Switch-Matrix Architecture and Routing for FPDs

Guang-Min Wu and Yao-Wen Chang

Department of Computer and Information Science
National Chiao Tung University
Hsinchu 300, Taiwan ROC
{gis85815, ywchang}@cis.nctu.edu.tw

Abstract

An FPD switch module M with w terminals on each side is said to be *universal* if every set of nets satisfying the dimensional constraint (i.e., the number of nets on each side of M is at most w) is simultaneously routable through M [8]. Chang, Wong, and Wong have identified a class of universal switch *blocks* in [8]. In this paper, we consider the design and routing problems for another popular model of switch modules called *switch matrices*. Unlike switch *blocks*, we prove that there exist no universal switch *matrices*. Nevertheless, we present *quasi-universal* switch matrices which have the maximum possible routing capacities among all switch matrices of the same size, and show that their routing capacities converge to those of universal switch blocks. Each of the quasi-universal switch matrices of size w has a total of only $14w-20$ ($14w-21$) switches if w is even (odd), $w>1$, compared to a fully populated one which has $3w^2-2w$ switches. We prove that no switch matrix with less than $14w-20$ ($14w-21$) switches can be quasi-universal. Experimental results demonstrate that the quasi-universal switch matrices improve routabilty at the chip level.

1 Introduction

Field-Programmable Devices (*FPDs*) refer to any digital, user-configurable integrated circuits used to implement logic functions. Due to their short production time and low prototyping cost, FPDs have become a very popular alternative to realizing logic designs. Figure 1 shows the architectures of major commercially available FPDs. As illustrated in Figures 1(a) and (b), a *Field-Programmable Gate Array* (*FPGA*) consists of an array of logic modules that can be connected by general routing resources. The logic modules contain combinational and sequential circuits that implement logic functions. The routing resources consist of horizontal and vertical channels and their intersection areas. An intersection area of a horizontal and a vertical channels is referred to as a *switch module*. A net can change its routing direction via a switch module; this requires using at least one programmable switch inside the switch module. A large circuit that cannot be accommodated into a single FPGA is divided into several parts; each part is realized by an FPGA and these FPGAs are then interconnected by a *Field-Programmable Interconnect Chip* (*FPIC*) (see Figure 1(c)). In a *Complex Programmable Logic Device* (*CPLD*), logic modules are surrounded by *continuous* horizontal and vertical routing tracks (see Figure 1(d)). Similar to FPGAs, an intersection area of a horizontal and a vertical channels in an FPIC or a CPLD is also referred to as a switch module.

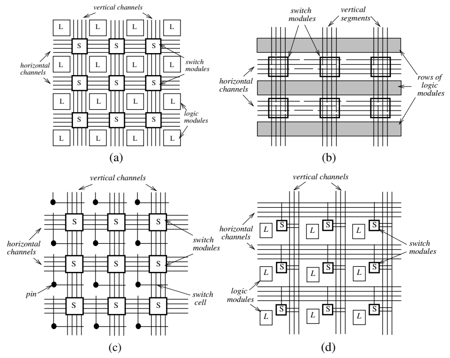

Figure 1: Major FPD architectures. (a) The symmetric-array FPGA model. (b) The row-based FPGA model. (c) The FPIC model. (d) The CPLD model.

Recent works by [5, 16] have shown that the feasibility of FPGA design is constrained more by routing resources than by logic resources, and often routing delays, rather than logic-module delays, dominate the performance of FPGAs. Therefore it is desirable to facilitate routing in the design of FPGAs and FPICs. Switch modules are the most important component of the routing resources in FPDs. Studies by [6, 8, 14] have shown that the higher the routability of the switch modules, the smaller the track count is needed to achieve 100% routing completion. Hence, increasing the routability of a switch module also improves the area performance of a router. Therefore, it is of significant importance to consider switch-module design. In current technology, FPD programmable switches usually consume a large amount of area. Due to the area constraint, the number of switches that can be placed in a switch module is usually limited, implying limited routability. Therefore, there is a basic trade-off between routability and area for switch-module architectures.

There are two types of switch modules in commercially available FPDs, *switch matrices* and *switch blocks*. (See Figure 2 for their models.) The effects of switch-module architectures on routing for the symmetric-array-based FPGAs were first studied experimentally by Rose and Brown [14]. A theoretical study of flexibility and routability was later presented based on a stochastic model [6]. The primary conclusion in both of the studies in [6, 14] is that high pin-to-track connectivity together with relatively low switch-module connectivity is a better solution to the routability and area trade-off. Therefore, the architecture of a switch module is of particular importance, due to a relatively small switch population in a switch module. Chang *et al.* [8] proposed a class of high-routability switch blocks and analyzed three types of well-known switch blocks; they showed theoretically and experimentally that switch blocks with higher routability usually lead to better area performance, which confirms the findings by [6, 14].

