Dedicated Synthesis for MZI-based Optical Circuits based on AND-Inverter Graphs

Arighna Deb[1] Robert Wille[2,3] Rolf Drechsler[3,4]
[1]School of Electronics Engineering, KIIT University, Bhubaneswar, India
[2]Institute for Integrated Circuits, Johannes Kepler University Linz, Austria
[3]Cyber Physical Systems, DFKI GmbH, Bremen, Germany
[4]Institute of Computer Science, University of Bremen, Germany
arighna.deb@gmail.com robert.wille@jku.at drechsler@uni-bremen.de

***Abstract*—Optical circuits received significant interest as a promising alternative to existing electronic systems. Because of this, also the synthesis of optical circuits receives increasing attention. However, initial solutions for the synthesis of optical circuits either rely on manual design or rather straight-forward mappings from established data-structures such as BDDs, SoPs/ESoPs, etc. to the corresponding optical netlist. These approaches hardly utilize the full potential of the gate libraries available in this domain. In this paper, we propose an alternative synthesis solution based on *AND-Inverter Graphs* (AIGs) which is capable of utilizing this potential. That is, a scheme is presented which dedicatedly maps the given function representation to the desired circuit in a one-to-one fashion – yielding significantly smaller circuit sizes. Experimental evaluations confirm that the proposed solution generates optical circuits with up to 97% less number of gates as compared to existing synthesis approaches.**

I. Introduction

Advances in *silicon photonics* made optical circuits a promising alternative to conventional electronic circuits and unlocked several promising applications for the future. While e.g. optical technology has been considered as a potential candidate for developing ultra-high speed and low power interconnects [1], [2], corresponding solutions still suffered from the fact that those optical interconnects require frequent transformations of signals from the electronic domain to signals from the optical domain and vice-versa. This is a significant drawback, which can be avoided if the system becomes a full-fledged optical system. Motivated by these prospects, also commercial vendors started to get interested in optical circuits [3], [4].

These developments also raised the question how to efficiently design the corresponding optical circuits. This led to the emergence of several optical circuits realizing important arithmetic components such as adders, multiplexers, etc. (e.g. see [5]–[9]). Besides that, also optical circuits realizing functions such as *Chirp-Z Transform*, *Travelling Salesman Problem*, etc. have recently been developed and demonstrated [4], [10].

However, in order to efficiently realize more complex functionality, these efforts need to be supported by automatic methods for design automation. Design automation usually starts with a logic level abstraction and, afterwards, moves down to the physical layout, where the desired circuit is refined with respect to the corresponding technological constraints. As logic synthesis and optimization is the first step in such a design flow, the automatic synthesis of optical circuits has received significant attention recently.

In fact, automatic synthesis of optical logic circuits for arbitrary Boolean functions was initially developed in works such as [11]–[14] which rely on *Binary Decision Diagrams* (BDDs, [15]) – an efficient data structure for Boolean function representation. Later, synthesis approaches relying on other function representations such as *Sum-of-Products* (SoPs), *Exclusive-Sum-of-Products* (ESoPs) [16], [17], *AND-Inverter graphs* (AIGs) [18], or corresponding derivatives have been proposed [19], [20]. However, they all followed a rather naive mapping scheme from the respectively given function description to the resulting optical circuit and, consequently, lead to rather large circuits (Section III-A discusses these related work in more detail later).

In this work, we propose an alternative synthesis approach which overcomes these drawbacks. In fact, relying on AND-Inverter graphs, a dedicated scheme for mapping AIGs to optical circuits is proposed which exploits the potential of the available (optical) gate library and realizes corresponding AIG nodes by a single gate (where previouisly proposed solutions usually required more than one gate). Overall, this yields substantial reductions in the number of gates the resulting circuits are composed of.

Experimental evaluations confirm these benefits: Compared to a broad variety of previously proposed solutions which constitute the state-of-the-art in the synthesis of optical circuits (namely BDD-, SoP-, ESoP-, OIG-, as well as the naive AIG-based synthesis), we can observe reductions of up to 97% (more precisely, reductions of 51%, 97%, 88%, 38% and 38% are obtained, respectively).

The remainder of this work is structured as follows: Section II reviews the basics on optical circuits, while Section III discusses related work and introduces the general ideas of the dedicated synthesis scheme proposed in this work. Afterwards, details of the resulting synthesis approach are provided in Section IV. Section V summarizes the obtained experimental results before Section VI concludes the paper.

II. Optical Circuits

Optical circuits are usually realized by means of *Mach-Zehnder Interferometer* (MZI) switches which are based on *Semiconductor Optical Amplifiers* (SOAs). In the logic domain, the resulting structure is abstracted to a so-called *MZI gate*. Each MZI gate has two input ports and two output ports. The inputs can either be sourced by light (representing the binary "1") or darkness (representing the binary "0"). Logically, an MZI gate is defined as follows [21], [22].

