Optimizing Effective Interconnect Capacitance for FPGA Power Reduction

Safeen Huda, Jason Anderson
Dept. of ECE, University of Toronto
Toronto, ON, Canada

Hirotaka Tamura
Fujitsu Laboratories Limited
Kawasaki, Japan

ABSTRACT

We propose a technique to reduce the effective parasitic capacitance of interconnect routing conductors in a bid to simultaneously reduce power consumption and improve delay. The parasitic capacitance reduction is achieved by ensuring routing conductors adjacent to those used by timing critical or high activity nets are left floating - disconnected from either V_{DD} or *GND*. In doing so, the effective coupling capacitance between the conductors is reduced, because the original coupling capacitance between the conductors is placed *in series* with other capacitances in the circuit (series combinations of capacitors correspond to lower effective capacitance). To ensure unused conductors can be allowed to float requires the use of tri-state routing buffers, and to that end, we also propose low-cost tri-state buffer circuitry. We also introduce CAD techniques to maximize the likelihood that unused routing conductors are made to be adjacent to those used by nets with high activity or low slack, improving both power and speed. Results show that interconnect dynamic power reductions of up to $\sim$15.5% are expected to be achieved with a critical path degradation of $\sim$1%, and a total area overhead of $\sim$2.1%.

1. INTRODUCTION

Process scaling has served as the linchpin of the unprecedented growth the semiconductor industry has seen over the past 50 years. Through process scaling, the performance, energy efficiency, and density of a digital system can all be increased, and these trends combine to offer significant increases in the total computational power of chips with each passing process node. However, as the industry continues to venture into deep submicron territory, several challenges have emerged which are beginning to degrade the expected returns from process scaling. One of the most significant bottlenecks has been the poor scalability of interconnect power; for example, [1] and [2] show that interconnect is becoming an ever more dominant factor in the performance and power dissipation of a digital system. As such, the optimization of interconnect is becoming an ever more critical aspect of digital system design.

While interconnect optimization has been explored extensively in the ASIC domain [3, 4], there is little published literature on the optimization of interconnect in FPGAs. This is because in ASICs, the width and spacing of wires can be arbitrarily chosen and as such they can be optimized to meet a target delay and/or power consumption on a net-by-net basis. In contrast, FPGAs have fixed, prefabricated interconnect structures whose geometry (width and spacing to other interconnect) cannot be altered to optimize the power and/or performance of a specific net.

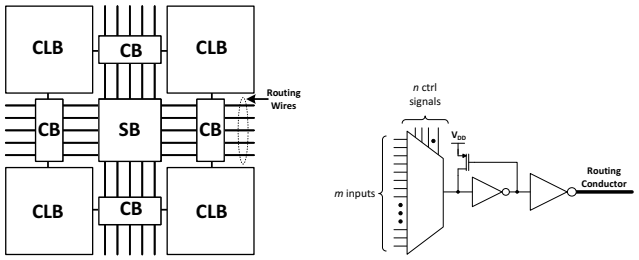
In this work, we propose a technique to reduce the total parasitic capacitance of FPGA interconnect structures. We first note that the majority of the capacitance in the interconnect lies in the coupling capacitance, C_C, between adjacent routing conductors, and moreover, the coupling capacitance is expected to become an increasingly dominant component of interconnect capacitance, based on ITRS projections [5]. We then note that if a routing conductor is left in a floating state, the adjacent routing conductors will see reduced capacitive loading. This is because (as we will demonstrate) the effective capacitance seen by an adjacent conductor is that of the series combination of the coupling capacitance between the two conductors, and the total capacitance to ground of the conductor left in a floating state. Since the series combination of two capacitors results in an equivalent capacitance which is *less* than the capacitance of either of the two original capacitors, it follows that the total capacitance seen between a conductor and an adjacent floating conductor is less than in the case where the adjacent conductor is tied to a rail (V_{DD} or *GND*).

To implement this desired capacitance reduction technique, we propose a novel lightweight buffer topology which allows for tri-state operation, while requiring minimal area overhead. We further propose CAD techniques to ensure that nets which would benefit the most from reduced parasitic capacitance (i.e. timing critical nets or nets with high activity factor) have unused adjacent routing conductors, thus allowing our optimization to be applied. Results show that interconnect dynamic power reductions of up to $\sim$15.5% can be achieved as C_C approaches a value three times as large as C_P (the plate capacitance – the capacitance between a wire and adjacent metal layers/substrate). The power savings are achieved at a critical path degradation of $\sim$1% (on average), and a total area overhead of $\sim$2.1%.

The remainder of this paper is organized as follows: Section 2 reviews FPGA interconnect hardware, describes interconnect scaling trends, and general approaches to the physical optimization of interconnect. The proposed effective parasitic capacitance reduction technique and necessary circuits are presented in Section 3 and Section 4, respectively. The interconnect architecture model and CAD tool modifications needed to maximize the available gains of our proposed technique are described in Sections 5 and 6, respectively. The experimental study is described in Section 7. Finally, Section 8 concludes the paper.

FPGA'14, February 26–28, 2014, Monterey, CA, USA.

http://dx.doi.org/10.1145/2554688.2554788.



(a) Top level organization of CLBs, CBs, SBs, and routing wires.

(b) Circuit-level structure of routing switch.

Figure 1: Conventional FPGA circuit structures.

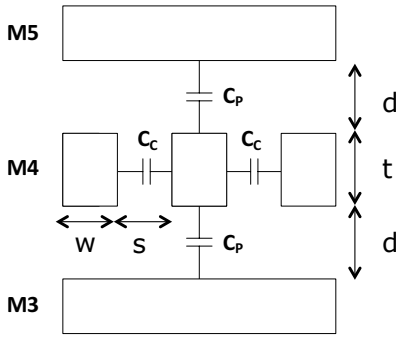


Figure 2: Interconnect capacitance components.

