Segmented Channel Routability Via Satisfiability

WILLIAM N. N. HUNG
Intel Corporation
XIAOYU SONG
Portland State University
EL MOSTAPHA ABOULHAMID
University of Montréal
ANDREW KENNINGS
University of Waterloo
and
ALAN COPPOLA
Cypress Semiconductor

Segmented channel routing is fundamental to the routing of row-based FPGAs. In this paper, we study segmented channel routability via satisfiability. Our method encodes the horizontal and vertical constraints of the routing problem as Boolean conditions. The routability constraint is satisfiable if and only if the net connections in the segmented channel are routable. Empirical results show that the method is time-efficient and applicable to large problem instances.

Categories and Subject Descriptors: B.6.3 [**Logic Design**]: Design Aides—*automatic synthesis; optimization; verification*; B.7.1 [**Integrated Circuits**]: Types and Design Styles—*gate arrays*

General Terms: Design, Verification

Additional Key Words and Phrases: Satisfiability

1. INTRODUCTION

Segmented channel routing arises in field-programmable gate array (FPGA) design [Roychowdhury et al. 1993]. In this article, we propose a satisfiability-based (SAT) approach for solving the segmented channel routing problem. The method

Authors' present addresses: W. N. N. Hung, Synplicity, Inc., 600 W. California Avenue, Sunnyvale, CA 94086; X. Song, Department of ECE, Portland State University, Portland, OR 97207; E. M. Aboulhamid, DIRO, University of Montréal, Montréal, Que., Canada; A. Kennings, Department of ECE, University of Waterloo, Waterloo, Ont. Canada; A. Coppola, Cypress Semiconductor, Beaverton, OR 97124; Corresponding e-mail address: william_hung@alumni.utexas.net.

encodes the horizontal and vertical constraints as Boolean conditions. The resulting routability constraint is satisfiable if the conjunction of all the conditions are satisfiable. Devadas [1989] showed a simple formulation of classical two-layer channel routing as Boolean satisfiability. We extend his method to segmented channel routing.

According to Trimberger [1995], FPGA architectures can be classified into two types: island-style (symmetric) FPGAs and row-based (segmented channel) FPGAs. For island-style (symmetric) FPGAs, some routing approaches via satisfiability has been done [Wood and Rutenbar 1998; Nam et al. 1999, 2001]. They assumed that each wire segment spans only one block, used a heuristic global router to find a routing path (cells) for each net, and then used satisfiability to perform detailed routing in the cells along the globally routed paths. Xu et al. [2003] presented a slightly different approach for symmetric FPGAs with a partial (incomplete) satisfiability result at faster runtime.

Row-based FPGAs are architecturally different from island-style FPGAs. For example, there are more than one wire segments per track in row-based FPGAs. None of the above satisfiability approaches considered the segmented channel routability of row-based FPGAs. To study this problem, Roychowdhury et al. [1993] and Roy [1993] described search-based strategies for segmented channel routing and proved that it is NP-complete. Yang et al. [2000] proposed heuristic algorithms based on graph clique for some restricted segmented channel routability check.

Other related problems have also been studied. Wu and Marek-Sadowska [1993] analyzed graph-based approaches to FPGA routing. Thakur et al. [1997] investigated FPGA switch modules. Heuristic detailed routing for FPGAs were investigated [Brown et al. 1992; Limieux and Brown 1993]. Symmetric approaches to solving the satisfiability problem (for island-style FPGA routing and other decision problems) were explored [Aloul et al. 2002]. In the board level, Song et al. [2003] used satisfiability to solve the interchip FPGA and FPIC (field-programmable interconnect chip) routing problem.

The organization of this article is as follows. Section 2 presents the basic definitions of our routing model. Sections 3 and 4 develop the satisfiability formulation for dogleg-free and doglegged routing respectively. Section 5 describes implementation issues and shows the promising experimental results for some benchmarks. Finally, Section 6 offers some concluding remarks.

2. PRELIMINARIES AND DEFINITIONS

As defined by Roychowdhury et al. [1993], let Γ be a set of T tracks and C be a set of M nets. A channel of width T and length N is a rectangular grid. Track 0 and Track $\mathrm{T}+1$ are the lower and upper boundaries of the channel. Tracks are horizontal segments of length N. There are switches on each track to divide the track into a set G_t of g_t adjacent segments. A column is a vertical segment connecting terminals on the boundary. A net is a collection of terminals to be connected. Each terminal occupies a column. The span of net c is defined by its leftmost and rightmost columns, *left*(c) and *right*(c). Let $L(c)$ be the set of terminals belonging to net c and *left*$(c) \leq i \leq$ *right*(c) for all $i \in L(c)$.

