# Optimal algorithm for minimizing the number of twists in an on-chip bus*

Liang Deng *and* Martin D. F. Wong
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign
Email: {*ldeng,mdfwong*}*@uiuc.edu*

***Abstract— Complementary bus architecture is used to achieve higher speed and lower power in VLSI chips. However, in deep submicron circuit design, the effects of crosstalk become more and more serious, especially in the bus structure where wires are placed close to each other. Complementary bus architecture with twisted wires can reduce the coupling noise. But in current chip design flow, engineering change order (ECO) happens commonly to meet improvement requirement. Layout changes due to ECO introduce obstacles to the twists, which could reduce the number of twists and increase the coupling noise. In this paper, an ECO algorithm for generating twisted complementary architecture is proposed based on the shortest path algorithm. Our algorithm guarantees to give the minimum number of twists along the bus wires under noise constraints. Experimental results show that the twist patterns generated by our algorithm can effectively reduce the capacitive coupling noises.***

## 1. INTRODUCTION

As integrated circuits continue to scale into nanometer region, multi-millions gates can be integrated on a single chip [1]. It makes system-on-chip (SoC) design possible and can achieve an operating frequency of multi GHz. But this deep sub micron (DSM) technology also brings up a new issue: interconnection becomes a more critical in IC design. The on-chip buses connect tens to hundreds of modules in a SoC chip. They can consume a large fraction of system power and influence system performance greatly. Complementary bus architecture is proposed in [2] to solve this problem for high speed and low power operation. It transmits one bit of information with complementary signals through a pair of wires as shown in Fig 1. Low swing voltage can be applied to the bus wires to transmit the signals so that power consumption is reduced. With the differential amplifier at the output driver, high speed signal transmission can be achieved. The same technology has been widely used in memory chips for a long time [3]–[6]. Signals with enhanced slew rate can be transmitted on this bus architecture to get better system performance. Lower voltage can also be applied to this bus so that the system power consumption is reduced.

However, on-chip bus architecture becomes most sensitive to noise as technology scales down: taller and thinner wires are placed closer to each other, and bus wires run in parallel for longer distance. Noise has become a more dominant problem for the on-chip design compared to area, timing, and power. Moreover, enhanced slew rate can bring more severe coupling noise, and lower voltage on bus lines also reduces the noise margin. So various methods are proposed to solve the noise problem in on-chip bus architecture. Repeater insertion [8] is widely used to decrease wire delay and effectively reduce crosstalk noise. Bus shielding can also be used for noise avoidance for critical signals. In the complementary bus architecture, twisting wires can be used to reduce the capacitance coupling noise more effectively with little area or power penalty. With the twists, wires in each pair see the same coupling noise from other bus wires, and the differential amplifier will cancel this coupling noise at the output node. Twisted bus architecture is widely used in semiconductor memory chips as the global bit-lines. Twist bundle [7], a similar technology, can also reduce the inductive coupling effectively.

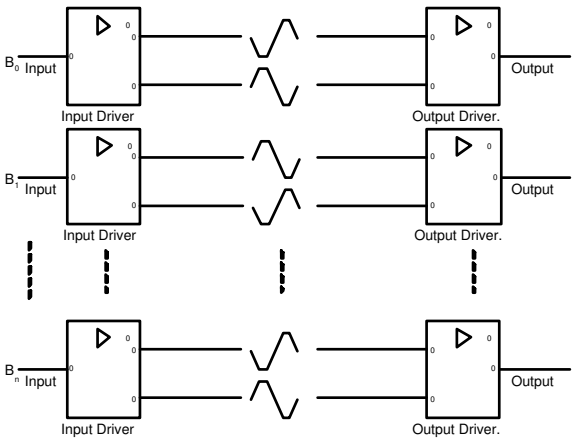


Fig. 1. Complementary bus architecture

Twists can be made easily with two metal layers in the bus architecture if space is reserved intentionally along the bus lines. But nowadays, iterations in design flow are inevitable. To meet the target changes from manufacturing, marketing, reliability, or performances, engineering change order (ECO) happens frequently in the real design flow. When an ECO happens, the layout is also changed. These changes may introduce obstacles that overlap with existing twists, as shown in Fig 2. Besides the ECO introduced obstacles, many routing wires, which are placed in the bus structure area to reduce

*This work was partially supported by the National Science Foundation under grants CCR-0244236 and CCR-0306244.