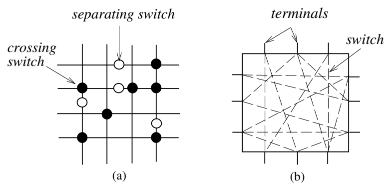

Figure 2: Switch-module models. (a) Switch matrix. (b) Switch block.

In this paper, we focus on switch *matrices*. Not much work has been reported on switch-matrix design. Zhu, Wong, and Chang in [19] first explored the feasibility conditions for switch matrices and presented a design heuristic based on a stochastic approach. Chang, Wong, and Wong in [7] applied a network-flow based heuristic for switch-module design. Sun *et al.* in [15] studied the effects of using the two switch-module architectures—switch matrices and switch blocks—on routing. Based on the study in [15], an FPGA/FPIC with switch matrices in general needs fewer switches but more routing tracks for routing completion than that with switch blocks. This work shows the trade-offs in using the two types of switch modules.

In this paper, we consider the design and routing problems for *universal* switch matrices. An FPD switch module M with w terminals on each side is said to be *universal* if every set of nets satisfying the dimensional constraint (i.e., the number of nets on each side of M is at most w) is simultaneously routable through M [8]. We prove that there exist no universal switch *matrices*. Nevertheless, we present *quasi-universal* switch matrices which have the maximum possible routing capacities among all switch matrices of the same size, and show that their routing capacities converge to those of universal switch modules. Each of the quasi-universal switch matrices of size w has a total of only $14w - 20$ ($14w - 21$) switches if w is even (odd), $w > 1$, compared to a fully populated one which has $3w^2 - 2w$ switches. We prove that no switch matrix with less than $14w - 20$ ($14w - 21$) switches can be quasi-universal. Experimental results demonstrate that the quasi-universal switch matrices improve routabilty at the chip level.

The rest of the paper is organized as follows. Section 2 gives the preliminaries for our problem. Section 3 explores the feasibility conditions of switch matrices. Section 4 presents the quasi-universal switch matrices. Section 5 gives an example graph modeling for detailed routing. Finally, experimental results are reported in Section 6.

2 Preliminaries

A *switch matrix* consists of a grid of w horizontal and w vertical tracks. We represent a switch matrix by M_w (or M if w is not of concern). There are two types of switches in a switch matrix, *crossing switches* and *separating switches*. (See Figure 2(a).) If a crossing switch at the intersection of a horizontal and a vertical tracks is 'ON," the two tracks are connected; if it is 'OFF," the tracks are not connected and are thus electrically non-interacting. If a separating switch on a track is 'OFF," the track is split into two electrically non-interacting routing segments so that the terminals on opposite sides can be used independently; if it is 'ON," the track becomes a single electrical track. In Figure 2(a), the crossing switches are represented by solid circles and the separating switches by hollow circles. Switch matrices are used in various symmetric-array FPGAs [4, 10], row-based FPGAs [1, 9], FPICs [3], and CPLDs [2].

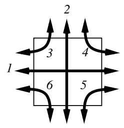

Figure 3: Six types of connections.

A *connection* is an electrical path between two terminals on different sides of a switch module. Connections can be of six types, each of which is characterized by two sides of a module, as shown in Figure 3. For example, type-6 connections connect terminals on the left and the bottom sides of a module. Type-1 and type-2 connections are *straight connections* whereas the others are *bent connections*.

A *routing requirement vector* (RRV) $\vec{n}$ is a six-tuple $(n_1, n_2, \ldots, n_6)$, where n_i is the number of type-i connections required to be routed through a switch module, $0 \leq n_i \leq w$, $i = 1, 2, \ldots, 6$. An RRV $\vec{n}$ is said to be *routable* on a switch module M, denoted by $\vec{n} \propto M$, if there exists a routing for $\vec{n}$ on M.

An RRV $\vec{m} = (m_1, m_2, \ldots, m_6)$ is *dominated* by another RRV $\vec{n} = (n_1, n_2, \ldots, n_6)$, denoted by $\vec{m} \leq \vec{n}$, if and only if $m_i \leq n_i$, $\forall i, 1 \leq i \leq 6$. Any RRV $\vec{m}$ is routable on a switch matrix M if there exists an RRV $\vec{n}$ that is routable on M and $\vec{m} \leq \vec{n}$; i.e., $\vec{n} \propto M \wedge \vec{m} \leq \vec{n} \Longrightarrow \vec{m} \propto M$. An RRV $\vec{n}$ is called *maximally routable* on a switch module M if $\vec{n} \propto M$ and no additional nets can be routed through M; in other words, $\vec{n}$ is maximally routable if $\vec{n}$ is not dominated by any other routable RRV on M.