Let $s_{t,i}$ denote segment i on track t, where the index i is numbered from left to right along the track. Let $left(s_{t,i})$ and $right(s_{t,i})$ be the leftmost and rightmost columns in which the segment is present respectively. Since the switches between adjacent segments are placed between consecutive columns, we have $left(s_{t,i+1}) = right(s_{t,i}) + 1$ for all $t = 1, \ldots, T$ and $i = 1, \ldots, g_t - 1$.

In dogleg-free segment routing, a net is assigned to at most one track. A routing R of a set of nets is an assignment of each net to a track such that no segment is occupied by more than one net. A K-segment channel routing is a routing where each net occupies at most K segments on a track. If a net can be assigned to more than one track, the routing is called *generalized routing* [Roychowdhury et al. 1993] or doglegged segment routing.

3. CHARACTERIZATION OF DOGLEG-FREE SEGMENT ROUTING

Following the method of Devadas [1989], we define a variable vector $\vec{v}(c)$ for each net c as the binary (encoding) representation of the track index where net c is assigned to, that is, $\vec{v}(c) = v_1(c)v_2(c)\ldots v_w(c)$ where $w = \lceil log_2(T) \rceil$ and $v_i(c) \in \{0, 1\}$ for $i = 1, 2, \ldots, w$.

Since each net has to be assigned to one track,

$$\forall c \in C.\ 0 \leq \vec{v}(c) \leq T - 1 \tag{1}$$

where $\bigwedge$ denotes the boolean conjunctive operator.

Since all the definitions of dogleg-free segment routing involve only the leftmost and rightmost columns of each net, the solution applies to two-terminal as well as multiterminal routing.

3.1 Unlimited Segment Routing

Each segment can be occupied by at most one net. We define an occupancy function h for net c and segment s on track t:

$$h(c, s, t) = (right(s) \geq left(c)) \wedge (left(s) \leq right(c)) \wedge (\vec{v}(c) = t - 1)$$

The function h is true if and only if segment s on track t is occupied by net c. The segment occupancy constraint for each net can be defined in terms of the occupancy function h:

$$\bigwedge_{t=1}^{T} \bigwedge_{s \in G_t} \left(\bigwedge_{m,n \in C}^{m \neq n} \neg(h(m, s, t) \wedge h(n, s, t)) \right) \tag{2}$$

The constraint states that for any arbitrary pairs of nets m and n, the segment s on track t cannot be occupied by both nets.

The necessary and sufficient condition for dogleg-free unlimited segment routing is the conjunction of formulae (1) and (2).

3.2 *K*-Segment Routing

Let $mleft(c, t)$ and $mright(c, t)$ be the *identification numbers* (IDs) of the leftmost segment and the rightmost segment occupied by net c on track t, respectively.

$$\begin{aligned} mleft(c,t) &= min\{j \,|(right(s_{tj}) \geq left(c)) \wedge (left(s_{tj}) \leq right(c)) \wedge (s_{tj} \in G_t)\} \\ mright(c,t) &= max\{j \,|(right(s_{tj}) \geq left(c)) \wedge (left(s_{tj}) \leq right(c)) \wedge (s_{tj} \in G_t)\} \end{aligned}$$

Given *left*(s) and *right*(s), *left*(c) and *right*(c) are deducible from the problem specification, the values of *mleft*(c, t) and *mright*(c, t) can be computed before the satisfiability check.

To limit the maximum number of segments that the nets of a segmented channel can occupy, we need to ensure that the difference between indices for the leftmost segment and the rightmost segment on the track to which this net is assigned is less than K. Otherwise, the net cannot be assigned to this track. The K-segment constraint is:

$$\forall t \in \Gamma . \forall c \in C. \ (\vec{v}(c) = t) \Rightarrow (K > mright(c,t) - mleft(c,t)) \tag{3}$$

The overall routability check for dogleg-free K-segment routing is the conjunction of formulae (1), (2), and (3).

3.3 Performance Driven Tracks

A performance driven track is a track whose delay is shorter than the average track delay. We use Ψ to represent the set of nets on the critical path. We use $\wp$ to represent the set of performance driven tracks.

When connecting nets on the critical path, we force them to be assigned on certain *performance driven tracks* to meet the timing requirements. This requirement introduces an additional constraint:

$$\forall c \in \Psi . \exists t \in \wp . \ \vec{v}(c) = t \tag{4}$$

The overall routability check for performance driven track routing is the conjunction of formulas (1), (2), and (4).

