A Power Reduction Scheme for Data Buses by Dynamic Detection of Active Bits

Muroyama, Masanori
Department of Computer Science and Communication Engineering Kyushu University

Hyodo, Akihiko
Department of Computer Science and Communication Engineering Kyushu University

Okuma, Takanori
Department of Computer Science and Communication Engineering Kyushu University

Yasuura, Hiroto
Department of Computer Science and Communication Engineering Kyushu University

https://doi.org/10.15017/6117

出版情報：IEICE Transactions on Electronics. E87-C (4), pp.598-605, 2004-04. IEICE
バージョン：
権利関係：

PAPER Special Issue on Low-Power System LSI, IP and Related Technologies

A Power Reduction Scheme for Data Buses by Dynamic Detection of Active Bits

Masanori MUROYAMA[†a)], **Akihiko HYODO**[†b)], **Takanori OKUMA**[†c)], ***Nonmembers***, ***and*** **Hiroto YASUURA**[†d)], ***Member***

SUMMARY To transfer a small number, we inherently need a small number of bits. However all bit lines on a data bus change their status and redundant power is consumed. To reduce the redundant power consumption, we introduce a concept named *active bit*. In this paper, we propose a power reduction scheme for data buses using active bits. Suppressing switching activity of inactive bits, we can reduce redundant power consumption.

We propose various power reduction techniques using active bits and the implementation methods. Experimental results illustrate up to 54.2% switching activity reduction.

key words: *active bit, low power, datapath, data bus, dynamic*

1. Introduction

Recently, many portable devices have been made by low power CMOS circuit design. In these devices low power is an important and necessary issue for achieving long battery life, low cost for package, high reliability and so on. A typical digital system such as a processor usually has a datapath part and a controller part. The datapath is the largest power consuming component in a system, because buses, memories and arithmetic circuits consume a great deal of power.

Datapath width, the bit width of buses and operational units such as arithmetic circuits in a system, is an important design parameter for power optimization. The datapath width can be optimized for power minimization by means of analyzing required bit-width of variables [1]. The technique is a static approach, because the optimization is done in design phase. In this paper, we propose a dynamic approach applicable in run time.

To transfer a small number, although we inherently need a small number of bits, all bit lines on a data bus change their status and redundant power is consumed. We define *active bits* as necessary bits of the data for computation, and *inactive bits* as remaining bits except active bits in the data. For example, an unsigned decimal number 1000 needs 10 bits in binary representation. In a 32 bits system, lower 10 bits are active bits and upper 22 bits are inactive ones.

Some techniques using similar concepts are studied. Okuma, Cao, Muroyama, and Yasuura [2] presented a memory power reduction technique using active bits. They achieved significant energy reduction compared to the monolithic memory, 52.2% and 84.2% for JPEG and MPEG-2, respectively. An arithmetic circuit power reduction technique is also introduced by Brooks and Martonosi [3]. They proposed a method to reduce power in the integer execution unit of a processor with aggressive clock gating inactive bits. They achieved 45% - 60% power reduction for the integer unit.

Interconnect wires account for a significant fraction of the energy consumed in an integrated circuit [4], and this fraction is only expected to grow in future. In fact, it is projected that, as technology scales to the nanometer regime, the delay and energy consumption of global interconnect structures will prove to be a major bottleneck for SoC design [5], [6]. Since RC delays of long bus lines are significant compared to gate delays and clock times, huge amounts of repeaters (the number of repeaters is about 20 in 0.18 μm process) are usually inserted to the bus lines. Therefore power consumption of long bus lines is quite large [6].

The power consumption in a CMOS circuit can be classified as static power consumption and dynamic power consumption. The dynamic power consumption is affected by both base capacitances and wire-to-wire capacitances. In deep sub-micron designs, leakage and coupling effects between close bus lines must be addressed. Techniques considering these effects have been proposed [7], [8]. In this paper, coupling effect and static power is ignored and only dynamic power is considered. In our future work we will study these effects on power consumption in deep submicron era.

The dynamic power consumed can be defined as follows:

$$P_{chip} \propto V_{DD}^2 \cdot f \cdot \sum_{i=1}^{N} \cdot CL_i \cdot \alpha_i$$

where P_{chip}:total power consumption of a chip, N:total number of nodes of the circuit on the chip, f:the clock frequency, CL_i: the load capacitance at

[†]The authors are with the Department of Computer Science and Communication Engineering at Kyushu University, 6-1 Kasuga-Koen, Kasuga-shi, Fukuoka, 816-8580 Japan

a) E-mail: muroyama@c.csce.kyushu-u.ac.jp
b) E-mail: akihiko@c.csce.kyushu-u.ac.jp
c) E-mail: okuma@c.csce.kyushu-u.ac.jp
d) E-mail: yasuura@c.csce.kyushu-u.ac.jp

node i, V_{DD}:the power supply voltage and α_i:the activity factor at node i. Various low power bus encoding techniques considering these parameters have been proposed. Asada et al. summarised low power bus techniques [9] such as a signal swing reduction technique (by control of V_{DD}) in bus lines, effective capacitance (CL_i) reduction techniques by bus partition, and a signal transition density α_i reduction technique by encoding.