2. BACKGROUND

2.1 FPGA Routing Circuitry

Electrical connectivity between logic blocks (CLBs) in an FPGA is facilitated by the *programmable routing network*. The routing network is comprised of fixed-length routing wires, Connection Boxes (CBs) which connect CLBs to routing wires, and Switch Boxes (SBs) which are used to stitch together routing wires to form paths. A simple organization of routing wires is shown in Fig. 1(a). The switch boxes and connection boxes are simple pass-transistor based multiplexors, with an output driver circuit to buffer and recondition the output signal. Routing switch circuitry is shown in Fig. 1(b). The pull-up PMOS on the multiplexer output is present because the mux is typically implemented using NMOS transistors (which are poor at passing logic "1").

2.2 Interconnect Power and Capacitance Scaling Trends

Due to the large capacitances of the routing wires in a modern FPGA, the power dissipated in the routing network is a dominant component of the total power dissipated, as much as 62% [6]. As such, any effort to reduce the power consumption in the routing network will potentially allow for an appreciable reduction in overall power. The significant power consumption in the routing network is attributed to the large capacitance of routing conductors. We estimate that the total load capacitance of a length-4 wire in a commercial FPGA fabricated in a 65nm CMOS process is near 200fF as follows: Based on the tile area data in [7], we estimated a single tile length in each dimension as 170μm for an FPGA in 65nm CMOS, which implies a wire capacitance of 170fF for a length-4 wire, assuming a wire capacitance per length of 0.25fF/μm. We anticipate that the additional parasitic capacitance of a length-4 wire (resulting from the diffusion capacitance of the many multiplexors connected to a length-4 wire) will push the total load capacitance close to 200fF. Fig. 2 shows the two primary sources of capacitance in a routing conductor: C_P, the plate capacitance, which is the capacitance between a routing conductor and conductors in adjacent layers and/or the substrate, and C_C, the capacitance between a routing conductor and adjacent routing conductors within the same metal layer. Due to the evolution of the parameters related to wire geometries through process scaling, minimum width wires in intermediate metal layers have considerably greater height than width, as depicted in the figure. This leads to a growing disparity in the capacitances of C_C and C_P. Using ITRS projections [5] and simple models for wire capacitance, it can be estimated that C_C will scale to be approximately twice as large as C_P, although prior work using more sophisticated capacitance models has estimated that C_C is approximately 85% of the total interconnect capacitance [8] - implying that $C_C/C_P \approx 5.7$ - in a 90nm process. Moreover, in FPGAs specifically, it is believed that C_C can be up to three times larger than C_P, depending on wire width and spacing [9]. Thus, it is widely believed that C_C is the dominant component of the total capacitance of interconnect, and as such, any technique to reduce C_C will lead to favourable reductions in total wire capacitance.

2.3 Wire Planning in ASICs and FPGAs

Due to the increasing sensitivity of key system-level metrics (such as speed and power) on interconnect parameters, much work has been done on interconnect optimization in the ASIC domain [10],[11]. Broadly speaking, the interconnect optimization problem serves to optimize a chip's speed and/or power consumption through judicious selection of wire width and spacing, under constraints of total wiring area. For example, a timing critical net would be routed with an intermediate wire width (to reduce interconnect resistance and thus reduce interconnect delay) and increased wire spacing to reduce wire capacitance (and thus further reduce delay). On the other hand, a high activity signal would be routed with both minimum width and maximum spacing to ensure minimum routing conductor capacitance, thus minimizing dynamic power consumption.

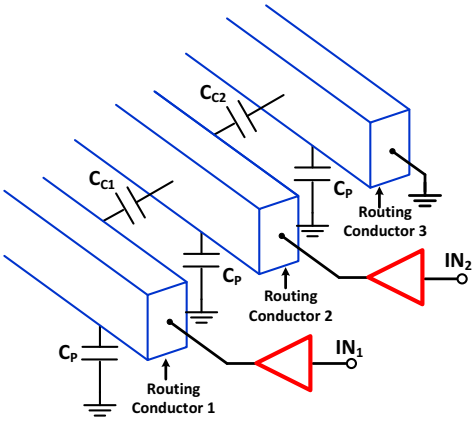
Previous work has investigated the optimization of interconnect in FPGAs [12], however it only considered the optimization of wire spacing and width of the prefabricated routing network. These optimizations provided some control over the wire width and spacing of conductors used by a particular net. For example, if a net were timing critical, the router would endeavor to route the net using the portion of the interconnect network which had favorable wire width/spacing. However, depending on placement and wiring congestion, such low-capacitance conductors may not be readily accessible, and moreover, the prior work considered only timing optimization, not power.

3. PROPOSED PARASITIC CAPACITANCE REDUCTION TECHNIQUE

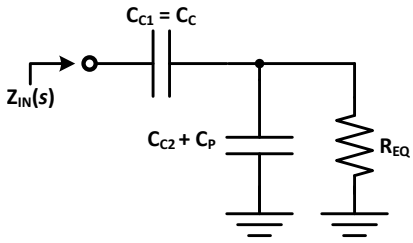
The proposed parasitic capacitance reduction technique is inspired by the observation that two capacitors connected in series results in a total capacitance that is smaller than either of the two capacitors. More specifically, if two capacitors, C_1 and C_2 are connected in series, the total resulting capacitance is:

$$C_{eq} = \frac{C_1 \cdot C_2}{C_1 + C_2} \quad (1)$$

We are therefore inspired to reduce the effective parasitic interconnect capacitance of a routing conductor by making use of this property. We again note that the coupling capacitance between adjacent routing conductors is becoming increasingly dominant, and



(a) Interconnect drivers and their respective loads.



(b) Equivalent circuit representing routing conductors 2 and 3, from the perspective of routing conductor 1.

Figure 3: Effective impedance of coupled routing conductors.

so it would be advantageous if we could reduce the coupling capacitance by combining this capacitance with a second capacitance in series. We devised an approach whereby this can be achieved under certain circumstances.

