Architectural Energy Optimization by Bus Splitting

Cheng-Ta Hsieh and Massoud Pedram

Dept. of EE-Systems

University of Southern California

Los Angeles, CA 90089

Abstract

This paper proposes split shared-bus architecture to reduce the energy dissipation for global data exchange among a set of interconnected modules. The bus splitting problem for minimum energy is formulated as a Minimum-Exchange Bus Split problem, which is shown to be NP-complete. The problem is solved heuristically by using a max-weight matching algorithm and combinatorial search. Experimental results show that the energy saving of split-bus architecture compared to monolithic-bus architecture varies from 16% to 50%, depending on the characteristics of the data transfer among the modules and the configuration of the split-bus. The proposed split-bus architecture can be extended to multi-way split-bus architecture when large numbers of modules are to be connected.

1 Introduction

To increase the level of integration and the performance of microelectronic systems, system-on-a-chip design has been widely employed in today's designs. In such designs, communication resources are allocated to connect the on-chip modules in order to provide a means of data exchange. Two widely used communication architectures are 1) *point-to-point connection* (uni-directional) and 2) *shared-bus* (bi-directional). In addition to system-on-a-chip designs, microprocessors, digital signal processors, and embedded controllers also use these two types of interconnection architecture. This paper proposes a split shared-bus architecture (see Figure 1) to reduce the energy consumption of monolithic shared-bus architecture (see Figure 2).

In [1], a bus architecture that provides high performance while scaling across a range of chip sizes is described. The system on a chip design in which it has been implemented includes both a dedicated processor with a set of embedded system peripherals and system support logic that may be reconfigured by a user in the field. Multiple masters and slaves are provided for in the architecture and included in the dedicated portion of this chip. Designers configure additional bus slave peripherals and support functions in the programmable logic. Dedicated structures extend the bus throughout the user-configurable system logic. The bus is pipelined, uses OR gates, and has separate read and write data. The bus signals (data, addresses, and controls) are distributed throughout the user-configurable system logic (CSL) independent of the user logic. Within the CSL, switched paths are provided that can connect the bus and selector signals to some of the conventional routing signals. The bus has a complete separation of the write data paths from the read data paths. This split bus makes it possible to use both of these data paths for different transactions at the same time. The

notion of split bus used in this architecture is, however, quite different from that proposed in this paper.
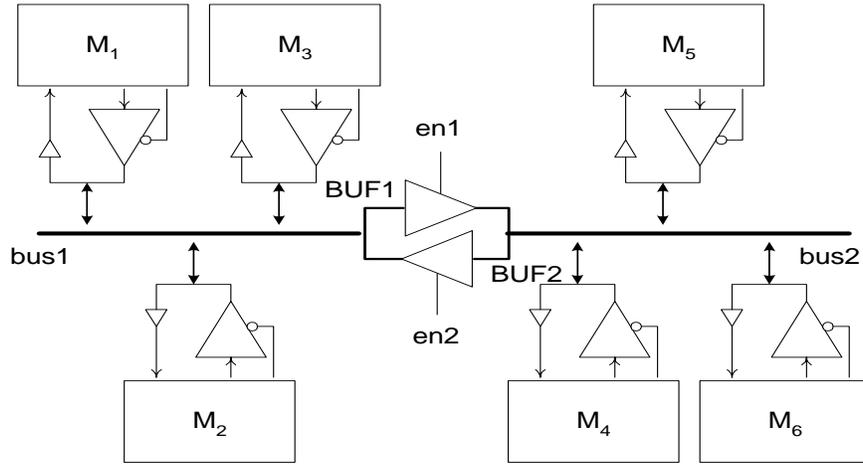
Figure 1. Split shared-bus architecture
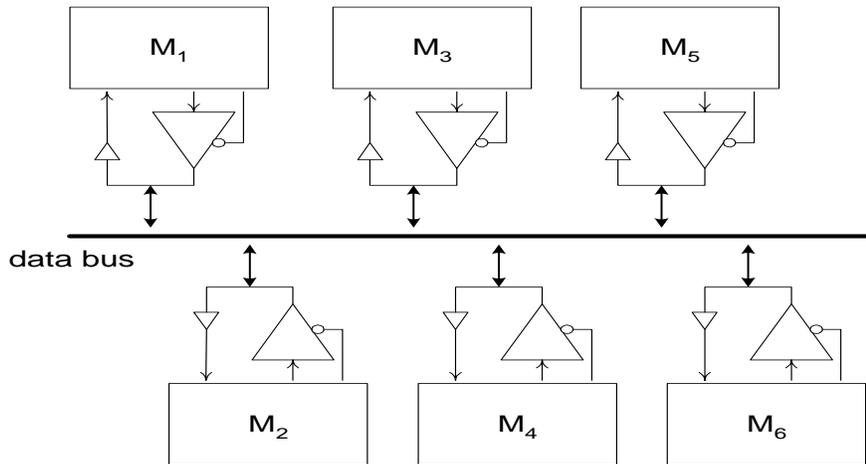
Figure 2. Monolithic shared-bus architecture