Lowering the activity factor (α_i) is a very promising way of decreasing the power consumption of buses. Therefore we now focus on reduction of α_i. Several signal coding schemes have been already proposed to minimize transition activity on buses. When statistical properties are unknown a priori, the bus-invert technique [10] and the on-line adaptive scheme [11] can be applied to coding randomly distributed signals. The concepts of these methods are not available in a whole system. In address buses, highly correlated data patterns exhibit a spatio-temporal locality, so Panda and Dutt [12] exploited the characteristics for energy reduction. Similar approaches are proposed using Gray code addressing [13], the T0 method [14], and the working-zone coding [15]. These methods are not efficient for data buses on which distribution of the data change dynamically. In this paper, we propose a power reduction scheme for data buses, on which characteristics of the data distribution are often changed and hard to expect. In our scheme, the active bits are detected on the fly, and suppressing the wasteful switching activity of the inactive bits.

This paper is structured as follows: Section 2 introduces a concept of active bits. In section 3 we propose low power bus techniques using active bits, and show the power-optimized implementation of the encoder-decoder system. Experimental results are given in section 4. We conclude in section 5.

2. Active Bits

There are many representations for data; an unsigned integer form, a signed integer representation, a fixed point digit one and a floating point digit one. We can define active bits for each representation. In this paper we assume that all data are represented as the unsigned and signed integers.

Now we define inactive bits and active bits. For the unsigned integers, inactive bits are continuous string of 0's from the most significant bit. The remaining part is defined as active bits (Fig.1(a)). For the signed integers, sign extension is also inactive bits (Fig.1(b)).

The length of active bits is called as active bitwidth. In particular, we make active bitwidth zero when value of data is 0 (all bits are zero). Figure 2 shows an example of active bits and active bitwidth in data sequences in the case of unsigned integer representation. The underlines in Fig.2(b) indicate active bits. The remaining bits after deleting the continuous 0's in Fig.2(b) and the length of active bits are active bits and active bitwidth in Fig.2(c), respectively.

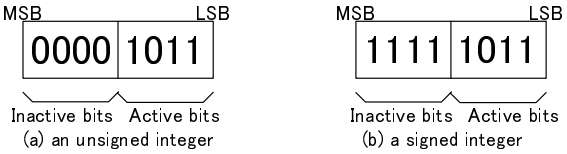


Fig. 1 An example of active bits

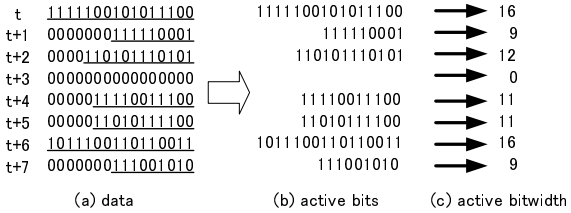


Fig. 2 An example of active bits in data sequences(unsigned integers)

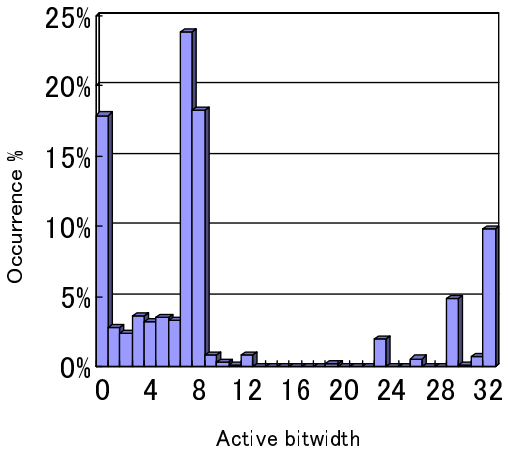


Fig. 3 Active bitwidth on 32-bit data bus (app.: mpeg2play,data size: 245k byte)

Figure 3 shows an example of occurrence each active bitwidth in a 32-bit datapath width system. An mpeg2play program with a 245k byte picture is adopted as a sample application. The mpeg2play is a player for MPEG-1 and MPEG-2 video bitstreams. It is based on mpeg2decode by the MPEG Software Simulation Group. As shown in Fig.3, 80% of data on the data bus are 8 or less active bitwidth. When active bits only are activate on the bus, the bus activity is 30.0%.

