# Chapter 8
# Design Automation for On-Chip Nanophotonic Integration

**Christopher Condrat, Priyank Kalla, and Steve Blair**

**Abstract** Recent breakthroughs in silicon photonics technology are enabling the integration of optical devices into silicon-based semiconductor processes. Significant developments in silicon photonic manufacturing and integration are enabling investigations into applications beyond that of traditional telecom: sensing, filtering, signal processing, quantum technology—and even optical computing. In effect, we are now seeing a convergence of communications and computation, where the traditional roles and boundaries of optics and microelectronics are becoming blurred. As the applications for opto-electronic integrated circuits (OEICs) are developed, and manufacturing capabilities expand, design support is necessary to fully exploit the potential of this technology. Photonic design automation represents an opportunity to take OEIC design to a larger scale, facilitating design-space exploration, and laying the foundation for current and future optical applications—thus fully realizing the potential of this technology.

This chapter describes our work on design automation for integrated optic system design. Using a building-block model for optical devices, we provide an EDA-inspired design flow and methodologies for optical design automation. Underlying these flows and methodologies are new supporting techniques in behavioral and physical synthesis. We also provide modeling for optical devices, and determine optimization and constraint parameters that guide the automation techniques. Starting from a logic design model, we describe how conventional logic synthesis and physical design techniques (placement, global and detail routing) can be applied in a top-down fashion to engineer a fully automated design flow for integrated optical systems.

This research was funded in part by a sub-contract to AFOSR grant FA9550-09-1-0661.

C. Condrat
Calypto Design Systems, Wilsonville, OR, USA
e-mail: chris@g6net.com

P. Kalla (✉) • S. Blair
Department of Electrical and Computer Engineering, University of Utah, Salt Lake City, UT, USA
e-mail: kalla@ece.utah.edu; blair@ece.utah.edu



R.O. Topaloglu (ed.), *More than Moore Technologies for Next Generation Computer Design*, DOI 10.1007/978-1-4939-2163-8_8

## 8.1 Introduction

Advancements in integrated optics are expanding the role of optical devices in system design. Opto-electronic integrated circuits (OEICs) [1], merging optics and control electronics on a monolithic substrate, are now a reality and enable optical integration in a diverse set of applications, such as sensing, signal processing, communications, and also computing [2–9]. The driving forces behind optics technology come from different, but inter-related areas. One area is *optical interconnects.* As semiconductors feature sizes have scaled downward, metal interconnects are now the dominant cause of delay and power usage in system design. In addition, the trend towards greater parallelism at the system level [10] has prioritized the role of communications in computing. Optics are therefore being pushed as an inter- and intra-chip *interconnect technology* to provide high-speed, long-haul, *low-power* communications [11–16].

A second driving force behind optical technology is that of manufacturing. Silicon is the mainstay of the semiconductor industry. The ease of manufacturing for semiconductors in well-characterized silicon-based processes, and steady improvements in performance and density at each process node makes CMOS-based technology the dominant computing manufacturing technology. For the same reasons, attempts were also made to develop silicon-based integrated optics—*silicon photonics*. Silicon's high refractive index and transparency to telecom wavelengths make it a suitable material for integrated optical waveguides, but silicon's use had traditionally been limited to *passive* optical devices such array waveguide gratings (AWGs) [17]. In the active domain, silicon's indirect band gap limiting silicon-based lasers, inability to detect light at telecom wavelengths, and slow modulation due to weak or absent electro-optic effects [18, 19] stymied nanophotonic device development. III-V semiconductor compounds such as gallium arsenide (GaAs) and indium phosphide (InP), or materials such as lithium niobate ($LiNbO_3$) would become the materials of choice for active photonic devices.

This changed in 2005, when Intel Corporation announced the first all-silicon optical modulator operating beyond the 1 Ghz threshold [20]. Fast optical modulation would be a significant breakthrough in silicon photonics, enabling viable optical networks to be fabricated in all-silicon processes. This development ushered in a number of subsequent breakthroughs in silicon photonic device development, including faster modulators [21, 22], hybrid lasers [23], and other device technologies [24] including hybrid silicon-germanium processes [25] for on-chip detectors. This promise of monolithic integration of OEICs in silicon-based processes opens the door to a great number of opportunities in system design. Already a number of architectures have been proposed for connecting systems via optical interconnect networks [14, 26], including as separate layers in 3D ICs. Investigations have also been made into optical digital signal processing [27], sensing, and even computing frameworks that can leverage optics in ways that would have been cost-prohibitive. In essence, silicon-based integrated optics are enabling the *convergence of computation and communication.*

As optical devices are integrated on larger scales, the need for design automation becomes apparent to handle greater levels of complexity in design. Scalability requires abstractions, which in turn enables and requires the use of optimization algorithms, design methodologies and tool-flows. What is now required is an Electronic Design Automation (EDA) type tool flow replicated and adapted to the optical domain. Such a *Photonic Design Automation (PDA)* represents an opportunity to take OEIC design to a larger scale, facilitating design-space exploration, and laying the foundation for current and future optical applications—thus fully realizing the potential of this technology. In this chapter, we describe the research and development efforts towards PDA by our research group: proposing a design automation flow with abstractions, optimization algorithms, tool-flows, and methodologies —enabling the synthesis of OEICs through automated means. We demonstrate our approach on *optical computing* applications, even though our approach is quite generic and not restricted to optical computing.

*Contributions:* In our work, we consider optical switching devices such as Mach-Zehnder Interferometers (MZI) and ring resonators connected by waveguides, splitters, couplers, detectors, etc., to form optical logic computing systems. We describe how Boolean logic synthesis techniques can be adapted to such a technology to design optical logic circuits. Subsequent to logic design, we describe a physical design methodology for placement and routing of optical circuits. We analyze technology-specific cost metrics during each stage of the synthesis process, and design algorithmic techniques that optimize for them. The chapter provides an overview of our contributions, and interested readers are referred to [2, 28–32] for more details.

*Organization:* The chapter is organized as follows: The following subsection depicts the photonic design flow proposed in this work. Section 8.2 describes the overall view of an integrated optical system. Section 8.3 covers the background and switching device models employed in this work. Section 8.4 describes the proposed photonic logic synthesis model. Section 8.5 describes the physical design automation methodology and cost models. Section 8.6 covers the global routing approach, whereas the detail routing model is elaborated in Section 8.7. Section 8.8 concludes the chapter.

### *8.1.1 The Photonic Design Automation Flow*

Our design flow, depicted in Fig. 8.1, draws inspirations from EDA design flows and methodologies, and it consists of behavioral synthesis and physical synthesis. Ancillary to this flow is *technology modeling,* where the groundwork is laid for design automation in terms of building-block models and optimization metrics used throughout the design flow. System integration also plays an important role by introducing external constraints and effects on the optical system such as: area-limitations, packaging, and thermal interactions between on-chip heat sources and optical devices.

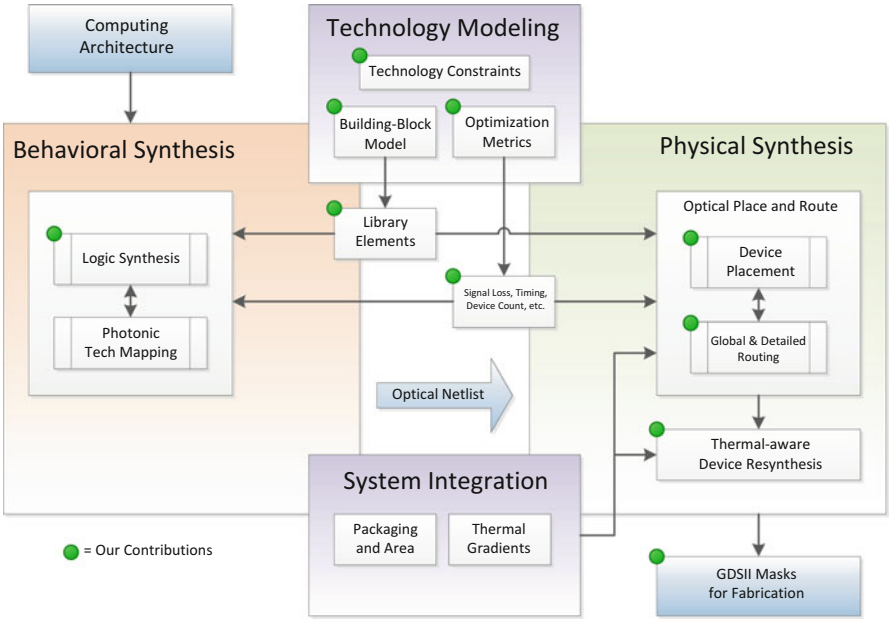


**Fig. 8.1** The proposed design flow

## 8.2 Integrated Optic Systems

Figure 8.2 depicts a high-level view of an integrated optics system. We describe the components of this system and their operations; the details of the individual devices can be found in [33]. At the optical inputs of a system are lasers that provide light at the wavelengths the system is designed for, around 1, 550 nm for SOI systems. For silicon-based processes, this light is usually coupled into the system from outside using fiber couplers or grating couplers. To inject data into the system, modulation devices such as Mach Zehnder interferometers (MZIs), are used to vary the intensity of the input light. The light is then routed throughout the substrate using waveguides and optical switching devices with electrical switching inputs or in some cases employing all-optical switching.

The routing network also includes passive devices such as waveguide splitters, waveguide crossings, and passive multiplexing devices such as array waveguide gratings. Splitters divide the input among two outputs, with each output receiving half the input power, minus losses. Crossings are necessary for waveguides to cross each other on the single-layer planar substrate with minimal losses; crossings will feature into our physical design work in subsequent sections. Devices such as array waveguide gratings enable (de)multiplexing of various wavelengths, and have been a useful application for 1st-generation silicon photonics.