Fig. 3(a) shows three routing conductors, with their respective driver circuits. Routing conductor 1 is coupled with routing conductor 2 through coupling capacitor C_{C1} and similarly, routing conductor 2 is coupled with routing conductor 3 through C_{C2}. From the perspective of routing conductor 1, routing conductor 2 and coupling capacitors C_{C1} and C_{C2} together form the equivalent, approximated circuit shown in Fig. 3(b). In the figure, R_{eq} is the effective output resistance of the driver driving routing conductor 2, and it is assumed (pessimistically, as will later become apparent) that the effective driver resistance for the buffer driving routing conductor 3 is near 0 (the conductor is grounded). The input impedance of this circuit has the following s-domain transfer function:

$$Z_{in}(s) = \frac{R_{eq}(2 \cdot C_C + C_P)s + 1}{C_C s[R_{eq}(C_c + C_P)s + 1]} \quad (2)$$

where it is assumed that $C_{C1} = C_{C2} = C_C$. We now note that if $R_{eq} >> T_{PW\_AVG}/2\pi(C_C + C_P)$, where T_{PW_AVG} represents the average signal pulse width, $Z_{in}(s) \approx (2 \cdot C_C + C_P)/(C_C(C_C + C_P))$. This implies that if the driver impedance can be set to be very large, the effective impedance seen from routing conductor 1 looking towards routing conductor 2 is that of capacitor C_{C1} in *series* with the parallel combination of capacitors C_{C2} and C_P, which means that the effective parasitic loading on routing conductor 1 can therefore be *reduced*. Note however, that setting the driver resistance to be large enough such that such reductions in effective parasitic capacitance are achievable is not a practical solution, as this will make the buffer excessively slow (since the buffer delay is, to first order, equal to the buffer resistance multiplied by the total output loading). Note however that a buffer with a reasonably small effective output resistance can be made to have a much larger output resistance if that buffer can be put into a tri-state mode. Putting a buffer into tri-state mode requires disconnecting all paths to either V_{DD} or *GND*, which ensures that there are no low impedance paths to either rail.

Therefore, we have shown that the parasitic capacitance arising from the coupling between two adjacent conductors can be reduced when the driver of the neighbouring conductor is put into tri-state mode. In this work, we assume that a routing conductor can in general be put into a tri-state mode, only if it is unused. As such, parasitic capacitance reduction can be observed only when an active routing conductor is adjacent to one or more unused routing conductors. The remaining sections will detail the circuitry to enable tri-state mode capable buffers, and CAD tool support to encourage routing conductors to remain unused if they are adjacent to a used routing conductor.

Returning now to routing conductor 3 in Fig. 3(a), the analysis above assumed the conductor was grounded. The reason this is a pessimistic assumption as far as capacitance reduction is concerned is as follows: if instead conductor 3 were tri-stated, as may be the case in a real FPGA, the consequence would be that capacitor C_{C2} would be in *series* with the plate capacitance of conductor 3 (C_P). This would mean that the capacitance of the capacitor labeled $C_{C2} + C_P$ in Fig. 3(b) would in fact be $C_P + C_{C2}[C_P/(C_{C2} + C_P)]$ which is less than $C_{C2} + C_P$. This would provide a further reduction in coupling capacitance between conductor 1 and conductor 2 (for the case of conductor 2 being tri-stated).

4. TRI-STATE BUFFERS

A conventional tri-stateable buffer is shown in Fig. 4. In this buffer, a header transistor is added to cut off all paths from V_{DD} to the input and output stages of the buffer. As can be seen in Fig. 4, when signal TS is high, transistor M5 is shut off, thus disconnecting the output from V_{DD}. Given that an unused routing switch will have its input held at V_{DD}, this leads to the gate of M6 being pulled to *GND*, which ensures that the output is also disconnected from *GND*; thus, the output is completely disconnected from either rail when an unused switch is put into tri-state mode. Note that transistor M3 cannot be disconnected from V_{DD}, as this ensures that the input of the buffer can be held at V_{DD}. This approach to designing tri-stateable buffers however is costly because of the transistor stacking in the pull-up paths. To maintain the same buffer delay

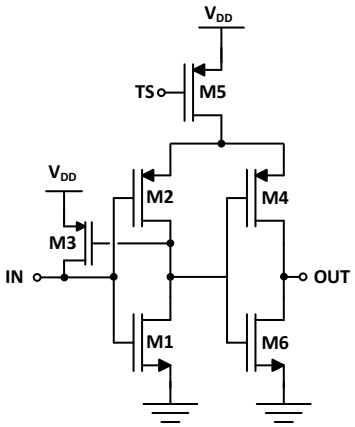


Figure 4: Conventional tri-state buffer.

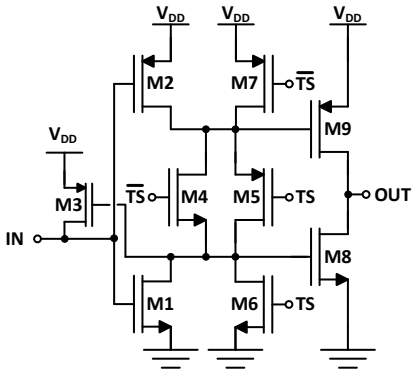


Figure 5: Alternative conventional tri-state buffer.

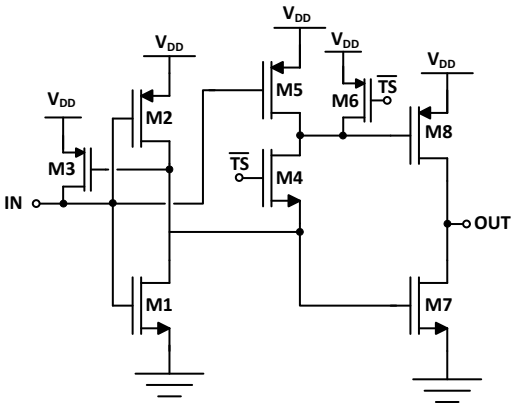


Figure 6: Proposed tri-state buffer.

as that of a conventional buffer, the transistor widths need to be at least doubled. In reality, the transistor width must be increased by a factor greater than two to compensate for speed degradation due to the body effect. Therefore, the traditional approach to designing tri-stateable buffers results in high area cost, since it not only requires an increased number of transistors, but it also necessitates a significant increase in the widths of the PMOS transistors in the input and output stages of the buffer.

