Universitat Autònoma de Barcelona

Design Automation methods and tools for building Digital^2 Printed Microelectronics Circuits

Manuel José Llamas Rodríguez

Design Automation methods and tools for building Digital^2 Printed Microelectronics Circuits

Ph.D. dissertation
Electronic and Telecommunication Engineering

Author:
Manuel José Llamas Rodríguez

Supervisors:
Jordi Carrabina Bordoll (also tutor)
Lluís Terés Terés

Universitat Autònoma de Barcelona (UAB)
Department of Microelectronics and Electronic Systems (MiSE)

2017, Barcelona, Spain

UAB
Universitat Autònoma de Barcelona

cnm
Centro Nacional de Microelectrónica IMB

The undersigned Prof. Dr. **Jordi Carrabina Bordoll**, professor of the Department of Microelectronics and Electronic Systems, of the Universitat Autònoma de Barcelona (UAB), and Dr. **Lluís Terés Terés**, scientific researcher at IMB-CNM (CSIC) and associated professor at UAB,

CERTIFY

That the dissertation entitled "Design Automation methods and tools for building Digital^2 Printed Microelectronics Circuits" has been written by **Manuel José Llamas Rodríguez** under their supervision, in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

And hereby to acknowledge the above, sign the present

Signature **Jordi Carrabina Bordoll** Signature **Lluís Terés Terés**

Barcelona, Spain, 18th July 2017.

Abstract

Organic/Printed Electronics are, day by day, increasing on interest, as new applications are being proposed and developed. This kind of technologies do not intend to compete directly with the Silicon-based well-established industry, but rather to complement it with new devices that are advantageous for certain situations, whether in terms of cost or others.

However, in the digital processing domain there is still much work to be done to, slowly but steadily, follow the steps of the conventional fabless model that rules today's semiconductor market. I am referring not only to progresses at fabrication level, but also on the field of Electronic Design Automation.

Our research group conceived a novel strategy to efficiently produce Printed Electronics digital circuit designs based on what we called Inkjet-configurable Gate Arrays, which takes advantage of digital printing techniques. The Inkjet Gate Arrays consist in matrices of transistors over flexible substrates that, after being connected by digital printing techniques, they describe logic gates, and thus circuits.

The work presented in this dissertation targets a specific stage of any common Integrated Circuit design flow, referred to as physical synthesis. Specifically, my contribution provides a new approach to the Placement and Routing problem, where circuits are mapped onto the Inkjet Gate Arrays in a technology independent yield-aware manner. I tackle the issue of dealing with different Printed Electronics technologies that might present distinct yield properties, usually due to the intrinsic high variability of current fabrication processes. In such cases, being able to effectively process the IGA's fault distribution information is key to ensure that the mapped circuits will be capable of working correctly, from a functional perspective. In addition to the yield awareness concept, the circuit personalization capabilities of the novel P&R heuristic proposed herein allow more mapping flexibility, depending on different possible reasons/purposes (e.g. congestion).

This approach is not only convenient for today's first steps of digital circuit prototyping over Organic Electronics, but also scalable to future technological improvements at yield level, and on sizes and integration density.

Resumen

La electrónica orgánica/impresa está continuamente creciendo en interés, con la aparición de nuevas propuestas y aplicaciones. Este tipo de tecnologías no pretenden competir directamente con las que provienen de la industria tradicional basada en Silicio, sino que tienen como propósito complementarla con nuevos dispositivos que proporcionen ciertas ventajas en determinadas situaciones, ya sea en términos de coste u otras.

Sin embargo, en lo que se refiere al campo del procesado digital queda mucho trabajo por hacer para, paulatinamente, ir siguiendo los pasos del modelo 'fabless' que rige el mercado de semiconductores actual. Este modelo consiste en la deslocalización entre los equipos de diseño y los fabricantes. Respecto a dicho progreso me refiero no solo a las mejoras que acontecen a nivel de procesos de fabricación, sino también en el campo de la automatización de los procesos de diseño.

Nuestro grupo de investigación concibió una novedosa estrategia para producir, de manera eficiente, diseños de circuitos digitales para electrónica impresa, basados en lo que denominamos Inkjet-configurable Gate Arrays, aprovechando las ventajas de la impresión digital. Estos Inkjet Gate Arrays consisten en matrices de transistores sobre sustratos flexibles que, una vez conectados mediante impresión digital, conforman puertas lógicas; las cuales, en su conjunto, materializan circuitos.

El trabajo presentado en esta tesis se centra en una etapa específica de cualquier flujo de diseño común de circuitos integrados, llamada síntesis física. En concreto, este trabajo proporciona una novedosa metodología para resolver el problema de ubicar y conectar, 'Placement and Routing', los circuitos sobre las mencionadas matrices de transistores, teniendo en cuenta su rendimiento, y con independencia de la tecnología de fabricación. Se aborda la manera de cómo tratar con tecnologías impresas diferentes, que puedan presentar distintos niveles de rendimiento, normalmente debidos a la alta variabilidad intrínseca a los procesos de fabricación actuales. En tales casos, un factor clave para asegurar que la colocación de los circuitos sea funcionalmente correcta es poder procesar de manera efectiva la información sobre la distribución de fallos de las matrices. Además del concepto de mapeo según el rendimiento, la novedosa heurística aquí propuesta proporciona la capacidad de personalizar los circuitos, lo que permite mayor flexibilidad en su construcción, dependiendo de distintas razones u objetivos posibles (p. ej. congestión).

Esta metodología no solo es conveniente para los primeros pasos que, en la actualidad, se están llevando a cabo en el desarrollo de prototipos de circuitos digitales para la electrónica orgánica, sino que también es escalable hacia nuevas mejoras en el rendimiento de las tecnologías de fabricación, así como en tamaños y densidad de integración.

Acknowledgements

I would like to express my gratitude to Jordi Carrabina and Lluís Terés, for giving me the opportunity to work under their supervision. Their passion on tech, from the very basics to the most complex systems, and their interest on applied and industrial research trends is inspiring. I came to Barcelona years ago with the objective of learning more cool stuff. I didn't plan on making a Ph.D. However, by staying, I was able to keep on learning from their vast knowledge and experience in many different areas, as from the many other talented researchers who I have worked with. Overall, I'm very pleased about my progress and achievements over this period.

I would like to give my thanks, for their help and insights at any given moment, to Mohammad Mashayekhi, Francesc Vila, Jofre Pallarès, Keith Sabine, Matthias Köfferlein, Shinya Takamaeda-Yamazaki, Eloi Ramon and Carme Martínez-Domingo.

Special thanks to André Reis and Jody Matos for their priceless feedback and help. Thanks to them coming from UFRGS we were able to discuss and clarify many of the topics covered herein and elsewhere.

Many thanks to the people at CEA-Liten for the warm welcome (and for handling all the bureaucratic nightmare of my stage). Especially to Micaël Charbonneau and Adrica Kyndiah. My time at CEA was short but awesome thanks to them.

Thanks also to Lluís Ribas-Xirgo, Josep Velasco and Borja Martínez for sharing their knowledge and experience in teaching/academic (and even life) matters.

Additional thanks go to Marc Codina and Jordi Guerrero for their support with a great many things.

Special gratitude for the sake of humanity's knowledge must go to Alexandra Elbakyan (Sci-Hub) and Aaron Swartz. I will gladly interact and discuss with anyone about any of these topics. Especially with those who lack interest in moving forward to the XXI century, improving current processes/practices. It's about time.

Publication list

2016:

- M. Llamas, L. Terés and J. Carrabina. "Technology Independent Yield-Aware Place & Route Strategy for Printed Electronics Gate Array Circuits". 7th International Conference on Computer Aided Design for Thin-Film Transistors - CAD-TFT 2016 Conference Paper (Beijing, China).

- J. Carrabina, J. Pallarès, F. Vila, M. Llamas, M. Mashayekhi, J. Matos, A. Reis and L. Terés. "Process Design Kit and EDA Tools for Organic/Printed Electronics". 7th International Conference on Computer Aided Design for Thin-Film Transistors - CAD-TFT 2016 Conference Paper (Beijing, China).

- J. Carrabina, M. Mashayekhi, M. Llamas, J. Pallarès, J. Matos, A. Reis and L. Terés. "Ink-Jet Configurable Gate Arrays". International Workshop on Flexible Electronics - WFE 2016 Conference Paper (Tarragona, Spain).

- M. Llamas, L. Terés and J. Carrabina. "Novel EDA techniques for Printed Electronics Circuits". Design, Automation and Test in Europe - DATE 2016, PhD Forum, Conference Paper (Dresden, Germany).

2015:

- J. Carrabina, M. Mashayekhi, M. Llamas, S. Ogier, T. Pease, M. Matti, L. Mika-Matti and L. Terés. "Customization Technology and Tools for Building Printed Circuits". 6th International Conference on Flexible and Printed Electronics - ICFPE 2015 Conference Paper (Taipei, Taiwan).

- M. Llamas, M. Mashayekhi, A. Alcalde, J. Pallarès, F. Vila, A. Conde, L. Terés and J. Carrabina. "Development of Digital Application Specific Printed Electronics Circuits: From Specification to Final Prototypes". IEEE/OSA Journal of Display Technology, vol. 11, no. 8, pp. 652-657. Article.

- J. Carrabina, M. Llamas, J. Matos, M. Mashayekhi and A. Reis. "Technology Mapping Tools for Building Optimal Circuits". 7th International Exhibition and Conference for the Printed Electronics Industry - LOPEC 2015 Conference Paper (Munich, Germany).

2014:

- M. Mashayekhi, M. Llamas, J. Pallarès, F. Vila, L. Terés and J. Carrabina. "Development of a Standard Cell Library and ASPEC design flow for organic Thin Film Transistor Technology". XXIX Conference on Design of Circuits and Integrated Systems - DCIS 2014 Conference Paper (Madrid, Spain).

- M. Llamas, M. Mashayekhi, J. Pallarès, F. Vila, L. Terés and J. Carrabina. "Top-down Design Flow for Application Specific Printed Electronics Circuits (ASPECs)". XXIX Conference on Design of Circuits and Integrated Systems - DCIS 2014 Conference Paper (Madrid, Spain).

- J. Matos, M. Llamas, M. Mashayekhi, J. Carrabina and A. Reis. "Optimization on Cell-library Design for Digital Application Specific Printed Electronics Circuits". 24th International Workshop on Power and Timing Modeling, Optimization and Simulation - PATMOS 2014 Conference Paper (Palma de Mallorca, Spain).

- M. Mashayekhi, M. Llamas, J. Pallarès, F. Vila, L. Terés and J. Carrabina. "Fault-Tolerant Inkjet Gate Array for Application Specific Printed Electronics Circuits". 6th International Conference on Computer Aided Design for Thin-Film Transistors - CAD-TFT 2014 Conference Paper (Nanjing, China).

- M. Llamas, M. Mashayekhi, J. Pallarès, F. Vila, L. Terés and J. Carrabina. "A novel Application Specific Printed Electronics Circuits (ASPEC) design flow". 6th International Conference on Computer Aided Design for Thin-Film Transistors - CAD-TFT 2014 Conference Paper (Nanjing, China).

- J. Carrabina, M. Mashayekhi, M. Llamas, C. Martínez-Domingo, E. Ramon, A. Alcalde, J. Pallarès, F. Vila, A. Conde and L. Terés. "Inkjet Gate Array: Novel concept to implement electronic systems". 6th International Exhibition and Conference for the Printed Electronics Industry - LOPEC 2014 Conference Paper (Munich, Germany).

- M. Mashayekhi, M. Llamas, C. Martínez-Domingo, E. Ramon, J. Pallarès, F. Vila, A. Conde, L. Terés and J. Carrabina. "Formalization of Design Rules and generation of related structures using PCells". 6th International Exhibition and Conference for the Printed Electronics Industry - LOPEC 2014 Conference Paper (Munich, Germany).

Table of Contents

List of figures

List of tables

List of acronyms

ADC	Analog to Digital Converter
API	Application Programming Interface
ASIC	Application Specific Integrated Circuits
ASPEC	Application Specific Printed Electronics Circuits
AST	Abstract Syntax Tree
BBC	Basic Bulk Cell
CAD	Computer-Aided Design
CLB	Configurable Logic Block
CMOS	Complementary Metal-Oxide-Semiconductor
CSV	Comma-Separated Values
DEF	Design Exchange Format
DFF	D Flip Flop
DRC	Design Rule Check
EDA	Electronic Design Automation
E^{2}PROM	Electrically Erasable Programmable Read-Only Memory **(**EEPROM)
ERC	Electrical Rule Check
FET	Field-Effect Transistor
FOSS	Free and Open Source-Software
FPGA	Field-Programmable Gate Array
GA	Gate Array
GDS/GDSII	Graphic Database System
HCG	Horizontal Constraint Graph
HDL	Hardware Description Language

IDM	Integrated Devices Manufacturers
I/O	Input/Output
IC	Integrated Circuit
IoT	Internet of Things
IP	Intellectual Property
IRDS	International Roadmap for Devices and Systems
ITRS	International Technology Roadmap for Semiconductors
KGO	Known Good OTFTs
LEA	Left-Edge Algorithm
LEF	Library Exchange Format
LSI	Large Scale Integration
LVS	Layout Versus Schematic
MOSFET	Metal-Oxide-Semiconductor Field-Effect Transistor
NMOS	N-type Metal-Oxide-Semiconductor
NVRAM	Non-Volatile Random-Access Memory
OE	Organic Electronics
OE-A	Organic Electronics Association
OLED	Organic Light-Emitting Diode
OOP	Object Oriented Programing
OPV	Organic photovoltaics
OTFT	Organic Thin Film Transistor
OTP	One-Time Programmable
P&R	Place and Route
PAL	Programmable Array Logic
PCell	Parameterizable Cell

PDK	Process/Physical Design Kit
PE	Printed Electronics
PLD	Programmable Logic Device
PMOS	P-type Metal-Oxide-Semiconductor
R&D	Research and Development
RF	Radio Frequency
RFID	Radio Frequency Identification
RTL	Register-Transfer Level
SaaS	Software as a Service
SoA	State of the Art
SoG	Sea-of-Gates
SC	Standard Cell
STA	Static Timing Analysis
TVD	Test Vehicle Description
VCG	Vertical Constraint Graph
VLSI	Very Large Scale Integration

1. Introduction

1.1. Printed/Organic/Flexible Electronics

Printed Electronics (PE), also referred to as Organic, Flexible or Large Area electronics, depending on which characteristic is emphasized, is an interesting approach to develop low cost non-critical electronic devices in a complementary “More-than-Moore” pathway.

The intention of this approach is not to compete or substitute traditional Silicon-based chips, which follow a “More-Moore” course of action, but to complement those devices with alternative electronic elements that might be more suitable/cost-effective for certain applications. “Moore” refers to the well-known Moore’s Law [1] from 1965 which states that the number of transistors per Silicon-based substrate would double every 2 years approximately, thus increasing integration density, performance and reducing chip costs per transistor, mostly associated to area and energy, since the speed is almost limited due to technology. Differences between “More-Moore” and “More-than-Moore” approaches can be explored in [2]. Basically, digital logic, microprocessors and mass memories are associated with “More-Moore” Complementary Metal-Oxide-Semiconductor (CMOS) scaling, whereas non-digital, different sensors, radio frequency (RF) devices, batteries, displays, etc. do not scale as fast as in Moore’s Law and therefore fall into the “More-than-Moore” category. Recent convergences between “More-Moore”, “More-than-Moore” and “Beyond CMOS” classifications make these divisions blurry and less relevant.

Periodically the semiconductor industry associations, formed by many institutions and firms, publish the set of documents, state-of-the-art reviews and market trends forecasts known as the International Technology Roadmap for Semiconductors (ITRS) [3]. These reports are good start points to explore recent events and tendencies in the vast knowledge areas covered by the semiconductor industry, for both “More-Moore” and “More-than-Moore” paths. A new roadmap has arisen recently, called International Roadmap for Devices and Systems (IRDS) [4], which is also worth checking.

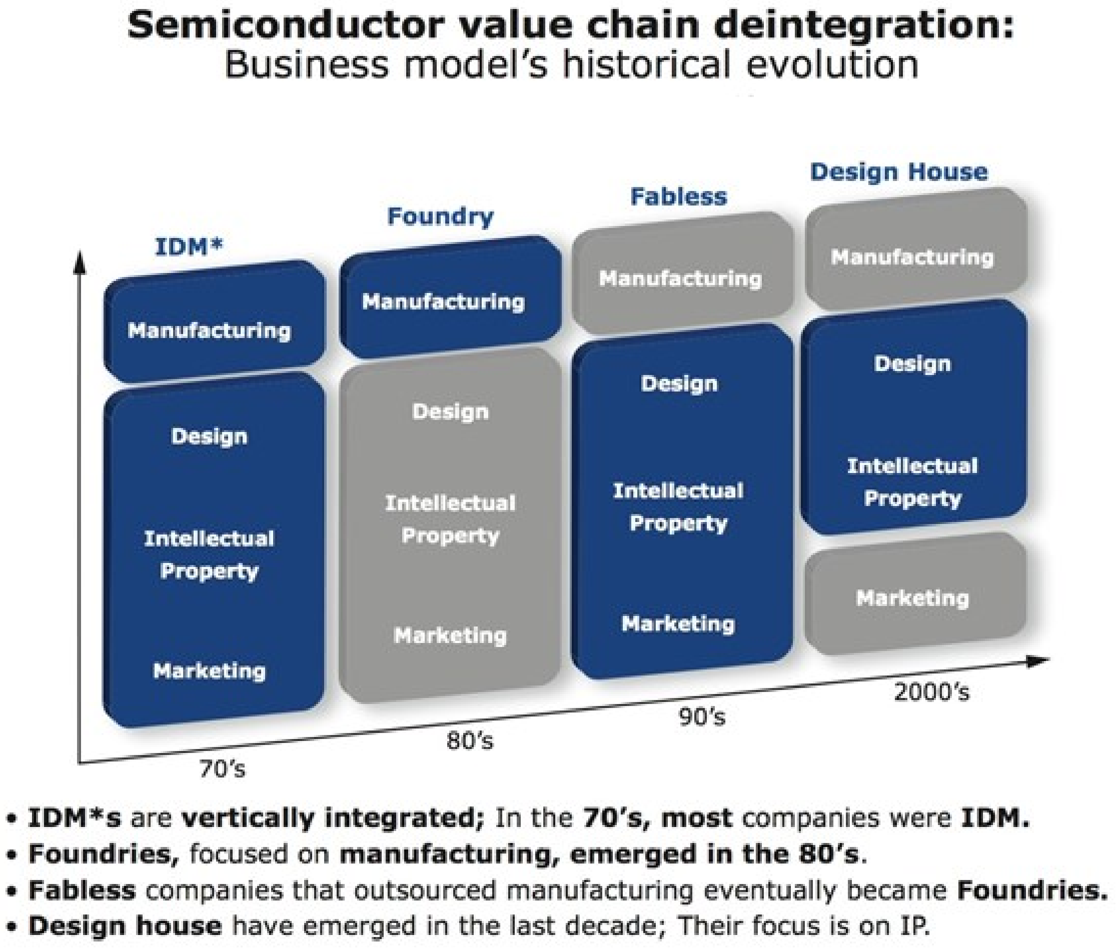
In a similar manner the Organic Electronics Association (OE-A) [5] makes their own reviews, forecasts and roadmap, but focusing only in the Organic Electronics (OE) market. They periodically organize conferences, courses, projects and gatherings; and they are a reliable source of state-of-the-art progress information as well as enablers for the common advancement of the printed technologies and their applications.

In Chapter 2, dedicated to the state of the art (SoA), the reader will find the latest PE R&D trends and achievements, with special emphasis at circuit and systems level.

1.2. Motivations for this work

While traditional silicon-based chips have achieved a massive penetration into today's market, due to their impressive reduction in sizes and prizes, variety of purposes, and even flexibility in the sense of reprogrammability; Printed Electronics advantages are not to be underestimated, although they are not yet in mass market except for some few niches such as Organic Light-Emitting Diode (OLED) displays and lighting, organic photovoltaics (OPVs) and touch panels.

As Figure 1 depicts in the traditional electronic industry the business model evolved from Integrated Devices Manufacturers (IDM), where they designed and fabricated everything in one same place, to Foundries/Fabless/IP/EDA/Design houses where distributed teams designed circuits according to each Foundry's technologies [6].



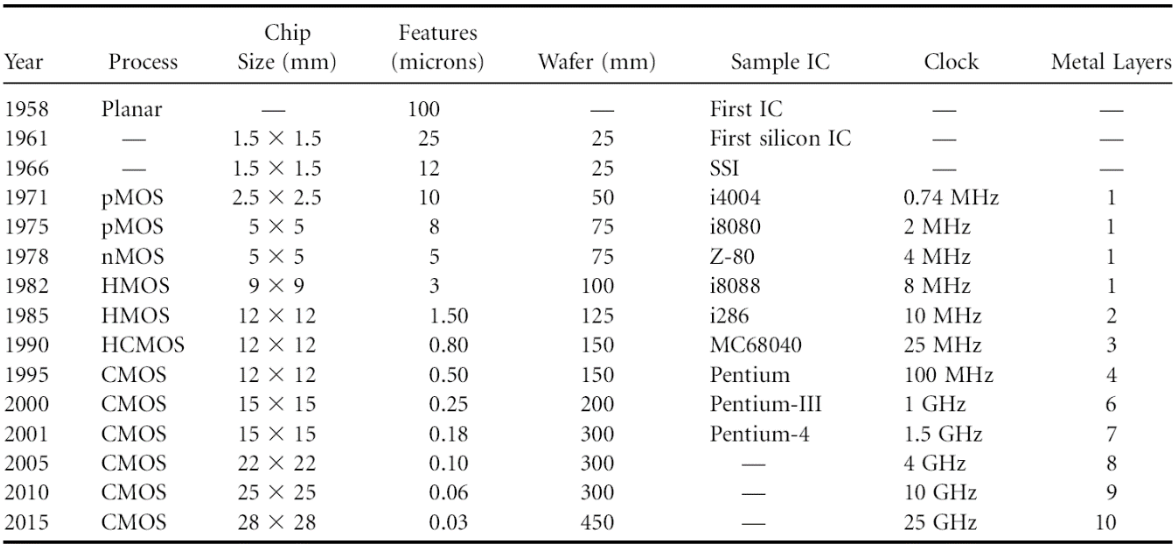
Semiconductor value chain deintegration:
Business model's historical evolution

- **IDM*s** are **vertically integrated;** In the **70's, most** companies were **IDM.**
- **Foundries,** focused on **manufacturing, emerged in the 80's**.
- **Fabless** companies that outsourced manufacturing eventually became **Foundries.**
- **Design house** have emerged in the last decade; Their focus is on IP.

Figure 1. Business model's historical evolution for electronics. [1]

[1] Image source: Successful Semiconductor Fabless conference, Yole Développement, Mar. 2013.

Printed Electronics, as a non-mature technology, seem to walk again a very similar pathway, not only at technological scaling level (although at slower pace than Moore's Law) but at business model as well, now still quite vertical but expected to move to horizontal. With the advantage that tech foundries are much more affordable than silicon-based massive clean room facilities. Thanks to this, design teams should be able to fabricate prototypes with the technology they have. Table 1, from [7], summarizes the evolution of Integrated Circuits (IC) over silicon-based technologies; and from it we can recognize and place what would be the current state of PE technology and make comparisons. Current ICs over PE technologies, only considering digital processing capabilities, would be situated somewhere around the equivalent to year 1970's approximately, in terms of feature sizes while they usually implement large area circuits.



Year	Process	Chip Size (mm)	Features (microns)	Wafer (mm)	Sample IC	Clock	Metal Layers
1958	Planar	—	100	—	First IC	—	—
1961	—	1.5 × 1.5	25	25	First silicon IC	—	—
1966	—	1.5 × 1.5	12	25	SSI	—	—
1971	pMOS	2.5 × 2.5	10	50	i4004	0.74 MHz	1
1975	pMOS	5 × 5	8	75	i8080	2 MHz	1
1978	nMOS	5 × 5	5	75	Z-80	4 MHz	1
1982	HMOS	9 × 9	3	100	i8088	8 MHz	1
1985	HMOS	12 × 12	1.50	125	i286	10 MHz	2
1990	HCMOS	12 × 12	0.80	150	MC68040	25 MHz	3
1995	CMOS	12 × 12	0.50	150	Pentium	100 MHz	4
2000	CMOS	15 × 15	0.25	200	Pentium-III	1 GHz	6
2001	CMOS	15 × 15	0.18	300	Pentium-4	1.5 GHz	7
2005	CMOS	22 × 22	0.10	300	—	4 GHz	8
2010	CMOS	25 × 25	0.06	300	—	10 GHz	9
2015	CMOS	28 × 28	0.03	450	—	25 GHz	10

Table 1. Integrated Circuit Process Improvement with Time. [2]

Like in the Fabless model, designs should be sent, after adequate Intellectual Property (IP) protection [8, 9], to Foundries to be fabricated.

From our point of view, it is especially relevant the Technology & Knowledge Transfer paradigm considerations and best practices, for collaborating with other R&D agents to set up the whole business environment. Some examples were commented in [10, 11].

In the case of UAB/IMB-CNM-CSIC a new set of industrial Research and Development (R&D) opportunities were explored and launched for these increasingly interesting organic/printed technologies. Several ambitious projects (TDK4PE [12], ASPEC-TDK [13], MEF3-IRX [14], PEC4 [15]) obtained public funding and were successfully carried out over the years by the research groups I have been involved with. The preferred modus-operandi of the partnerships to develop Printed Electronics

[2] Table source: V. G. Oklobdzija. [7]

prototypes is shown in Figure 2. The UAB/CNM fabless role would target a specific technology provided by a partner foundry, besides developing their own inkjet printed process. Then we would produce Design Kits, based on our preferred Top-Down Design Flows and open/free-tools environments, so that anyone (ourselves, or designers from the foundry itself, or any other third party fabless design team) could produce prototypes for tape-out in such technologies.

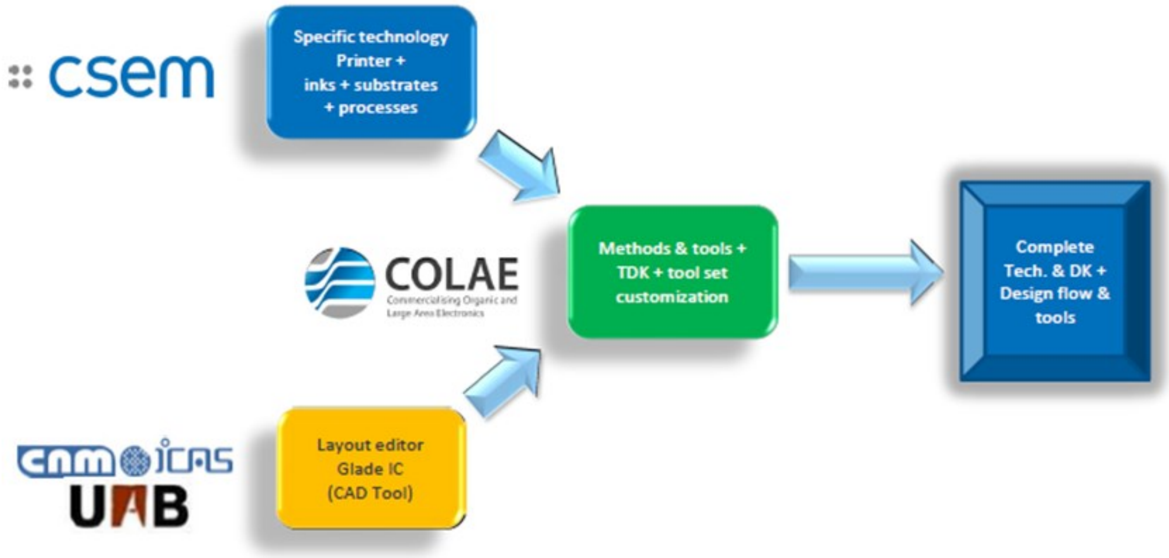

csem
Specific technology Printer + inks + substrates + processes
COLAE
Commercialising Organic and Large Area Electronics
Methods & tools + TDK + tool set customization
Complete Tech. & DK + Design flow & tools
cnm icas
UAB
Layout editor Glade IC (CAD Tool)

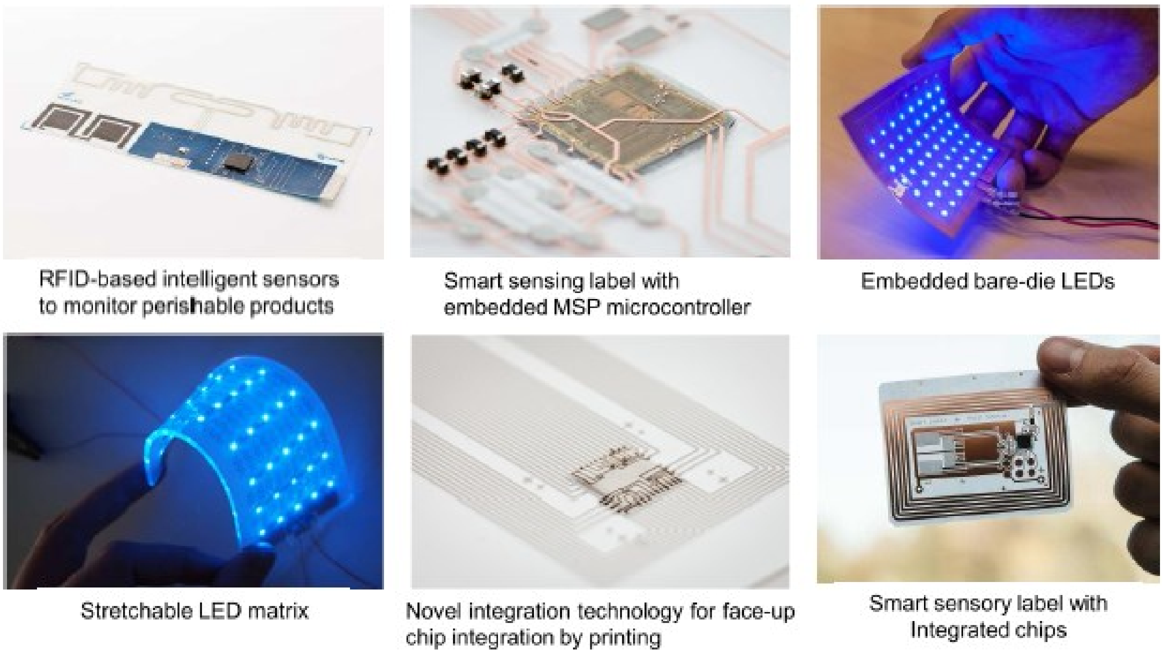
Figure 2. Example of business model for UAB/IMB-CNM-CSIC R&D interests. [3]

We have worked along with different partners, academic, industrial and fostering organizations, like CPI [16], NeuDrive [17], CEA [18], CSEM [19], ENEA [20], COLAE [21], CETEMMSA [22], Flexink [23], Infiniscale [24], Sensing Tex [25], 3D Micromac [26], PhoeniX Software [27], URV [28], UFRGS [29], TUC [30], UAlg [31]… Each one of them provided their contributions in their respective areas of expertise, from modelling and simulations, to design kits, fabrication technologies, prototypes, characterization and testing processes. The idea is to obtain circuits that work, first with very simple functionalities, and then capable of scaling according to the technologies available.

It has been demonstrated that it is possible to achieve interesting applications using organic/printed technologies. But even so, a typical question arises on standard silicon-based environments. Why would it be necessary to perform any kind of digital processing using these technologies, instead of using the more mature, small, reliable and efficient silicon-based devices? The answer is clear: heterogeneous integration. Having everything integrated, for example, sensing + processing + energy in the same saleable substrate, can profit from the benefits of printed technologies for certain applications, especially when flexibility is required (e.g. wearables, automotive, etc.). To establish the

[3] Image source: TDK4PE documentation. [12]

desired functionalities is key to set the choice among technologies. So far, hybrid approaches have been the most market-ready options, and they are a perfectly valid middle-point segment. In Figure 3 we can observe some examples of hybrid prototypes, including RFID-based sensors, labels, LEDs, and so on. These few samples can also perform computing works thanks to their Silicon-based associated chips (hence the terms intelligent/smart).



RFID-based intelligent sensors to monitor perishable products

Smart sensing label with embedded MSP microcontroller

Embedded bare-die LEDs

Stretchable LED matrix

Novel integration technology for face-up chip integration by printing

Smart sensory label with Integrated chips

Figure 3. Examples of hybrid electronics prototypes. [4]

But to obtain the desired integration in Printed Electronics it is necessary to be able to make stand-alone functional circuits. The typical structures for achieving circuits are basic logic gate libraries (inverters, AND, NAND, NOR…) that allow digital circuitry, analog cells for signal conditioning, Analog-to-Digital Converters (ADC), memories and I/O structures.

The main goal of the EDA field [32-34] has always been to automate design tasks and processes so that is easier, faster and, ideally, more reliable to obtain final electrical products. For this purpose there are many strategies, algorithms and tools from different vendors, all along the full design and fabrication flows, developed through many years for conventional Large Scale Integration (LSI) and Very Large Scale Integration (VLSI) target circuits. In this EDA context we can see that there is much R&D to be done and improved for helping Printed Electronics designers and, by extension, its industry. Sometimes it is possible to benefit from research and achievements done in Silicon-based environments, but in other cases it is necessary to define new ideas and strategies to deal

[4] Image source: Holst Centre. [292]

with issues and particularities that are not contemplated in those areas, such as the reliability of the materials, fluidic properties, degradation, high variability, yield and fault distribution differences; even within the same technological process.