3.4 An Example

Figure 1 gives an example for dogleg-free segmented channel routing. The upper half of the figure shows the net connections that we want to establish. The lower half of the figure shows the segmented channel that we have to route on. In this segmented channel, we have 10 columns ($N = 10$) and 3 tracks ($T = 3$). The segments for each track are listed in Table I. There are 4 net connections ($M = 4$) that need to be routed. The leftmost and rightmost columns for each net connection are listed in Table II.

Based on the number of tracks ($T = 3$), we use 2 bits ($w = \lceil log_2(T) \rceil = 2$) to encode the track assignment of each net:

$$\begin{aligned} \vec{v}(c_1) &= v_1(c_1)v_2(c_1) \\ \vec{v}(c_2) &= v_1(c_2)v_2(c_2) \\ \vec{v}(c_3) &= v_1(c_3)v_2(c_3) \\ \vec{v}(c_4) &= v_1(c_4)v_2(c_4). \end{aligned}$$

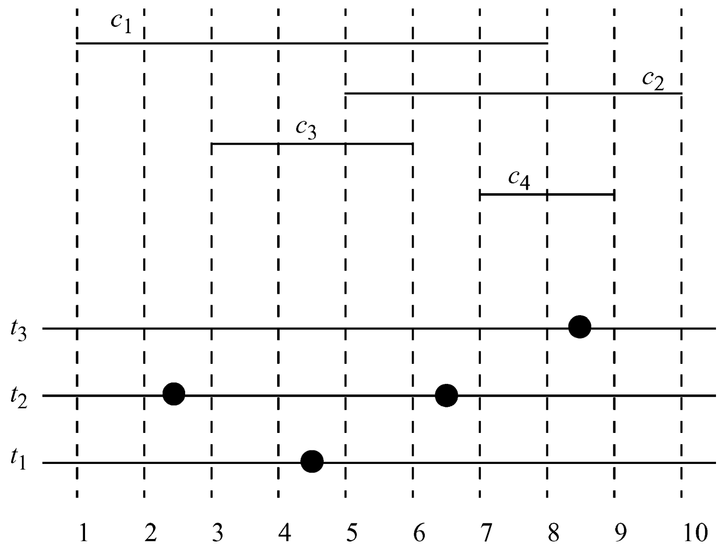

Fig. 1. Segmented channel routing example.

Table I. Segments for Each Track

Track	Segment	left(s)	right(s)
t_1	s_{11}	1	8
	s_{12}	9	10
t_2	s_{21}	1	2
	s_{22}	3	6
	s_{23}	7	10
t_3	s_{31}	1	4
	s_{32}	5	10

Table II. Net Connections

Net	left(c)	right(c)
c_1	1	8
c_2	5	10
c_3	3	6
c_4	7	9

The constraint for the domain of each track assignment is based on formulas (1):

$$0 \leq \vec{v}(c_1) \leq 2$$
$$0 \leq \vec{v}(c_2) \leq 2$$
$$0 \leq \vec{v}(c_3) \leq 2$$
$$0 \leq \vec{v}(c_4) \leq 2.$$

Each segment can be occupied by at most one net. We need to compute the occupancy function h for each net c and each segment s on each track t. Here, we demonstrate the occupancy function for the first net and the first segment

Table III. $h(c, s, t)$

t	s	c_1	c_2	c_3	c_4
t_1	s_{11}	$\neg v_1(c_1) \wedge \neg v_2(c_1)$	$\neg v_1(c_2) \wedge \neg v_2(c_2)$	$\neg v_1(c_3) \wedge \neg v_2(c_3)$	$\neg v_1(c_4) \wedge \neg v_2(c_4)$
	s_{12}	FALSE	$\neg v_1(c_2) \wedge \neg v_2(c_2)$	FALSE	$\neg v_1(c_4) \wedge \neg v_2(c_4)$
t_2	s_{21}	$\neg v_1(c_1) \wedge v_2(c_1)$	FALSE	FALSE	FALSE
	s_{22}	$\neg v_1(c_1) \wedge v_2(c_1)$	$\neg v_1(c_2) \wedge v_2(c_2)$	$\neg v_1(c_3) \wedge v_2(c_3)$	FALSE
	s_{23}	$\neg v_1(c_1) \wedge v_2(c_1)$	$\neg v_1(c_2) \wedge v_2(c_2)$	FALSE	$\neg v_1(c_4) \wedge v_2(c_4)$
t_3	s_{31}	$v_1(c_1) \wedge \neg v_2(c_1)$	$v_1(c_2) \wedge \neg v_2(c_2)$	$v_1(c_3) \wedge \neg v_2(c_3)$	FALSE
	s_{32}	$v_1(c_1) \wedge \neg v_2(c_1)$	$v_1(c_2) \wedge \neg v_2(c_2)$	$v_1(c_3) \wedge \neg v_2(c_3)$	$v_1(c_4) \wedge \neg v_2(c_4)$

