Reducing Library Design Effort with Cadabra Layout Automation

Saroj Kumar Satapathy, Saroj.Satapathy@lsi.com
Pappu Satyanarayana, Satya.Pappu@lsi.com
Vishnu Kanchi, Vishnu.Kanchi@synopsys.com

ABSTRACT

Standard Cell library development plays a key role in the design of ASICs (Application Specific Integrated Circuits). The standard cell library development flow starts with the optimized netlist of logic functions, which are used as an input to get the physical layout through layout design tools. The standard cells are then used as building blocks for high level design. As the process nodes shrink, the design rules are becoming more complex making it imperative for one to change from the manual mode to the usage of automated solutions for layout generation to reduce the Time-to-Result (TTR). This paper aims at describing how Cadabra, the Synopsys solution for automated layout generation, can be used to optimize the TTR for the standard cell library development, from user's perspective. This paper covers use of Cadabra for automating creation of standard cells, optimizing library yield and supporting custom layout styles. Different flows and their advantages are explored covering full netlist to GDS synthesis, various design reuse flows and manual assisted flows to quickly improve results for very complex cells. In conclusion, TTR is compared between manual design and Cadabra automated layouts. We also propose some futuristic developments that might be of use to the library development teams across the globe.

Table of Contents

Table of Figures

1.0 Introduction

In semiconductor design, standard cell methodology is a method of designing Application Specific Integrated Circuits (ASICs) with mostly digital-logic features. Standard cell methodology is an example of design abstraction, whereby a low-level VLSI-layout is encapsulated into an abstract logic representation (such as a NAND gate). Cell-based methodology (the general class that standard-cell belongs to) makes it possible for one designer to focus on the high-level (logical function) aspect of digital-design, while another designer focused on the implementation (physical) aspect. Along with semiconductor manufacturing advances, standard cell methodology was responsible for allowing designers to scale ASICs from comparatively simple single-function ICs (of several thousand gates), to complex multi-million gate devices (SoC) [1].

A standard cell library is a collection of low level logic functions such as AND, OR, INVERT, flip-flops, latches and buffers. These cells are realized as fixed height, variable width full custom cells. The key aspect with these libraries is that they are of a fixed height, which enables them to be placed in rows, easing the process of automated digital layout. The cells are typically optimized full custom layouts, which minimize delays and area. They also may contain the following additional components

- A full layout of the cells
- Spice models of the cells
- Verilog models or VHDL Vital models
- Parasitic extraction models
- DRC rule decks

A layout represents the design physically, which information is used further for the mask preparation. This mask is helpful for the fabrication units to manufacture the design on silicon. During a standard cell library activity, the layout takes one of the largest amounts of time. Again, the verification activity (DRC/ LVS checks) lengthens the output time of the final layout of the cell. A simple flow of standard cell library to produce the final layout is depicted in figure 1. Also with technology advance, design rules for layout pattern have been complicated. Life cycle of manufacturing technology, however, is getting shorter, and design time for library design must be shortened [2].

1.1 Why Automate Layout Creation?

Although manual design still is preferred for quality layouts, the effort and time required always put it to disadvantage. As discussed, for a standard cell library, its key and uniform aspects can be used to automate the process with efficient algorithms to compete the quality manual designs, without consuming the same effort and time. This thereby improves the time-to-market of the final ASICs/SoCs. In addition, the increase in the complexity of the design rules and the need to have large number of cell variants in a library make automation all the more necessary and efficient. Although there are several layout generation tool available in the market, this paper deals with Synopsys tool for automatic layout synthesis, Cadabra. It describes the features and uses of the tool with a few statistics as experienced by its users.

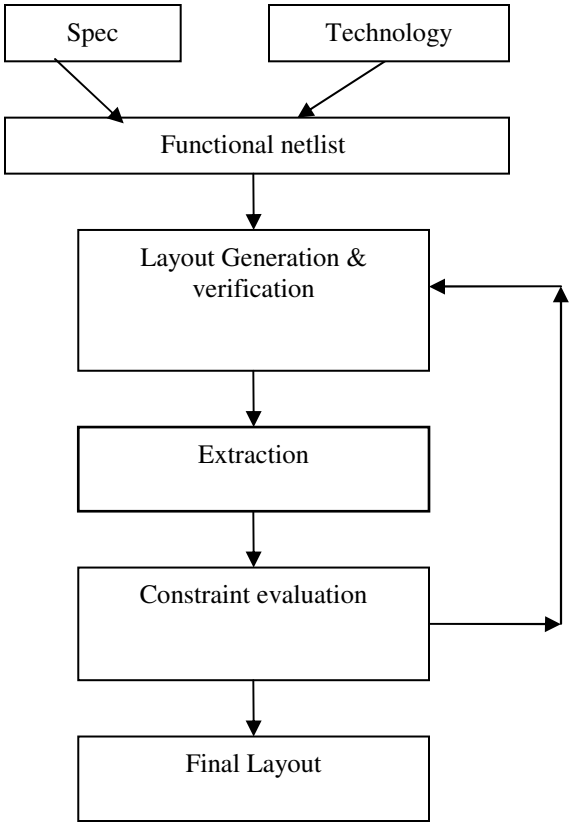


Figure 1 – Flow chart for a standard cell library flow

2.0 Cadabra – Automatic Layout Synthesis Tool

Cadabra provides an automated solution for both migration and creation of custom standard cell layouts. Cadabra helps to migrate custom standard cell libraries from one process technology to new process technologies, with same/different transistor sizes with/without changes in architectures, and create LVS and DRC correct cell layouts [3]. Cadabra can also be used to create the layouts from the spice netlists by providing the technology and the architecture files. The tool has many features and methodologies that establish it as an effective tool for layout synthesis. Some of the main features are being discussed in following sections.

2.1 Features and Methodology of Cadabra

Cadabra is an automated tool that's used to generate layout from spice file input or an initial layout (reads in as gds format). Its usage is generally for standard libraries. The tool generally works on grid based library architecture. Its output formats are universal and flexible. It also outputs in the gds format, one of the standardized formats throughout the industry. The tool has a good Graphic User Interface (GUI) for the users which makes it user friendly. The tool can also be customized as per the needs of the designer through the AL language interface.

Cadabra provides three main automated processes: Migrate GDS, ATL (Automated Transistor Layout), and Migrate-ATL.

Migrate GDS uses cell netlists, GDSII files, a target architecture and a technology file as input, it then imports, compacts, finalizes, and exports cell layouts in this flow.
ATL uses cell netlists, architecture and technology as input, ATL places, routes, compacts, finalizes, and exports cell layouts.
Migrate-ATL is a hybrid of both Migrate GDS and ATL flow, which uses cell netlists, GDSII files, a target architecture and technology as input, Migrate-ATL runs the Migrate GDS process first, then based on the input provided by the user decides to perform or not perform ATL.
A flowchart showing the functional flow for Cadabra is represented in the figure 2.