1.3. Digital Circuit Design styles

As briefly described in [35, 36] we can classify circuit design styles according to different criteria; most importantly regarding flexibility versus design effort. In this scenario flexibility corresponds to whether the physical design (layout), to be later fabricated, is going to be designed in a completely handcraft manner (full-custom); by using different libraries (semi-custom); or by using IP components (processors, buses, NoCs, peripherals, memories, ADC…) to build complex systems (SoC, MPSoC). This work is centered in the semi-custom design methodology, according to the current state of the art of the technology. Its basic choices as predesigned and pre-characterized libraries (Standard-Cells) or predefined templates with prefabricated transistors or gates that will be custom connected (routed), according to the desired functionality (Gate-Array). We can find comparisons between design styles at low level of design abstraction (full- and semi-custom), including most notable benefits and drawbacks of each methodology [37-39].

Table 2 depicts a brief comparison of the main differences between circuit design styles. The colors of the arrows illustrate whenever a Gate Array (GA) style choice is advantageous or not, since it is more similar to the Inkjet Gate Array (IGA) design style that we will propose to use.

Design style	Design flexibility	Design effort	Integration density	Prototyping costs	Performance
Full-Custom	-	-	-		-
Macro-Cells				+	
Standard-Cells	+	+	+	-	+
Gate Array					

Table 2. Design style comparison.

1.4. Design Flows & Process Design Kits

1.4.1. ASIC overview

A very simplified traditional (Silicon-based) complete manufacturing flow is shown in Figure 4.

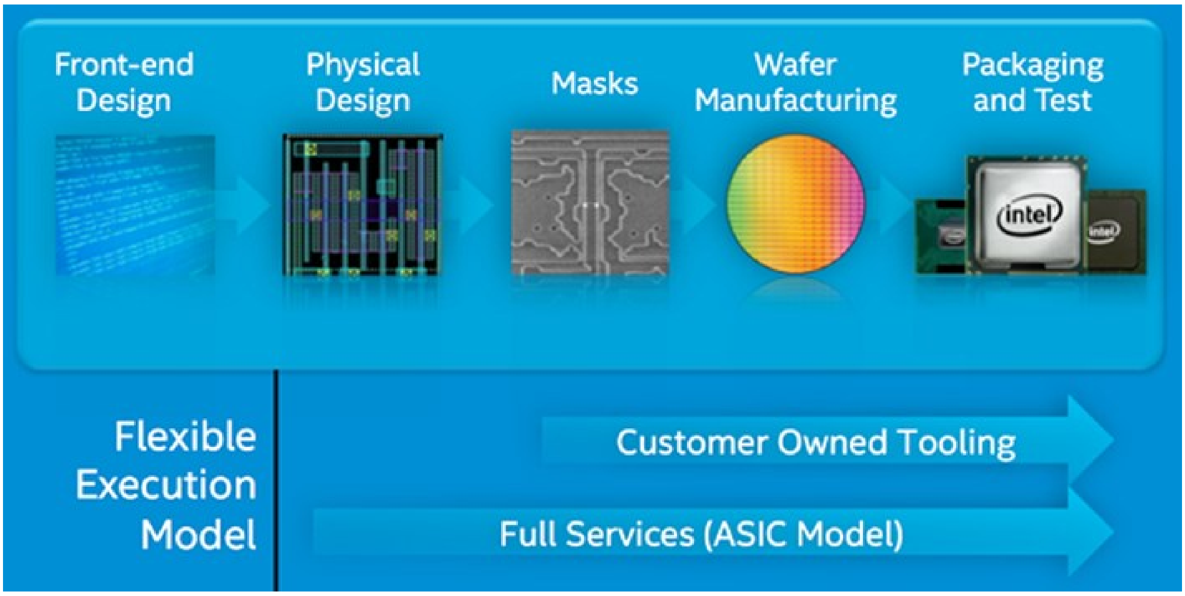


Figure 4. Simplified complete manufacturing flow. [5]

This complete flow typically starts with a Register-Transfer Level (RTL) specification that describes the functionality of the system to be produced [40]. The most common Hardware Description Languages (HDL) for RTL coding are the VHDL and Verilog standards [41]. References for better understanding VHDL and Verilog languages and usage can be found in [42-51], and also some concise comparisons in [52, 53]. There are many software tools capable of creating (as an Integrated Development Environment IDE), parsing, compiling, elaborating, synthesizing, simulating and verifying HDLs into different hardware platforms (depending on the vendors). Typically, although not necessarily, VHDL is more used to build small synchronous designs or virtual components (IPs) in Field Programmable Gate Arrays (FPGA), from firms such as Xilinx, Lattice, Altera (now part of Intel), and many others who provide their own IDEs; while Verilog is more used for other design flows such as for Cadence, Mentor Graphics or Synopsys tools (commonly known as “the big 3” in the EDA business). It is also possible to start with a schematic circuit description, instead of using an RTL abstraction. In both cases components can be created or reused from other libraries, included the ones supplied by the vendors for their devices.

[5] Image source: Intel. [291]

A physical design, also called layout, represents the different shapes that will generate the so-called masks that will be used for fabricating the circuits. Circuit layouts need to be created in compliance with the design rules of the targeted technology, that report of the feature sizes set to obtain a required yield and quality. These are verified via Design Rule Checks (DRC). Furthermore, circuit electrical rules are verified via Electrical Rule Checks (ERC). Automatic physical verification procedures, such as DRCs and ERCs, are also referred to as signoff checks. Typically they are included into PDKs to check the correctness of designs before sending them to the foundries. They usually have visual debugging capabilities.

Another kind of signoff procedure is a Static Timing Analysis (STA). Such analysis needs to be done for ensuring that timing requirements are satisfied [34, 54, 55]. The STA is a process that checks exhaustively that a circuit has the proper timing. It does so by calculating all the possible delays: intra-cell delays, paths, cycles, etc. An STA is not intended to check for functional correctness (i.e. functional verification).

As a final corroboration step an LVS is performed. For that, the layout has to be extracted into a netlist which includes Input/Output (I/O) pins, transistors and nets. This netlist is then compared to the schematic netlist, which should contain the same information. If both netlists match perfectly structurally then they are both representing the same circuit functionality and, only then, the layout is ready for fabrication (tape-out).

Fabrication processes start with producing the masks to be used on the wafers as photolithographic (e-beam or others) methods require, whether they are for addictive or subtractive techniques.

After fabrication, there will be a packaging step to encapsulate and protect the chips. Their I/O pins will make them connectable to other devices. Lastly, the chips will have to pass functionality tests and endure reliability tests, depending on the maximum stress and expected lifetime given by the manufacturer's warranty, to be ready for the market.

Evidently, each one of the steps in this flow can be further decomposed into smaller steps, as shown in Figure 5. Special attention is given to the physical design part, which is essential for the scope and goals of this dissertation.

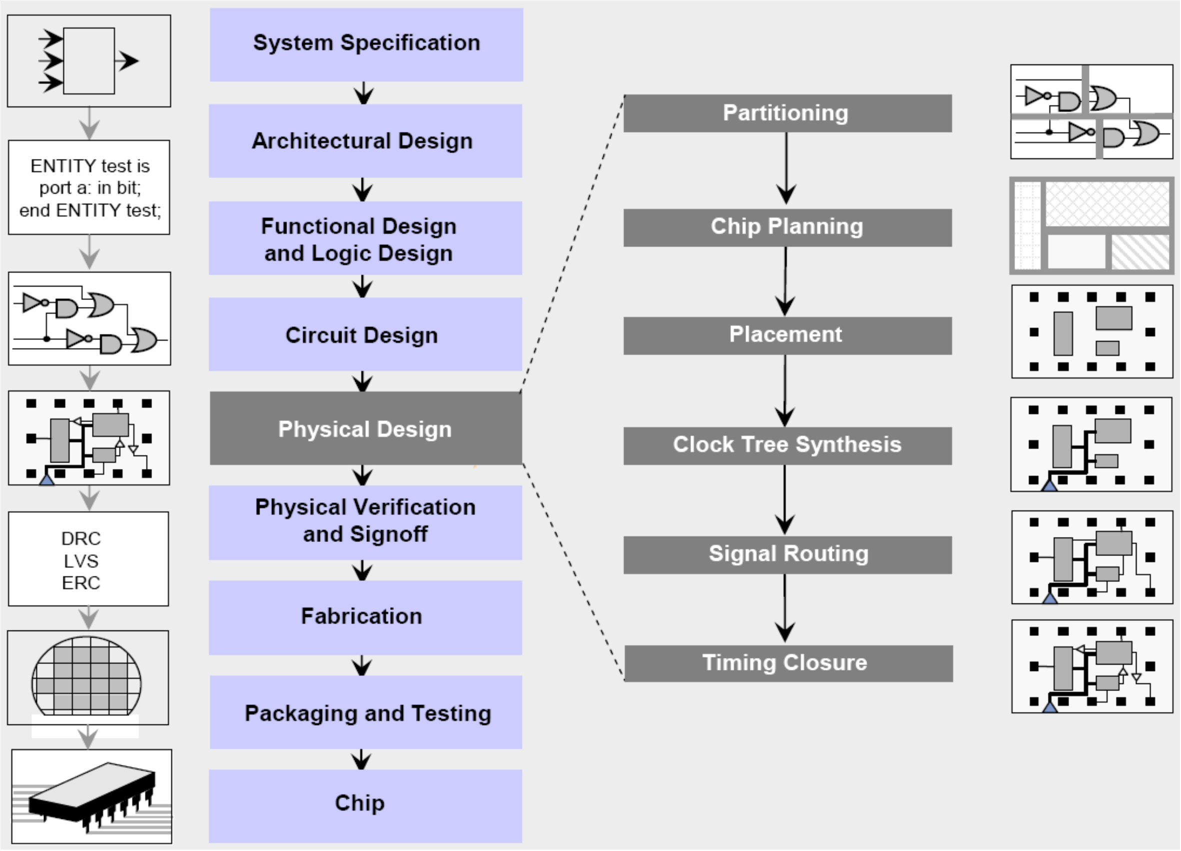


Figure 5. VLSI typical flow. [6]

Usually, physical design, where this research is focusing, comprises several stages that might include Partitioning (dividing the design into blocks according to the preferred strategy), Floorplanning (dividing the available substrate area into regions), Placement (of all cells assigned to each region), Clock Tree synthesis (for providing proper synchronized clock signals to every sequential component of the circuit), Routing (connecting locally and globally, elements, components and regions), and Timing Closure (which verifies that the entire design meets the timing constraints specified, such as the critical path, which determines the maximum clock frequency). An interesting reading comprising several relevant areas is the selection given in [56].

1.4.2. ASPEC overview

The design flows needed to be adapted to PE/OE technologies conforming to their inherent particularities. An initial idea was proposed in [57] trying to develop a semi-custom design methodology targeting Standard-Cell [58, 59] and Inkjet-configurable Gate Array methodologies [60-62] tailored to the particularities of the technologies. The

[6] Image source: A. B. Kahng. [182]

complete flow set-up for this research together with the related EDA tools is shown in Figure 6, as given in [62].

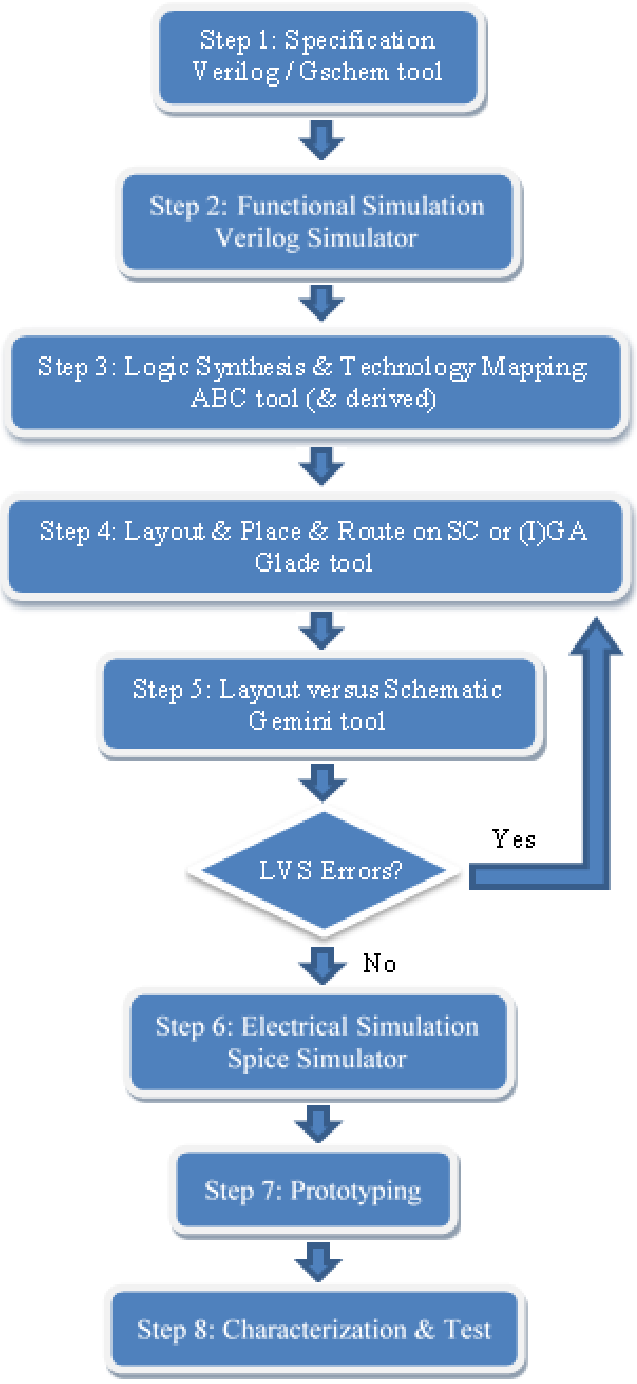

Step 1: Specification
Verilog / Gschem tool
Step 2: Functional Simulation
Verilog Simulator
Step 3: Logic Synthesis & Technology Mapping
ABC tool (& derived)
Step 4: Layout & Place & Route on SC or (I)GA
Glade tool
Step 5: Layout versus Schematic
Gemini tool
LVS Errors?
Yes
No
Step 6: Electrical Simulation
Spice Simulator
Step 7: Prototyping
Step 8: Characterization & Test

Figure 6. Complete ASPEC flow.

We can observe that this flow covers the entire chip production, from specification to final prototypes. The selection and usage of Free and Open-Source Software (FOSS) tools aligns perfectly with the current low complexity of PE prototyping technologies available to designers[7]. This way, small design teams do not require spending on costly infrastructures and expensive licenses. In this research work my goal is to prove that this flow is effective and can be reapplied for any ASPEC as long as the corresponding technology files are available.

As commented for VLSI environments, a design can start either with a specification in HDL (in our case Verilog) or with a schematic drawing. For schematics we selected Gschem; part of the gEDA tool distribution [63]. For the HDL circuits expressed in Verilog I selected Icarus Verilog [64] for synthesis, and GTKWave [65] for waveform screening of functional simulations.

Logic synthesis and technology mapping are key steps for any technology, including PE. In this area, there are EDA tools for circuit optimization such as minimizing the number of gates (and consequently transistors) by applying Boolean algebra and producing combinational logic reductions. The selected tool for this purpose is ABC [66]. Area is a significant factor due to the fact that PE technologies are still showing low integration density (in terms of transistor per mm^2). It is obvious that if we manage to reduce the gate count we can integrate the same chip functionalities while saving substrate area. The technology mapping process, placed between logic and physical synthesis, takes into account technology dependence after a technology independent synthesis step. In a semi-custom design methodology, circuit implementing requires an associated library of digital cells.

For instance, in some technologies such as unipolar (i.e. PMOS or NMOS only) some particular gates might not be available, due to the low performance or functional inabilities of series transistors. Hence it is necessary to specify a differentiated cell library for every technology. In the particular case of the PE technologies that I got access to, most of them are based in PMOS-only design style. This is due to the fact that in many OE technologies, the NMOS transistors don't reach a good enough mobility and conductivity as the P-type ones [67, 68]. Therefore, for most foundries8, the CMOS design style [69] is not yet enabled for digital circuits. However, CMOS is vastly preferred in silicon-based VLSI systems mainly because of its low static energy consumption and high noise immunity, which makes it more robust. A detailed explanation of interconnection noise can be found in [70].

Once we have minimized and mapped the circuit to a particular library, we continue with the physical design, either following a Standard Cell or Gate Array approach, using layout tools such as Glade [71] or KLayout [72]. The placement and routing will be done according to the circuit netlist.

As mentioned in the VLSI case, after physical synthesis it is necessary to perform physical verification checks, including DRC, ERC and LVS, to check wether the layout

[7] More advanced technologies (e.g. Thinfilm Technologies or PragmatIC) are oriented to application markets, thus not available for third parties.

[8] Except those mixing organic (for PMOS) and oxide (for NMOS) transistors.

is well drawn and matches the circuit netlist. For this, we use both Glade and Gemini LVS tools [73].

Electrical simulations are needed in order to check performances including parasitic capacitances and resistances. In this case we use customized models for Spice simulators [74] such as Ngspice [75] or AIM-Spice [76].

As a last design step, Glade is used to generate the GDSII files that are sent to the foundry to be fabricated in the targeted process. The obtained circuits go through a series of characterization and test procedures. These will determine, for instance, the overall yield achieved (and the fault distribution of the OTFTs, for technologies with mid-yield in OTFTs). This characteristic is key for the proposal of our EDA tool strategy, as it will be explained in detail later on, concretely for technologies that do not achieve a high yield (close to 100%). Their results are required to obtain circuits out of mid-yield OTFT fabrication processes and will help to fine tune the models and design rules for future improvements on the targeted technologies.

1.5. Digital printing for personalized digital circuits

The Inkjet-configurable Gate Array (IGA) design style concept, introduced in [60] and developed through [77-82], relies on a crucial PE characteristic: the ability to use digital printing for metallization on top of different PE/OE substrates. As described in detail in [83] digital printing is a fabrication process that allows reproducing text, images and patterns in different coordinates of the substrate with control and precision. Since the design to be printed is described digitally (e.g. in our case a layout represented in the standard Graphic Database System GDSII format [84-86]), mapping circuits onto PE/OE substrates with prefabricated transistors in a GA disposition should be cheaper and more reliable than having to fabricate different masks for every circuit. Furthermore, it allows circuit-by-circuit personalization as a key aspect.

Traditionally in silicon-based environments designs are imprinted onto wafers by using a unique set of masks for a particular chip design. This allows for mass production (high volume) of that design over a big amount of wafers. Such masks are very expensive and are fixed. They only permit the production of one particular design repeatedly for a targeted process technology. Whenever design changes, masks (equivalent to plates in the conventional printing industry) also have to be changed (analog printing).

The classical silicon Gate Array (GA) or Sea-of-Gates (SoG) approach allowed circuit personalization for specific circuits by changing only metallization (and via) masks at the last stages (layers) of the fabrication process. Cost reduction was achieved by sharing the substrates among different circuits but they still need to fabricate those individual masks.

The IGA concept starts from that GA or SoG concept and moves masks (for equivalent analog printing) into files (for digital printing).

In this scenario the circuits are fabricated in two separated steps:

1) Fabricating the elementary devices on the substrate (e.g. 50 micron thickness of PolyEthylene Naphthalate PEN). The IGA substrate consists in an array of unconnected OTFTs (or cells), deep-metal wire cells and I/O pads.
2) Personalizing the circuit on top of that IGA substrate; which consists in mapping a circuit over the IGA template by Placing and Routing gates and cells using the OTFTs available.

The circuit layout obtained after the Placement and Routing is digitally printed over the IGA substrate, by using any of the suitable customization technologies available. Some of these techniques have been explained in [80]. In this work this is done at transistor-level, which allows more flexibility. While the IGAs could be designed at digital cell-level and even higher cell-level hierarchies (e.g. logic blocks in FPGAs). These options increase the current integration density at the cost of limiting circuit flexibility.

With this IGA design methodology, digital printing is highly beneficial, from the point of view of manufacturing costs, as opposed to the use of masks which would only allow to map circuits in a fixed pattern. However, it is slower since it is not as highly parallelized fabrication process as when using masks.

Moreover digital printing also provides the ability to change the function of the design imprinted over the different substrates, by changing the Placement and Routing design depending on the restrictions that we want to impose.

1.6. Placement and Routing problem/strategy

Placement and Routing (P&R), also referred to as physical synthesis, is one of the most important and challenging process in any circuit design flow. It is one of the key steps due to the time and efforts required by any design team, especially when scaling to VLSI chips and beyond. Thus, P&R design automation has been an historical focus point of the scientific and industrial community, in order to introduce into the market more reliable, compact and higher performance electronic products in less time (time to market).

In our case, the Inkjet-configurable Gate Array design style, together with the digital printing capabilities and the two-step fabrication process, provides us with the chance of performing P&R in a yield-aware manner trying to overcome, one of the main bottlenecks for many PE technologies.

The yield awareness is a critical constraint in our methodology. It is evident that, the lower the yield, the lower it is the probability of obtaining a circuit working correctly. This relation is not directly proportional; i.e. having a yield of 70% in terms of transistors, with a random failure distribution, will dramatically reduce the chances of obtaining a working circuit to almost none. This is especially relevant for mid-yield/high variability PE technologies, where several OTFTs could potentially fail at completely random locations in the substrate.

Whenever the expected yield of a technology is lower than the desirable 99.99% our strategy introduces a new step in between the other two fabrication processes (fabrication of the IGA substrate and circuit personalization). This middle step consists in a testing and characterization process of all OTFT devices present in the IGA substrate, resulting in a Known Good OTFTs map (KGO) that list the location of the working transistors in the array. This KGO map is very likely to change from foil to foil. Some reasons causing failing OTFTs can be found in [87]. The Placement and Routing will produce circuits that take this KGO map into account. Thus, the P&R will adapt to the fault distribution to produce working circuits out of mid-yield OTFTs foils since we eliminate the possibility of using faulty OTFTs. Since we rely on real test data, we do not need to use probabilistic models over virtual layouts for yield prediction, as in [88].

This middle characterization stage has also currently some associated drawbacks: (1) equipment, labor and time required for individual OTFT testing (i.e. using probe stations); and (2) pads required in the layout design for the probes to contact and test each OTFT. Moving to cell-based IGA these pad area requirements are lowered so it is possible to increase the integration density.

A detailed review of the P&R state of the art is provided in chapter 2, while our specific proposal is developed in chapter 4.

1.7. Goals of this work

In summary, the main goal of this research work is to develop the EDA algorithms that provide full physical synthesis capabilities for Inkjet-configurable Gate Array Printed Electronics circuits.

We will be able to obtain functional circuits by the means of automatically generated layouts, according to our IGA designs style focusing on digital printing customization.

Since the fault distribution at transistor level might vary from foil to foil, even in the same technology and fabrication run, the powerfulness of our approach is that we can run the algorithm as many times as required for every specific PE circuit after its characterization to know each foil specific KGO map (whenever necessary, according to the expected yield of each technology), thus allowing its individual wiring, what maximizes yield at circuit level.

The intention is to provide a good P&R solution, according to the given constraints. For that purpose, this dissertation will detail how a conjoint constructive greedy P&R heuristic was devised, demonstrating its convenience for PE-based circuits.

This research work is intended to increase PE circuit design automation, hence reducing the time and efforts needed to successfully implement PE circuits, depending on the technology addressed and its corresponding yield. To clarify, the title of this dissertation includes the term Digital^2, which is just a reference to both digital circuits and digital printing.

This methodology can also be used for other purposes than circuit yield increase. Functionalities such as circuit personalization, for which we can print individual circuit identification codes (such as a MAC internet address), can avoid the use of complex E^{2}PROM or NVRAM devices.

In the following chapters of this dissertation the concepts will be explained to the reader, as well as the current state of the art, our approach to the problem at hand, the strategy and means to solve it in the most efficient way as currently possible, results obtained and final observations.

2. State of the art for PE & EDA

2.1. Digital printing techniques for building digital circuits

Analog printing techniques refer to the traditional methods of printing. In the conventional industry this would mean relying on a set of predefined plates that will imprint the same patterns (e.g. letters) into a targeted substrate (e.g. paper) over and over again. This allows for mass production (high volume) of a unique design. Analog printing methods are, for instance, flexography, gravure, offset, and screen printing. In the semiconductor industry the plate concept is the same, yet it has a different nomenclature. In this case chip fabrication relies on a set of predefined masks which will allow to imprint the designs onto and within the substrates (i.e. wafers) by combining a series of advanced fabrication processes such as photolithography, deposition, oxidation, etching, etc…

On the other hand digital printing techniques do not require masks for fixed designs. They allow to change the pattern to be imprinted over the substrate on demand. This provides more flexibility to produce different designs and lower fabrication costs, whenever such designs are intended to vary, regardless of the amount to be produced. These techniques consist in depositing the inks over the substrates without applying any kind of contact ever (and thus no pressure) on them. Some current digital printing methods are Inkjet (and other higher resolution variants such as Electrohydrodynamic Inkjet printing or Superfine Inkjet printing), and Aerosol jet printing.

The following Figure 7, from the OE-A, as it appears in their original 6th Edition report, gives a brief idea of what kind of printing techniques can be used to produce PE/OE applications, comparing their current usual resolutions and throughputs.

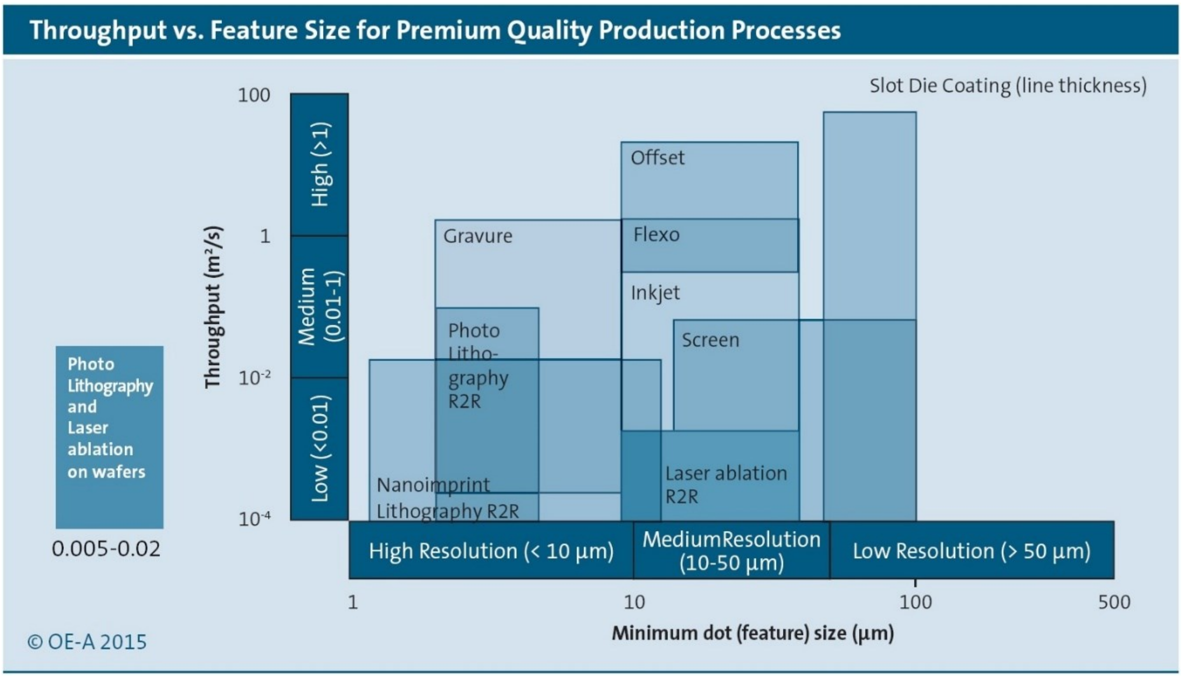


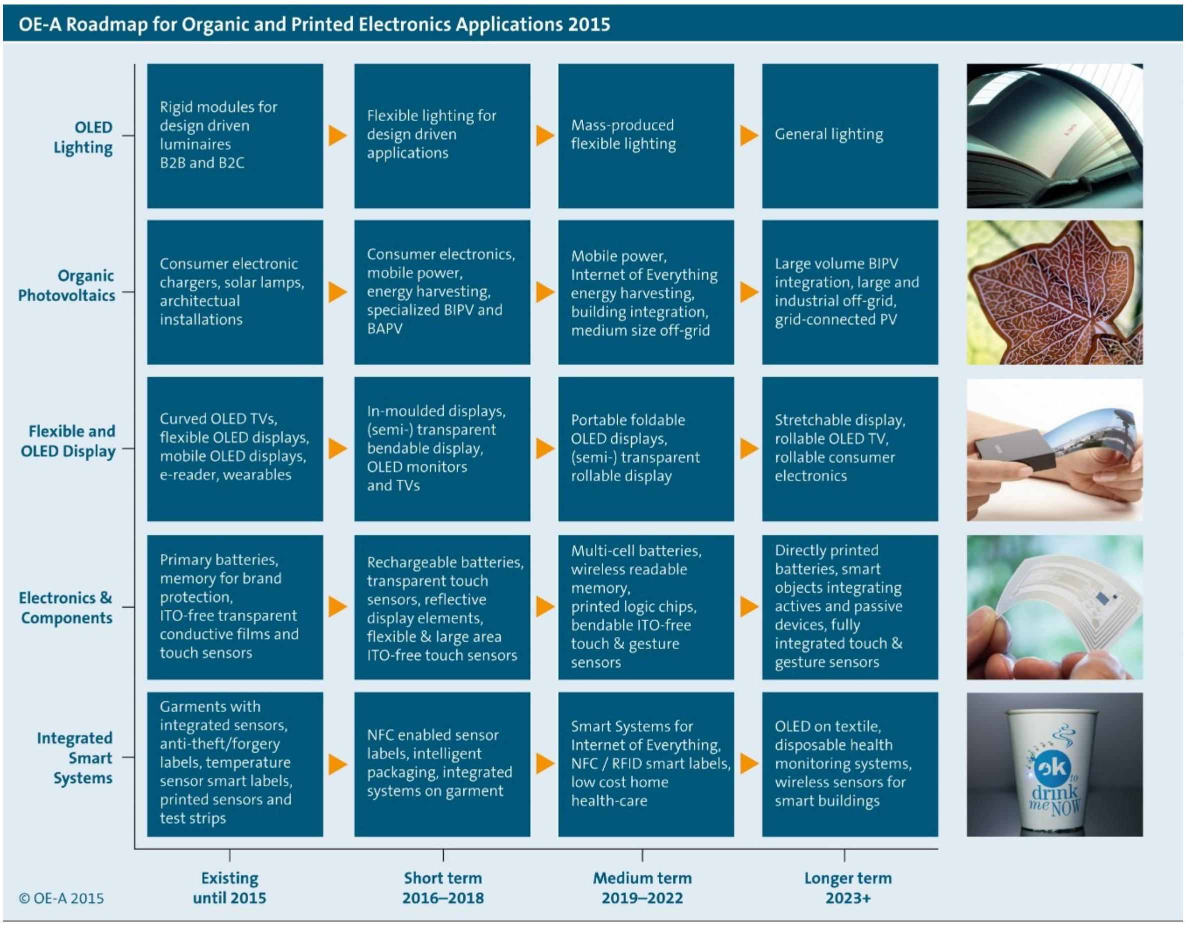
Figure 7. Resolution and throughput for highest quality levels of common deposition and patterning technologies for organic electronics. [9]

As it could be expected, techniques based on photolithography, have a very high resolution and accuracy, but also lower throughput levels; while other processes that require less precision, such as offset, flexography, inkjet or screen-printing, have a much better production rate. In-depth comparisons, advantages and drawbacks between printing techniques, whether analog or digital, can be found in [77, 89, 90].

2.2. Driving applications for PE circuits

In Figure 8 and Figure 9 it is possible to see the latest roadmap and examples of current and potential applications summarized by the OE-A.