The advantages of shared-bus architecture include simple topology, low area cost, and extensibility. The disadvantages of shared-bus architecture are larger load per data-bus line, longer delay for data transfer, larger energy consumption, and lower bandwidth. Fortunately, the above disadvantages, with the exception of the lower bandwidth, may be overcome by using a low-voltage swing signaling technique [2]. In low-voltage swing architecture, the signal being transferred from a module is first converted to a low-voltage swing signal and then propagated along the shared-bus. The low-voltage swing is finally converted back into a full-swing signal at the input of the receiving module. In this way, the amount of charge on the bus will only change by $\Delta V \times C_{BUS}$, where ΔV is the voltage swing on the bus and C_{BUS} is the capacitive load of the bus. Therefore, the low-voltage swing bus achieves an energy reduction of $(V_{dd}-\Delta V) * V_{dd}$ compared to a full-swing bus. The signal delay on the bus is also reduced by $\Delta t = \dfrac{C_{BUS}(Vdd - \Delta V)}{I}$, where I is the average current of the driver.

For a detailed overview of energy minimization techniques, interested reader may refer to Ref. [3] [4]. Notice that the bus-encoding and low-voltage signaling techniques reviewed in these references can be used to reduce the energy consumption of the on-chip bus regardless of whether the bus is split or not.

The remainder of this paper is organized as follows. Section 2 describes the structure of a monolithic bus and provides expressions for the energy consumption on this bus and the average energy dissipation in the bus drivers. Section 3 presents the split-bus architecture and provides simulation-based and probabilistic energy consumption models for the split-bus architecture under given inter-module data transfer probabilities. Section 4 describes algorithms (optimal and heuristic) for splitting modules (linearly-ordered and unordered) connected to a bus into two subsets so as to minimize the average energy dissipation of the split-bus per clock cycle. Variations on the split-bus topology are shown in Section 5. Experimental results and concluding remarks are given in Sections 6 and 7, respectively.

2   Monolithic-bus Structure and Bus Drivers

Without loss of generality, a one-bit bus is considered. Results for a k-bit bus can be obtained by scaling the one-bit bus results by k. Assume that there are n modules $M_1, M_2,..., M_n$ connected to each other through a bi-directional shared-bus as shown in Figure 2. During the architectural simulation, the system is simulated for p cycles, from cycle 1 to cycle p. In each cycle i, the data with a logic value of V_i is transferred from module $M_{SRC(i)}$ to module $M_{DST(i)}$.

Assume that the receiver gate for each module is at its minimum size and its input capacitance is C_g. Furthermore, the output capacitance of the driver for each module M_i is $C_{o,i}$. C_{BUS} is calculated as follows:

$$C_{BUS} = L_{BUS} \cdot (C_u + C_c) + \sum_{i=1}^{n} (C_{o,i} + C_g)$$

where L_{BUS} is the physical length of the bus, C_u denotes the capacitance per unit length of the bus, C_c denotes the coupling capacitance per unit length of the bus due to the parallel running bus wires as well as other nearby wires on adjacent metal layers, and n is the number of modules connected to the bus.

The average energy consumption during p cycles is:

$$E = 0.5 \cdot C_{BUS} \frac{V_{dd}^2}{p} \cdot \sum_{i=1}^{p} (V_{i-1} \oplus V_i) + \sum_{i=1}^{n} E_{driver,Mi}$$

where $E_{driver,Mi}$ is the average internal energy dissipation per clock cycle of the bus driver in module M_i. Notice that the exclusive-or (EXOR) operation evaluates to 1 exactly when the value on the bus changes.
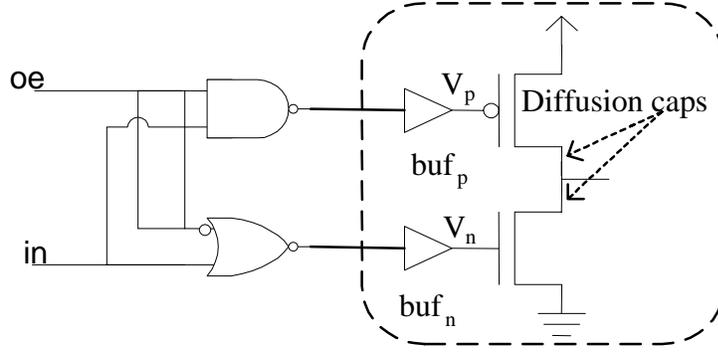
Figure 3. Circuit diagram of a tri-state bus driver

A typical tri-state driver is shown in Figure 3. Notice that $V_p = \overline{oe \cdot in}$ and $V_n = oe \cdot \overline{in}$. The switching activities for V_p and V_n are:

$$sw(V_p) = prob(oe = 1 \rightarrow 1, in = 0 \rightarrow 1) + prob(oe = 1 \rightarrow 1, in = 1 \rightarrow 0)$$
$$+ prob(oe = 1 \rightarrow 0, in = 1 \rightarrow x) + prob(oe = 0 \rightarrow 1, in = x \rightarrow 1)$$
$$sw(V_n) = prob(oe = 1 \rightarrow 1, in = 0 \rightarrow 1) + prob(oe = 1 \rightarrow 1, in = 1 \rightarrow 0)$$
$$+ prob(oe = 1 \rightarrow 0, in = 0 \rightarrow x) + prob(oe = 0 \rightarrow 1, in = x \rightarrow 0)$$

where $prob(oe = v1 \rightarrow v2, in = v3 \rightarrow v4)$ denotes the probability of $(oe, in) = (v1, v3)$ in the current cycle and $(oe, in) = (v2, v4)$ in the next clock cycle; x denotes a don't care term. If input in is not correlated with oe, the above equations can be simplified to:

$$sw(V_p) = 2 prob(in) prob(oe)[1 - prob(in) prob(oe)]$$
$$sw(V_n) = 2 prob(in = 0) prob(oe)[1 - prob(in = 0) prob(oe)]$$

where *prob(x)* and *prob(x=0)* denote the probability for *x=1* and *x=0* in a clock cycle.