Definition 1: An MZI gate realizes a Boolean function $\mathbb{B}^2 \rightarrow \mathbb{B}^2$ composed of two optical inputs p and q as well as two optical outputs f and g. The absence of any input signal leads to the logic value 0 at the output f. The presence of input signal p and the absence of input signal q leads to the logic value 1 at the output g. In the presence of both input signals, the outputs f and g produce 1 and 0, respectively. Therefore, the functions

$$f = p \wedge q \qquad \text{and} \qquad g = p \wedge q'$$

are realized. Fig. 1(a) provides the graphical representation of an MZI gate.

In addition, splitters and combiners are used as optical logic elements in order to realize logic functions.

Definition 2: A *splitter* divides an optical signal into two signals – each with only half of the incoming signal power. In contrast, a *combiner* merges two optical signals into a single

(a) MZI gate (b) Splitter (c) Combiner

Fig. 1: Optical gates

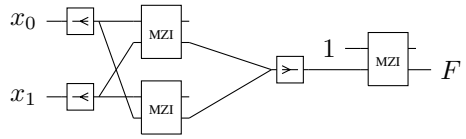


Fig. 2: Optical circuit

one and, by this, inherently realizes the OR-function. A splitter (combiner) may have more than two outputs (inputs). Then, in case of a splitter, the strength of the signal is divided by the number of outputs. Fig. 1(b) and Fig. 1(c) provide the graphical representation of both elements.

Together these logic elements form a *gate library* that allows to realize any Boolean function.

Example 1: Fig. 2 shows an optical circuit realizing a Boolean function $F = (x_0 \wedge x_1) \vee (x_0' \wedge x_1')$. The circuit is composed of three MZI gates, two splitters, and one combiner i.e. a total of six optical gates.

III. Related Work and Motivation

In this section, we review and discuss related work on the synthesis of MZI-based optical circuits. Afterwards, we review AND-Inverter Graphs, which are employed in this work to overcome the drawbacks of the current state-of-the-art. Based on both, we illustrate the main ideas and prospects of the proposed synthesis scheme. By this, this section serves as motivation of the synthesis solution which, afterwards, is described in detail in the next section.

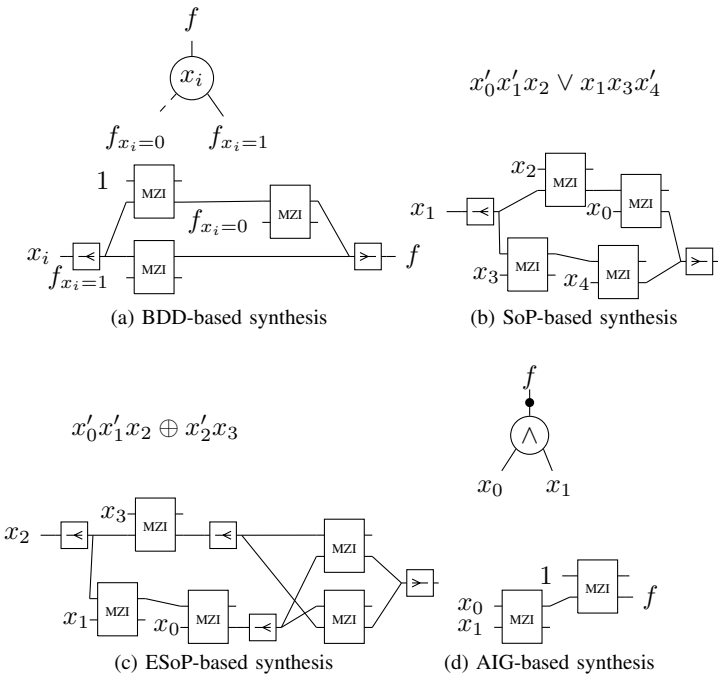
A. Synthesis of MZI-based Optical Circuits

Synthesis is the task of generating a logic circuit netlist which realizes a given (Boolean) function to be synthesized based on the respectively considered gate library. The typical input for synthesis approaches is a Boolean function representation such as two-level representations (i.e. *Sum of Products* (SoPs) and *Exclusive Sum of Products* (ESoPs)), *Binary Decision Diagrams* (BDDs) [15], *AND-Inverter Graphs* (AIGs) [18], etc. The corresponding function representation is then mapped to a netlist of gates using the gate library available in the considered technology.

To this end, a one-to-one relation between the function representation and the considered gate library is desired and may allow for a cheaper circuit realization. For example, in conventional technologies, a node of a BDD corresponds to a MUX gate, while a node of an AIG directly corresponds to a NAND gate. Since a NAND gate is significantly cheaper than a MUX gate (which usually is realized by several elementary gates), NAND-based circuits are usually preferred over MUX-based circuits. This makes AIGs, a preferred function representation for conventional technologies.