[9] Image source: OE-A. [5]



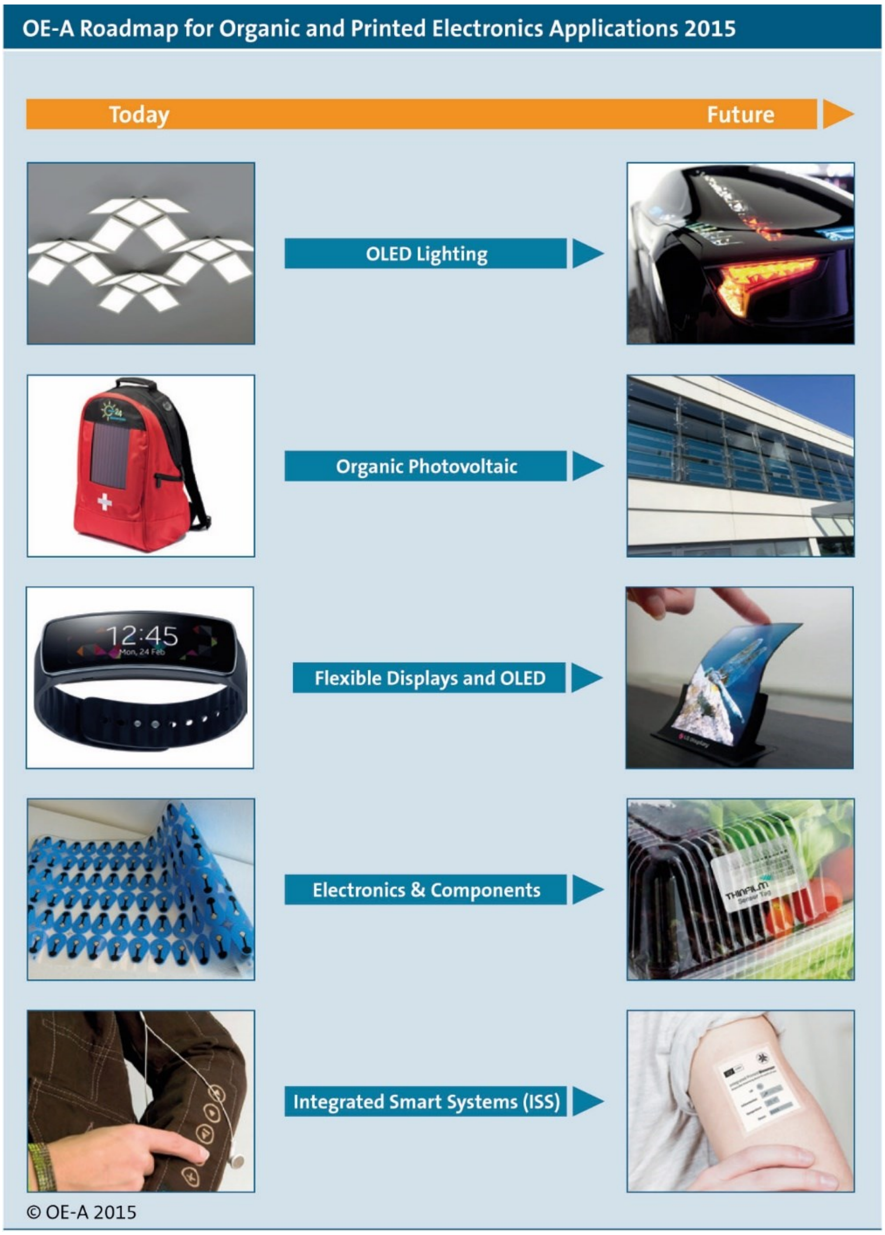
OE-A Roadmap for Organic and Printed Electronics Applications 2015

	Existing until 2015	Short term 2016–2018	Medium term 2019–2022	Longer term 2023+
OLED Lighting	Rigid modules for design driven luminaires B2B and B2C	Flexible lighting for design driven applications	Mass-produced flexible lighting	General lighting
Organic Photovoltaics	Consumer electronic chargers, solar lamps, architectural installations	Consumer electronics, mobile power, energy harvesting, specialized BIPV and BAPV	Mobile power, Internet of Everything energy harvesting, building integration, medium size off-grid	Large volume BIPV integration, large and industrial off-grid, grid-connected PV
Flexible and OLED Display	Curved OLED TVs, flexible OLED displays, mobile OLED displays, e-reader, wearables	In-moulded displays, (semi-) transparent bendable display, OLED monitors and TVs	Portable foldable OLED displays, (semi-) transparent rollable display	Stretchable display, rollable OLED TV, rollable consumer electronics
Electronics & Components	Primary batteries, memory for brand protection, ITO-free transparent conductive films and touch sensors	Rechargeable batteries, transparent touch sensors, reflective display elements, flexible & large area ITO-free touch sensors	Multi-cell batteries, wireless readable memory, printed logic chips, bendable ITO-free touch & gesture sensors	Directly printed batteries, smart objects integrating actives and passive devices, fully integrated touch & gesture sensors
Integrated Smart Systems	Garments with integrated sensors, anti-theft/forgery labels, temperature sensor smart labels, printed sensors and test strips	NFC enabled sensor labels, intelligent packaging, integrated systems on garment	Smart Systems for Internet of Everything, NFC / RFID smart labels, low cost home health-care	OLED on textile, disposable health monitoring systems, wireless sensors for smart buildings

© OE-A 2015

Figure 8. OE-A Roadmap for Organic and Printed Electronics Applications 2015 (1). [10]

[10] Image source: OE-A. [5]



OE-A Roadmap for Organic and Printed Electronics Applications 2015

Today → Future

OLED Lighting

Organic Photovoltaic

Flexible Displays and OLED

Electronics & Components

Integrated Smart Systems (ISS)

© OE-A 2015

Figure 9. OE-A Roadmap for Organic and Printed Electronics Applications 2015 (2). [11]

As shown in these figures, market driven applications that are already available are, for instance, OLED lightings, photovoltaic chargers, OLED displays for multiple consumer electronics devices, batteries, conductive films, sensors and labels of various types. All of these examples can be introduced in the industries of Internet of Things (IoT), automotive, health care, wearables, packaging, leisure… With the continuous improvement of the technologies at every step of the manufacturing chain the applications are expected to become more flexible, efficient, reliable and complex; allowing more possible functionalities while benefiting from the low costs associated to Printed

[11] Image source: OE-A. [5]

Electronics. Some examples in the scientific literature can be found, for instance, about organic photovoltaics [91, 92], flexible batteries [93, 94], electro-optic devices [95-97], OLED displays [98-100], logic and memory components, including Field Effect Transistors (FETs) and Organic Thin Film Transistors (OTFTs) [101-106], sensor arrays [107-113] and Radio Frequency Identification (RFID) tags [114-118]. Even a microprocessor was shown in [119, 120].

More examples of recent accomplishments on PE circuits and systems can be found in [59, 121-126], even with Programmable Array Logic (PAL) [127]. Additional progresses and findings with different focuses and objectives are excellently explained in the dissertations written by my colleagues [82, 90, 128]. Another recent interesting dissertation, related to OTFT modelling and simulation, is [129]. In relation to Design Automation for PE applicability I thoroughly encourage the reader to review [130, 131] as well, since they are closer to the scope of this work.

In the case of our research group we explored different circuit design styles, from a full-custom approach to Standard-Cells (SC) and Gate Array (GA) (also called masterslice) based configurations. The first ones apply a similar vision of the Application Specific Integrated Circuits (ASIC) model [132, 133], and therefore were adapted to what we called Application Specific Printed Electronics Circuits (ASPEC). The latter was a novel concept introduced in [60], and extended through [77-82], that provides more flexibility and deals better with Printed Electronics intrinsic characteristics and issues. Over the years, our research group proposed methodologies, design flows and Electronic Design Automation (EDA) / Computer-Aided Design (CAD) tools for minimizing all kinds of costs (free licensed, portable, customizable, standardized and powerful, yet user friendly) [58, 62, 128, 134-140]. A recent review of our own group's inkjet fabrication process can be found in [141].

For further knowledge about PE/OE and possible applications I suggest the books [142-144].

2.3. Relevant EDA tools

The EDA tool landscape is quite vast, addressing a wide range of missions at any step of any chip design flow. Besides each tool's purposes, there is also another major distinction to be done among such software products: usage fees.

In the commercial domain we can clearly highlight the so called 'big 3' companies: Cadence, Synopsys and Mentor Graphics (recently acquired by Siemens). All of them provide tools and licenses for any of the ASIC designs steps. Their business models cover many aspects of R&D, including monitoring the protected/FOSS/academic breakthroughs for incorporating novel ideas (the ones that have profit potential, of course) into their design flows and tools [56]. We can remark the pertinent solutions that each one of them provide:

1) Cadence: Within the Cadence tool portfolio the most interesting ones for the topics covered by this dissertation are Cadence Encounter RTL Compiler, and now successor Genus Synthesis. They are both complex tools for RTL (usually Verilog) and physical synthesis (P&R) of VLSI systems. They work together with the Innovus placement tool and Virtuoso suite for schematics and layouts. Most of their features can be checked at their website [145]. They accept standard format files for different steps of the design flow such as Verilog, LEF/DEF files, GDSII, Liberty, map, etc.
2) Synopsys: In the case of Synopsys we have the Design Compiler for RTL synthesis, and IC Compiler II for P&R. These tools require the Verilog netlist plus an additional design constraint file .sdc. As with any other tool, many variables/commands can be passed/executed via scripts; commonly Tcl [146]. They work with their Custom Compiler suite for schematics and layouts. There is more information available at their website [147]. From Synopsys (Ciranova) it was also interesting the PyCell Studio tool [148] for universal OpenAccess PyCell development in Python, with their own layout Application Programming Interface (API). The idea was quite convenient since OpenAccess-based PCells can be imported on Cadence, Mentor and, of course, Synopsys tools; rather than using SKILL or other languages for PCell creation/operations. This way PCells were intended to be portable to the major vendors. The OpenAccess [149, 150] philosophy and technology together with the OpenPDK project are some of the proposals fostered by the Silicon Integration Initiative Si2 [151].
3) Mentor Graphics: The digital design alternatives provided by Mentor Graphics that grab our attention are Oasys-RTL for physical RTL synthesis, Olympus-SoC for P&R and Calibre InRoute for signoff. Also the Tanner Place and Route tool. More details can be seen at their website [152].

On the other hand, for FOSS EDA tools we can highlight the following contributions:

1) Qflow: The Qflow suite [153] is an open-source digital synthesis flow. Typically a set of tools like this one manages a subset of FOSS target-specific tools for each step of the design flow. This framework also handles/converts the I/O files of each tool so that the flow is consistent and can work without needing further conversions. Qflow is firstly composed by Yosis [154] for HDL-Verilog synthesis, logic synthesis and technology mapping using ABC [66]. It also integrates Graywolf [155] for placement, Qrouter [156] for routing, and Magic [157] as layout editor.
2) UMpack: The UMpack suite [158] is also an open-source set of tools for physical design. It includes Parquet for floorplanning, MLPart for partitioning, and Capo for placement. As a suite it also contains support libraries and additional files for interoperability.
3) Alliance: The Alliance suite [159] is a set of free tools that includes all major steps of a complete design flow. It performs HDL-VHDL synthesis, logic synthesis,

and placement and routing. It was the set of tools utilized in [130], which is a very interesting work for us, according to our scope.

4) Python-based: Python [160] is a highly versatile and powerful interpreted language. There are many libraries, packages, and applications available online. It is utilized by several EDA tools (e.g. Glade [71], KLayout [72]…) mainly for PCell developments, scripting, and layout/database APIs. But there are more Python-based EDA tools, not only for layout editors. Some of them are particularly interesting for us. For instance, for processing GDSII layouts I have used gdsCAD [161], fixing some of its non-solved issues, since it is no longer being maintained. For HDL-Verilog processing I have used Pyverilog [162], which also utilizes Icarus Verilog [64]. Pyverilog is explained in detail in [163]. Both packages have further dependencies that require installation. These two frameworks, gdsCAD and Pyverilog, were used by my P&R tool in order to handle the input files needed and to demonstrate our P&R ideas on the IGA designs. Since they are open-source it is possible to see the inner data structures, procedures, and features implemented within them; and add/modify them whenever convenient/necessary.

There are many other flows and tools for different purposes, whether with commercial or free licenses, such as Symica [164], XicTools [165], eSim [166], Astran [167], MyHDL [168], KiCad [169], LayoutEditor [170], Electric [171], and so on and so forth.

Other interesting works concerning Analog IC Design Automation can be found in [172-176]. They comment on the main differences and challenges between the digital and analog design automation domains, and provide insights about design flows, tools, physical design and PCells.

More specific details of P&R tools and related algorithms are explained in detail in the following subchapter.

2.4. Placement and Routing

2.4.1. Preliminary concepts

To better understand the P&R problem we will review henceforth some of the basics of computational science and algorithms, with special attention to their applicability on circuit design automation.

I must recommend the following reference books [177-182], since they are very convenient for the scope of this dissertation. Typically this kind of books have chapters where they introduce algorithmic complexity. To date, in the scientific mathematical/computational domain, a solvable problem, can be divided into P or NP problems. P problems are the ones that can be solved in Polynomial time by a deterministic Turing machine [183, 184]; while NP problems are those that can be solved by a Non-deterministic Turing machine also in Polynomial time; i.e. a problem is in NP if we can test whether a supposed solution is correct in a fast manner (polynomial time). Typically time complexities for classifying algorithms' speed are expressed using the Big O notation; a subset of Bachmann–Landau (asymptotic) notations [185, 186]. Some of those complexities are shown in Table 3, ordered by speed. Generally the time complexities are considered for their worst case scenario although sometimes a solution might be found in less time, what is designated as best and average times. A chart with their corresponding convenience related to the number of elements (n) being processed is shown in Figure 10.

Name (time)	**Big O notation**
Constant	O(1)
Logarithmic	O(log n)
Linear	O(n)
Log-linear	O(n log n)
Quadratic	O(n^2)
Polynomial	O(n^k)
Exponential	O(k^n)
Factorial	O(n!)

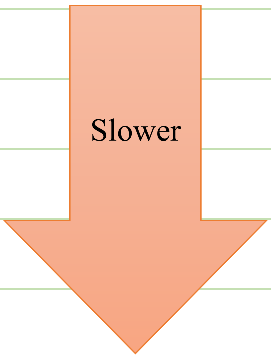


Table 3. Most common time complexities.

As explained in [187], by the sum rule for the big O notation, the sum of different procedures with their own complexities can be reduced to the worst case complexity of all the considered procedures. This is because at some threshold point in the growth of the number of elements (n), the worst complexity will dominate over the others, thus they can be neglected when scaling up.

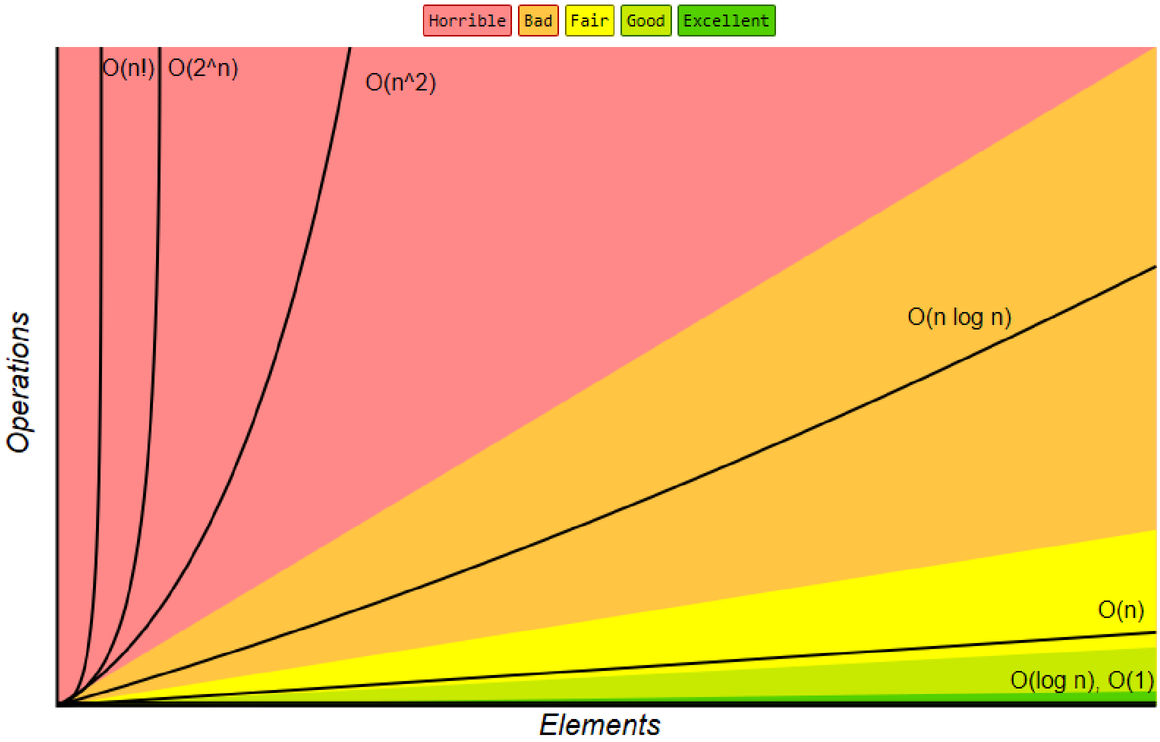


Figure 10. Time complexity convenience. [12]

The choice between which data structures to use for representing different kinds of information is highly correlated with the speed and efficiency of the preferred algorithms. It is also important to take into account the number of elements (n) to be processed, since the scalability of a particular data structure might affect the optimality of an algorithm, even to the point of making it unfeasible. For our purposes, just as an example, we can take a look at the data structures proposed in [188] for netlist handling. Common representations of graphs include adjacency matrices or adjacency lists. The first ones consist in matrices with each row and column representing a vertex. Edges connect between them with their corresponding weight. The latter ones consist on a master list with all the vertices of the graph pointing to all the rest of the vertices to which they are connected to. Unlike adjacency lists, adjacency matrices are not good for sparse graphs, when most of the cells of the matrix are empty (i.e. vertices not connected). Problems usually do not require such high degree of connectivity ($O(|V|^2)$ space).

Most programming languages have already built-in types (data structures) and procedures designed to optimize distinct operations; e.g. for appending elements on a list, getting and setting items, sorting, etc. They customarily allow to define proprietary data structures and procedures for specific purposes, or to import them from already existing libraries whenever convenient.

[12] Image source: bigocheatsheet.com.

Partitioning, Placement, Floorplanning, Pin assignment and Routing are all different separated NP-complete problems [178, 189-191]. NP-complete problems are those that are both NP and NP-hard. The latter ones are those considered to be at least as hard as the hardest problems in NP. In other words, for a problem A to be NP-complete, A has to be in NP, and every problem in NP has to be reducible to A. Figure 11 represents a drawing of the different sets. For an in-depth analysis of these concepts I suggest the reading of [192, 193].

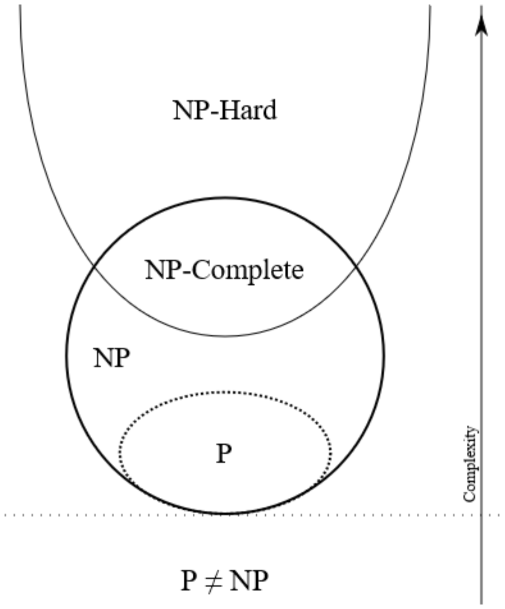


Figure 11. Euler diagram for P, NP, NP-complete and NP-hard set of problems. [13]

In practice, this means that it is not possible to find the unique best solution for any of our particular problems in a fast manner. It is common to search and apply heuristic algorithms to find the optimal or any good-enough solution that meet the requirements (cost functions), while trying to keep reasonable low time complexities.

We can classify algorithms in two ways: either by the results they produce, or by the way they work. When considering the results they obtain we can have the following two types:

1) Deterministic: these algorithms will always make the same decisions over any input, and therefore they will always produce the same output results for the same input data.
2) Stochastic: these algorithms make random decisions and, as a consequence, it is highly probable to have different output results for the same input data.

[13] Image source: Wikipedia.org.

On the other hand, if we want to classify algorithms considering how they work we can mention the following two types:

1) Constructive: they start selecting a component; then other components will be picked and added to the partial solution, according to specific criteria, until a complete solution is obtained. Once a component is chosen to be part of one partition it is never moved in future steps of the constructive procedure.
2) Iterative: they start receiving the input problem and an initial solution. They will try to improve the solution according to the targeted cost functions, and when no improvements can be done they will stop.

Usually, constructive algorithms are deterministic while iterative algorithms may be deterministic or stochastic. Generally, a constructive algorithm is normally used to find an initial solution of a problem, and afterwards iterative algorithms can try to optimize that outcome.

It is important to mention the greediness concept too. A greedy algorithm is a heuristic procedure that constructs a solution of a problem step by step, choosing the best local option at each decision making point. As advantages, they are typically simple to describe and efficient, since they can execute in a reasonable amount of time. However it is hard to find the right approach to solve a problem in this manner, and once found it is also hard to verify its correctness. Having chosen the best local solution at each stage of a greedy algorithm does not guarantee that the global optimal solution will be obtained either, although that is its aim.

2.4.2. Current SoA algorithms

The typical modus operandi of P&R algorithms can be divided in two stages: (1) to apply a constructive algorithm to build an initial feasible solution of the problem; and (2) to iterate over that initial solution optimizing a concrete (set of) objective(s) until it reaches an optimal/good-enough state or cost function(s).

Usual VLSI back-end structures (related to netlist's handling) rely on graph representation, where vertices model gates, and their associated well-known algorithms [182]. Some examples are: 1) Fleury's algorithm for Eulerian paths [194], 2) Search algorithms like Depth-First Search, Breadth-First Search or Topological Search; 3) Minimum Spanning Tree algorithms like Kruskal's [195], Prim's [196] or Borůvka's [197]; 4) Shortest Path algorithms for the problems of Single Pair Shortest Path (Dijkstra [198]) or All Pairs Shortest Paths; 5) Matching algorithms; 6) Min-cut / Max-cut algorithms; and 7) Steiner Tree [199] algorithms. Many of the mentioned problems and strategies can be applied to physical design, including Partitioning, Placement, Floorplanning, Pin assignment and Routing. More on graphs and algorithms can be read in [200-202].

There are also algorithms concerning front-end structures, for physical design (basically layout and geometrical objects). For instance, some examples of algorithms for detecting line intersections are: 1) Line Sweep Method; and 2) Extended Line Sweep Method [178].

Through the years a vast number of Place and Route algorithms were researched and developed for traditional Silicon-based microelectronics. Many heuristic algorithms have been proposed for all the purposes mentioned before, since they are NP-Complete problems. Modern EDA tools (free, open-source, academic, commercial…), and the functionalities embedded within them, cover mostly Standard-Cell VLSI placement for CMOS design style.

It is difficult to classify VLSI-related algorithms, since there are many different criteria to do so. Some of the established methodologies and algorithms for VLSI placement techniques are roughly based on Local Search, Recursive Partitioning, or Analytical Placement.

1) Local Search:

Simulated Annealing [203] is the most common algorithm based on Local Search in VLSI systems. Starting from a random partitioning, it is an iterative stochastic algorithm that randomly chooses local improved solutions from each partition to ultimately obtain a good global result (although probably not optimal). The method runs for a given time decreasing a parameter T, updating specified scores. It basically simulates the annealing fabrication processes of a metal, cooling a material with a decreasing Temperature T. The most representative SoA tools using this methodology are TimberWolf [204], and its known FOSS fork Graywolf [155], which is part of the Qflow distribution [153].

There are also other Local Search algorithms such as Simulated Evolution [205, 206] inspired by biological evolution, or Genetic Algorithms [207-209].

2) Recursive Partitioning:

Old algorithms in this area, but important to mention, are the ones from Kernighan-Lin [210] and Fiduccia-Mattheyses [211]. They are also known as group migration algorithms. More recent strategies use minimum cut algorithms which evolved from the previous ones, such as [212]. Essentially, Min-cut methodologies divide a netlist (represented by a graph), either in a bisection or quadrisection manner, to perform a top-down placement of cells. There are several variations that can apply randomization and other strategies for partitioning and balancing the subsets. In this case the most emblematic tool would be Capo [213, 214].

3) Analytical Placement:

This methodology allows to initially place overlapping cells. The cost function to target is the netlength minimization. A Linear net_length minimization strategy is too slow. Therefore the most common method is Quadratic Placement [215], which minimizes quadratic net_length in a faster 2-dimensional manner. As representative tools we can mention GORDIAN [216], Kraftwerk2 [217], and SimPL [218].

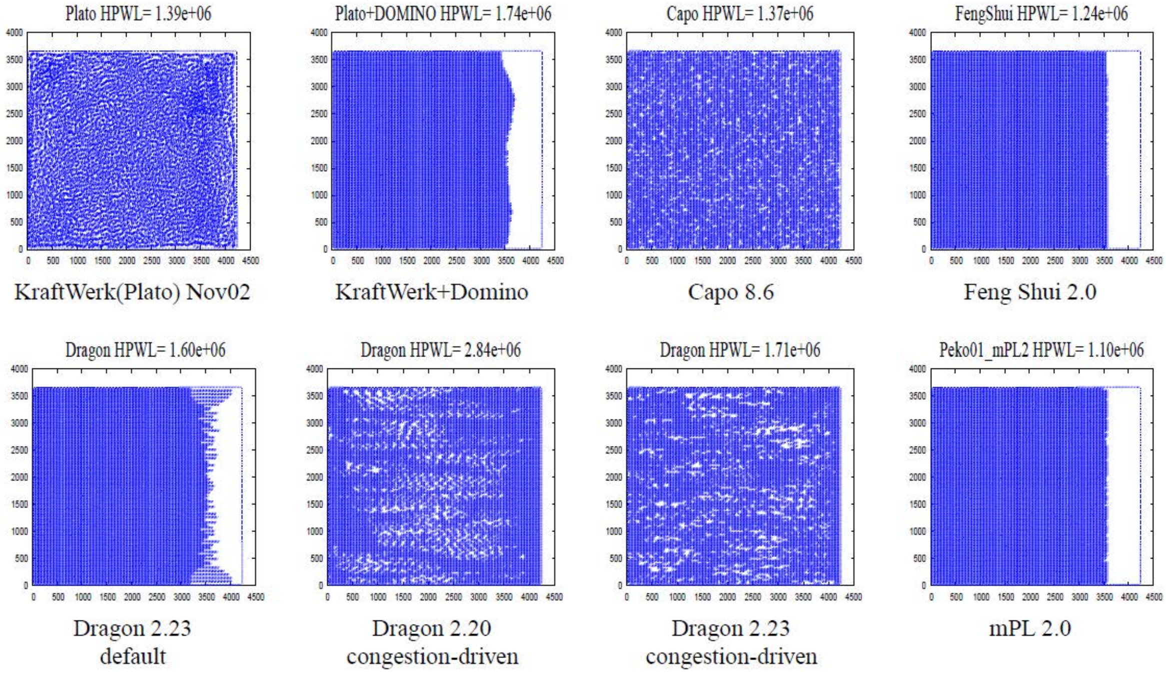
Choosing between those methodologies and their corresponding tools mostly depends on scalability and empirical results obtained from benchmarking comparisons, when available. Some might be more efficient for small circuits with a particular subset of constraints and cost functions, while others could work better for other kind of circuits and conditions. A comparison between them is done in [219]. Historically Simulated Annealing was more popular in the 80's, Min-cut in the 90's, and Quadratic methods until nowadays. There are also many heuristics that combine the mentioned techniques, potentially delivering better results depending on the circuits/applications targeted. Some benefits and drawbacks of each approach are shown in Table 4.

Placement technique	Benefits	Drawbacks
Simulated Annealing	Good performance when considering multiple goals. More flexible. Better when targeting small circuits. Can converge to near optimal solution if cooling time is slow enough.	Poor scalability for bigger circuits (VLSI). Running time can be very high. Lack of stability. Difficult to handle modules of different sizes.
Min-cut	More efficient (run time). Good scalability to VLSI. Good at mixed-size legalization.	Difficult to consider multiple goals. Lack of stability. Poor whitespace management.
Quadratic Placement	Fast and scalable (VLSI). Unique optimum solution, hence stable. Better quality for large scale designs. Plenty of information about relative positions. Good at whitespace management. Good when considering multiple goals.	Optimizes quadratic instead of linear netlength. Buffer insertion changes results. Difficult to legalize large macros. Hard to optimize macro orientations. Requires fixed pins.

Table 4. Comparison of Placement techniques.

More information of some of these techniques can also be found in [220]. For a very recent wider historical review I thoroughly recommend the reading of the 2015 IEEE special issue on EDA, particularly [221].

The concept of whitespaces should also be succinctly explained. Whitespaces, as in [222], are valid placement locations intentionally left empty by the placement tool, according to its mapping strategy. Tools handle whitespaces as they see fit. In traditional Silicon microelectronics whitespaces are used as a way to decrease density in regions with lower yield, or to avoid congestion-related issues. In Figure 12, extracted from [223], we can observe, just to illustrate the concept, how Capo tends to occupy the whole die, distributing whitespaces uniformly. This way routability seems easier. Other tools compact cells to the left to minimize wire lengths, leaving the right size of the die empty (with more or less uniformity).



KraftWerk(Plato) Nov02

KraftWerk+Domino

Capo 8.6

Feng Shui 2.0

Dragon 2.23
default

Dragon 2.20
congestion-driven

Dragon 2.23
congestion-driven

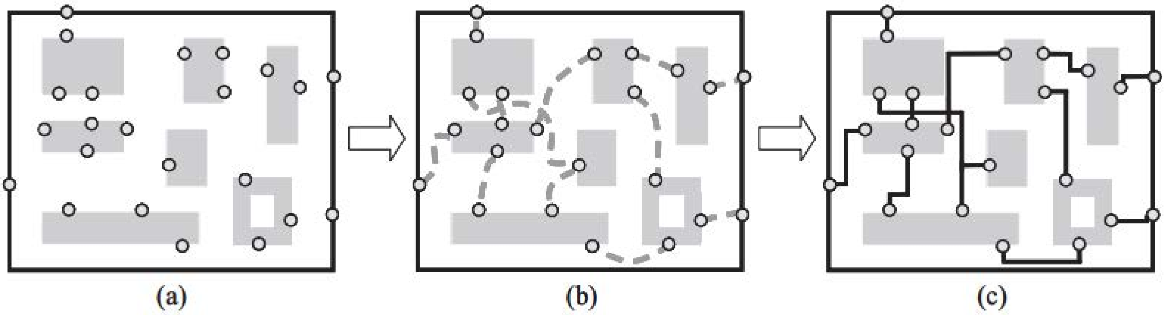
mPL 2.0

Figure 12. Comparison of whitespace management. [14]

Any given Placement methodology must ensure routability. Ideally Routing should not be less efficient nor require a longer execution time due to an unfavorable Placement process. Thus, although they are different problems, they are highly correlated. A P&R process can usually have one or more of the following design objectives: 1) meeting timing constraints, minimizing critical paths, cycle time and clock skew; 2) area minimization/compaction, avoiding possible congestion-related issues (e.g. power consumption/dissipation, parasitic effects, noise…).

[14] Image source: S. N. Adya. [223]

Traditionally, routing is divided into global routing where nets are assigned to different regions of the floorplan, according to the previous placement; and detailed routing where all nets of each particular region are fixed into their definitive positions. Figure 13 depicts these concepts. There are also special nets that require a tailored treatment, such as power, clock trees and critical nets that can affect performance substantially.



(a) A given placement result with fixed locations of blocks and pins.

(b) Global routing.

(c) Detailed routing.

Figure 13. Global and detailed routing. [15]

In the routing context there are many different techniques. Their classification is a difficult task too. We can roughly comment some of them such as Maze routing, Line search, Steiner Tree-based, or Integer Programming algorithms.

1) Maze routing:

The most famous algorithms are Lee's [224], Hadlock's [225] and Soukup's [226]. These are very old algorithms that have been evolving and adapting to different scenarios and purposes through the years. For instance, Lee's algorithm, although it guarantees to find a connection between two terminals and that it will be minimum, it requires plenty of memory storage for VLSI circuits and it is rather slow when the grid increases; disadvantages that have been optimized as mentioned. They are designed to work using two-terminal nets (i.e. a net that connects only two pins) over grids. Additionally they require that the nets have a pre-established order.

2) Line search:

The most common are Mikami-Tabuchi's [227] and Hightower's [228] algorithms. There are many improvements and variations done as well. They are designed to work

[15] Image source: Y. W. Chang. [33]. T.-C. Wang.

using two-terminal nets by using escape points over lines. Because they don't use grids their time complexity is theoretically lower (instead of O(M*N) for an M*N matrix, it would be O(L), where L is the number of line segments generated). However they cannot guarantee that a found path is of minimum length.

3) Steiner Trees:

There are many Steiner Tree algorithms [199], of which special interest fall upon Minimum Rectilinear Steiner Trees. They are for multi-terminal nets (i.e. a net that connects multiple pins). Note that the order on the words can change and therefore its acronym, in particular do not confuse with a Rectilinear Minimum Spanning Tree. A Minimum Rectilinear Steiner Tree is a Rectilinear Minimum Spanning Tree with its wire length already optimized. It usually has shorter wire length, but less routing flexibility and more difficult data structures; whereas a Spanning Tree provides more flexibility at the cost of a longer wire length. We can highlight the contributions from Hanan [229] and Hwang [230], although there are many more algorithms dealing with these kinds of trees for routing applications.

4) Integer Programming:

The above strategies follow a sequential approach, where nets are routed one at a time; but it is also possible to use a concurrent scheme such as a hierarchical integer (linear) programming methodology [231-233]. In this case each net present a set of possible Steiner Trees, from which one has to be chosen in parallel for each net, minimizing the total wire length.

In the case of detailed routing there are many methodologies as well. We can distinguish two main types: channel routing or switchbox routing.

1) Channel routing:

First let us review some basic terms for channel-based routing. A horizontal segment of a net is called. A trunk can be placed occupying a portion or an entire (horizontal) track in a grid. A vertical segment of a net is called branch and it can be placed occupying a portion or an entire (vertical) column in a grid. Branches connect a trunk to its corresponding terminals of cells in the upper and/or lower boundaries of a channel. The local channel density is given by the number of trunks that cross each column. Hence, the channel density (i.e. net capacity) would be the maximum local density. Obviously, the minimum number of tracks of a channel would be the channel density; but a higher number of tracks is recommended to ensure routability, and also for special nets. Figure 14 depicts these concepts.