At the point of compilation, ways of access to memories are determined by instructions. There are three access types; word access (per 32 bits), half-word access (per 16 bits), byte access (per 8 bits). 28%, 27% and 45% of all access is word, half-word and byte access respectively. If the bus is activated based on access

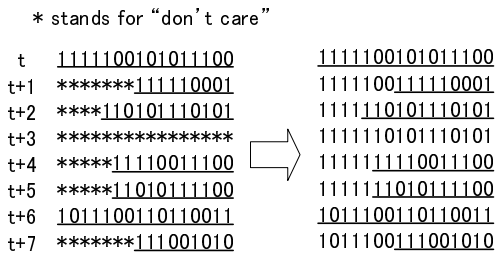


Fig. 4 An example of encoding

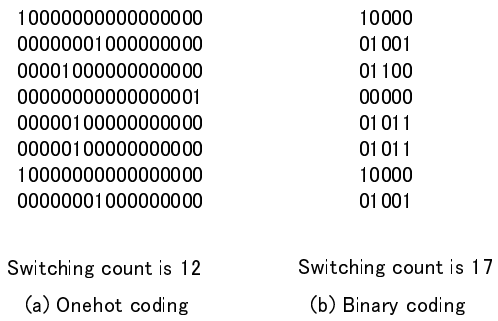


Fig. 5 An example of extra bits

unit, the bus activity is 52.8%. In this case, the part including inactive bits is also activated. Therefore it is important to detect active bits and inactive bits dynamically.

3. Detection and Coding Mechanisms

3.1 Overview

All information is included in active bits. This means that values of inactive bits can be ignored. Decreasing α_i, that is, increasing unchanged signal values, induce power reduction. Therefore it is possible to reduce power consumption by using a technique in which signal values of inactive bits at time t hold previous values on the bus at time t-1. Figure 4 presents the result of applying the technique to the example of Fig.2. The underlined sequences of bits and don't care bits indicate active bits and inactive bits, respectively. Total switching count is 51 in the original data sequences, while total switching count is 29 in the encoded data sequences.

When active bitwidth of previous data is large and active bitwidth of current data is small (in other words, the difference of active bitwidth of two consecutive data is large), the switching can be reduced more effectively. In the case of mpeg2play program with the data streaming in Fig.3, the difference of active bitwidths of two consecutive data is often large. In consequent, the proposed approach can be effective for low correlated data patterns.

Extra bits are required when a receiver finds active bits in the data. We consider two techniques to indicate active bitwidth: *onehot coding* and *binary coding* (Fig.5). Onehot coding is one by which a bit is used to indicate a state (an active bitwidth). Using binary coding, whole states are encoded in binary form.

In an N-bit bus, there are N states to indicate active bitwidth. The number of extra bits in case of onehot coding is N+1. Binary coding requires $\lfloor \log_2 (N+1) \rfloor$ extra bits. In the both cases, a state of active bitwidth zero should be considered. Next, we introduce extra bits reduction techniques.

3.2 Granularity Control in Active Bitwidth

Increase of extra bits causes increase of area complexity and power consumption of extra bits. Hence it is important to reduce extra bits. It is effective to reduce the number of states to indicate active bits. Firstly, we inquire into the relation between the number of states for active bits detection and switching count of extra bits.

We encode γ states by using onehot and binary coding. We assume that γ is power of 2 $(= 1, 2, 4, 8, ...)$ and the probability of occurrence of each state is equal.

Binary coding requires $log_2\gamma$ bits. The probability of occurrence of 1 (or 0) of each bit is 1/2, because the probability of occurrence of each state is equal. The probability of switching of a bit is 1/2. We define E_{bin} as an expected value of switching count of extra bits by using binary coding. In the case, E_{bin} is $(log_2\gamma)/2$.

Onehot coding requires γ bits. The probability of occurrence of 1 of each bit is $1/\gamma$. The probability of of switching of a bit is $(2\gamma - 2)/\gamma^2$. We define E_{one} as an expected value of switching count of extra bits by using onehot coding. In the case, E_{one} is $(2\gamma - 2)/\gamma$.

Assume $\gamma = 16$, for binary coding the number of extra bits is 4 and $E_{bin} = 2.00$. For onehot coding, the number of extra bits is 16 and $E_{one} = 1.88$. In the case of $\gamma = 8$, for binary coding extra bits is 3 and $E_{bin} = 1.50$. For onehot coding the number of extra bits is 8 and $E_{one} = 1.75$. For low power, binary coding is superior to onehot coding when the number of states is 8 or less (remember γ is power of 2). Onehot coding for extra bits is better than binary coding when the number of states is over 8. In all cases, the number of extra bits for onehot coding are larger than binary ones.

