DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Electrical Engineering

Signal Distribution Networks in Automatic QCA Standard Cell Placement and Routing

Benjamin Hien

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Electrical Engineering

Signal Distribution Networks in Automatic QCA Standard Cell Placement and Routing

Signalverteilungs-Netzwerke in automatisierter QCA Standard Zellen Plazierung und Verdrahtung

Author:	Benjamin Hien
Supervisor:	Prof. Dr. Robert Wille
Advisor:	Dr. Marcel Walter
Submission Date:	08.02.2023

I confirm that this master's thesis in electrical engineering is my own work and I have documented all sources and material used.

Munich, 08.02.2023 Benjamin Hien

Acknowledgments

Abstract

New technologies to compete with CMOS, one of them QCA
Placement and Routing as key to producibility.
Challenges of Placement and Routing in previous algorithms.
Goals of this work:
I minimizing area/tiles,
II making it possible to place majority-gates (which is a promising aspect of QCA),
III making it possible to P&R sequential circuits
This is done by introducing several signal distribution networks
Results are compared with already existing algorithms...

Contents

1 Introduction

1.1 Motivation

About the technology, why its promising and important.
Lack of automated algorithms for P&R
P&R as sign of producibility
Why the distribution networks are able to make QCA better producible / cheaper

1.2 Objective

The thesis is divided in ...

2 Preliminaries

2.1 Representation of Logic Circuits

Independent of the underlying technology, digital circuits can be represented by logic functions. The Boolean Algebra, formed by mathematician George Boole in 1847, proposes these logic functions and provides a foundation to discuss about them.

2.1.1 Boolean Functions

The definition as given here is based on an addition to Boolean calculus by Edward V. Huntington. CITE The basis of a Boolean algebra is given as follows:

Definition 2.1.1 (Basis for Boolean algebra)**.** Given a finite set S, two binary functions $\cdot$: S × S → S and + : S × S → S, and one unary function ¬ : S → S, the tuple (S, ·,+,¬) is called a Boolean algebra iff the following constraints hold for all a, b, c ∈ S:

(1)	$a \cdot b = b \cdot a$	$a + b = b + a$
(2)	$a \cdot (b + c) = (a \cdot b) + (a \cdot c)$	$a + (b \cdot c) = (a + b) \cdot (a + c)$
(3)	$\exists 1 \in S : a \cdot 1 = a$	$\exists 0 \in S : a + 0 = a$
(4)	$\exists 0 \in S : a \cdot \neg a = 0$	$\exists 1 \in S : a + \neg a = 1.$

With the constraints describing (1) *commutativity*, (2) *distributivity*, (3) *neutrality*, and (4) *complementarity*.

In the original definition a Boolean algebra is defined by the 6-tuple $(\mathbb{B}, \vee, \wedge, \neg, 0, 1)$, where $\vee$ and $\wedge$ are another denotations for the binary operands for disjunction $+$ and conjunction $\cdot$ in $\mathbb{B}$, the known unary negation function $\neg$, and two distinct elements 0 and 1. Negation $\neg a$ is also commonly notated as $\bar{a}$.

Since this definition is restricting the use of only the three Boolean functions $(\vee, \wedge, \neg)$, we want to extend by the following definition:

Definition 2.1.2. A function $f : \mathbb{B}^n \to \mathbb{B}$, where $n \in N$, is called a Boolean function. Analogously, a function f : $\mathbb{B}^n \to \mathbb{B}^m$, where $n, m \in N$, is called multi-output Boolean

function and can be interpreted as $f_v = (f_{v1}, ..., f_{vm})$, where $f_{vi} : \mathbb{B}^n \to \mathbb{B}$, for all $1 \leq i \leq m$.

A common notation for Boolean functions are the *conjunctive normal form* (CNF) and *disjunctive normal form* (DNF), using literals.

Definition 2.1.3. A literal is an atom or the negation of an atom. In the former case the literal is positive, in the latter case it is negative.