Similar observations can be made for synthesis of optical circuits. For example:

- BDD-based Synthesis (which has been considered in [11]–[14]) utilizes the fact that each node of a BDD is mapped to an MZI sub-circuit realizing the respective MUX-operation (as illustrated in Fig. 3(a)). But since no direct mapping from a BDD node to an elementary optical gate exists, several gates are required to realize just a single node. This obviously leads to optical circuits of high gate costs.
- Synthesis approaches based on SoPs or ESoPs (e.g. similar to the ones proposed in [19]) are also quite straightforward since each product term can be realized by a cascade of MZI gates. In case of SoP expressions, all such cascades of MZI gates realizing product terms are combined using a combiner. Fig. 3(b) illustrates an example of an SoP expression and the corresponding realization in terms of an optical circuit. For realizing ESoP expressions, the exclusive disjunction (XOR) of products is realized as $(C_i C_j') \vee (C_i' C_j)$, where, C_i and C_j represent product terms of an ESoP. An ESoP expression and its realization in terms of an optical circuit are illustrated in Fig. 3(c). However, also these approaches generate optical circuits with a very large number of gates.
- Finally, approaches proposed e.g. in [20] rely on AIGs as well as a corresponding derivative called *OR-Inverter Graphs* (OIGs). AIGs (OIGs) are function representations which employ two-input AND (OR) nodes along with regular and complemented edges. This allows for a one-to-one mapping from AIGs (OIGs) to optical circuits as each AND (OR) node is realized by an MZI gate (a combiner) and each complement edge (i.e. inversion) is realized by an MZI gate. Fig. 3(d) shows an AIG node and its corresponding optical circuit realization. However, although AIG/OIG-based synthesis is indeed promising since many nodes can directly be realized by a single MZI gate/combiner, each negation (usually a simple operation) requires a full MZI gate.



(a) BDD-based synthesis (b) SoP-based synthesis

(c) ESoP-based synthesis (d) AIG-based synthesis

Fig. 3: Illustration of related work

Overall, the existing state-of-the-art tries to follow the established synthesis schemes from conventional synthesis and maps the given function representation to corresponding circuit structures. But all function representations considered thus far (namely, BDDs, SoPs, ESoPs, AIGs, or OIGs) lead

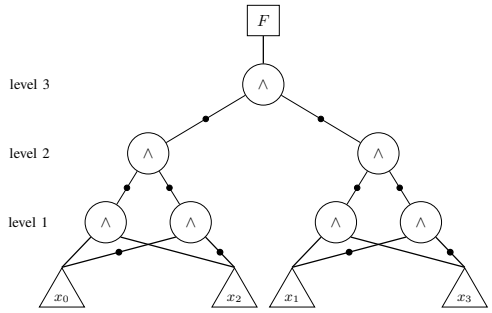


Fig. 4: AND-Inverter graph for function F

to rather large circuit costs since they either do not allow for a one-to-one mapping of nodes/products to elementary gates (in case of BDDs, SoPs, and ESoPs) or realize trivial sub-functionality such as negations/inverters by full-fledged gates (in case of AIGs and OIGs). Motivated by this, we are considering a dedicated synthesis of MZI-based optical circuits whose general idea is introduced in the following. As this idea relies on *AND-Inverter Graphs* (AIGs), we are reviewing the basic concepts of AIGs before.

B. AND-Inverter Graphs

An *AND-Inverter Graph* (AIG) is a directed acyclic graph denoted as $G = (V, E)$ where V represents a set of vertices or nodes and E indicates a set of directed edges interconnecting the nodes. A node of any AIG either corresponds to a primary input, a primary output (terminal), or a Boolean AND operation. Edges in AIGs can either be regular or complemented leading to the actual functionality or the negated functionality, respectively. More formally, an AIG is defined as follows:

Definition 3: An *AND-Inverter graph* (AIG) over the primary input variables $X = \{x_1, x_2, \cdots, x_n\}$ and with the primary output variables $Y = \{y_1, y_2, \cdots, y_m\}$ is a directed acyclic graph $G = (V(= \{V_X \cup V_g \cup V_Y\}), E)$ with the following properties:

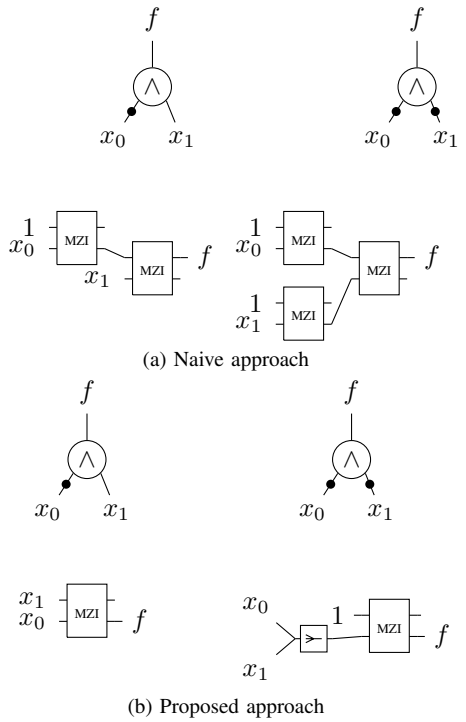
- Each primary input (PI) node $v \in V_X$ is labeled by $x_i \in X$ and has no incoming edges.
- Each primary output (PO) node $v \in V_Y$ is a terminal labeled by $y_j \in Y$ and has no outgoing edges.
- Each non-terminal node $v \in V_g$ represents a Boolean conjunction (AND) of the functions represented by the two incoming edges.
- An edge $e \in E$ connecting a source node $u \in V$ to a target node $v \in V$ is either a regular or a complement edge i.e. $e = \{(u, (v \times p)) | u, v \in V, u \notin V_Y, v \notin V_X\}$ with p denoting whether the edge is a regular edge ($p = 1$) or a complement edge ($p = 0$).