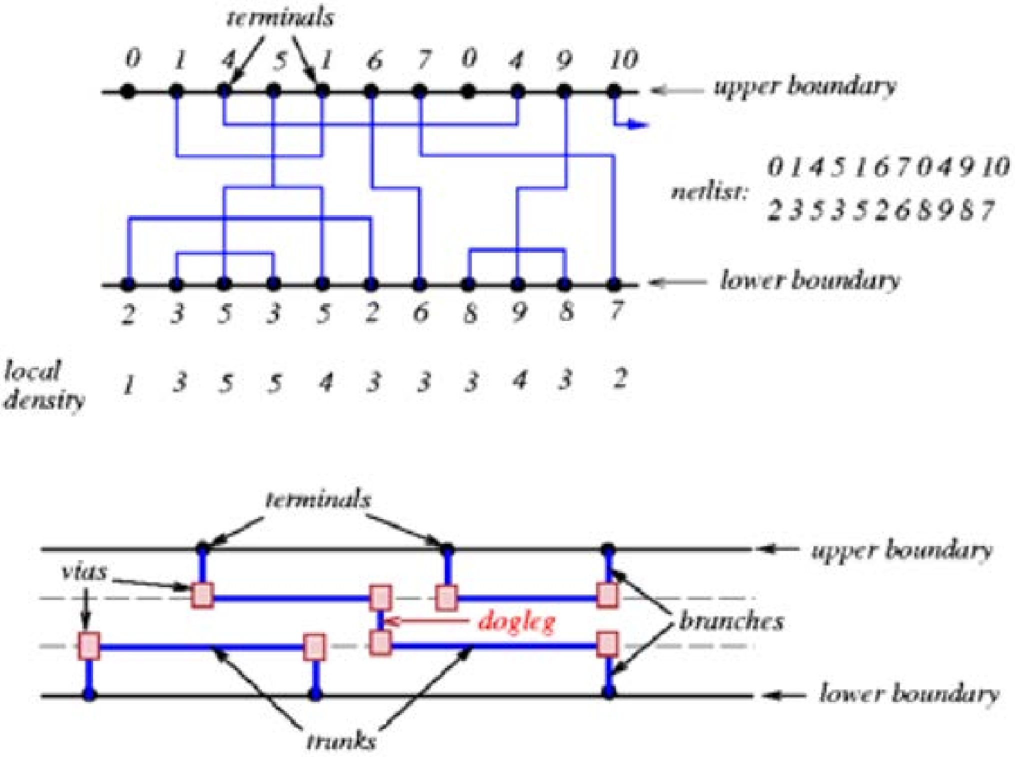


Figure 14. Channel routing terminology. [16]

We can distinguish two kinds of graph representations: 1) Horizontal Constraint Graphs (HCG), and Vertical Constraint Graphs (VCG) [234]. HCGs are undirected graphs that represent horizontal restrictions between nets, meaning that those nets cannot be placed in the same track to avoid their overlap; whereas VCGs are directed graphs that represent vertical restrictions between nets, where terminals in the upper boundary have a transition to the lower boundary only when their terminals are different (for one column-different upper and lower terminals).

In the Standard Cell design methodology the main goal of channel routing is to minimize the channel height, which will reduce the overall area. If a routing process does not succeed at fitting the nets into a channel then it will have to expand its height. For this objective it is important to point out the Dogleg concept [235]. Doglegs are a mechanism that allows placing trunks from a net over different tracks. This provides routing flexibility for further minimization of the channel height, with the drawback of potentially needing more vias. In the following subchapter we will further comment on two-layer routing algorithms, as they are closer to our IGA point of view.

[16] Image source: Y. W. Chang.

2) Switchbox routing:

The switchbox methodology is similar to the channel routing, but in this case it is allowed to have cells, and consequently terminals, in every side of the channel, not just on the upper and lower boundaries. An L-shape channel can be decomposed into two channels plus a switchbox (that corresponds with the corner box).

Some algorithms may include a rip-up and re-route step [236, 237] that is triggered whenever a routing process fails, for instance because of the presence of blockages that impede it.

Similarly to the Placement problem, many Routing heuristics arose trying to mix well-known strategies in order to obtain better performances depending on the design styles and targeted circuits. In the end, this is a trade-off.

It is important to highlight the value of a priori wire length estimations, as in [238], based on Rent's rule on interconnection complexities and later works [239-242]. Rent's empirical rule is as follows:

$$P \;= KB^r$$

It expresses the relationship between the number of I/O pins *P*, and the number of terminals per cell *K* multiplied by the number of cells *B* to the *r* power. *K* and *r* are constants. This Rent's *r* exponent must be between values 0 and 1 (typically $0.5 < r < 0.8$).

In addition to the works mentioned earlier, there are many others that emphasize the importance of wiring estimations, such as [243-248], since the circuit performance is depending on them.

2.4.3. P&R SoA related to our IGA case

When targeting PE technologies there are few foundries offering different fabrication processes (not to say none) publicly. Technologies are often accessed through R&D projects or specific collaborations. For most of those PE processes, there is only one or two metal layers available for connection, since most technologies are not evolved enough to allow more, through evaporation processes or others such as gravure or inkjet that rely in complex fluidic behavior (i.e. silver nanoparticle inks). Meanwhile in silicon VLSI it is common to have several layers of metal connectivity (from 2 to 10). Our approach follows a Gate Array design strategy, using PMOS-only design styles (mainly due to the PE conductivity reasons commented in [67]), but adaptable to CMOS.

Even though our design efforts are mostly focused on Gate Array it is necessary to consider the main characteristics and differences between Standard Cell and Gate Array placement and routing techniques, which are well explained in [249]. Basically, in a fixed-die Standard Cell design style, the cells are placed in rows of fixed height, determined by the maximum height of a cell. These rows are ideally of the same width with the intention of minimizing the area spent. The routing channels can vary in height, which would change the net capacity of each channel. The cell library might also contain feedthrough cells to allow the connection between rows of cells that are not adjacent. In this case a router will try to minimize channel height to reduce the overall circuit area. In addition over-the-cell routing [250, 251] is quite advantageous since routing channels can be largely reduced. On the other hand, in a Gate Array design style, the position of the cells and routing channels are already fixed. In this case the main objective of a routing process would be to ensure that the design is routable. Other goals might include to minimize maximum and/or total wire length. An example of a much less common objective would be to minimize the number of vias (typically bends) needed, since they increase the cost of a chip by adding more metal connections into the stack of layers, with their associated yield and congestion-related issues. Notice that a channelless GA that allows routing over its cells, thanks to the VLSI stack of layers, is called Sea-of-Gates [190, 252].

Gate Array literature is quite extensive and varied. However, most of the works that best match our intentions are old, since Standard-Cell designs dominate the current VLSI-ULSI industry. References that are more interesting for our GA design choice are [253, 254]. Several works demonstrated how convenient it is in certain situations to treat different VLSI problems conjointly [255-259]. Actually, when VLSI problems are processed simultaneously they might profit from such perspective, mainly to avoid iterations, but other factors can be considered too.

Programmable Logic Devices (PLD) and FPGAs [260, 261] are much more complex than the IGAs. Only One-Time Programmable (OTP) antifuse-based FPGAs [262] are conceptually closer to the IGAs. But their pre-built Logic Blocks (Configurable Logic Blocks CLBs) still make their architecture more intricate than our transistor-level blocks, or as we denominated them: Basic Bulk Cells (BBCs). FPGA elementary structures usually handle logic plus memory cells (e.g. Lookup Tables LUTs, D Flip Flops DFFs…), complex wiring and configuration memories. It is certainly interesting that reconfiguration based on faulty logic resources have also been proposed in this area [263, 264].

2.4.4. Selected P&R strategies

The placement technique selected for our case will start at the bottom left corner of an already designed IGA substrate, and follow a row-based growth. This is similar to a cluster growth methodology [265], but adapted to our BBC topologies.

For routing there are single-layer and multi-layer heuristics, depending on the number of metal layers allowed for connectivity. In single-layer designs the routing has to be a planar graph. In multi-layer designs there can be layers that cross each other, so there is no need for planarization when vias are permitted. In VLSI, the crossings are done at different stack levels, and connections are done by using vias whenever necessary. Typically in a two-layer approach one metal will be used only for horizontal trunks and the other for vertical branches.

Our IGA case would fall within a 2+1 layer GA channel routing scheme. This means that it is similar to a two-layer methodology but adds another metallic connection for its intrinsic personalization step. The intention remains the same: connecting horizontal with vertical tracks, regardless of whether the technology allows the implementation of vias or not. This is an important situation to consider, since some PE technologies might not allow the use of vias. It might be because of fabrication process limitations or maybe just convenience, as some via sizes are so big that their inclusion into the routing channels would increase the area required beyond measure, due to design rules.

Left-Edge Algorithms (LEA) are for two-layer channel routing based on VCG. The basic LEA [266] is the routing algorithm that is closest to fulfill our IGA routing requirements. In some designs targeting PE technologies once a trunk of a net is placed the entire track will be occupied by that net value, whether it is a local track (for only one BBC) or a global one (for a row of BBCs). This means that the basic LEA is not applicable to our problem, because Left-Edge tries to use the same wire track when there is no overlap between horizontal trunks. Hence, by extension, versions with doglegs are not allowed either.

My P&R methodology for the IGA designs is fully explained and detailed in Chapter 4, while the design concepts from which it relies are shown in Chapter 3.

2.5. Summary of the chapter

In Chapter 2 we have reviewed the state of the art in PE/OE with special attention to circuits and systems applications. We have reviewed the current EDA tool landscape, taking into account our IGA approach/strategy. We reviewed the most important P&R algorithms and the most interesting/resembling ones to our specific purposes, discovering that there were no similar strategies publicly available.

3. Targeting PE Technologies

This chapter is devoted to elaborate the information related to the fabrication processes from the PE foundries that is required for circuit design using cell libraries. This basically includes: (1) layer information that accounts for the available resources for devices and interconnections; (2) design rules, concerning area constrains and (3) simulation models to account for device performance

On top of that information, designers will built cell libraries (either standard cells or gate arrays) that have to be formatted according the EDA tool requirements. Selected process technologies will be: (1) a full-inkjet process from TUC/UAB; (2) An evaporation photolithographic process from CPI/Neudrive and (3) a combined Printed Electronics process from CEA-Liten.

3.1. Elementary technology information

Designers require the minimum amount of process technology information in order to design electronic devices and circuits. That information is usually delivered in paper and in electronic format, as technology design kits that can be used by the EDA tools to allow the full-custom design methodology and to build cells and structures for the semi-custom design methodology used in this dissertation. Following, we summarize the main process technology characteristics for the three fabrication process considered in this research.

During the last years our research group, got access to different PE technologies, thanks to common projects and partnerships. Even, our team at UAB set up its own printing facilities and produced its own OTFT process, taking advantage of the relative low cost inkjet printer. Our own technology was developed in cooperation with TUC and ENEA, under the framework of the TDK4PE European project. More details can be seen in [267]. Currently, this equipment has been transferred to the IMB-CNM (CSIC) facilities, to join other equipment on a new printing laboratory for evolving this research line.

Printed microelectronics technologies make more progresses when the industry cooperates with academic and publicly funded institutions for improving the whole value chain from applications to products, including design. Joining forces is essential for growing business. In this applied research field, many results are obtained by experimentation through trial and error iterations. Multidisciplinary teams with complementary areas of expertise need to interact effectively to successfully propose new ideas and solve issues arising at any stage of the chain. This way, we reduce the existing gap between application and design teams and PE technologies and materials providers.

In addition to the UAB/CNM's PE facilities we also collaborated in a fabless manner with CPI/Neudrive and the CEA-Liten. We have provided them design kits based on their technological characteristics as well as a variety of prototype designs based on full-custom, Standard-Cell and IGA styles. Many iterations were done over test structures and devices (as in [268]) to improve performance according to their device and design rules.

3.1.1. Full-inkjet technology by TUC/UAB

Details of the technology are available in [90, 141, 267, 269]. In those references we can observe the evolution of the performance characteristics. The OTFT yield is up to 78%. Basic facts for this technology are:

- Bottom Gate Bottom Contact (BGBC) configuration.
- PMOS-only. 4 material layers. 2 of them are metal layers. Allows one more metal layer for interconnections/customization. Feature size: L_{min} = 40 µm. Typically L = 40 µm, W_T = 10 mm, W_T/L = 250.
- Performance: Carrier mobility (μ_p) up to $1.9 \cdot 10^{-4}$ cm^2/V·s; $V_T \approx -0.9$ V; V_{GS}, V_{DS} ≈ +10..-30 V; I_{On}/I_{Off} ratio up to 150; and Drain-Source current of 1.16 µA.
- Figure 15 shows an example of OTFT layout.

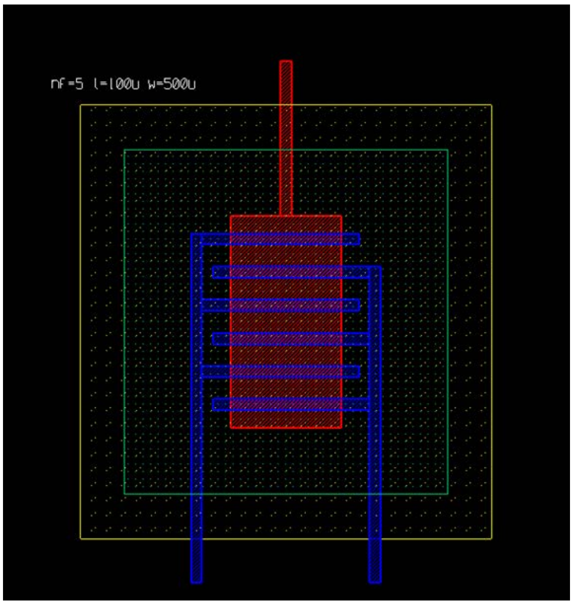


Figure 15. OTFT layout (TUC/UAB).

- Figure 16 shows an example of fabricated OTFT.

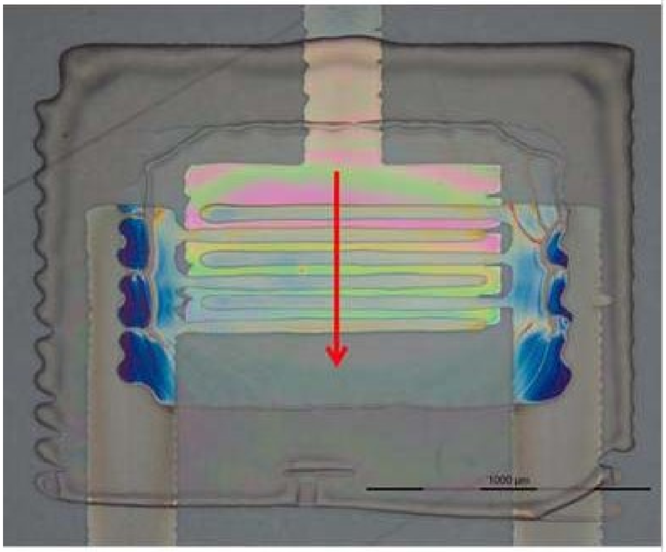


Figure 16. OTFT (TUC/UAB). [17]

3.1.2. Photolithographic technology by CPI/Neudrive

The CPI/Neudrive is a PMOS-only OTFT technology closer to a TFT-display or silicon-based clean room because the designs are implemented by using masks in photolithographic steps [59]. Therefore, the design rules for this technology are much smaller, their OTFTs seem to have better performance and its yield is nearly 100%. Further details of their technology are available in [82, 270, 271]. Basic facts for this technology are:

- Top Gate Bottom Contact (TGBC) configuration.
- PMOS-only. 6 material layers. 3 of them are metal layers. Allows one more metal layer for interconnections/customization.
- Feature size: $L_{min} = 4$ µm. Typically L = 4 µm, $W_T = 360$ µm, $W_T/L = 90$.
- Performance: Carrier mobility (μ_p) 3 cm^2/V·s; $V_T \approx 10$ V, V_{GS}, $V_{DS} \approx +15..-30$ V; $I_{On}/I_{Off} \approx 10^6$, and Drain-Source current of 320 µA.
- Figure 17 shows an example of OTFT layout.

[17] Image source: E. Ramon.

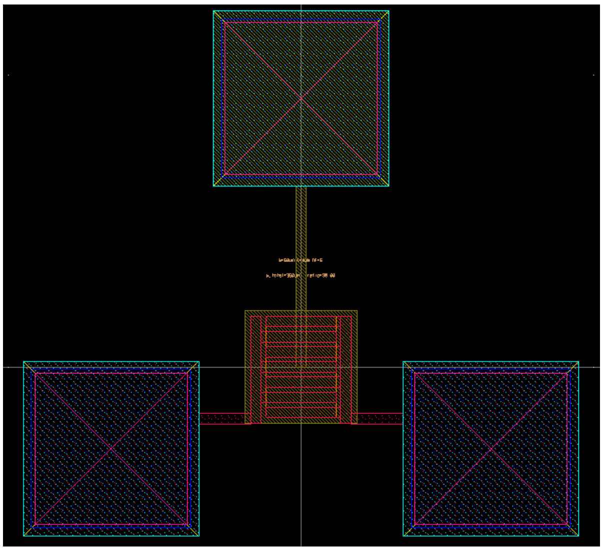


Figure 17. OTFT layout (CPI/Neudrive).

- Figure 18 shows an example of fabricated OTFT.

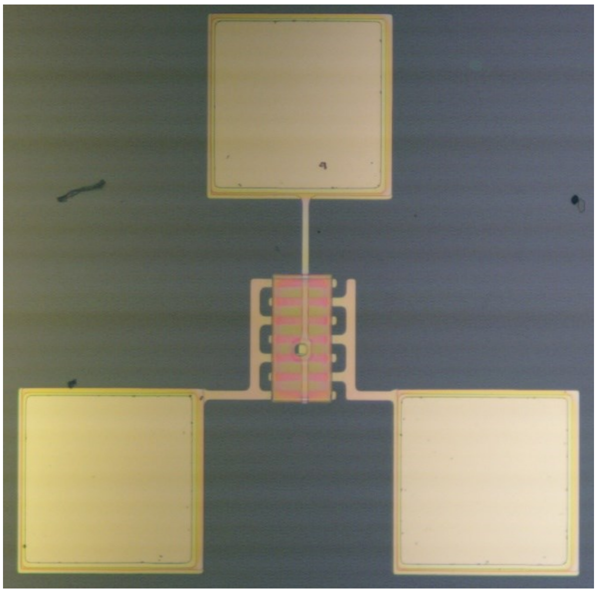


Figure 18. OTFT (CPI/Neudrive). [18]

3.1.3. Printed Electronics technology by CEA-Liten

The CEA-Liten technology (PICTIC platform) uses a combination of spin-coating, sputtering, photolithography or laser ablation, and screen printing for their different layers. Details on their technology are available in [124, 272, 273]. In this case,

[18] Image source: M. Mashayekhi.

there is the possibility to select a CMOS design style (with an OTFT yield higher than 98%), although less mature than the PMOS-only design style which requires less area. Basic facts for this technology are:

- Top Gate Bottom Contact (TGBC) configuration.
- PMOS-only. 5 material layers. 2 of them are metal layers. Allows one more metal layer for interconnections/customization.
- CMOS. 5 material layers (no extra buffer layer as in PMOS-only). 2 of them are metal layers. Allows one more metal layer for interconnections/customization.
- Feature size: L_{min} = 20 μm. Typically in PMOS-only L = 20 μm, W_T = 500 μm, W/L = 25; while in CMOS L = 100 μm, W_T = 2 mm, W/L = 20.
- Performance PMOS-only: Carrier mobility (μ_p) 0.8 $cm^2/V{\cdot}s$; $V_T \approx 5$ V, V_{GS}, V_{DS} ≈ +10..-20 V; $I_{On}/I_{Off} \approx 5{\cdot}10^7$, and Drain-Source current of 60 μA.
- Performance PMOS on CMOS: Carrier mobility (μ_p) 1.5 $cm^2/V{\cdot}s$; $V_T \approx -0.2$ V, V_{GS}, V_{DS} ≈ +10..-20 V; $I_{On}/I_{Off} \approx 10^7$, and Drain-Source current of 2 μA.
- Performance NMOS on CMOS: Carrier mobility (μ_n) 0.55 $cm^2/V{\cdot}s$; $V_T \approx 1$ V, V_{GS}, V_{DS} ≈ +10..-20 V; $I_{On}/I_{Off} \approx 2{\cdot}10^7$, and Drain-Source current of 1 μA.
- Figure 19 shows an example of OTFT layout.

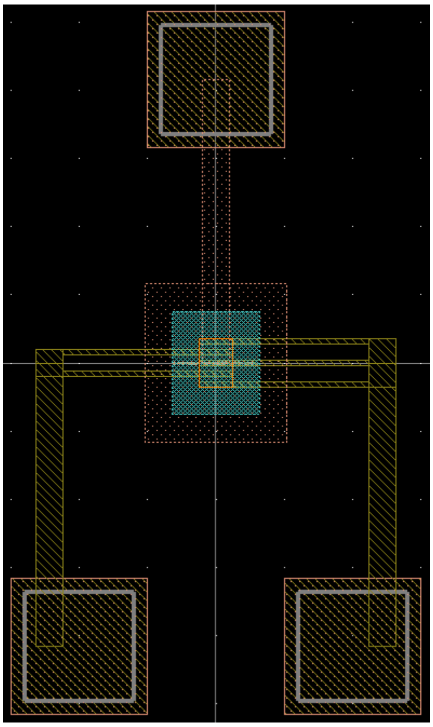


Figure 19. OTFT layout (CEA-Liten).

- Figure 20 shows an example of fabricated OTFT.

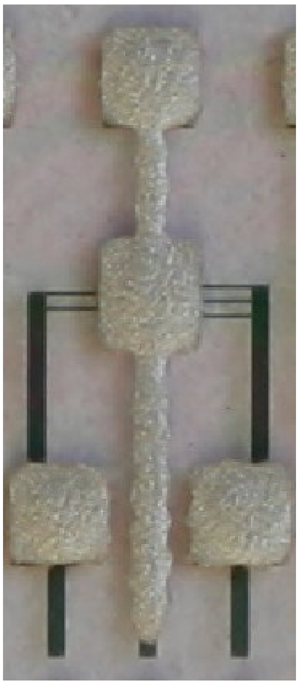


Figure 20. OTFT (CEA-Liten).

3.2. PMOS & CMOS Cell Design styles

The election on the cell design style is strongly correlated with the target technology and its performance characteristics.

Figure 21 shows the most common logic cell design styles. From left to right we have a) Pseudo-NMOS, then b) CMOS, and c) Pseudo-PMOS. This last case also depicts three different load configurations.

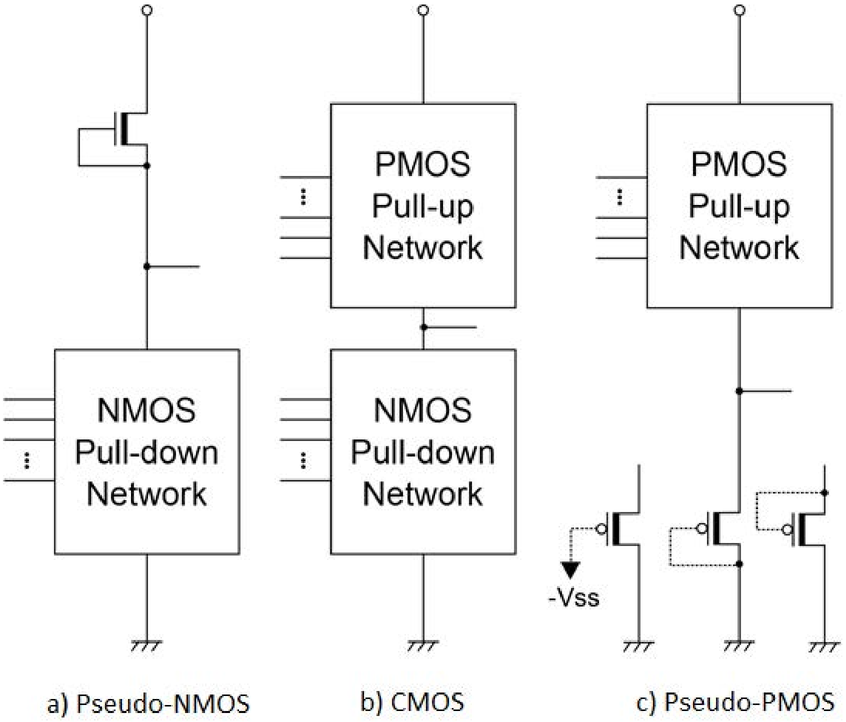


Figure 21. Logic design styles.

Logic gates following the pseudo-PMOS style have only one pull-down PMOS transistor acting as a discharge load. As shown in Figure 21 c), this single transistor can use the following configurations: (1) Zero-V_{GS}, when its gate is connected to the output; (2) diode-connected, when the gate is grounded; and (3) biased to -V_{SS}. The pull-up network is composed of a set of series and parallel switching transistors that implement the actual logic function of the gate. The Zero-V_{GS} style is the best choice for PMOS-only technologies with positive V_T, as explained in [82].

Due to the current state of most PE technologies a pseudo-PMOS (ratioed PMOS-only) design style is preferred over today's most common CMOS designs in silicon, because of the usual threshold voltage values. The design style choice has a strong impact in the number of transistors required to implement any specific functionality. For a CMOS style the number of transistors required to implement a gate is 2N, where N is its fan-in (i.e. number of inputs). Whereas in a PMOS-only style the number of transistors is reduced to N+1, being the extra one the load transistor. This is an advantage because it saves area in technologies with low integration density. This comes at the cost of lower noise margins (voltage range between logic 1 and logic 0 values), asymmetric rise and fall propagation delays (and therefore lower speed that CMOS equivalent circuits) and static power consumption (when the output has value 0 so that both pull-up and pull-down networks are active).

The aspect ratio between Drive and Load OTFT sizes, for our selected technologies, are discussed in detail in [82]. The basic principle is to obtain the maximum carrier mobility, conductivity, noise margin, and consistent logic values at the output. The choice depends on several variables such as the OTFT's channel dimensions, inks and materials, thickness of each layer (organic semiconductor, inorganic metals, dielectric insulator…), etc. At design level the OTFT layouts can be adjusted mainly according to their channel length, width, and number of fingers (in an interdigitated OTFT style). This way we obtain different performance characteristics. Trial and error iterative processes are essential to determine the optimal configurations due to the high variability of the fabrication processes that are not stable enough (compared to the silicon ones). Typical aspect ratio values are 1/3, 1/5 and 1/7.

Figure 22 shows schematic examples of a few pseudo-PMOS cells, including an inverter, a 2 input NAND, an XOR and a DFF, in this case for a V_{SS} controlled gate.

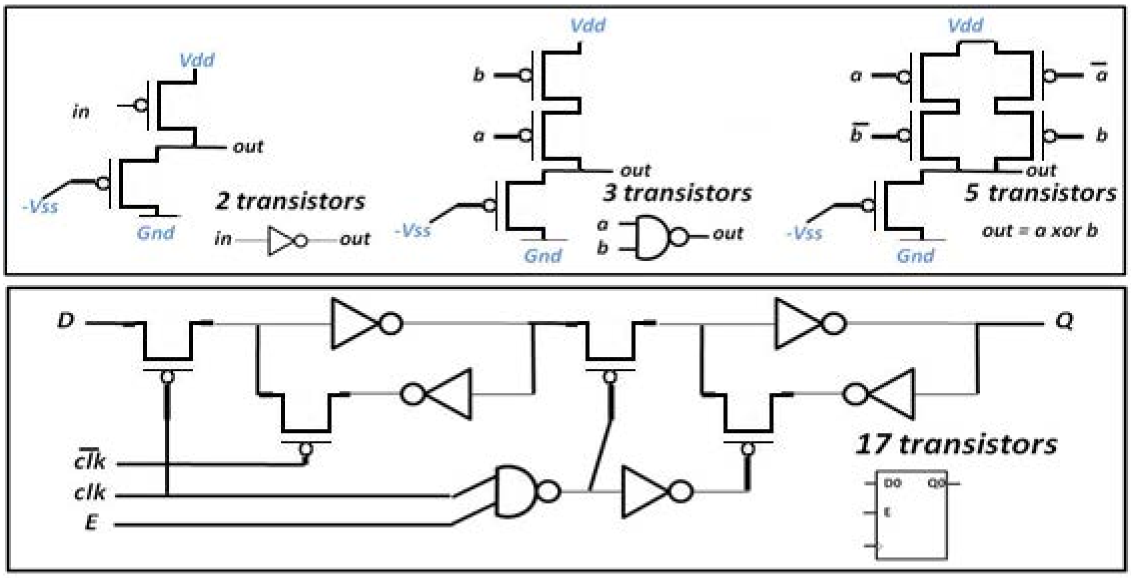


Figure 22. Examples of PMOS-only cells.

3.3. PMOS & CMOS Cell Libraries

The following section presents some of the cell libraries our group has developed for the mentioned technologies [274, 275], which include passive and active devices, as well as basic logic gates, among others. Some of the cells were used to demonstrate our ASPEC design flow, as explained in [58, 59].

3.3.1. TUC/UAB

Table 5 shows the developed PMOS cell library.

Cell library	Main parameters
Linear resistor	Length, Width
Linear resistor	Resistance, Width
Snake resistor	Length, Width, Meanders
Snake resistor	Resistance, Width, Meanders
Capacitor	Length, Width
Capacitor	Capacitance, Width
Square inductor	Radius, Width, Turns
Octagonal inductor	Radius, Width, Turns
Schottky diode	Length, Width
POTFT	Length, Width, Fingers
NAND2	Ratio
NOR2	Ratio

Table 5. Cell library (TUC/UAB).

3.3.2. CPI/Neudrive

Table 6 shows the developed PMOS cell library.

Cell library	Main parameters
POTFT (interdigitated)	Length, Width, Fingers
POTFT (corbino)	Length, Width
Inverter (i and c)	Ratio
NAND2 (i and c)	Ratio
NAND3 (i and c)	Ratio
NOR2 (i and c)	Ratio
NOR3 (i and c)	Ratio
Fill (i and c)	Width
Feed (i and c)	Columns

Table 6. Cell library (CPI/Neudrive).

Figure 23 depicts an inverter (1:5 ratio).

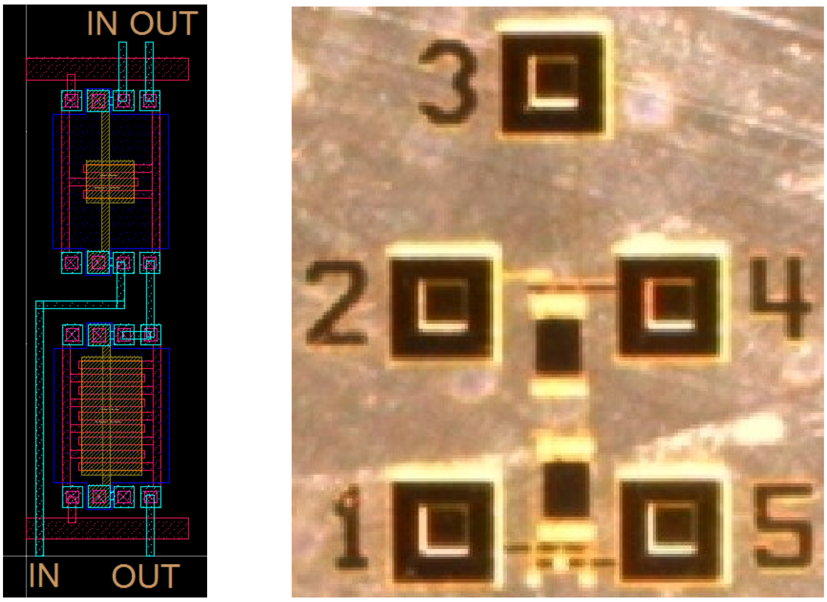


Figure 23. Inverter (CPI/Neudrive).[19]

3.3.3. CEA-Liten

Table 7 shows the developed PMOS-only and CMOS cell libraries.

Cell library	Main parameters
POTFT (PMOS-only)	Length, Width, Fingers
POTFT (CMOS)	Length, Width, Fingers
NOTFT (CMOS)	Length, Width, Fingers
Inverter (PMOS-only)	Ratio
Inverter (CMOS)	Ratio
NAND2 (PMOS-only)	Ratio
NAND3 (PMOS-only)	Ratio

Table 7. Cell library (CEA-Liten).

Figure 24 depicts an inverter for PMOS-only (1:3 ratio).

[19] Layout and image source: M. Mashayekhi.

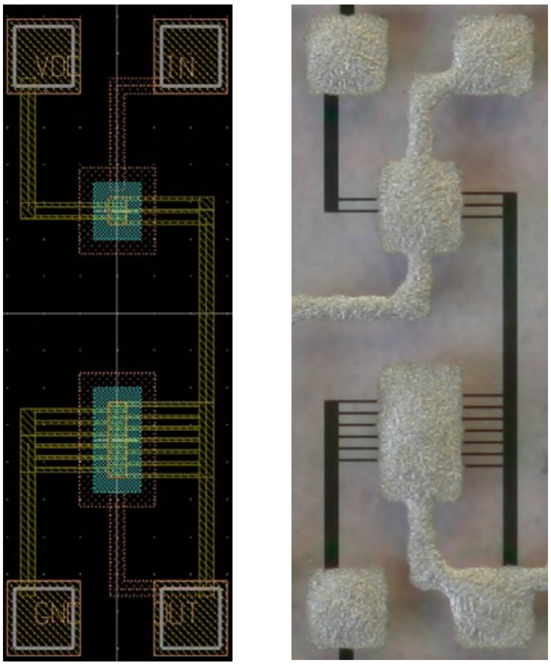


Figure 24. Inverter (CEA-Liten).

3.4. PMOS Inkjet-configurable Gate Array

3.4.1. 2-step Fabrication process

One of the key characteristics of an IGA prototype is that its fabrication is done in two steps separated in time: 1) fabricating the IGA template, including its arrays of transistors, I/O pads and unconnected wire tracks; and 2) personalizing it, which means mapping a functionality/circuit over them, using any of the digital printing techniques that best befit the IGA at hand.

Some printing technologies for the IGA metallization (i.e. interconnections) have been studied and reported in [80]. These include inkjet, aerosol jet, superfine inkjet, and photolithographic techniques. The conclusions of this work are as expected, meaning that photolithographic procedures present the highest accuracy, less process variability, and thus highest interconnection yield (97%). Whereas aerosol jet and superfine inkjet present yields of 75% and 88% respectively. As usual, the larger the design rules and the thicker the deposited layers the safer the designs are against any possible failure, at the cost of area.

The wire cells would be composed by global tracks (power, clock or any other critical signals) and local tracks which are unconnected by default. The opposite approach would be to have all wire tracks connected by default, and then laser-cut the tracks that will not be used.

Next we can observe the IGA template layout examples for the three technologies. More details can be found in [81, 82].

3.4.2. TUC/UAB

The main characteristics of this implementation are: 4 Load + 7 Drive OTFTs per BBC, 7 local and 3 global tracks per column, 160 OTFTs, 0 I/O Pads (direct connections to tracks). Figure 25 depicts this IGA.