The average internal energy dissipation of the driver stage per clock cycle is:

$$E_{driver} = 0.5(sw(V_p)C_{eff,bufp} + sw(V_n)C_{eff,bufn})V_{dd}^2$$

where $C_{eff,bufp}$ ($C_{eff,bufn}$) denotes the physical capacitance driven by NAND2 (NOR2).

3 The Split-bus Architecture

In a long bus line, the parasitic resistance and capacitance are quite high. For example, in Figure 2, the propagation delay from module M_1 to module M_6 is large. To improve the timing and energy consumption of the long bus, the bus can be partitioned into two bus segments as shown in Figure 1. The dual-port driver at the boundary of bus1 and bus2 relays the data from one bus to the other when such a data transfer is needed. Therefore the split-bus architecture works in the same way as a monolithic-bus. If the intrinsic delay (and energy consumption) of the dual-port driver is small

compared to the rest of the bus, which is the case for a long bus connection, then the new bus architecture will be preferable to the monolithic-bus architecture.

Advantages of bus splitting are:

- Smaller parasitic load: because the bus length is reduced, the parasitic load of each bus segment is reduced.

- Larger timing slack: due to the smaller parasitic load of the two bus parts and because smaller output capacitances from the driver outputs are added as load to any part of the split-bus, the timing slack becomes larger.

- Smaller driver size: because the timing slack is larger, the driver size can be made smaller while meeting the timing constraint.

- Lower energy consumption: since a smaller load and smaller drivers are used, the effective physical capacitance for each bus part is smaller. In the case of data being transferred within the same bus partition, the energy consumption is significantly reduced because there is no switching activity in the other bus partition.

- Lower noise problems: the parallel running buses are at the greatest risk with respect to coupling noise. Reducing the bus wire length effectively reduces the amount of capacitive coupling noise.

In Figure 1, modules M_1, M_2, and M_3 reside in the bus on the left (i.e., bus1), and modules M_4, M_5, and M_6 sit on the other side (i.e., bus2). Let *BUS1* be the set of modules in the left bus and *BUS2* denote the set of modules in the right bus. When *en1* is '1,' BUF1 will relay the data from bus1 to bus2. Similarly, BUF2 will pass the data from bus2 to bus1 when *en2* is '1.' Note that *en1* and *en2* should not be set to '1' at the same time. When both *en1* and *en2* are '0,' bus1 and bus2 are isolated from one another. In this section, we assume the driver sizes are fixed.

3.1 Assumptions and energy consumption models

It is assumed that the output diffusion capacitances of the drivers are zero and that the internal energy consumption of the drivers is negligible. Furthermore, we assume that the logic and routing overhead of the split-bus architecture, that is, the energy dissipated to generate and connect the control signals for the bus drivers that are inserted in the split-bus architecture is negligible in comparison to the energy dissipated in the bus drivers themselves (however, please see discussion at the end of Section 6).

The data being transferred by any module on the data bus is modeled as an independent random variable with an average switching activity equal to *sw*. The average energy consumption of the monolithic bus architecture is calculated as: $E1 = 0.5 \cdot sw \cdot C_{BUS} . V_{dd}^2$.

Let C_{BUS1} and C_{BUS2} denote the physical capacitance on bus1 and bus2. The average energy consumption of the split-bus architecture per clock cycle is calculated as:

$$E2 = 0.5.sw.V_{dd}^2.[C_{BUS1} \sum_{i \in BUS1} \sum_{j \in BUS1, j \neq i} xfer(M_i, M_j) + C_{BUS2} \sum_{i \in BUS2} \sum_{j \in BUS2, j \neq i} xfer(M_i, M_j)$$

$$+(C_{BUS1} + C_{BUS2}) \sum_{i \in BUS1} \sum_{j \in BUS2} (xfer(M_i, M_j) + xfer(M_j, M_i))]$$

where $xfer(M_i,M_j)$ denotes the *probability of data transfer* from module M_i to module M_j in any clock cycle. In the following examples, we set sw=0.5 and normalize $\frac{C_{BUS1}}{|BUS1|} = \frac{C_{BUS2}}{|BUS2|} = \frac{C_{BUS}}{n} = 1$, V_{dd}=1 where |BUS1| and |BUS2| denote cardinalities of the two buses. Analytical and experimental results presented in the remainder of this paper are based on the assumptions stated above and are not be applicable to cases in which these assumptions do not hold.