The size of an AIG is measured in terms of the total number of AND nodes.

Example 2: Consider the function $F = (x_0 \wedge x_1 \wedge x_2 \wedge x_3) \vee (x_0 \wedge x_1' \wedge x_2 \wedge x_3') \vee (x_0' \wedge x_1 \wedge x_2' \wedge x_3) \vee (x_0' \wedge x_1' \wedge x_2' \wedge x_3')$. Using DeMorgan's theorem, the function can be written as $F = ((x_0 \wedge x_2)' \wedge (x_0' \wedge x_2')')' \vee ((x_1 \wedge x_3)' \wedge (x_1' \wedge x_3')')'$. The corresponding AIG is shown in Fig. 4, in which an edge with a solid dot denotes a complement edge.

C. Dedicated AIG-based Synthesis

As discussed above, AIGs have been utilized before in order to realize an optical circuit for a given function to be synthesized. However, here a rather simple scheme has been considered in which AND nodes and complement edges are naively mapped to explicit MZI gates. In fact, the MZI gate as reviewed in Section II realizes two functions at once, namely



(a) Naive approach

(b) Proposed approach

Fig. 5: Naive vs. proposed approach

$f = p \wedge q$ and $g = p \wedge q'$. But the naive AIG-based method utilizes the first output only to realize the main building block of the AIG (the AND node) and the second output only to realize the negation caused by complement edges (additionally setting $p = 1$ and, hence, yielding $g = 1 \wedge q' = q'$, i.e. the negation of an input). As complement edges frequently occur in AIGs, this frequently causes several MZI gates to realize a single AIG node (one for the AND and others for the negation) as illustrated in Fig. 5(a).

In this work, we try to avoid this overhead by exploiting the potential of a single MZI gate. We propose a dedicated AIG-based synthesis method which realizes both, the AND node and possible complement edges at once. This is indeed possible, because

- cases in which no complemented input occurs can be realized using the first output f of an MZI gate,
- cases in which a single complemented input occurs can be realized using the second output g of an MZI gate as illustrated in Fig. 5(b) (here, we only have to make sure that the complement input is connected to the MZI-input q; eventually yielding the conjunction of one input with the inverse of the other input), and
- cases in which both inputs are complemented can be transformed using DeMorgan rules (here, the AND with negative inputs can be transformed to an OR without negative inputs; this can eventually be realized by a combiner and a single MZI gate as illustrated in Fig. 5(b)).

That is, following these ideas almost all AIG nodes (no matter whether they depend on regular or complemented inputs) can be realized by a single MZI gate only. Overall, this leads to a more dedicated AIG-based synthesis scheme which allows for a significant reduction in the number of gates.

IV. Resulting Synthesis Scheme

Based on the discussions in Section III, we propose a synthesis flow for the efficient realization of optical circuits, which involves two major steps: (1) the generation of an AIG and (2) the mapping of an AIG to an optical circuit. Since an AIG can be generated using existing methods employed in tools such as ABC [23], we primarily focus on how to efficiently map AIGs to optical circuits composed of MZI gates, combiners, and splitters.

Given an AIG, $G = (V(= V_X \cup V_g \cup V_Y), E)$ representing the function f to be synthesized, an optical circuit can be derived by traversing the graph and substituting each AND node $v \in V_g$ with a corresponding sub-circuit. To this end, we consider all the possible functional behaviours of AND nodes which may occur in an AIG and for which a corresponding sub-circuit is required. More precisely, the following node configurations are considered:

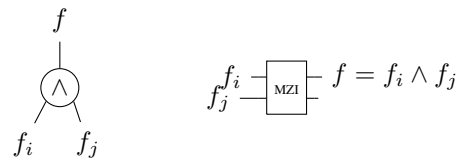
1) *An AND node $v \in V_g$ with two regular incoming edges f_i and f_j representing the function $f = f_i \wedge f_j$*: In this case, an MZI gate is added to the circuit in which f_i is applied to input port p and f_j is applied to input port q. The output is obtained from output port f of the corresponding MZI. This is shown in Fig. 6(a).
2) *An AND node $v \in V_g$ with either a regular incoming edge f_i and a complement incoming edge f'_j or vice-versa representing either function $f = f_i \wedge f'_j$ or $f = f'_i \wedge f_j$*: In this case, an MZI gate is added to the circuit where f_i (f_j) is applied to input port p and f_j (f_i) is applied to input port q. The output is obtained from output port g which realizes the function $f_i \wedge f'_j$ ($f'_i \wedge f_j$). This is shown in Fig. 6(b).
3) *An AND node $v \in V_g$ with a complement outgoing edge realizing primary output function f'* (terminal case): This case is realized by an MZI-NOT gate, i.e. an MZI gate with a fix input and using the output port which negates the initial value of the other input as shown in Fig. 6(c).
4) *An AND node $v \in V_g$ realizing a primary output f with both incoming edges f_i and f_j are complemented i.e. $f = f'_i \wedge f'_j$*: This is realized by a sub-circuit composed of a combiner and an MZI-NOT gate as shown in Fig. 6(d).