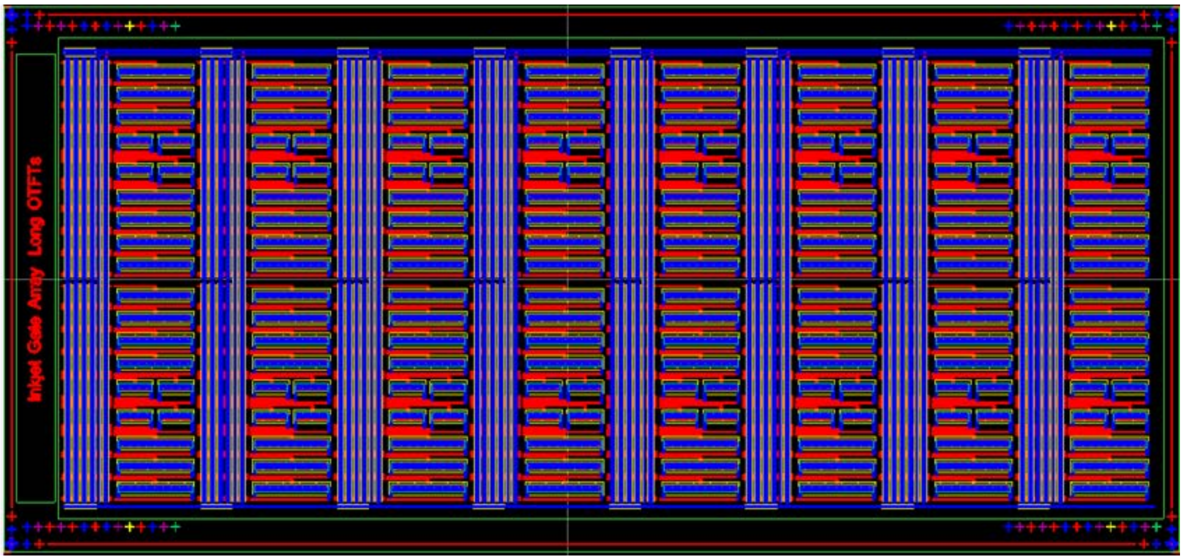


Figure 25. IGA (TUC/UAB). [20]

3.4.3. CPI/Neudrive

The main characteristics of this implementation are: 3 Load + 6 Drive OTFTs per BBC, 6 local and 7 global tracks per row, 54 OTFTs, 52 I/O Pads. The IGA is shown in Figure 26. We have used this IGA as a proof of concept for the demonstration of our P&R algorithm, because the corresponding fabrication process of this foundry is one of the most stable ones that we were able to access (since it is mask-based). It features a high integration density and an almost perfect yield. During this R&D period the designs were improved according to the results of each fabrication run. Overall, due to these considerations, this technology was the best choice in order to eventually demonstrate functional circuitry, along with the scalability of the proposed solution.

[20] Layout source: M. Mashayekhi.

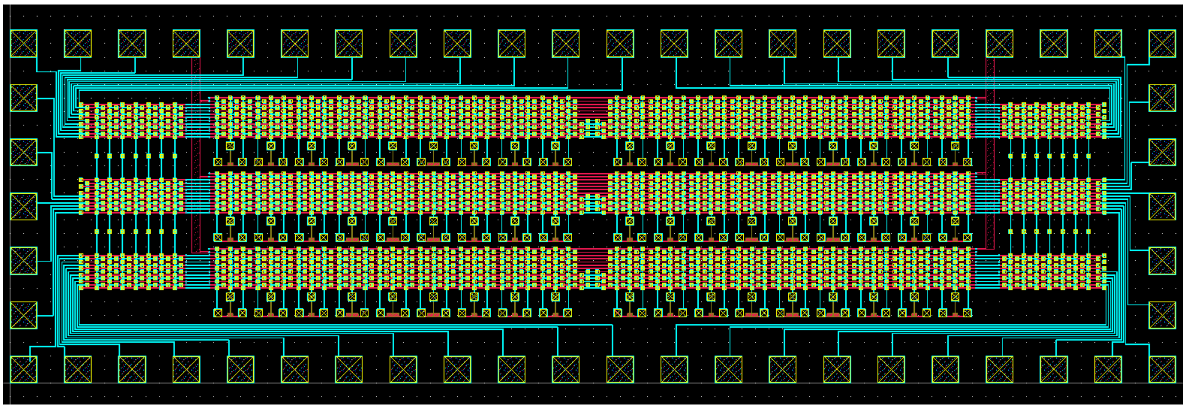


Figure 26. IGA (CPI/Neudrive). [20]

3.4.4. CEA-Liten

The main characteristics of this implementation are: 3 Load + 6 Drive OTFTs per BBC, 12 local and 2 global tracks per row, 54 OTFTs, 8 I/O Pads. The IGA can be seen in Figure 27. There are several differences between this IGA and the CPI/Neudrive one, which affect P&R heuristic decisions. The most important one is that there are no vias in the wire cells. The connectivity zones are gaps between metals in the two dimensional plane, as opposed to the superposed layers of traditional vias. The horizontal tracks of each wire cell are discontinuous, while vertical tracks are still continuous. Because of this, the OTFTs of the upper versus lower rows cannot be mapped simultaneously to represent different logic, as this will incur in shorts between nets. They are placed to provide redundancy in case of transistor fault. Additionally, the I/O pins are not preassigned to specific tracks so there is more flexibility for the pin assignment problem.

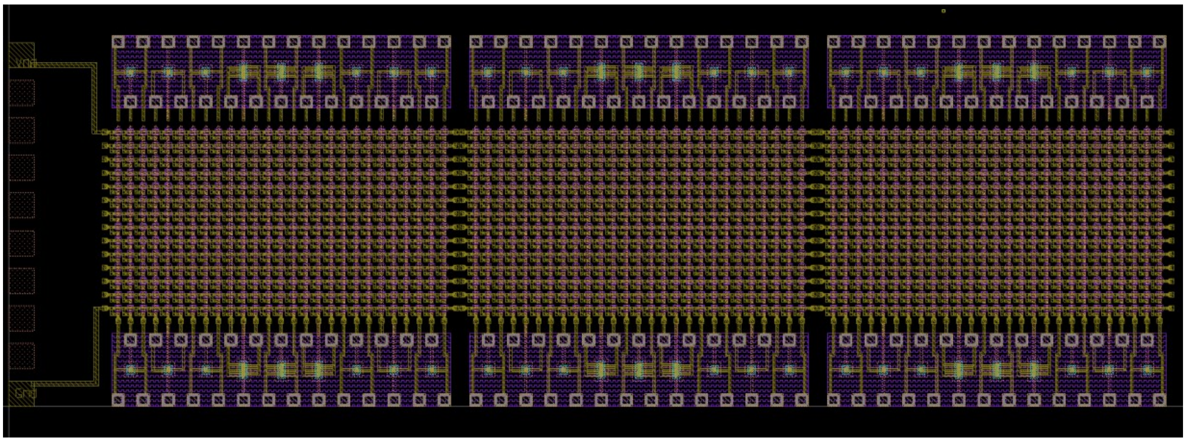


Figure 27. IGA (CEA-Liten).

3.5. Summary of the chapter

In this chapter we have addressed the printed electronics technologies selected and their main characteristics for full-custom and semi-custom design methodologies. We have reviewed PMOS and CMOS design styles and cell libraries. We have shown several examples targeting the technologies accessed. Those examples include the IGA templates intended for later customization. That customization stage will be the main target of our P&R algorithms. This is mainly due to the fact that using digital printing allows individual foil personalization. Therefore, running the P&R for mapping the same functionality to every circuit will produce different output results either for parameter identification or for yield improvement by avoiding faulty transistors.

4. P&R Algorithms for Digital^2 PE Circuits

4.1. Managing circuit personalization files

A key element of our strategy is to obtain OTFT failure (distribution) maps, or their conceptually opposite, KGO maps, based on the results of an intermediate characterization step. This characterization is necessary whenever a target technology has an expected yield which is not close to 100%. Such KGO map will act as a constraint to the P&R process.

The characterization process marks the OTFTs that present good enough characteristics, according to the desired thresholds, as usable for building circuits. While these conditions would also depend on device performance, with respect to our P&R strategy, this will have to be transformed to a binary option, either usable or unusable.

The input for the characterization process is a Test Vehicle Description (TVD) file, as described in [82, 268]. This file is just a simple Comma-Separated Values (CSV) file, which include the X-Y positions of each OTFT (on their bottom-left corner) of the IGA substrate. It can contain ID labels, and any other information that might be relevant at transistor level.

A KGO map file has a similar appearance to a TVD file, having the same CSV format, although it contains different data. In this case, a KGO map will include the positions of the characterized OTFTs, ID labels to identify the BBC where they belong and their position within them, the type of the transistor (Drive or Load), and the binary usability flag. Table 8 shows an example.

X	Y	ID	Type	KGO
3759	2605	BBC2TFT1	D	1
4509	2605	BBC2TFT2	D	1
5259	2605	BBC2TFT3	D	0
6009	2605	BBC2TFT4	L	1
6759	2605	BBC2TFT5	L	1
7509	2605	BBC2TFT6	L	0
8259	2605	BBC2TFT7	D	1
9009	2605	BBC2TFT8	D	1
9759	2605	BBC2TFT9	D	1

Table 8. Example of simple KGO map.

This table corresponds to a portion of an IGA template that has a configuration of 3 Drive OTFTs, followed by 3 Load, and followed again by 3 Drive (3D-3L-3D). Only OTFTs which have a 1 on the KGO field are usable for circuit mapping. If all OTFTs are marked as KGOs then the yield is 100%. Consequently the yield is reduced whenever failures are introduced into the file.

The KGO map is a direct result of the characterization process, and its data represent the working status of the OTFTs for each IGA substrate. However it is possible to introduce failures and even restriction zones on this map, either randomly or intentionally, for testing purposes of the P&R. Furthermore, it is possible to add more constraints into this file for more complex P&R functionalities. Notice that even if the OTFTs are marked as KGOs their utilization will still depend on the P&R procedure, which will be detailed in following subchapters.

4.2. Logic synthesis and technology mapping

As commented in the introductory chapter, logic synthesis is an extremely important step of any chip design flow, regardless of the technology being targeted. Logic synthesis is the set of techniques aimed at minimizing the circuit size, by reducing its number of gates and thus transistors. This will consequently reduce the substrate area required by the chips, hence reducing overall fabrication costs. Logic synthesis processes perform combinational logic reductions applying Boolean algebra.

Logic synthesis has a strong correlation with technology mapping. In fact, many tools combine both functionalities to optimize circuits even more, whenever possible. A technology mapping process synthesizes a circuit functionality into a particular cell library.

For technology independent and technology dependent (after mapping) minimizations we used the ABC tool [66]. ABC achieves excellent results by applying reduction algorithms to And-Inverter-Graphs (AIG) [276] combined with other transformations over different graph-based representations, such as Binary-Decision-Diagrams (BDD) and others.

Table 9 and Table 10 show the minimizations achieved by the ABC tool for a combinational ASPEC example which was mapped to different cell libraries. This ASPEC is a tic-tac-toe game functionality circuit presented in [58].

Gates & Costs	**Cell library**			
	NAND2 / INV1	**NAND2/ NAND3/ INV1**	**NAND2/ NAND3/ NAND4/ INV1**	**NAND2/ NAND3/ NAND4/ NAND5/ INV1**
# INV1	12	2	0	0
# NAND2	105	85	83	83
# NAND3	0	10	8	8
# NAND4	0	0	2	2
# NAND5	0	0	0	0
TOTAL # Gates	117	97	93	93
Cost (#OTFT)	339	299	291	291

Table 9. Logic synthesis and technology mapping example (before ABC).

Gates & Costs	**Cell library**			
	NAND2/ INV1	**NAND2/ NAND3/ INV1**	**NAND2/ NAND3/ NAND4/ INV1**	**NAND2/ NAND3/ NAND4/ NAND5/ INV1**
# INV1	30	9	8	8
# NAND2	55	13	12	12
# NAND3	0	21	20	20
# NAND4	0	0	1	1
# NAND5	0	0	0	0
TOTAL # Gates	85	43	41	41
Cost (#OTFT)	225	141	137	137
Cost reduction %	33,62%	52,85%	52,93%	52,93%

Table 10. Logic synthesis and technology mapping example (after ABC).

The maximum reduction on the number of transistors is achieved by using a library of Inv/NAND2/3/4. However, because the NAND4 is only appearing one time we can omit its inclusion into our cell library and also due to the fact that it has more complex sizing strategy. This way, we reduce the design and testing effort that requires creating a new cell, while keeping the library and final design more homogenous. The availability of a NAND5 gate has no effect in the reductions whatsoever (at least for the proposed circuit). Depending on the available library of gates, the more fan-in that gates might have the less Load OTFTs are necessary in a BBC. But the number of gates with high fan-in (4, 5…) in most circuits is low compared to gates with low fan-in (2, 3). This behavior was observed through empirical technology mapping experiments with benchmarks. Further investigations on the ideal cell libraries for PE and technology mapping tools were also shared in [121]. In the end we concluded that a library of Inv, NAND2/3 is the most convenient for our purposes, since it is possible to build every circuit with them [277]; and the logic synthesis and technology mapping processes seem to give better results, according to the experiments.

Additionally, as explained in [62], I proposed as examples the small sequential benchmark circuits coming from ISCAS-89 [278] and ICT-99 [279]. Those were selected for their small size, taking into account most of the current IGA capacities. Table 11 and Table 12 show the minimizations achieved mapped to our selected cell library.

Gates & Costs	Benchmark circuits				
	ISCAS-89		ICT-99		
	s27	s208.1	b01	b02	b06
# INV1	9	32	11	5	11
# NAND2	6	34	21	13	21
# NAND3	1	23	11	4	13
TOTAL # Gates	16	89	43	22	45
Cost (#OTFT)	40	258	129	65	137

Table 11. Benchmark examples over our fixed PMOS-only cell library (before ABC).

Gates & Costs	Benchmark circuits				
	ISCAS-89		ICT-99		
	s27	s208.1	b01	b02	b06
# INV1	8	28	11	6	7
# NAND2	6	38	16	10	23
# NAND3	1	6	11	5	9
TOTAL # Gates	15	72	38	21	39
Cost (#OTFT)	38	194	114	62	119
Cost reduction %	5%	24,8%	11,6%	4,6%	13,1%

Table 12. Benchmark examples over our fixed PMOS-only cell library (after ABC).

The given data corresponds to the mapping over a PMOS-only library. If the library was CMOS we can compare the data and infer the behavior of the technology mapping tools. Table 13 shows the mapping over CMOS.

Gates & Costs	Benchmark circuits				
	ISCAS-89		ICT-99		
	s27	s208.1	b01	b02	b06
# INV1	8	28	13	6	9
# NAND2	6	38	18	10	25
# NAND3	1	6	9	5	7
TOTAL # Gates	15	72	40	21	41
Cost (#OTFT)	46	244	152	82	160
Cost increase %	21,05%	25,77%	33,33%	32,26%	34,45%

Table 13. Benchmark examples over our fixed CMOS cell library (after ABC).

From Table 13 we can detect how the mapping is the same for the circuits s27, s208.1 and b02, regardless of the targeted technology. The differences are in b01 and b06 circuits. In CMOS the initial map process assigns more Inverters and Nand2, but less Nand3 than in PMOS-only; so the ABC tool is optimizing the mapping according to the target library and its corresponding gate transistor counts (which is higher for Nand3).

Optimizing for CMOS and PMOS technologies is different due to the distinct impact of inverters vs. logic gates costs. Since CMOS always requires a higher number of transistors than PMOS, the lower it is the fan-in of the gates available in the library, the better the cost minimizations will be.

The last row of Table 13 indicates the difference in costs respect to the PMOS-only counterparts.

4.3. Placement and Routing strategy and algorithms for IGA

4.3.1. Preliminary considerations and terminology

We have already reviewed how Placement and Routing (as main stages of physical synthesis for small circuits) are different problems, yet highly correlated. As a matter of fact, they can be processed simultaneously. In our IGA case, Placement and Routing have to be inevitably considered as a conjoint process. This is because the Placement itself is done by occupying the routing tracks locally, in the personalization step. As a reminder, Placement in the IGA is done at transistor-level, not at gate-level, as it could be expected from a conventional standard cell process.

In an environment where we still do not have timing information (which would allow the proper modelling of race conditions, glitches, etc.); and, consequently, we cannot iterate a P&R solution over timing constraints (as cost functions), the initial P&R solution has to take this into account in order to implicitly minimize propagation delays caused from various path lengths, e.g. gate distances to I/O pins, to other gates, etc.

To this end, a specific cell was designated, called Basic Bulk Cell (BBC). A BBC is a cell that acts as a partition/container of the floorplan, where a limited number of OTFTs, and consequently gates, can fit. It forces a constraint for the transistor-level placement of gates, which is that all the transistors necessary to build a particular gate must be placed in the same BBC. This way we avoid having transistors from a gate dispersed through the IGA, which will probably result in excessive intra-gate delays that, ultimately, might cause the gate not to work properly (due to wiring and via resistance that can affect voltage transfer curve); apart from the potential increase of occupied tracks due to global routing.

Let us review a BBC in more detail. Figure 28 shows a schema of a proposed BBC structure and its associated wire cell.

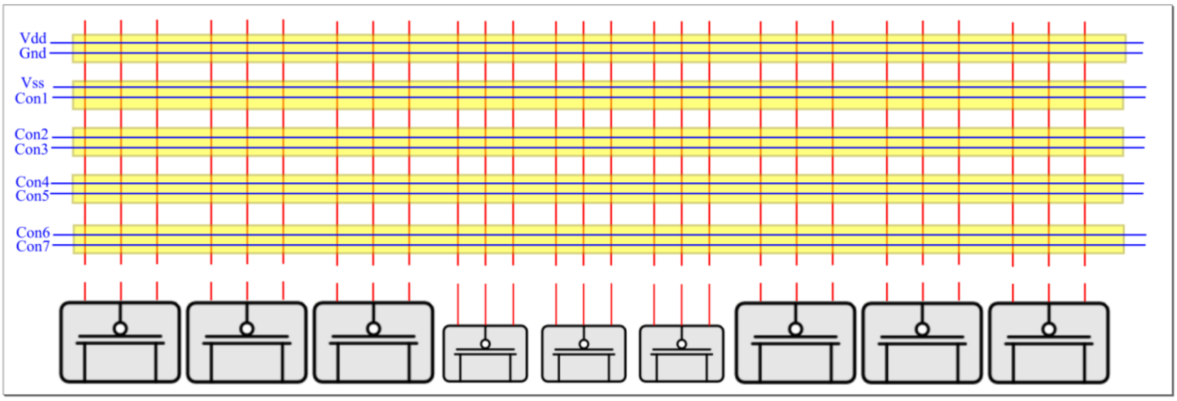


Figure 28. BBC & wire cell schema (example 1).

This BBC has a configuration of 3 Drive OTFTs, followed by 3 Load OTFTs, followed by 3 more Drive OTFTs (3D-3L-3D). Once the IGA is fabricated, and if the yield of the technology is not expected to reach the desirable goal (close to 100% yield), we will proceed with an intermediate characterization step of all OTFTs, right in between the 2-step fabrication process. In our notation we will denominate the properly-working OTFTs as Known Good OTFTs (KGO). Therefore, after characterization and test we will obtain a failure distribution map, or its conceptually opposite: a KGO map.

Figure 29 depicts the mapping of a Nand2 gate over the BBC. We can assume that in this case there is one OTFT that we have considered to be a faulty one, whatever our threshold might be for making such consideration (e.g. high gate current leakages, unstable devices, etc.). Therefore that OTFT has to be avoided for any circuit mapping in the remaining personalization step.

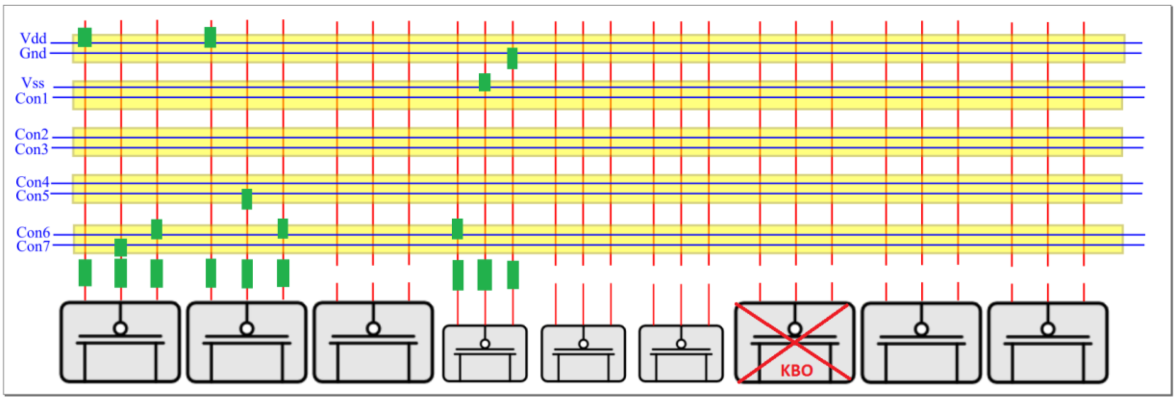


Figure 29. Nand2 mapping over BBC.

As we know, for a PMOS-only design style the number of KGOs needed for mapping a gate would be its number of inputs (corresponding to the Drive KGOs) plus one Load KGO as pull-down network. In this particular example the yield achieved would be 8/9 => 89%. The failing OTFT is not affecting the mapping of that Nand2 into the BBC, but if the P&R process continues to map gates into that BBC then the P&R will abstain from using it.

Figure 30 illustrates the mapping of a Nand2 gate in a BBC that has a different KGO map. In this case the yield would be 5/9 => 55%.

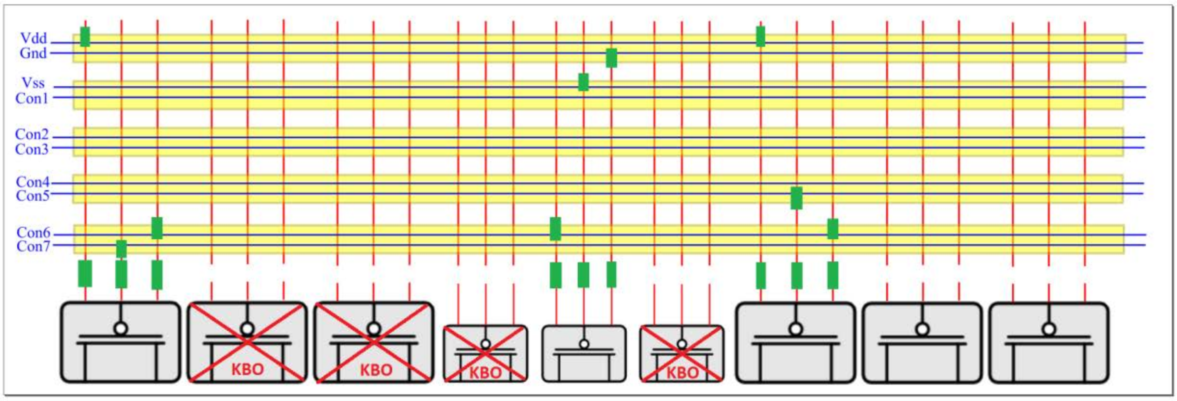


Figure 30. Nand2 mapping over BBC with another KGO map.

The P&R process will skip the faulty OTFTs to map the gate. Since the only Load KGO of that BBC has already been used, no more gates can be placed within that BBC. Hence, the two remaining KGOs will never be utilized.

We can explore what are the possible gate combinations and KGO occupation depending on the BBC configurations for our PMOS-only library, composed by Inverter, Nand2 and Nand3. If the design style is PMOS-only then the maximum number of gates that can be mapped onto a BBC is limited by the number of Load KGOs. For example, if the configuration is 1D-1L-1D it is possible to map there only one Inverter (and 1D KGO will be lost, since it is a surplus), or one Nand2 which will occupy the cell completely. If the design style is CMOS the maximum number of gates that can be mapped onto a BBC would require the same number of PMOS and NMOS KGOs for the fan-in of the gates that we would like to allocate. For instance, if we have a configuration of 2P-2N-2P I can only map either two Inverters or only one Nand2, where two is the fan-in that occupies 2N-2P KGOs, and 2P KGOs will be lost.

Table 14 shows some combinations of mapping examples for a PMOS-only design style for our proposed library of gates, depending on the BBC configuration that any IGA might present, supposing a yield of 100%.

BBC Configuration	# NAND3	# NAND2	# INV	# KGO Excess
1D-1L-1D	N.A.	1	0	0
		0	1	1D
1D-2L-1D	N.A.	1	0	1L
		0	2	0
2D-1L-2D	1	0	0	1D
	0	1	0	2D
	0	0	1	3D
2D-2L-2D	1	0	1	0
	0	2	0	0
	0	1	1	1D
	0	0	2	2D
2D-3L-2D	1	0	1	1L
	0	2	0	1L
	0	1	2	0
	0	0	3	1D
3D-1L-3D	1	0	0	3D
	0	1	0	4D
	0	0	1	5D
3D-2L-3D	2	0	0	0
	1	1	0	1D
	1	0	1	2D
	0	2	0	2D
	0	1	1	3D
	0	0	2	4D
3D-3L-3D	2	0	0	1L
	1	1	1	0
	1	0	2	1D
	0	3	0	0
	0	2	1	1D
	0	1	2	2D
	0	0	3	3D
3D-4L-3D	2	0	0	2L
	1	1	1	1L
	1	0	3	0
	0	3	0	1L
	0	2	2	0
	0	1	3	1D
	0	0	4	2D

Table 14. Mapping examples over different BBC configurations.

From Table 14 notice the following:

1) A 1D-xL-1D configuration where x>2 is not of interest. Increasing # L does not have any effect on the mapping possibilities when 1D-1D is fixed.
2) Incrementing # L could be beneficial whenever there are configurations that have D excesses in any of their combinations. However, it might not be convenient to do so, because the increment of # L will also increase the combinations that have L OTFT surpluses. This will depend on the fan-in statistics that each circuit (i.e. netlist) might present, once they have been mapped to a particular technology/library; and it will also depend on the circuit's placement strategy.
3) For CMOS, a BBC configuration that has an equal amount of P-N OTFTs would obviously be the best fit, since the pull-up and pull-down networks are complementary; while in PMOS-only it would be # L ≈ # D/2.

Further enlargements of the BBC in respect to the number of OTFTs will increase greatly the local connectivity required when gates are mapped onto them, while it does not necessarily reduce the global routing (and consequently the wire tracks needed along the IGA). Hence, it is better to have a compromise so that fewer tracks are required for the total IGA by using relatively small size BBCs. In general, a 3D-3L-3D topology is a good choice for mapping our gate library since it keeps the wiring numbers relatively low while it allows some individual OTFT failures[21].

The collection of KGOs is a subset of the collection of OTFTs, {KGO} ⊆ {OTFT}. In our case only the KGOs will be considered for mapping gates. As mentioned before, the maximum number of gates that can be placed into a BBC is limited by its number of Load KGOs. Every OTFT has a list of characterization pads associated that can connect to the connectivity zones of the wiring tracks. These connectivity zones represent areas where the customization metal could be imprinted, whether vias, gaps or any other shape. For every KGO, we only need the position tuple (***x,y***) of each of its pads/vias (i.e. connectivity zones) and their width (which should be uniform for the entire IGA). We can denote this as the set of positions of the KGO's pads {***pkp***} ∈ KGO. We also have vias (or connectivity zones) in the wire cells which are also defined by their (***x,y***) position tuple and their width. We can denote those {***pwv***} ∈ ***wt***, where ***wt*** is one of the wire tracks ({***wt***} ∈ ***wc***), and ***wc*** is the wire cell corresponding to a BBC. The set of I/O nets of a gate ***g*** can be denoted as {***i_net(g)***} and {***o_net(g)***}. Additional control parameters are:

1) The number of BBCs ***n_bbcs*** and their configuration (will vary between processes, e.g. 3D-3L-3D).
2) The number of Drive ***n_d(bbc)*** and Load ***n_l(bbc)*** KGOs within each BBC (will vary between substrates and they are determined by characterization).

[21] For the technologies we have used there are no studies about fault statistics. Thus, according to the collected experience, we are considering them as randomly distributed. Any bias could easily be taken into account in our work.

3) The number of wire tracks ***n_wt*** per wire cell (should remain the same for a particular process and specific IGA architecture).
4) The number of I/O pins ***n_p***.

In this context placement consists in deciding where to implement the required wire connections from the DKGO and LKGO pads to the connectivity zones of the wire tracks. This is how a gate is built (placed) within a BBC, which merges also with the concept of local routing at BBC level.

Before any P&R takes place, it is necessary to give wiring estimations for the IGA at design level. This will depend on the BBC configurations, design style and final size of the IGA in terms of number of BBCs and OTFTs within them. Since the main objective of the P&R process is to guarantee routability we should always address worst case scenarios, even if their probability of occurrence is low.

Figure 31 shows a detailed example of mapping, where we can observe the wire cell and how the connections are done with each KGO's terminals. It is clear that wiring estimations will increase the bigger the BBC configuration is. If the horizontal tracks were not separated but they were forming a continuous line this would mean that each one of the inputs and outputs would require its own horizontal track, whether if they were actually connected vertically or not. In this case the more BBCs are in a row, the more horizontal tracks will be necessary (linear relationship). This happens whenever all the I/O of the gates within a BBC are global. In case some of the I/Os were local (for example in Figure 31, if the output of Nand3 O_N3 was connected to input of Inv I_Inv) we could save some tracks (in the mentioned example the track I_Inv). Therefore, with local connections we could decrease the number of necessary tracks by one each time there is a gate that connects to another one within the same BBC. In the proposed example if all gates (Nand3, Nand2, Inv) were connected locally this would mean 2 local gate connections which would achieve a reduction of 2 horizontal tracks (the input ones). However, since wire estimations should consider worst case scenarios so that mapping does not fail, we cannot take for granted these optimizations in an IGA P&R strategy. Still, the gate selection criteria for P&R takes into account these possible scenarios in order to occupy the tracks efficiently.

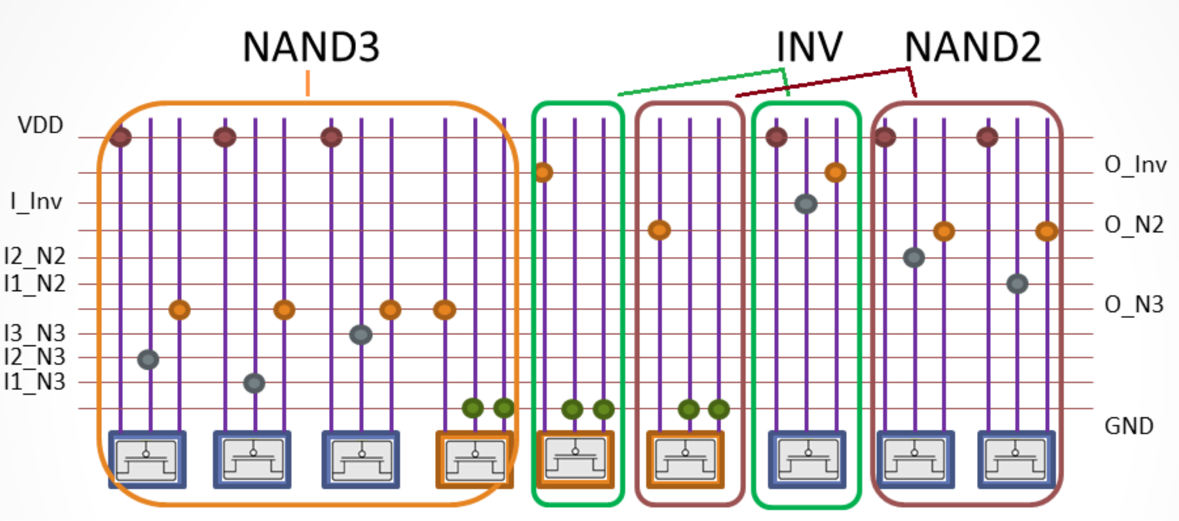


Figure 31. Mapping example in detail.

Horizontal wires should not be connected between BBCs by default, only on demand depending on the mapping and global routing. This is how it is done in some IGA designs but not in every horizontal track. Power tracks are of course continuous but also other tracks in the wire cells. This was done for convenience, as sometimes it might seem that it could be useful to have continuous tracks that are global. In terms of data structures and heuristics it would be easier not to have to detect which wires are continuous by default and which ones are not. Also for design homogeneity and repeatability, for locating connectivity zones such as via coordinates.

All things considered, the maximum number of tracks will be reached when the BBC is fully occupied. In the previous case 3D-3L-3D this would be for 6 inputs + 3 outputs, which is exactly the number of KGOs + 2 power tracks, which in total are 11 horizontal tracks for just one BBC. If the horizontal tracks were not disjoint between adjacent BBCs, and considering the worst case scenario (when the circuit mapping does not benefit from any local connectivity optimizations and, as a consequence, does not reduce the wire tracks needed), this would require (9 wires · #BBCs along one row) + 2 power tracks (VDD, GND) common for all. Hence, it is a worst case situation that could occupy plenty of substrate area.

Figure 31 shows that, for every mapped gate, every D KGO needs to be connected to VDD while every L KGO needs to be connected to GND. Since OTFTs have a symmetric distribution, it does not matter if the drain and source are on the left or right pad. For uniformity reasons, we always placed VDD on the left pad and GND on the right pad; even when, like in the latest designs, the OTFTs are rotated 180º for saving area. In the latest IGA designs VDD and GND tracks are the most upper and lower wires of the wire cell respectively. The inputs shall connect with the gate terminal of every D, which in our OTFT layouts is the middle pad. There is no VSS power track so L KGO gate terminals are grounded or output-connected, depending on the preferred style: diode-connected or Zero-V_{GS}, respectively.

Once the routability goal is guaranteed we can consider other objectives. In our case those would be:

1) Maximizing compaction / minimizing whitespaces and area spent.

The approach is to map the gates starting on the bottom left corner and continue with a row-based growth (similar to cluster growth). It is a constructive greedy methodology (selecting one gate at a time until there are no more left to place). There will not be any whitespaces, as they are defined, since no mapping spaces will be left empty intentionally. In any case, the gate selection criteria will try to minimize the KGO excesses for every BBC being processed.