Definition 2.1.4. A formula F is in conjunctive normal form (CNF) if it is a conjunction of disjunctions of literals:

$$\bigwedge_i \bigvee_j (\neg) v_{ij},$$

where $v_{ij} \in \mathbb{B}$.
A formula F is in disjunctive normal form (DNF) if it is a conjunction of disjunctions of literals:

$$\bigvee_i \bigwedge_j (\neg) v_{ij},$$

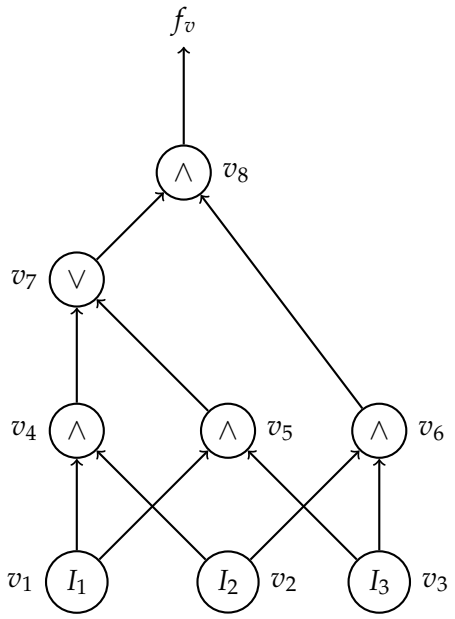
where $v_{ij} \in \mathbb{B}$.

Using the CNF or rather the DNF and *De Morgan's laws* following from the definitions in 2.1.1, it follows that any Boolean Algebra can be reduced to only two operands, e.g. conjunction ($\vee$) and negation ($\neg$). Any set of such two Boolean functions is called *universal*.

2.1.2 Logic Networks

There are many ways of representing Boolean Functions. But most of them, including e.g. truth tables or reduced sum of products, suffer from drawbacks like exponential growth of size with the number of arguments and functions with exponential representations. The representation of combinational circuits as logic networks overcomes these restrictions and has proven to be very useful in the logic synthesis process. Following definition for directed acyclic graphs is shown in CITE:

Definition 2.1.5 (Function graph)**.** A function graph is a rooted, directed graph with vertex set V containing two types of vertices. A *nonterminal* vertex v has as attributes an argument index $index(v) \in \{1, ..., n\}$, and two children $low(v), high(v) \in V$. A *terminal* vertex v has as attribute a value $value(v) \in \{0, 1\}$.



The corresponding recursive Boolean functions read:

$$
\begin{array}{ll}
f_v = f_v(v_8) & f_v(v_6) = f_v(v_2) \vee f_v(v_3) \\
f_v(v_8) = f_v(v_7) \wedge f_v(v_6) & f_v(v_5) = f_v(v_3) \vee f_v(v_1) \\
f_v(v_7) = f_v(v_4) \wedge f_v(v_5) & f_v(v_4) = f_v(v_2) \vee f_v(v_1)
\end{array}
$$

with primary inputs: $f_v(v_1), f_v(v_2), f_v(v_3) \in \{0, 1\}$

Figure 2.1: Binary Logic Network of Majority Function

Furthermore, for any nonterminal vertex v, if $low(v)$ is also nonterminal, then we must have $index(v) < index(low(v))$. Similarly, if $high(v)$ is nonterminal, then we must have $index(v) < index(high(v))$.

Since the definition dosen't yet include Boolean Functions and reduces the number of children connected to a vertice to two, therefore only allowing binary Boolean Functions, a custom definition is given:

Definition 2.1.6 (Logic Network)**.** A logic network is a rooted, directed graph with vertex set V containing two types of vertices. A *nonterminal* vertex v has as attributes an argument index $index(v) \in \{1, ..., n\}$, and l children $child_1(v), ..., child_l(v) \in V$. A

terminal vertex v has as attribute a value $value(v) \in \{0, 1\}$.

Furthermore, for any nonterminal vertex v, if $child_i(v)$ with $1 \leq i \leq l$, then we must have $index(v) < index(child_i(v))$ respectively.