In the equations presented throughout this paper, we assume that there is a bus transaction in each clock cycle. In practice, the processor may be in a (possibly idle) state where there are no bus transactions. However, if there are no bus transactions, the monolithic bus and the split bus have the same exact energy dissipation (i.e., zero if we ignore the leakage current). Therefore, we can simply exclude the bus idle cycles from consideration and state that the sum of data transfer probabilities between all pairs of modules on the bus add up to 1 in each cycle in every cycle when the bus is active. More precisely, let $x(M_i,M_j)$ denote the *probability of data exchange* between module M_i and module M_j in any clock cycle when the bus is used. Then, $x(M_i,M_j) = xfer(M_i,M_j) + xfer(M_j,M_i)$ and $\sum_i \sum_{j>i} x(M_i, M_j) \equiv 1$.

***Example* 1:** Assume we have n=2k modules and $|BUS1|$=k-a, $|BUS2|$=k+a where $a \in \{0,1 \dots k\text{-}2\}$. The probability of data transfer from module M_i to module M_j in any clock cycle is $\frac{1}{2k(2k-1)}$, for i=1,…, 2k and j=1,.., 2k, $i \neq j$.

$$E1 = 0.5k$$

$$E2 = 0.25 \frac{3k^3 - k^2 + a^2(k-1)}{2k^2 - k}$$

The energy saving of the split-bus over the monolithic bus can be calculated by:

$$\frac{E1 - E2}{E1} = 0.5 \frac{k^3 - k^2 - a^2(k-1)}{2k^3 - k^2}$$

The energy saving is maximized when a=0.

For the case of k=2 and a=0, energy saving is 16%. When $k \to \infty$ and a=0, the energy saving is 25%.

If a is set to k, which is the case in a monolithic bus, then the energy saving is 0.

***Example* 2:** Assume that there are four modules connected to the bus. The probability of data exchange between module M_i and module M_j in any active clock cycle, $x(M_i,M_j)$, is specified by the label of the edge (M_i,M_j) in Figure 4.
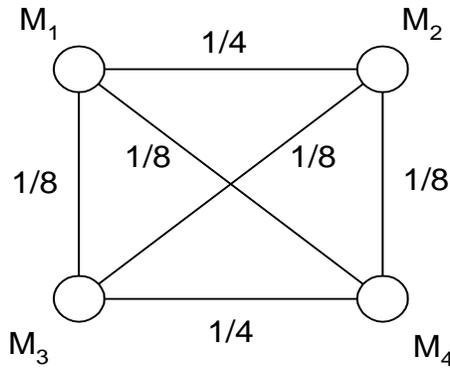
Figure 4. Data exchange probabilities for Example 2

The energy consumption for various architectures is summarized in the following table:

Architecture	Energy
$BUS=\{M_1,M_2,M_3,M_4\}$	1
$BUS1=\{M_1,M_2\}$ $BUS2=\{M_3,M_4\}$	0.75
$BUS1=\{M_1,M_3\}$ $BUS2=\{M_2,M_4\}$	0.875
$BUS1=\{M_1,M_4\}$ $BUS2=\{M_2,M_3\}$	0.875

The bus partitioning solution with $BUS1=\{M_1,M_2\},BUS2=\{M_3,M_4\}$ consumes the least energy because more data transfers are performed within each part.
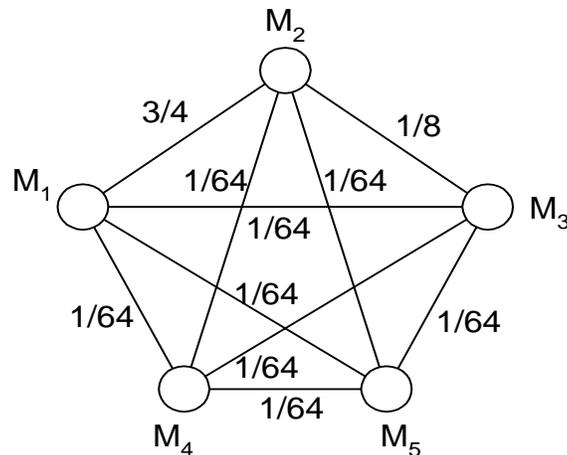
Figure 5. Data exchange probabilities for Example 3

Example 3: For the five-module configuration shown in Figure 5, the energy consumption for several configurations are listed below:

Architecture	Energy
$BUS=\{M_1,M_2,M_3,M_4,M_5\}$	1.25
$BUS1=\{M_1,M_2\}$ $BUS2=\{M_3,M_4,M_5\}$	0.66
$BUS1=\{M_1,M_2,M_3\}$ $BUS2=\{M_4,M_5\}$	0.79
$BUS1=\{M_2,M_3\}$ $BUS2=\{M_1,M_4,M_5\}$	1.13

The second bus-splitting configuration has the lowest energy consumption, achieving a 47% reduction in the energy consumption when compared to the monolithic-bus architecture. Note that although edge (M_2, M_3) has a weight of 1/8, which is the second largest value in this example, adding M_3 to $BUS1=\{M_1,M_2\}$ increases C_{BUS1} and hence results in higher energy dissipation.