on the first track:

$$\begin{aligned} h(c_1, s_{11}, t_1) &= (right(s_{11}) \geq left(c_1)) \wedge (left(s_{11}) \leq right(c_1)) \wedge (\vec{v}(c_1) = 1 - 1) \\ h(c_1, s_{11}, t_1) &= (8 \geq 1) \wedge (1 \leq 8) \wedge (v_1(c_1)v_2(c_1) = 00) \\ h(c_1, s_{11}, t_1) &= \neg v_1(c_1) \wedge \neg v_2(c_1). \end{aligned}$$

Almost all the values needed to compute the occupancy function are already given in the problem specification. The only exception is the track assignment for each net $\vec{v}(c)$, which is exactly what we want to find out from the satisfiability formulation. The remaining values for the occupancy function are shown in Table III. The column and row headers represent the various choices for the parameters (c, s, t) for function h. The entries of the table indicate the values of the function under these parameters.

Given function h, we formulate the segment occupancy constraint based on formulae (2). The occupancy function values shown in Table III are substituted into the formula to obtain the actual constraint:

$$\bigwedge_{t=1}^{3} \bigwedge_{s \in G_t} \left(\bigwedge_{m,n \in C}^{m \neq n} \neg(h(m, s, t) \wedge h(n, s, t)) \right)$$

Routability of the segmented channel is the conjunction of the domain constraints from formula (1) and the segment occupancy constraint from formula (2). A solution for dogleg-free routing is shown in Figure 2. Each net connection is mapped to a corresponding track.

4. CHARACTERIZATION OF DOGLEGGED SEGMENT ROUTING

Given a segmented channel, we aim at finding a generalized routing where doglegs are allowed. With technology advance, doglegs will be allowed by additional hardware support. Doglegs can help to reduce the wiring area to reach a more compact layout. The problem is more complicated than dogleg-free routing, since we cannot use one vector $\vec{v}(c)$ to encode (denote) all the tracks to which the net c is assigned.

4.1 Track Variables

Let $(left(c, l_1), (l_1 + 1, l_2), (l_2 + 1, l_3), \ldots, (l_{p-1} + 1, right(c)))$ be p $(p \geq 1)$ parts of a routing for net c on different tracks. Let $p_m(c)$ be the maximum number of parts for net c, we have:

$$\begin{gathered} 1 \leq p \leq p_m(c) \\ p_m(c) = right(c) - left(c). \end{gathered}$$

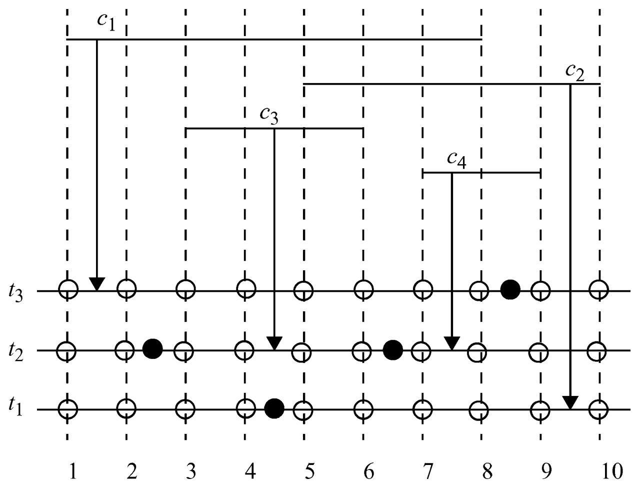

Fig. 2. Routed example.

For each net c, use a set of variable vectors to denote the tracks where the net is assigned to:

—$\vec{u}(c, 1)$ is the track of net c between *left*(c) and *left*$(c) + 1$;
—$\vec{u}(c, 2)$ is the track of net c between *left*$(c) + 1$ and *left*$(c) + 2$;
—$\vec{u}(c, 3)$ is the track of net c between *left*$(c) + 2$ and *left*$(c) + 3$;
—......
—$\vec{u}(c, p_m(c))$ is the track of net c between *right*$(c) - 1$ and *right*(c).

