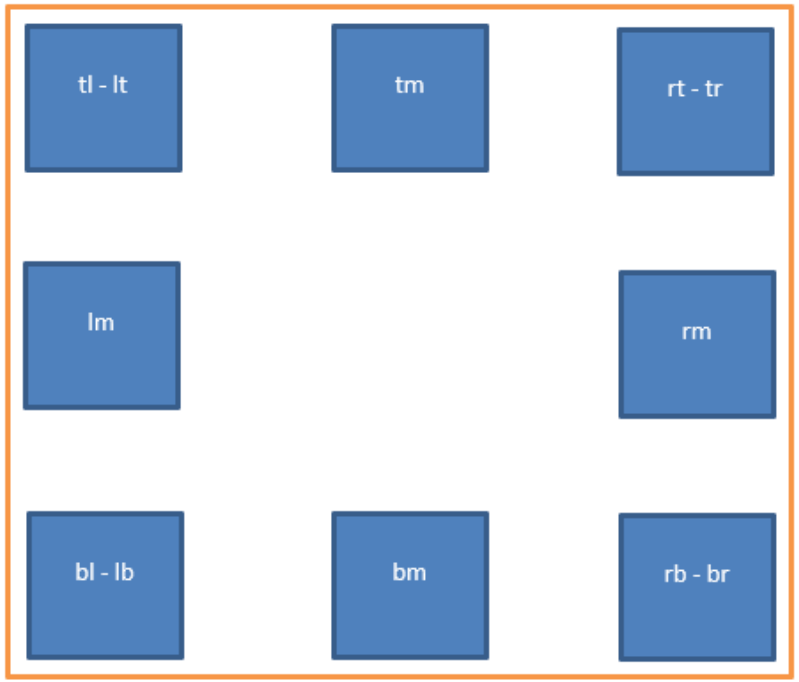
Relative **M**acro **P**lacement

- Simple Idea of relative placement.
- Builds upon placement of already placed macros.
- Analogous to building a puzzle.

Methodology

Relative Macro Placement

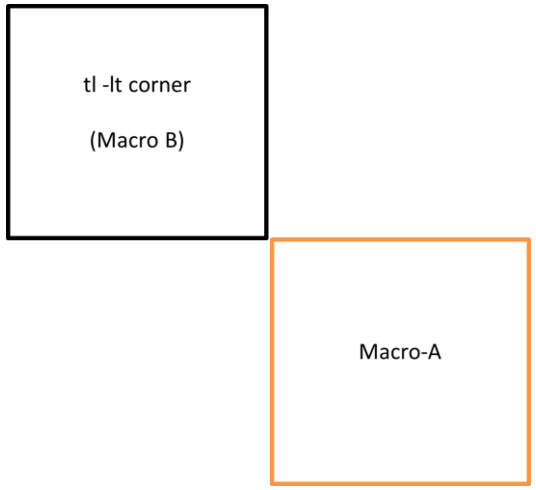
Macro placement can be



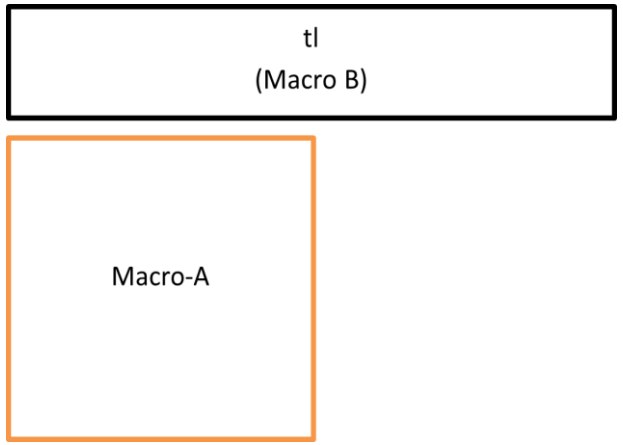
With respect to core

With respect to another macro

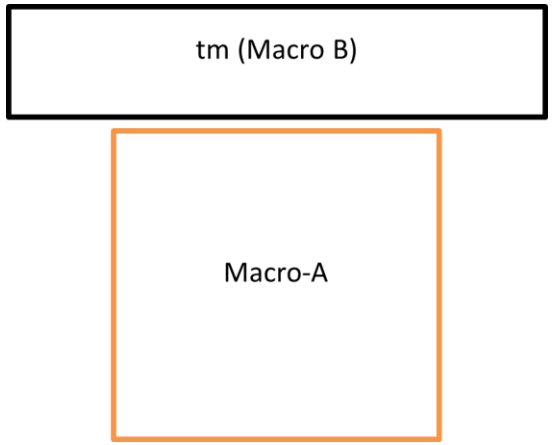
Relative placement applicable in both the scenarios