Another benefit of compaction is that, if the circuit mapped over the IGA is too small compared to the IGA capacity and given yield, the empty right part of the IGA might be used to map other test structures in the personalization step (provided that there are still enough I/O pins available if needed), to evaluate more circuits/structures in less foils. Even redundant logic can be another possibility for occupying more the empty parts of each IGA.

2) Minimizing delays.

This is achieved partly thanks to the compaction goal, but also with the BBC container constraint, and the gate selection criteria. With the BBC constraint we ensure that no gate is placed dispersedly over the IGA, minimizing intra-gate delays. The gate selection criteria will try to place the gates that have I/O fan-in or fan-out closer to the I/O pins/pads. The track selection and occupation will always be the one that are free and closer to the BBC being processed.

The gate selection criteria for placement is the following:

1) First the gates that have more inputs coming from the I/O pins.
2) Then the gates that have more inputs coming from already placed gates.
3) Then any other unplaced gates, from highest to lowest fan-in, until is no longer possible to fit anymore into the current BBC.

If comparable, it would resemble more a Breadth-First methodology rather than a Depth-First one, since it does not traverse through the netlist selecting gates in a depth-based manner. A Depth-First variant of the gate selection criteria would be feasible, although it would not provide as optimal results in terms of the mentioned secondary objectives of our P&R process. In our case, we check the BBC capacity and according to it we select a gate that has the maximum fan-in that can fit given our criteria. Once a BBC is complete (no more Load/Drive KGOs are available within it), or the rest of the unplaced gates from the netlist do not fit in it; then we move on to place the next gate on the contiguous BBC, first moving in the Y axis from bottom to top, and then in the X axis from left to right. The I/O pins are being mapped clockwise, starting at the bottom left corner like the placement.

This algorithm is deterministic, hence it will produce the same results as long as the fault distribution (or its opposite KGO map) does not change.

4.3.2. Proposed Algorithm

Table 15 shows a brief outline of the P&R strategy. The level of abstraction of such summary is quite high, written in pseudocode, in order to easily understand the functionality.

Place and Route strategy

Inputs: IGA layout, KGO map, circuit netlist, and functional library.

Output: layout with the resulting P&R.

```
1: while circuit netlist has not been completely placed and routed, and there are still
   free BBCs (i.e. with enough D-L KGOs) in the IGA do
2:     Select BBC.
3:     Select gate according to criteria.
4:     Place & Route gate. This is done by locally (inside each BBC)
       routing/occupying the IGA tracks. Placement also determines global
       routing (between BBCs).
5: end while // 1
```

Table 15. Outline of the P&R strategy.

Some notation explanations for the pseudocode are in Table 16. These are necessary definitions for whenever the names are not straightforward nor self-explanatory.

Specific notation	
$\{e\}$ ≡ set of elements e (e.g. BBCs…)	$\lvert\{e\}\rvert$ ≡ Cardinality of the set
Ug ≡ Unplaced gate	***Pg*** ≡ Placed gate
Cz ≡ Connectivity zone (e.g. via, gap…)	***n_d(bbc)*** ≡ number of free Drive KGOs of current BBC
Sg ≡ selected gate	***n_l(bbc)*** ≡ number of free Load KGOs of current BBC

Table 16. Algorithm's specific notation.

There are some situations where the P&R process will not work. These cases are preferably expressed by returning error codes ('return' in C) instead of stopping the execution immediately ('exit' in C), so that the program can be utilized as a library and the errors can be handled by a caller module. All the codes are collected in Table 17. Additional codes can be added along with other checks or features.

Return codes	Meaning
return(0)	The main program or subroutine finishes correctly, and returns to the calling program.
return(-1)	The circuit has no gates to map.
return(-2)	There are no BBCs available for P&R, according to the KGO map.
return(-3)	Place & Route not finished satisfactorily because there were gates that could not be mapped.

Table 17. Return codes.

The pseudocode provided in the forthcoming tables presents a relatively high level of abstraction compared to more simple programs. There are several reasons for this, including, but not limited to, the following ones:

1) The P&R tool alone, in its current state, is roughly more than 1500 lines of functional code, excluding other third party libraries that are also being imported and utilized. Such amount of information needs to be expressed in a more concise way, which means abstracting more from unnecessary details, to a certain extent.
2) The designs vary from one technology to another, and even they might vary from one fabrication run to the next one over the same technology. This signifies that the data can be different, affected by global and local connections, wire tracks and track continuity, BBC configurations, initialization values, identifiers, presence or absence of vias, auxiliary wire cells, and so on and so forth.
3) Different programming languages might have some data structures or operations included as built-in types or procedures, or available in other libraries. These data structures and operations might have distinct time complexities, and could be more convenient than others depending on the processing flows. They also have their own mechanisms for updating their values. A higher level of abstraction allows more freedom of implementation, so it makes sense from a language-agnostic algorithmic point of view.

Table 18 depicts the main P&R algorithm's pseudocode.

Place and Route algorithm

Inputs: IGA layout, KGO map, circuit netlist, and functional library.

Output: layout with the resulting P&R for digital printing.

```
1: {Pg} ← Ø
2: n_pg ← |{Pg}|
3: Get({Ug})
4: if {Ug} == Ø then
5:      return(-1)
6: n_ug ← |{Ug}|
7: Get({BBC})
8: if {BBC} == Ø then
9:      return(-2)
10: Get({Cz})
11: for all c ∈ {BBC} do :
12:     if (n_l(c) > 0) and (n_d(c) > 0) then
13:          Sg ← Select_Gate({Ug}, n_d(c))
14:          if Sg != Ø then
15:               Place&Route(sg, c, {Cz}, {Ug}, {Pg})
16:          n_pg ← |{Pg}|
17:          if ({Ug} == Ø) and (n_ug == n_pg) then
18:               return(0)
19: end for // 11
20: n_pg ← |{Pg}|
21: if ({Ug} == Ø) and (n_ug == n_pg) then
22:     return(0)
23: else :
24:     return(-3)
```

Table 18. Main algorithm's pseudocode.

In Table 18 it would be possible to do an initial P&R feasibility analysis right before the 'for' loop in line 11. This preliminary check would try to avoid the entire P&R

execution whenever it is obvious that the number of KGOs of a substrate is less than the number of transistors needed to map a circuit netlist. This check can be done in two ways. Either the values are pre-known and passed directly as arguments, or the procedure calculates them in a pure automated design flow (i.e. without a user having to intervene). In the first case the final number of transistors would be obtained after the logic synthesis and technology mapping process, while the number of KGOs of each type would be given by the KGO map. If those two parameters were passed directly (hard-coded or as user inputs) it would not be necessary to traverse the data structures to obtain them. In the latter case, the computing overhead introduced for calculating the feasibility justifies the avoidance of its inclusion, since it will be necessary to traverse the data structures twice, a first time for the check and a second for the P&R processing. In any case, whenever a feature requires possibly as much time to execute as the main program, without doing anything more than what the main program actually does in due time, this feature can be certainly discarded.

Table 19 shows the subroutine Select_Gate:

Select_Gate({Ug}, n_d(c))

Input: the set of unplaced gates and the number of free Drive KGOs (maximum fan_in possible) of the current BBC.

Output: selected gate.

```
1: selected_gate ← Ø
2: n_io, n_pn, n_upn ← 0
3: for all g ∈ {Ug} do :
4:     if fan_in(g) ≤ n_d(c) then
5:         for ip ← 0 … n_input_ports(g)-1 do :
6:             if port_type[ip] == I/O_pin then
7:                 n_io ← n_io + 1
8:             else if port_type[ip] == placed_net then
9:                 n_pn ← n_pn + 1
10:            else : // unplaced_net
11:                n_upn ← n_upn + 1
12:        end for // 5
13:        if n_io == n_d(bbc) then
14:            selected_gate ← g, (n_io, n_pn, n_upn)
15:            return(selected_gate)
16:        if (n_io > selected_gate(n_io)) or ((n_io == selected_gate(n_io)) and
           (n_pn > selected_gate(n_pn))) or ((n_io == selected_gate(n_io)) and
           (n_pn == selected_gate(n_pn)) and (n_upn > selected_gate(n_upn)))
           then
17:            selected_gate ← g, (n_io, n_pn, n_upn)
18:     n_io, n_pn, n_upn ← 0
19: end for // 3
20: return(selected_gate)
```

Table 19. Subroutine 'Select_Gate'.

In Table 20 the pseudocode for the subroutine called ‘Place&Route’ is shown:

Place&Route(g, bbc, {Cz}, {Ug}, {Pg})

Input: selected gate, current BBC, set of connectivity zones, and set of unplaced gates and placed gates.

Output: gate placed in the BBC (back-end data and front-end layout), removed from {Ug} and inserted in {Pg}, updating the connectivity zones, wire cell occupation statuses, and number of D KGOs and L KGOs of the BBC.

```
1:  for d_kgo ← 0 … fan_in(g)-1 do :
        // 3 terminals per DKGO
2:      wt_id ← wt_VDD
3:      wt_value[wt_id] ← VDD
4:      cz(pwv) ← (Occupied, VDD)
5:      Draw_custom_metal_rectangle(cz(pwv))
6:      Replace(cz(pwv), {Cz})
7:      cz(pwv) ← Next_free(cz(pwv))
8:      wt_id ← Next_free(wt_id)
9:      found ← False
10:     wt_found ← Ø
        // tries to find a wire track (wt) that has the current input already placed
        elsewhere
11:     while (not found) and (wt_id ≤ max_wt_id({Cz})) do :
12:         if wt_value[wt_id] == d_input(g) then
13:             found ← True
14:             wt_found ← wt_value[wt_id]
15:         else :
16:             wt_id ← Next_free(wt_id)
17:     end while // 11
18:     if (found == True) and (wt_found != Ø) then
19:         if wt_found ∈ bbc then
                // local connectivity within BBC
20:             wt_value[wt_id] ← wt_found
21:             cz(pwv) ← (Occupied, wt_found)
22:             Draw_custom_metal_rectangle(cz(pwv))
```

```
23:             Replace(cz(pwv), {Cz})
24:             cz(pwv) ← Next_free(cz(pwv))
25:             wt_id ← Next_free(wt_id)
26:         else :
            // global connectivity between BBCs
27:             chosen ← False
28:             while (not chosen) and (wt_global_id ≤
            max_wt_global_id({Cz})) do :
29:                 if wt_global_id_status == free then
30:                     chosen ← True
31:                     wt_global ← wt_found
32:                     cz(pwv) ← (Occupied, wt_global)
33:                     Draw_custom_metal_rectangle(cz(pwv))
34:                     Replace(cz(pwv), {Cz})
35:                     cz(pwv) ← Next_free(cz(pwv))
36:                     wt_global_id ← Next_free(wt_global_id)
37:                 else :
38:                     wt_global_id ← Next(wt_global_id)
39:             end while // 28
40:             wt_value[wt_id] ← wt_found
41:             cz(pwv) ← (Occupied, wt_found)
42:             Draw_custom_metal_rectangle(cz(pwv))
43:             Replace(cz(pwv), {Cz})
44:             cz(pwv) ← Next_free(cz(pwv))
45:             wt_id ← Next_free(wt_id)
46:     else :
        // new free wire track
47:         wt_value[wt_id] ← d_input(g)
48:         cz(pwv) ← (Occupied, d_input(g))
49:         Draw_custom_metal_rectangle(cz(pwv))
50:         Replace(cz(pwv), {Cz})
51:         cz(pwv) ← Next_free(cz(pwv))
52:         wt_id ← Next_free(wt_id)
```

```
    if d_kgo == 0 then
        // the first time the output is placed on a free track
        wt_value[wt_id] ← output(g)
        wt_output(g) ← output(g)
        cz(pwv) ← (Occupied, output(g))
        Draw_custom_metal_rectangle(cz(pwv))
    else :
        // following times the output is placed in the same wire track in a
        different free connectivity zone
        wt_value[wt_id] ← wt_output(g)
        cz(pwv) ← (Occupied, wt_output(g))
        Draw_custom_metal_rectangle(cz(pwv))
    Replace(cz(pwv), {Cz})
    cz(pwv) ← Next_free(cz(pwv))
    wt_id ← Next_free(wt_id)
    DKGO(g)[d_kgo] ← placed
end for // 1
  // 3 terminals per LKGO
wt_id ← wt_GND
wt_value[wt_id] ← GND
cz(pwv) ← (Occupied, GND)
Draw_custom_metal_rectangle(cz(pwv))
Replace(cz(pwv), {Cz})
cz(pwv) ← Next_free(cz(pwv))
cz(pwv) ← (Occupied, GND)
Draw_custom_metal_rectangle(cz(pwv))
Replace(cz(pwv), {Cz})
cz(pwv) ← Next_free(cz(pwv))
wt_id ← wt_output(g)
wt_value[wt_id] ← wt_output(g)
cz(pwv) ← (Occupied, wt_output(g))
Draw_custom_metal_rectangle(cz(pwv))
Replace(cz(pwv), {Cz})
LKGO(g)[0] ← placed
```

```
83: Remove g from {Ug}
84: Append g in {Pg}
85: n_l(bbc) ← n_l(bbc) - 1
86: n_d(bbc) ← n_d(bbc) - fan_in(g)
87: return(0)
```

Table 20. Subroutine 'Place&Route'.

In Table 20 notice the difference between PMOS-only and CMOS design styles. In PMOS the ‘for’ loop at line 1 is intended for all the inputs that will correspond to each DKGO of the gate being processed. Since only one LKGO is needed for mapping a gate in PMOS-only, this loop is not necessary for the LKGO. However, if we were addressing a CMOS design style the loop should be replicated in a similar manner to complement the fan-in of the gate. This means that the input ports would also have to be directed to the corresponding gate terminals on the pull-down network. Likewise, in PMOS the number of Load ***n_l(bbc)*** KGOs of each BBC has to be decremented by one when one gate is placed. In CMOS it should be decremented the same number of Drive ***n_d(bbc)*** KGOs, i.e. by the ***fan_in(g)*** of the gate to be placed. A BBC in CMOS should have a topology with a ***n_l(bbc)*** equal (or at least as close as possible) to ***n_d(bbc)***.

Also, the lines 72 and 73 can vary depending on the preferred choice for the Load KGO configuration. For instance, in this case a diode-connected form for the pull-down transistor is represented. For representing a Zero-V_{GS} style, instead of connecting the gate terminal of the LKGO to the connectivity zone in the wire track that is occupied by GND, that gate terminal should be connected to the connectivity zone of the wire track ***wt_output(g)*** already given by the previous connections made for the DKGOs; in a similar way as the drain terminal.

Furthermore, it would certainly be possible to separate the back-end processing, related to netlist handling and statuses updating, from the front-end layout drawings. This would increase the modularity and overall clarity of the implementation. However, it would not be algorithmically efficient because it will introduce a computing overhead for having to traverse the data one more time for making the corresponding layouts in a posterior procedure.

In any case, there is always room for optimization and additional checks and features. But it will always be necessary to take into account what are the computational costs of such features, and if it would be desirable to increase the scope and functionalities of the strategy proposed herein. Some proposals will be mentioned in the section for future work.

4.4. Results

4.4.1. Proposed circuit benchmarks and IGA templates

For demonstrating the P&R strategy and algorithm we specifically targeted the CPI/Neudrive technology and its corresponding IGA design, in the latest fabrication topology [82]. I selected basic combinational circuit functionalities which are presented in the following Table 21. The circuits were mapped to our library of Nand3, Nand2 and Inverter gates. C17 is from ISCAS-85 [280].

Circuit	Fan -in	Fan -out	#NAND3	#NAND2	#INV	#Total	Cost #OTFTs
Multiplexer	2	1	0	3	1	4	11
Demultiplexer	1	2	0	2	3	5	12
C17	5	2	0	6	0	6	18
Decoder	2	4	0	4	6	10	24
Encoder	4	2	3	4	6	13	36
Tic-Tac-Toe	18	9	21	13	9	43	141

Table 21. Benchmarks for P&R demonstrations.

4.4.2. Implementation framework

The P&R tool, which targets the IGA strategy by including the algorithms explained previously, has been implemented utilizing the Python [160] language, currently compatible with versions 2.7.x. The decision of choosing this language is mainly influenced by the previous utilization of the layout tool Glade [71], which supports Python natively through its own database/layout API. This framework is convenient since it is FOSS, portable, and multiplatform. It provides machine independence, one of the key aspects that CAD systems should have, according to [281]. The tool requires the following third party packages and their additional dependencies in order to execute successfully:

1) gdsCAD: gdsCAD [161] is a powerful Python package for handling GDSII layout files. The code publicly available in the repositories was not working properly, and is discontinued, so I had to do some modifications to it. The data structures and organization of this package follows the Object Oriented Programing (OOP) paradigm. The layout information is parsed form the GDS file and stored into their corresponding objects for later processing.

2) Pyverilog: Pyverilog [162] is a very useful tool for Verilog processing, also following the OOP paradigm. It includes a code parser that generates an Abstract Syntax Tree (AST) from the input HDL, a dataflow analyzer, a control-flow analyzer, and a code generator. It uses Icarus Verilog [64] for compiling and synthesizing.

Figure 32 depicts the flow of the implementation. The input files are the IGA layout, KGO map, circuit netlist, and functional library. The P&R processing will generate the digital printing layout pattern according to the yield and KGO map of the targeted technology.

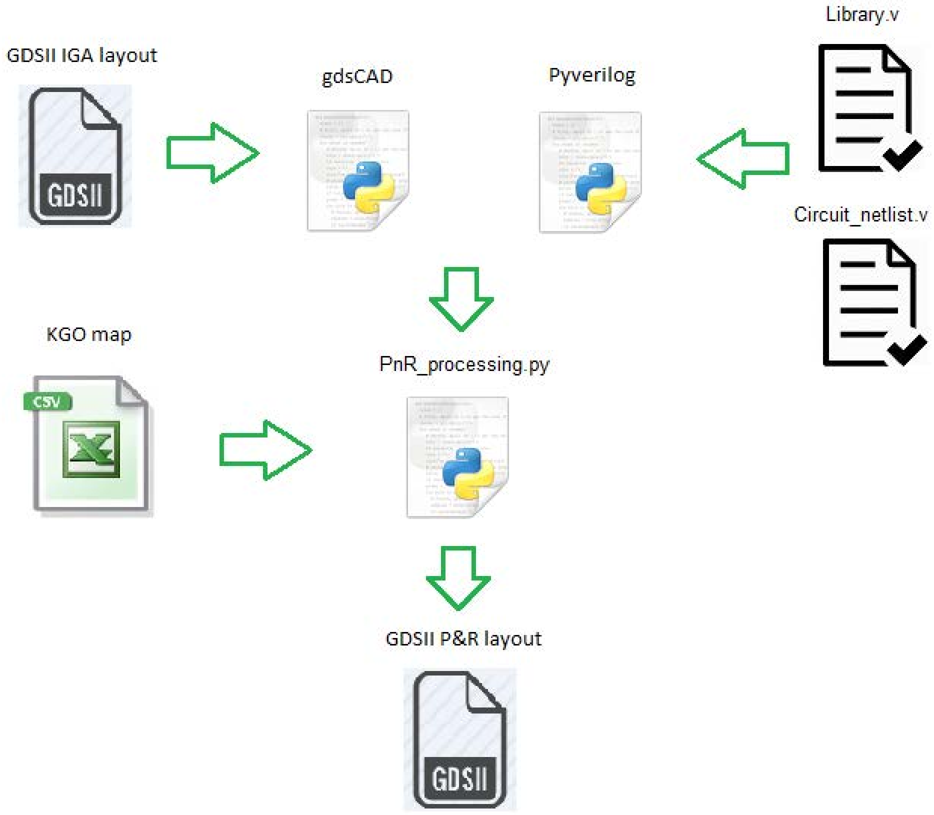


Figure 32. P&R implementation flow.

The implementation also follows the good practices and optimizations, on specific Python data structures and algorithmic properties, explained in several references such as [282-287].

4.4.3. Complexity analysis of current implementation

The time complexity of the P&R solution depends greatly on the data structures of the third-party packages used in the implementation, especially for the importing (parsing) procedures. These are Pyverilog and gdsCAD (and their respective additional dependencies).

As mentioned in [163], Pyverilog parses the RTL circuits represented in Verilog into an AST. From this AST it performs elaboration of the netlist into a dataflow graph-based representation. In doing so, it is already performing with O(c*n), being ‘c’ a constant greater than 1 (because the AST is traversed more than once), and ‘n’ the number of gates of the input gate-level netlist. This package relies mainly on lists, tuples and dictionaries (hash tables), providing methods that work over those structures.

The gdsCAD package works with the layout, matching its data structures to those defined in the GDSII specification [161]. It stores collections of cells into dictionaries. Elements can be added to each cell, and are implemented as python lists of Boundary or Path polygons (with coordinates, layer and datatype). Methods provided in this package usually operate with dictionaries and lists.

Nevertheless, the structures included in the previous packages by default are not enough to provide an efficient P&R heuristic. It is necessary to combine information about I/O pins, wires (nets), cell fan-in and fan-out, wire cell structures, auxiliary connections, particularities of the wire tracks, available KGO types, positions, BBC belonging and occupation; and layout objects such as positions, dimensions, layer types and properties. The construction of the data structures requires additional time, no longer than O(k*n + L + t), where ‘k’ is a constant greater than 1 as well, for the passes needed to traverse the netlist, ‘L’ is the number of layout objects (i.e. shapes) of the GDS, and ‘t’ is the number of transistors of the kgo map file.

Note that Python lists, despite being called ‘lists’, are represented internally as arrays in the CPython implementation (default) [286, 287]. This makes accessing the elements faster, with O(1). Append and Pop will also have O(1) (working over the last element). Knowing the specific details of Python data structures and primitives is essential to improve efficiency. However, the algorithms may perform differently over other implementations and/or languages. It is also especially relevant to be aware of what objects are being modified directly, and what others are not. As a relevant example, performing deep copies of data will help preserve valuable information (at the cost of memory) for later processing whenever necessary.

The P&R processing depends on all the previous data structures and considerations. The worst case time complexity of the algorithm can be considered as O(z*c*n*p + ((p-1)*3*w + 3*w)), where ‘z’ are all the connectivity zones, ‘c’ all the BBCs, ‘n’ all the gates, ‘p’ all of their corresponding ports, and ‘w’ all of the wire tracks. However, worst case scenarios are uncommon. Average times are more reduced, especially if we take into account several processing optimizations (e.g. current positions, wire track reutilizations for already mapped trunks and branches, etc.). Ultimately, we managed to deliver a polynomial time heuristic for the mentioned NP-complete problems. Although, as a trade-off, we cannot guarantee its optimality, due to the inherent nature of this kind of problems.

4.4.4. Experimental results

The running environment was a computer with Intel i7-4700HQ 2.4GHz (4 cores, 8 threads), 16GB DDR3L 1600 MHz RAM, Samsung 840 Evo SSD (where Python and all third party and modified packages were installed), and HDD 7200 rpm (where the P&R tool and the necessary I/O files were handled), with SATA III interface; and 64 bit Windows 10 Pro or 64 bit Ubuntu 16.04 LTS. Note that since the Python implementation is the canonical CPython, it does not actually take advantage of multicore processors because of the Global Interpreter Lock (GIL). This is especially inefficient when dealing with I/O (e.g. waiting for disk operations). This issue, however, does not have a high impact over the P&R tool in its current state.

The execution time was measured using the native ***timeit*** Python module for its 2.7.x distribution. Specifically using the method ***default_timer()*** appropriately. This is the recommended module for performing benchmark comparisons, although it does not measure isolated CPU time (process-wide), but wall clock time (system-wide). This means that other processes running in the computer might affect the measurements, since they are sharing the resources. This explains how the timing values usually vary slightly between Operative Systems and executions. Another reason of the time variability is the presence of previous information in the cache memories between consecutive executions, their architecture, and reading and writing policies. The ***default_timer()*** selects the best timer option depending on the OS where it is executed. In a Windows OS ***time.clock*** has the best precision, whereas in other platforms ***time.time*** is better. Active logging also increases the execution time, depending on the level (severity) set.

The memory utilized was also measured, in this case using the ***memory_profiler*** [288] and ***psutil*** [289] third party Python modules. The memory shown is the memory allocated by the OS for the process being executed, not necessarily the real memory used by Python's interpreter. ***memory_profiler*** does not perform an exhaustive analysis of the allocated objects, but gives a decent approximation. In addition, results will most certainly vary slightly due to the different behaviors of Python's garbage collector. In both, run time and memory used, it is possible to focus and analyze different procedures independently or the complete program as a whole.

Table 22 shows the results obtained for the proposed circuits and conditions.

Circuit	Yield %	Finish code	Execution time (s)		Maximum memory used (MiB)	
			Windows	Ubuntu	Windows	Ubuntu
Multiplexer	100%	**return**(0)	2.73	1.78	82.73	100.07
	74%		2.68	1.72	82.51	99.71
	50%		2.57	1.75	82.09	99.78
Demultiplexer	100%	**return**(0)	2.75	1.82	82.97	100.37
	74%		2.65	1.79	82.72	100.46
	50%		2.61	1.74	81.81	100.02
C17	100%	**return**(0)	3.06	1.79	83.23	100.43
	74%		3.02	1.75	83.09	100.27
	50%		2.92	1.81	82.68	100.25
Decoder	100%	**return**(0)	3.66	1.89	84.53	101.59
	74%		3.51	1.81	83.59	101.02
	50%	**return**(-3)	3.42	1.79	83.32	101.44
Encoder	100%	**return**(0)	4.55	1.91	85.42	102.36
	74%	**return**(-3)	4.62	1.95	85.05	102.48
	50%		4.13	1.82	84.28	101.62
Tic-Tac-Toe	100%	**return**(-3)	8.12	2.06	89.38	106.89
	74%		7.68	2.14	89.11	105.55
	50%		6.26	1.96	87.91	105.48

Table 22. P&R results.

There is a big amount of possible combinations and results, depending on the yield and KGO map; and that there is no specific threshold yield for P&R. Total randomness, or regions with high density of defects can also be introduced in the KGO map at will. Consequently, Table 22 depicts only some representative results. A few examples of layouts showing the digital printing patterns, after having followed our P&R heuristic, can be found in Annex A.

Each IGA foil for Neudrive technology has a capacity of 54 OTFT (36D-18L). The number of KGOs that each circuit requires was detailed in Table 21. To distinguish between Drive and Load KGOs is straightforward since each gate only requires one Load KGO (Pull-Down Network).

In a 100% yield situation only the Tic-Tac-Toe circuit cannot fit in the template[22]. In fact, this circuit is the only one from the benchmarks that cannot be mapped to any foil, since its number of gates (and thus OTFTs) clearly exceeds the IGA capacity. The others can be mapped correctly, though.

As an example, in the 74% yield case this means that 14 OTFTs are not working (whether Drive or Load), out of 54. The presented data consider the same failure distribution for each circuit's synthesis, and that the faulty OTFTs are randomly, but evenly, distributed over the die (i.e. there are no regions with a particular higher concentration of faulty OTFTs). In such scenario, the Encoder is not mapped successfully.

Another example is a 50% yield situation. In this case the Tic-Tac-Toe, Encoder, and Decoder were not mapped successfully. In particular the decoder needs 24 OTFTs, which is actually lower than the 50% of 54 (27). However, it requires 10 gates over the proposed library, and it does not fit in with the chosen fault distribution.

The performance achieved could vary between executions, for the reasons already commented, but, due to the deterministic nature of the proposed P&R, the obtained layouts for digital printing will remain fixed as long as the constraints do not change. In terms of computational complexity, given a particular IGA and KGO map, the number of elements 'n' corresponds to the number of gates of the gate-level netlist to be mapped. The bigger the circuit is, the more time the P&R process will take. This aligns correctly with the obtained results.

Overall, the execution speed is much faster in Ubuntu OS. However, for that case, the memory consumption is bigger than in Windows OS. The tradeoff makes sense from a computational point of view.

4.4.5. P&R for circuit personalization

Results for circuit personalization can be considered a subset of previous results in which P&R is applied to every individual circuit with its own KGO. Thus, both time and memory results are still valid for a given circuit.

[22] Another larger IGA structure was formerly designed to deal with that circuit but it was not ported to later runs.

4.4.6. IGA vs Standard Cell comparison

The work presented in [290] follows a more conventional[23] approach. It performs the physical synthesis of the benchmarks proposed herein, but following a Standard Cell methodology using Cadence tools (Encounter/Virtuoso). For that purpose, the tech files (LEF) and library of Standard Cells (GDS) were designed. Also other files, netlists, scripts, etc. were adapted to Cadence flows.

This work also targeted CPI/Neudrive's fabrication technology. Therefore, direct comparisons with the IGA can be done, mainly in terms of area, and hopefully, in the short future, in terms of performance as well.

The IGA from Figure 26, which was selected as proof of concept for our P&R, has a fixed total size of 21.5 x 6.5 (mm^2). The transistor density is 54 OTFTs / 1.3975 cm^2 ≈ 38. OTFTs have a feature size of 4 µm. However, as shown in [82], the current size of the substrates is up to 20 x 20 (cm^2) which allows for much bigger IGA sizes. And consequently bigger circuits can be mapped onto them. In any case, our P&R is scalable so it can handle the changes in IGA and circuit sizes.

On the other hand, due to the nature of Standard Cell design style, each circuit has its own size, depicted in Table 23.

Circuit	Area (μm^2)	Transistors / cm^2
Multiplexer	2360 x 2326 = 5489360	11 / 0.05489 ≈ 200
Demultiplexer	2545 x 2292 = 5833140	12 / 0.05833 ≈ 205
C17	3176 x 2462 = 7819312	18 / 0.07819 ≈ 230
Decoder	3074 x 2972 = 9135928	24 / 0.09135 ≈ 262
Encoder	3312 x 3788 = 12545856	36 / 0.12545 ≈ 286

Table 23. Benchmark area based on Standard Cell design.

The IGA-based circuits have a bigger size than their Standard Cell-based counterparts, since they are mapped onto the fixed IGA templates. However, the IGA design style provides more design flexibility, thanks to the personalization possibilities using digital printing techniques, along with the lower substrate costs between circuits. In addition, the IGA design choice allows to tackle the fault tolerance (i.e. yield and fault distribution) issue, as I have explained throughout this dissertation.

[23] Covering most of the current industrial practices and design flows.

4.5. Summary of the chapter

In this chapter we have reviewed the KGO map as the main constraint for personalizing the IGA templates. We have seen how logic synthesis is of great importance in order to minimize the circuits and reduce the required number of OTFTs to implement logic functions. We have reviewed/defined specific characteristics/considerations regarding our IGA approach and, based on them, proposed a deterministic constructive P&R heuristic as a main development in this PhD. Details were given about the P&R tool implementation, in order to validate the ideas and claims given throughout this research work. Time complexity analysis was also explained for the proposed implementation, according to the main parameters and worst case scenarios that affect it. A set of small size combinational benchmarks was proposed. The results of significant executions were depicted and explained. And finally, a comparison with the Standard Cell design methodology was presented, concluding that, even if the IGA choice might require more area, it is a better option for the deployment of PE systems, given the inherent characteristics of such technologies.

5. Conclusions and future work

5.1. Overview and conclusions

There are no physical synthesis EDA tools that specifically target Printed/Organic Electronics technologies, at placement and routing level, with their particular characteristics and issues. By extension, none of the existing methodologies addresses the yield and variability issues as we did. This work intends to reduce design efforts by increasing the automation for mapping circuits over PE foils, considering (1) their possible yields and fault distributions and/or (2) circuit individual personalization. The ideas and solutions presented throughout this dissertation not only can be applied for different PE technologies, but even for different substrates from the same foundry and fabrication run, thanks to the use of digital printing techniques.

In this dissertation, I have presented a strategy and the corresponding algorithm for successfully performing Placement and Routing over PE Inkjet-Configurable Gate Array designs, and therefore obtain functional circuit designs. The P&R procedure takes advantage of the 2-step fabrication process inherent to current PE developments. In such scheme, first the Inkjet Gate Array templates are fabricated. Afterwards they are customized by mapping circuits onto them, using digital printing. In between the mentioned steps, it is necessary to add a characterization stage, whenever a technology is expected to have a yield lower than the ideal 100%.

The characterization process provides a KGO map, determining what OTFTs can be used for P&R according to specific criteria. This yield-awareness is a key differentiating component. The IGA methodology is used to maximize the yield for the circuit mapping process, whenever necessary depending on the technology addressed; something that the conventional Standard Cell methodology cannot provide efficiently. Nevertheless, whenever PE technologies are foretold to achieve a near perfect yield, the characterization step can be omitted (along with the corresponding characterization pads, which will increase integration), and the KGO map constraint can be safely ignored or substituted by design for testability + 100% fault coverage and exhaustive samples test to discard the bad ones. In such scenario, our IGA approach and P&R solution remain a good choice for manufacturing PE circuits, since they still provide more flexibility than conventional designs based on Standard-Cells, thanks to the fact that it can allow individual circuit personalization (i.e. for identification codes). However, as a drawback, the IGA-based circuits require more substrate area than their Standard Cell-based counterparts, which present a higher degree of compaction.

During this period we have demonstrated the convenience of the IGA approach, covering a full design flow, from HDL specification to final designs, having completed tape-out successfully, and fabricated prototypes in the available foundries. We used the presented tools at each of the design stages, including logic synthesis and technology mapping to PMOS-only libraries.

The P&R strategy and heuristic presented herein constitutes the main novelty and contribution to our particular field. It follows a conjoint constructive greedy methodology, starting with the Basic Bulk Cell on the bottom left corner of the foil and continuing with a row-based growth, selecting and placing gate after gate, according to the selection criteria, and depending on the KGO map, until the netlist is completely processed. The selection criteria tries to minimize all the possible wiring needed, and the corresponding delays caused by their length (e.g. intra-gate, from/to I/O, and so on). The P&R process performs local and global routing depending on the configuration of the BBCs and wire cells. It is similar to the LEA algorithm but more limited, because for some IGA designs (e.g. Neudrive) some of the tracks intended for trunks are connected by default, which disallows the LEA heuristic from the literature. An example of an IGA that could permit that algorithm would be the CEA one, since in this case none of such tracks are connected by default, allowing for more flexibility and routing optimizations (such as doglegs). However, due to this higher number of discontinuous tracks, the yield at interconnections is more prone to errors, which might severely compromise circuit functionality. In any case, for the strategy to be technology independent, the more restrictive version was the one applied, so that it was able to work regardless of the targeted PE process.