Each variable vector $\vec{u}(c, i)$ is a binary (encoding) representation of a track used by net c:

$$\vec{u}(c, i) = u_1(c, i), \ldots, u_q(c, i) \text{ for } i = 1, 2, \ldots, p_m(c),$$

where $q = \lceil log_2 T \rceil$ and $u_j(c, i) \in \{0, 1\}$ for $j = 1, 2, \ldots, q$.

The constraint for the domain of $\vec{u}(c, i)$ is:

$$0 \leq \vec{u}(c, i) \leq T - 1 \text{ for all } c \in C \text{ and } i = 1, 2, \ldots, p_m(c). \tag{5}$$

4.2 Horizontal Wiring Constraints

For every part of net c on track $\vec{u}(c, i)$, we can deduce the left and right columns of that part of the connection:

$$\begin{aligned} \mathit{pleft}(c, i) &= \mathit{left}(c) + i - 1 \text{ for } i = 1, 2, \ldots, p_m(c) \\ \mathit{pright}(c, i) &= \mathit{left}(c) + i \text{ for } i = 1, 2, \ldots, p_m(c). \end{aligned}$$

The left and right columns $(\mathit{pleft}(c, i), \mathit{pright}(c, i))$ for each part are effectively adjacent to each other. Thus $\mathit{pright}(c, i) = \mathit{pleft}(c, i) + 1$.

We define an occupancy function h_d for the ith part of net c and segment s on track t:

$$h_d(c, i, s, t) = (\mathit{right}(s) \geq \mathit{pleft}(c, i)) \wedge (\mathit{left}(s) \leq \mathit{pright}(c, i)) \wedge (\vec{u}(c, i) = t - 1),$$

where $i = 1, 2, \ldots, p_m(c)$.

The function h_d is true if and only if segment s on track t is occupied by the ith part of net c. The segment occupancy constraint for each net can be defined in terms of the occupancy function h_d:

$$\bigwedge_{t=1}^{T} \bigwedge_{s \in G_t} \left(\bigwedge_{m,n \in C}^{m \neq n} \left(\bigwedge_{i=1}^{p_m(m)} \bigwedge_{j=1}^{p_m(n)} \neg(h_d(m,i,s,t) \wedge h_d(n,j,s,t)) \right) \right). \tag{6}$$

Notice the similarity between formula (2) and formula (6). Their difference is their occupancy functions h_d and h. The constraint here in formula (6) states that for any arbitrary pairs of wiring pieces, (ith part of net m and jth part of net n), the segment s on track t cannot be occupied by both nets.

4.3 Vertical Wiring Constraints

Each column can be occupied by at most one net. A column is occupied if any net has a terminal on that column or if any net changes tracks on that column. The terminal locations are already given by the problem specification. But the places where each net changes tracks must be resolved so that they do not conflict with the terminals or track changes of other nets.

Each net c changes tracks on column i if and only if it spans over that column and its parts on the left side and right side of column i are assigned to different tracks. We use a boolean value $x(c,i)$ to denote a change of track for net c on column i :

$$x(c,i) = \begin{cases} \mathit{FALSE} & (i \leq \mathit{left}(c)) \vee (i \geq \mathit{right}(c)) \\ (\vec{u}(c,i) \neq \vec{u}(c,i+1)) & \text{otherwise.} \end{cases}$$

If net c does change tracks on column i, this column is occupied by the net c. This column cannot be used by another net (either as a terminal or to change tracks). This requirement is formulated as the following constraint:

$$\bigwedge_{t=1}^{N} \left(\bigwedge_{m,n \in C}^{m \neq n} (x(m,i) \Rightarrow \neg(x(n,i) \vee (i \in L(n)))) \right). \tag{7}$$

The necessary and sufficient condition for doglegged segmented channel routing is the conjunction of formulas (5), (6) and (7).

4.4 *K*-Segment Routing

To limit the maximum number of segments used by any net, we need to find out if segment s on track t is occupied by net c.

$$\mathit{used}(c,s,t) = \exists i \in \{1, \ldots, p_m(c)\}.\, h_d(c,i,s,t).$$

The total number of segments used by net c for all segments on all tracks cannot exceed K.

$$K \geq \sum_{t=1}^{T} \sum_{s \in G_t} \mathit{used}(c,s,t) \quad \text{for all } c \in C. \tag{8}$$

Notice that the arithmetic addition operator that is used to implement the summation is applied to boolean formulas here. We compute the addition over Boolean symbolic formulas and create a constraint that limits the sum to be less than K.