Note that the AND node with two complement incoming edges shown in Fig. 6(d) is realized as a functionally equivalent OR node based on DeMorgan's theorem. That is, the function $f = f'_i \wedge f'_j$ is realized as $f = (f_i \vee f_j)'$ using a combiner and an MZI-NOT gate. Treating such AND nodes as functionally equivalent OR nodes inverts the polarities of the outgoing edges of the corresponding nodes as depicted in Fig. 7. The MZI-NOT gate is appended after the combiner only in case, a primary output node is connected through a complement edge to the AND node having two complement incoming edges. Otherwise, only a combiner is used to realize the AND node with two complement incoming edges as its successor nodes can be treated as normal AND node configurations as mentioned above.

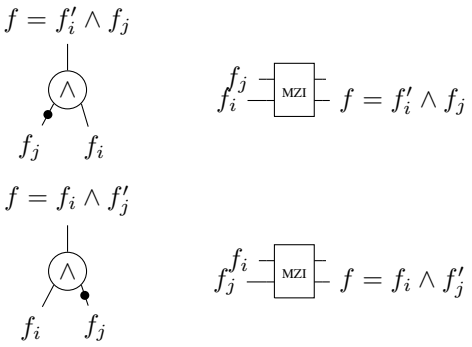
Moreover, the node configurations shown in Figs. 6(a) and 6(b) can be realized using a single MZI gate if two nodes form a pair. Two AND nodes can be considered as pair if they satisfy the following definition.

Definition 4: Two AND nodes realizing either functions $(f_i \wedge f_j)$ and $(f'_i \wedge f_j)$ or $(f_i \wedge f_j)$ and $(f_i \wedge f'_j)$ can be treated as a pair of nodes if both nodes are connected to the same predecessor nodes.

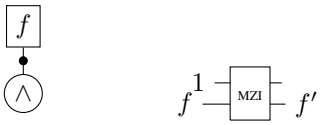
Fig. 8 depicts the case described in Definition 4 and the corresponding mapping to an MZI gate. The upper part of



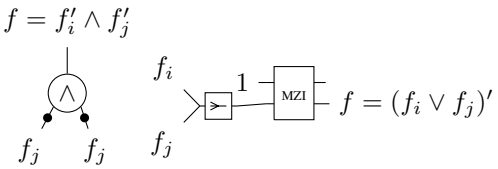
(a) W/ two regular incoming edges



(b) W/ regular and complemented incoming edge



(c) Terminal case



(d) W/ both complement edges

Fig. 6: AIG nodes and corresponding optical circuit

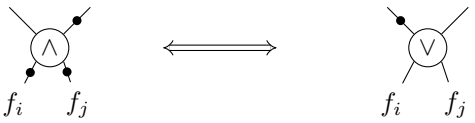


Fig. 7: Nodes with functional equivalence

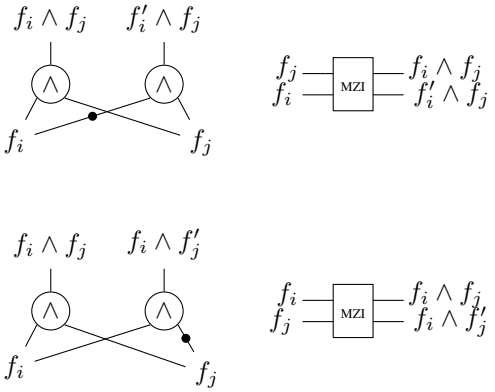


Fig. 8: Mapping pair of nodes to an MZI gate

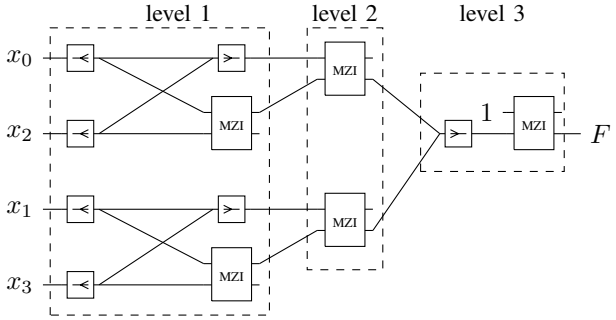


Fig. 9: Mapping the AIG from Fig. 4 to an optical circuit

Fig. 8 shows that by applying f_j to input port p and f_i to input port q of an MZI gate. The two functions $(f_i \wedge f_j), (f_i' \wedge f_j)$ are respectively obtained from the output ports f and g of the MZI gate. This indicates, a single MZI gate is sufficient to realize a pair of AND nodes of the form $(f_i \wedge f_j), (f_i' \wedge f_j)$. Similarly, to realize a pair of AND nodes of the form $(f_i \wedge f_j), (f_i \wedge f_j')$, the respective inputs f_i and f_j are applied to input ports p and q of the MZI gate as shown at the lower part of Fig. 8.