An alternative conventional tri-stateable buffer topology is shown in Fig. 5 - this topology consists of a split inverter (formed by transistors M1 and M2), separated by a transmission gate (transistors M4 and M5) which together form the input stage, and a conventional output stage consisting of transistors M8 and M9. Note that in this topology, there is no transistor stacking in the output stage. As such, this topology does not require increased transistor sizing of the transistors in the output stage (which dominate the area of the buffer), thus allowing for a more area efficient design. The buffer works as follows: when TS is low, the transmission gate consisting of transistors M4 and M5 is on which effectively shorts the gate of M9 to the gate of M8. Thus, in this mode M1 and M2 are connected to form a single inverter which directly drives the output stage. When TS is high however, M4 and M5 are both off, which leads to the connection between the gates of M8 and M9 being severed; M6 and M7 are both turned on pulling the gates of M8 and M9 to GND and V_{DD}, respectively. Thus, in this mode both M8 and M9 are off, leaving the output tri-stated. This topology therefore uses a clever arrangement of transistors in the input stage of the buffer to ensure that tri-state mode is achieved without transistor stacking in the output stage, which results in area savings compared to the design in Fig. 4.

In this work, we propose to further optimize the buffer topology shown in Fig. 5 for use in FPGAs. Referring to Fig. 5, we can note some potential optimizations. We first observe that in FPGAs, an unused routing buffer will always have its input set to V_{DD}; this means that in tri-state mode, M1 is always guaranteed to be on, which makes transistor M6 unnecessary. Furthermore, because the input is guaranteed to be set to V_{DD}, the drain of M2 can safely be connected directly to the gate of M8; thus M1 and M2 would form a conventional inverter driving the gate of M8. This is advantageous because previously the path to V_{DD} from the gate of M8 was through two stacked PMOS transistors: M2 and M5. By eliminating this stack of transistors, we are able to reduce the sizes of the PMOS transistors in the input stage, which results in appreciable area reduction since the area of PMOS transistors dominates the total area of the input stage. Finally, transistor M5 can be repositioned to have its source connected to V_{DD} and its drain connected to the gate of M9 - again this eliminates any stacking in the pull-up path from the gate of M9 to V_{DD}, and also marginally reduces the parasitic capacitance connected to the gate of M8 (we have eliminated the diffusion capacitance arising from the drain of M5 at this node). Our optimized tri-stateable buffer topology is shown in Fig. 6, and together with the set of optimizations we discussed above, aims to offer a very area efficient tri-stateable buffer topology.

A detailed description of our proposed buffer is as follows: When the buffer is used (i.e. it is not tri-stated), since TS is low, transistor $M6$ is off, while $M4$ is on. During a rising transition at the input, $M1$ turns on, and the gates of $M8$ and $M7$ are both pulled to ground, which turns $M8$ on, $M7$ off, and allows the output to be pulled to V_{DD}. During a falling transition at the input, $M2$ and $M5$ both turn on, which results in $M8$ turning off, $M7$ turning on, thus causing the output to be pulled to GND. Observe that $M5$ pulls the gate of $M8$ to rail V_{DD} when the input is low.

When operating in tri-state mode, TS is high and therefore, $M6$ is on, $M4$ is off, which keeps the gate of $M8$ pulled to V_{DD} (thus keeping it turned off). Since an unused buffer has its input tied to V_{DD} (the input stage of the buffer acts as a latch when the input is at V_{DD}, since $M3$ forms a feedback loop allowing the input to hold its state at V_{DD}), $M1$ turns on, thus pulling the gate of $M7$ to GND and turning it off.

There is one issue regarding the tri-state buffer of Fig. 6 that warrants additional discussion, which is regarding the feasibility of "floating" unused conductors. This practice is acceptable in the FPGA context because such unused conductors feed downstream switches and pins via input NMOS-based multiplexors (see Fig. 1(b)). Naturally, the select inputs of such downstream multiplexors would not be configured to pass a signal from an incoming unused conductor. Note that the ability to tri-state wires is unique to the inherent interconnect architecture of FPGAs; tri-stating signals in general CMOS logic is not acceptable, owing to the potential for significant short-circuit current if a CMOS gate's input floats to a voltage mid-way between the two rails.

4.1 Buffer Comparison

To assess the merits of the proposed tri-state buffer over the two conventional topologies, we sized the buffers to allow for a target delay of approximately 185 ps - representing a $\sim$10% delay overhead over a conventional buffer (not tri-stateable). Comparisons of the area overhead (in minimum transistor widths) and tri-state leakage power reduction (relative to a conventional, non-tri-stateable buffer) are shown in Table 1.

The comparison shows that the conventional tri-state buffer topology (Fig. 4) offers superior leakage power reductions when oper-

Buffer Topology	Area Overhead [Min W. Xtors]	Tri-state Mode Leakage Reduction [%]
Conventional (Fig. 4)	99	45
Alternative Conventional (Fig. 5)	6.5	11
Proposed (Fig. 6)	3	25.4

Table 1: Comparison of area and tri-state leakage of buffer topologies.

ating in a tri-state mode. This is expected since the conventional topology employs transistor stacking, which results in reduced V_{SG} of the PMOS transistors, thus resulting in reduced leakage. The alternative conventional tri-state buffer (Fig. 5) offers dramatic reductions in area overhead, owing to the fact that there are no stacked transistors in the output stage of the buffer (recall that the transistors in the output stage of the buffer are the dominant component of the total buffer area). Thus, the area overhead is reduced from 99 minimum width transistors to ~6.5 minimum width transistors. However, the lack of transistor stacking results in poor tri-state mode leakage power, as we only observed an 11% reduction in leakage power compared to a conventional buffer which is not tri-stateable. The proposed buffer offers further reductions in area overhead compared to the alternative conventional tri-state buffer, as the area overhead is reduced from ~6.5 minimum width transistors to 3 minimum width transistors. The proposed buffer topology also shows decreased tri-state mode leakage power compared to the alternative conventional design. This is because the proposed topology is able to employ narrower transistors in the input stage, which results in reduced leakage current. However, the proposed buffer is still unable to match the low leakage power of the conventional tri-state buffer; again, this is because the proposed topology does not employ stacking, and as such, cannot offer the same levels of leakage reduction. Thus, the combined low cost/low power of the proposed topology makes it an attractive option to implement tri-state buffers in an FPGA. Note however, our proposed capacitance reduction technique does not rely on any specific topology, indeed a conventional tri-state buffer topology may be employed, albeit at a much larger area penalty.