Secondly, we introduce a technique for reducing the number of states. A bus is divided into multiple parts. After division, each partition is called a segment. We define variables as follows:

- N_{seg}: the number of segments
- b_i: the bitwidth of segment i $\quad (b_i \geq 1)$

The variables must satisfy the following formula.

$$\sum_{i=1}^{N_{seg}} b_i = N$$

where N is the datapath width (equal to data bus width). In this subsection, we consider that all b_i are equal to N/N_{seg} (subsection 3.4 introduces a case that different bitwidth for each segment is permitted). We define an active segment as a segment including one or more active bit(s). A segment not including active bits is called as an inactive segment. The number of active segments for each data is defined as active segmentwidth. To simplify discuss, the representation of data is the unsigned one only.

The number of active segments plus one (one bit is used for active segmentwidth 0) is equal to the number of states (γ) to be encoded by using onehot and binary coding. If N_{seg} is small (small states), the small number of extra bits and small switching count of extra bits is achieved generally. To reduce the number of states accomplishes reduction of switching count and the number of extra bits. However it may reduce the advantage of switching count reduction on an original data bus, because inactive bits in active segments may increase. The effectiveness of power saving of the original data bus by using the granularity control depends on characteristics of each application. To analysis this feature is our future work.

3.3 Embedding Approach

This paragraph introduces a coding using active bits without extra bits. The reason why extra bits are not used is that the information of active bitwidth is embedded in the data. This encoding consists of three stages. At first stage, in active bits the current values on the bus are held. At second stage, in the most significant active bit the current value on the bus is inverted. Finally in remaining active bits the values of the original data are used.

In a reciever comparing previous and current bits from the most significant bit to the least significant bit between two data, a first bit which the two values of two bits are different indicates a start position of active bits. We call this technique an *embedding* approach. By using the number of the same bits from the most significant bit between two data (we define as s), active bitwidth can be obtained (N-s).

Figure 6 shows an example of the embedding approach. In the Fig., underline and square indicate active bits and start position of the most significant of active bits, respectively. Total switching count is 37 in the data sequences. It should be noted that the decoder cost to hold previous values on the bus is large (more detail in section 3.5).

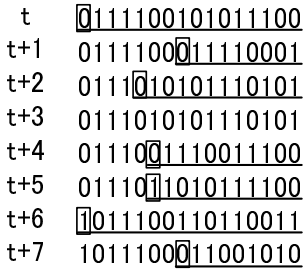


Fig. 6 An example of embedding approach

3.4 Application-Specific Granularity Control in Active Bitwidth

This section presents a technique that the detection granularity of active bits is tailored to a particular application. It means that b_i can not be constant. Initially, we define the probability of appearances of active bitwidth i as $P_{act}(i)$. We assume that the probability of each active bitwidth is given. This application-specific information can be provided from simulation in advance. In the final, a bus is partitioned into M segments with optimized b_i.

In initial condition, we consider finest granularity detection that $\forall i, b_i = 1$. Next we combine two segments, which are continuous ones. Combining segments makes the number of segments and extra bits decrease. We don't combine three or more segments in the single operation because of avoiding computational complexity (this approach is heuristic approach). In this paper, an onehot coding technique is adopted to encode active segmentwidth. Application-specific detection granularity control by using a binary one is future work.

We deal with two continuous segments; a lower segment is segment i, an upper segment is segment i+1. To combine the segments makes the sum of switching count changing. The increasing and decreasing of switching in the original bus and extra bits are considered separately. We define the expected value of the increase and decrease of the switching count in the original and extra parts as $\Delta E_{bus}, \Delta E_{extra}$, respectively. In the original bus, switching count increase in the only one case, when the segment i is active segment and the segment i+1 is inactive segment, respectively. ΔE_{bus} is defined as follows:

$$\Delta E_{bus} = (b_{i+1}/2) \cdot P_{act}(i) \cdot PT$$

where PT is the sum of data access on the bus.

The number of increase and decrease of switching in extra bits is defined as follows:

$$\Delta E_{extra} = -2P_{act}(i) \cdot P_{act}(i+1) \cdot PT$$

Therefore the sum of increase of switching count ΔE_{total} is

```
Input: P_act(i), for all i
Output: optimized segment width, for all segments
Procedure Application-Specific Detection
   while until the number of segments is M
      forall pairs of continuous two segments
         compute delta E_total by using P_act(i)
      end forall
   (1) select a pair whose delta E_total is minimum
   (2) two segments of the pair are combined
   (3) the combined segments are dealt with a segment
   (4) compute the probability of occurrence of
      each active segments
   end while
end Procedure
```

Fig. 7 Procedure for application-specific granularity control

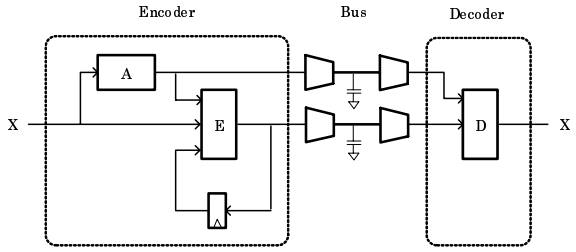


Fig. 8 Low-power encoder-decoder framework

$$\Delta E_{total} = P_{act}(i) \cdot (b_{i+1}/2 - 2P_{act}(i+1)) \cdot PT$$

After computing ΔE_{total} of all two continuous segments, we combine a pair of two segments whose ΔE_{total} is the minimum. After that, the probability of occurrence of the segment is calculated by adding the probability of occurrence of each ones. This operation repeats until the sum of segments is M. We summarize this procedure in Fig.7. The information of $\forall i, P_{act}(i)$ is given in advance by tracing the data. The values of b each segment are outputed by this procedure.