1530-1591/04 $20.00 (c) 2004 IEEE

chip area, can also be obstacles to twists. So when ECO-introduces obstacles to existing twist locations, twists cannot be easily moved new locations with the guarantee that the new twist pattern can still reduce the coupling noise effectively. Currently this problem is solved manually which requires extreme design effort to search the space for twists in the complicated IC layout. Usually, the number of twists will be reduced to avoid the obstacles brought by ECO. But the noise will increase if the twist number is decreased, and thus the distance between two adjacent twists increases as shown in Fig 3. A computer algorithm to make twist pattern is greatly needed to reduce the capacitive coupling in complementary bus architecture.

It is easy to generate twists along the bus line at all possible locations that wires can be twisted. However, more twists bring more resistive and capacitive contributions because of the metal-to-metal via in a twist. So a trade-off must be made to determine the twist number, i.e. reducing the capacitive coupling effectively with minimal twist number. In this paper, we develop an efficient algorithm that can generate an optimal twist pattern automatically to reduce the crosstalk noise while minimizing the number of twists in the on-chip bus structure. Our twist pattern generation (TPG) algorithm can make a twist pattern in polynomial time by solving the shortest path problem formulated from the ECO twist generation problem.

In the next section, we present the twisted complementary bus architecture and formulate an ECO twist generation problem in Section 3, we propose a polynomial time TPG algorithm to generate a twist pattern that can reduce the coupling noises while minimizing the number of twists. Some extensions are given in Section 4. Finally, we show some experimental results in Section 5 and conclude the paper in Section 6.

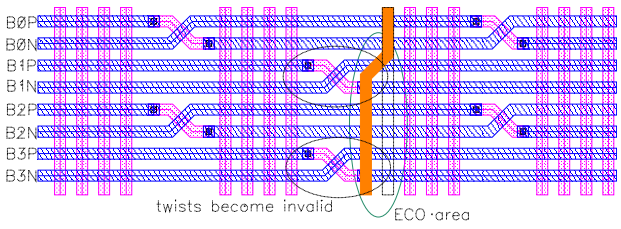


Fig. 2. ECO happens and one metal wire is changed. the original metal wire is changed from break lined area to dark shaded area and becomes obstacle to twists. The twist pattern should be revised according to the ECO.

## 2. Problem Formulation

A complementary bus is composed of pairs of wires. We will use terms "complementary bus architecture" and "paired bus structure" interchangeably throughout this paper. The wires in the complementary bus are routed in parallel to make a folded untwist bus architecture as shown in Fig 4(a). Every pair consists of two adjacent wires which use complementary signal to transfer one bit of information. So a bus with $n$-bit width needs $2n$ wires, i.e. $n$ pairs. $BLi$ and $\overline{BLi}$ are used to denote the $i^{th}$ pair of wires in the bus. When $i = 0, 2, 4, \ldots, 2n$, the pairs are called *even pairs*. The rest are called *odd pairs*. This complementary bus architecture can achieve higher speed and lower power than the common bus structure. It is widely used in memory chips as global bit-line or in SoC chips to connect the interfaces of IPs.

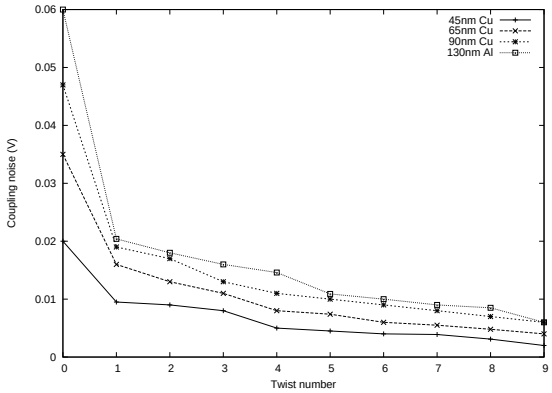


Fig. 3. Number of twists *vs.* capacitive coupling noise

The crosstalk noise in the bus structure is usually from the coupling capacitance and coupling inductance. However, inductive coupling will be reduced naturally by the twisted bus structure, which will be explained later in this section. Compared to the prominent capacitive coupling, we ignore the inductive coupling in this problem. "Capacitive coupling", "crosstalk noise" and "coupling noise" will be used interchangeable in the rest of this paper. Our algorithm addresses the problem of generating a twist pattern to effectively reduce the capacitive coupling.