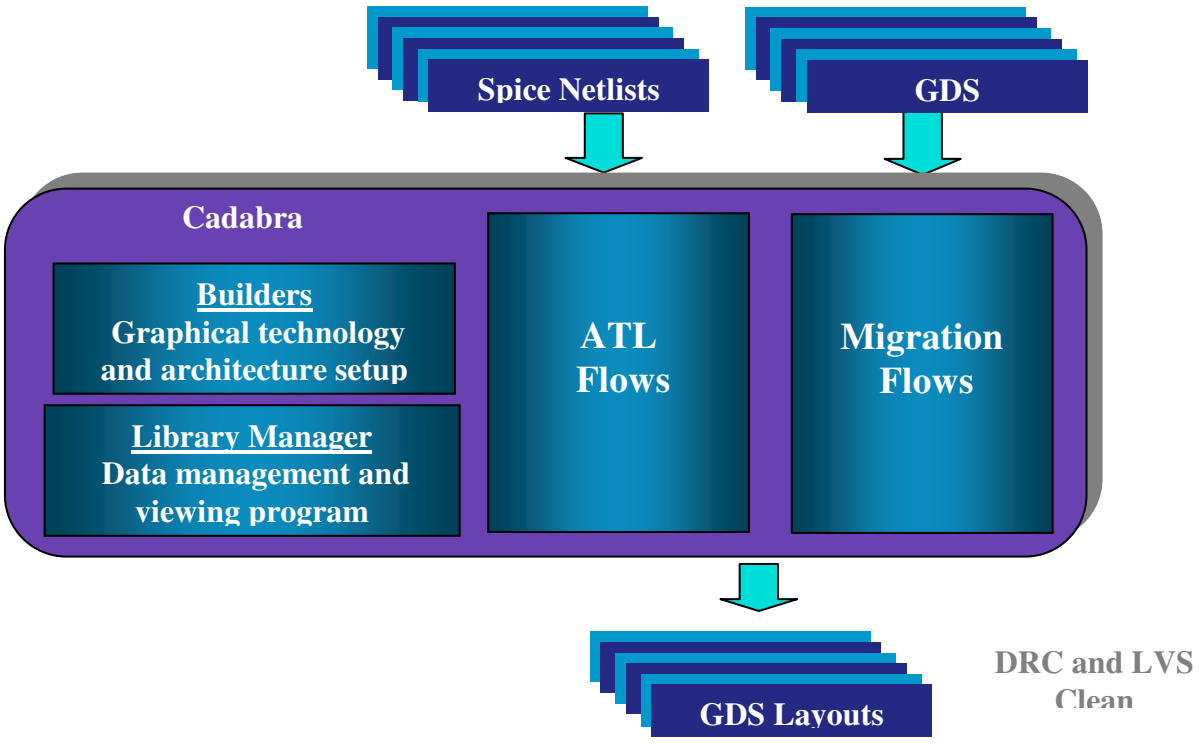


Figure 2 – A flowchart showing the functional flow for Cadabra

2.1.1 Cadabra Set-up and Library Manager

The setup to create the standard cell layouts includes entering the technology rules and capturing the architecture requirements. Cadabra uses the Technology Builder to capture all the technology rules as per the design rule manual. Complex rules such as common run, width dependent, spanned edge etc can also be added through the GUI in addition to the simple separation, dimension and extension rules. Architecture Builder GUI is used to capture all the architecture requirements in cadabra. Architecture file defines all the common elements of the cells in the library (Ex. the height of every cell, location of the N-well etc). Once we have the setup ready, we can create a library to process the cells, all the cells in a library share the same technology and the architecture. The library is accessed via the Library Manager [4]. The spreadsheet view depicting the library manager is shown in figure 5.

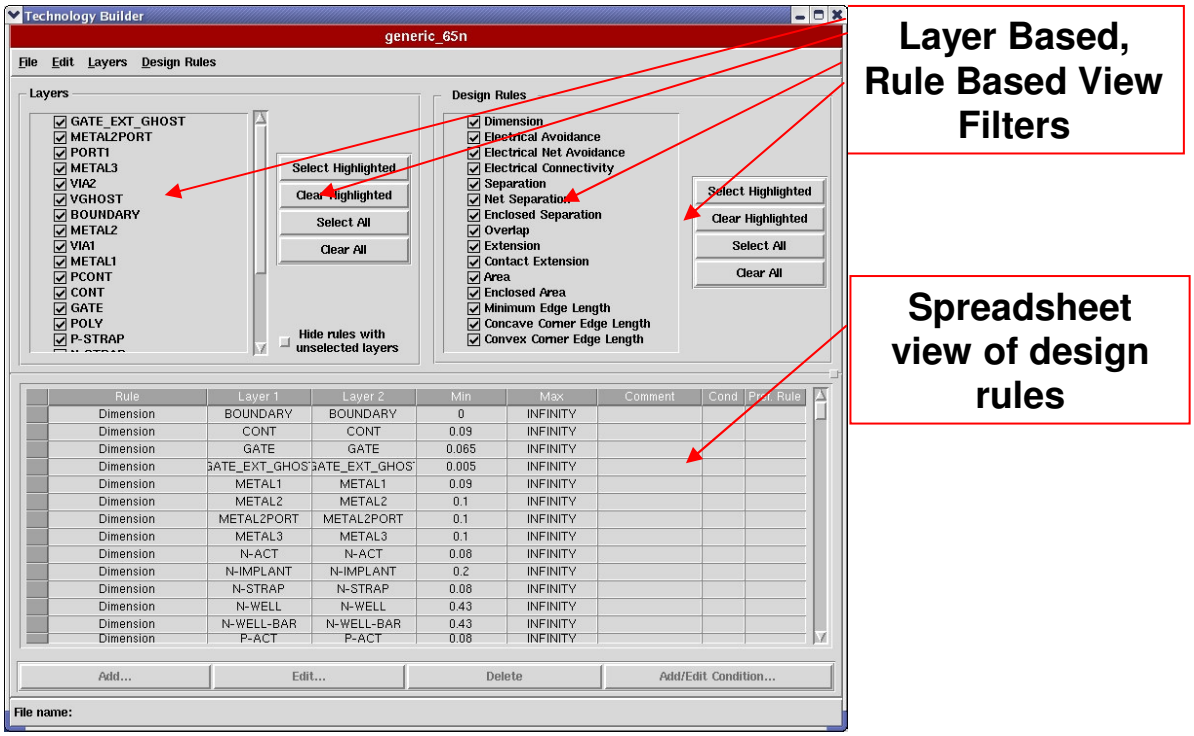


Figure 3 – A figure showing the technology builder form.