Overall, this yields a synthesis flow which works as follows:

1) Generate an AIG $G = (V, E)$ representing the function F to be synthesized.
2) Traverse G in a breadth-first manner.
3) For each node, apply the corresponding sub-circuit as shown in Fig. 6.
4) For nodes having multiple outgoing edges, add splitters at the output of the corresponding gate realizing the node.
5) Connect the inputs of the sub-circuits appropriately to the outputs of the sub-circuits depending on the successor-predecessor relationship among the nodes of G.

Example 3: Consider the AIG representing the function F as depicted in Fig. 4. The mapping begins with traversing the AIG in a breadth-first manner and applying the substitutions shown in Figs. 6 and 8 to each node of the AIG. For level 1, the resulting optical gates are shown on the left-hand side of Fig. 9. In the next step, the AND nodes at level 2 are mapped to corresponding gates shown in the middle of Fig. 9. Each AND node at level 2 is connected to an AND node of level 1 which is realized using a combiner. This realizes the AND nodes at level 2 as discussed before by means of Fig. 6(b). In a similar fashion, the AND node at level 3 is handled – leading to the gates shown on the right-hand side of Fig. 9 and, hence, the overall circuit realizing F.

Overall, the proposed scheme yields circuits with a significantly less number of gates than the previously proposed AIG-based method in which AND nodes and complement edges are naively mapped to explicit MZI gates. In fact, the proposed dedicated AIG-based scheme eventually realizes a direct mapping of an AIG node to a single gate in almost all cases and, by this, also outperforms the other solutions proposed in the past (and reviewed in Section III-A). Experimental results which are summarized in the next section confirm these improvements.

V. Experimental Results

In this section, we present the experimental results obtained by the proposed AIG-based synthesis for optical circuits and compared them to the state-of-the art solutions reviewed in Section III-A. To this end, the synthesis flow described in Section IV has been implemented in C++. First, an AIG of the function to be synthesized is created using the tool *ABC* [23]. Then, the mapping method as described above is applied which reads the generated AIG description and generates a netlist of optical gates. The previously proposed methods based on BDDs, SoPs, ESoPs, OIG, as well as the naive AIG-based method have been re-implemented as described in the corresponding related work [12], [19], [20]. As test cases, several benchmarks from the MCNC suite have been considered which have also been used in these related works. All experiments have been carried out on a Linux machine with a 2.8 GHz Intel Core i7 processor and 8 GB memory. All circuits have been obtained in negligible run-time (i.e. not more than one CPU minute) which is why a detailed run-time discussion has been omitted in the following.

Table I summarizes the obtained results. The first column provides the details of the considered benchmarks i.e. their names as well as the number of primary inputs (*PI*) and primary outputs (*PO*). The second column lists the total number of gates (*Gate count*) for the resulting circuits obtained by the SoP-, ESoP-, BDD-, OIG-, (existing) naive AIG-, and (newly proposed) dedicated AIG-based synthesis approaches. The final columns report the percentage reduction in number of optical gates compared to the existing synthesis approaches.

Recall that the previously proposed solutions based on BDDs, SoPs, ESoPs, as well as the naive AIG/OIG mapping scheme lead to rather large circuits, since they either do not allow for a one-to-one mapping of nodes/products to gates (in case of BDDs, SoPs, and ESoPs) or realize rather trivial sub-functionality such as inversions/inverters by fully-fledged gates (in case of AIGs and OIGs). The dedicated AIG-based method proposed in this work overcomes this drawback. This is also confirmed by the obtained results: On average, the proposed dedicated AIG-based methods generates optical circuits with $97\%, 88\%, 51\%, 38\%$ and 38% less number of MZI gates than the previous SoP-based, ESoP-based, BDD-based, OIG-based and AIG-based approaches, respectively.

VI. Conclusion

In this work, we presented an alternative synthesis solution which overcomes the drawbacks of the state-of-the-art in the synthesis of MZI-based optical circuits. Relying on AND-Inverter graphs (AIGs), the proposed approach introduces a dedicated scheme for mapping AIGs to optical circuits which utilizes the expressive power of the available (optical) gate library. In contrast to previously proposed solutions which required more than one gate to realize a single node with complement edges, the proposed scheme realizes an AIG node with complement edges by a single gate only. As a result, substantial reductions in the number of gates is achieved. Experimental results confirmed the benefits of the proposed approach compared to the state-of-the-art.

Acknowledgments

This work has been supported by the Austrian Agency for International Cooperation in Education and Research (OeAD) within a project under grant no. IN 08/2017.

References

[1] M. Mohamed, Z. Li, X. Chen, L. Shang, and A. R. Mickelson, "Reliability-Aware Design Flow for Silicon Photonics On-Chip Interconnect," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 8, pp. 1763–1776, 2014.

[2] T. Sato, K. Takeda, A. Shinya, M. Notomi, K. Hasebe, T. Kakitsuka, and S. Matsuo, "Photonic Crystal Lasers for Chip-to-Chip and On-Chip Optical Interconnects," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 21, no. 6, pp. 728–737, 2015.

[3] Optalysys– revolutionary optical processing technology, www.optalysys.com.