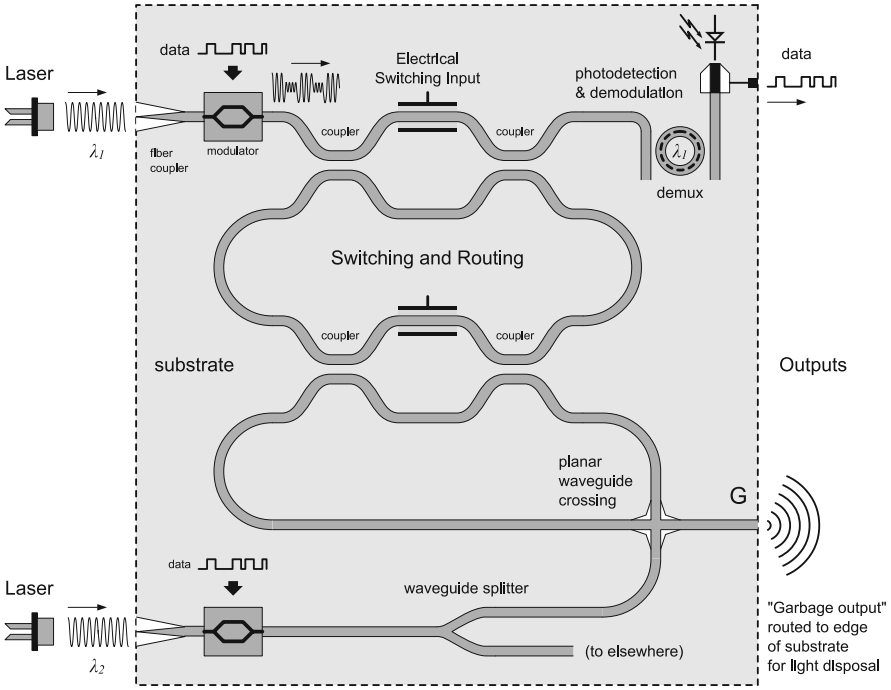


**Fig. 8.2** High-level view of an integrated optic system

At the outputs of the system are demultiplexers for multi-wavelength systems, photodetectors and garbage outputs. Waveguides can support ranges of wavelengths, and therefore multiple channels of data may be present on a waveguide that need to be demultiplexed at the output. After demultiplexing, a photodetector (receiver) is required to translate optical signals into electrical signals, to read the transmitted data. Such photodetectors utilize materials such as germanium [34], which are incorporated into modern silicon photonics processes [35]. Finally, some routing networks need to dispose of unused light. To prevent interference and noise, the light from these "garbage outputs" must either be routed to the edge of the substrate for disposal, or absorbed by a material such as germanium, placed near the exit-point of the waveguide.

## 8.3 Device Models for Synthesis

One of the goals of this work is to develop synthesis techniques that utilize conventional integrated optics devices that can be fabricated with current technology, while also being applicable to future design processes. We describe the basic operation of the integrated optic devices we utilize.

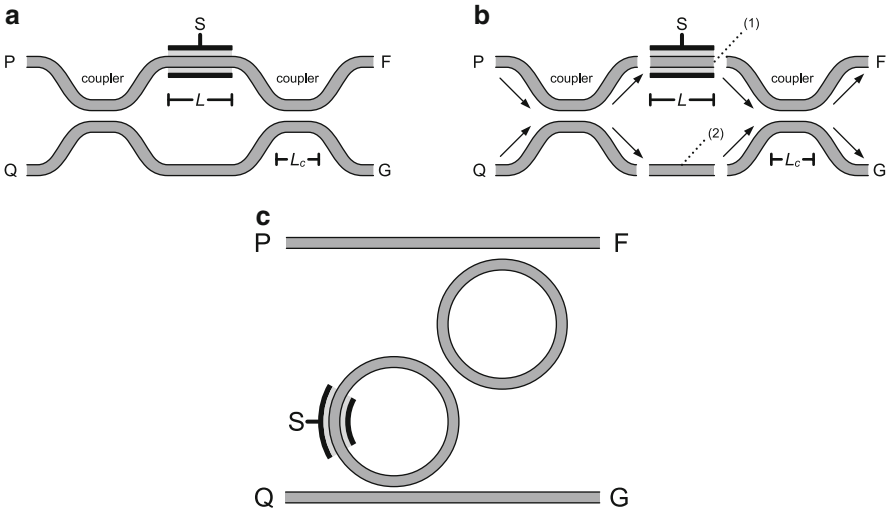


**Fig. 8.3** Mach-Zehnder interferometer routing devices. (**a**) Mach-Zehnder interferometer (MZI); (**b**) MZI in parts; (**c**) Ring-resonator modulator

Routing light using waveguides is performed through the use of coupling and controlled interference. Consider the Mach-Zehnder Interferometer (MZI) depicted in Fig. 8.3a. The paths connected between $P$ and $F$ and $Q$ and $G$ are waveguides. Under certain conditions, when waveguides are brought in close proximity to each other, energy transfers between one waveguide to the other, and vice-versa. The couplers in this device are 3dB couplers, dividing and/or combining the signal from both inputs equally between the two outputs. The actual routing is controlled by input $S$, described by the following equations:

$$\phi_1 = \frac{\omega}{c} \cdot n \cdot L \qquad \phi_2 = \frac{\omega}{c} \cdot (n + \Delta n) \cdot L \tag{8.1}$$

$$\Delta\phi = |\phi_2 - \phi_1| = \pi = \frac{\omega}{c} \cdot \Delta n \cdot L \tag{8.2}$$

where $\omega$ is the angular frequency of the light (dependent on wavelength), $\phi_1$ and $\phi_2$ represent the phase of the light in the two center waveguides, and $n$ is the index of refraction for the waveguide.

The input $S$ is used to change the refractive index of Fig. 8.3b(1) by $\Delta n$ via heating, carrier injection, or other means. This causes a path-length difference, and therefore a phase difference, between the signals in Fig. 8.3b(1) and b(2), causing constructive or destructive interference at the second coupler. A phase difference of 0 or $\pi$ [36] will route each input *completely* to one output or the other, and the device acts as the controlled crossbar depicted in Fig. 8.4a. Similarly, other designs [16,37], as depicted in Fig. 8.3c, can be used to reduce the amount of phase-shift needed and the size of the overall device. Changing the refractive index can be accomplished by

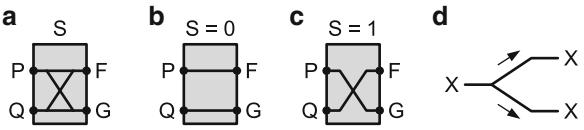


**Fig. 8.4** Crossbar switch, and different routing configurations. (**a**) Gate; (**b**) Bar; (**c**) Cross; (**d**) Splitter

using a microheater or more advanced methods such as the MOS-capacitors used in Intel's high-speed modulator [20]. Modulation is also possible using devices such as ring resonators. The operation of such devices will be covered in later chapters. In our work, we can utilize either an MZI or ring resonators as an electrically controlled optical crossbar switch to design digital optical logic.

The operation of the MZI allows us to model it as a crossbar *gate* that routes light signal completely between two paths depending on the state of **S**, and depict it symbolically in Fig. 8.4a, with its two states Fig. 8.4b and c (bar and cross respectively). The waveguides are sourced by light (logical "1") or darkness ("0"), and the output of a function is read using optical receivers at the end. In our model, the switching input **S** is an *electrical* signal; it is an outside signal that controls the cross/bar configuration and cannot be switched by optical inputs. Connections to **p** and **q**, and **f** and **g** are waveguides, and for simplicity, light is assumed to move from the **p** and **q** side to **f** and **g**. In our model, *an optical signal cannot directly switch a crossbar's* ***S*** *input*[1]. More formally:

$$\begin{aligned}(S = 0) &\Rightarrow (P = F) \wedge (Q = G)\\ (S = 1) &\Rightarrow (Q = F) \wedge (P = G)\end{aligned} \tag{8.3}$$

These constraints affect how functions may be composed, and imply that the inputs to a crossbar are the primary inputs for that network. Waveguide connections between crossbar gates are depicted symbolically as black "wires." All designs created using the above model can be physically realized, including allowing waveguides to cross each other without interference.

In addition to MZIs, we also utilize *optical splitters,* depicted symbolically in Fig. 8.4d. A splitter divides the light from one waveguide into two output waveguides, each of which contain the original signal, but at half the power (a 3 dB loss). In our model, splitters are a significant signal degradation mechanism for a given topology; losses due to waveguide bends, waveguide crossings and insertion losses for MZI devices are the other mechanisms.

[1]Switching a crossbar gate with an optical signal requires an opto-electrical interface comprising an optical receiver unit feeding switching hardware. This can be expensive and slow, and is currently beyond the scope of the synthesis technique applied to this device model.

## 8.4 Optical Boolean Logic

Static-CMOS benefits from two important properties: metals and semi-conductors conduct when physically connected, and logic is restorative in nature. These two properties grant static-CMOS a great level of flexibility for implementing and optimizing logic functions, especially as it allows fanout for multi-level logic implementation. Unfortunately, this flexibility does not extend to optical circuits.

Consider the two networks in Fig. 8.5 implementing functions $f_1 = a + b$ and $f_2 = c \cdot (a + b)$. The first network implements $f_2$ by using the output of $f_1$ to drive the switching input of a gate. This is an unworkable design under our model, because an optical signal $f_1$ cannot switch the electrical input of another gate. A more optimal solution is found in the second design Fig. 8.5b, which uses $f_1$ as an *optical* input to another gate. This design benefits from using fewer gates, but more importantly, the sub-function is kept entirely in the optical domain. In such a way sub-functions *can* be shared, but with limitations.

### *8.4.1 Waveguide Splitters*

The device which enables signal sharing using waveguides is the *waveguide splitter*. A waveguide splitter shares the signal of the input waveguide between two output waveguides, dividing the input power between two outputs, generally with a 50:50 ratio (3 dB loss). As the outputs of the splitter have only half the power of the original signal, there are limitations on how many may be used, which can serve as a cost-metric in the design of an optical logic network. Furthermore, as an optical signal, the sub-function may still only be switched and routed further using primary inputs to the network.

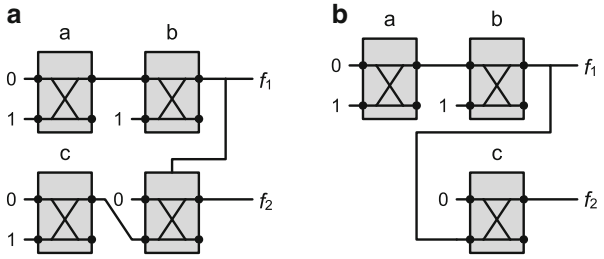


**Fig. 8.5** Two configurations for $f_1 = a + b$ and $f_2 = c \cdot (a + b)$. (**a**) Incompatible design. (**b**) Compatible design

### 8.4.2 *Garbage Outputs*

A "garbage output" is a waveguide output that is not connected to a receiver (a function output), i.e. it is left unused. These unconnected outputs cause problems because the signals, and the light/energy it carries, may interfere with the operation of the network if not properly "disposed." This is demonstrated in Fig. 8.6, which is the visual output of a Finite Difference Time Domain (FDTD) simulation [38] of an MZI device. The FDTD simulation technique models wave propagation through a (discrete) wave medium; Fig. 8.6 depicts the MZI device routing light from the top-left input to the lower-right output. The lower-right output of the device is left unconnected. Light arriving at this unconnected output can do a number of things, including dispersing into the substrate as noise and heat (as shown in the figure as ripples in the substrate) and/or reflecting back into the device, interfering with other signals.