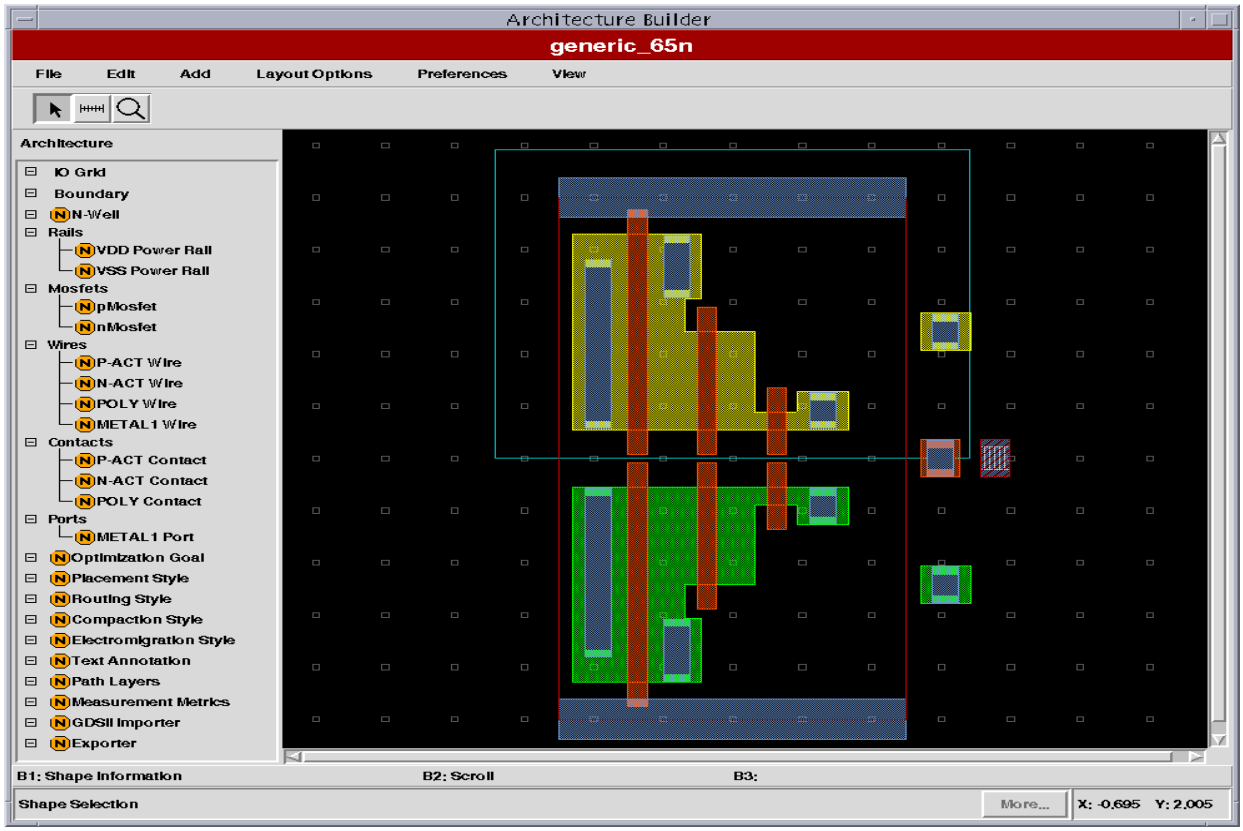


Figure 4 – A figure showing the architecture builder form.

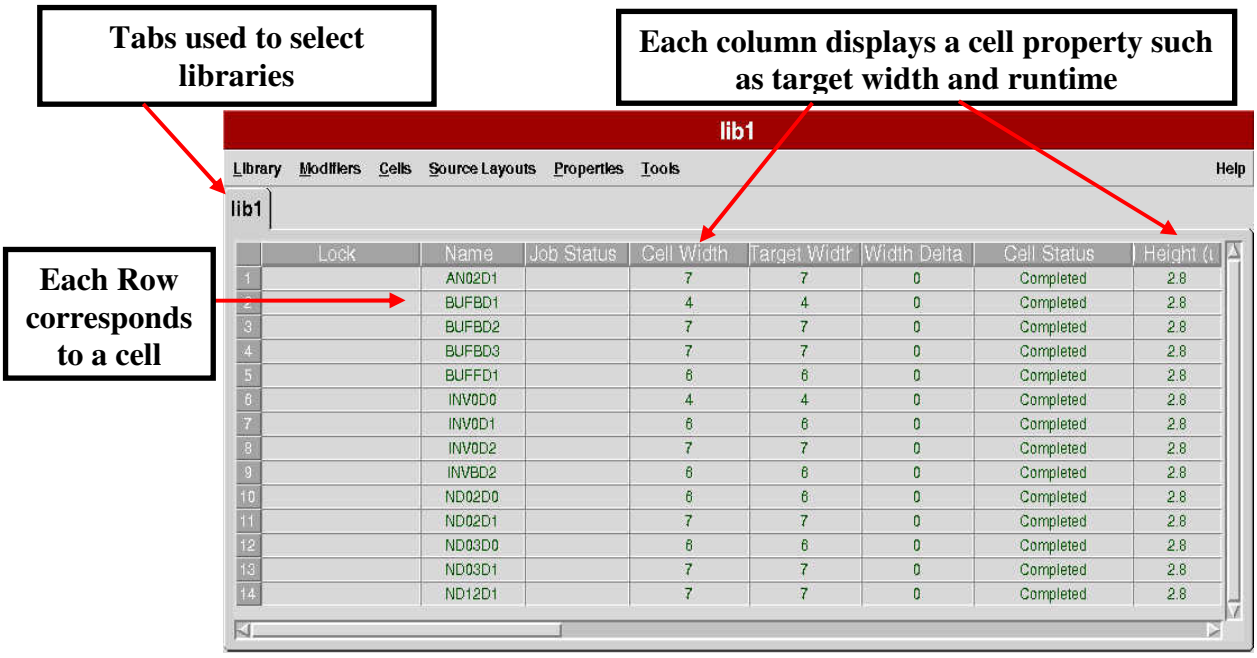


Figure 5 - A spreadsheet view of the graphic manager.

Cadabra process the cells in form of a design tree that provides graphical representation of the step-by-step cell design process as shown in figure 6. A design tree is a collection of design points and design steps. Different design points represent different status of the process while being processed through the design steps. The finalized design point is exported for the output layout. Cadabra constructs the layouts using symbolic devices instead of treating them as polygons (Ex: a contact is a symbolic device consists of 3 shapes, a contact cut and 2 cover shapes).