[4] R. Courtland, "The Ising on the Computer Chip," *IEEE Spectrum*, vol. 54, no. 1, 2017.

TABLE I: Optical circuit synthesis results

Benchmark		Gate count						% Reduction in gate count				
Name	PI/PO	SoP	ESoP	BDD	OIG	naive AIG	dedicated AIG	SoP	ESoP	BDD	OIG	naive AIG
apex5	117/88	6071	11709	5393	1769	1663	1066	82.4%	90.9%	80.2%	39.7%	35.9%
mish	94/43	126	446	444	218	162	141	-11.9%	68.4%	68.2%	35.3%	13.0%
soar	83/94	3025	5175	3178	1609	1556	724	76.1%	86.0%	77.2%	55.0%	53.5%
x2dn	82/56	449	792	884	385	387	264	41.2%	66.7%	70.1%	31.4%	31.8%
x7dn	66/15	4479	7499	2291	831	844	489	89.1%	93.5%	78.7%	41.2%	42.1%
ti	47/72	2248	6730	3243	1825	1775	1100	51.1%	83.7%	66.1%	39.7%	38.0%
apex1	45/45	1784	7047	4910	3957	3957	2426	-36.0%	65.6%	50.6%	38.7%	38.7%
xparc	41/73	11252	45306	7555	6087	6038	3848	65.8%	91.5%	49.1%	36.8%	36.3%
seq	41/35	16438	23526	5927	3611	3581	2137	87.0%	90.9%	63.9%	40.8%	40.3%
0410184	14/14	229376	2200	304	239	238	156	99.9%	92.9%	48.7%	34.7%	34.5%
4mod5	4/1	18	24	15	18	17	12	33.3%	50.0%	20.0%	33.3%	29.4%
4mod7	4/3	52	68	54	47	48	22	57.7%	67.6%	59.3%	53.2%	54.2%
5xp1	7/10	237	361	125	218	217	123	48.1%	65.9%	1.6%	43.6%	43.3%
add6	12/7	1860	1597	239	108	111	75	96.0%	95.3%	68.6%	30.6%	32.4%
adr4	8/5	278	333	78	68	69	47	83.1%	85.9%	39.7%	30.9%	31.9%
alu	5/1	71	27	25	22	22	13	81.7%	51.9%	48.0%	40.9%	40.9%
alu1	12/8	41	101	58	69	70	49	-19.5%	51.5%	15.5%	29.0%	30.0%
alu2	10/6	1760	752	776	844	843	508	71.1%	32.4%	34.5%	39.8%	39.7%
alu3	10/8	232	524	331	153	153	97	58.2%	81.5%	70.7%	36.6%	36.6%
alu4	14/8	6869	6086	2713	2258	2264	1337	80.5%	78.0%	50.7%	40.8%	40.9%
apex2	39/3	13499	41093	1861	619	626	369	97.3%	99.1%	80.2%	40.4%	41.1%
bw	5/28	242	1408	554	296	301	203	16.1%	85.6%	63.4%	31.4%	32.6%
cordic	23/2	17188	18364	174	127	134	91	99.5%	99.5%	47.7%	28.3%	32.1%
e64	65/65	2144	2522	442	1011	928	915	57.3%	63.7%	-107.0%	9.5%	1.4%
ex5p	8/63	2105	4339	724	987	988	709	66.3%	83.7%	2.1%	28.2%	28.2%
ham15	15/15	491520	601	304	400	415	249	99.9%	58.6%	18.1%	37.8%	40.0%
ham3	3/3	24	36	23	30	31	17	29.2%	52.8%	26.1%	43.3%	45.2%
hwb4	4/4	64	92	75	64	68	36	43.8%	60.9%	52.0%	43.8%	47.1%
hwb5	5/5	160	278	144	166	171	97	39.4%	65.1%	32.6%	41.6%	43.3%
hwb6	6/6	384	647	271	388	394	243	36.7%	62.4%	10.3%	37.4%	38.3%
hwb8	8/8	2048	4087	747	1742	1750	1032	49.6%	74.7%	-38.2%	40.8%	41.0%
mod5d2	5/5	161	70	43	51	47	28	82.6%	60.0%	34.9%	45.1%	40.4%
One-two-three	3/3	19	49	29	24	25	16	15.8%	67.3%	44.8%	33.3%	36.0%
pdc	16/40	33056	6644	1919	1693	1660	1052	96.8%	84.2%	45.2%	37.9%	36.6%
plus127mod8192	13/13	106497	220	106	257	265	167	99.8%	24.1%	-57.5%	35.0%	37.0%
plus63mod4096	12/12	49153	206	98	219	226	135	99.7%	34.5%	-37.8%	38.4%	40.3%
plus63mod8192	13/13	106497	227	107	240	250	173	99.8%	23.8%	-61.7%	27.9%	30.8%
rd53	5/3	120	118	82	102	105	50	58.3%	57.6%	39.0%	51.0%	52.4%
rd73	7/3	709	409	135	247	248	144	79.7%	64.8%	-6.7%	41.7%	41.9%
rd84	8/4	1944	577	181	358	358	218	88.8%	62.2%	-20.4%	39.1%	39.1%
spla	16/46	34984	6890	1816	1673	1631	1175	96.6%	82.9%	35.3%	29.8%	28.0%
urf1	9/9	4608	9279	3233	4293	4302	2589	43.8%	72.1%	19.9%	39.7%	39.8%
urf2	8/8	2048	3723	1802	1978	1986	1196	41.6%	67.9%	33.6%	39.5%	39.8%
urf5	9/9	4608	1851	1595	1946	1954	1175	74.5%	36.5%	26.3%	39.6%	39.9%

All results have been obtained in negligible run-time i.e. just a few CPU seconds.