3.5 Implementation

Bus coding inherently introduces area, delay, and power overhead due to encoding and decoding circuits. In this subsection, we present the implementation results of encoding and decoding circuits for our proposed techniques. Figure 8 shows a structure of the encoder-decoder system. The encoder consists of two blocks: an active bitwidth detector-encoder A and an encoding block E. A decoder block D decodes the encoded data stream into original data stream X.

We designed encoders and decoders with Verilog-HDL. Then, the circuits are synthesized by Design Compiler (Synopsys corp.) under constraint of power optimization by using process technology of HITACHI $0.18\mu m$. Table 1 and 2 report the evaluation of synthesized circuits. In the tables u, s, onehot and binary indicate the unsigned integer, the signed integer, onehot coding and binary coding for active bitwidth, respectively.

We compared the Bus-Invert (BI) [10] technique with our proposed techniques. The BI technique is one of the most popular coding techniques. The BI and our techniques are well suited to a data bus. We evaluated the encoders and decoders of our techniques and BI with random input patterns. For the encoders, the BI consumes power about three times more than ours. The encoder circuit of BI uses plural FAs and XORs as basic cells. The power of these cells is quite large. For the decoders, the circuit for the embedding approach consumes more power. The reason is that in a decoder that implements the embedding approach there are many latches to hold previous values.

If a whole system is designed by using active bits, it is possible for some componets of the system to share their active bitwidth detectors. Therefore overhead can be reduced when multiple components in a whole system are designed by using active bits. As an illustration, assume that in a data bus and an arithmetic circuit, the techniques using active bits are used. When an encoded data using our codings is transfered from the data bus to inputs of the arithmetic circuit without decoding the data, a decoder in the bus and an active bit detector in the arithmetic circuit are not necessary.

When only an encoder ($N_{seg} = 32$ (the unsigned representation and onehot coding for active bitwidth)) is used without an active bitwidth detector and a decoder in the bus, delay penalty is 3.18 n *sec.* Hence we can save hardware cost by using techniques by Okuma et al. and Brooks et al. all together.

4. Experimental Results

We demonstrate experimental results by using our proposed techniques and BI one. The SimpleScalar [16], which is the processor simulator, is utilized as a tool to obtain data on the bus. This model has a 32-bits data bus. We selected the mpeg2play program and several SPEC2000 benchmarks as experimental application programs. For data dependency analysis, three images were used for the mpeg2play program: 116k byte, 245k byte and 649k byte images. We first traced data sequences on the data bus when memory accesses are occurred. After that, we applied our techniques and the BI technique to the data sequences. In the BI coding, if the Hamming distance between the present data and the last data of the bus is lager than N/2, the present data is transmitted with each bit inverted. An extra bit, called invert line, is required to signal the receiver side whether the bus is inverted or not.

Table 3 shows the results of switching count reduction by various techniques. Five techniques are used to encode data sequences of each benchmark. The five techniques are as follows:

Table 1 Evaluation of encoders

	our approach				
	$N_{seg} = 32$ (u:onehot)	$N_{seg} = 32$ (s:onehot)	$N_{seg} = 32$ (u:binary)	embedding	BI
extra bits	32	32	5	0	1
Cells	196	261	193	279	288
Area (μm^2)	10775	14100	10990	12157	17625
Power(μW)	377	593	409	452	1560
Delay(n sec)	6.63	9.89	5.53	8.78	5.91

Table 2 Evaluation of decoders

	our approach				
	$N_{seg} = 32$ (u:onehot)	$N_{seg} = 32$ (s:onehot)	$N_{seg} = 32$ (u:binary)	embedding	BI
extra bits	32	32	5	0	1
Cells	80	124	111	255	32
Area (μm^2)	2865	4369	4062	12995	1966
Power(μW)	177	281	297	665	215
Delay(n sec)	4.25	4.79	3.06	7.72	0.19

- Finest granularity detection ($N_{seg} = 32$) for unsigned integers (u)
- Finest granularity detection ($N_{seg} = 32$) for signed integers (s)
- Embedding technique
- operand: word, half-word, byte access
- BI (Bus-Invert) technique

In the operand case, we assume that the number of active bits of word, half-word, byte access of memory is 32 and 16 and 8, respectively.