3.2 A cycle-accurate energy consumption model for the split-bus architecture

Similar to the case of the monolithic bus, the physical capacitance on bus1 and bus2 can be calculated as:

$$C_{BUS1} = L_{BUS1} \cdot C_u + C_{c,1} + \sum_{i \in BUS1} (C_{o,i} + C_g) + C_{o,BUF1} + C_{in,BUF2}$$

$$C_{BUS2} = L_{BUS2} \cdot C_u + C_{c,2} + \sum_{i \in BUS2} (C_{o,i} + C_g) + C_{o,BUF2} + C_{in,BUF1}$$

where L_{BUS1} and L_{BUS2} are the physical lengths of bus1 and bus2; $C_{c,1}$ and $C_{c,2}$ are the coupling capacitances of bus1 and bus2; $C_{o,BUF1}$ and $C_{o,BUF2}$ are the output capacitances of BUF1 and BUF2; and $C_{in,BUF1}$ and $C_{in,BUF2}$ are the input capacitances of BUF1 and BUF2. Here we assume that the wire widths of both buses are the same. Again, we assume the minimum gate size for the receiver of each module.

The logic values on bus1 and bus2 in each clock cycle *i* are calculated as follows:

$V_{BUS1,i} = V_{BUS1,i-1}$ *if $M_{SRC(i)} \notin BUS1$ and $M_{DST(i)} \notin BUS1$*

      $= V_i$ *otherwise*

$V_{BUS2,i} = V_{BUS2,i-1}$ *if $M_{SRC(i)} \notin BUS2$ and $M_{DST(i)} \notin BUS2$*

      $= V_i$ *otherwise*

where V_i denotes the logic value being transferred in clock cycle i.

The average energy consumption of the split-bus architecture is calculated as:

$$E = E_{BUS1} + E_{BUS2} + E_{driver}$$

$$= 0.5C_{BUS1}\sum_{i=1}^{p}(V_{BUS1,i-1} \oplus V_{BUS1,i})\frac{V_{dd}^2}{p} + 0.5C_{BUS2}\sum_{i=1}^{p}(V_{BUS2,i-1} \oplus V_{BUS2,i})\frac{V_{dd}^2}{p}$$

$$+ \sum_{i=1}^{n}E_{driver,Mi} + E_{driver,BUF1} + E_{driver,BUF2}$$

where $E_{driver,Mi}$ and $E_{driver,BUFx}$ are the average energy consumptions per clock cycle for module M_i and buffer x and are calculated by the equations in Section 2. Note that p is the number of simulated cycles.

3.3   A probabilistic energy consumption model for the split-bus architecture

In general, p must be very large to become representative of real application data. To speed up the energy consumption calculation, a probabilistic model can be used. Note that the model is only exact under the assumption that the application data is stationary [5].

Assume that the data being transferred from each module can be modeled as a time-invariant random process with probability *prob(M_i)* for the data value to be '1.' Furthermore, assume that the data transfer at clock i $(M_{SRC(i)},M_{DST(i)})$ is not correlated to the data transfer pair $(M_{SRC(i+1)}, M_{DST(i+1)})$ at clock $i+1$.

Let *xfer(BUS1, BUS2)* denote the probability of bus1 transferring data to bus2 in any clock cycle. It is calculated as:

$$xfer(BUS1, BUS2) = \sum_{i \in BUS1} \sum_{j \in BUS2} xfer(M_i, M_j)$$

xfer(BUS2,BUS1) is calculated similarly.

xfer(BUS1), which denotes the probability of data transfers occurring on bus1, is calculated as:

$$xfer(BUS1) = xfer(BUS2, BUS1) + \sum_{i \in BUS1} \sum_{j \in BUS1 \cup BUS2, j \neq i} xfer(M_i, M_j)$$

xfer (BUS2) is defined similarly.

prob(BUS1), which denotes the probability for bus1 to assume a logic value '1' in a clock cycle, is calculated as:

$$prob(BUS1) = \{ \sum_{i \in BUS1} \sum_{j \in BUS1 \cup BUS2, j \neq i} xfer(M_i, M_j)prob(M_i) + \sum_{i \in BUS2} \sum_{j \in BUS1} xfer(M_i, M_j)prob(M_j) \} / xfer(BUS1)$$

prob(BUS2) is defined similarly.

The switching activities of bus1 and bus2 (assuming temporal independence of data values on the bus) are:

$$sw(BUS1) = 2prob(BUS1)[1 - prob(BUS1)]xfer(BUS1)$$
$$sw(BUS2) = 2prob(BUS2)[1 - prob(BUS2)]xfer(BUS2)$$

Therefore, the average energy consumption per clock cycle of the split-bus architecture is calculated as:

$$E = 0.5(C_{BUS1}sw(BUS1) + C_{BUS2}sw(BUS2))V_{dd}^2 + \sum_{i=1}^{n} E_{driver,Mi} + E_{driver,BUF1} + E_{driver,BUF2}$$

where $E_{driver,x}$ can be calculated by the equations in Section 2.

4 Bus Splitting for Low Energy

The minimum energy bus-splitting problem is defined as partitioning the modules into two equal-size sets such that the average energy dissipation per clock cycle of the split-bus architecture is at a minimum.

Consider that bus splitting is performed after the modules on the bus have been physically placed and the bus wires have been routed. During this design phase, the linear ordering of the modules on the bus is already known; therefore the only degree of freedom is in selecting a bus segment, from 2 to *n*-2, to place the dual-port driver. Let $sw_{BUS1}(i)$ and *E(i)* denote the switching activity of the data on bus1 and energy dissipation on bus1 with the dual-port driver positioned at bus segment *i*. The symbols with subscript '*BUS2*' are also defined similarly.