The *routing capacity* of a switch module M is referred to as the number of distinct routable RRVs on M; that is, the routing capacity of M is the cardinality $|\{\vec{n} | \vec{n} \propto M\}|$. The *universal switch module* (*USM* for short) is defined in [8] as follows:

Definition 1 *[8] A switch module M_w is called* universal *if the following set of inequalities is the necessary and sufficient conditions for an RRV $\vec{n} = (n_1, n_2, \ldots, n_6)$ to be routable on M:*

$$
\begin{aligned}
n_1 + n_3 + n_6 &\leq w \qquad (1)\\
n_2 + n_3 + n_4 &\leq w \qquad (2)\\
n_1 + n_4 + n_5 &\leq w \qquad (3)\\
n_2 + n_5 + n_6 &\leq w. \qquad (4)
\end{aligned}
$$

Note that the number of nets routed through each side of M can not exceed w; this dimensional constraint is characterized by the preceding four inequalities, one for each side.

Therefore, a USM has the maximum routing capacity, and it is desirable to find such a universal switch *matrix*, if any. In this paper, we design a class of switch matrices with best possible routing capacities and give the qualitative and quantitative analyses for the matrices.

3 Feasibility Conditions

In this section, we explore the feasibility conditions of switch matrices.

Lemma 1 *An RRV $\vec{n}$ is routable on a switch matrix M_w ($\vec{n} \propto M_w$) only if $\vec{n} = (w, w, 0, 0, 0, 0)$ or the following set of inequalities is satisfied:*

$$n_1 + n_3 + n_6 \leq w \quad (5)$$
$$n_2 + n_3 + n_4 \leq w \quad (6)$$
$$n_1 + n_4 + n_5 \leq w \quad (7)$$
$$n_2 + n_5 + n_6 \leq w \quad (8)$$
$$n_1 + n_2 + \max\{n_3 + n_5, n_4 + n_6\} \leq 2w - 1. \quad (9)$$

By Lemma 1, we have the following negative result.

Theorem 1 *There exists no universal switch matrix.*

Also, by Lemma 1, an RRV is simply unroutable on any switch matrix if the RRV fails to satisfy Inequalities (5)–(9). We call an RRV *non-trivial* if it satisfies Inequalities (5)–(9); otherwise, it is *trivial* (trivially unroutable). A switch matrix M is said to be *quasi-universal* if all non-trivial RRVs are routable on M. We give the formal definition of a *quasi-universal switch matrix* (*Q-USM* for short) as follows:

Definition 2 *A switch matrix M_w is called* quasi-universal *if Inequalities (5)–(9) are the necessary and sufficient conditions for an RRV $\vec{n} = (n_1, n_2, \ldots, n_6)$ to be routable on M_w.*

Since Inequalities (5)–(9) are the most fundamental routing constraints, a Q-USM has the maximum routing capacity among all switch matrices. It is thus of particular importance to find such class of switch matrices.

4 Quasi-Universal Switch Matrices (Q-USM)

In this section, we present procedures for constructing the Q-USM and give quantitative analyses for the matrices.

4.1 Procedures for Q-USM Design

Figures 4(a), (b), (c), and (d) show configurations for routing $\vec{n_1} = (0,0,6,0,5,0)$, $\vec{n_2} = (0,0,5,0,6,0)$, $\vec{n_3} = (0,0,0,6,0,5)$, and $\vec{n_4} = (0,0,0,5,0,6)$ on switch matrices $M1, M2, M3$, and $M4$ of size $w = 6$, respectively. In particular, they are the only configurations for routing $\vec{n_1}, \ldots, \vec{n_4}$ on $M1, \ldots, M4$. Clearly, for a switch matrix M to be quasi-universal, $\vec{n_i}, 1 \leq i \leq 4$, must be routable on M. This observation motivated our construction for the *diagonal switch matrix* shown in Figure 5. For the purpose of concise description, we refer to the *sub-diagonals* of a switch matrix M as the conceptual slanted lines parallel and adjacent to the two diagonals of M. Therefore there are four sub-diagonals on each switch matrix (see Figure 5). A diagonal switch matrix D is constructed based on the following three rules:

- Rule 1: Place crossing switches on the two diagonals of D;
- Rule 2: Place crossing switches on the four sub-diagonals of D;
- Rule 3: Place separating switches between the diagonals and sub-diagonals of D.