5. INTERCONNECT MODELLING

In this work, we assume that the layout of the the routing conductors forming the interconnect network matches that of the standard VPR-style staggered interconnect model; Fig. 7 depicts the assumed organization of the routing conductors comprising a routing channel.

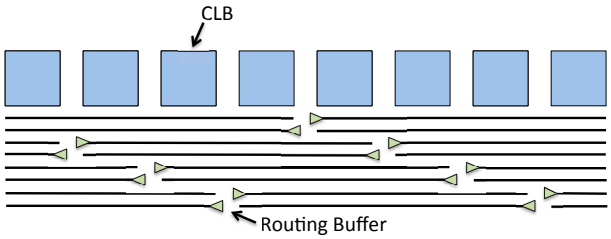


Figure 7: Assumed layout of routing conductors within a channel.

Note in this assumed organization of routing conductors, adjacent conductors do not necessarily have complete overlap with one another. Rather, because the start points of interconnect are staggered, some adjacent conductors will only partially overlap with one another. Fig. 8 shows a detailed view of the coupling capacitances between adjacent conductors. Note that because of the staggered nature of the routing layout considered in this work, all tracks (except for those at the boundaries of a channel) will have exactly three neighbours. In the figure, track (i, j) is coupled with routing tracks $(i-1, j)$, $(i-1, j+1)$, and $(i+1, j)$. Thus track (i, j) has three neighbours which for this paper we number as follows: track $(i-1, j)$ is neighbour 1, track $(i-1, j+1)$ is neighbour 2, and track $(i+1, j)$ is neighbour 3. We note that tracks (i, j) and $(i+1, j)$ completely overlap with one another, and model the coupling between these two tracks with a capacitor with capacitance C_C. As previously mentioned, some tracks do not overlap completely with adjacent tracks, and this is the case with tracks $(i-1, j)$ and $(i-1, j+1)$, which have a 75% and 25% overlap with track (i, j), respectively. As such, we model the coupling between tracks $(i-1, j)$ and (i, j) and between tracks $(i-1, j+1)$ and (i, j) with capacitances of $0.75C_C$ and $0.25C_C$, respectively. Note that this example showed the various overlaps with neighbouring tracks when i is an odd number. When i is an even number, track (i, j) will have complete overlap with track $(i-1, j)$, 25% overlap with track $(i+1, j-1)$ and 75% overlap with track $(i+1, j)$.

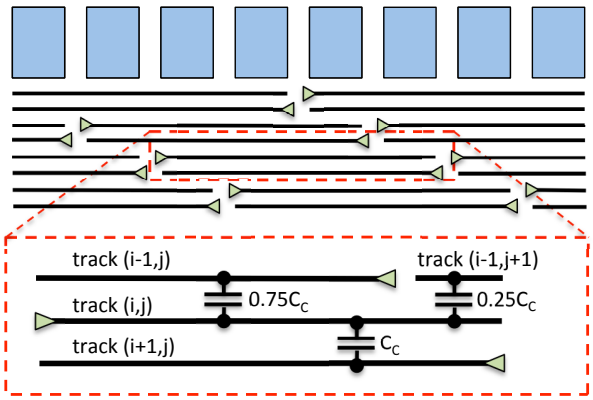


Figure 8: Detailed view of routing conductor layout.

6. TOOL SUPPORT

One of the premises of the proposed capacitance reduction technique is that power and/or timing critical switches will have adjacent tracks which are *unused* and thus can be put into tri-state mode. Therefore, special effort must be made during the physical implementation of a circuit to ensure that such situations arise so as to optimize both power and timing. In order to ensure that power and timing critical nets have unused adjacent tracks, we have made modifications to a conventional FPGA router. The router is modified to encourage power and timing critical signals to be routed with unused adjacent tracks.

The modifications to the router in this work bears some similarity to a previously proposed crosstalk-aware router for FPGAs [13], in that signals are routed such that they have unused adjacent tracks (if they have high activity or are timing critical). However, the difference between this work and prior work is that here we proposed to leave adjacent conductors unused to reduce the effective capacitance of used routing conductors, thereby reducing power and improving speed. This is in contrast to previous work, where adjacent conductors were left unused (and were grounded, not tri-stated as is the case in our work) to reduce coupling between active routing conductors, thereby mitigating the delay penalty due to crosstalk.

The VPR router incorporates the PathFinder algorithm [14]. PathFinder routes individual driver/load connections one at a time. A cost function is used to find a low-cost path through the routing fabric from a driver to a load. The cost function incorporates metrics of timing performance and *routability* (i.e. the ability to legally route all of the connections in the circuit) in order to yield a routed circuit with a favorable quality of results. The cost function implemented in VPR is:

$$\begin{aligned} Cost_n &= (1-Crit_i)\cdot cong\_cost_n \\ &\quad + Crit_i \cdot delay\_cost_n \end{aligned} \tag{3}$$

where n is the routing resource (e.g. wire segment) being considered for addition to a partially-completed route, i is the driver/load connection being routed, $Crit_i$ is the timing criticality of the connection (equal to 1 for connections on the critical path, and decreasing to 0 as the slack of the connection increases), $cong\_cost_n$ is the congestion cost of routing resource n which gives an indication of the *demand* for the routing resource among nets, while $delay\_cost_n$ is the delay of routing resource n.

The rationale for cost function (3) is that when connection i is being routed to a routing resource n, if the connection has high criticality (i.e. $Crit_i$ is close to 1), then the cost of using the routing resource is principally given by the delay of that resource (equal to $delay\_cost_n$). For such timing-critical connections, the principal concern should be to route the connections using the fastest routing resources, with less focus on the demand for such resources by other nets. On the other hand, if the criticality of the connection being routed is low (i.e. $Crit_i$ is close to 0), then the $cong\_cost_n$ is the dominant part of the cost of using a resource. As such, connections with sufficient timing slack are encouraged to use resources with less demand, potentially pursuing more circuitous routes to reduce demand on overcongested nodes.