Definition 2.1.7 (Logic Network Boolean Functions)**.** A set of nary Boolean Functions $\mathbb{B}$ is accessed via the argument index, assigning a Boolean Function f_v to every vertex:

1. If v is a terminal vertex:
   a) If $value(v) = 1$, then $f_v = 1$
   b) If $value(v) = 0$, then $f_v = 0$
2. If v is a nonterminal vertex with $index(v) = i$, then f_v is the function $f_v(x_1, ..., x_n) = f_v(f_v(child_1(v)), ..., f_v(child_l(v)))$.

The binary Logic Network of the ternary majority function is depicted figure 2.1:

Definition 2.1.8 (Majority Function)**.** The ternary Boolean majority function is defined as: $< a, b, c >= ab + ac + bc$, so that the function value equals the majority of it's incoming values.
It follows: $< a, b, 0 >= ab$ and $< a, b, 0 >= a + b$.

Because of the names used in the underlying libraries used to program the algorithms proposed in chapter... the vertices are referred to as nodes, while edges connecting a vertex with one of its children is called a signal. Terminal vertices, found at the beginning of the graph and therefore possessing the lowest indices, are called primary inputs. The enumerated vertices f_{vi}, representing the boolean functions are called primary outputs.
!!ADD a Denotation
As already mentioned in subsection 2.1.1, a set of two certain Boolean functions can form any Boolean algebra. As long as this universality is contained the set of node functions can be extended. Using conjunction and negation as the only node functions of a logic network, we get the so called *AND-Inverter Graphs* (AIGs). Another widely used binary logic network is the *Majority-Inverter Graphs* (MIGs) utilizing the ternary majority function and negation. But there also exists a wide range of logic networks permitting more than just two node functions, like *XOR-AND-Inverter Graphs* (XAGs).

Since the logic network represents the combinational circuit in the given technology, a suitable logic network representation has to be determined. Because, even though these logic networks can implement any Boolean function given in a specification, not every logic network can be synthesized into any given technology. Looking at the current

standard technology *complementary metal-oxidesemiconductor* (CMOS), the logic network is then synthesized by using building blocks consisting of *metal-oxide-semiconductor field-effect transistors* (MOSFETs), the elemental unit in this technology. The process of turning a circuit specification into a logic gate representation is called *logic synthesis*.

Given these logic network characteristics, none of the representations are *canonical*, which means that a given function can be represented by different logic networks. This property can be explained by the fact that nodes with the identity function are allowed. Even the exclusion of such identity nodes has no impact, since simple node combinations, like two negotiation nodes, collapse to the identity function. Following this argumentation, there exists an infinite number of logic networks representing one Boolean Function, resulting in he widely accepted assumption, that the determination of an optimal logic network is a $\mathcal{NP}$-complete problem. Attempts to create canonical logic networks, seem to evade this problem, but include co$\mathcal{NP}$-complete problems in itself. Algorithms used for logic synthesis are therefore based on approximate solutions.

2.2 QCA Technology

Following the well known Moores law CMOS technology is facing a multitude of challenges, e.g. short channel effect, impurity variations, and most importantly the heat, resulting from static and dynamic power losses. To tackle these challenges the International Roadmap for Devices and Systems (IDRS), former ITRS, proposes solutions within the semiconductor domain, e.g. new materials and multi-core architectures. But also new technologies are researched including Quantum computing and the domain of *Field-Coupled Nanocomputing* (FCN). This work focuses on one of the most promising FCN technologies, namely *Quantum dot cellular automata* (QCA). The main difference of this technology compared to CMOS is the representation of logical modes, using the location of electron pairs in QCA-cells, instead of voltage levels. Data between cells is transferred based on Coulomb repulsion, utilizing electromagnetic fields. This enables the technology to achieve high performance in terms of device density, clock frequency and power consumption.

Point out why majority gates are a big part of this technology.
Point out why wires and especially wire crossings are so expensive to produce.