These unconnected, or "garbage" outputs are problematic, and must be properly routed to the edges of the substrate where they can be dispersed away from the logic devices. The additional waveguides needed for this can cause congestion and complicate the overall physical routing of a network. *Every crossbar gate output that is left unconnected is a garbage output.* For example, the network shown in Fig. 8.5b would require three garbage outputs to be routed to the edges of the substrate, leading to a far-less compact design. Minimizing gate count, in general, reduces the number of garbage outputs, and is an important part of any synthesis procedure.

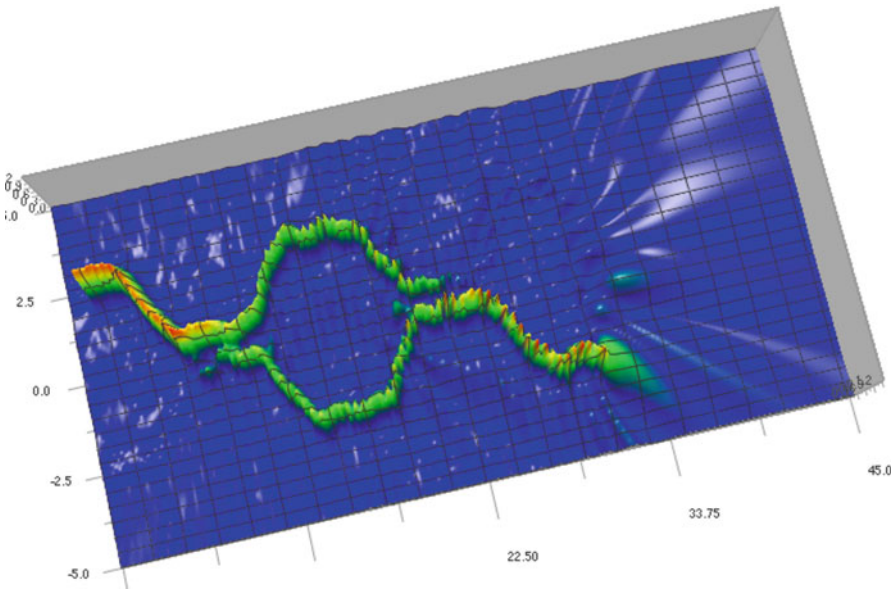


**Fig. 8.6** Dispersion of light into the substrate from a garbage output

With these constraints in mind, we now explore two basic design styles/methods for creating optical crossbar logic networks: BDD-based design and Virtual Gate design. We show how these design styles operate, and highlight their abilities, as well as limitations. These limitations motivate more advanced approaches using Boolean decomposition as a means to derive designs that may be more optimal and beyond the ability of the other approaches to optimize for. All these described methods lend themselves to automation, and provide a comparison of these approaches near the end of the chapter, using metrics which are described in the coming sections.

### *8.4.3 BDD Based Design*

The $2 \times 2$ crossbar can be modeled as two multiplexers with complemented inputs. As multiplexers, each crossbar gate effectively implements Shanon's expansion in one variable:

$$f = \bar{x} f_{\bar{x}} + x f_x \tag{8.4}$$

$$output_f = \bar{s} p + s q \tag{8.5}$$

$$output_g = s p + \bar{s} q$$

We can therefore utilize logic structures that employ Shanon's expansion, namely (Reduced Order) Binary Decision Diagrams (BDDs) [39] for direct implementation using crossbar gates.

Consider the ROBDD in Fig. 8.7a, which implements two functions: $f_1 = ab + c$ and $f_2 = \bar{a}b + c$, using variable order $a \prec b \prec c$. A dashed line indicates the negative cofactor, and a solid line the positive cofactor, which are connected to the $p$ and $q$ ports of a gate respectively. This is reflected in Fig. 8.7b. A crossbar network can therefore be technology-mapped from the BDD. The BDD's variable-switched function form directly maps to crossbar gate networks, and does not violate our crossbar model. In addition, the properties of the resulting network are also directly related to the properties of the BDD structure, including the effects of variable ordering on the canonical structure of an ROBDD.

#### 8.4.3.1 Salient Features

A BDD-based crossbar network will, in general, have a number of garbage outputs equal to the number of nodes present in the BDD. The physical aspects of crossbar gates also mean that networks cannot take advantage of ROBDD extensions such as complemented edges as the signal in a waveguide cannot be "inverted" without extra hardware; complemented functions will need to be derived as separate BDD

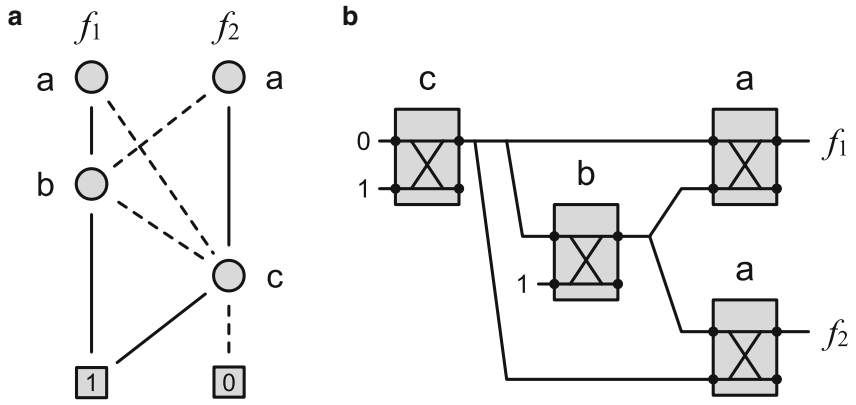


**Fig. 8.7** BDD-based design for $f_1 = ab + c$, $f_2 = \bar{a}b + c$. (**a**) BDD Graph; (**b**) Resulting BDD-based design

function. Common subexpression extraction is possible in the form of shared functions is possible through the use of splitters; however, the effects of the signal degradation must be accounted for.

BDD-crossbar networks are relatively path-delay balanced, as they have a feed-forward design topology. The longest path is computed as:

$$l_{max} = h \cdot l_0 \tag{8.6}$$

where $h$ is the height of the BDD graph.

Where BDD-based design suffers is in the number of garbage outputs produced by the approach. Each gate has the potential to produce a garbage output that must be accounted for through routing or a light absorbing structure. The canonical structure of ROBDDs can also lead to networks of extremely large gate counts for a given function. Though BDD-based design is attractive for its predictable signal delay, the number of garbage outputs and unpredictability of logic composition in terms of gate counts leads us to abandon this logic composition method for crossbar gate logic. We therefore investigate a composition methodology using "virtual gates."

### *8.4.4 Virtual Gates Based Design*

Consider the device networks depicted in Fig. 8.8. We denote these logic composition functions "Virtual Gates" (VGs). A *virtual gate* (VG) is—functionally and conceptually—a crossbar gate that is switched by a *function,* not necessarily a primary input. The gate is "virtual" in the sense that it is a black box for a function composed of "real" gates—those driven by primary inputs—as well as other virtual gates. A novel form of nesting can be used to compose VG function

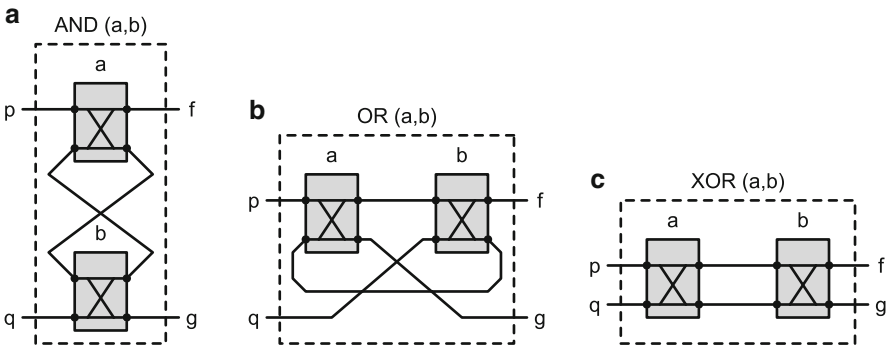


**Fig. 8.8** Virtual gate functions for 2-input Boolean operators. (**a**) AND; (**b**) OR; (**c**) XOR

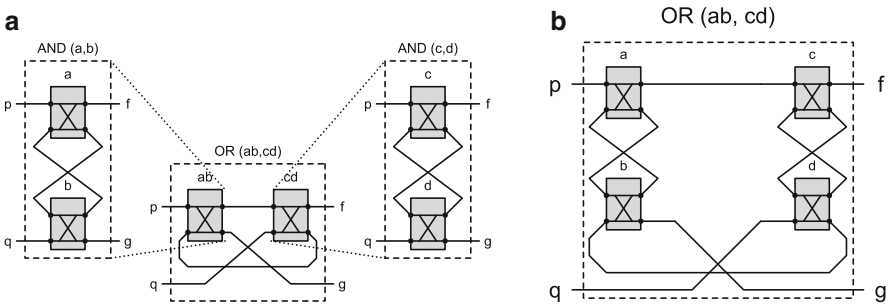


**Fig. 8.9** Composing functions with virtual gates. (**a**) Virtual gates implementing $f = ab + cd$; (**b**) Resulting network

implementations, where Boolean operators are implemented by replacing child gates with other gates, a real or virtual.

A given VG implementation comprises two input waveguide ports **p** and **q** connected by waveguides and crossbar gates to two output ports **f** and **g**. The nesting operation comprises the Boolean operator forms depicted in Fig. 8.8, and is illustrated in Fig. 8.9a where two AND virtual gates are nested within an OR virtual gate, creating the final function $ab+cd$. Evaluation of a VG, given a primary input assignment, involves assigning **p** and **q** inputs logical 0 and 1 respectively, and applying *cross* or *bar* configurations to gates as defined in Fig. 8.4. The output of the function is detected at **f**, with $\mathbf{g} = \neg\mathbf{f}$.

The process of composition is illustrated in Fig. 8.9a, where a function $f = ab + cd$ is implemented by replacing (or *nesting*) the gates of an OR function with VGs implementing $a \cdot b$ and $c \cdot d$. The result is depicted in Fig. 8.9b.