4.1 An optimal algorithm for linearly ordered modules

When the modules are linearly ordered, the following simple, yet efficient, algorithm can easily solve the problem:

1. Calculate the *sw(BUS1,i)* and *sw(BUS2,i)* for buffer position at segment *i, i=2...n-2*
2. Calculate *E(i)* for buffer position at segment *i, i=2...n-2*
3. Find the minimum *E(i)*.

The computational complexity of the algorithm is dominated by that of the first step, which is $O(p \cdot n)$ where n is the number of modules and p is the number of simulated cycles. The algorithm is clearly optimal.

4.2 Problem complexity for unordered modules

When the bus splitting is performed before the system-level floor planning is completed, there is a freedom to rearrange the order of the modules to minimize the average energy consumption per clock cycle. This problem is difficult to formalize in its general form because of the variable capacitance coefficients and the effect of buffers 1 and 2 on the average energy dissipation. Therefore, a simpler problem is formulated here that intuitively yields a low-energy solution. The decision version [6] of this new problem is stated next.

Recall that $x(M_i, M_j)$ is the probability of module M_i having a data exchange with module M_j in a clock cycle and that $x(M_i, M_j) = xfer(M_i, M_j) + xfer(M_j, M_i)$.

Definition: *Minimum-Exchange Bus Split*

Instance: A collection M of modules with data exchange probabilities $x(M_i, M_j)$ s.t.

$$\sum_i \sum_{j>i} x(M_i, M_j) \equiv 1 \text{ and a positive real value } T.$$

Question: Is there a partition of M into two equal-size disjoint sets such that the sum of data transfer probabilities between modules in one set and those in the other set is no more than T?

Theorem: *The Minimum-Exchange Bus Split problem is NP-complete.*

Proof: We show that the 'Minimum Cut into Bounded Sets' (MCBS) problem [6] is polynomial-time reducible to the 'Minimum-Exchange Bus Split' (MEBS) problem.

Let $G(V, E)$, with specified vertices s and t and weight $w(e) \in Z^+$ for each $e \in E$, be any instance of MCBS. We must construct a partition of V into equal-size disjoint sets V_1 and V_2 such that $s \in V_1$ and $t \in V_2$ and such that the sum of the weights of the edges from E that have one endpoint in V_1 and one endpoint in V_2 is no more than some integer K. The MCBS problem remains NP-hard for $w(e)=1$ for all $e \in E$.

Let W_{sum} denote the summation of all edge values in $G(V, E)$ and $|V| = n$. Without loss of generality, we assume that n is an even integer. For each vertex v_i in V, a module M_i is created. We also denote the two modules corresponding to s and t as M_1 and M_n, respectively. For each edge $e=(V_i, V_j)$ with edge weight $w_{i,j}$ an edge from module M_i to module M_j is added. The data exchange probabilities between M_i and M_j are defined as:

$$x(M_i, M_j) = \frac{1}{n(n-1)} \cdot (\frac{w_{i,j}}{W_{sum}} + 1) \; \forall i \; and \; j > i \; (note: \; i \neq 1 \; or \; j \neq n)$$

$$x(M_1, M_n) = \frac{1}{n(n-1)} \cdot \frac{w_{1,n}}{W_{sum}}$$

It can be easily verified that $\sum_i \sum_{j>i} x(M_i, M_j) \equiv 1$. Notice also that the data exchange probabilities are defined so that M_1 and M_n are forced to different sets. Next we must define the cut size for the MEBS problem so as to match the cut size for the MCBS problem. Consider that the cut in M creates two sets with sizes $n/2$. The number of edges between the two parts in M is $n^2/4$, one of which is the edge between M_1 and M_n. Assuming that the corresponding cut in $G(V,E)$ has a cut size of no more than K, the cut size in M is bounded from above by:

$$T = \frac{1}{n(n-1)} \cdot \left\{ \frac{K}{W_{sum}} + \frac{n^2}{4} - 1 \right\}$$

This construction can obviously be accomplished in polynomial time. It is easy to see that graph G can be partitioned into two equal-size sets with cut size no more than K if and only if module set M can be split into two equal-size parts with a data exchange rate no higher than T. ∎

4.3   A heuristic algorithm for unordered modules

Because bus splitting with an unknown module order is an NP-hard problem, an exhaustive search procedure is used for small values of n. The number of all possible splittings (with a subset size > 0) is 2^{n-1}-1. In the experimental results, by using the probabilistic energy consumption model, the exhaustive search for $n=30$ was completed in less than 3 minutes on an 800-Mhz Pentium-III machine. When n is large, a module-clustering step is performed to make the effective n (i.e., the number of clusters) less than or equal to some predefined value, e.g., 30. Minimizing the inter-cluster data exchanges while maintaining a size constraint on each cluster can generate the desired clustering solution. A recursive **max-weight matching algorithm** [7] performs the clustering step. Next, all possible ways of bus splitting are exhaustively enumerated for the set of module clusters.