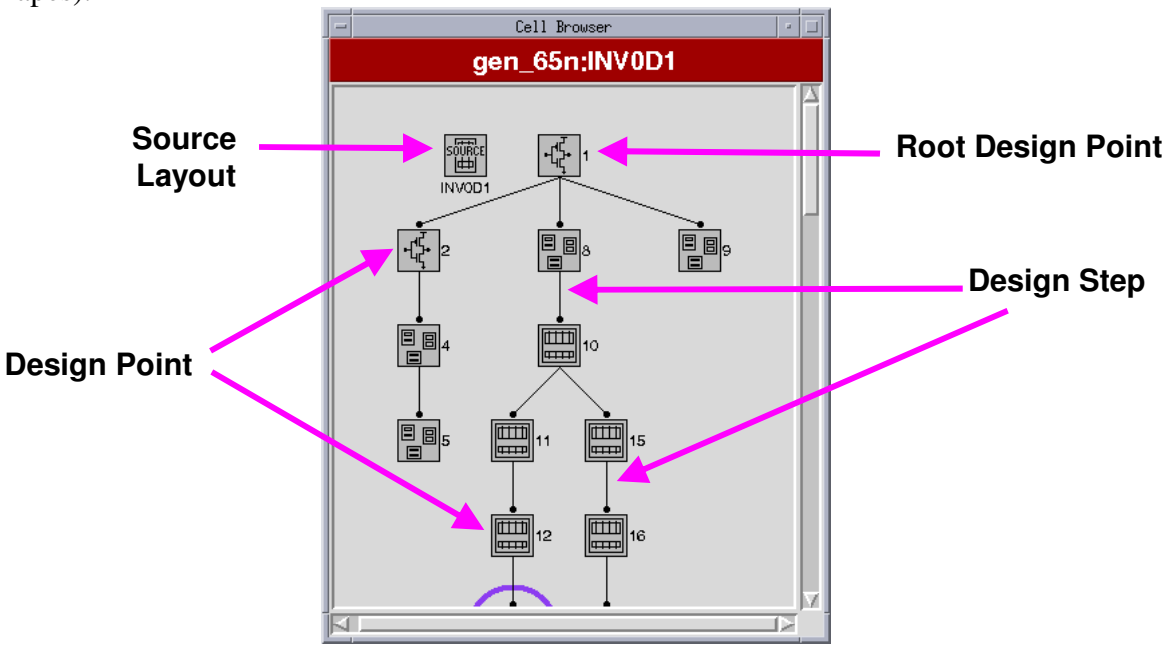


Figure 6 – An example of a design tree with its components.

2.1.2 ATL (Automated Transistor Layout)

ATL automatically creates cell layouts using a netlist, technology rules and the architecture parameters defined [4]. ATL flow consists of the following steps:

- Placement: The placement process arranges all the MOSFETs in the netlist symbolically. These MOSFETs would be in terms of symbolic devices that are placed in the layout in as optimum way as possible (even taking into account the diffusion sharing).
- Routing: After placement this process connects all the MOSFETs as specified in the netlist. So, this step generally takes care of the LVS part with most optimum connectivity.
- Compaction: This process enforces the design rules as per the technology. It also takes care of some of the architecture requirements like growing contacts, ports, implementing DFM rules etc. Compaction, as the name suggests makes the layout compact and ensures that the layout comes out with optimum size and quality.
- Finalize Layout: This process has a series of design steps that convert the symbolic layout into normal layout form. So it prepares data and exports to output format.

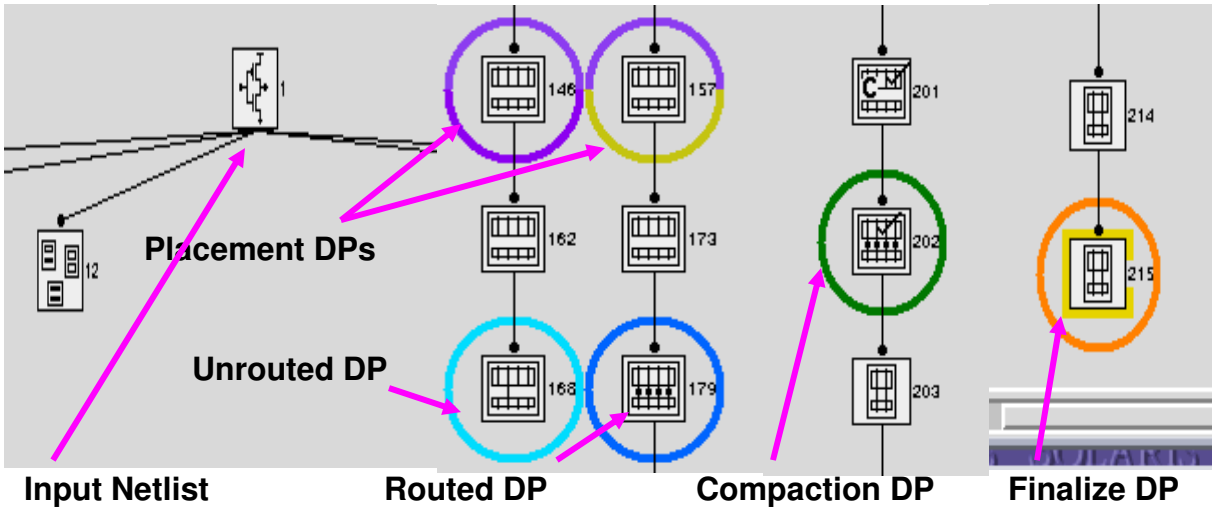


Figure 7 – ATL flow with important design points

2.1.3 GDS Migrate

GDS Migration utilizes the topology of the source layout to create a new layout in a different technology with/without any architecture changes. The connectivity of the source layout must match the connectivity of the target netlist, however MOSFET sizes can vary. GDS Migration does not support major topology changes [4]. GDS migration flow consists of the following steps as illustrated in figure 8:

- Import: This step converts the GDSII layout into symbolic layout. As Cadabra processes symbolic layouts, this step is required for starting the processing of the cell.
- Compaction: It has the same functionality as described in ATL methodology.
- Finalize Layout: It has the same functionality as described in ATL methodology.

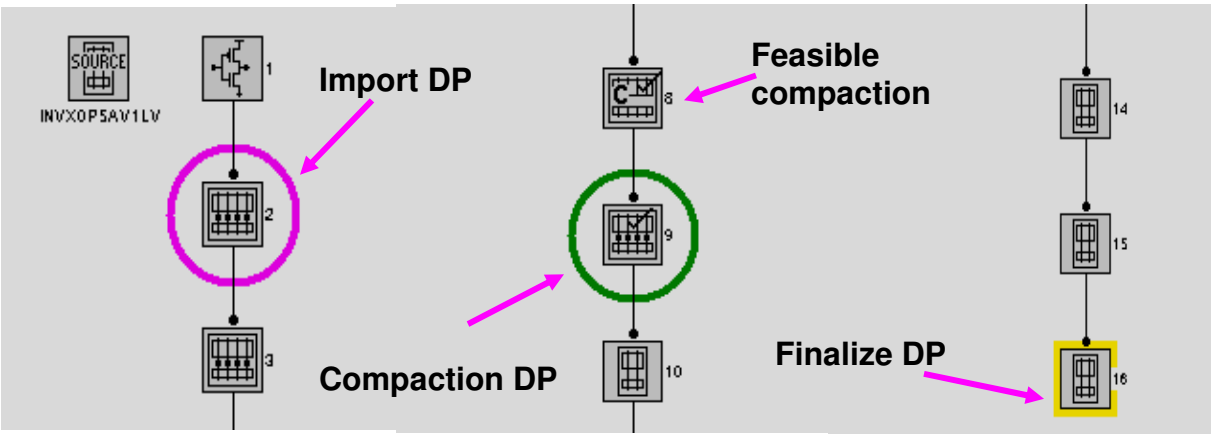


Figure 8 – A GDS Migrate flow with important design points.