While it may seem strange to see feedback loops in device designs, the physical devices can indeed implement self-feedback. As an experiment, the model for the AND gate depicted in Fig. 8.8a was simulated in a 2D FDTD simulator

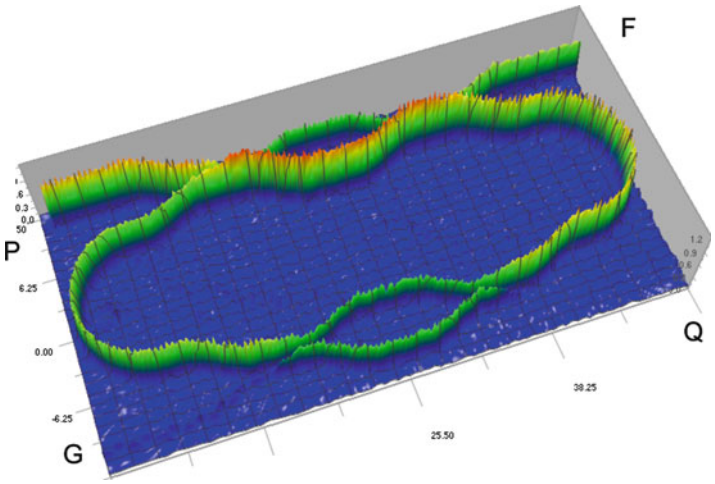


**Fig. 8.10** FDTD simulation of an AND virtual gate

OptiFDTD®by Optiwave Software; the visual output[2] of which can be seen for $a = 1, b = 0$ in Fig. 8.10. The signal from the top-left crosses in the top gate, but passes through in the bottom gate, returning to the top gate where it crosses again to appear in the top-right output.

#### 8.4.4.1 Salient Features

Networks composed of virtual gates have exactly two optical inputs $p$ and $q$ and two outputs $f$ and $g$, as the entire network is, in itself, a virtual gate; in addition, for a given function, a maximum of one garbage output is created. The existence of a complete logic enables virtual gates to implement any logic function using crossbar gates *comprising only primary inputs.* This includes factored functions, and any other single-output representation using Boolean operators. Control signals ($S$) are connected via the primary inputs of the function. The $f$ port implements the function, and $g = \neg f$. Furthermore, the total number of *real* gates is the number of primary literals in the original logic expression the network is derived from.

[2]Note that there are differences from the virtual gate diagram: the bottom two ports are swapped because the waveguides are not crossed in the center, and that the "light" source is positioned at the $p$ input rather than at the $q$ input.

Virtual gates also suffer from very unbalanced signal paths, depending on the state of the switches, with the potential for a signal to traverse every waveguide present in a VG network. The maximum signal path $l_{max}$ is roughly computed as:

$$l_{max} = 2 \cdot p \cdot l_0 \tag{8.7}$$

where $p$ is the number of operators in the virtual gate, and $l_0$ is a "unit length" of waveguide. This is based on the fact that all virtual gate operators connect two gates (virtual or real) by two waveguides, and a signal could possibly traverse all paths to reach the destination. For example, the network in Fig. 8.9a would have a $2 \cdot 3 \cdot l_0 = 6l_0$ long maximum signal path, which is close to the longest possible signal path from $p$ to $f$ with variable assignment $\{a, b, c, d\} = \{1, 0, 1, 0\}$ at $5l_0$. The value $l_{max}$ is a reasonable rough estimate; it can be further refined by estimating routing distances for operators and physical network topology.

#### 8.4.4.2 Expression Sharing

The major limitation of designing with virtual gates is that the nesting of gates *prevents the extraction/sharing of arbitrary common sub-expressions (CSE)*. For example, in Fig. 8.11 one cannot simply share the $ab$ term from $f = ab + bc$ for use with another gate; assignments such as $abcd = \{1, 1, 1, 1\}$ will cause all crossbar gates to assume a cross-configuration, isolating the top input of the $h$-gate from the optical inputs of the network. In effect, any operator employing feedback for its inputs can produce an undefined state. Only the XOR operator does not exhibit this behavior as it has no feedback, but XOR-based CSE is not well studied in contemporary logic synthesis. To address this issue particularly for optical logic synthesis, we investigated a XOR-based functional decomposition technique for CSE, and implemented it within our virtual-gate paradigm. Interested readers may refer to our publication [2] for more details.

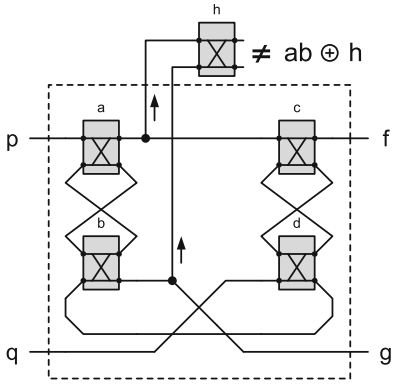


**Fig. 8.11** Internal functions of virtual gates cannot be shared

## 8.5 Physical Synthesis Methodology for Integrated Optics

Subsequent to high-level and logic design, the need for automated design space exploration and optimization also begins to appear for *physical synthesis of integrated electro-optical systems*. For this reason, the Electronic Design Automation (EDA) community is investigating how automatic design space exploration techniques can be adapted to the photonics domain [40–44]. Such circuits are complex in their device interconnections, often featuring high device counts and large amounts of feedback loops. These designs comprise a set of pre-designed optical devices—modulators, switches, splitters—placed on a planar substrate, connected together via waveguides. For example, in our previous work [2], our multi-level logic synthesis methodology for implementing logic demonstrates how optical designs can scale beyond the ability of custom design. The physical synthesis of such applications now has to be addressed. For this purpose, we describe the design constraints, layout models and methodologies for integrated optics automation.

### *8.5.1 Design Constraints*

At the physical automation level, we identify signal power and substrate area as our core guiding metrics.

#### 8.5.1.1 Signal Power

Signal power is the primary guiding metric in our methodology. All devices, including bulk waveguides, have insertion losses, measured in decibels (dB). Our assumption is that these losses are pre-characterized through device-analysis (FDTD, etc.) for the following type devices:

- **Pre-designed devices [device-specific]** (e.g. modulator devices, switches, splitters, etc.). Losses are characterized from inputs to outputs. For example, waveguide splitters have their signal power from the input effectively halved at each output (a 3dB loss).
- **Waveguide crossings [0.1–0.2 dB / crossing]** Per-crossing losses are on the order of 0.1–0.2dB per crossing [45–47], affecting both crossing waveguides.
- **Waveguide bends [0.001–0.3 dB / bend]** Losses dependent on inherent waveguide properties (materials, geometry, etc.), radius of curvature of the bend, and surface roughness due to fabrication [48–50].
- **Bulk waveguides [0.01–2 dB / cm]** As these losses are extremely low (dB per *centimeter*, e.g. 0.03dB/cm [51]), we consider bulk waveguides essentially *lossless*.

Losses due to the presence of pre-designed devices are effectively fixed. Therefore, the design automation problem concerns itself with designing within the permitted losses *between* such devices–the routing fabric. We identify three main routing loss mechanisms in descending importance: 1) *waveguide crossings,* which induce a relatively large fixed loss per crossing; 2) waveguide bends, especially bends close to the minimum radius of curvature; 3) bulk waveguides, which generally have low losses; however surface roughness can induce losses over larger distances for smaller waveguides.

#### 8.5.1.2 SOI Waveguides

Si-photonic waveguides, with their large refractive index differentials, provide strong mode confinement, and therefore bends can be much sharper, saving area. While waveguide bends can be effectively lossless given a large enough radius of curvature, accepting small per-bend losses can be advantageous in reducing the area occupied by a bend [49]. The choice of minimum routing grid size can therefore affect the weighting of metrics used to guide the routing, whether losses due to bends, waveguide crossings [45], or area.

#### 8.5.1.3 Area

Many optical devices, such as those used for switching, are designed such that their input and output ports appear on only opposing sides. This feed forward device design often extends to the device networks as a whole, resulting in overall networks that are very wide. Wide substrates may not be desirable when integrating optics into designs, and a more suitable aspect ratio may need to be enforced. The side-effect of this is that devices must be rearranged on the substrate in a manner that can affect inter-device locality as well as increase waveguide routing complexity. This becomes an important part of the placement phase of our methodology.

### *8.5.2 Methodology*

We propose the following methodology for the overall physical design problem for integrated optics. As depicted in Fig. 8.12c, pre-designed optical devices are represented as rectangular blocks (a) that are arranged (placed) in fixed-width columns (b). Such a placement gives rise to *vertical routing channels* (c), which are routing regions that separate the placed devices. Waveguides are routed between devices at “ports” (d) that face the channels. For ports in different columns, these waveguides may pass through *horizontal routing channels*, as depicted in (e). While the substrate is planar, waveguides may also cross each other perpendicularly (f) without sharing signals.

Overall, the physical design methodology requires that the problem be solved in three steps:

- Placement of optical switching devices into columns, i.e. a grid-based layout.
- Global routing of waveguides that connect these devices. Global routing solution will determine the overall routing topology of all the nets.
- Detailed routing of all the nets, which manifests itself as a well-defined channel routing problem.

While this methodology is analogous to that employed in the VLSI domain, the design and optimization constraints imposed by the optical technology are different. Any CAD solution to this problem will have to incorporate such technology specific constraint models and design rules.

### *8.5.3 Device Placement*

Pre-designed optical devices are placed into columns. Consider the layout of devices in Fig. 8.12a. While devices maintain ports on only their left and right sides, connections may be made to any other device in the network by routing through vertical columns and between columns. In such a manner, connectivity is preserved, but the overall network has a smaller aspect ratio. Placement techniques, such as those used for row placement and chip floorplanning [52], can therefore be employed for placing devices within an optical substrate. The placement of devices into columns enables us to utilize routing techniques designed for such placement strategies. In our applications we use the Capo placer [52] to arrange devices in rows given a specific aspect ratio. Connected devices are localized as much as possible, reducing congestion. Such a placement simplifies the subsequent signal-loss constrained global and local routing methods for integrated optics.