The K-segment generalized routability is the conjunction of formulas (5), (6), (7) and (8).

4.5 Track Limitations

To limit the maximum number of distinct tracks used by each net c, we define a Boolean function for the distinct formulation for each part i of net c:

$$\begin{aligned} \mathit{distinct}(c,1) &= \mathit{TRUE} \\ \mathit{distinct}(c,i) &= \forall j \in \{1,\ldots,i-1\}.\,(\vec{u}(c,i) \neq \vec{u}(c,j)) \quad \text{for } i = 2,3,\ldots,p_m(c). \end{aligned}$$

The constraint on the maximum number of distinct tracks is phrased using the sum of the above Boolean bits:

$$l \geq \sum_{t=1}^{p_m(c)} \mathit{distinct}(c,i) \quad \text{for all } c \in C. \tag{9}$$

A generalized routing that uses at most l different tracks for routing any net is essentially the conjunction of formulas (5), (6), (7) and (9).

4.6 Column Limitations

Given a segmented channel, we want to find a generalized routing where each net c can only switch tracks at predetermined columns. Let $\Theta(c)$ be the set of columns where net c is allowed to switch tracks. The constraint can be formulated as:

$$\bigwedge_{c\in C}\left((p_m(c) > 1) \Rightarrow \bigwedge_{i=1}^{p_m(c)-1} (((\mathit{left}(c)+i) \notin \Theta(c)) \Rightarrow (\vec{u}(c,i) = \vec{u}(c,i+1)))\right). \tag{10}$$

A generalized routing where connections can switch tracks only at predetermined columns is essentially the conjunction of formulas (5), (6), (7) and (10).

5. EXPERIMENTS

We formulated the segmented routability problems with Boolean equations in Sections 3 and 4, compiled them [Junttila and Niemela 2000] into DIMACS format, and used zChaff [Zhang et al. 2001; Moskewicz et al. 2001] to solve these problem instances. The runtime is in seconds on an Intel® Pentium® III processor at 850MHz with 1GB RDRAM (only small part of memory has been used).

Table IV shows the routability checks for conventional segmented channels with no doglegging. The columns N, M, T corresponds to the number of columns, nets, tracks respectively. The channel segmentation and net connection specifications are both randomly generated. It takes a lot longer time to verify unroutability (for $N = 100$, $M = 18$, $T = 10$) than routability. This is because it's much easier for the SAT checkers to find one satisfiable instance

Table IV. Dogleg-Free Segmented Routability

N	M	T	clauses	literals	routability	zChaff
30	15	10	4979	10271	Y	0.04
50	15	15	6651	13675	Y	0.01
50	18	15	9564	19576	Y	0.01
50	20	15	11594	23686	Y	0.03
75	25	20	26141	53555	Y	55.04
100	15	10	5100	10544	Y	1.22
100	18	10	7163	14739	N	17171.20
100	30	25	46340	94328	Y	0.41
100	30	30	54092	109892	Y	0.17
100	30	36	66549	136157	Y	0.53
100	40	36	111937	227953	Y	1.42
100	45	36	140375	285339	Y	1.35
100	50	36	172741	350581	Y	3.93
500	75	36	169633	344365	Y	2.84

Table V. Routing for Zhu and Wong's Segmented Channels

	Variables (k)					Clauses (k)					zChaff runtime (s)					sat	not	out
	min	1/4	mid	3/4	max	min	1/4	mid	3/4	max	min	1/4	mid	3/4	max	%	%	%
D1	4.7	20.5	45.4	79.8	134.1	27.5	122.5	271.1	477.1	802.9	0.02	0.13	0.42	0.74	1112.59	82.1	0	17.9
D2	4.8	13.2	30.1	54.0	91.5	28.4	78.5	179.9	322.7	547.7	0.02	0.14	0.39	1.2	647.52	96.8	0	3.2
D3	4.4	41.9	79.8	147.6	267.1	26.1	250.2	477.7	883.8	1600.0	0.01	0.22	0.62	1.09	1145.89	82.1	1.1	16.8
D4	5.8	22.4	82.4	181.8	444.2	34.3	133.9	492.8	1088.6	2661.6	0.02	0.1	0.56	1.24	418.76	85.3	1.1	13.6
D5	4.9	21.3	56.7	97.7	200.4	28.8	127.1	339.2	584.8	1199.8	0.03	0.16	0.61	1.16	194.53	82.1	1.1	16.8
D6	4.6	15.4	32.0	60.0	113.2	27.2	91.9	191.0	359.1	677.7	0.02	0.09	0.28	0.68	418.35	92.6	1.1	6.3
D7	3.1	24.0	160.2	361.5	887.8	18.0	143.2	958.4	2165.3	5321.2	0.01	0.06	0.59	2.04	559.11	72.6	1.1	26.3
Geo	3.4	56.9	143.4	320.5	722.1	20.1	339.9	858.3	1919.5	4327.7	0.01	0.19	0.74	1.68	792.81	74.7	0	25.3
Norm	5.4	22.5	62.8	114.2	179.1	32.2	134.5	376.3	684.3	1073.2	0.02	0.11	0.62	1.44	691.49	89.5	1.1	9.4
Poiss	3.1	31.6	92.9	208.6	382.2	18.4	188.3	555.6	1248.6	2289.2	0.01	0.12	0.58	1.48	123.45	84.2	0	15.8