In table 3, all results contain the switching count of extra bits. Power saving from 20% to 34% is achieved by using our techniques. In the best case, 54.2% of total switching count is reduced. Most cases using our techniques are better than the operand and BI cases. The reason that the operand case can not reduce switching count is that the processor almost access to the memory by the word access type. The BI technique has no effect when the hamming distance do not change drastically. Therefore in the case of some applications switching count is not reduced. We cannot observe data dependency for the mpeg2play program, because power is reduced about the same rates for all input data streams to the mpeg2play program.

The experimental results of granularity control are introduced in table 4. Bold types in the table 4 indicate the most efficient case of each application. In the onehot coding for active bitwidth, fine granularity segmentation achieve high switching count saving. In the binary coding for active bitwidth, contrary, by using rough granularity segmentation, high switching count reduction is achieved.

Next, we present the results of the application-specific granularity control technique in Table 5 and Fig.9. We adopted the mpeg2play program with a 245k byte image as a sample. Selecting small N_{seg}, we can save extra bits. In general, small N_{seg}, however, reduces the effectiveness of our proposed power saving techniques. By using the application-specific approach, high switching count saving is achieved even if we use small N_{seg}. In Table 5, the effectiveness of the application-specific approach is shown in the case of small N_{seg}. In addition, in Fig. 9, we can see that each segmentwidth is differenct after applying the application-specific granularity control. If you can know an application apriori, the technique introduced in 3.4 is efficient.

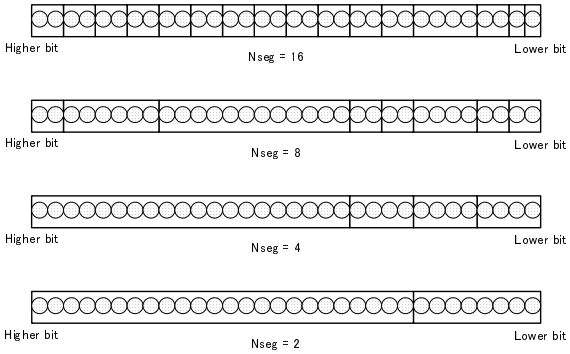


Fig. 9 Application-specific division

We discuss the total power consumption of the bus including the extra circuits. The coding system consists of three parts; encoder, decoder, and data bus. Figure 10 shows the comparison of total power consumption by using the coding techniques. This results is taken the average of the all applications. In Fig.10, u, s, onehot and binary stand for the unsigned integer representation, the signed integer representation, onehot coding and binary coding, respectively. In the case of binary coding, $N_{seg} = 4$ is used bacause of the most effective granularity in the binary cases. We implemented an encoder-decoder system for $N_{seg} = 4$. Then, we evaluated the hardware costs. The X axis is a capacitance scale, which is proportional to wire length on a bus. We vary capacitance from 5 pF to 50 pF, which is for a traditional package with traditional printed circuit board. For multichip module technology, the total off

Table 3 Comparison of switching count reduction

App.	unencoded #switching	reduction(%) $N_{seg}=32$(u:onehot)	$N_{seg}=32$(s:onehot)	embedding	operand	BI
mpeg2play (116k)	116570540	37.1	40.1	38.7	21.1	41.2
(245k)	196764229	45.6	49.0	49.6	32.5	48.0
(649k)	578222755	34.4	37.0	34.4	17.9	38.0
go	812346509	33.8	33.8	54.2	0.00	0.00
gzip	1054170245	10.5	10.2	16.6	-3.08	10.8
bzip2	8858426	17.8	17.8	36.2	26.6	0.00
perl	1828596	14.5	20.4	24.1	0.00	11.2
mesa	1744339	19.0	19.0	31.5	-0.13	1.11
wupwise	22486	18.0	19.0	27.9	4.18	14.9
average	–	22.7	24.4	34.3	8.58	12.6

Table 4 Switching count reduction ratio by using granularity control (%)

App.	onehot coding						binary coding				
	N_{seg}										
	32	16	8	4	2	1	32	16	8	4	2
mpeg2play (116k)	37.1	35.5	34.2	31.8	22.9	6.48	30.0	35.4	37.9	**38.6**	26.9
(245k)	45.6	44.1	42.7	40.3	27.0	5.19	40.7	44.8	**46.5**	45.3	30.9
(649k)	34.4	32.1	31.0	28.0	21.9	6.86	26.7	31.3	34.4	**35.8**	26.0
go	33.8	28.8	28.8	38.9	30.5	20.3	23.7	28.8	39.0	**49.1**	20.3
gzip	10.5	12.0	9.19	7.63	1.94	1.58	8.83	11.6	**12.7**	12.2	8.13
bzip2	17.8	18.5	19.1	18.5	19.2	29.6	7.19	13.1	19.1	25.6	**31.1**
perl	14.5	13.8	11.8	11.5	3.67	5.25	13.3	10.8	14.2	**16.7**	12.0
mesa	19.0	17.1	16.1	13.6	10.8	15.9	6.75	9.30	14.5	20.0	**22.1**
wupwise	18.0	16.6	15.6	16.2	9.71	5.02	12.3	13.9	17.7	**18.7**	15.2
average	22.7	21.6	20.5	20.9	14.7	11.8	16.1	18.9	23.1	**26.8**	20.0