Results have been presented and compared over different platforms, including performance measurements. Comparisons with other tools/methodologies are not possible due to the novelty of our approach and the undeniable underlying differences, which have been extensively explained throughout this dissertation.

With our approach we achieve our main goal of mapping circuits over the IGA templates while maximizing the yield at circuit-level, out of foils with variable yield, taking into account the KGO distribution and BBC topologies. The proposed solution is scalable to bigger IGA templates and circuits.

5.2. Future work

Traditionally logic synthesis and physical synthesis have been separated. Such approach is believed to be not optimal because of the timing closure predicament. In current advances in CMOS technology nodes the interconnection delays are more important than cell delays. To my knowledge, most (if not all) PE technologies lack accurate parasitic models (including wire load capacitance and via resistance models) for proper estimation of interconnect delays in synthesis. There is still much work to be done in combining both steps of the design flow, by managing the feedback provided by such processes and iterating on their results over the corresponding information libraries. This work does not necessarily have to target PE technologies. The applications of such approach are technology independent, and they are especially relevant when scaling to the nanometer scale.

Since PE/OE is a field where “open technologies”[24] are still evolving, model analysis and development are a continuous area of work, according to the empirical data obtained through characterization and test procedures. Semiconductor, dielectric and conductive base materials, and also ink formulations and recipes are, slowly but steadily, getting better. This evolution should lead to better performance, reliability and longevity. Technological advances matter from the EDA point of view. Better models imply more reliable electrical simulations, which are key to confirm that the designs meet the constraints and are reliable enough to send to the foundry for fabrication; receiving back working circuits that perform as expected.

From a P&R perspective my algorithm depends on the yield and fault distribution at transistor level, according to our IGA strategy. However, it does not contemplate the yield on the interconnections, and this is important on PE technologies relying on digital printing. Up to now, the way to maximize yield at interconnects is targeting the best printing technologies available for the customization step, and making design decisions on how to make the wire cells of the IGAs more reliable for each addressed technology (i.e. Design For Manufacturability). The most important factor in this case would be the design rules allowed by each foundry. Parameterizing most (if not all) of the cells/designs would be ideal, but it is time consuming. And it is not efficient when technologies and design rules change rapidly between fabrication runs (or nodes). However, the benefits of parameterization might be worth the efforts, provided that there are enough implementation resources. Such advantages include, but are not limited to, speeding-up design processes for both novice and expert engineers, providing a more reliable framework, and creating additional useful information for later processing.

The idea of making an incremental P&R might be interesting as well. It would consist in saving the state of the P&R progress, so that it can be resumed/reused later, whenever convenient (e.g. minimum changes between fault distributions or circuit netlists). This can be done by generating/handling ID codes and Cyclic Redundancy Checks (CRC) polynomial coefficients, each time the P&R process is executed.

Beyond the scope of this work, other general trends nowadays which have great potential for the semiconductor industry (including research) are cloud-based tools and Machine/Deep Learning methodologies. Currently more EDA tools are exploring the possibilities that the cloud offers, following the Software as a Service (SaaS) business model. There is much work to be done in this field, especially in the optimization of the huge amount of data to be transferred, standardization, and overall security (server-client). In the case of Machine Learning, relevant practices and findings can be incorporated in the design flows, and even into the EDA tools themselves, targeting several complex EDA problems. However, learning to make reliable predictions effectively requires an enormous amount of iterations consuming plenty of hardware resources (CPUs + GPUs).

[24] Those accessible by third parties.

Bibliography

[1] G. Moore, "Cramming more Components onto Integrated Circuits (reprinted)," *Proceedings of the IEEE,* vol. 86, no. 1, pp. 82-85, 1998.

[2] W. Arden, M. Brillouët, P. Cogez, M. Graef, B. Huizing and R. Mahnkopf, "“More-than-Moore” White Paper," ITRS, 2010.

[3] S. I. Associations, "International Technology Roadmap for Semiconductors," 2015.

[4] S. I. Associations, "International Roadmap for Devices and Systems," 2016.

[5] OE-A, "Organic Electronics Association," [Online]. Available: http://www.oe-a.org/.

[6] D. Nenni and P. McLellan, Fabless: The Transformation of the Semiconductor Industry, Ed. SemiWiki.com, 2013.

[7] V. G. Oklobdzija, Digital Design and Fabrication, CRC Press, 2008.

[8] G. Qu and M. Potkonjak, Intellectual Property Protection in VLSI Designs. Theory and Practice, Ed. Kluwer Academic Publishers, 2003.

[9] M. A. Bader, Intellectual Property Management in R&D Collaborations, Physica-Verlag, 2006.

[10] K. Hishida, Fulfilling the Promise of Technology Transfer. Fostering Innovation for the Benefit of Society, Springer, 2013.

[11] J. Ferreira, M. Raposo, R. Rutten and A. Varga, Cooperation, Clusters, and Knowledge Transfer. Universities and Firms Towards Regional Competitiveness, Springer, 2013.

[12] UAB and IMB-CNM-CSIC, "Technology & Design Kits for Printed Electronics (TDK4PE)," European project. Ref: FP7-ICT-2011-7, 287682, 2011.

[13] UAB and IMB-CNM-CSIC, "Application Specific Printed Electronics Circuits - Technology and Design Kit Development (ASPEC-TDK)," Spanish project. Ref: TEC2011-29800-C03, 2011.

[14] UAB and IMB-CNM-CSIC, "Flexible microelectronics for X-Ray imaging (MEF3-IRX)," Spanish project. Ref: TEC2014-59679-C2, 2014.

[15] "Printed Electronics Cluster (PEC4)," [Online]. Available: https://sites.google.com/a/pec4.net/home/.

[16] "Centre for Process Innovation (CPI)," [Online]. Available: https://www.uk-cpi.com/.

[17] "Neudrive," [Online]. Available: http://www.neudrive.com/.

[18] "Commissariat à l'énergie atomique et aux énergies alternatives (CEA)," [Online]. Available: http://www.cea.fr/.

[19] "Centre Suisse d'Electronique et de Microtechnique (CSEM)," [Online]. Available: https://www.csem.ch/.

[20] "Agenzia nazionale per le nuove tecnologie, l'energia e lo sviluppo economico sostenibile (ENEA)," [Online]. Available: http://www.enea.it/.

[21] "Commercialisation of Organic and Large Area Electronics (COLAE)," [Online]. Available: http://www.colae.eu/.

[22] "CETEMMSA (Eurecat)," [Online]. Available: https://eurecat.org/.

[23] "Flexink," [Online]. Available: http://www.flexink.co.uk/.

[24] "Infiniscale (Silvaco)," [Online]. Available: http://www.silvaco.com/products/variation_analysis/variation_analysis.html.

[25] "Sensing Tex," [Online]. Available: http://sensingtex.com/.

[26] "3D Micromac," [Online]. Available: http://3d-micromac.com/.

[27] "PhoeniX Software," [Online]. Available: http://www.phoenixbv.com/.

[28] "Universitat Rovira i Virgili (URV)," [Online]. Available: http://www.urv.cat/.

[29] "Universidade Federal do Rio Grande do Sul (UFRGS)," [Online]. Available: http://www.ufrgs.br/.

[30] "Technische Universität Chemnitz (TUC)," [Online]. Available: https://www.tu-chemnitz.de/.

[31] "Universidade do Algarve (UAlg)," [Online]. Available: https://www.ualg.pt/.

[32] P. McLellan, EDAgraffiti, Ed. SemiWiki.com, 2010.

[33] L.-T. Wang, Y.-W. Chang and K.-T. Cheng, Electronic Design Automation, Morgan Kaufmann (Elsevier), 2009.

[34] L. Scheffer, L. Lavagno and G. Martin, EDA for IC Implementation, Circuit Design, and Process Technology, Taylor & Francis Group, 2006.

[35] J. E. Ayers, Digital Integrated Circuits: Analysis and Design, CRC Press, 2005.

[36] G. d. Micheli, Synthesis and Optimization of Digital Circuits, Ed. McGraw-Hill, 1994.

[37] H. Eriksson, P. Larsson–Edefors, T. Henriksson and C. and Svensson, "Full-custom vs. standard-cell design flow - an adder case study," in *Asia S. Pac. Design Automation Conf.*, 2003.

[38] D. Chinnery and K. Keutzer, "Closing the Gap Between ASIC and Custom: An ASIC Perspective," in *Design Automation Conference*, 2000.

[39] R. J. Baker, CMOS Circuit Design, Layout, and Simulation, John Wiley & Sons, 2010.

[40] F. Vahid, Digital Design with RTL Design, Verilog and VHDL, John Wiley and Sons, 2011.

[41] IEC/IEEE, "Standard Hardware Description Languages," IEEE, [Online]. Available: http://ieeexplore.ieee.org/browse/standards/collection/ieee.

[42] S. Brown and Z. Vranesic, Fundamentals of Digital Logic with VHDL Design, McGraw-Hill, 2009.

[43] S. Brown and Z. Vranesic, Fundamentals of Digital Logic with Verilog Design, McGraw-Hill, 2009.

[44] W. Kafig, VHDL 101: Everything you need to know to get started, Newnes Newton, 2011.

[45] A. Rushton, VHDL for Logic Synthesis, John Wiley & Sons, 2011.

[46] L. Bening and H. Foster, Principles of Verifiable RTL Design. A functional coding style supporting verification processes in Verilog, Ed. Kluwer Academic Publishers, 2002.

[47] V. Sagdeo, The Complete Verilog Book, Ed. Kluwer Academic Publishers, 2002.

[48] D. E. Thomas and P. R. Moorby, The Verilog Hardware Description Language, Ed. Kluwer Academic Publishers, 2002.

[49] S. Chonnad and N. Balachander, Verilog: Frequently Asked Questions. Language, Applications and Extensions, Ed. Springer, 2004.

[50] J. M. Lee, Verilog Quickstart, A Practical Guide to Simulation and Synthesis in Verilog, Kluwer Academic Publishers, 2002.

[51] S. Sutherland, The Verilog PLI Handbook, Kluwer Academic Publishers, 2002.

[52] D. J. Smith, "VHDL & Verilog Compared & Contrasted - Plus Modeled Example Written in VHDL, Verilog and C," in *Design Automation Conference*, 1996.

[53] S. Bailey, "Comparison of VHDL, Verilog and SystemVerilog (white paper)," Model Technology, 2003.

[54] R. B. Hitchcock, "Timing verification and the timing analysis program," in *Design Automation Conference*, 1982.

[55] J. Bhasker and R. Chadha, Static Timing Analysis for Nanometer Designs. A Practical Approach., Springer, 2009.

[56] A. Kuehlmann, The best of ICCAD. 20 Years of Excellence in Computer-Aided Design, Springer, 2003.

[57] J. Mujal, E. Ramon and J. Carrabina, "Methodology and tools for inkjet process abstraction for the design of flexible and organic electronics," *International Journal of High Speed Electronics and Systems,* vol. 20, no. 4, pp. 829-842, 2011.

[58] M. Llamas, M. Mashayekhi, J. Pallarès, F. Vila, L. Terés and J. Carrabina, "Top-down Design Flow for Application Specific Printed Electronics Circuits (ASPECs)," in *XXIX Conference on Design of Circuits and Integrated Systems - DCIS 2014*, 2014.

[59] M. Mashayekhi, M. Llamas, J. Pallarès, F. Vila, L. Terés and J. Carrabina, "Development of a Standard Cell Library and ASPEC design flow for organic Thin

Film Transistor Technology," in *XXIX Conference on Design of Circuits and Integrated Systems - DCIS*, 2014.

[60] J. Carrabina, M. Mashayekhi, M. Llamas, C. Martínez-Domingo, E. Ramon, A. Alcalde, J. Pallarès, F. Vila, A. Conde and L. Terés, "Inkjet Gate Array: Novel concept to implement electronic systems," in *6th International Exhibition and Conference for the Printed Electronics Industry - LOPEC*, 2014.

[61] M. Mashayekhi, M. Llamas, J. Pallarès, F. Vila, L. Terés and J. Carrabina, "Fault-Tolerant Inkjet Gate Array for Application Specific Printed Electronics Circuits," in *6th International Conference on Computer Aided Design for Thin-Film Transistors - CAD-TFT*, 2014.

[62] M. Llamas, M. Mashayekhi, A. Alcalde, J. Pallarès, F. Vila, A. Conde, L. Terés and J. Carrabina, "Development of Digital Application Specific Printed Electronics Circuits: From Specification to Final Prototypes," *IEEE/OSA Journal of Display Technology,* vol. 11, no. 8, pp. 652-657, 2015.

[63] "Gschem," gEDA, [Online]. Available: http://www.geda-project.org/.

[64] S. Williams, "Icarus Verilog," [Online]. Available: http://iverilog.icarus.com/.

[65] "GTKWave," [Online]. Available: http://gtkwave.sourceforge.net/.

[66] A. Mishchenko, "ABC: A System for Sequential Synthesis and Verification," Berkeley Logic Synthesis and Verification Group, [Online]. Available: http://www.eecs.berkeley.edu/~alanmi/abc/.

[67] Q. Wu, J. Zhang and Q. Qiu, "Design Considerations for Digital Circuits Using Organic Thin Film Transistors on a Flexible Substrate," in *IEEE Int. Symp. Circuits Syst.*, 2006.

[68] B. Zamanlooy, A. Ayatollahi, S. M. Fakhraie and M. Chahardori, "Investigating Different Circuit Styles for Digital Circuits Using Organic Transistors," in *International Symposium on Integrated Circuits (ISIC)*, 2007.

[69] J. P. Uyemura, CMOS Logic Circuit Design, Ed. Kluwer Academic Publishers, 2002.

[70] F. Moll and M. Roca, Interconnection Noise in VLSI Circuits, Kluwer Academic Publishers, 2004.

[71] K. Sabine, "Glade IC Layout Editor," Peardrop Design Systems, [Online]. Available: http://www.peardrop.co.uk/.

[72] M. Köfferlein, "KLayout IC Layout Editor," [Online]. Available: https://www.klayout.de/.

[73] C. Ebeling, "Gemini," University of Washington, [Online]. Available: http://www.cs.washington.edu/node/2177/.

[74] "Simulation Program for Integrated Circuits Emphasis (SPICE)," University of California, Berkeley, [Online]. Available: http://bwrcs.eecs.berkeley.edu/Classes/IcBook/SPICE/.

[75] "Ngspice," gEDA, [Online]. Available: http://ngspice.sourceforge.net/.

[76] T. Ytterdal, "AIM-Spice," AIM-Software, [Online]. Available: http://www.aimspice.com/.

[77] M. Mashayekhi, M. Llamas, J. Pallarès, F. Vila, L. Terés and J. Carrabina, "Fault-Tolerant Inkjet Gate Array for Application Specific Printed Electronics Circuits," in *6th International Conference on Computer Aided Design for Thin-Film Transistors, CAD-TFT*, 2014.

[78] J. Carrabina, M. Mashayekhi, M. Llamas, S. Ogier, T. Pease, M. Matti, L. Mika-Matti and L. Terés, "Customization Technology and Tools for Building Printed Circuits," in *6th International Conference on Flexible and Printed Electronics - ICFPE 2015*, 2015.

[79] J. Carrabina, M. Mashayekhi, M. Llamas, J. Pallarès, J. Matos, A. Reis and L. Terés, "Ink-Jet Configurable Gate Arrays," in *International Workshop on Flexible Electronics - WFE 2016*, 2016.

[80] M. Mashayekhi, L. Winchester, L. Evans, T. Pease, M.-M. Laurila, M. Mäntysalo, S. Ogier, L. Terés and J. Carrabina, "Evaluation of Aerosol, Superfine Inkjet, and Photolithography Printing Techniques for Metallization of Application Specific Printed Electronic Circuits," *IEEE Transactions on Electron Devices,* vol. 63, no. 3, pp. 1246-1253, 2016.

[81] J. Carrabina, M. Mashayekhi, J. Pallarès and L. Terés, "Inkjet-Configurable Gate Arrays (IGA)," *IEEE Transactions on Emerging Topics in Computing,* vol. 5, no. 2, pp. 238 - 246, 2016.

[82] M. Mashayekhi, "Inkjet-Configurable Gate Arrays. Towards Application Specific Printed Electronic Circuits," Ph.D. Thesis. Microelectronics and Electronic System Department (MiSE). Universitat Autònoma de Barcelona (UAB), 2016.

[83] I. M. Hutchings and G. D. Martin, Inkjet technology for digital fabrication, John Wiley & Sons, 2013.

[84] S. M. Rubin, Computer Aids for VLSI Design, Ed. Addison-Wesley, 2009.

[85] C. Company, "GDSII Stream Format Manual. v 6.0," Ed. Calma Company, 1987.

[86] C. D. Systems, "Design Data Translator's Reference. v 5.1.41," Ed. Cadence Design Systems, 2007.

[87] E. Sowade, E. Ramon, K. Y. Mitra, C. Martínez-Domingo, M. Pedró, J. Pallarès, F. Loffredo, F. Villani, H. L. Gomes, L. Terés and R. R. Baumann, "All-inkjet-printed thin-film transistors: manufacturing process reliability by root cause analysis," *Scientific Reports,* vol. 6, no. 33490, 2016.

[88] S. W. Director, W. Maly and A. J. Strojwas, VLSI Design for Manufacturing: Yield Enhancement, Ed. Kluwer Academic Publishers, 1990.

[89] G. A. T. Sevilla and M. M. Hussain, "Printed Organic and Inorganic Electronics: Devices To Systems," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems,* vol. PP, no. 99, pp. 1-14, 2016.

[90] E. Ramon, "Inkjet Printed Microelectronic Devices and Circuits," Ph.D. Thesis. Microelectronics and Electronic System Department (MiSE). Universitat Autònoma de Barcelona (UAB), 2014.

[91] R. Janssen and M. Gratzel, "Themed issue: solar cells," *Journal of Materials Chemistry,* vol. 19, no. 30, p. 5276, 2009.

[92] V. Shrotriya, "Organic photovoltaics: Polymer power," *Nature Photonics,* vol. 3, no. 8, pp. 447-449, 2009.

[93] M. Hilder, B. Winther-Jensen and N. B. Clark, "Paper-based, printed zinc–air battery," *Journal of Power Sources,* vol. 194, no. 2, pp. 1135-1141, 2009.

[94] C. C. Ho, K. Murata, D. a. Steingart, J. W. Evans and P. K. Wright, "A super ink jet printed zinc–silver 3D microbattery," *Journal of Micromechanics and Microengineering,* vol. 19, no. 9, p. 4013, 2009.

[95] Y. Yoshioka and G. E. Jabbour, "Desktop inkjet printer as a tool to print conducting polymers," *Synthetic Metals,* vol. 156, no. 11, p. 779–783, 2006.

[96] S.-C. Chang, J. Liu, J. Bharathan, Y. Yang, J. Onohara and J. Kido, "Multicolor Organic Light-Emitting Diodes Processed by Hybrid Inkjet Printing," *Advanced Materials,* vol. 11, no. 9, pp. 734-737, 199.

[97] A. Argun, P.-H. Aubert, B. C. Thompson, I. Schwendeman, C. L. Gaupp, J. Hwang, N. J. Pinto, D. B. Tanner, A. G. MacDiarmid and J. R. Reynolds, "Multicolored Electrochromism in Polymers: Structures and Devices," *Chemistry of Materials,* vol. 16, no. 23, p. 4401–4412, 2004.

[98] "FlexEnable," [Online]. Available: http://www.flexenable.com/.

[99] "Plastic Logic," [Online]. Available: http://www.plasticlogic.com/.

[100] "Samsung Electronics," [Online]. Available: http://www.samsung.com/.

[101] T. Sekitani, H. Nakajima, H. Maeda, T. Fukushima, T. Aida, K. Hata and T. Someya, "Stretchable active-matrix organic light-emitting diode display using printable elastic conductors," *Nature Materials,* vol. 8, no. 6, pp. 494-9, 2009.

[102] S. Gamerith, A. Klug, H. Scheiber, U. Scherf, E. Moderegger and E. J. W. List, "Direct Ink-Jet Printing of Ag–Cu Nanoparticle and Ag-Precursor Based Electrodes for OFET Applications," *Advanced Functional Materials,* vol. 17, no. 16, p. 3111–3118, 2007.

[103] D. Kim, S. Jeong, S. Lee, B. K. Park and J. Moon, "Organic thin film transistor using silver electrodes by the ink-jet printing technology," *Thin Solid Films,* vol. 515, no. 19, p. 7692–7696, 2007.

[104] S. H. Ko, J. Chung, H. Pan, C. P. Grigoropoulos and D. Poulikakos, "Fabrication of multilayer passive and active electric components on polymer using inkjet printing and low temperature laser processing," *Sensors and Actuators A: Physical,* vol. 134, no. 1, p. 161–168, 2007.

[105] K. Song, D. Kim, X.-S. Li, T. Jun, Y. Jeong and J. Moon, "Solution processed invisible all-oxide thin film transistors," *Journal of Materials Chemistry,* vol. 19, no. 46, p. 8881, 2009.

[106] A. E. Amraoui, M. Bocquet, F. Barros, J.-M. Portal, M. Charbonneau, S. Jacob, J. Bablet, M. Benwadih, V. Fischer, R. Coppard and R. Gwoziecky, "Printed Complementary Organic Thin Film Transistors based decoder for ferroelectric memory," in *ESSCIRC*, 2014.

[107] V. R. Marinov, Y. Atanasov, A. Khan, D. Vaselaar, A. Halvorsen, D. L. Schulz and D. B. Chrisey, "Direct-Write Vapor Sensors on FR4 Plastic Substrates," *IEEE Sensors Journal,* vol. 7, no. 6, p. 937–944, 2007.

[108] F. Loffredo, G. Burrasca, L. Quercia and D. D. Sala, "Gas Sensor Devices Obtained by Ink-jet Printing of Polyaniline Suspensions," *Macromolecular Symposia,* vol. 247, no. 1, p. 357–363, 2007.

[109] K. Chang, Y. H. Kim, Y. J. Kim and Y. J. Yoon, "Functional antenna integrated with relative humidity sensor using synthesised polyimide for passive RFID sensing," *Electronics Letters,* vol. 43, no. 5, p. 259, 2007.

[110] R. Möller, T. Schüler, S. Günther, M. R. Carlsohn, T. Munder and W. Fritzsche, "Electrical DNA-chip-based identification of different species of the genus Kitasatospora," *Applied Microbiology and Biotechnology,* vol. 77, no. 5, p. 1181–8, 2008.

[111] J. B. Chang, V. Liu, V. Subramanian, K. Sivula, C. Luscombe, A. Murphy, J. Liu and J. M. J. Fréchet, "Printable polythiophene gas sensor array for low-cost electronic noses," *Journal of Applied Physics,* vol. 100, no. 1, p. 14506, 2006.

[112] A. Aliane, V. Fischer, T. Card, R. Coppard and I. Chartier, "Large area printed temperature sensors on flexible substrate," in *5th IEEE International Workshop on Advances in Sensors and Interfaces - IWASI*, 2013.

[113] I. Chartier, S. Jacob, M. Charbonneau, A. Aliane, A. Plihon, R. Coppard, R. Gwoziecki, J. Verilhac, C. Serbutoviez, E. Cantatore, O. Dhez and F. D. D. Santos, "Printed OTFT complementary circuits and matrix for Smart Sensing Surfaces applications," in *ESSDERC*, 2014.

[114] S. B. Fuller, E. J. Wilhelm and J. M. Jacobson, "Ink-jet printed nanoparticle microelectromechanical systems," *Journal of Microelectromechanical Systems,* vol. 11, no. 1, p. 54–60, 2002.

[115] V. Subramanian, J. M. J. Frechet, P. C. Chang, D. C. Huang, J. B. Lee, S. E. Molesa, A. R. Murphy, D. R. Redinger and S. K. Volkman, "Progress Toward Development of All-Printed RFID Tags: Materials, Processes, and Devices," *Proceedings of the IEEE,* vol. 93, no. 7, p. 1330–1338, 2005.

[116] V. Fiore, P. Battiato, S. Abdinia, S. Jacobs, I. Chartier, R. Coppard, G. Klink, E. Cantatore, E. Ragonese and G. Palmisano, "An Integrated 13.56-MHz RFID Tag in a Printed Organic Complementary TFT Technology on Flexible Substrate," *IEEE Transactions on Circuits and Systems,* vol. 62, no. 6, pp. 1668-1677, 2015.

[117] E. Bènevent, E. Bergeret, M. Egels, P. Pannier, A. Aliane, A. Daami and R. Coppard, "Full-printed inductors on flexible plastic foils for electromagnetic energy harvesting," in *42nd European Microwave Conference*, 2012.

[118] M. Guerin, E. Bergeret, E. Bènevent, P. Pannier, A. Daami, S. Jacob, I. Chartier and R. Coppard, "Design of organic complementary circuits for RFID tags application," in *Custom Integrated Circuits Conference (CICC)*, 2012.

[119] K. Myny, E. v. Veenendaal, G. H. Gelinck, J. Genoe, W. Dehaene and P. Heremans, "An 8b Organic Microprocessor on Plastic Foil," in *ISSCC*, 2011.

[120] K. Myny, E. v. Veenendaal, G. H. Gelinck, J. Genoe, W. Dehaene and P. Heremans, "An 8-Bit, 40-Instructions-Per-Second Organic Microprocessor on Plastic Foil," *IEEE Journal of Solid-State Circuits,* vol. 47, no. 1, pp. 284-291, 2012.

[121] J. Matos, M. Llamas, M.Mashayekhi, J. Carrabina and A. Reis, "Optimization on Cell-library Design for Digital Application Specific Printed Electronics Circuits," in *24th International Workshop on Power and Timing Modeling, Optimization and Simulation - PATMOS*, 2014.

[122] S. Abdinia, M. Benwadih, R. Coppard, S. Jacob, G. Maiellaro, G. Palmisano, M. Rizzo, A. Scuderi, F. Tramontana4, A. v. Roermund1 and E. Cantatore, "A 4b ADC Manufactured in a Fully-Printed Organic Complementary Technology Including Resistors," in *IEEE International Solid-State Circuits Conference*, 2013.

[123] S. Abdinia, M. Benwadih, E. Cantatore, I. Chartier, S. Jacob, L. Maddiona, G. Maiellaro, L. Mariucci, G. Palmisano, M. Rapisarda, F. Tramontana and A. v. Roermund, "Design of analog and digital building blocks in a fully printed complementary organic technology," in *38th European Solid-State Circuit conference - ESSCIRC*, 2012.

[124] R. Coppard, S. Jacob, M. Charbonneau, M. Benwadih, J. Bablet, V. Fischer, R. Gwoziecki, I. Chartier, S. Abdinia, E.Cantatore, L. Maddionac, G. Maiellarod, L. Mariuccie, M. Rapisardae and F. Tramontana, "Printed Organic TFTs on Flexible Substrate for Complementary Circuits," *ECS Transactions,* vol. 54, no. 1, pp. 153-163, 2013.

[125] M. Guerin, E. Bergeret, E. Bènevent, A. Daami, P. Pannier and R. Coppard, "Organic Complementary Logic Circuits and Volatile Memories Integrated on Plastic Foils," *IEEE Transactions on Electron Devices,* vol. 60, no. 6, pp. 2045-2051, 2013.

[126] J. Chang, X. Zhang, T. Ge and J. Zhou, "Fully printed electronics on flexible substrates: High gain amplifiers and DAC," *Organic Electronics,* vol. 15, no. 3, pp. 701-710, 2014.

[127] A. Sou, S. Jung, E. Gili, V. Pecunia, J. Joimel, G. Fichet and H. Sirringhaus, "Programmable logic circuits for functional integrated smart plastic systems," *Organic Electronics,* vol. 15, no. 11, p. 3111–3119, 2014.

[128] F. Vila, "From Characterization Strategies to PDK & EDA Tools for Printed Electronics," Ph.D. Thesis. Microelectronics and Electronic System Department (MiSE). Universitat Autònoma de Barcelona (UAB), 2015.

[129] F. Zanella, "Organic thin-film transistors: from technologies to circuits," Ph.D. Thesis. École Polytechnique Fédérale de Lausanne (EPFL), 2014.

[130] M. A. Torres-Miranda, "Design of Analog & Digital Circuits using Flexible Organic Electronics," Ph.D. Thesis. Université Pierre et Marie Curie (UPMC), 2015.

[131] T.-C. Huang, J.-L. Huang and K.-T. Cheng, "Design, Automation, and Test for Low-Power and Reliable Flexible Electronics," *Foundations and Trends in Electronic Design Automation,* vol. 9, no. 2, pp. 99-210, 2015.

[132] M. J. S. Smith, Application-Specific Integrated Circuits, Addison-Wesley, 1997.

[133] H. Bhatnagar, Advanced ASIC Chip Synthesis; Using Synopsys Design Compiler, Physical Compiler and PrimeTime, Kluwer Academic Publishers, 2002.

[134] M. Llamas, M. Mashayekhi, J. Pallarès, F. Vila, L. Terés and J. Carrabina, "A novel design flow for Application Specific Printed Electronics Circuits," in *6th International Conference on Computer Aided Design for Thin-Film Transistors - CAD-TFT*, 2014.

[135] J. Carrabina, M. Llamas, J. Matos, M. Mashayekhi and A. Reis, "Technology Mapping Tools for Building Optimal Circuits," in *7th International Exhibition and Conference for the Printed Electronics Industry - LOPEC*, 2015.

[136] J. Carrabina, J. Pallarès, F. Vila, M. Llamas, M. Mashayekhi, J. Matos, A. Reis and L. Terés, "Process Design Kit and EDA Tools for Organic/Printed Electronics," in *7th International Conference on Computer Aided Design for Thin-Film Transistors - CAD-TFT*, 2016.

[137] F. Vila, J. Pallarès, L. Terés, J. Carrabina and K. Sabine, "A complete set of open/free EDA tools for PE physical design," in *Design Automation & Test in Europe - DATE*, 2013.

[138] F. Vila, J. Pallarès, A. Conde and L. Terés, "A fully-automated methodology and system for printed electronics foil characterization," in *International Conference on Microelectronic Test Structures - ICMTS*, 2015.

[139] E. Ramon, C. Martínez-Domingo and J. Carrabina, "Geometric Design and Compensation Rules Generation and Characterization for All-Inkjet-Printed Organic Thin Film Transistors," *Journal of Imaging Science & Technology,* vol. 57, no. 4, pp. 40402-1-40402-12, 2013.

[140] F. Vila, B. Labarías, J. Pallarès, L. Terés, A. Bakkerz, N. Olij, R. Stoffer and J. Carrabina, "PDK development for Printed/Organic Electronics based on XML," in *XVIII Conference on the Design of Circuits and Integrated Systems - DCIS*, 2013.

[141] E. Ramon, C. Martínez-Domingo, A. Alcalde-Aragonés and J. Carrabina, "Development of a Simple Manufacturing Process for All-Inkjet Printed Organic Thin Film Transistors and Circuits," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems,* vol. 7, no. 1, pp. 161-170, 2016.

[142] D. Gamota, P. Brazis, K. Kalyanasundaram and J. Zhang, Printed Organic and Molecular Electronics, Springer, 2004.

[143] E. Cantatore, Applications of Organic and Printed Electronics. A Technology-Enabled Revolution., Springer, 2013.

[144] K. Suganuma, Introduction to Printed Electronics, Springer, 2014.

[145] Cadence, "https://www.cadence.com/," [Online].

[146] J. Ousterhout, "Tcl," [Online]. Available: https://www.tcl.tk/.

[147] Synopsys, "https://www.synopsys.com/," [Online].

[148] "PyCell Studio," Synopsys, [Online]. Available: https://www.synopsys.com/cgi-bin/pycellstudio/req1.cgi.

[149] S. I. I. (Si2), "http://www.openeda.org/openaccess/," [Online].

[150] T. Blanchard, "Assessment of the OpenAccess Standard: insights on the new EDA industry standard from Hewlett-Packard, a Beta partner and contributing developer," in *International Symposium on Quality Electronic Design*, 2003.

[151] S. I. I. (Si2), "http://www.si2.org/," [Online].

[152] M. Graphics, "https://www.mentor.com/," [Online].

[153] T. Edwards, "Qflow," [Online]. Available: http://opencircuitdesign.com/qflow/.

[154] C. Wolf, "Yosys," [Online]. Available: http://www.clifford.at/yosys/.

[155] Y. University, "Graywolf," [Online]. Available: https://github.com/rubund/graywolf.

[156] T. Edwards, "Qrouter," [Online]. Available: http://opencircuitdesign.com/qrouter/.

[157] J. Ousterhout, "Magic," [Online]. Available: http://opencircuitdesign.com/magic/.

[158] U. o. Michigan, "UMpack," [Online]. Available: http://vlsicad.eecs.umich.edu/BK/PDtools/.

[159] U. P. e. M. C. LIP6, "Alliance," [Online]. Available: https://soc-extras.lip6.fr/en/alliance-abstract-en/.

[160] G. v. Rossum, "Python," [Online]. Available: https://www.python.org/.

[161] A. Mark, "gdsCAD," [Online]. Available: https://github.com/hohlraum/gdsCAD.

[162] S. Takamaeda-Yamazaki, "Pyverilog," [Online]. Available: https://github.com/PyHDI/Pyverilog.

[163] S. Takamaeda-Yamazaki, "Pyverilog: A Python-based Hardware Design Processing Toolkit for Verilog HDL," *Applied Reconfigurable Computing (ARC 2015), Lecture Notes in Computer Science,* vol. 9040, pp. 451-460, 2015.

[164] "Symica," [Online]. Available: http://www.symica.com/.