[5] A. K. Cherri and A. S. Al-Zayed, "Circuit Designs of Ultra-fast All-Optical Modified Signed-Digit Adders using Semiconductor Optical Amplifier and Mach-Zehnder Interferometer," *Optik - International Journal for Light and Electron Optics*, vol. 121, no. 17, pp. 1577 – 1585, 2010.

[6] K. Datta, T. Chattopadhyay, and I. Sengupta, "All Optical Design of Binary Adders using Semiconductor Optical Amplifier Assisted Mach-Zehnder Interferometer," *Microelectronics Journal*, vol. 46, no. 9, pp. 839–847, 2015.

[7] K. Datta and I. Sengupta, "All Optical Reversible Multiplexer Design using Mach-Zehnder Interferometer," in *International Conference on VLSI Design*, 2014, pp. 539–544.

[8] Y. Aikawa, S. Shimizu, and H. Uenohara, "Demonstration of All-Optical Divider Circuit Using SOA-MZI-Type XOR Gate and Feedback Loop for Forward Error Detection," *Journal of Lightwave Technology*, vol. 29, no. 15, pp. 2259–2266, 2011.

[9] T. Chattopadhyay and D. K. Gayen, "All-optical 2's complement number conversion scheme without binary addition," *IET Optoelectronics*, vol. 11, no. 1, pp. 1–7, 2017.

[10] N. Q. Ngo, "Optical Chirp Z-Transform Processor: Design and Application," *Journal of Lightwave Technology*, vol. 33, no. 11, pp. 2213–2221, 2015.

[11] C. Condrat, P. Kalla, and S. Blair, "Logic Synthesis for Integrated Optics," in *Great lakes symposium on VLSI*. ACM, 2011, pp. 13–18.

[12] E. Schönborn, K. Datta, R. Wille, I. Sengupta, H. Rahaman, and R. Drechsler, "BDD-Based Synthesis for All-Optical Mach-Zehnder Interferometer Circuits," in *International Conference on VLSI Design*, 2015, pp. 435–440.

[13] R. Wille, O. Keszocze, C. Hopfmuller, and R. Drechsler, "Reverse BDD-based Synthesis for Splitter-free Optical Circuits," in *Asia and South Pacific Design Automation Conference*, 2015, pp. 172–177.

[14] A. Deb, R. Wille, O. Keszocze, S. Shirinzadeh, and R. Drechsler, "Synthesis of optical circuits using binary decision diagrams," *Integration, the VLSI Journal*, vol. 59, pp. 42 – 51, 2017.

[15] R. E. Bryant, "Graph-Based Algorithms for Boolean Function Manipulation," *IEEE Transactions on Computers*, vol. 100, no. 8, pp. 677–691, 1986.

[16] G. D. Micheli, *Synthesis and Optimization of Digital Circuits*. McGraw-Hill Higher Education, 1994.

[17] T. Sasao, *Switching Theory for Logic Synthesis*. Kluwer, Dordrecht, 1999.

[18] A. Kuehlmann, V. Paruthi, F. Krohm, and M. K. Ganai, "Robust Boolean Reasoning for Equivalence Checking and Functional Property Verification," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, no. 12, pp. 1377–1394, 2002.

[19] A. Deb, R. Wille, O. Keszöcze, S. Hillmich, and R. Drechsler, "Gates vs. Splitters: Contradictory Optimization Objectives in the Synthesis of Optical Circuits," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 13, no. 1, pp. 11:1–11:13, 2016.

[20] A. Deb, R. Wille, and R. Drechsler, "OR-Inverter Graphs for the Synthesis of Optical Circuits," in *International Symposium on Multiple-Valued Logic (ISMVL)*, 2017, pp. 278–283.

[21] M. P. Scaffardi, P. Ghelfi, E. Lazzeri, L. Poti, and A. Bogoni, "Photonic Processing for Digital Comparison and Full Addition based on Semiconductor Optical Amplifiers," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 14, no. 3, pp. 826–833, 2008.

[22] Q. Wang, G. Zhu, H. Chen, J. Jaques, J. Leuthold, A. B. Piccirilli, and N. K. Dutta, "Study of All-Optical XOR using Mach-Zehnder Interferometer and Differential Scheme," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 40, no. 6, pp. 703–710, 2004.

[23] R. Brayton and A. Mishchenko, "ABC: An Academic Industrial-strength Verification Tool," in *Proc. of International Conference on Computer Aided Verification*, 2010, pp. 24–40.