2.2.1 Cells

As already mentioned, the elemental unit of this technology is a QCA-cell. Since there in no uniform way of build the quantum dots and connecting them to cells, we look at

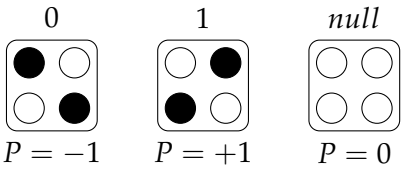


Figure 2.2: QCA-Cell sates

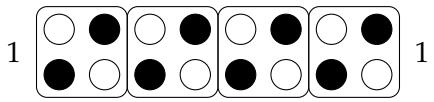


Figure 2.3: Adjacent QCA-cells forming a wire

a rather lower-level abstraction depicted in figure 2.4. The four circles in the corners of the QCA-cells show quantum-dots, that can be implemented by any charge container with discrete electrical energy states. Further a cell contains two excess electrons, which can be localized by the quantum dots. The energy barriers or the quantum dots are able to trap the charge of the electron. If an electron is trapped inside a quantum-dot it is filled black. Due to the Coulomb repulsion the electrons occupy diagonally opposed quantum dots, resulting in two possible stable cell configurations and one unstable cell configuration.
A stable states indicates, that it is well distinguishable of the usual energy band and therefore has a energy difference to another stable state of minimum the thermal noise energy ($k_B T$). Only such states are suited for information transfer. The stable states can be derived from the cell Polarization, which can be $+1$ and -1 or *null* in the unexcited state. The two stable states contain the same electrostatic energy and are used to encode the binary values 0 and 1.
In order to transfer information, cells are placed side by side, whereby the polarization of the driver cell, which is the left most cell inputting the information, changes the polarization of the adjacent cell. When the adjacent cell is polarized it can give its state to the next cell and so on. This simple structure is representing a wire in QCA-technology and its function is depicted in Figure 2.3.

2.2.2 Clocking

Explain the clocking of cells.
Show different clocking schemes and the ideas behind them.
Point out that one clocking zone is represented by exactly one cell and not multiple cells.

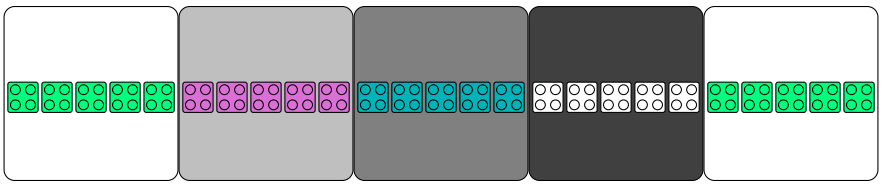


Figure 2.4: QCA-Cell wire with corresponding clock zones

Point out the local and global clocking constraints

2.2.3 Gates

Talk about QCA Standard Cell Library

Latches and Registers

2.3 P&R problem

Describe the challenges of P&R in QCA, further details are in the next chapter

3 State of the Art

3.1 Combinational P&R Algorithms

Quick recap on the Algorithms from the book
Quick recap on ortho and why i selected ortho for my work.
Maybe point out that balancing is not used.

3.2 Sequential P&R

Present the ideas in papers for QCA standard cell placement and routing.
Point out why they are not actionable: Reasons like clocking or cells aren't producible, no automated algorithms

4 Methodology

4.1 Input Network

Reduces area and wire crossings.
Sorts the inputs.
Idea is simple place Fanout nodes at the beginning since they produce the most crossings
Needs a conditional east west coloring.

4.2 Majority Gates Placement Network

Reference to the importance of this part in the QCA technology.
Point out that 2DD-Wave is still used but clocking scheme is adjusted to place majority gates.
This is why buffers have to be used.
Point out the overhead that is produced by buffers.
Maybe assumption why it doesn't make sense to only use RES. (you lose really much space for the cells which are no majority gates)

4.3 Sequential Circuits Placement

Importance of sequential circuits.
Point out how the registers are implemented.
Show how the distribution network is generated, where Ris and Ros are placed and how they are treated within the network.
Make clear that this implementation is slowing down the circuit significantly.

5 Experimental Evaluation

5.1 Benchmarks

For combinational and sequential

5.2 Results

Everything

List of Figures

List of Tables

Bibliography

[1] L. Lamport. *LaTeX : A Documentation Preparation System User's Guide and Reference Manual*. Addison-Wesley Professional, 1994.