Table 5 Application-specific detection approach results compared to the fixed granularity control(%)

	N_{seg}				
	32	16	8	4	2
fixed $b_i(=N/N_{seg})$	45.6	44.1	42.7	40.3	27.0
variable b_i	45.6	44.1	44.1	43.6	38.8

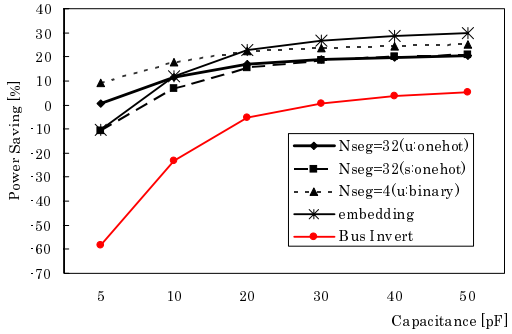


Fig. 10 Comparison of total power consumption

chip capacitance is 10 pF [4]. In all cases, our proposed techniques are more power-efficient compared to the BI coding. In short bus, the $N_{seg}=4$ (the unsigned representation and binary coding for active bitwidth) is the most effective in terms of power saving. On the other hand, the embedding approach is the most low power coding for long bus.

5. Conclusions

This paper presents a novel low power bus scheme based on reducing switching activity using dynamic active bit detection. Suppressing switching activity of inactive bits, we can reduce redundant power consumption. Our scheme is better suited for applications whose active bitwidths are of low correlation.

We presented various power reduction techniques using active bits and evaluated their implementations. A system designer can choose the best technique from them in view of a system specification. It is also possible to obtain optimized granularity for switching activity saving when an application is specified in a system. Since it is available to use active bits in a whole datapath for low overhead, we will study integration of a data memory[2] and an arithmetic circuits[3] and our data bus.

In ultra deep submicron VLSI designs coupling effects between on-chip interconnects and leakage current must be addressed. Since switching activity of lower bits is more lower than higher bits when our scheme is

applied, we can utilize lower bits for shielding against coupling effects like bus shuffling coding [7], [8]. Extra bits cause additional leakage energy. Therefore we must suppress increasing extra bits. We will study low power bus techniques considering these effects on power consumption in deep submicron era.

Acknowledgement

This work has been supported by the Grant-in-Aid for Creative Scientific Research No. 14GS0218 and the Silicon Sea-Belt Project "Establishing Project of a Cluster for System-LSI Design and Development". We are grateful for their support.

The VLSI chip in this study has been fabricated in the chip fabrication program of VLSI Design and Education Center(VDEC), the University of Tokyo in collaboration with Hitachi Ltd. and Dai Nippon Printing Corp. and Cadence Design Systems, Inc. and Synopsys, Inc.

References

[1] H. Yamashita, H. Tomiyama, A. Inoue, F. N. Eko, T. Okuma, and H. Yasuura, "Variable size analysis for datapath width optimization," Proc. 5th Asian Pacific Conference on Harware Description Language, pp.69–74, Jul. 1998.

[2] T. Okuma, Y. Cao, M. Muroyama, and H. Yasuura, "Reducing access energy of on-chip data memory considering active data bitwidth," Proc. 2002 International Symposium on Low Power Electronics and Design, pp.88–91, Aug. 2002.

[3] D. Brooks and M. Martonosi, "Value-based clock gating and operation packing: dynamic strategies for improving processor power and performance," ACM Trans. on Computer Systems, vol.18, no.2, pp.89-126, 2000.

[4] D. Liu and C. Svensson, "Power Consumption Estimation in CMOS VLSI Chips," IEEE Journal of Solid-State Circuits, vol.29, no.6, Jun. 1994.

[5] R. Ho, K. Mai and M. Horowitz, "The future of wires," Proc. IEEE, vol.89, issue 4, pp.490–504, Apr. 2001.

[6] D. Sylvester and K. Keutzer, "A global wiring paradigm for deep submicron design," IEEE Trans. on CAD, pp.242–252, Feb. 2000.

[7] Y. Shin and T. Sakurai, "Coupling-driven bus design for low-power application-specific systems," Proc. Design Automation Conference, pp.750–753, Jun. 2001.

[8] Jörg Henkel and Haris Lekatsas, "A^2BC: Adaptive address bus coding for low power deep sub-micron designs," Proc. Design Automation Conference, pp.744–749, Jun. 2001.