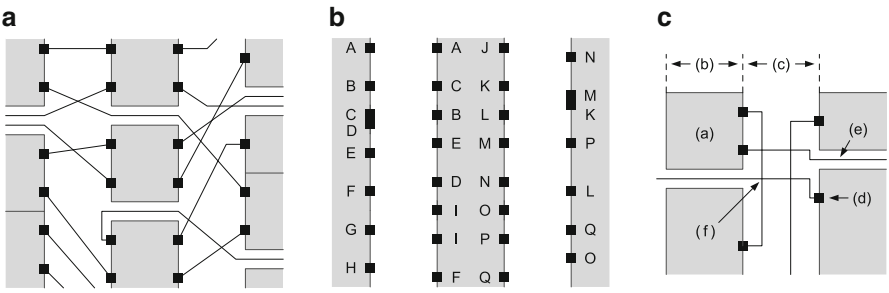


**Fig. 8.12** Stages of the Physical Design Methodology. (**a**) Columns of optical devices, and global routes; (**b**) Resulting channels for detailed routing; (**c**) Ports, routes and channels

### 8.5.4 Global and Detail Routing

Global routing determines the high level topology a signal may take through the channels from source to destination. The chosen routes induce bends and crossings with other nets. The optimization goal of the global router is to minimize losses due to waveguide crossings and waveguide bends. In addition, global routing also takes into account overall net lengths and routing congestion.

Given a device placement, a graph is derived from the vertical and horizontal channels separating the device blocks. Nodes are placed at locations where ports are located, and where horizontal and vertical routing regions meet. Any device placement topology may also be used; however, we assume a channel-based placement is used. In a channel-based placement, such as depicted in Fig. 8.13a, nodes and edges are first derived for the vertical channels from the location of device ports and horizontal channels. These channels are then connected to other channels via horizontal inter-channel edges, such as depicted in Fig. 8.13b.

The presence of inter-route loss can affect signal quality more than route length, forcing longer routes to be exercised. Consider the nets in example Fig. 8.13c, where a net $q$ can utilize one of two distinct routes (1) and (2). Route (1), though shorter than route (2), must cross the chosen route for $p$; to avoid the crossing, route (2) could be utilized. Route (2), however, crosses over the chosen route for $r$. Should route $r$ have less stringent loss constraints than $p$, route (2) may be chosen over (1), despite a longer overall path. Ultimately, the final route choice is derived from a combination of all loss factors.

The global router provides a set of vertical routing channels with net/port connectivity, such as depicted in Fig. 8.12c. At this stage, detail routing is performed, determining the actual placement of horizontal and vertical connections

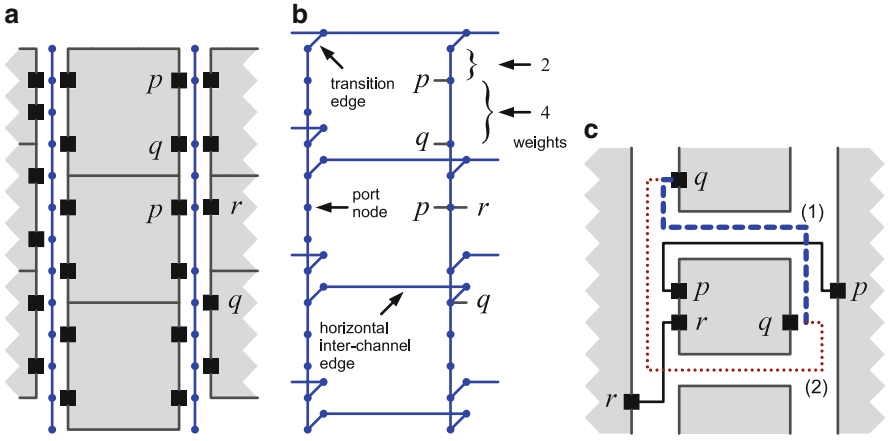


**Fig. 8.13** Construction of routing graph from channel layout. (**a**) Vertical nodes and edges from layout; (**b**) Complete routing graph; (**c**) Different route choices inducing different crossings

within the vertical channel. Consider the routing channels depicted in Fig. 8.12b. The channel routing area is a grid between the pins on either side of the channel, where waveguides are routed between pairs of pins. Traditional VLSI channel routing seeks to minimize the area of a fully routed channel. In our channel routing techniques, we optimize for crossings and bends, with channel height a subsequent metric. The details of our channel routing approaches are found in Sect. 8.7.

### *8.5.5 Routing Grid Realization*

The result of routing algorithms must be transformed into the physical waveguide layout. This entails converting the routing grids into waveguide bends satisfying the material bend constraints, which are generally defined in terms of minimum radius of curvature and coupling distance.

A rectilinear routing grid is realized as waveguides by converting all 90° grid transitions to 90° waveguide bends. This requires that such bends complete within a quarter of the routing grid. This is illustrated in Fig. 8.14b where a horseshoe-shaped bend utilizes two 90° waveguide bends, each taking place within a quadrant of the routing grid. This mapping represents the smallest grid that can be suitably used for complete routing grid flexibility.

The physical routing can also exploit the spacing between curves at the corners of grids. These "knock-knee" style bends, as depicted in Fig. 8.14c, enable additional track sharing—potentially reducing the overall number of tracks needed for a routing. For example, in the solution depicted in Fig. 8.15b, the knock-knee bends between signals C-E, F-G, D-I, and G-J allow each respective pair to occupy the same track, with the net effect of reducing the total number of tracks to four (4). Routing techniques enabling knock-knee track sharing must account for shared rectilinear grid locations, e.g. Fig. 8.15a, during channel construction.

The waveguide's minimum radius of curvature $r$ has an important role in determining the routing grid's minimum size. In some cases, $r$ may be chosen for

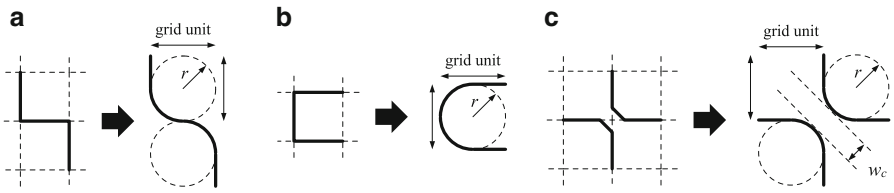


**Fig. 8.14** Conversion of grid units to waveguide curves. (**a**) S-shaped grid to bends; (**b**) Horseshoe-shaped grid to waveguide bends; (**c**) Knock-knee grid with 90° bends, radius of curvature $r$, and minimum coupling distance $w_c$

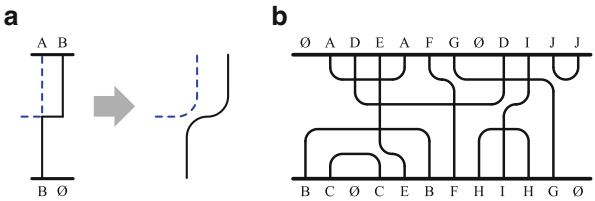


**Fig. 8.15** Knock-knee model for grid spacing. (**a**) Shared grid corners enable knock-knees; (**b**) Channel routing incorporating knock-knee bends (Four tracks, eight crossings)

area reduction, at the expense of per-bend losses [49]. For example, to enable knock-knee routing patterns, the distance $w_c$ in Fig. 8.14c must be sufficient to prevent significant coupling between waveguides.

## 8.6 Global Routing for Integrated Optics

Global routing provides the high-level overall placement of routes throughout the device network, while detailed routing determines the localized routing necessary to complete routing. In the VLSI domain, global routers are mostly concerned with 1) wire-length, 2) congestion, and 3) overflow—all of which are interrelated. For integrated optics, some of these aspects, such as wire-length, are deemphasized and in their place the router must also account for signal loss in terms of crossings and bends. In addition, optical routing must be performed on a *single routing layer.*

Despite the advanced state of VLSI global routers [53–56], their applicability is limited within the optical routing domain. VLSI routing is inherently multi-layer, and VLSI global routers are designed to take advantage of multiple layers in order to produce routing solutions. As such, global routers are also not designed to minimize crossings. Minimization techniques for metrics such as vias, though applicable to bends, cannot be applied to waveguide crossings, as a single via can facilitate multiple crossings due to multiple-layers. We therefore investigate global routing specifically for integrated optics.

### *8.6.1 Routing Using Mixed Integer Linear Programming*

We conceptually frame global routing through mixed integer linear programming (MILP). Each net $i$ has a set of $n_i$ candidate routes. Only one route $k$ may be chosen for a given net, and that route has a cost associated with it $\alpha_k^i$. The cost $\alpha_k^i$ is formulated in terms of signal loss: *static* losses induced by bends and length, and *inter-route* losses caused by crossings between of different routes of different nets. More formally

$$\alpha_k^i = \alpha_{k,static}^i \cdot x_k^i + \sum_{j}^{j \neq i} \sum_{k=1}^{n_i} \sum_{l=1}^{n_j} \alpha_{k,l}^{i,j} \cdot x_{k,l}^{i,j} \tag{8.8}$$

$$\alpha_{total}^i = \sum_{k=1}^{n_i} \alpha_k^i \qquad \alpha_{total}^i < \alpha_{max}^i \tag{8.9}$$

where $x_k^i = 1$ if net $i$ uses route $k$, otherwise 0, and $x_{k,l}^{i,j} = 1$ if respective nets $i$ and $j$ use routes $k$ and $l$, respectively, otherwise 0. A loss-coefficient $\alpha_{k,l}^{i,j}$ is the inter-route loss associated with those two routes. $\alpha_{total}^i$ constrains the maximum losses acceptable for the given net. Equations (8.8) and (8.9) provide the basic structure for optimization. What remains is to determine the coefficient weights $\alpha_{k,static}^i$ and $\alpha_{k,l}^{i,j}$.

### 8.6.2 Route Analysis

Routes are defined on a graph $G$ comprising a set of grid-edges $E$ derived from layout of the device placement. For example, the routing regions between devices in Fig. 8.16 produce a set of edges connecting between port-endpoints.

Static route costs $\alpha_{k,static}^i$ are derived from the set of edges $E_k^i$ that a route traverses, comprising a sum of edge-cost weights. Waveguide bends also have a cost associated with them, as they can be a significant loss mechanism. In order to penalize their use, we modify the graph by adding weighted *via-edges* connecting between vertical and horizontal edges, as depicted in Fig. 8.16. Though straight-waveguide losses at these scales are negligibly small, longer routes have a greater potential for intersecting other routes, potentially causing more crossings. Edges are therefore weighted according to their length in the substrate to favor locality. For simplicity, a basic approach for choosing the set of candidate routes for a given net is to choose the set of routes with the least static route-costs. Other metrics, such as potential edge-capacity utilization, and diversity of routes may also provide better route candidates.