5   Bus Topology Variation

Instead of aligning all the modules horizontally, it may be beneficial to resort to other connection topologies, when allowed, to improve the timing or energy consumption. A T-shaped configuration is suitable for unbalanced partitioning while an H-shaped configuration is suitable for balanced partitioning. Note that both configurations have better delay characteristics than a horizontally aligned configuration.
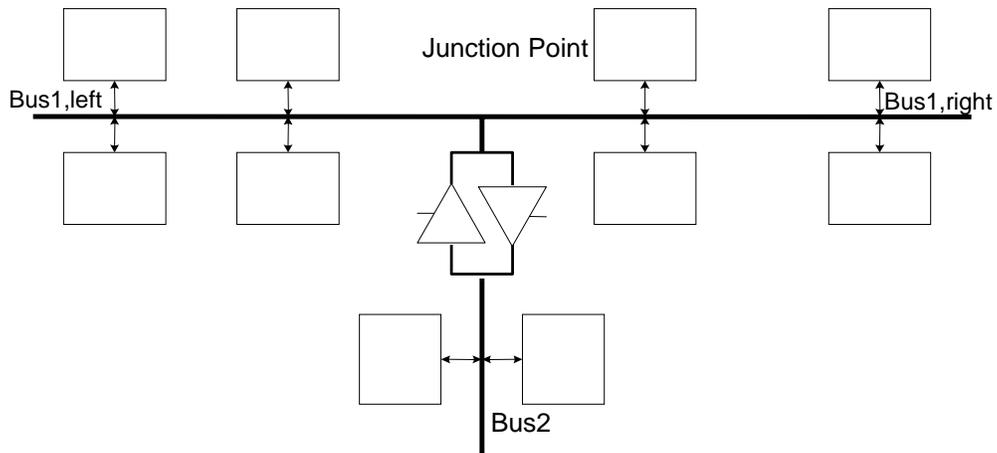
Figure 6. The T-shaped bus structure
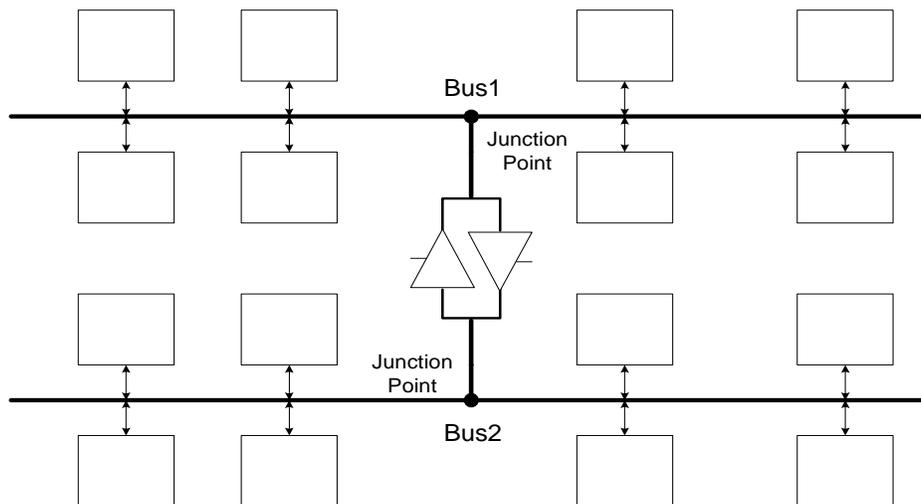
Figure 7. The H-shaped bus structure

6 Experimental Results

There are no existing benchmarks to use for this problem. Therefore, it is necessary to generate new test benches. In the experimental setup, the assumptions discussed in Section 3.1 are adopted. In addition, the data exchange frequency between any two modules M_i and M_j is randomly weighted by an integer between 0 and 9 and follows one of four probability distributions (i.e., normal, exponential, uniform and delta function) in a randomly generated test case. The height of each bar in **Error!**

Reference source not found. shows that the (relative) probability of the data exchange frequency between a pair of modules is equal to the x-axis value.
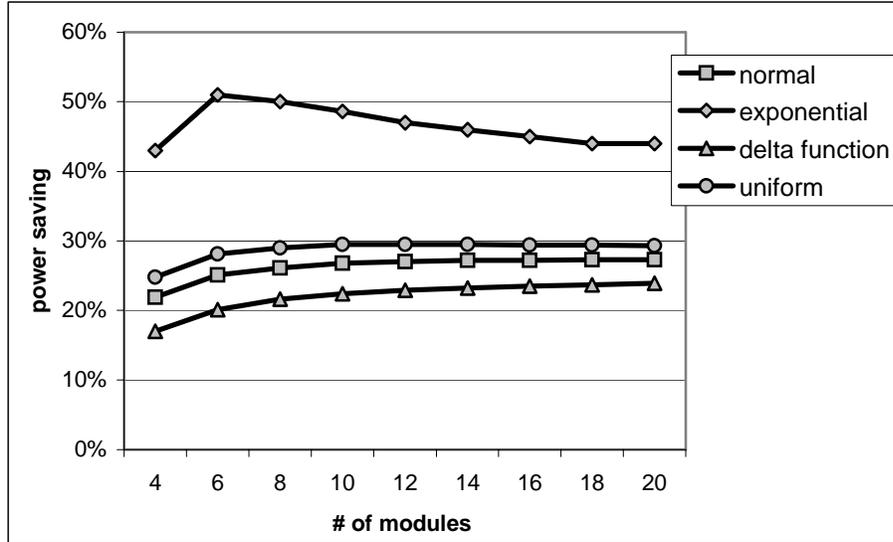
Figure 8. Energy saving for a number of data exchange distributions

Each point in Figure 8 represents the average energy saving of the split-bus over the monolithic-bus architecture, given that *k* modules *(k=4...20)* are connected to the bus, for 500 randomly generated test cases in which the transfer frequencies between pair of modules follow a given distribution *dist*. In the following discussion, each point in Figure 8 is referred to as (*k*, *dist*), e.g., (4, normal).