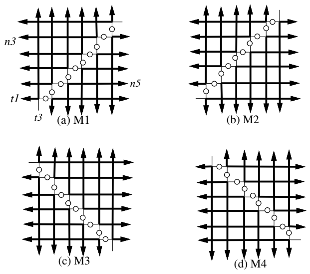

Figure 4: Routing configurations for four RRVs. (a) $\vec{n_1} = (0,0,6,0,5,0)$. (b) $\vec{n_2} = (0,0,5,0,6,0)$. (c) $\vec{n_3} = (0,0,0,6,0,5)$. (d) $\vec{n_4} = (0,0,0,5,0,6)$.

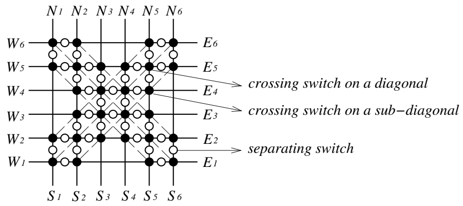

Figure 5: The diagonal switch matrix of size 6 (D_6).

4.2 Proof of the Quasi-Universality

In this subsection, we show that the diagonal switch matrices constructed by the procedures mentioned earlier are "cheapest" Q-USMs. Let

$$\begin{aligned} \Omega_w = \{(w, w, 0, 0, 0, 0)\} \cup \{(n_1, n_2, n_3, n_4, n_5, n_6)| \\ n_1 + n_3 + n_6 \leq w,\ n_2 + n_3 + n_4 \leq w, \\ n_1 + n_4 + n_5 \leq w, n_2 + n_5 + n_6 \leq w, \\ n_1 + n_2 + \max\{n_3 + n_5, n_4 + n_6\} \leq 2w - 1\}. \end{aligned}$$

To prove that a diagonal switch matrix D_w is quasi-universal, we must show that all RRVs in Ω_w are routable on D_w. The proof of the quasi-universality is based on mathematical induction and is informally described as follows. It is trivial to show that D_1 and D_2 are quasi-universal. Assume that all diagonal switch matrices are quasi-universal for $w \leq m$. To prove that the claim holds for the case where $w = m + 2$, we give constructive routings for all maximally routable RRVs in $\Omega_{m+2} - \Omega_m$ on D_{w+2} by applying the routings for the cases where $w \leq m$. Further, we claim that the number of switches used by each of our diagonal switch matrices is, in fact, the minimum requirement for a switch matrix to be quasi-universal.

To establish the proof, we first need some definitions and lemmas. Let $r_{+90}(\vec{n})$ ($r_{-90}(\vec{n})$) denote a 90-degree rotation counterclockwise (clockwise), $r_h(\vec{n})$ ($r_v(\vec{n})$) a reflection along the horizontal (vertical) axis for a routing configuration for $\vec{n}$. For example, if $\vec{n} = (1,2,3,4,5,6)$, $r_{+90}(\vec{n}) = (2,1,4,5,6,3)$, $r_{-90}(\vec{n}) = (2,1,6,3,4,5)$, $r_h(\vec{n}) =$

$(1,2,6,5,4,3)$, and $r_v(\vec{n}) = (1,2,4,3,6,5)$. We have the following definition and lemmas:

Definition 3 *Two routing configurations for $\vec{n}$ and $\vec{m}$ are* equivalent *if $\vec{n}$ ($\vec{m}$) can be obtained by performing a sequence of r_{+90}, r_{-90}, r_h, and/or r_v operations on $\vec{m}$ ($\vec{n}$). We say that $\vec{n}$ and $\vec{m}$ are in the same* equivalence class *and denote $\vec{n}$ and $\vec{m}$ by $\vec{n} \equiv \vec{m}$.*

Lemma 2 $\forall \vec{m}, \vec{n}, \vec{m} \equiv \vec{n} \Longrightarrow (\vec{m} \propto D \Longleftrightarrow \vec{n} \propto D)$.

Lemma 3 $(n_1, n_2, n_3, n_4, n_5, n_6) \propto D_w \Longrightarrow (n_1 + 2, n_2 + 2, n_3, n_4, n_5, n_6) \propto D_{w+2}$.

Lemma 4 $(n_1, n_2, n_3, n_4, n_5, n_6) \propto D_w \Longrightarrow (n_1, n_2, n_3 + 1, n_4 + 1, n_5 + 1, n_6 + 1) \propto D_{w+2}$.