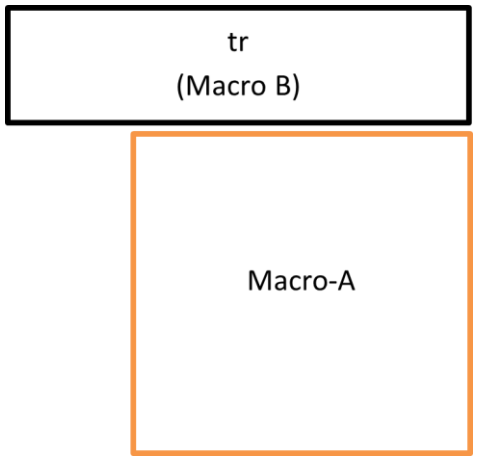
Top-left or Left-Top corner



Top-left

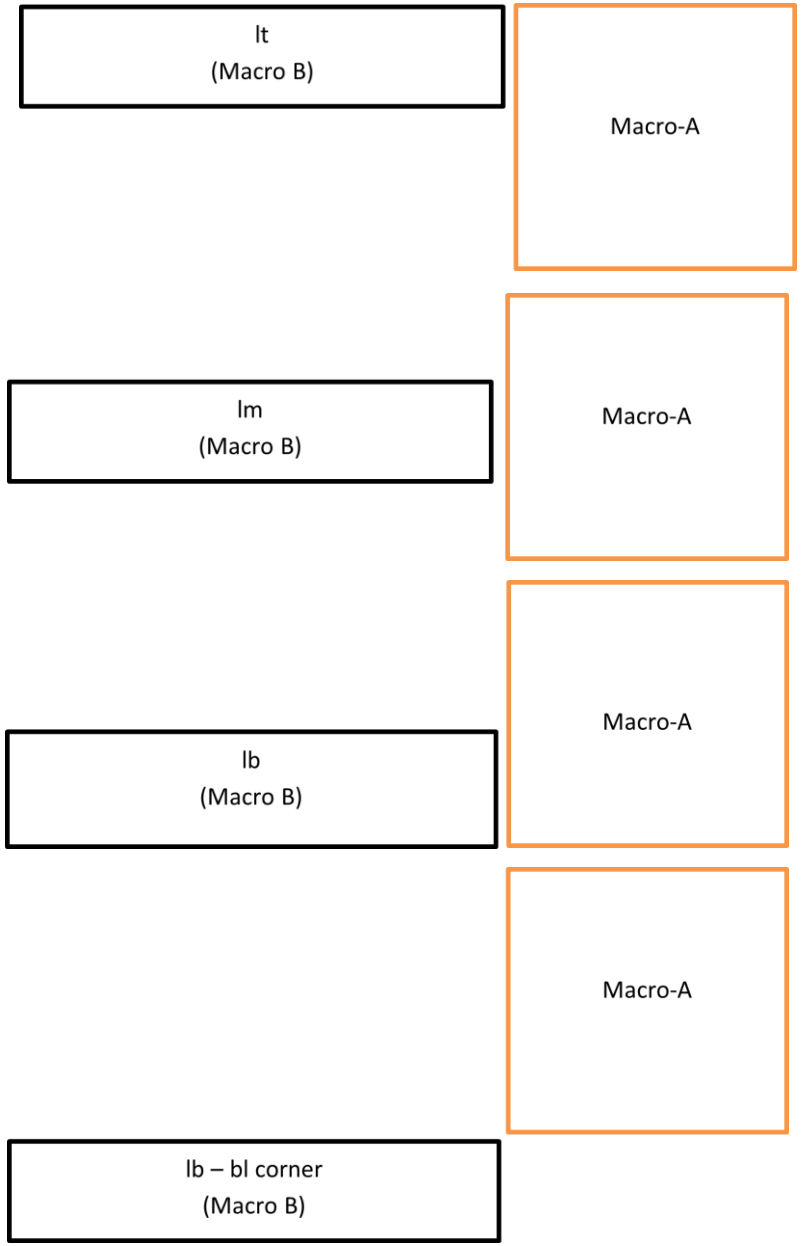


Top-middle



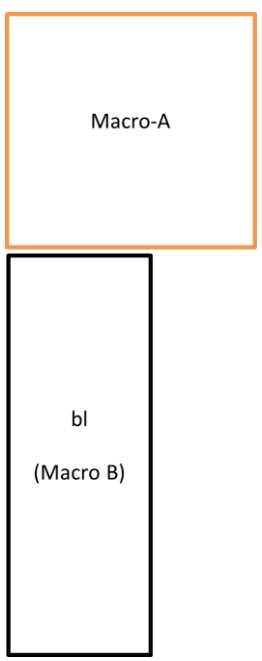
Top-right

Methodology

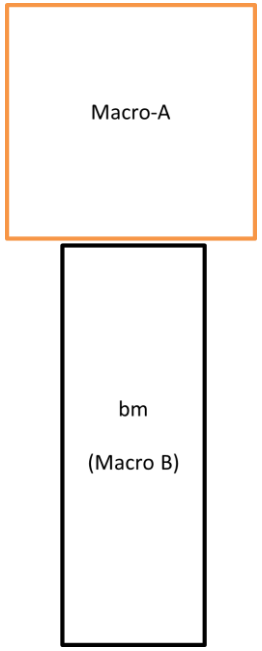


Methodology

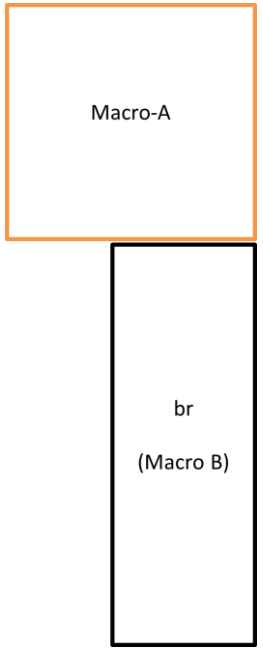
Methodology



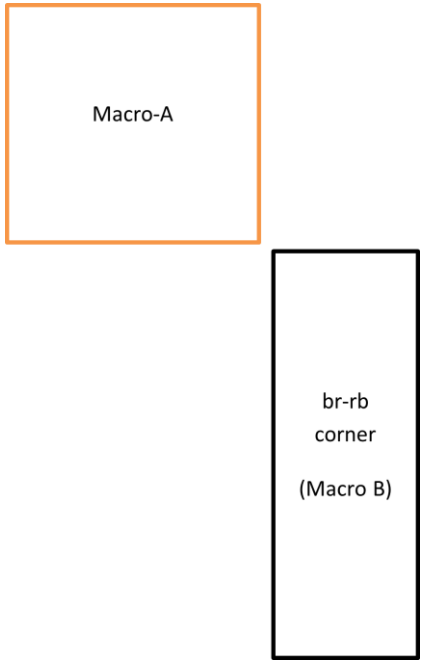
Bottom Left



Bottom Middle

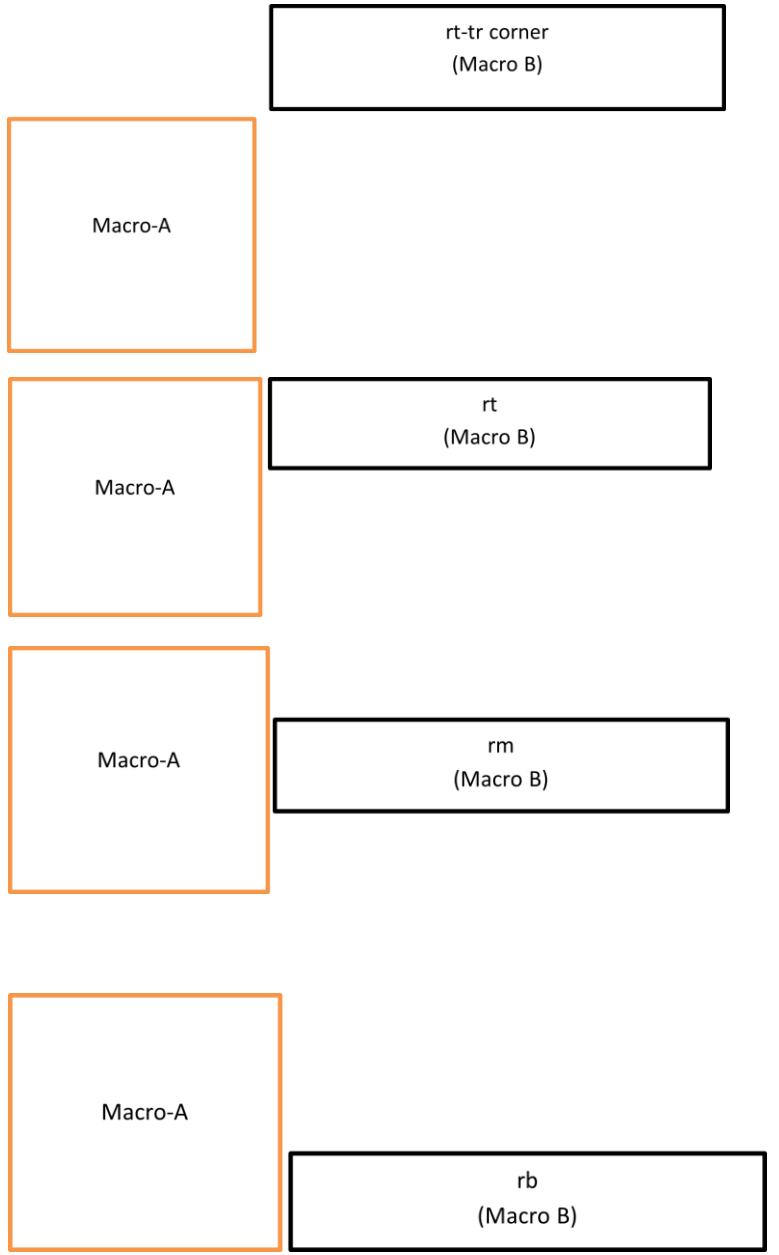


Bottom Right



Bottom-right or Right-Bottom corner

Methodology



Right top or Top Right corner

Right Top

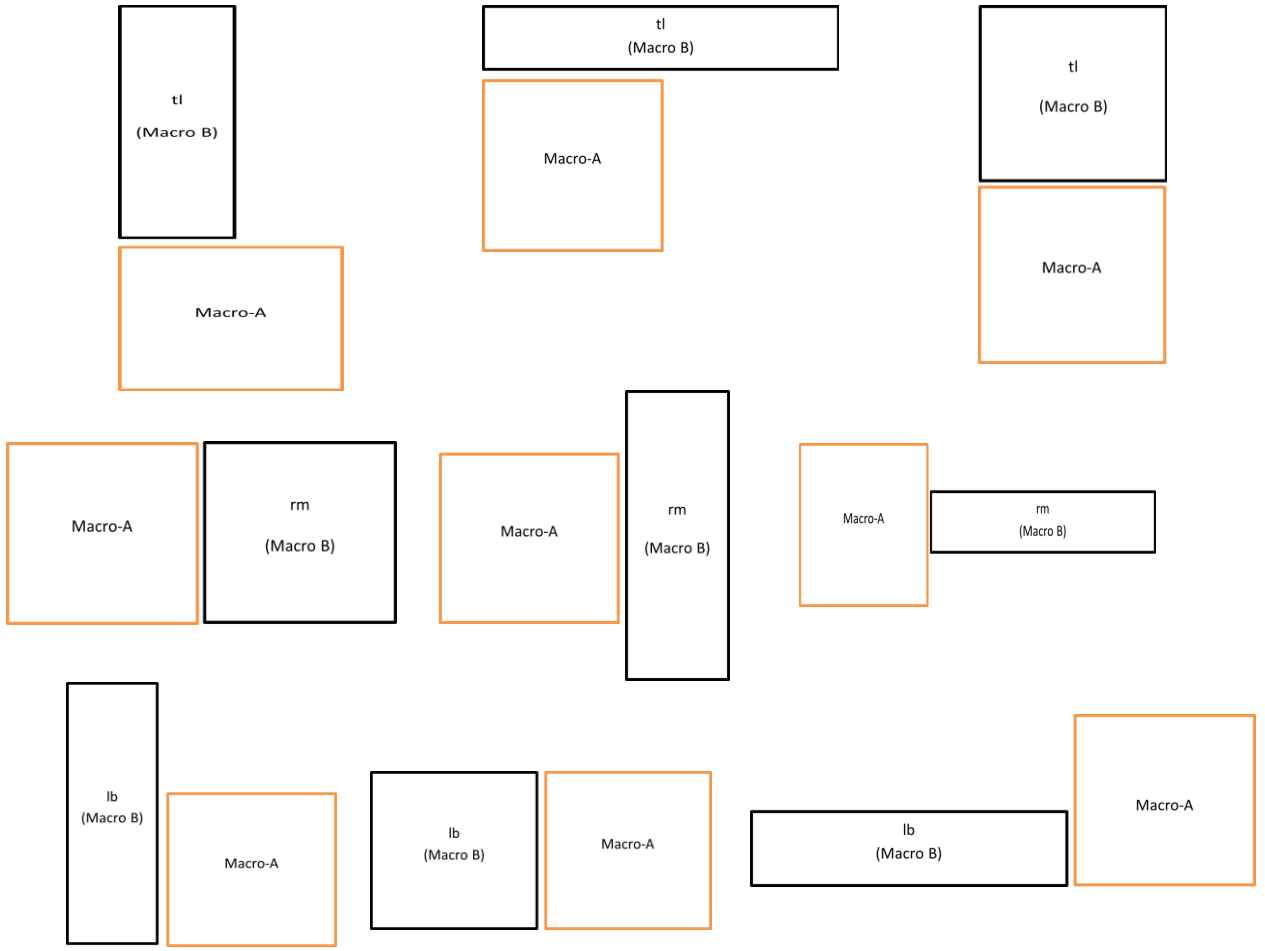
Right Middle