Turning now to our modification of the VPR router, for a routing segment, n, we use the following cost function:

$$\begin{aligned} Cost_n &= (1-Crit_i)\cdot [cong\_cost_n \\ &\quad + PF \cdot (\alpha_i \cdot cap\_cost_n \\ &\quad + power\_cost\_infringe_n)] \\ &\quad + Crit_i \cdot delay\_cost_n \end{aligned} \tag{4}$$

where α_i is the activity factor for connection i being routed (normalized to a [0:1] range), PF is a scalar tuning parameter (determined empirically), and cap_cost_n is the effective capacitance of node n (taking into account current occupancies of adjacent conductors). The term $power\_cost\_infringe_n$ is equal to:

$$\sum_{m=1}^{3} neighbour\_cost(m) \cdot max\_activity(n,m) \tag{5}$$

where $neighbour\_cost(m)$ models the difference in coupling capacitance seen by neighbour m (recall our definitions for neighbour numbers from Section 5) if node n is not in a floating state, and $max\_activity(n,m)$ is the maximum activity factor corresponding to the nets using the mth neighbouring conductor of node n. Recall that during routing, multiple nets may be using a single routing conductor (it is only towards the end of routing when congestion is resolved that a conductor is used by at most one net), and as such we pessimistically assume the worst case activity factor among all nets using a neighbouring conductor in our cost function. Thus $power\_cost\_infringe_n$ models the worst case increase in power if node n is not kept in a floating state.

The motivation for the modification to the cost function is as follows: while routing a circuit, we wish to route connections which have high switching activity and/or criticality such that their neighbouring tracks are unoccupied. Occupied adjacent tracks cannot be put into a floating state, and therefore, such a scenario would lead to a lost opportunity for capacitance reduction. In general, when routing connection i, we need to consider two cases: 1) connection i has sufficient timing margin, or 2) connection i is timing critical. For the former case, we strive to optimize the routing for two different situations. First, we attempt to guide nets with high activity factor to use tracks with unoccupied neighbours; the term $(1-Crit_i)\alpha_i \cdot PF \cdot cap\_cost_n$ attempts to optimize for this goal. The term cap_cost_n models the capacitance of a track given the occupancies of adjacent tracks (i.e. unoccupied adjacent tracks will lead to a lower capacitance). As such if the current net has high activity factor (α_i), the router will attempt to find routing solutions where adjacent tracks are unused for this net. The second goal is to avoid using tracks adjacent to high activity nets. This is to ensure that the capacitance of the tracks seen by such nets in minimized. This optimization goal is dealt with by the term $(1-Crit_i)\cdot PF \cdot power\_cost\_infringe_n$. The term $power\_cost\_infringe_n$ models the impact to power resulting from use of the current track.

On the other hand, if the connection being routed has high criticality, then as always, we seek to optimize the timing of this net. While the cost function has no specific changes to optimize the timing of critical nets, in our cost function the $delay\_cost_n$ term models the effect of used/unused adjacent tracks (i.e. tracks with unused adjacent tracks will see a lower value for $delay\_cost_n$). As such, the router will naturally seek out paths where tracks have unused adjacent tracks in a bid to optimize the timing for that net.

7. EXPERIMENTAL STUDY

In order to assess the merits of the proposed capacitance reduction technique, we used our modified version of VPR to place and route the set of benchmark circuits packaged with VPR 6.0 [15] as well as circuits from the MCNC benchmark suite. We performed post-routing analysis to estimate the power reductions achievable using our proposed technique versus a baseline architecture. Our baseline architecture contains unidirectional wire segments (direct-drive) which span 4 CLB tiles, uses the Wilton switch block [16], and has logic blocks with ten 6-LUTs/FFs per CLB. All benchmark circuits were initially routed on the baseline architecture to determine the minimum number of tracks per channel needed to route each circuit successfully (W_{min}). For each circuit, we then computed $W = 1.3 \times W_{min}$ to reflect a medium-stress routing scenario. The computed W value for each circuit was used for all experimental runs of the circuit. We use the ACE switching activity estimator tool [17] to compute switching activity for each signal in each benchmark circuit; these are required by our modified CAD flow (see (4)) to optimize routing for power reduction. We used SPICE simulations with a commercial 65nm process to determine the added delay of our proposed tri-stateable buffer over a conventional buffer, to assess the capacitance reductions achievable for the various values of C_C/C_P considered in this study, and to study leakage power under various modes of operation.

7.1 CAD Tool Optimization

We first begin by analyzing the impact of our cost function modifications on power reduction/critical path delay, and optimize the scalar term (PF) in (4) accordingly. Fig. 9 shows the variation in both interconnect power reduction and critical path delay as PF is varied from 0 to 50, assuming that $C_C/C_P = 1.5$. The results show interesting trends: First, as PF is varied from 0 to 10, we see a steady increase in the achieved power reductions. This is intuitively expected because as PF is increased, the router is encour-

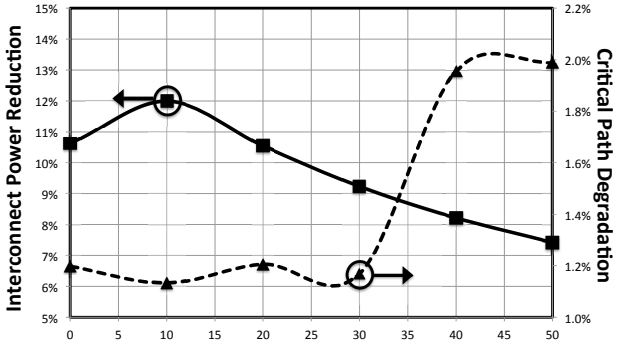


Figure 9: Interconnect dynamic power and critical path vs PF.