Lemma 5 $(n_1, n_2, n_3, n_4, n_5, n_6) \propto D_w \Longrightarrow (n_1 + 1, n_2 + 1, n_3 + 1, n_4, n_5, n_6) \propto D_{w+2}$ *(also,* $(n_1 + 1, n_2 + 1, n_3, n_4 + 1, n_5, n_6), (n_1 + 1, n_2 + 1, n_3, n_4, n_5 + 1, n_6)$*, and* $(n_1 + 1, n_2 + 1, n_3, n_4, n_5, n_6 + 1) \propto D_{w+2}$*).*

Lemma 6 $(n_1, n_2, n_3, n_4, n_5, n_6) \propto D_w \Longrightarrow (n_1 + 1, n_2, n_3 + 1, n_4 + 1, n_5, n_6) \propto D_{w+2}$ *(also,* $(n_1, n_2 + 1, n_3 + 1, n_4, n_5, n_6 + 1), (n_1 + 1, n_2, n_3, n_4, n_5 + 1, n_6 + 1)$*, and* $(n_1, n_2 + 1, n_3, n_4 + 1, n_5 + 1, n_6) \propto D_{w+2}$*).*

Lemmas 3–6 give constructive routings for most maximally routable RRVs in $\Omega_{w+2} - \Omega_w$ on D_{w+2} by extending the routings for D_w. However, there exist some RRVs in $\Omega_{w+2} - \Omega_w$ that can not be derived from Lemmas 3–6. We proceed to identify those RRVs in the following discussion.

If an RRV $\vec{n} = (n_1, n_2, n_3, n_4, n_5, n_6)$ is non-trivial for D_w, then any of the following RRVs (say $\vec{n'}$) $(n_1 + 2, n_2 + 2, n_3, n_4, n_5, n_6)$, $(n_1, n_2, n_3 + 1, n_4 + 1, n_5 + 1, n_6 + 1)$, $(n_1 + 1, n_2 + 1, n_3 + 1, n_4, n_5, n_6)$, $(n_1 + 1, n_2 + 1, n_3, n_4 + 1, n_5, n_6)$, $(n_1 + 1, n_2 + 1, n_3, n_4, n_5 + 1, n_6)$, $(n_1 + 1, n_2 + 1, n_3, n_4 + 1, n_5, n_6 + 1)$, $(n_1 + 1, n_2, n_3 + 1, n_4 + 1, n_5, n_6)$, $(n_1 + 1, n_2, n_3, n_4, n_5 + 1, n_6 + 1)$, $(n_1, n_2 + 1, n_3 + 1, n_4, n_5, n_6 + 1)$, and $(n_1, n_2 + 1, n_3, n_4 + 1, n_5 + 1, n_6)$ is non-trivial for D_{w+2}. We say an RRV such as $\vec{n'}$ to be *derivable* from Ω_w, since it is in Ω_{w+2} and can be obtained by performing some operation defined in Lemmas 3–6 on an RRV $\vec{n} \in \Omega_w$; it is *underivable*, otherwise. Based on Lemmas 3–6, we have the fact that $\vec{n} \propto D_w \Longrightarrow \vec{n'} \propto D_{w+2}$. Does there exist any underivable RRVs? As an example, $(0, 0, w+1, 1, w+1, 1) \in \Omega_{w+2}$ is underivable from Ω_w. (At first glance, it seems that the RRV can be obtained by performing the operation defined in Lemma 4 on $(0, 0, w, 0, w, 0)$; unfortunately, $(0, 0, w, 0, w, 0) \notin \Omega_w$.) We have the following lemmas.

Lemma 7 *Table 1 lists all underivable RRVs in $\Omega_{w+2} - \Omega_w$.*

Lemma 8 *The five classes of underivable RRVs listed in Table 1 are routable on diagonal switch matrix.*

Based on Lemmas 3–8, we have the following theorem.

Theorem 2 *The diagonal switch matrices are quasi-universal.*

Thus we have shown that the diagonal switch matrices are quasi-universal, and they thus have the maximum routing capacity among all switch matrices of the same size. It is easy to see that each diagonal switch matrix of size w contains $6w - 8$ ($6w - 9$) crossing switches if w is even (odd), and $8w - 12$ separating switches, $w > 1$. In particular, the numbers of switches are also the minimum requirement for a switch matrix to be quasi-universal.

Theorem 3 *No switch matrix with less than $6w - 8$ ($6w - 9$ if w is odd) crossing switches and $8w - 12$ separating switches can be quasi-universal, $w > 1$.*

Hence, the diagonal switch matrices are the "cheapest" quasi-universal switch matrices. Note that the number of switches required for a diagonal switch matrix is very small, compared to a fully populated switch matrix which has w^2 crossing switches and $2w^2 - 2w$ separating switches.

Class	Type of RRVs
Class 1	$(1, 0, w + 1, 1, w, 0)$
Class 2	$(2, 1, w, 1, w - 1, 0)$
Class 3	$(a, a, w - a + 2, 0, w - a + 1, 0)$, where $a = 0 \vee 1$
Class 4	$(a, a, w - a + 1, 1, w - a + 1, 1)$, where $a = 0 \vee 1$
Class 5	$(a, a - 1, w - a + 2, 0, w - a + 2, 0)$, where $a = 1 \vee 2$.