Right Bottom

Methodology

Relative **M**acro **P**lacement

- Applicable irrespective of macro sizes.
  - Macro B can be wider or taller or same size as Macro A.



Methodology

- Demonstration of placement carried out using place_mem command that is based on the relative placement methodology.
  - Initial placement w.r.t. core
  - Subsequent placements w.r.t. to already placed macros that builds upon existing placement analogous to building a puzzle.
  - Gap between macros when placing.

Methodology

Relative Placement

- **w.r.t. core:**
  - place_mem "tr" "my" wt16k 2 3 0 0 C
  - “tr” => Relative alignment keyword
  - “my” => Instance orientation
  - “wt16k 2 3 0 0” => Instance name abbrev.
  - “C” => Argument symbolizing placement w.r.t. core
- **w.r.t. another macro**
  - place_mem "bl" "my" wt16k 2 3 0 1 S wt16k 2 3 0 0 0 "d2c" "br“
  - “bl” => Relative alignment keyword
  - “my” => Instance orientation
  - “wt16k 2 3 0 1” => Instance name abbrev.
  - “S” => Argument symbolizing standard placement w.r.t another macro
  - “wt16k 2 3 0 0” => Relative instance abbrev.
  - 0 => Distance between macros
  - “d2c” => Preciseness of placement.
  - “br” => Grid alignment keyword.