To save chip area, space between wires is minimized in bus structure. So the crosstalk reduction problem here is quite different from the noise minimization in channel routing [11] or global routing [12], in which the location of wires can be changed to reduce the capacitive coupling. Some other techniques can be used to reduce the crosstalk, such as inserting buffers. But adding extra circuits will increase the power consumption. To solve this problem, twists along the wires are proposed to reduce the capacitive coupling as shown in Fig. 4(b) and (c). The twisted bit-line(TBL) bus architecture [3] in Fig 4(b) is now widely used in semiconductor industry. Multiple twisted bit-line (MTBL in Fig 4(c))technique is proposed in [4] to eliminate more coupling noise in memory chips.

Fig. 4(a) shows the coupling capacitance in a folded untwisted complementary bus structure. Because the capacitive coupling is a short-range noise [10], coupling noise decreases greatly as the space increases between two wires. To simplify the capacitive coupling modeling, the second-order effects of capacitive coupling are ignored, and only the adjacent wires have coupling capacitance to each other. The coupling capacitance between two wires in one pair is intra-BL coupling; the coupling from other pairs is called inter-BL coupling. The inter-BL coupling noise can be reduced effectively in the twisted bus in Fig. 4(b). For example, The inter-BL couplings

on $BL1$ pair are from $BL0$ pair and $BL2$ pair. The capacitive coupling from $BL2$ pair on $BL1$ is decided by $C_{i0}$ and the coupling to $\overline{BL1}$ is $C_{i1}, (i = 0, 1, 2)$. Assuming the bus is divided into several parts, each part has length $L_i$. If the length of one part is not too long, (i.e. $L_i \leq \Delta_{max}$), according to the transmission line model, we have $C_{i0} = C_{i1}$. Thus $\sum_i C_{i0} = \sum_i C_{i1}$, and the capacitive coupling noise on the $BL1$ pair from the $BL2$ pair will be canceled in the differential amplifier, which is also called sense amplifiers(SA), at the end of the $BL1$ pair. It is the same with the capacitive coupling from the $BL0$ pair to the $BL1$ pair. Thus the inter-BL coupling to the $BL1$ pair is eliminated. Now consider the coupling capacitances to $BL2$ from $BL1$ pair and the $BL3$ pair. The coupling noise will still be reduced: Since $BL1$ pair and $BL3$ pair transfer complement signals. In each pair, one line is pulled up and the other one must be pulled down. The coupling noise from $BLi$ and $\overline{BLi}$ can cancel each other. For example, the coupling from $C_{30}$ and $C_{31}$ will cancel each other. So the worst case in the folded untwisted bus, which adjacent lines always switch to opposite voltage levels, will not happen in this twisted bus structure.

In the MTBL scheme, twists can reduce both the inter-BL coupling and the intra-BL coupling as shown in Fig. 4(c). If $L_i \leq \Delta_{max}$, only one multiple twist is needed in one part to meet noise requirement of design. The twist can be placed in the middle of $L_i$, and the capacitive coupling is reduced in each $L_i$ part.

Though twists can reduce the coupling noise in the paired bus structure, they can also introduce more capacitance and resistance to the bus line, which will cause excessive signal delay. So two twists cannot be too near, i.e. $\Delta L \geq \Delta_{min}$, where $\Delta L$ is the distance between two adjacent twists on paired wires. Together with the constraint that two adjacent twists cannot be too far away as discussed previously, the distance between two adjacent twist $L$ must obey the *boundary rule*:

$$\Delta_{min} \leq \Delta L \leq \Delta_{max}$$

When designing the layout of the twist pattern in a complementary bus, $\Delta_{min}$ and $\Delta_{max}$ are determined by state-of-art simulation method according to the noise constraints.

When ECO happens in a twisted complementary bus, the twist pattern needs to be redesigned to meet the design specification. We can formulate it as a *ECO twist generation problem* that generates twist pattern on a complementary bus to reduce the coupling noise under the following rules:

(i) Twists must not overlap on obstacles.
(ii) Twists must obey the boundary rule.
(iii) The number of twists should be minimized.