Table 1: The five classes of generic maximally routable RRVs on D_{w+2} that are underivable from Ω_w.

4.3 Routing-Capacity Analysis

Let D_w and U_w be a diagonal switch matrix and a universal switch module of size w, respectively. Let F_{D_w} be the *feasible set* for D_w; that is, $F_{D_w} = \{\vec{n} | \vec{n} \propto D_w\}$. F_{U_w} is similarly defined. We have the following theorem.

Theorem 4 $|F_{D_w}| = \left\lfloor \frac{5}{6!}(2w^6 + 24w^5 + 119w^4 + 312w^3 + 464w^2 + 96w + 144) \right\rfloor$.

Based on the above lemma, it is simple to verify the following theorem.

Theorem 5 *(Capacity ratio)*

(1) $|F_{D_w}|/|F_{U_w}|$ *is a strictly increasing function of w;*

(2) $\lim_{w \to \infty} |F_{D_w}|/|F_{U_w}| = 1$.

Therefore, the routing capacity of a diagonal switch matrix converges to that of a universal switch module of the same size.

5 Graph Modeling for Detailed Routing

In the previous section, we showed theoretically that the diagonal switch matrices have high routing capacities. To explore the effects of switch-matrix architectures on chip-level routing, we shall test the area performance of a router on an FPD chip using benchmark circuits. To develop a router for experimentation, we may model an FPD as a graph

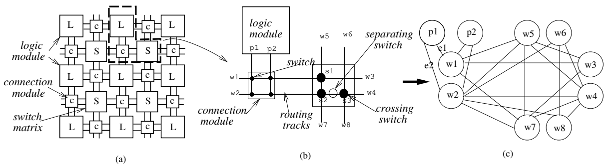

Figure 6: The graph modeling. (a) A symmetrical-array-based FPGA architecture. (b) Switches in the connection module and the switch matrix. (c) The graph topology.

and apply the graph-search technique to FPD routing. In this section, we use a symmetrical-array-based FPGA as an example to demonstrate the graph modeling.

Given an FPGA architecture, we use a vertex to represent a wire segment or a logic-module pin, and an edge to represent a connection that can be established by programming a switch or by using a track in the switch matrix. See Figure 6 for an illustration. Figure 6(b) shows a logic module with two pins on one side. We introduce two vertices p_1 and p_2 for the two pins shown in the figure. There are two horizontal (vertical) routing tracks partitioned into four wire segments (for the portion considered here), two on each of the left and right (the top and bottom) sides of the switch matrix. We introduce a vertex for each wire segment ($i_1, i_2, \ldots, i_8$ in Figure 6(c)) and an edge between the two vertices associated with each pair of wire segments abutted on the switch matrix (edges $(i_1, i_3), (i_6, i_8)$, etc). If there is a crossing switch connected to any pin or wire segment, then introduce an edge between the two corresponding vertices. (Note that the switches in the connection module can be viewed as crossing switches. A connection module is, in fact, a switch module with no separating switches.) For instance, since pin p_1 can connect to wire segment i_1 (i_2), an edge e_1 (e_2) between vertices p_1 and i_1 (i_2) is created. For each crossing switch, we create additional four edges for its incident wire segments (thus a clique for the four vertices associated with those segments is formed). The graph modeling is thus done. In addition to the graph modeling, however, we need to use two data structures to cope with the problems of connection conflicts. The conflicts arise in two forms:

- *Mutually exclusive crossing switches:* Two crossing switches with no separating switch between them are mutually exclusive. For example, in Figure 6, crossing switches s_1 and s_2 can not be used for different connections at the same time since there is no separating switch between them. In this case, we say that s_1 and s_2 are in the same *exclusion set*. Algorithm 7 provides a method to find the exclusion sets for a switch matrix.
- *Mutually exclusive connections:* Two connections are mutually exclusive if they do not belong to the same net and are incident on the same crossing switch. For example, in Figure 6, edges (i_1, i_5) and (i_3, i_7) can not be used for different connections at the same time since the connections are mutually exclusive. It is easy to derive an algorithm similar to Algorithm 7 to identify all mutually inclusive connections.

With the data structures, we can incorporate the detection for illegal connections and exclusion sets into a router.