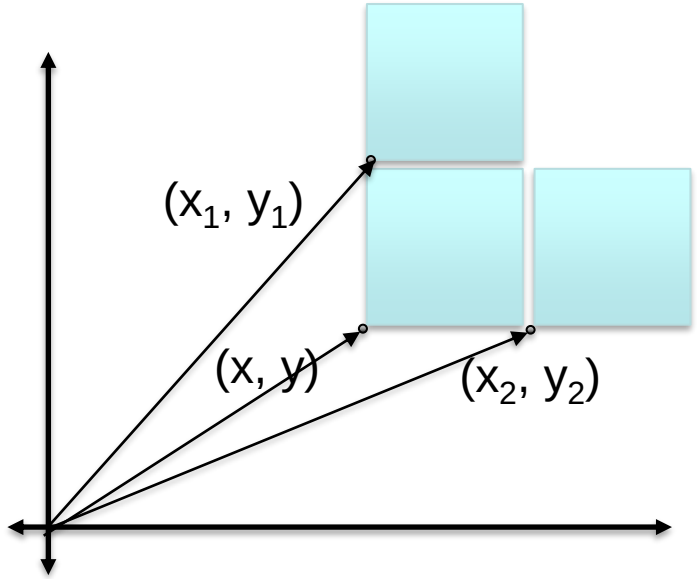
Absolute Placement

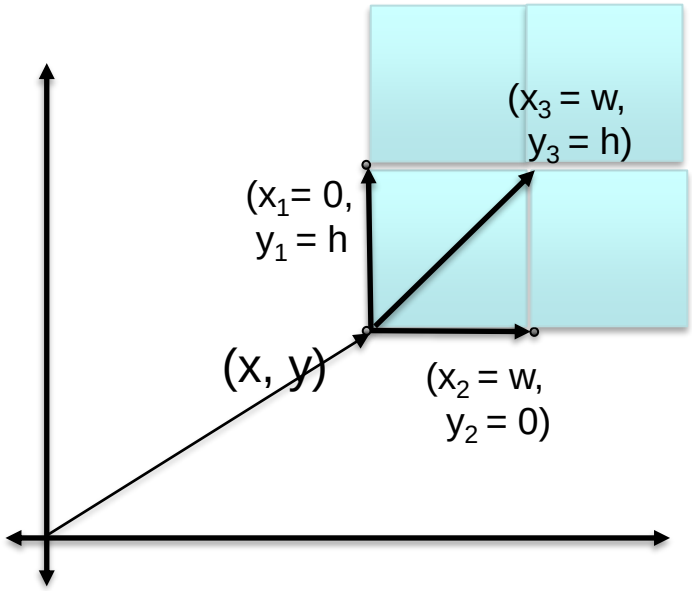
- place_inst $inst "$X $Y" $inst_orient –fixed
  - “$inst” => Instance name to be placed
  - $X => X coordinate of the instance to be placed
  - $Y => Y coordinate of the instance to be placed
  - $inst_orient => Orientation of the instance
  - -fixed/-placed/-soft_placed => Placement Tag

place_mem runs place_inst under the hood right now but this can be developed further. Each macro can be thought of as an object which stores its neighboring macros.

Our Methodology

Advantages





Absolute placement:

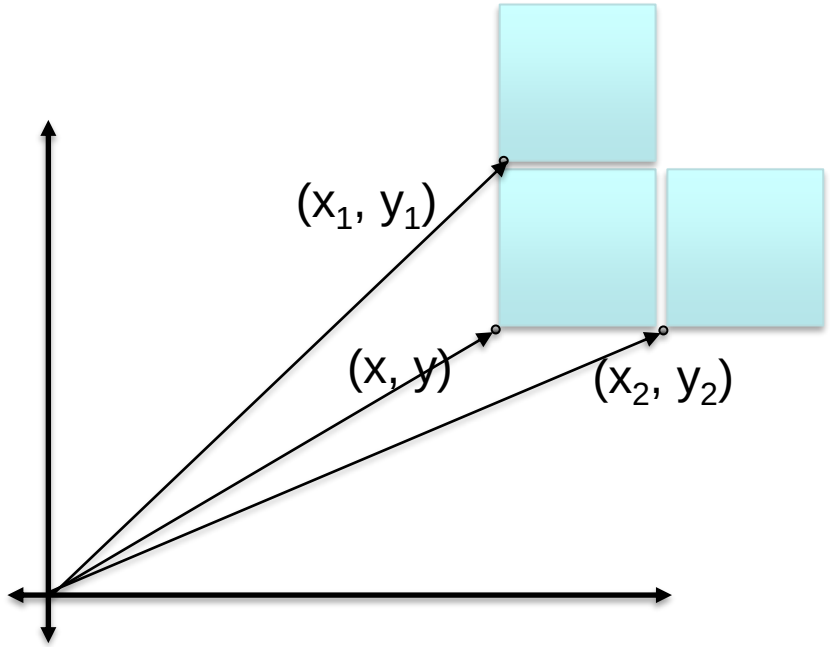
- Requires multiple recalculation for each macro placed.
- Time consuming and tedious process