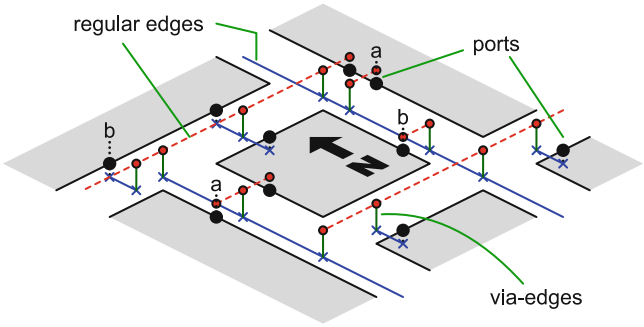


**Fig. 8.16** Routing graph derived from device placement

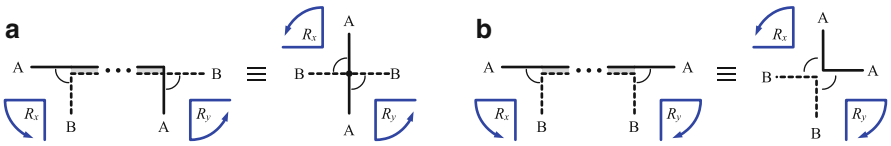


**Fig. 8.17** Functionally equivalent configurations of path-endpoints and their rotations. (**a**) Same direction ($A \rightarrow B$) induce crossings; (**b**) Opposite directions require no crossings

Given the sets of candidate routes, inter-route losses $\alpha_{k,l}^{i,j}$ are determined by pairwise analyzing candidate routes for different nets to determine whether routes cross. The given pair of routes may have multiple shared sets of edges (paths) where a crossing may occur. A crossing, if it is required, will occur only once for a given shared path; we can treat the shared path edges as a single node that retains the crossing of the original.

Consider the two nets depicted in Fig. 8.17, where route $A$ and $B$ share edges in the middle. At the endpoints of the shared edges, the two routes diverge; we denote these as diverging endpoint edges (DEEs). For a given endpoint, *rotation* is the direction a route's DEE must rotate towards DEE of the other route, pivoted on their shared node, on the arc that does not contain the shared route edges. For example, in the left path endpoint of Fig. 8.17a, the DEE of $A$ rotates counter-clockwise towards the DEE of $B$. Likewise, on the right side, DEE of $A$ again rotates counter-clockwise towards the DEE of $B$. *A crossing is only required if-and-only-if the rotation of both endpoints is the same, otherwise no crossing is required.*

#### 8.6.2.1 Minimization Function

With the per-route and per-net equations in place, and their coefficient weights determined, the final minimization function is a sum of all net costs. The minimization function is implemented as

$$minimize : \sum_{i=1}^{m} W_i \cdot \alpha_{total}^{i} \tag{8.10}$$

where $m$ is the total number of nets and each $W_i$ is a per-net weight to prioritize certain nets over others during optimization. Though not detailed here, congestion can be accounted for by the number of routes that utilize given edges in the routing graph.

The presented global router is relatively basic as compared to contemporary VLSI routers; however, it accounts for many aspects specific to single-layer integrated optic routing. For VLSI routers, crossing minimization at a global level is not incorporated. Therefore, instead of utilizing VLSI-centric global routers, we

developed our own global router for integrated optics. Subsequent to the global routing, the final detail routing problem is formulated and solved as a channel routing problem.

## 8.7 Channel Routing for Integrated Optics

In column-based optical device placement, the detailed routing problem manifests itself as a *channel routing* problem, where (Silicon) optical waveguides are fabricated on a planar substrate and are connected to devices at the ends of the channel. Planar routes require waveguides to bend (curve) and cross each other—causing loss of signal power. Channel routing techniques are therefore needed that minimize waveguide crossings and bends. We present a channel router based on crossing-aware, graph-constraint track-assignment. The router minimizes signal loss as a function of waveguide crossings and bends within the channel, while also reducing area.

### *8.7.1 Optimization Objective*

The primary optimization objective in our routing formulation is *signal loss* minimization. Within the channel, this is achieved by: 1) minimization of the total number of waveguide crossings; and 2) minimization of the number of waveguide bends. Minimization of the number of tracks (channel height) is the subsequent secondary objective.

We optimize for the *total signal loss within the channel* due to optical feedback within the system. A signal may be routed such that it enters a given channel multiple times and may cross multiple other nets. Therefore, instead of minimizing losses on a per-net basis, we minimize for *total* losses within a channel (Fig. 8.18).

### *8.7.2 Left-Edge-Style Channel Routing*

Traditional left-edge-style channel routers [57–59] represent the channel routing problem using horizontal and vertical constraint graphs (HCG, VCG). An alternate representation of the HCG is the zone representation, which is derived from the HCG, where every zone is defined by a maximal clique. The number of signals in the largest zone is the lower bound on the number of tracks needed for routing. These graphs encode constraints on how tracks may be assigned to nets in the channel. Consider the channel routing problem depicted in Fig. 8.19a. The resulting zone representation is depicted in Fig. 8.19b. Likewise, the VCG for the problem is represented in Fig. 8.20a.

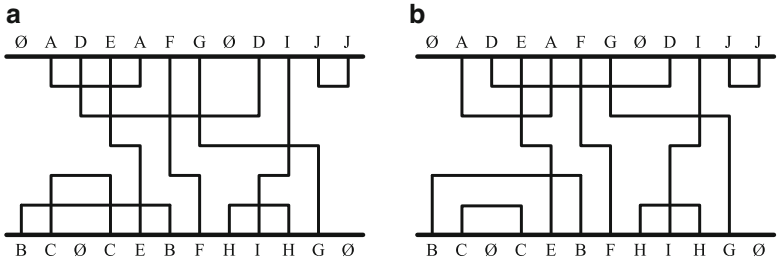


**Fig. 8.18** Channel routing solutions under differing constraints. (**a**) Track-optimized (five tracks, ten crossings); (**b**) Crossing-constrained (five tracks, eight crossings)

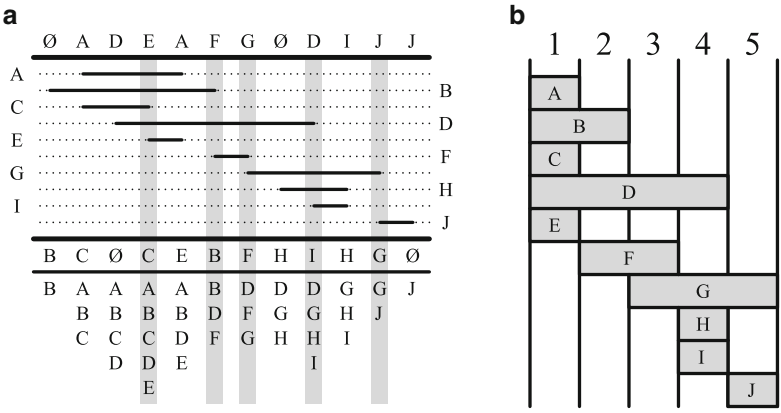


**Fig. 8.19** Horizontal constraints and zone representation. (**a**) Five maximal subsets of signals; (**b**) Resulting five zones

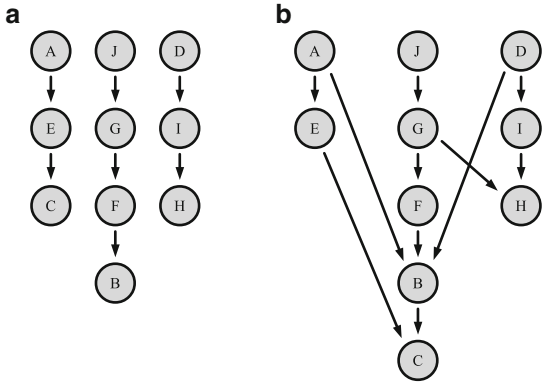


**Fig. 8.20** Crossing-constraints modifications to the VCG. (**a**) Original VCG; (**b**) With crossing constraints

A net may be assigned to a track should it have no descendants on the VCG, and have no overlapping zone conflicts with previously assigned nets on a given track. Nets are removed from the VCG as they are assigned to tracks. When a track cannot contain more nets, a new track is created and the process is repeated until no more nets are left for assignment.

Multiple nets can be candidates for assignment to a given track, each with different horizontal overlaps. Therefore heuristics are used to choose which nets are assigned first. One of the simplest is a greedy heuristic used in *constrained left-edge channel routing* [57], where the left-most available nets in channel are assigned first to tracks. This can lead to sub-optimal track-utilization; more sophisticated heuristics analyze the graph structure for better results, such as [59], which attempts to reduce the longest path in the VCG for better track utilization. We refer to the class of track assignment algorithms above generically as "left-edge-style" channel routing. The approach we describe below can be incorporated into any such techniques.

#### 8.7.2.1 Crossing-Constrained Track Assignment

Figure 8.18a depicts the output of a (VLSI) left-edge 2-layer channel router, and Fig. 8.18b, a channel routing constrained for crossing-minimization. Both solutions are minimal in terms of tracks; however, the total number of crossings in Fig. 8.18a is 10, compared to 8 in Fig. 8.18a. The discrepancy in the number of crossings is attributed to the two crossing points caused by nets $B$ and $C$. By forcing $C$ to appear below $B$, two crossings are avoided. However, transforming from Fig. 8.18a to b is not as simple as moving net $C$ below $B$, not if track height is to be kept minimal. Crossing minimization must therefore be encoded into the routing process itself as constraints.

We constrain the channel routing problem to favor crossing minimization. The VCG is modified such that avoidable crossings impose vertical constraints on the net ordering. Only nets that share zones have the possibility of crossing, and pairwise analysis takes place after the zones are derived.

A crossing constraint is only encoded into the VCG if a crossing can be avoided. For example, the pair of nets in Fig. 8.21c would not normally be constrained in the VCG; however, a net crossing can be avoided if $B$ is assigned a track above $A$. Therefore, an edge connecting $B$ to $A$ is added to the VCG. Conversely, the two nets in Fig. 8.21b cannot avoid crossing, and therefore no constraint is added.

We introduce the concept of *pin-rotation* to detect avoidable crossings. If we were to map the pins of nets on a unit circle, a crossing is unavoidable if rotating from one pin to the next is not possible without first passing through the pin of another net. Consider the nets depicted in Fig. 8.21a. Collapsing the shared horizontal region, and considering the areas Fig. 8.21a(1) and a(2) shows how pins of a given side rotate with respect to each other (clockwise/counter-clockwise) around an axis fixed at the center. In the case of Fig. 8.21a(1), the rotation of the left pin of $A$ to the left pin of $B$ is *counterclockwise,* and likewise the pins on the right-side also rotate in the same *counterclockwise* direction. If the pins on both left and right terminals rotate in the same direction a crossing is unavoidable.