3.0 LSI Adoption of Cadabra

In addition to using the normal ATL and the Migrate-GDS flows in Cadabra, LSI also uses some additional flows in Cadabra that help optimize the layout generation time and the productivity of the physical design team. These are some of the advanced flows used by LSI for their highly customized and complex libraries:

- Reference cell and Cloning flow
- ECO Migration
- Manual Assisted flow

3.1 Reference cell and Cloning flow

The reference cell and the clone flow as shown in figure 9, gives an edge to the tool in terms of design reuse. In case a cell has to inherit a part or total cell topology, this concept can be used to produce the target cell. It gives more consistency between layouts which is useful.

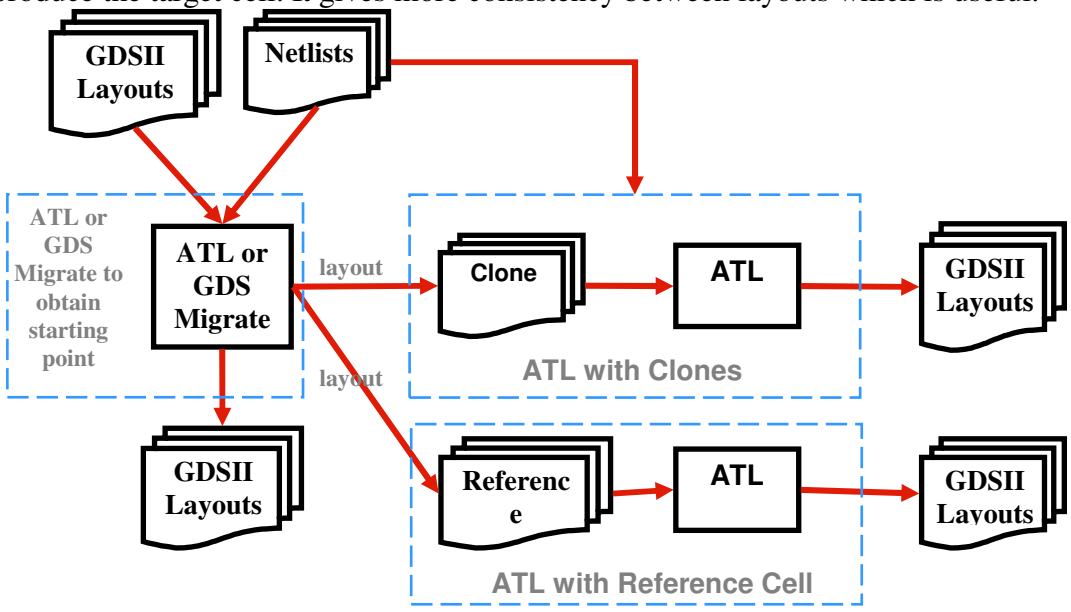


Figure 9 - A flowchart showing the Reference cell and clone flow.

3.2 ECO Migration

The ECO migration flow, as shown in figure 10, enhances the normal migration flow to get an advanced customization on the layout with added user constraints. It gives added user constraints or preserve with different priorities according to the requirements.

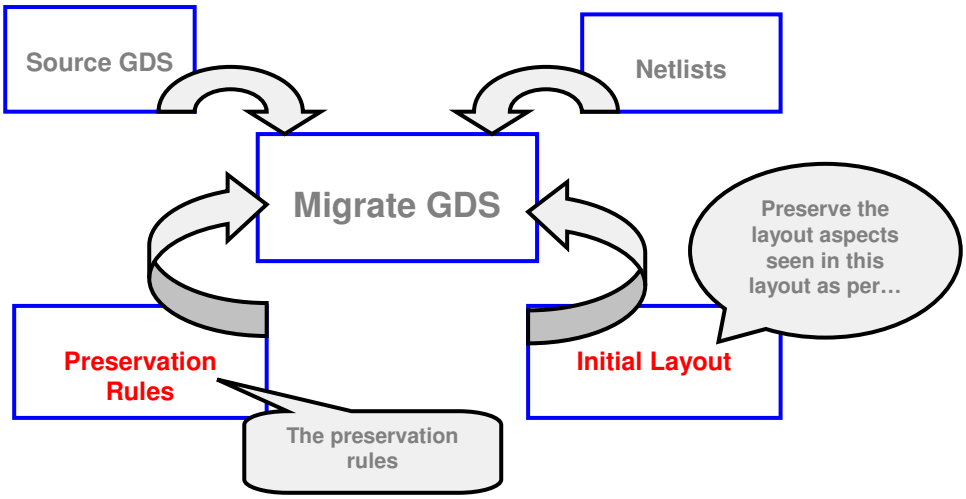


Figure 10 - Flowchart depicting the ECO Migration flow.

3.3 Manual Assisted flow

Though, Cadabra does a good job for most of the cells in the library, there would be some portion of the cells that will not come out directly from the tool. The cells that don't finish either come out unrouted/infeasible. In this case, the user has two options, 1- fall back on the layout editor tool and do the cell in the tool OR 2- use the cell edit feature inside cadabra to finish the routing or solve the infeasibility and then compact the cell using cadabra to get the DRC clean layout. At LSI, we have found that option 2 gives us considerable benefit when it comes to the Time to Result for the layout generation. User can use the compaction browser to understand the infeasibility and then solve the infeasibility by using cell editor. Figure 11 shows one of the snapshot of cell-editor.