Based on the graph modeling, we may formulate the routing problem as finding a set of disjoint trees (a forest), one

Algorithm: Mutually_Exclusive_Set(M)
Input: M—A switch-matrix configuration.
Output: EP—Set of mutually exclusive switch pairs in M.

```
EP ← ∅;
for each crossing switch pair (s_i, s_j) on M
    do if there is no separating switch between s_i and s_j
        then EP ← EP ∪ {(s_i, s_j)};
Output EP.
```

Figure 7: Algorithm for finding the set of mutually exclusive switch pairs on a switch matrix.

tree for a net and each tree connecting all terminals of a net. Any graph search-based algorithm such as maze router can be used for detailed routing.

6 Experimental Results

To explore the effects of switch-matrix architectures on routing, we implemented a maze router based on the graph modeling mentioned in the preceding section in the C language and ran on a SUN Ultra workstation. We tested the area performance of the router based on the CGE [14] and SEGA [12] benchmark circuits. A logic-module pin was connected to any of the w tracks in the adjacent routing channel. The switch-matrix architectures used were the diagonal switch matrices, randomly generated switch matrices with the same numbers of switches as those in the diagonal switch matrices, and the switch matrices designed by [19].

The quality of a switch matrix was evaluated by the area performance of the detailed router. Table 2 shows the results. For the results listed in this table, we determined the minimum number of tracks w required for 100% routing completion for each circuit, using the three kinds of switch matrices. Because net ordering often affects the performance of a maze router, we routed the benchmark circuits by using the following three net-ordering schemes to avoid possible biases: (1) net order as given in the original benchmark circuits, (2) shortest net first (non-decreasing order of net lengths), and (3) longest net first (non-increasing order of net lengths). Also, since our main goal is to make fair comparisons for various switch-matrix architectures, no rip-and-reroute phase was incorporated in the maze router (optimization might bias the comparison). The running times ranged from 3 sec for the smallest circuit (`9symml`) to 160 sec for the largest one (`Z03`). Our results show that, among the three kinds of switch matrices, the diagonal switch matrices usually needed the minimum w's for 100% routing completion, no matter what order was used. The results show that our diagonal switch matrices can improve the routability at the chip level.

It should be noted that the design in [19] is based on a different switch-matrix routing model from ours—in [19], only one crossing or separating switch can be used for routing a connection on *a* switch matrix, and at most $2w$ separating switches can be placed on a switch matrix, whereas ours allows multi-switch routing and does not have the upper-bound constraint, $2w$; therefore, it is impossible to make completely fair comparison with [19]. (This is why we also compared our designs with those randomly generated switch matrices.)

We also performed experiments to explore the effects of net density on the area performance of switch matrices. We randomly generated connections on a 15×15 (number of logic modules) FPGA. For this purpose, we assume that the number of pins on each logic module is unlimited (so that we could test denser circuits). As shown in Table 3 and Figure 8, the denser the circuit, the better the diagonal switch matrices than the randomly generated switch matrices. This phenomenon reveals the facts that the routability of a single switch matrix plays a more important role when (1) the connection density on a chip gets denser and (2) the switch matrices become larger. Notice that denser applications and larger chips are trends of the commercial applications and products. Therefore, we expect that the switch-matrix architectures will have even greater impact on FPD chip routability than they are now.

Number of connections	#Tracks needed for detailed routing Diagonal	Random
200	12	12
400	16	17
600	22	24
800	31	35
1000	34	38
1200	38	43
1400	40	45

Table 3: Comparison of the area performance by using the diagonal switch matrices and randomly generated switch matrices for various connection densities, based on a 15×15 FPGA.

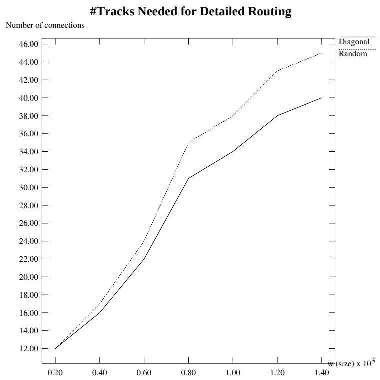

Figure 8: Comparison of the area performance by using the diagonal switch matrices and randomly generated switch matrices for different numbers of connections on a 15×15 FPGA.

References

[1] Actel Corp., *FPGA Data Book and Design Guide*, 1996.

[2] Altera Corp., *FLEX 10K Handbook*, 1996.

[3] Aptix Inc., *FPIC AX1024D*, Preliminary Data Sheet, Aug., 1992.

[4] AT&T Microelectronics, *AT&T Field-Programmable Gate Arrays Data Book*, Apr. 1995.

[5] Bhat, N. and Hill, D., "Routable technology mapping for LUT FPGAs," *in Proceedings of the Internation Conference on Computer Design, VLSI in Computers and Processors*, pp. 95–98, 1992.