than searching every possible case for unsatisfiability. The parameters of some channel models ($N = 100$, $T = 36$) are close to Actel's ACT2 family A1280 FPGA [Actel Corporation 1991].

For more practical benchmarks, we used the segmentation design by Zhu and Wong [1992], which is also the basis of study in Yang et al. [2000] and Massoud et al. [1994]. For each case, a set of 300 net connection specifications were created, and the 2-segmentation model was generated using a program described in Zhu and Wong [1992]. The parameters of the channel model, ($N = 100$, $T = 36$) were based on Actel's ACT2 family A1280 FPGA. The result is shown in Table V. We used ten different types of channel distributions: D1-D7, Geometric, Normal and Poisson. For each distribution, we report the number CNF variables and clauses, and the zChaff [Zhang et al. 2001] runtime for finished instances. We list the percentage of routed (sat%) instances, unsatisfiable (not%) instances, and timed out (out%) instances. Detailed statistics were also listed for variables, clauses and runtime. For each statistic, we report the minimum (min), 25% lower quartile (1/4), median (mid), 75% upper quartile (3/4), and maximum (max). From these statistics, we can see the majority of the benchmark instances are routable and finished within two seconds on the Pentium® III, which is comparable to prior published [Yang et al. 2000] heuristic results mostly around 40 to 100 seconds on SUN Sparc, while there are some outlying instances that take extremely long time for the SAT solver.

Table VI. Doglegged Segmented Routability

N	M	T	K	t-limit	c-limit	clauses	literals	routability	zChaff
20	6	5	∞	none	none	1447	3167	Y	0.01
20	7	5	∞	none	none	2061	4491	Y	0.03
20	8	5	∞	none	none	2155	4619	N	0.02
20	9	5	∞	none	none	1697	3483	N	0.02
20	9	7	∞	none	none	2643	5473	Y	0.01
50	20	15	∞	none	none	32396	70882	Y	0.60
75	20	20	2	none	none	495503	1024543	Y	25.19
75	20	20	∞	2	none	206905	446665	Y	32.92
75	20	20	∞	none	50%	73018	164946	Y	21.54
75	20	20	∞	2	50%	171835	364891	Y	11.07
75	20	20	2	none	50%	428452	867892	Y	11.40
75	25	20	∞	none	none	126027	283005	Y	18999.80

Table VI shows the routability checks for segmented channels with doglegging. Again, the channel segmentation and net connection specifications are randomly generated. Since there are more flexibility in doglegging, the search time is generally longer compared to dogleg-free cases. For the case of 75 nets on 20 columns and 20 tracks, we experimented with 2 segment limitation, 2 track limitation, and column limitations where alternate nets can switch tracks on every other columns (i.e., 50% of the total columns), and combinations of column with segment/track limitations. Notice that the runtime with column limitations are shorter than the runtime without column limitations with the same number of nets, columns and tracks. This is because the column limitations in formula (10) force the variable vectors associated with columns in the unswitchable columns (i.e., outside the set $\Theta(c)$) to be equal. Thus, significantly reducing the search space for those variables.

6. CONCLUSION

We studied the routability of segmented channel routing via satisfiability. Our method encodes the horizontal and vertical constraints as Boolean conditions. The routability constraint is satisfiable if the conjunction of all the constraints are satisfiable. We presented satisfiability formulations for dogleg-free and doglegged routing. In general, doglegged routing cases have more flexibility than dogleg-free cases and takes longer runtime; but this runtime can be reduced by introducing column limitations in doglegging. Experimental results demonstrated that our approach is time-efficient and is applicable to large problem instances. Our approach can be extended to take into consideration other variety of routing problems, such as permutation routing or terminal movable routing.