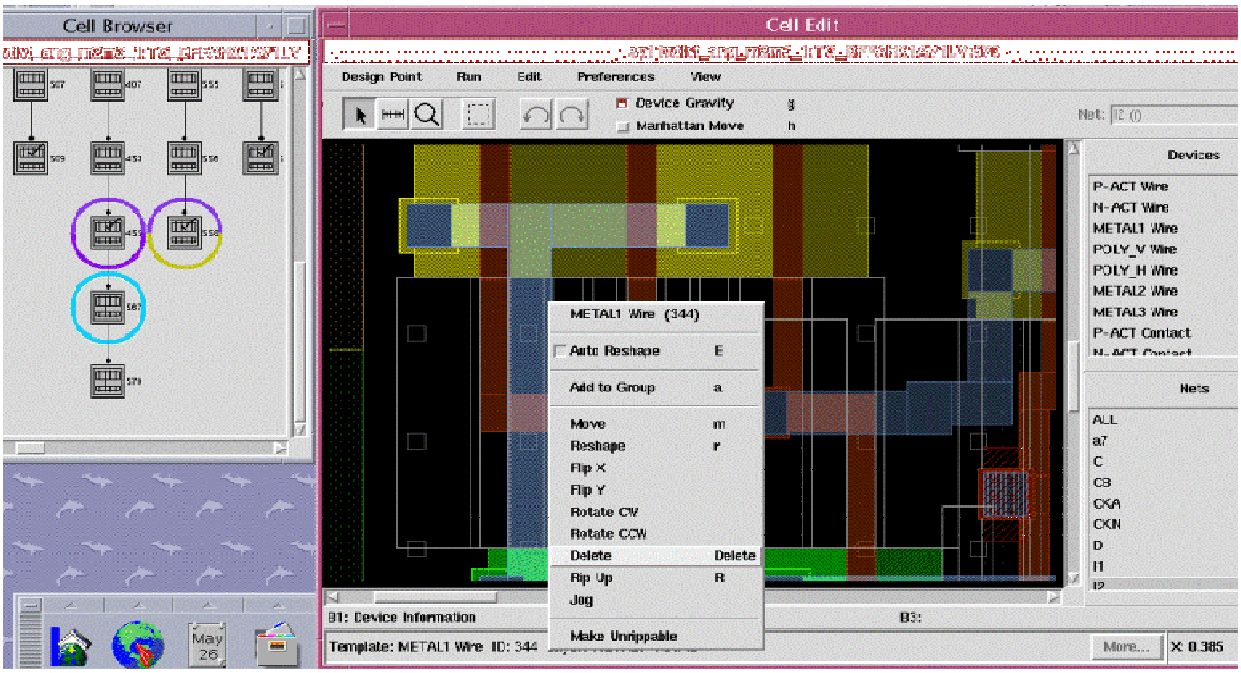
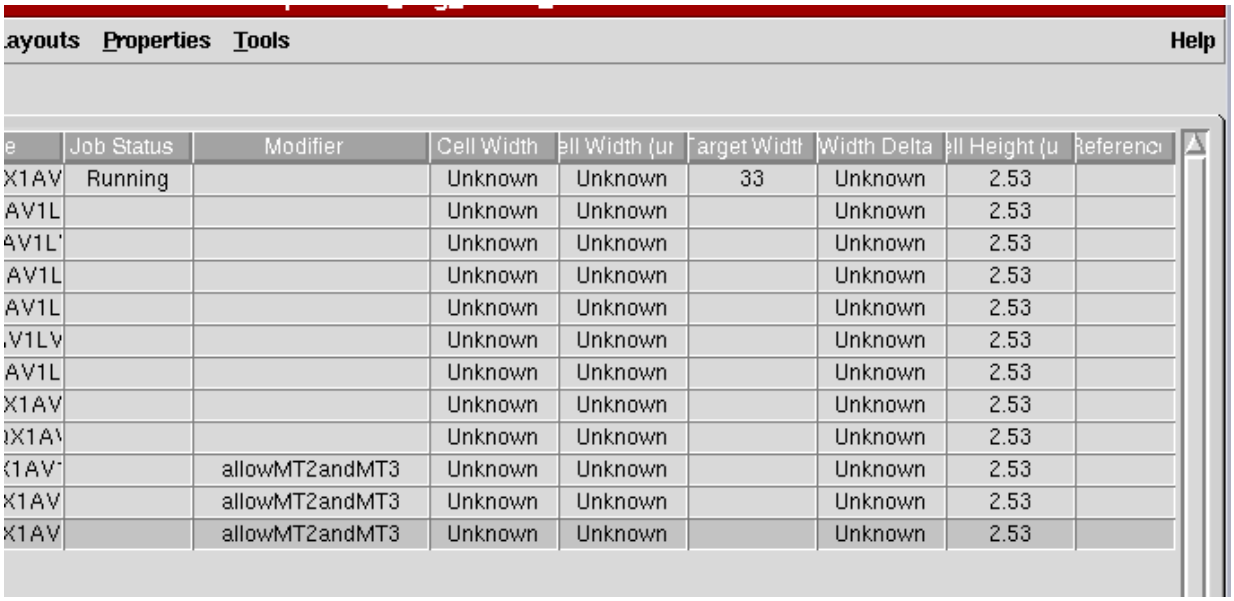


Figure 11 – Cell editor implementation on a design point.

3.4 Other features

- There are different options to submit a run. It can be used to run at different effort level to balance the run-time and efficiency. Moreover, the option to submit the jobs to local machine or LSF/SGE queues is present. It supports all the features present in LSF and SGE.
- A cell modifier gives the ability to change the layout style defaults on a per-cell basis as shown in figure 12 (Cell-level granularity).



e	Job Status	Modifier	Cell Width	ell Width (ur	arget Width	Width Delta	ell Height (u	Referenc
X1AV	Running		Unknown	Unknown	33	Unknown	2.53	
AV1L			Unknown	Unknown		Unknown	2.53	
AV1L'			Unknown	Unknown		Unknown	2.53	
AV1L			Unknown	Unknown		Unknown	2.53	
AV1L			Unknown	Unknown		Unknown	2.53	
V1LV			Unknown	Unknown		Unknown	2.53	
AV1L			Unknown	Unknown		Unknown	2.53	
X1AV			Unknown	Unknown		Unknown	2.53	
X1A			Unknown	Unknown		Unknown	2.53	
1AV		allowMT2andMT3	Unknown	Unknown		Unknown	2.53	
X1AV		allowMT2andMT3	Unknown	Unknown		Unknown	2.53	
X1AV		allowMT2andMT3	Unknown	Unknown		Unknown	2.53	

Figure12 – Implementation of cell modifiers on a library at cell by cell basis.

- Cadabra has Double Height Cell creation capability through ATL and Migration flows. Cadabra supports 3 different row styles for double height cells. The ease of use is enhanced by the fact that with the same setup we can produce both the single height and double height layout topologies.

3.5 Implementation work with Cadabra –Statistics

The completion rates, DRC correctness and the quality of the layouts obtained from Cadabra augment the physical design team and help in improving the productivity of the team. Cadabra generated layouts have less notches and jogs that's been an important factor while descending to sub-micron technologies thereby reducing DFM errors. This is an important factor for LSI Corporation to embrace the tool. The data here provided is the general user experience of Cadabra and are an average and approximate indications.