[6] S. D. Brown, J. Rose, and Z. G. Vranesic, "A stochastic model to predict the routability of field-programmable gate arrays," *IEEE Trans. Computer-Aided Design*, vol. 12, no. 12, pp. 1827–1838, Dec. 1993.

[7] Y.-W. Chang, D. F. Wong, and C. K. Wong, "Design and analysis of FPGA/FPIC switch modules," in *Proc. IEEE Int. Conf. Computer Design*, pp. 394–401, Austin, TX, Oct. 1995.

[8] Y.-W. Chang, D. F. Wong, and C. K. Wong, "Universal switch modules for FPGA design," *ACM Trans. Design Automation of Electronic Systems*, vol. 1, no. 1, pp. 80–101 , Jan. 1996.

[9] A. El Gamal, *et al.*, "An architecture for electrically configurable gate arrays," *IEEE J. Solid-State Circuits*, vol. 24, no. 2, pp. 394–398, Apr. 1989.

[10] H.C. Hsieh, *et al.*, "Third-generation architecture boosts speed and density of field-programmable gate arrays," in *Proc. IEEE Custom Integrated Circuits Conf.*, pp. 31.2.1–31.2.7, May 1990.

[11] C. Y. Lee, "An algorithm for path connections and its applications," *IRE Trans. Electornis Computer.*, vol. EC-10, pp. 346–365, Sept. 1961.

[12] G. G. Lemienx and S. D. Brown, "A detailed routing algorithm for allocating wire segments in field-programmable gate arrays," in *Proc. ACM/SIGDA Physical Design Workshop*, pp. 215–216, Lake Arrowhead, CA, 1993.

[13] G. G. Lemienx, S. D. Brown and D. Vranesic, "On two-step routing for FPGAs," in *Proc. ACM International Symposium on Physical Design*, pp. 60–66, Napa Valley, April 1997.

[14] J. Rose and S. Brown, "Flexibility of interconnection structures for field-programmable gate arrays," *IEEE J. Solid State Circuits*, vol. 26, no.3, pp. 277–282, Mar. 1991.

[15] Y. Sun, T. -C, Wang, C. K. Wong, and C. L. Liu, "Routing for symmetric FPGAs and FPICs," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, pp. 486–490, Santa Clara, Nov. 1993.

[16] Trimberger, S. and Chene, M., "Placement-based partitioning for lookup-table-based FPGA," in *Proc. International Conference on Computer Design, VLSI in Computers and Processors*, pp. 91–94, 1992.

[17] Y.-L. Wu, S. Tsukiyama, and M. Marek-Sadowska, "Graph Based Analysis of 2-D FPGA routing," *IEEE Trans. Computer-Aided Design*, vol. 15, no. 1, pp. 33–44, Jan. 1996.

[18] Xilinx Inc., *The Programmable Logic Data Book*, 1996.

[19] K. Zhu, D.F. Wong and Y.-W. Chang, "Switch module design with application to two-dimensional segmentation design," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, pp. 481–486, Santa Clara, Nov. 1993.

	Number of tracks needed for detailed-routing completion								
	Original order			Non-decreasing length order			Non-increasing length order		
Circuit	Q-USM	Random	[19]*	Q-USM	Random	[19]*	Q-USM	Random	[19]*
BUSC	9	9	10	9	9	9	9	9	9
DMA	10	11	10	10	10	10	12	13	12
BNRE	11	12	12	9	9	10	13	14	13
DFSM	11	12	11	10	11	10	12	13	12
Z03	15	15	15	12	13	12	15	16	16
9symml	10	11	10	9	11	10	9	10	10
alu2	11	11	11	9	10	9	11	11	10
alu4	12	13	12	11	12	12	13	13	13
apex7	9	10	10	7	8	8	9	10	10
example2	13	14	14	10	11	10	13	14	13
k2	14	14	14	12	12	12	15	18	14
term1	9	10	9	8	8	8	9	10	9
too_large	11	11	11	10	11	10	11	11	12
vda	13	13	13	11	12	10	14	14	13
Total	158	166	162	137	147	140	165	176	166

Table 2: Number of tracks needed for detailed-routing completion for the CGE (top 5) and SEGA (bottom 9) benchmark circuits by using the three schemes for net order: (1) original net order as given in the benchmark circuits, (2) shortest net first (non-decreasing order of net lengths), and (3) longest net first (non-increasing order of net lengths). *Note that the design in [19] is based on a different switch-matrix routing model from ours—in [19], only one crossing or separating switch can be used for routing a connection on *a* switch matrix and at most $2w$ separating switches can be placed on a switch matrix, whereas ours allows multi-switch routing and does not have the upper-bound constraint, $2w$; therefore, it is impossible to make completely fair comparison with [19].