In the previous discussion, we totally ignored the inductive coupling. However, as the clock frequency continues to increase, on-chip inductance becomes evident, especially the mutual inductance. It should be considered for high performance bus architecture. If the following conditions are met, inductance will be significant and must be count in design concern [9]:

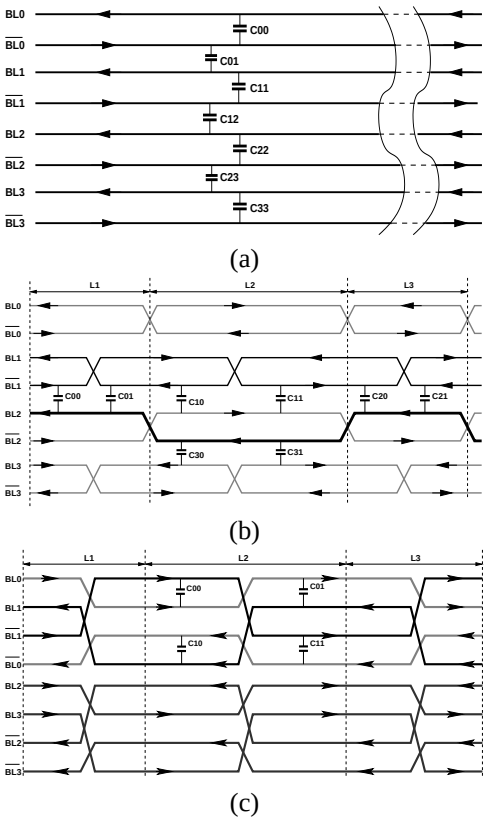


Fig. 4. (a) A 4-bit complementary bus without twists. The current flows shown in this figure illustrate the worst case for the coupling noise. (b) Twisted bit-line(TBL) scheme to reduce the inter-BL coupling (c) Multiple Twisted bit-line (MTBL) scheme to reduce both inter-BL and intra-BL coupling noise

- $R < \omega L$,where $R$ is the unit length resistance of the wire, and $L$ is the unit length inductance of wire. $\omega$ is the characteristic frequency.
- $t_r < 2t_f$, where $t_f = l\sqrt{LC}$, $C$ is the unit length capacitance and $l$ is the length of the wire. $t_r$ is the typical slew time $t_r = 2\pi\omega$

In the GHz+ VLSI design, the inductance cannot be ignored. In twisted complementary bus architecture, the mutual inductance is reduced by the complementary signals together with the twist structure. For example, in TBL bus structure, the currents in one pair are always in opposite directions; thus, the magnetic fluxes made by these currents are in opposite direction. One of the mutual inductances made by this pair is positive and the other is negative. The inductive coupling in a complementary bus is reduced. Consider the best case, in every pair the line that is nearer to line $BLi$ has the opposite current direction to the current in $BLi$ as shown in Fig. 4(a). And in worst case that the current directions are reversed. The matrix $U$ is used to express the current direction in the lines upper to $BLi$, i.e. $\{BL(i-1), \overline{BL(i-1)}, BL(i-2), \ldots, BL0\}$, total $a$ lines. And $L$ is to express the current direction in lines lower to $BLi$, i.e. $\{\overline{BLi}, BL(i+1), \overline{BL(i+1)}, BLi+2, \ldots, \overline{BL(N)}\}$, total $b$ lines:

$$U_{best} = [-1, -1, +1, -1, +1, -1, +1, \ldots]$$

$$L_{best} = [-1, +1, -1, +1, -1, +1, -1, \ldots]$$
$$U_{worst} = [+1, -1, +1, -1, +1, -1, +1, \ldots]$$
$$L_{worst} = [-1, +1, -1, +1, -1, +1, -1, \ldots]$$

Assuming the current magnitude in every line is same and define matrix $K$ as:

$$K_n = [L_1, L_2, L_3 \ldots, L_n]$$

$L_i$ is the magnitude of mutual inductance between this line and the $i^{th}$ line next to this line. So the effective inductance on $BL_i$ can be expressed as:

$$L_{i,eff} = L_{i0} + UK_a^T + LK_b^T$$

where $L_{i0}$ is the self inductance of $BL_i$. It is obvious that the complementary bus architecture can reduce the inductive coupling. When $a = b$, the bound of inductance of $L_i$ can be expressed as:

$$L_{i0} - 2K_0 \leq L_{i,eff} \leq L_{i0}$$

So we ignore the inductive coupling in this problem and only consider to generate twist pattern to reduce the capacitive coupling noise in bus structure.

## 3. Twist Pattern Generation (TPG) Algorithm

Both the TBL and MTBL schemes can reduce the coupling noise in the complementary bus architecture. The TBL scheme is more widely used in current IC designs. We will introduce our TPG algorithm to generate the twist pattern in TBL bus first. It is easy to extend this algorithm to MTBL scheme, which will be discussed in Section 4.