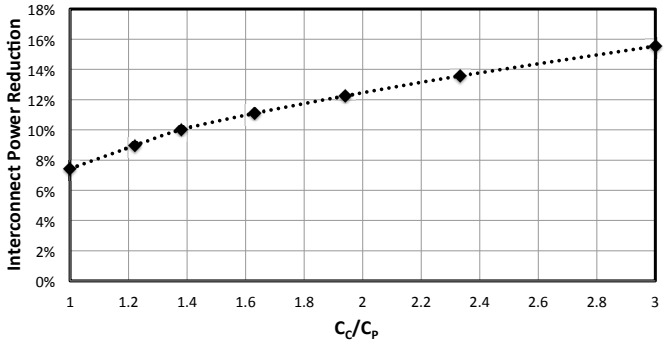


Figure 10: Interconnect dynamic power ratio vs C_C/C_P.

aged to spend more effort in creating situations where high activity nets have unused adjacent conductors. Thus, as PF is increased, more high activity nets will have unused adjacent conductors, thus allowing for increased power reduction. However, as PF is further increased from 10 to 50, we see a decrease in the achieved power reductions. This is because larger values of PF causes the router to place too much emphasis in creating situations where conductors adjacent to high activity nets are unused; in doing so, the router ends up increasing the average wirelength used to route each net, thus increasing dynamic power consumption. Thus a PF value of 10 serves as an reasonable balance between creating situations where the proposed technique can be used to decrease capacitance for high activity nets, while the average wirelength used by the router does not degrade the overall achieved power reductions.

The plot also shows the variation in critical path degradation as a function of PF. Due to the increased pull-down resistance of the first stage of our proposed tri-stateable buffer, our buffer suffers from a $\sim$10% delay increase compared to a conventional buffer. However, we are able to reduce the overall buffer delay penalty by using our proposed capacitance reduction technique. Thus, careful attention must be paid to the range of PF values which leads to minimized increases in critical path. The trend in Fig. 9 shows that the degradation to critical path remains constant over the range $0 \leq PF \leq 30$, indicating the router is able to route the critical path with a sufficient number of unused neighbouring conductors such that the overall degradation in critical path is restricted to $\sim$1% within this range. However, as PF is further increased from 30 to 50 we see a distinct degradation in critical path; this is attributed to the fact that increasing PF in these ranges results in increased wirelength for the critical path, thus resulting in increased critical path delay. Since a PF value of 10 results in the best power reduction and the minimal critical path degradation, we assume a PF value of 10 for the remainder of this paper.

7.2 Power Reduction Results

In lieu of a specific value for C_C/C_P which would vary with process node, and would depend on the specific layout of an FPGA tile, we assess the available power reductions and performance improvement as the ratio C_C/C_P is varied from 1-3. Fig. 10 shows the interconnect dynamic power reduction as a function of the ratio C_C/C_P. Note that previously it was shown that interconnect dynamic power accounts for 62% [6] of the total power consumption in a 90nm FPGA. Though, given that interconnect power does not scale as favourably as other contributors to total dynamic power (such as the power dissipated in individual logic elements), it is likely that interconnect dynamic power accounts for an even greater portion of the total dynamic power in 32nm and beyond.

As expected, the power reductions afforded by the proposed technique increase as the ratio C_C/C_P increases. Given that ITRS data shows the ratio C_C/C_P increasing with every passing technology node, it follows that the proposed technique offers excellent scalability – it is expected to allow for improved quality of results with each passing technology node. The interconnect power reductions exceed 15% as C_C/C_P approaches a value of 3 (i.e. C_C accounts for 75% of the total capacitance). Finally, Table 2 shows a circuit-by-circuit breakdown of power reductions, critical path degradations, and area overheads for the circuits considered in this work. Two sets of results are given for each circuit, one for $C_C/C_P = 1$ and a one for $C_C/C_P = 2$, since we believe that C_C/C_P for interconnect of an FPGA fabricated in the latest process technology would have a value between these two bounds.

7.3 Area

The proposed tri-stateable buffer uses three additional transistors: $M4$, $M5$, and $M6$. These transistors were sized such that the proposed buffer had a worst-case delay overhead of $\sim$10%. The resulting area overhead of the additional transistors is thus 3 minimum-width transistors ($M4$, $M5$ and $M6$ were all minimum width) in addition to an SRAM cell needed for the mode configuration of each routing buffer. This results in an increase in routing area of 5.1%, and an overall area overhead of 2.1%.

7.4 Leakage Power Reduction

Note that while the main thrust of this work is in the use of tristate buffers to reduce coupling capacitance and thus save dynamic power, one additional benefit worth noting is that tri-state buffers may be used to reduce leakage power for unused blocks; this was alluded to in Section 4.1, where we noted that simulation results showed that standby leakage power of the proposed buffer was $\sim$25% less than a conventional buffer. In our experiments, we counted the number of used and unused buffers in each design, and determined the overall reductions in leakage power of the routing buffers on a circuit-by-circuit basis; results show that routing static power can be reduced by $\sim$14.5%.

8. CONCLUSION

In this work, we presented a novel technique to reduce the capacitance of nets in FPGAs. We have shown that keeping unused routing conductors tied to a rail (V_{DD} or GND) is suboptimal and imposes an unnecessary increased coupling capacitance to used routing conductors. Thus, we showed how unused routing conductors may be exploited, since when they are left floating, they result in decreased capacitance to adjacent routing conductors. We then presented a low-cost tri-stateable buffer to allow for our proposed capacitance reduction technique with minimal area overhead. We