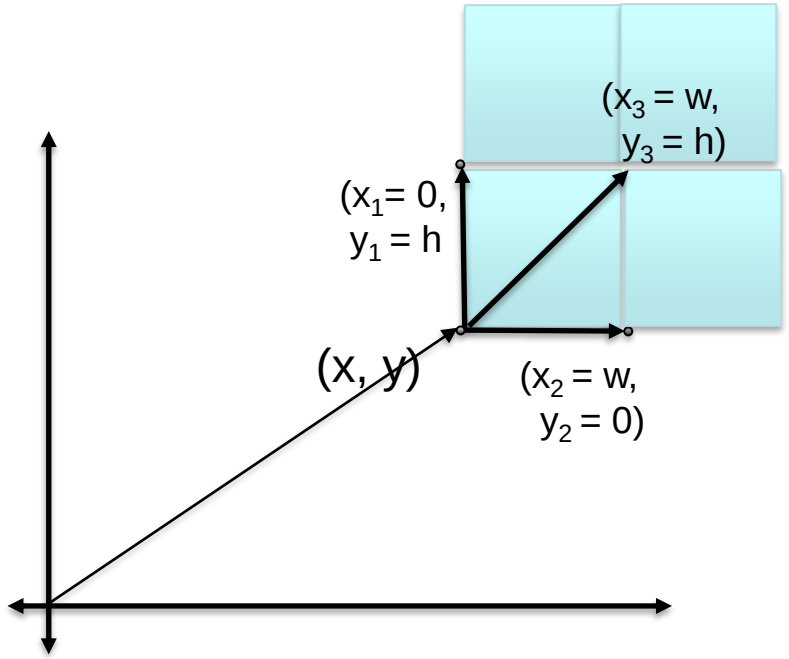
Relative placement:

- Requires initial calculation for the first macro placed. Coordinates for rest of the macros are known as height and width of the macro is known
- Allows grouping of macros which is less compute intensive.

Our Methodology

Advantages





Absolute placement:

- Requires multiple recalculation for each macro placed.
- Time consuming and tedious process

Relative placement:

- Requires initial calculation for the first macro placed. Coordinates for rest of the macros are known as height and width of the macro is known
- Allows grouping of macros which is less compute intensive.

Experimental Results

- The script developed based on the Relative Placement methodology improved the turn-around time for the task of placement from around two weeks for a subset of the design i.e. Performance Functional Unit (PFU) to 3-4 days for the complete CNN sub-processor placement provided by our industry partner [4].

- Optimized placement for the design space exploration was achieved in a span of two months.

Experimental Results

Placement build date	WNS(ns)	TNS(ns)	Wirelength (µm)	Utilization (µm)	Area ($µm^2$)
Jan 28	-0.01	-0.82	40495153	5.29	17119490
Mar 07	-0.03	-2.91	38248240	21.73	19172197
Mar 22	0.00	-0.02	23138843	23.3	13320277
Mar 29	-0.08	-32.85	25093516	42.61	13343707
Mar 30	-0.03	-4.52	25021953	42.24	13336693
Apr 12	-0.04	-3.87	24719999	25.34	13321562
Apr 24	0.00	0.00	23528766	30.47	12799161
Apr 29	0.00	0.00	21730693	44.68	12684283

Table II: Placement metrics obtained during the design space exploration of the CNN sub-processor