TPG algorithm has two phases. Phase 1 is to build a graph from the bus structure, abstracting the vertices from the bus wires, and building edges according to both obstacles and the boundary rule. Phase 2 is to make the twist pattern by solving the shortest path problem. Phase 2 guarantees that the number of twists is minimized. Before presenting the algorithms, we will first express the paired bus structure using a matrix:

$$\mathbf{B} = \begin{bmatrix} b_{11} & b_{12} & \ldots & b_{1P} \\ b_{21} & b_{22} & \ldots & b_{2P} \\ \vdots & \vdots & \vdots & \vdots \\ b_{M0} & b_{M1} & \ldots & b_{MP} \end{bmatrix}$$

The bus structure can be expressed in the matrix $\mathbf{B}$ by the following rules:

1) A paired bus structure has $2N$ lines. And $M$ is defined as $M = 2N$.
2) $[b_{i1}, b_{i2}, \ldots, b_{iP}]$ represents the $i^{th}$ line of the bus structure.
3) Each line has a length $L$. It can be divided into $P$ parts with uniformed length $l$, $l = L/P$. And $b_{ij}$ corresponds the $j^{th}$ part in $i^{th}$ bus line.

The matrix can also include the obstacle information in the bus structure. We can define the value of $b_{ij}$ as follows:

$$b_{ij} = \begin{cases} 0 & \text{twist CANNOT be made at } b_{ij} \\ 1 & \text{twist CAN be made at } b_{ij} \end{cases}$$

Thus, the paired bus structure can be expressed in matrix $\mathbf{B}$. Then we can construct a graph $G(V, E)$ easily from matrix $\mathbf{B}$. In TBL scheme, twist can be made on odd pairs when $b_{ij} = 1\,(i \mod 4 = 0 \; or \; i \mod 4 = 1)$. It can be made on even pairs when $b_{ij}\,(i \mod 4 = 2 \; or \; i \mod 4 = 3)$. A twist made on even pair is always at the middle of two adjacent twists on odd pair. Thus the coupling noise will be reduced effectively as described in previous section. Twists should be introduced at the same locations in all even pairs or all odd pairs. Two vectors can be used to represent the obstacles in a bus: $V_{even} = [v_1.even, v_2.even, \ldots, v_P.even]$ and $V_{odd} = [v_1.odd, v_2.odd, \ldots, v_P.odd]$. Obstacle vectors $V_{even}$ and $V_{odd}$ store the obstacle information in even pairs and odd pairs, respectively.

*Algorithm 1 (build_obstacle_vectors_TBL):*

```
1  for j := 1 to P
2      for i := 1 to N/2
3          s := i;  t := i + 1
4          if (b_sj = 0) and (b_tj = 0) then
5              v_j.even := 0;  break;
6          else
7                  v_j.even := 1;
8          s := i + 2;  t := i + 3
9          if (x_sj = 0) and (x_tj = 0) then
10             v_j.odd := 0;  break;
11         else
12                 v_j.odd := 1;
13 v_1.even := 1;  v_1.odd := 1;
14 v_P.odd = 1;  v_P.even = 1;
```

The graph $G(V, E)$ can be built from obstacle vectors and the boundary rule. The term $v_1$ is the beginning of the bus structure, and is assumed as the source node of the graph. the term $v_P$ is the end part of the bus line that is connected to the differential sense amplifier, and that is assumed to be the sink node.

*Algorithm 2 (build_graph_TBL):*

```
1 {V} = {v_1};
2 {E} = ∅;
3 for i := 0 to (P − Δ_min)
4     for j := Δ_min to Δ_max
5         if (i + j > P) then break;
6         else if ((v_{i+j}.odd = 1) and (v_{(i+j)/2}.even = 1)
7                 and (v_i ∈ {V})) then
8                 {V} = {V} ∪ {v_{i+j}};
9                 {E} = {E} ∪ (v_i, v_{i+j});
```

In $G(V, E)$, the weight on every edge is set to 1. If the shortest path from $v_1$ to $v_P$ is found, the number of twists in the bus structure is minimized. So we can solve the shortest path problem to generate twist pattern for the paired bus structure:

*Algorithm 3 (generate_twist_TBL):*

```
1 build_obstacle_vectors_TBL;
2 build_graph_TBL;
3 Calculate the shortest path P from v_1 to v_P;
```

4 foreach edge *(u,v)* in $\mathbf{P}$
5 make twists at *v* on odd pairs
6 make twists middle of *(u,v)* on even pairs

Fig. 5 shows a simple example of TPG algorithm. The bus structure has 4 line to transfer 2-bit signal. It is divided into several parts. The boundary rule is noted in Fig. 5(c). The shaded area are obstacles to twists along the bus lines. The blank area means there is no obstacle on this part. From this bus structure, a graph in Fig. 5(b) can be built. And the shortest path is calculated out by TPG algorithm stated in this section. The twisted complementary bus architecture with obstacles is shown in Fig. 5(c). The *build_graph_TBL* algorithm guarantees