ACKNOWLEDGMENTS

The authors would like to thank Prof. W. K. Mak and Dr. K. Zhu for the experimental benchmarks.

REFERENCES

Actel Corporation 1991. *ACT Family FPGA Databook*. Actel Corporation.

Aloul, F. A., Ramani, A., Markov, I. L., and Sakallah, K. A. 2002. Solving difficult SAT instances in the presence of Symmetry. In *Proceedings of the Design Automation Conference*. 731–736.

Brown, S., Rose, J., and Vranesic, Z. G. 1992. A detailed router for field programming gate arrays. *IEEE Trans. CAD 11*, 620–628.

Devadas, S. 1989. Optimal layout via Boolean satisfiability. In *Proceedings of the ACM/IEEE ICCAD*. ACM, New York, 294–297.

Junttila, T. and Niemela, I. 2000. Towards an efficient tableau method for Boolean circuit satisfiability checking. In *Proceedings of the 1st International Conference on Computational Logic*, J. L. et al., Ed. Lecture Notes in Artificial Intelligence, vol. 1861. Springer-Verlag, New York, 553–567.

Limieux, G. and Brown, S. 1993. A detailed router for allocating wire segments in FPGAs. In *Proceedings of the ACM Physical Design Workshop*. ACM, New York.

Massoud, P., Nobandegani, B., and Preas, B. 1994. Design and analysis of segmented routing channels for row-based FPGA's. *IEEE Trans. CAD 13*, 12.

Moskewicz, M., Madigan, C., Zhao, Y., Zhang, L., and Malik, S. 2001. Chaff: Engineering an efficient SAT solver. In *Proceedings of the Design Automation Conference*.

Nam, G., Aloul, F., Sakallah, K., and Rutenbar, R. 2001. A comparative study of two Boolean formulations of FPGA Detailed routing constraints. In *Proceedings of the ACM International Symposium on Physical Design*. ACM, New York, 222–227.

Nam, G., Sakallah, K., and Rutenbar, R. 1999. Satisfiability-based detailed FPGA routing. In *Proceedings of the IEEE International Conference on VLSI Design*. IEEE Computer Society Press, Los Alamitos, Calif.

Roy, K. 1993. A bounded search algorithm for segmented channel routing for FPGA's and associated channel architecture issues. *IEEE Trans. CAD 12*, 11, 1695–1705.

Roychowdhury, V. P., Greene, J., and Gamal, A. E. 1993. Segmented channel routing. *IEEE Trans. CAD 12*, 1, 79–95.

Song, X., Hung, W. N. N., Mishchenko, A., Chrzanowska-Jeske, M., Coppola, A., and Kennings, A. 2003. Board-level multiterminal net assignment for the partial cross-bar architecture. *IEEE Trans. VLSI Syst. 11*, 3 (June), 511–514.

Thakur, S., Chang, Y. W., Wong, D. F., and Muthukrishnan, S. 1997. Algorithms for an FPGA switch module problem with application to global routing. *IEEE Trans. CAD 16*.

Trimberger, S. 1995. Effects of FPGA Architecture on FPGA Routing. In *Proceedings of the ACM/IEEE Design Automation Conference*. ACM, New York.

Wood, R. G. and Rutenbar, R. A. 1998. FPGA routing and routability estimation via Boolean satisfiability. *IEEE Trans. VLSI Syst. 6*, 2.

Wu, Y. L. and Marek-Sadowska, M. 1993. Graph based analysis of FPGA routing. In *Proceedings of the European Design Automation Conference*. 104–109.

Xu, H., Rutenbar, R. A., and Sakallah, K. 2003. sub-SAT: A formulation for relaxed Boolean satisfiability with applications in routing. *IEEE Trans. CAD*, 635–638.

Yang, C., Chen, S., Ho, J., and Tsai, C. 2000. Efficient routability check algorithms for segmented channel routing. *ACM Trans. Design Automat. Electron. Syst. 5*, 3, 735–747.

Zhang, L., Madigan, C. F., Moskewicz, M. W., and Malik, S. 2001. Efficient conflict driven learning in a Boolean satisfiability solver. In *Proceedings of the International Conference on Computer-Aided Design*. 279–285.

Zhu, K. and Wong, D. F. 1992. On channel segmentation design for row-based FPGAs. In *Proceedings of the International Conference on Computer-Aided Design*.

Received April 2003; revised August 2003; accepted January 2004