Experimental Results

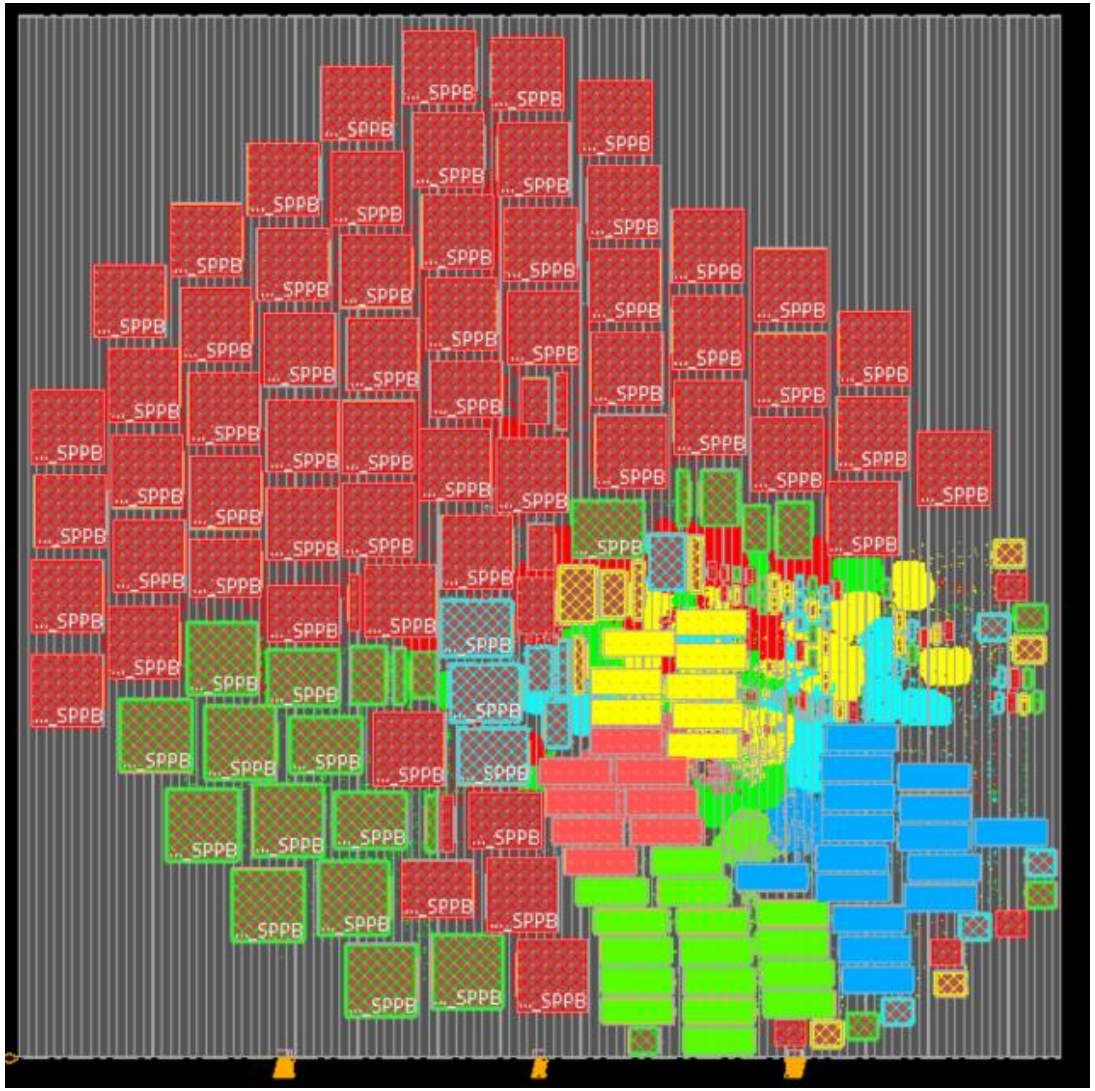


Figure 4: CAD Tool achieved placement

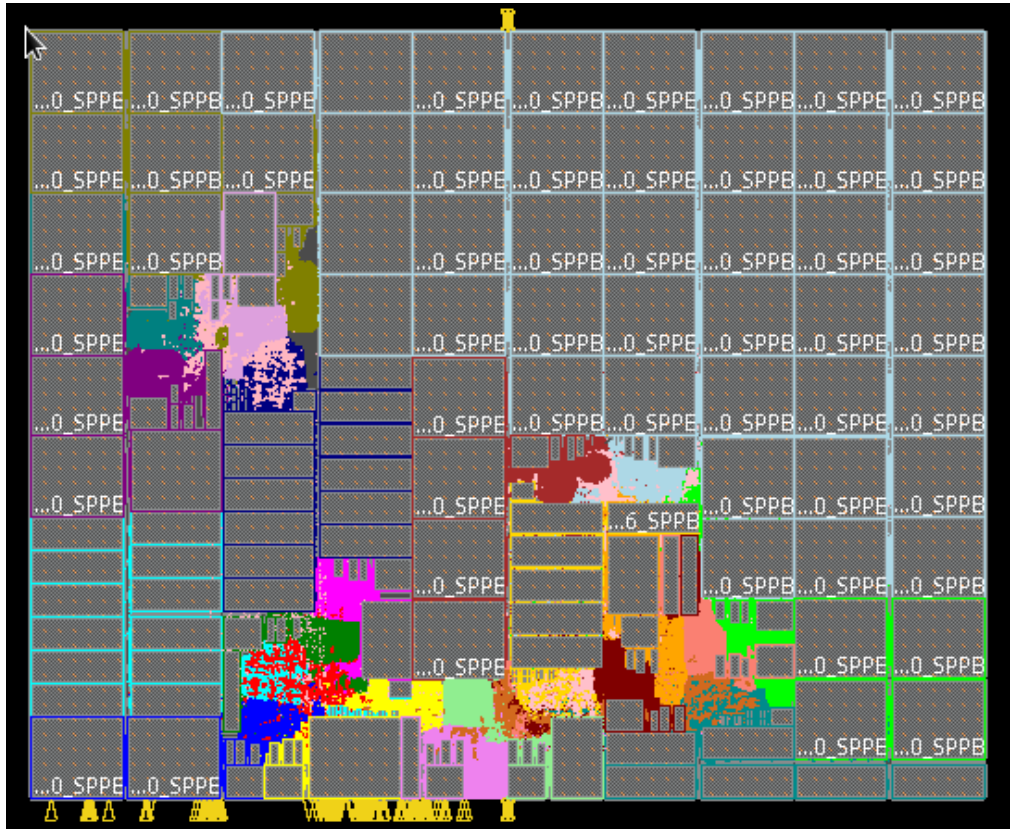


Figure 5: DSE achieved placement

Experimental Results

Figure presents the comparison of Area, Utilization, and Wirelength as a percentage of the maximum value of each metric.

Figure shows that the area and wirelength kept decreasing steadily as iterative improvements were done in the memory macro placement (DEF).

Utilization shows a similar trend but with a peak at placements 4 and 5 because of a congested placement which led to better utilization but with poor timing.