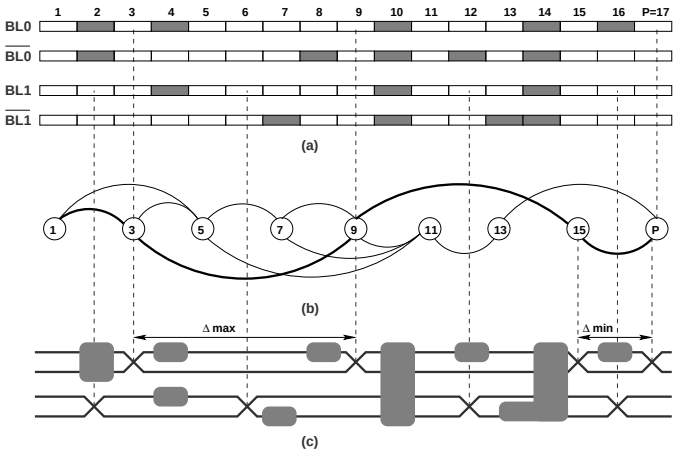


Fig. 5. (a) A bus with 2 pairs of lines with obstacles for twist generation. (b) Its corresponding graph $G(V, E)$ with shortest path highlighted. (c) The twist generation solution corresponds to the shortest path.

that the twist pattern can meet the rule (i) and (ii) mentioned in Section 2. By solving the shortest path problem, requirement (iii) is met. Thus, an optimal twist pattern is generated and the coupling noise can be effectively reduced in an ECO bus structure.

To build the graph $G$ corresponding the complementary bus architecture, every element in Matrix $\mathbf{B}$ should be explored to represent the obstacles. The complexity for building obstacle vectors is $O(P \times M)$. Edges are built from the obstacle vectors and boundary rule, which the complexity is $O(P \times (\Delta_{max} - \Delta_{min}))$. Some standard single-source shortest path algorithms, such as Dijkstra's algorithm [14], can be used here to solve the shortest path problem in $G$. The complexity is $O(V^2 + E)$ which is related to the $|V|$ and $|E|$. Table I in the Section 5 shows the statistical result of the relationship between the obstacle ratio and the $|V|$, $|E|$ and the TPG success ratio. $N$ is the total number of $v_i$ where a twist can be made, i.e. $v_i = 1$. $|E|$ is the average number of edges in graphs built from sample buses, and $t_{avg}$ is the average runtime.

## 4. Extension

To extend our TPG algorithm for the MTBL scheme, Phase 1 of the algorithm should be modified. The twists on odd and even pairs in MTBL bus are at the same location. So we modify the algorithms in Phase 1 as follows:

*Algorithm 4 (build_obstacle_vectors_MTBL):*
1 for $j := 1$ to $P$
2 for $i := 1$ to $M$
3 if $(b_{ij} = 0)$ then
4 $v_j := 0$; $break$;
5 else
6 $v_j := 1$;
7 $v_1 := 1$; $v_p := 1$;

*Algorithm 5 (build_graph_MTBL):*
1 $\{V\} = \{v_1\}$;
2 $\{E\} = \emptyset$;
3 for $i := 0$ to $(P - \Delta_{min})$
4 for $j := \Delta_{min}$ to $\Delta_{max}$
5 if $(i + j > P)$ then $break$;
6 else if $(v_{(i+j)/2} = 1)$ $and$ $(v_i \in \{V\})$
7 $\{V\} = \{V\} \cup \{v_{i+j}\}$;
8 $\{E\} = \{E\} \cup (v_i, v_{i+j})$;

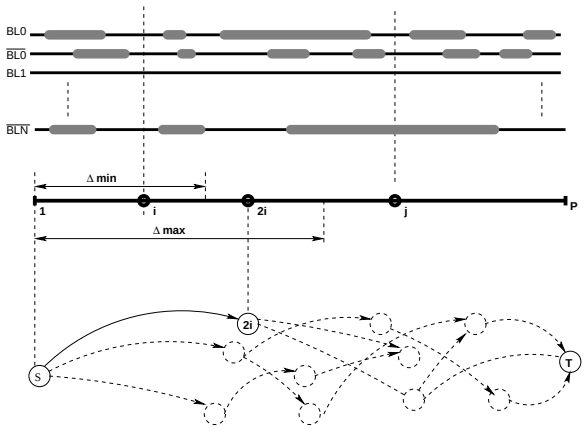


Fig. 6. Building the graph form the bus structure in MTBL scheme.