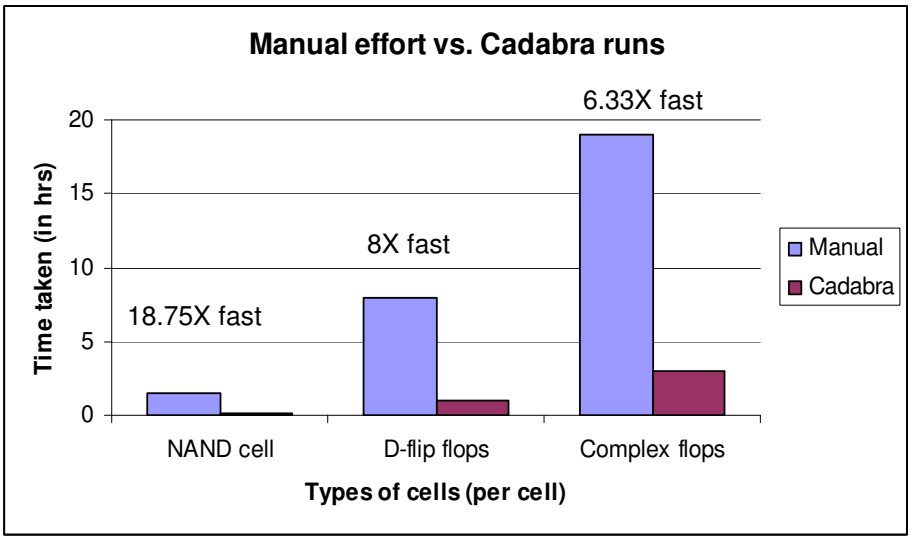
Table 1 – Statistics for time taken by manual effort vs. Cadabra tool effort for ATL activity.

Circuit Example	No. of Devices (approx)	Time taken by Manual effort(avg.)	Time taken by Cadabra tool (avg.)
NAND	4-10 MOS	1 and half hours	5 minutes
D-flip flop	20-25 MOS	1 day (8 hrs)	1 hrs (with slight manual edits)
Complex flops	50-60 MOS	>2 days (19 hrs)	3 hrs (with some manual edits)

One of the interesting statistics to compare is the manual effort to the Cadabra efficiency with respect to TTR. A comparison of the same for ATL flow is given in table 1, with respect to cell complexity. This data is taken with the following assumptions:

- This is based on an experience of about 15-20 custom libraries (for high speed, more dense etc) with around 100 cells per library.
- This data of time taken for Cadabra is estimated once the set-up for the library is already present. The jobs are run on the grid mainly.
- The estimation of manual effort may vary depending upon the engineer working on it. So, the data represents a possible average value taken by an engineer without concentrating too much on a layout quality.

Graph 1 – Manual effort vs. Cadabra runs for ATL activity



There are many important observations as can be presented from graph 1 which shows the data as presented in table 1. First of all, the efficiency largely depends upon the complexity type of the cells. Moreover, with more complex cells, it's being assumed to take more time for Cadabra tool mainly for the manual edits that it involves. So, it's a worst case scenario for the reason of assuming manual edit for each cell and the complex architecture involved. The layout quality is also not taken care during manual effort. So, at this situation we can assume that we are getting comparatively better quality layout from Cadabra tool as it depend upon chance for the engineer to get good quality layout within the time shown in the table.

The complete library statistics for a 100 cell library with 40 simple cells, 40 complex combinationals/simple sequentials and 20 complex sequentials are shown in Table 2.

Table 2 – Statistics for time taken by manual effort vs cadabra tool for ATL on a 100 cell library

Cell Type	**Time taken by Manual effort(avg.)**	**Time taken by Cadabra tool (1 license)**
Simple	40*1.5hr = 60hrs	40 * 5min = 3.33hrs
Complex Combinationals	40*8hr = 320hrs	40*1hr = 40hrs
Complex flops	20*19hr = 380hrs	20*3hr = 60hrs
Total Time	**760hrs**	**103.3hrs**

In table2, we can see that the time taken by cadabra to complete the library is almost reduced by 660hrs. This is with 1 license of Cadabra. Graph 3 shows how effective cadabra can be when compared to the manual layout.

Graph2: Manual Effort Vs Cadabra for ATL on a 100 cell Library

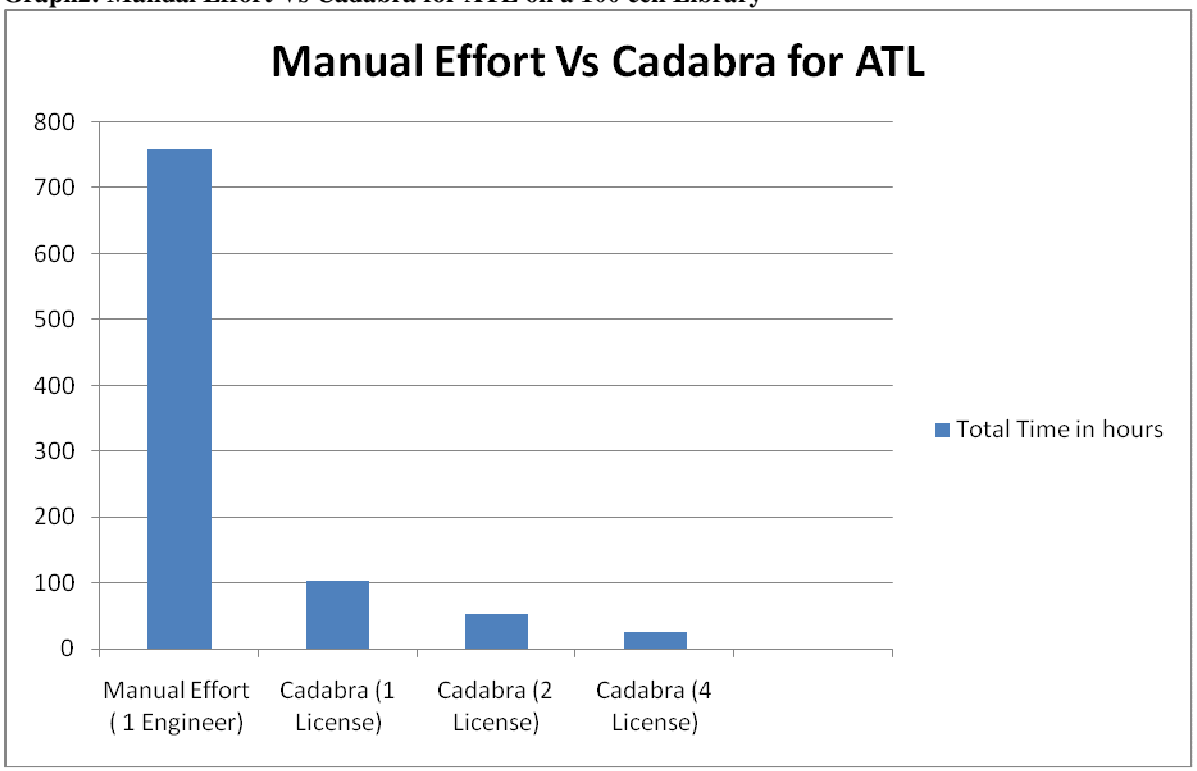
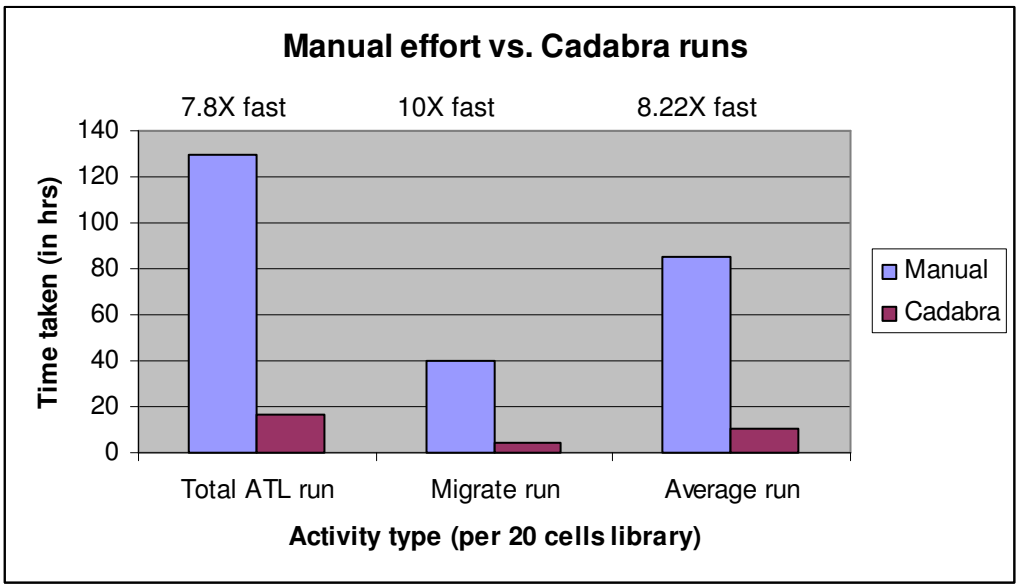