[165] S. Whiteley, "XicTools," [Online]. Available: http://www.wrcad.com/.

[166] "eSim," [Online]. Available: http://esim.fossee.in/.

[167] A. Ziesemer, "Astran," [Online]. Available: http://aziesemer.github.io/astran/.

[168] J. Decaluwe, "MyHDL," [Online]. Available: http://www.myhdl.org/.

[169] J.-P. Charras, "KiCad," [Online]. Available: http://kicad-pcb.org/.

[170] "LayoutEditor," [Online]. Available: http://www.layouteditor.net/.

[171] S. M. Rubin, "Electric," Static Free Software / GNU, [Online]. Available: https://www.gnu.org/software/electric/.

[172] E. Malavasi, E. Charbon, E. Felt and A. Sangiovanni-Vincentelli, "Automation of IC layout with analog constraints," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 15, no. 8, pp. 923-942, 1996.

[173] L. Zhang, U. Kleine and Y. Jiang, "An automated design tool for analog layouts," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* vol. 14, no. 8, pp. 881-894, 2006.

[174] J. Scheible and J. Lienig, "Automation of Analog IC Layout: Challenges and Solutions," in *International Symposium on Physical Design (ISPD)*, 2015.

[175] D. Marolt, J. Scheible, G. Jerke and V. Marolt, "SWARM: A Multi-agent System for Layout Automation in Analog Integrated Circuit Design," in *Agent and Multi-Agent Systems: Technology and Applications. Smart Innovation, Systems and Technologies, 58*, 2016.

[176] "Pulsic Animate," [Online]. Available: https://www.pulsic.com/animate/.

[177] T. Lengauer, Combinatorial Algorithms for Integrated Circuit Layout, John Wiley & Sons, 1990.

[178] N. A. Sherwani, Algorithms for VLSI Physical Design Automation, Ed. Kluwer Academic Publishers, 1999.

[179] S. H. Gerez, Algorithms for VLSI Design Automation, Ed. Wiley, 1999.

[180] S. M. Sait and H. Youssef, VLSI Physical Design Automation. Theory and Practice, Ed. World Scientific, 1999.

[181] C. J. Alpert, D. P. Mehta and S. S. Sapatnekar, Handbook of Algorithms for Physical Design Automation, CRC Press, 2009.

[182] A. B. Kahng, J. Lienig, I. L. Markov and J. Hu, VLSI Physical Design: From Graph Partitioning to Timing Closure, Springer, 2011.

[183] A. M. Turing, "On Computable Numbers, with an Application to the Entscheidungs problem," *Proceedings of the London Mathematical Society,* vol. 42, no. 2, pp. 230-265, 1936.

[184] G. Boolos, J. Burgess and R. Jeffrey, Computability and Logic, Cambridge University Press, 2007.

[185] P. Bachmann, Analytische Zahlentheorie, Bd. 2, Teubner, 1894.

[186] E. Landau, Handbuch der Lehre von der Verteilung der Primzahlen, Teubner, 1909.

[187] S.-K. Chang, Data Structures and Algorithms, World Scientific Publishing, 2003.

[188] S. Meyer, "A data structure for circuit net lists," in *Design Automation Conference*, 1988.

[189] W. E. Donath, "Complexity Theory and Design Automation," in *17th Conference on Design Automation*, 1980.

[190] D. Zhou, "A Parallel Algorithm for Global Routing in Sea-of-gates Technology," in *IEEE International Symposium on Circuits and Systems ISCAS*, 1991.

[191] L. Hyafil and R. L. Rivest, "Graph partitioning and constructing optimal decision trees are polynomial complete problems," Rep. 33. IRIA Laboratoire, 1973.

[192] M. R. Garey and D. S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, 1990.

[193] J. Hromkovič, Algorithmics for Hard Problems. Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics., Springer, 2004.

[194] M. Fleury, "Deux problèmes de Géométrie de situation," *Journal de mathématiques élémentaires,* vol. 2, pp. 257-261, 1883.

[195] J. B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," in *Proceedings of the American Mathematical Society*, 1956.

[196] R. C. Prim, "Shortest connection networks and some generalizations," *Bell System Technical Journal,* vol. 36, no. 6, pp. 1389-1401, 1957.

[197] J. Nešetřil, E. Milková and H. Nešetřilová, "Otakar Borůvka on minimum spanning tree problem. Translation of both the 1926 papers, comments, history," *Discrete Mathematics,* vol. 233, no. 1-3, pp. 3-36, 2001.

[198] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik,* vol. 1, no. 1, pp. 269-271, 1959.

[199] H. J. Prömel and A. Steger, The Steiner Tree Problem. A Tour through Graphs, Algorithms, and Complexity. (Advanced Lectures in Mathematics), Springer.

[200] W. D. Wallis, A Beginner's Guide to Graph Theory, Ed. Birkhäuser, 2007.

[201] B. Korte and J. Vygen, Combinatorial Optimization. Theory and Algorithms, Springer, 2006.

[202] E. Shimon, Graph Algorithms, Cambridge University Press, 2012.

[203] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, "Optimization by Simulated Annealing," *Science,* vol. 220, no. 4598, pp. 671-680, 1983.

[204] C. Sechen and A. Sangiovanni-Vincentelli, "The TimberWolf placement and routing package," *IEEE Journal of Solid-State Circuits,* vol. 20, no. 2, pp. 510-522, 1985.

[205] R. M. Kling and P. Banerjee, "ESp: Placement by simulated evolution," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 8, no. 3, pp. 245-256, 1989.

[206] S. M. Sait and J. A. Khan, "Simulated evolution for timing and low power VLSI standard cell placement," *Engineering Applications of Artificial Intelligence,* vol. 16, no. 5-6, pp. 407-423, 2003.

[207] J. P. Cohoon and W. D. Paris, "Genetic Placement," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 6, no. 6, pp. 956-964, 1987.

[208] V. Schnecke and O. Vornberger, "Genetic design of VLSI-layouts," in *Genetic Algorithms in Engineering Systems: Innovations and Applications*, 1995.

[209] P. R. Fernando, "Genetic algorithm based design and optimization of VLSI ASICs and reconfigurable hardware," PhD thesis. University of South Florida, 2009.

[210] B. W. Kernighan and S. Lin, "An Efficient Heuristic Procedure for Partitioning Graphs," *The Bell System Technical Journal,* vol. 49, no. 2, pp. 291-307, 1970.

[211] C. M. Fiduccia and R. M. Mattheyses, "A Linear-Time Heuristic for Improving Network Partitions," in *19th Design Automation Conference*, 1982.

[212] M. A. Breuer, "A class of min-cut placement algorithms," in *Design Automation Conference*, 1977.

[213] J. A. Roy, D. Papa, S. Adya, H. Chan, A. Ng, J. Lu and I. L. Markov, "Capo: Robust and Scalable Open-Source Min-Cut Floorplacer," in *Proc. International Symposium on Physical Design*, 2005.

[214] A. E. Caldwell, A. B. Kahng and I. L. Markov, "Can Recursive Bisection Alone Produce Routable Placements?," in *Design Automation Conference*, 2000.

[215] C. J. Alpert, T. Chan, D. J. Huang, I. Markov and K. Yan, "Quadratic placement revisited," in *Design Automation Conference*, 1997.

[216] J. M. Kleinhans, G. Sigl, F. M. Johannes and K. J. Antreich, "GORDIAN: VLSI placement by quadratic programming and slicing optimization," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems,* vol. 10, no. 3, pp. 356-365, 1991.

[217] P. Spindler, U. Schlichtmann and F. Johannes, "Kraftwerk2—A Fast Force-Directed Quadratic Placement Approach Using an Accurate Net Model," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 27, no. 8, pp. 1398-1411, 2008.

[218] M.-C. Kim, D.-J. Lee and I. L. Markov, "SimPL: An Effective Placement Algorithm," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 31, no. 1, pp. 50-60, 2012.

[219] Y.-W. Chang, Z.-W. Jiang and T.-C. Chen, "Essential Issues in Analytical Placement Algorithms," *IPSJ Transactions on System LSI Design Methodology,* vol. 2, pp. 145-166, 2009.

[220] K. Shahookar and P. Mazunder, "VLSI Cell Placement Techniques," *ACM Computing Surveys,* vol. 23, no. 2, pp. 143-220, 1991.

[221] I. L. Markov, J. Hu and M.-C. Kim, "Progress and Challenges in VLSI Placement Research," *Proc. of the IEEE,* vol. 103, no. 11, pp. 1985-2003, 2015.

[222] A. E. Caldwell, A. B. Kahng and I. L. Markov, "Hierarchical whitespace allocation in top-down placement," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 22, no. 11, pp. 1550-1556, 2003.

[223] S. N. Adya, M. C. Yildiz, I. L. Markov, P. G. Villarrubia, P. N. Parakh and P. H. Madden, "Benchmarking for large-scale placement and beyond," *IEEE*

Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 23, no. 4, pp. 472-487, 2004.

[224] C. Y. Lee, "An Algorithm for Path Connections and Its Applications," *IRE Transactions on Electronic Computers,* Vols. EC-10, no. 2, pp. 364-365, 1961.

[225] F. O. Hadlock, "A shortest path algorithm for grid graphs," *Networks,* vol. 7, no. 4, pp. 323-334, 1977.

[226] J. Soukup, "Fast maze router," in *Design Automation Conference*, 1978.

[227] K. Mikami and K. Tabuchi, "A computer program for optimal routing of printed circuit connectors," in *International Federation for Information Processing*, 1968.

[228] D. Hightower, "A solution to line-routing problem on the continuous plane," in *Design Automation Conference*, 1969.

[229] M. Hanan, "On Steiner's Problem with Rectilinear Distance," *SIAM Journal on Applied Mathematics,* vol. 14, no. 2, pp. 255-265, 1966.

[230] F. K. Hwang, "On Steiner Minimal Trees with Rectilinear Distance," *SIAM Journal on Applied Mathematics,* vol. 30, no. 1, pp. 104-114, 1976.

[231] Marek-Sadowska, "Router planner for custom chip design," in *ICCAD*, 1986.

[232] T.-H. Wu, A. Davoodi and J. T. Linderoth, "A parallel integer programming approach to global routing," in *Design Automation Conference*, 2010.

[233] J. Heisterman and T. Lengauer, "The efficient solution of integer programs for hierarchical global routing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 10, no. 6, pp. 748-753, 1991.

[234] M. Burstein and R. Pelavin, "Hierarchical Channel Router," in *Design Automation Conference*, 1983.

[235] D. Deutsch, "A "Dogleg" channel router," in *Design Automation Conference*, 1976.

[236] M. Bollinger, "A Mature DA System for PC Layout," in *Int. Printed Circuits Conference*, 1979.

[237] W. Dees and P. Karger, "Automated rip-up and reroute techniques," in *Design Automation Conference*, 1982.

[238] D. Stroobandt, A Priori Wire Length Estimates for Digital Design, Springer, 2001.

[239] B. S. Landman and R. L. Russo, "On a Pin Versus Block Relationship For Partitions of Logic Graphs," *IEEE Transactions on Computers,* Vols. C-20, no. 12, pp. 1469-1479, 1971.

[240] P. Christie and D. Stroobandt, "The Interpretation and Application of Rent's Rule," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* vol. 8, no. 6, pp. 639-648, 2000.

[241] M. Y. Lanzerotti, G. Fiorenza and R. A. Rand, "Microminiature packaging and integrated circuitry: The work of E. F. Rent, with an application to on-chip interconnection requirements," *IBM Journal of Research and Development,* vol. 49, no. 4.5, pp. 777-803, 2005.

[242] P. Raghavan and C. D. Thompson, "Randomized Routing in Gate-Arrays," University of California, Berkeley, 1984.

[243] W. R. Heller, W. F. Michail and W. E. Donath, "Prediction of wiring space requirements for LSI," in *Design Automation Conference*, 1977.

[244] P. A. Sandborn and P. Spletter, "A comparison of routing estimation methods for microelectronic modules," *Microelectronics International,* vol. 17, no. 1, pp. 36-41, 2000.

[245] A. E. Gamal and Z. A. Syed, "A New Statistical Model for Gate Array Routing," in *Design Automation Conference*, 1983.

[246] A. E. Caldwell, A. B. Kahng, S. Mantik, I. L. Markov and A. Zelikovsky, "On Wirelength Estimations for Row-Based Placement," in *International Symposium on Physical Design* , 1998.

[247] S. Sastry and A. C. Parker, "Stochastic Models for Wireability Analysis of Gate Arrays," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 5, no. 1, pp. 52-65, 1986.

[248] X. Song, Q.-Y. Tang, D. Zhou and Y. Wang, "Wire Space Estimation and Routability Analysis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 19, no. 5, pp. 624-628, 2000.

[249] H. N. Brady and J. Blanks, "Automatic placement and routing techniques for gate array and standard cell designs," *Proc. of the IEEE,* vol. 75, no. 6, pp. 797-806, 1987.

[250] J. Cong and C. L. Liu, "Over-the-Cell Channel Routing," *IEEE Transactions on Computer-Aided Design,* vol. 9, no. 4, pp. 408-418, 1990.

[251] H. E. Krohn, "An Over-Cell Gate Array Channel Router," in *Design Automation Conference*, 1983.

[252] M. E. d. Lima and D. J. Kinniment, "Sea-of-gates architecture," *Microelectronics Journal,* vol. 26, no. 5, pp. 431-440, 1995.

[253] S. Tsukiyama, I. Harada, M. Fukui and I. Shirakawa, "A New Global Router for Gate Array LSI," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 2, no. 4, pp. 313-321, 1983.

[254] J.-N. Song and Y.-K. Chen, "Two-Stage Channel Routing for CMOS Gate Arrays," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 7, no. 4, pp. 439-450, 1988.

[255] M. Burstein and S. J. .. Hong, "Hierarchical VLSI layout: Simultaneous placement and wiring of gate arrays," in *Proceedings of VLSI*, 1983.

[256] W. W.-M. Dai, B. Eschermann, E. S. Kuh and M. Pedram, "Hierarchical Placement and Floorplanning in BEAR," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 8, no. 12, pp. 1335-1359, 1989.

[257] A. A. Szepieniec, "Integrated placement/routing in sliced layouts," in *Design Automation Conference*, 1986.

[258] T. Koide and S. Wakabayashi, "A timing-driven global routing algorithm with pin assignment, block reshaping, and positioning for building block layout," in *Design Automation Conference. Asia and South Pacific.*, 1998.

[259] M. Burstein and M. Youssef, "Timing Infuenced Layout Design," in *Design Automation Conference*, 1985.

[260] I. Kuon, R. Tessier and J. Rose, "FPGA Architecture: Survey and Challenges," *Foundations and Trends in Electronic Design Automation,* vol. 2, no. 2, pp. 135-253, 2008.

[261] D. Chen, J. Cong and P. Pan, "FPGA Design Automation: A Survey," *Foundations and Trends in Electronic Design Automation,* vol. 1, no. 3, pp. 195-330, 2006.

[262] S. M. Trimberger, Field-Programmable Gate Array Technology, Springer, 1994.

[263] W. Feng, X. Chen, F. J. Meyer and F. Lombardi, "Reconfiguration of one-time programmable FPGAs with faulty logic resources," in *International Symposium on Defect and Fault Tolerance in VLSI Systems*, 1999.

[264] X.-T. Chen, W. Feng, J. Zhao, F. J. Meyer and F. Lombardi, "Reconfiguring one-time programmable FPGAs," *IEEE Micro,* vol. 19, no. 6, pp. 53-63, 1999.

[265] R. D. Newbould and J. D. Carothers, "Cluster growth revisited: fast, mixed-signal placement of blocks and gates," in *Southwest Symposium on Mixed-Signal Design*, 2003.

[266] A. Hashimoto and J. Stevens, "Wire routing by optimization channel assignment within large apertures," in *Design Automation Workshop*, 1971.

[267] C. Martínez-Domingo, A. Alcalde-Aragonés, J. Carrabina, E. Ramon, E. Sowade, K. Mitra, R. Baumann, H. Gomes and F. Villani, "All-inkjet Printed Devices and Circuits," in *Proc. DCIS*, 2013.

[268] M. Mashayekhi, A. Conde, T. N. Ng, P. Mei, E. Ramon, C. Martinez-Domingo, A. Alcalde, L. Terés and J. Carrabina, "Inkjet printing design rules formalization and improvement," *Journal of Display Technology,* vol. 11, no. 8, pp. 658-665, 2015.

[269] E. Ramon, C. Martínez-Domingo, A. Alcalde-Aragonés, A. Conde, J. Pallarès, L. Terés and J. Carrabina, "Large-Scale Fabrication of All-Inkjet Printed Organic Thin Film Transistors: a Quantitative Study," in *NIP & Digital Fabrication Conference*, 2014.

[270] S. Ogier, Y. Lee, M. Cooke, K. McCall, M. Palumbo, S. Chan, D. Bird, L. Evans, S. Rutter and T. Pease, "Developments of stable, high performance organic transistor technology for flexible display applications," in *International Meeting on Information Display - IMID Digest*, 2011.

[271] K. McCall, S. Rutter, E. Bone, N. Forrest, J. Bissett, J. Jones, M. Simms, A. Page, R. Fisher, B. Brown and S. Ogier, "High Performance Organic Transistors Using Small Molecule Semiconductors and High Permittivity Semiconducting Polymers," *Advanced Functional Materials,* vol. 24, no. 20, pp. 3067-3074, 2014.

[272] S. Jacob, M. Benwadih, J. Bablet, I. Chartier, R. Gwoziecki, S. Abdinia, E. Cantatore, L. Maddiona, F. Tramontana, G. Maiellaro, L. Mariucci, G. Palmisano and R. Coppard, "High performance printed N and P-type OTFTs for complementary circuits on plastic substrate," in *ESSDERC*, 2012.

[273] F. -. EMFT, "Complementary Organic Semiconductor and Metal Integrated Circuits (COSMIC)," European project. Ref: FP7-ICT-2009-4, 247681, 2010.

[274] M. Llamas, "Automatic Cell Library Generation for Inkjet Printed Electronics," M.Sc. Thesis. Microelectronics and Electronic System Department (MiSE). Universitat Autònoma de Barcelona (UAB), 2013.

[275] UAB and IMB-CNM-CSIC, "Application Specific Printed Electronics Circuits - Technology and Design Kit Development (ASPEC-TDK). Deliverables D6.3 & D6.4," Spanish project. Ref: TEC2011-29800-C03, 2016.

[276] A. Mishchenko, S. Chatterjee and R. Brayton, "DAG-aware AIG rewriting. A fresh look at combinational logic synthesis," in *Design Automation Conference*, 2006.

[277] T. Uehara and W. M. Vancleemput, "Optimal Layout of CMOS Functional Arrays," *IEEE Transactions on Computers,* vol. 30, no. 5, pp. 305-312, 1981.

[278] F. Brglez, D. Bryan and K. Kozminski, "Combinational Profiles of Sequential Benchmark Circuits," in *International Symposium on Circuits and Systems (ISCAS)*, 1989.

[279] F. Corno, M. Reorda and G. Squillero, "RT-level ITC'99 Benchmarks and First ATPG Results," *IEEE Design & Test of Computers,* vol. 17, no. 3, pp. 44-53, 2000.

[280] F. Brglez and H. Fujiwara, "A neutral netlist of 10 combinational benchmark circuits and a target translator in Fortran," in *Int. Symposium on Circuits and Systems, Special Session on ATPG and Fault Simulation*, 1985.

[281] S. M. Rubin, "A general-purpose framework for CAD algorithms," *IEEE Communications Magazine,* vol. 29, no. 5, pp. 56-62, 1991.

[282] B. N. Miller and D. L. Ranum, Problem Solving with Algorithms and Data Structures using Python, Ed. Franklin, Beedle & Associates, Inc., 2006.

[283] A. Downey, Think Python. How to Think like a Computer Scientist, Ed. Green Tea Press, 2012.

[284] H. P. Langtangen, Python Scripting for Computational Science, Ed. Springer-Verlag, 2008.

[285] C. Severance, Python for Informatics. Exploring Information, Ed. CreateSpace Independent Publishing Platform, 2013.

[286] "Python TimeComplexity," [Online]. Available: https://wiki.python.org/moin/TimeComplexity.

[287] R. Pattis, "Python lectures," [Online]. Available: https://www.ics.uci.edu/~pattis/ICS-33/lectures/complexitypython.txt.

[288] F. Pedregosa, "Memory Profiler (Python)," [Online]. Available: https://pypi.python.org/pypi/memory_profiler.

[289] G. Rodola, "psutil," [Online]. Available: https://github.com/giampaolo/psutil.

[290] E. Macias, "Physical synthesis on Benchmark digital circuits for Printed Electronics using Standard Cells on Cadence tools," M.Sc. Thesis. Microelectronics and Electronic System Department (MiSE). Universitat Autònoma de Barcelona (UAB), 2017.

[291] "Intel Corporation," [Online]. Available: https://www.intel.com.

[292] "Holst Centre," [Online]. Available: https://www.holstcentre.com/.

Annexes

Annex A: Digital printing examples

The following Figure 33 represents the metal customization layer for the P&R of a C17 circuit onto the CPI/Neudrive IGA, with a 100% yield; or a minimum yield of at least 18/54 => 33%, This would be the best case scenario, for a worst case yield required for this particular circuit, where all KGO transistors are located in the bottom left corner.

Figure 34 depicts the metal customization layer for the P&R of a C17 circuit, again over the CPI/Neudrive IGA, with a 31/54 => 57% yield, and a random dispersed fault distribution.

Figure 35 represents the same circuit, but onto the CEA IGA, with the same yield conditions as in Figure 33.

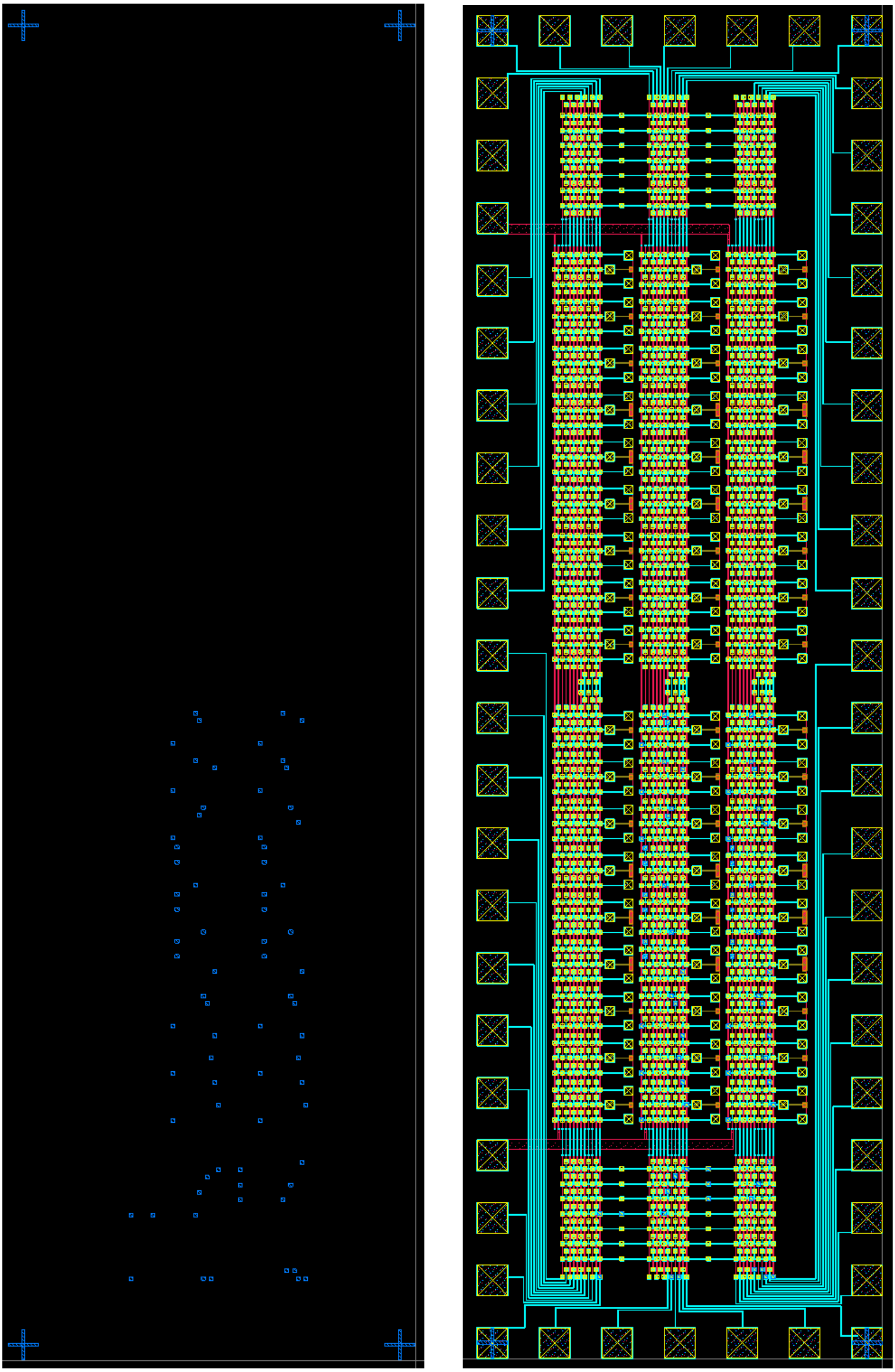


Figure 33. C17 over CPI/Neudrive IGA (case 1). Only personalization metal (left). Final IGA (right).

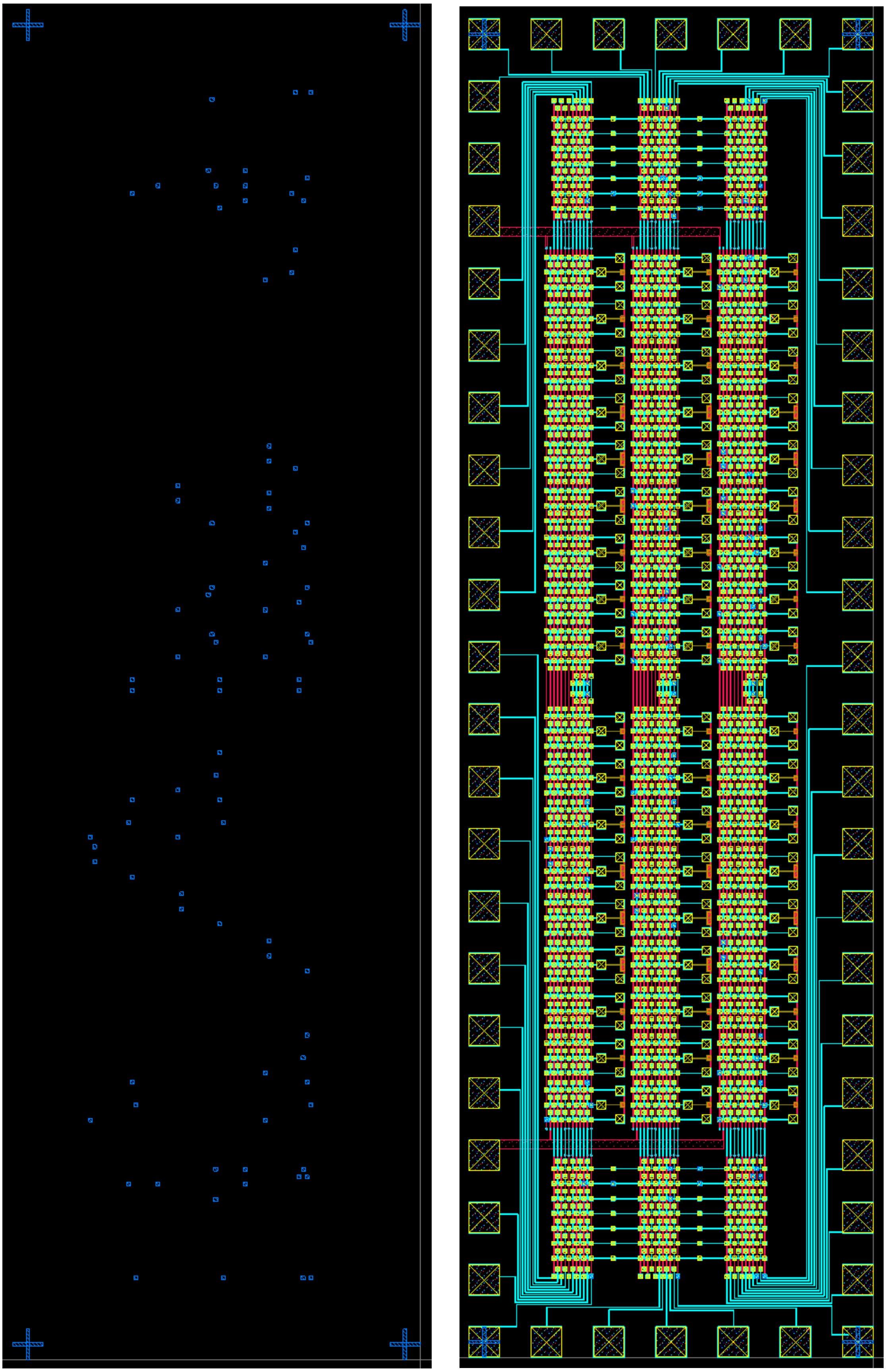


Figure 34. C17 over CPI/Neudrive IGA (case 2). Only personalization metal (left). Final IGA (right).

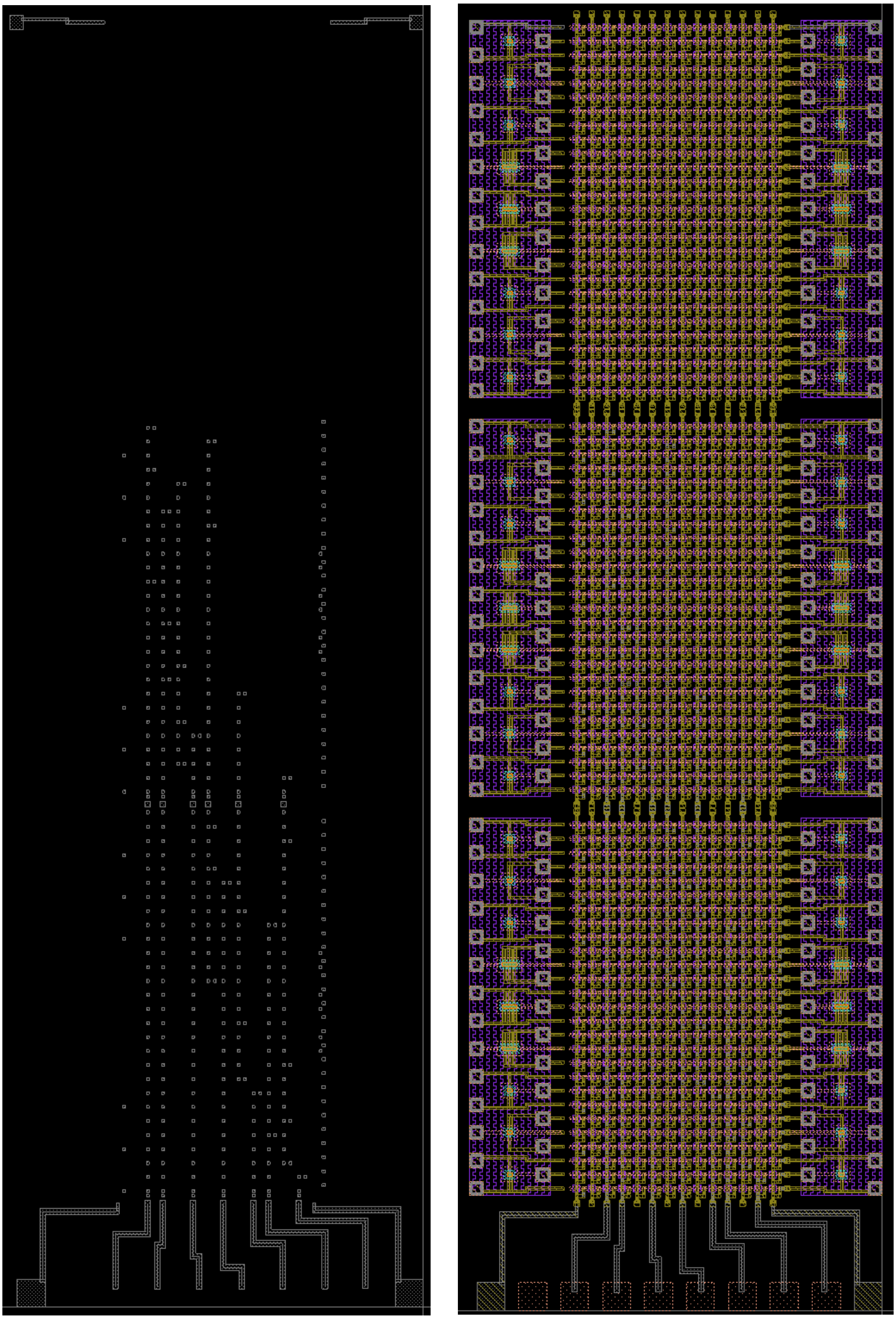


Figure 35. C17 over CEA IGA. Only personalization metal (left). Final IGA (right).

Annex B: Dissertation’s Copyright

All the ideas, statements, figures, tables, IPs, and information presented in this dissertation belong to their respective authors, including myself. All of them have been rigorously referenced. If there were no references for specific information it is because such knowledge is considered to be either well-known industry-related facts, or the results of my own original research and findings.

Annex C: Implementation’s Copyright & Disclaimer

The implementation of my own algorithms is protected by copyright and is not eligible for distribution as FOSS. All rights are reserved. The developed tools are only for demonstration purposes, in order to validate the ideas and findings presented in this dissertation. A standard legal text is given below.

****

Copyright © 2017 Manuel Llamas Rodriguez www.manullamas.info All rights reserved.

Unauthorized use, duplication, modification or distribution of this software is strictly prohibited by law. All uses of this software must be previously approved via written permission by the creator and owner of this code.

THE SOFTWARE DEVELOPED BY THE AUTHOR IS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

****