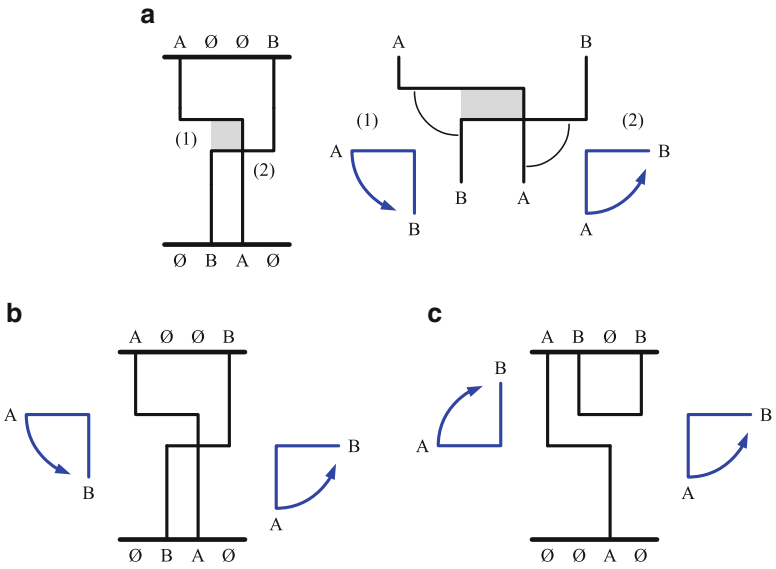


**Fig. 8.21** Crossing detection via rotation from $A$ to $B$. (**a**) Rotation direction with respect to pin locations; (**b**) Same rotation direction $\Rightarrow$ Unavoidable crossing; (**c**) Opposite rotation directions $\Rightarrow$ Avoidable crossing

More formally:

$$X^{left}_{A,B,CW} = \begin{cases} X^{left}_{B,top} & \textbf{if } C^{left}_{A} < C^{left}_{B} \\ \neg X^{left}_{A,top} & \text{otherwise} \end{cases} \tag{8.11}$$

$$X^{right}_{A,B,CW} = \begin{cases} X^{right}_{A,top} & \textbf{if } C^{right}_{A} < C^{right}_{B} \\ \neg X^{right}_{B,top} & \text{otherwise} \end{cases} \tag{8.12}$$

$$X_{\text{avoidable}}(A, B) = \left( X^{left}_{A,B,CW} \neq X^{right}_{A,B,CW} \right) \tag{8.13}$$

where $C^{left/right}_{N}$ is the integer-valued column-position of a pin of net $N$ on a given side (left, right); the Boolean variable $X^{left/right}_{N,top}$, using the same notation, denotes whether that pin resides on the top side of the channel. Equations (8.11) and (8.12) utilize the horizontal relationships of pins and their channel-sides (top/bottom) to determine the *clockwise rotation* (CW) of a given pair of *left* or *right* pins for nets $A$ and $B$, rotating from $A$ to $B$. A crossing is avoidable only if left and right rotations are *not* the same, the result of (8.13).

For example, in Fig. 8.21a, consider the left side of the shared span Fig. 8.21a(1):

- The variables $C_A^{left}$ and $C_B^{left}$ are the column positions of the respective left-terminals of nets $A$ and $B$. In the example, $C_A^{left} = 1$, $C_B^{left} = 2$.
- $C_A^{left} < C_B^{left}$ implies $X_{A,B,CW}^{left} = X_{B,top}^{left}$ from (8.11).
- The left pin of net $B$ is *not* on the top side of the channel ($X_{B,top}^{left}$ = **false**). Therefore, the left side of the pair of nets is *not* rotating clockwise from $A$ to $B$, i.e. $X_{A,B,CW}^{left} = X_{B,top}^{left}$ = **false**.
- On the right side of the shared span Fig. 8.21a(2), $C_A^{right} < C_B^{right}$. This condition implies that $X_{A,B,CW}^{right} = X_{A,top}^{right}$ = **false**. The right side is therefore also *not* rotating clockwise from $A$ to $B$.

Having the same direction of rotation ($X_{A,B,CW}^{left} = X_{A,B,CW}^{right}$ = **false**) implies that a crossing is *unavoidable,* as determined by (8.13); this is reflected in the figure.

Applying crossing constraints to the problem depicted in Fig. 8.19a results in the VCG depicted Fig. 8.20b. As compared to the original VCG Fig. 8.20a, the crossing-constrained VCG is more heavily constrained, ensuring that unnecessary crossings do not occur, such as the double-crossing of nets $B$ and $C$ in Fig. 8.18a.

### 8.7.2.2 Knock-Knee Track Sharing

Though the modified VCG is effective in preventing waveguide crossings, the additional constraints can affect overall track height, and may produce a worse solution in terms of number of tracks. However, we observe that the bend geometry of optical waveguides can be exploited to further reduce channel height. This is discussed below.

Consider the two nets in Fig. 8.22a. The endpoints of the two nets occupy the same column and therefore net $A$ should be placed above $B$ in the VCG. However, given the same track, the two nets would intersect at a corner of each horizontal

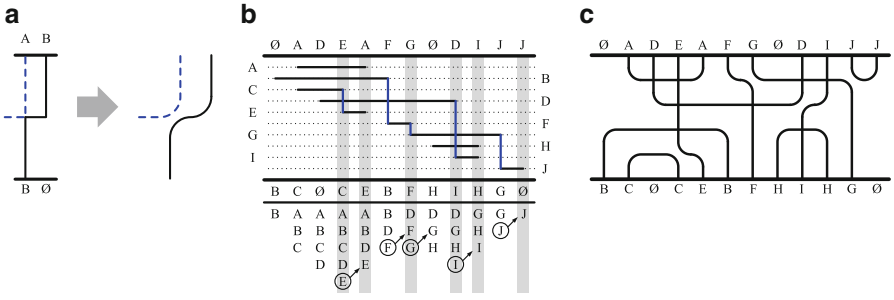


**Fig. 8.22** VCGs for Fig. 8.19a and knock-knee extension. (**a**) Knock-knee implementation; (**b**) Knock-knee-constrained zone representation; (**c**) 4-track routing solution utilizing knock-knees

span—a *knock-knee*. In VLSI, this situation is untenable, and different tracks would need to be assigned to each net. However, for waveguides, the minimum grid spacing for a channel can permit knock-knees in the routing grid. This is depicted in Fig. 8.22a, where a track is shared between the two nets without overlap.

A knock-knee occurs where one net ends and another begins, e.g. nets $C$ and $E$ in Fig. 8.22c. During zone construction, at columns where knock-knees appear, the net that is beginning its horizontal span is only added to the *subsequent* column set, rather than the current column set under consideration. For example, in Fig. 8.22c knock-knee signals $E$, $F$, $G$, $I$, and $J$ are removed from the marked columns and only appear in the subsequent columns.

The effect of this column change on the resulting zones is demonstrated in Fig. 8.22b, where there are six (6) zones rather than the five (5) from the previous zone analysis Fig. 8.19. Despite containing an additional zone, the largest column set now contains *one fewer* net than the original, resulting in the 4-track solution depicted in Fig. 8.22c.

Overall, the effect incorporating knock-knees into a routing solution is that two knock-knee nets can now occupy separate zones, and therefore can be placed on the same track. Additional zones may be created; however, those zones are equal in size or *smaller* in terms of nets—*potentially reducing the lower bound on the number of tracks required for routing*.

### 8.7.2.3 Cycles Induced by Crossing Constraints

Crossing constraints can induce cycles in the VCG. Consider the three nets depicted in Fig. 8.23a. Without crossing constraints, nets $A$ and $B$ would be unconstrained, and no cycle would occur; however, due to the constraint edge between $B$ and $A$ such a cycle occurs. Cyclic constraints cannot be routed without additional tracks and require "doglegging" to complete routing [58]. In order to avoid crossings, the routes for $A$ and $B$ are converted into doglegging routes as depicted in Fig. 8.23b, utilizing the same columns as the original. Unfortunately, this results in an additional

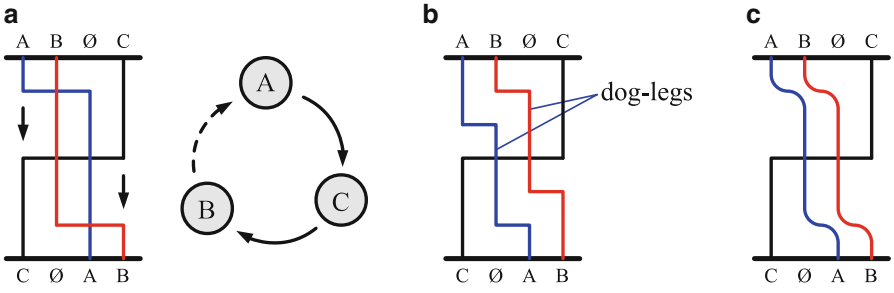


**Fig. 8.23** Cycles induced by crossing constraints. (**a**) Vertical cyclic constraints; (**b**) Dog-legging avoids crossings; (**c**) Knock-knees avoid additional tracks

two (2) tracks being added to the routing solution should spare tracks not be available adjacent to the cycle. However, in the presence of knock-knees, both the crossings, and the additional tracks can be avoided, as depicted in Fig. 8.23c. The experimental results show that knock-knees can have a marked difference in track utilization especially in the presence of cyclic constraints induced by crossings.

In our work, we have designed two detail routers based on the above channel routing and crossing-aware signal loss models. Our routers provide an effective means to automate optical waveguide routing and track assignment, with signal loss as the main metric. Interested readers can refer to our publication [60] for details on our algorithms and experimental results.

## 8.8 Conclusion

This chapter has described design automation for integrated optics. We have demonstrated photonics design automation through a building-block methodology with optics technology-specific constraints and objectives. Our design flow is broken into behavioral and physical synthesis stages. In the behavioral synthesis phase, we describe multi-level logic design and synthesis techniques for optical digital logic. Mach-Zehnder Interferometer (MZI) and ring resonators devices are employed as switching devices—connected with waveguides—to construct optical logic systems. A virtual gate based design methodology is introduced that enables device-minimal logic synthesis for such a technology.

Post logic synthesis, the logic network needs to be placed and the interconnection-network needs to be routed. For this purpose, we introduce a row/column based placement methodology that exploits the regularity of the MZI-based optical logic network. Post placement, a global and detail routing framework is presented that minimizes signal loss as the primary optimization constraint. Signal loss models are incorporated to account for insertion losses, waveguide crossing and bends incurred due to routing on a planar substrate. This work essentially demonstrates the feasibility of *silicon photonic design automation*, though significant research is needed to make silicon-photonics design technology widely applicable and scalable.