[9] K. Asada, M. Ikeda, S. Komatsu, "Approaches for reducing power consumption in VLSI bus circuits," IEICE Trans., vol.E83-C, no.2, pp.153–160, Feb. 2000.

[10] M. R. Stan and W. P. Burleson, "Bus-invert coding for low-power I/O," IEEE Trans. on VLSI Systems, vol.3, no.1, pp.49–58, Mar. 1995.

[11] L. Benini, G. D. Micheli, E. Macii, D. Sciuto, and C. Silvano,"Synthesis of low-overhead interfaces for power-efficient communication over wide buses," Proc. 36th Design Automation Conference, pp.128–133, June. 1999.

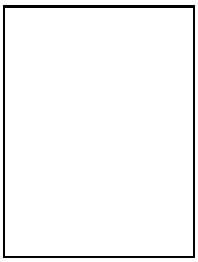
[12] P. R. Panda and N. D. Dutt, "Reducing address bus transitons for low power memory mapping," Proc. 1996 European Design and Test Conference, pp.63–67, Mar. 1996.

[13] H. Mehta, R. M. Owens, and M. J. Irwin, "Some issues in glay code addressing," Proc. 6th Great Lakes Symposium on VLSI, pp.178–180, Mar 1996.

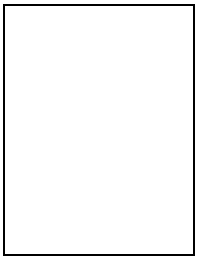
[14] L. Benini, G. D. Micheli, E. Macii, D. Sciuto, and C. Silvano, "Asymptotic zero-transition activity encoding for address buses in low-power microprocessor-based systems," Proc 7th Great Lakes Symposium on VLSI, pp.77–82, Mar. 1997.

[15] E. Musoll, T. Lang and J. Cortadella, "Working-zone encoding for reducing the energy in microprocessor address buses," IEEE Trans. on VLSI Systems, vol.6, no.4, pp.568–572, Dec. 1998.

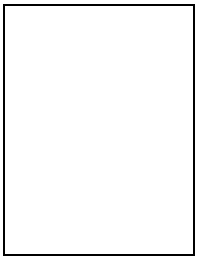
[16] SimpleScalar Simulation Tools for Microprocessor and System Evaluation, URL: http://www.simplescalar.com/.



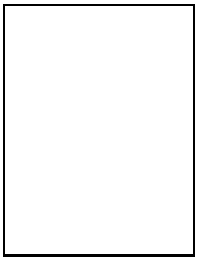
Masanori Muroyama received the B.E. and M.E. degrees in computer science and communication engineering, from Kyushu University, Fukuoka, Japan, in 2000 and 2002, respectively. Currently he is a Ph.D. candidate at Department of Computer Science and Communication Engineering, Graduate School of Information. His research interests include CAD for low power logic circuits design and low power VLSI system design. He is a student member of IEEE Computer Society.



Akihiko Hyodo received the B.E. and M.E. degrees in computer science and communication engineering, from Kyushu University, Fukuoka, Japan, in 1999 and 2001, respectively. Currently he is a Ph.D. candidate at Department of Computer Science and Communication Engineering, Graduate School of Information Science and Electrical Engineering, Kyushu University. His research interests include high performance and low energy microprocessor design and energy efficient VLSI system design.



Takanori Okuma received the B.E., M.E. and Ph.D. degrees in computer science and communication engineering, from Kyushu University, Fukuoka, Japan, in 1997, 1999 and 2002, respectively. Currently he has worked in the cadence design systems incorporation.



Hiroto Yasuura is a professor of Department of Computer Science and Communication Engineering, Graduate School of Information Science and Electrical En-

gineering, Kyushu University. He is also a director of System LSI Research Center in Kyushu University. Prof. Yasuura received the B.E., M.E. and Ph.D. degrees in computer science from Kyoto University, Kyoto, Japan, in 1976, 1978, and 1983 respectively. He was an associate professor in Kyoto University and moved to Kyushu University in 1991. Prof. Yasuura developed several CAD systems for VLSI and hardware algorithms of arithmetic operations, sorting and unification in Kyoto University. In Kyushu University, Prof. Yasuura has conducted research projects on the system LSI design methodology, which includes data-path width optimization, low-energy system design, SoC architecture and a core base LSI test method. He also developed an educational microprocessor, KUE-CHIP2, and promoted education of VLSI design in computer science area in Japan. His current interests include embedded system design, harware/software co-design, VLSI CAD and system design methodology. He served as a technical program chair and a general chair of ICCAD in 1997 and 1998, respectively. He is serving as a Vice President of IEEE CAS Society, ACM SIGDA advisory board member, and Steering Committee Chair of ASP-DAC2003. He is also the research director of Silicon Sea Belt Project in Fukuoka.