Circuit	$C_C/C_P = 1$			$C_C/C_P = 2$			Total Area Overhead %
	Routing Dynamic Power Reduction %	Critical Path Incr. %	Routing Static Power Reduction %	Routing Dynamic Power Reduction %	Critical Path Incr. %	Routing Static Power Reduction %	
alu4	8.1	1.6	14.9	12.0	0.9	14.9	1.8
apex2	7.9	0.9	14.1	11.9	1.4	14.1	2.1
apex4	7.5	-1.0	12.7	11.4	0.6	12.6	2.1
bigkey	9.7	1.1	13.7	14.3	1.6	13.8	1.8
clma	7.6	0.9	14.8	12.9	1.3	14.8	2.7
des	10.8	1.0	15.7	16.2	1.4	15.6	1.8
diffeq	8.0	0.5	14.6	12.5	0.7	14.8	2.1
dsip	9.4	0.9	15.6	14.2	2.3	15.7	2.1
elliptic	7.4	0.8	14.0	11.7	1.3	14.0	2.7
ex1010	6.7	1.1	13.6	11.1	1.7	13.6	2.4
ex5p	7.6	0.4	14.4	11.4	2.0	14.2	2.1
frisc	7.3	0.8	13.2	11.7	1.1	13.1	2.7
misex3	8.1	0.7	13.5	11.9	2.1	13.7	2.7
pdc	6.5	2.2	12.6	10.7	1.6	12.5	2.4
s298	7.9	0.2	14.4	13.0	0.2	14.3	1.8
s38417	7.1	1.1	13.4	11.6	1.6	13.4	2.4
s38584.1	6.7	0.8	13.0	11.0	1.1	13.1	2.4
seq	7.4	0.9	13.2	11.6	1.3	13.2	2.1
spla	6.1	5.0	12.9	10.1	1.8	12.9	2.4
tseng	10.5	-0.6	14.2	15.2	1.0	14.3	2.4
LU8PEEng	6.0	0.5	12.7	10.6	0.7	12.7	2.7
LU32PEEng	8.0	1.0	14.3	12.0	1.5	14.3	2.4
blob_merge	6.2	0.4	13.2	10.4	0.6	13.2	2.4
boundtop	8.5	0.7	13.5	13.3	0.9	13.6	2.1
ch_intrinsics	14.2	0.7	15.5	19.2	1.0	15.6	1.8
diffeq1	11.6	0.8	16.7	16.4	0.6	16.7	2.1
diffeq2	13.1	1.8	18.0	17.6	0.6	18.0	1.8
mcml	8.0	0.8	14.9	13.3	1.0	14.9	2.1
mkDelayWorker32B	7.7	2.7	18.1	13.2	1.0	18.1	2.1
mkPktMerge	10.0	0.7	21.0	16.5	0.9	21.0	1.5
mkSMAdapter4B	7.8	0.7	15.3	12.4	1.1	15.2	2.1
or1200	7.6	0.2	16.3	12.4	0.3	16.3	2.1
raygentop	8.1	0.6	13.8	12.9	0.8	13.9	2.1
sha	6.8	0.5	13.1	11.0	0.7	13.3	1.8
stereovision0	7.1	0.7	13.8	12.0	1.0	13.8	1.8
stereovision1	6.4	1.3	14.3	11.0	1.9	14.3	2.4
stereovision2	8.7	1.0	14.1	12.9	1.3	14.1	2.7
stereovision3	14.6	0.3	15.4	19.2	0.4	15.8	1.8
geomean	**8.2**	**0.9**	**14.5**	**12.8**	**1.1**	**14.5**	**2.1**

Table 2: Detailed power reductions, critical path increases and area overheads for each circuit.

further present CAD techniques to optimize the routing of a design to maximize capacitance reductions with our proposed technique. Results have shown interconnect dynamic power reductions of up to ∼15.5% when the ratio between coupling capacitance and plate capacitance of a wire, C_C/C_P, reaches a value of 3. These savings come with a critical path degradation of ∼1%, on average, and a total area overhead of ∼2.1%.

9. REFERENCES

[1] N. Magen, A. Kolodny, U. Weiser, and N. Shamir, "Interconnect-power dissipation in a microprocessor," in *SLIP '04*, 2004, pp. 7–13.

[2] R. Krishnamurthy, S. Mathew, M. Anders, S. Hsu, H. Kaul, and S. Borkar, "High-performance and low-voltage challenges for sub-45nm microprocessor circuits," in *ASICON 2005*, vol. 1, 2005, pp. 283–286.

[3] M. L. Mui, K. Banerjee, and A. Mehrotra, "A global interconnect optimization scheme for nanometer scale VLSI with implications for latency, bandwidth, and power dissipation," *IEEE T-ED*, vol. 51, no. 2, pp. 195–203, 2004.

[4] X.-C. Li, J.-F. Mao, H.-F. Huang, and Y. Liu, "Global interconnect width and spacing optimization for latency, bandwidth and power dissipation," *IEEE T-ED*, vol. 52, no. 10, pp. 2272–2279, 2005.

[5] ITRS, "ITRS 2011 Report." [Online]. Available: http://www.itrs.net/Links/2011ITRS/Home2011.htm

[6] T. Tuan, A. Rahman, S. Das, S. Trimberger, and S. Kao, "A 90-nm low-power FPGA for battery-powered applications," *IEEE TCAD*, vol. 26, no. 2, pp. 296–300, 2007.

[7] H. Wong, V. Betz, and J. Rose, "Comparing FPGA vs. custom CMOS and the impact on processor microarchitecture," in *ACM FPGA*, 2011, pp. 5–14.

[8] D. Sinha and H. Zhou, "Statistical timing analysis with coupling," *IEEE TCAD*, vol. 25, no. 12, pp. 2965–2975, 2006.

[9] Altera, personal communication.

[10] J.-A. He and H. Kobayashi, "Simultaneous wire sizing and wire spacing in post-layout performance optimization," in *ASP-DAC*, 1998, pp. 373–378.

[11] K. Moiseev, S. Wimer, and A. Kolodny, "Timing optimization of interconnect by simultaneous net-ordering, wire sizing and spacing," in *IEEE ISCAS*, 2006, pp. 21–24.

[12] V. Betz and J. Rose, "Circuit design, transistor sizing and wire layout of FPGA interconnect," in *IEEE CICC*, 1999, pp. 171–174.

[13] S. J. E. Wilton, "A crosstalk-aware timing-driven router for FPGAs," in *ACM FPGA*, 2001, pp. 21–28.

[14] L. McMurchie and C. Ebeling, "PathFinder: A negotiation-based performance-driven router for FPGAs," in *ACM FPGA*, 1995, pp. 111–117.

[15] J. Rose, J. Luu, C. W. Yu, O. Densmore, J. Goeders, A. Somerville, K. B. Kent, P. Jamieson, and J. Anderson, "The VTR project," in *ACM FPGA*, 2012, pp. 77–86.

[16] S. J. Wilton, "Architecture and algorithms for field-programmable gate arrays with embedded memory," Ph.D. dissertation, 1997.

[17] J. Lamoureux and S. Wilton, "Activity estimation for field-programmable gate arrays," in *IEEE FPL*, 2006, pp. 1–8.