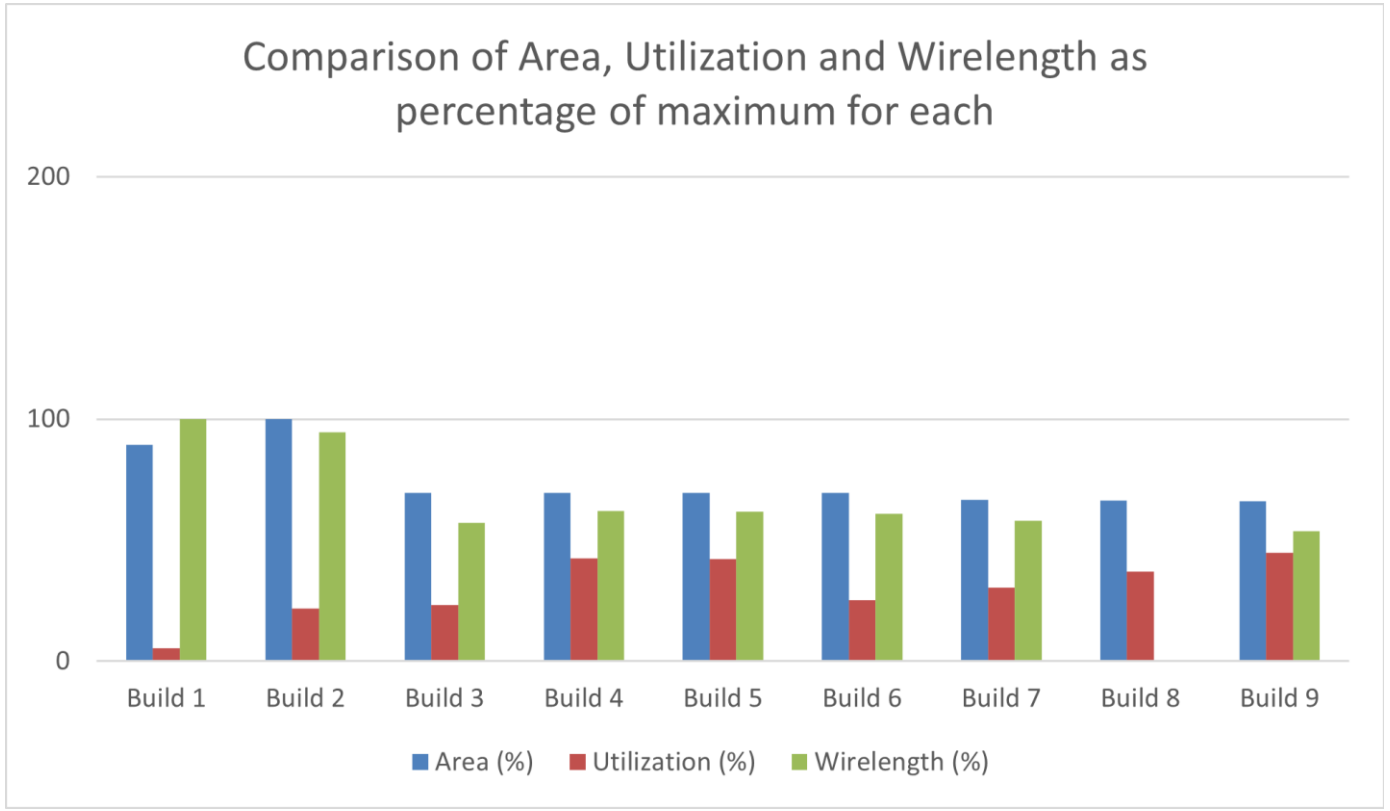


Figure 6: Bar Graph comparing placement metrics as DSE progressed.

Conclusion & Future Work

- Design space exploration led to a 25% area improvement which is a large saving in terms of cost. This DSE was made possible with the help of a relative placement methodology.
- This methodology eased the process of macro placement and made placement iterations faster.
- As a future task, the proposed relative placement script can be automatically generated by the tool based on GUI (Graphical User Interface) macro placements done by the user.
- The relative placement command place_mem can be built into the tool instead of using the place_inst command that does absolute macro placements
- Also, an RTL-Aware Dataflow-Driven Macro Placement tool that can achieve the physical placement of macros in an automated manner.

References

[1] Sarrafzadeh, M., Wang, M., Yang, X. (2003). Macro-Cell Placement. In: Modern Placement Techniques. Springer, Boston, MA. https://doi.org/10.1007/978-1-4757-3781-3_9

[2] I. L. Markov, J. Hu and M. -C. Kim, "Progress and Challenges in VLSI Placement Research," in Proceedings of the IEEE, vol. 103, no. 11, pp. 1985-2003, Nov. 2015, doi: 10.1109/JPROC.2015.2478963.

[3] Alex Vidal-Obiols, Jordi Cortadella, Jordi Petit, Marc Galceran-Oms, and Ferran Martorell. Rtl-aware dataflow-driven macro placement. In 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE), pages 186– 191, 2019.

[4] Pavel SINHA. Configurable processor for implementing convolution neural networks. https://patentscope.wipo.int/search/en/detail.jsf?docId= WO2021014215, January 2021. Accessed: 2022-10-01.

Thank You!