## References

1. Soref R. The past, present, and future of silicon photonics. IEEE J Sel Top Quantum Electron. 2006;12:1678–87
2. Condrat C, Kalla P, Blair S. Logic Synthesis for Integrated Optics. In: Proceedings of the 21st Edition of the Great Lakes Symposium on Great Lakes Symposium on VLSI, GLSVLSI '11, New York:ACM; 2011. pp. 13–18.
3. Condrat C, Kalla P, Blair S. Exploring Design and Synthesis for Optical Digital Logic. International Workshop on Logic Synthesis, 2010.

4. Caulfield HJ, Vikram CS, Zavalin A. Optical logic redux. Optik. 2006;117:199–209
5. Politi A, Matthews J, O'Brien J. "Shor's Quantum Factoring Algorithm on a Photonic Chip. Science. 2009;325:1221
6. Hardy J, Shamir J. Optics Inspired Logic Architecture. Opt Express. 2007;15:150–65
7. Caulfield et al. HJ. Generalized optical logic elements GOLEs. Opt Commun. 2007;271: 365–76
8. Ganapati P. Germanium laser breakthrough brings optical computing closer. Wired Mag. 2010
9. Blair S, Wagner K. Collision-based computing. Chapter gated logic with optical solitons. London: Springer, 2002. p. 355–80.
10. Shan A. Heterogeneous Processing: a Strategy for Augmenting Moore's Law (http://www.linuxjournal.com/article/8368). Linux J. 2006; 142
11. Dokania Rk, Apsel AB. Analysis of challenges for on-chip optical interconnects. In: GLSVLSI, GLSVLSI. New York: ACM; 2009. pp. 275–80.
12. Batten C, Joshi A, Stojanovic V, Asanovic K, Designing chip-level nanophotonic interconnection networks. IEEE J Emerging Sel Top Circuits Syst. 2012;2:137–53
13. Cianchetti M, Kerekes J, Albonesi D, Phastlane: A rapid transit optical routing network. In: Proceedings of the 36th annual International Symposium on Computer Architecture, ISCA '09, New York:ACM; 2009. p. 441–450
14. Beausoleil et al. R. A nanophotonic interconnect for high-performance many-core computation. Symposium on High-Performance Interconnects, 2008. p. 182–189
15. Chan J, Hendry G, Bergman K, Carloni L. Physical-layer modeling and system-level design of chip-scale photonic interconnection networks. IEEE Trans Comput-Aided Design Integr Circuits Syst. 2011;30(10):1507–1520.
16. Emelett SJ, Soref R. Design and simulation of silicon microring optical routing switches. J Lightwave Technol. 2005;23:1800
17. Pearson et al. MR. Arrayed waveguide grating demultiplexers in silicon-on-insulator. In: Proceedings of SPIE, vol. 3953, 2000. p. 11–18
18. Boyd RW. Nonlinear optics, third edition. 3rd ed. Academic Press: New York; 2008.
19. Dinu M, Quochi F, Garcia H. Third-order nonlinearities in silicon at telecom wavelengths. Appl Phys Lett. 2003;82:2954–56
20. Liao L et al. High speed silicon Mach-Zehnder modulator. Opt Express. 2005;13:3129–35
21. Green W et al. Ultra-compact, low RF power, 10 Gb/s silicon Mach-Zehnder modulator. Opt Express. 2007;15:17106–113
22. Liao L, Liu A, Basak J, Nguyen H, Paniccia M, Rubin D, Chetrit Y, Cohen R, Izhaky N. Gbit/s silicon optical modulator for highspeed applications. Electron Lett. 2007;43(22):1196–1197
23. Park H, Fang A, Kodama S, Bowers J. Hybrid silicon evanescent laser fabricated with a silicon waveguide and III-V offset quantum wells. Opt Express. 2005;13:9460–9464
24. Lipson M. Compact electro-optic modulators on a silicon chip. IEEE J Sel Top Quantum Electron. 2006;12:1520–1526
25. Gunn C, Masini GLI. Closing in on photonics large-scale integration. Photon Spectra. 2007
26. Miller DAB. Optical interconnects to electronic chips. Appl Opt. 2010;49:F59–F70
27. Madsen C, Zhao J. Optical filter design and analysis: a signal processing approach. NewYork: Wiley, 1999.
28. Condrat C, Kalla P, Blair S. A methodology for physical design automation for integrated optics. In: Proceedings of IEEE International Midwest Symposium on Circuits and Systems, 2012.
29. Condrat C, Kalla P, Blair S. Channel routing for integrated optics. In: Proceedings of ACM/IEEE System-Level Interconnect Prediction Workshop, 2013.
30. Condrat C, Kalla P, Blair S, Crossing-Aware Channel Routing for Photonic Waveguides. In: Proceedings of IEEE International Midwest Symposium on Circuits and Systems, 2013.
31. Condrat C, Kalla P, Blair S. Thermal-aware Synthesis of Integrated Photonic Ring Resonators. In: To appear in Proceeding of the International Conference on CAD (ICCAD), Nov. 2014.
32. Condrat C. Design Automation for Integrated Optics, PhD thesis, University of Utah, 2014.
33. Pollock C, Lipson M, Integrated photonics. Dordrecht:Kluwer Academic Publishers; 2003.

34. Koester SJ et al. Ge-on-SOI-detector/si-cmos-amplifier receivers for high-performance optical-communication applications. J Lightwave Technol. 2007;25:46–57
35. OpSIS: Optoelectronic System Integration in Silicon. http://www.opsisfoundry.org.
36. Okamoto K. Fundamentals of optical waveguides. London: Academic Press; 2000.
37. Emelett S, Soref R. Analysis of dual-microring-resonator cross-connect switches and modulators. Opt Express. 2005;13:7840–53
38. Shlager KL, Schneider JB. A Selective survey of the finite-difference time-domain literature. Advances in Computational electrodynamics: the finite-difference time-domain method. Boston:Artech House Inc; vol. 37, 1995. p. 39–56.
39. Bryant RE. Graph based algorithms for boolean function manipulation. IEEE Trans Comput. 1986;C-35:677–91
40. Ding D, Pan D. Oil: a nano-photonics optical interconnect library for a new photonic network architecture. In: System-level interconnect prediction workshop (SLIP), 2009.
41. Ding D, Zhang Y, Huang H, Chen RT, Pan DZ. O-Router: an optical routing framework for low power on-chip silicon nano-photonic integration. In: Design Automation Conference 2009. p. 264–69.
42. Orcutt J, Ram R. Photonic device layout within the foundry cmos design environment. IEEE Photonics Technol Lett. 2010.
43. Ding D, Yu B, Pan D. "GLOW: a global router for low-power thermal-reliable interconnect synthesis using photonic wavelength multiplexing. In: 2012 17th Asia and South Pacific Design Automation Conference (ASP-DAC), 30 Jan–02 Feb 2012, p. 621–26.
44. Zheng Y, Lisherness P, Gao M, Bovington J, Cheng K, Wang H, Yang S. Power-Efficient Calibration and Reconfiguration for Optical Network-on-Chip. J Opt Commun Networking. 2012;4:955–66
45. Bogaerts W, Dumon P, Thourhout DV, Baets R. Low-loss, low-cross-talk crossings for silicon-on-insulator nanophotonic waveguides. Opt Lett. 2007;32:2801–03
46. Sanchis P, Villalba P, Cuesta F, Håkansson A, Griol A, Galán JV, Brimont A, J. Martí J. Highly efficient crossing structure for silicon-on-insulator waveguides. Opt. Lett. 2009;34:2760–62.
47. Xu F, Poon AW, Silicon cross-connect filters using microring resonator coupled multimode-interference-based waveguide crossings. Opt. Express. 2008;16:8649–57.
48. Cardenas J, Poitras CB, Robinson JT, Preston K, Chen L, Lipson M. Low loss etchless silicon photonic waveguides. Opt. Express. 2009;17:4752–57.
49. Vlasov Y, McNab S. Losses in single-mode silicon-on-insulator strip waveguides and bends. Opt. Express. 2004;12:1622–31.
50. Qian Y, Kim S, Song J, Nordin GP, Jiang J. Compact and low loss silicon-on-insulator rib waveguide 90° bend. Opt. Express. 2006;14:6020–28.
51. Li G, Yao J, Thacker H, Mekis A, Zheng X, Shubin I, Luo Y, Lee J, Raj K, Cunningham JE, Krishnamoorthy AV. Ultralow-loss, high-density SOI optical waveguide routing for macrochip interconnects. Opt. Express. 2012;20:12035–39.
52. Roy J, Papa D, Adya S, Chan H, Ng A, Lu J, Markov I. Capo: robust and scalable open-source min-cut floorplacer. In: Proceedings of the 2005 international symposium on Physical design, ISPD '05. New York: ACM; 2005. p. 224–6.
53. Larry M, Carl E. PathFinder: A Negotiation-based Performance-driven Router for FPGAs. In: Proceedings of the 1995 ACM Third International Symposium on Field-programmable Gate Arrays, FPGA '95. New York: ACM; 1995. p. 111–7.

54. Pan M, Chu C. FastRoute 2.0: A High-quality and Efficient Global Router. In: Design Automation Conference, 2007. ASP-DAC '07. Asia and South Pacific, 2007. p. 250–5.
55. Cho M, Lu K, Yuan K, Pan DZ. BoxRouter 2.0: Architecture and Implementation of a Hybrid and Robust Global Router, Ť ICCAD. In: In Proceeding of ICCAD 2007, 2007. pp. 503–8.
56. Chang Y, Lee Y, Wang T. NTHU-Route 2.0: A fast and stable global router. In: IEEE/ACM International Conference on Computer-Aided Design, 2008. ICCAD 2008. 2008. p. 338–43.
57. Hashimoto A, Stevens J. Wire routing by optimizing channel assignment within large apertures. In: Proceedings of the 8th Design Automation Workshop, DAC '71. New York: ACM; 1971. p. 155–69.
58. Deutsch D. A dogleg channel router. In: Proceedings of the 13th Design Automation Conference, DAC '76. New York:ACM; 1976. p. 425–33.
59. Yoshimura T, Kuh ES. Efficient algorithms for channel routing. IEEE Trans Comput Aided Des Integr Circuits Syst 1982;1:25–35.
60. Condrat C, Kalla P, Blair S. Crossing-aware channel routing for integrated optics. IEEE Trans CAD, special section on optical interconnects 2014;33,6:814–25.