Fig. 6 illustrates the graph generated in the MTBL scheme. From the source node $v_1$, we scan the $v_j$ whose distance to $v_1$ is in the boundary rule. If $v_{2i}$ is in this boundary and $v_i = 1$ which means a twist can be made here, the node $v_{2i}$ is pushed into $V$ as a vertex and an edge is built between $v_1$ and $v_{2i}$. Similarly, $G$ can be built from the bus structure as shown in Fig. 6, and the weight of every edge is set to 1. This algorithm builds an edge when it can make a twist in the middle of bus between $v_i$ and vertex $v_{i+j}$. It guarantees that the space between any two twists is in boundary $[\Delta_{min}, \Delta_{max}]$.

*Algorithm 6 (generate_twist_MTBL):*
1 *build_obstacle_vectors_MTBL;*
2 *build_graph_MTBL;*
3 Calculate the shortest path $\mathbf{P}$ $form$ $v_1$ $to$ $v_P$;
4 foreach edge *(u,v)* in $\mathbf{P}$
5 make twists at middle of *(u,v)*

## 5. Experimental Results

Our algorithms are implemented in C and executed on a Sparc Ultra 250 workstation. In our implementation, we use

TABLE I

THE OBSTACLE RATIO AND THE TPG SUCCESS RATIO

| obstacle ratio | TBL scheme | | | |
|---|---|---|---|---|
| | TPG success | N | $\|E\|$ | $t_{avg}$ |
| 96%-99% | 0% | - | - | - |
| 91%-95% | 0.6% | 390 | 8 | 0.29s |
| 86%-90% | 18.2% | 592 | 47 | 0.24s |
| 81%-85% | 80.0% | 812 | 747 | 0.17s |
| 76%-80% | 98.6% | 1050 | 22091 | 0.18s |
| 71%-75% | 100% | 1298 | 242716 | 0.23s |
| 66%-70% | 100% | 1551 | 900288 | 0.27s |
| 61%-65% | 100% | 1800 | 2087370 | 0.32s |
| | MTBL scheme | | | |
| 96%-99% | 0% | - | - | - |
| 91%-95% | 0% | - | - | - |
| 86%-90% | 21.6% | 662 | 908 | 0.15s |
| 81%-85% | 88.2% | 807 | 3762 | 0.15s |
| 76%-80% | 98.6% | 1051 | 10340 | 0.19s |
| 71%-75% | 99.8% | 1298 | 18502 | 0.24s |
| 66%-70% | 100% | 1551 | 28258 | 0.28s |
| 61%-65% | 100% | 1799 | 39316 | 0.33s |

the algorithm to generate TBL and MTBL twist patterns in 64-bit bus that consists of 128 bus wires. Bus wires are 0.5mm in length.

HSPICE is used to simulate the coupling noises in several different CMOS technologies. The parameters are from ITRS2001 [1]. To get the worst-case result, we assumed opposite switching of all adjacent lines for all of the simulations. And the input resistance is set to $100\,\Omega$ . The output load capacitance is set to $15\,fF$. Every line is initially charged to half of the power supply voltage. To show the coupling noise, the delay of switch time on the victim bit-lines is set to 0.1 ns. A rising/falling edge of 0.025 ns is used for all drives. The wire length is $500\,\mu m$. In Table II, $V_{NOISE}$ is the maximum voltage difference due to the coupling noise.

Both the generated TBL and MTBL twist pattern has four twists on every bus wire, one twist less than the twist pattern before ECO happens. While in ECO TBL bus, the twist number is reduce to two on every bit-line because of the new obstacle made by ECO. The data Table II shows the twist pattern made by TPG algorithm can greatly reduce the coupling noise in the paired bus structure. Assuming the noise is 100% in a folded untwisted bit-line bus, generated twist pattern can reduce more than 80% of the noise. Thus it can improve the chip performance and reliability.

## 6. CONCLUSION

In this paper, a twist pattern generation algorithm is proposed. It can solve the ECO twist problem when ECO happens. This algorithms can generate twist pattern to minimize the number of twists while reducing the capacitive coupling effectively. It also ensure that a feasible solution, as long as one exists, can be found. The simulation results show that this algorithm is effective reducing the capacitive crosstalk noise in the complementary bus.