The results of the simulation show that the test cases with exponential distributions have the largest average energy saving while the test cases with delta function distributions have the smallest energy saving. This is because test cases with exponential distribution have fewer high-frequency transfers between modules, and therefore it becomes easier to keep the modules with high-frequency transfers within one part of the split-bus. On the other hand, the delta function distribution has no variation in transfer frequencies, and, therefore, it provides the smallest opportunity for optimization.

One important observation is that an anomaly occurs at point (6, exponential), which has the highest average energy saving compared to other points of the same distribution (see Figure 8). The reason for this is that (6, exponential) has a higher energy saving opportunity compared to (4, exponential) due to the fact that the unbalanced bus partitions (|*BUS1*|=2,|*BUS2*|=4) or (|*BUS1*|=4,|*BUS2*|=2) can result in larger energy saving as was illustrated in Example 3. For points (*k*, exponential) where k > 6, it is harder to achieve energy savings because modules are more likely to be tightly coupled. For distributions other than the exponential distribution, the frequency distributions have a much lower variance than the exponential distribution. Therefore, the results follow the trend predicted by Example 1 in Section 3.1.

We can derive the *critical length, L_{crit},* of an on-chip bus below which the energy savings of the bus splitting technique are offset by the energy dissipation overhead of the additional logic to generate the

enable signals, to route them to the driver buffers and of the input and internal capacitances of the buffers.

Example calculation. To derive a specific lower bound value, yet illustrate the general procedure, let us consider a 1-bit bus, which is implemented in Metal 3 or Metal 4, is 0.6μ wide, has 0.6μ spacing from its neighboring lines, and runs for a length of L mm on a 0.15 μ industrial technology process (i.e., the TSMC 0.15 μ, logic 1P7M Salicide 1.5V process), the total capacitance of such a bit line is *0.18 fF/μ *L mm=0.18 L pF*. The gate input capacitance of a typical gate in this process technology is about *0.9μ *0.3μ *10 fF/μ^2 = 2.7 fF*. Assuming a fanout of three (FO3) load per logic gate and furthermore assuming that 20 gates are required to implement the additional logic that produces *en1* and *en2*, the logic overhead is calculated as *20*3*2.7=162 fF*. The routing capacitance (assuming minimum-width local interconnect of *2 mm* length per enable line) is *2*0.05* fF/μ *2000 μ = 200 fF* for both enable signals. The input and internal capacitances of the driver buffers contribute another *120 fF*. Therefore, the total cost is *482 fF*. We leave a safety margin of 25%, ending up with *602 fF*. From this approximate calculation, it can be concluded that, in this process technology, only for a global bus that is longer than L_{crit} *= 602/.18/1000 mm= 3.34 mm*, it is worthwhile to do bus splitting.

7 Conclusions

Split-bus architecture was proposed to improve the speed and energy dissipation for global data exchange among a set of modules. The energy model for split-bus architecture was presented, and the bus splitting problem was solved by combinatorial techniques. Experimental results showed that the energy saving of the split-bus compared to the monolithic-bus architecture varies from 16% to 50%, depending on the characteristics of the data transfer among the modules and the configuration of the split-bus. T-shaped bus and H-shaped bus structures were proposed to further improve the bus performance. The proposed split-bus architecture can be extended to a multi-way split-bus architecture when large numbers of modules are to be connected.

References

[1]   S. Winegarden, "Bus architecture of a system on a chip with user-configurable system logic ," *IEEE Journal of Solid State Circuits*, Vol. 35, No. 3, pp. 425 –433, March 2000.

[2]   Y. Nakagome, K. Itoh, M. Isoda, K. Takeuchi, and M. Aoki, "Sub-1-V swing internal bus architecture for future low-power ULSIs," *IEEE Journal of Solid–State Circuits*, Vol. 28, No. 4, pp. 414-419, Apr. 1993.

[3]   J. Rabaey and M. Pedram, *Low power design methodologies*, Kluwer Academic Publishers, Norwell, Massachussetts, 1996.

[4]   E. Macii, M. Pedram, and F. Somenzi, "High-level power modeling, estimation and optimization," *IEEE Trans. on Computer Aided Design,* Vol. 17. No. 11, pp. 1061-1079, Nov. 1998.

[5]   A. Leon-Garcia, *Probability and Random Processes for Electrical Engineering*, Second Edition. Reading: Addison Wesley, 1993.

[6]   M. R. Garey and D. S. Johnson, *Computer and Intractability, A Guide to the Theory of NP-Completeness.* New York: W.H. Freeman and Company, 1979.

[7]   T. Lengauer, Combinatorial Algorithms for Integrated Circuit Layout. New York: John Wiley & Sons, 1990.