Table 3 – Statistics for time taken by manual effort vs. Cadabra tool for ECO migration activity

Library information	**Time taken by Manual effort (avg. for 1 engineer)**	**Time taken by Cadabra tool (avg. with 1 license)**
Around 20 cells	1 week (40 hrs)	4 hrs (with manual edits)

At LSI India we do a lot of migration work due to ECO changes or simple migration of cells from one architecture to another. The ECO Migration flow of Cadabra has been used by us greatly. The equivalent data for time comparison between manual effort and Cadabra run for the ECO Migration flow is represented in table 2 on a library basis. In this case, it is important to note that the complexity of the cell will make less significance difference as there is a reference cell generally on which cadabra has to work. With the preservation options in Cadabra, it also ensures the output layout retains as much as quality possessed by the original layout thereby exploiting the re-use capability.

Graph 3 – Manual effort vs. Cadabra runs for different activities and an average of both



A summary of Cadabra efficiency is presented in graph 3 for both the ATL and the Migrate run. On an average it has been found that Cadabra is atleast 8.22X faster than the manual generation. The above data is being presented per user per license basis. So, if a user has more number of licenses the efficiency multiplies. In return, it is many times necessary to slightly sand the cells out of cadabra which tends to increase the effort required for a Cadabra based flow. However, with all these data it is evident that Cadabra is certainly helpful for engineers. The quality of the cells is generally good in comparison to the effort supplied and the cells can always be customized according to the needs. A good example can be supplied with usage of higher metal, when prohibited in Cadabra set up can give a good quality cell that reflects at the top level Placement & Routing activity, where the congestion is really less.

Another advantageous feature experienced by LSI engineers with Cadabra tool is its good support and development activity. There is a major release of the tool every 9 months, in addition to that patches are released for critical bug fixes. The upgraded version from 2006 versions to 2007 versions, for example, have introduced many good capabilities in routing and compaction algorithms, multi-processing activity, faster run-times, user-friendly GUI capabilities, etc. One of the main features is ability to create double height cells. Due to lack of a sizeable amount of data double height cell statistics is not shown in this paper. However, we feel that with this addition to Cadabra we can use it to create double height cells for some of the complex cells that become too wide.

4.0 Conclusion

The market expectation always rises with time and it puts pressure on the technology to advance. With this advancement, the flow also gets complex to keep pace with the new technology. The need for standard library methodology has originated from this phenomenon. So, with more advancement it's becoming more inevitable to embrace tools like Cadabra which gives efficiency and quality thereby reducing time-to-market. Cadabra enables design teams to store library

creation knowledge into the environment, preserving IP for future libraries or library changes. Additionally, Cadabra can leverage existing library topologies in the creation of new cells. Because Cadabra can create many cells quickly, designers can build large libraries with a variety of drive strengths. Additionally, instead of getting a generic library from a library vendor, designers can optimize a library for power or speed by using Cadabra

However, it becomes increasingly essential that these tools should also improve everyday at par with the engineer's arising expectations. A general improvement for Cadabra can be increasing the efficiency of its algorithm to compete itself with the new technology and sets of design rules. With improvement in efficiency the tool developers can also aim for covering new horizons of library developments and methodologies. An advanced but researchable idea would be to see Cadabra also working for non-grid based libraries, so that it can work for analog/memory libraries that always don't look for grid-based methodology. Another idea can be including extraction aware layout methodology. By this, we mean Cadabra should be able to calculate the parasitic for nets while doing the layout and the user should have an option to provide the maximum parasitic allowed for a particular net.

Although, there is always scope for improvement, Cadabra tool with its features and the statistics mentioned in this paper exhibits its mighty capabilities. From the perspective of a user this tool has been an immense aid in past and would continue in future with the potential it demonstrates.

5.0 Acknowledgements

The authors would like to thank Mr. Ganesh Tadi of Custom Layout Team, LSI for his contribution to the user statistics with respect to Cadabra activity. The author would also like to thank Mr. Murali Gorur, Technical director, SPG, LSI and LSI Inc, for their encouragement and approval provided for participating at SNUG activity. At last, the author would like to acknowledge the support provided by Synopsys Cadabra team for guiding through this activity and presentation.

6.0 References

[1] Wikipedia, "Library (Electronics)", URL: http://en.wikipedia.org/wiki/Library_(electronics)

[2] Masanori Hashimoto, Kazunori Fujimori and Hidetoshi Onodera, "**Automatic Generation of Standard Cell Library in VDSM Technologies",** Department of Communications and Computer Engineering, Kyoto University, pp-1.

[3] "About Cadabra", Cadabra Manual, Cadabra Y-2006.03, Synopsys Inc, 2006.

[4] "Cadabra_Beginning_Users_Workshop-Y2006.03.SP1.ppt", Synopsys Inc, 2006, pp: 12-13, 16-17.