[1]"The ECO TBL bus" in Table II is the twisted bit-line bus after ECO happens. Because of the new obstacle the ECO made, the twists number is reduced to half of before ECO or even less.

TABLE II

SIMULATION RESULTS OF THE TWISTED BIT-LINE BUS STRUCTURE

| bit-line | twist | $V_{noise}$ (V) | $vdd$ (V) | Noise |
|---|---|---|---|---|
| 130 nm Al | folded untwist bus | 0.060 | 1.8 | 100% |
| | ECO TBL bus [1] | 0.011 | 1.8 | 18% |
| | generated TBL | 0.006 | 1.8 | 10% |
| | generated MTBL | 0.006 | 1.8 | 10% |
| 90 nm Cu | folded untwist bus | 0.047 | 1.5 | 100% |
| | ECO TBL bus | 0.011 | 1.5 | 23% |
| | generated TBL | 0.006 | 1.5 | 13% |
| | generated MTBL | 0.006 | 1.5 | 13% |
| 65 nm Cu | folded untwist bus | 0.035 | 1.2 | 100% |
| | ECO TBL bus | 0.008 | 1.2 | 23% |
| | generated TBL | 0.004 | 1.2 | 11% |
| | generated MTBL | 0.004 | 1.2 | 11% |
| 45 nm Cu | folded untwist bus | 0.020 | 1.2 | 100% |
| | ECO TBL bus | 0.005 | 1.2 | 25% |
| | generated TBL | 0.002 | 1.2 | 10% |
| | generated MTBL | 0.002 | 1.2 | 10% |

## REFERENCES


[1] Semiconductor Industry Association(SIA). "The International Technology Roadmap for Semiconductors 2001". http://public.itrs.net. Apr, 2001.

[2] Y. Nakase, A. Iwabu, K. Mashiko, Y. Matsuda, and T. Tokuda. "Complementary half-swing bus architecture and its application for wide band SRAM macros." In Proc. IEE Circuits Devices and Systems, vol. 145, No. 5, pp. 337-342, Oct. 1998.

[3] H. Hidaka, F. Fujishima, Y. Matsuda, M. Asakura, and T. Yoshihara. "Twisted bit-line architecture for Multi-Megabit DRAMs." In IEEE J. Solid-State Circuits, vol. 24, pp. 21C28, Feb. 1989.

[4] D. S. Min and D. W. Langer. "Multiple Twisted Dataline Techniques for Multigigabit DRAMs." In IEEE J. Solid-State Circuits, vol. 34, pp. 856-865, Jun. 1999.

[5] M. Redeker, B. F. Cockburn, and D. G. Elliott. "An Investigation into Crosstalk Noise in DRAM Structures." In The 2002 IEEE International Workshop on Memory Technology, Design and Testing, 2002.

[6] Noda, K., *et al*. "An ultrahigh-density high-speed loadless four-transistor SRAM macro with twisted bitline architecture and triple-well shield." In IEEE J. Solid-State Circuits, vol.36 issue 3, pp.510-515, Mar. 2001.

[7] G. Zhong, C. Koh, and K. Roy. "A Twisted-Bundle Layout Structure for Minimizing Inductive Coupling Noise." In proc. IEEE/ACM International Conference onComputer Aided Design, pp. 406-411, 2000.

[8] D. Li, A. Pua, P. Srivastava, and U. Ko. "A repeater optimization methodology for deep sub-micron, high-performance processors." Proc. of ICCD'97, pages 726-731, 1997

[9] K. L. Shepard and V.Natayanan. "Noise in Deep Submicron Digital Design." In Proc. International Conference on Computer Aided Design, pp. 524-531, 1996.

[10] T. Sakurai. "Closed form expresstions for interconnection delay, coupling and crosstalk in VLSI's." In IEEE Trans. Electron Devices, vol. 40, pp.118-124, Jan. 1993.

[11] T. Gao and C. L. Liu. "Minimum Crosstalk Channel Routing." In Proc. IEEE/ACM International Conference on Computer-Aided Design, pp. 692-696, 1993.

[12] T. Xue, E.S. Kuh, and D. Wong. "Post global routing crosstalk risk estimation and reduction." In Proc. IEEE/ACM International Conference on Computer-Aided Design, pp. 302-309, 1996.

[13] H. B. Bakoglu. "Circuits, Interconnections and Packaging for VLSI." Addison Wesley, Ch. 6-7, 1990.

[14] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, "Introduction to Algorithms." 2nd ed. MIT Press, McGraw-Hill, 2001.