Leakage Power Reduction for Battery-Operated Portable Systems

Cao, Yun
System LSI Research Center, Kyushu University

Yasuura, Hiroto
Graduate School of Engineering Sciences, Kyushu University

出版情報：IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences. E86-A (12), pp.3200-3203, 2003-12. IEICE
バージョン：
権利関係：

LETTER

Leakage Power Reduction for Battery-Operated Portable Systems

Yun Cao[†], *and* **Hiroto Yasuura**[††], *Member*

SUMMARY

This paper addresses bitwidth optimization focusing on leakage power reduction for system-level low-power design. By means of tuning the design parameter, bitwidth tailored to a given application requirements, the datapath width of processors and size of memories are optimized resulting in significant leakage power reduction besides dynamic power reduction. Experimental results for several real embedded applications, show power reduction without performance penalty range from about 21.5% to 66.2% of leakage power, and 14.5% to 59.2% of dynamic power.

key words: Bitwidth optimization, leakage power reduction, dynamic power reduction

1. Introduction

The increasing use of battery-operated portable computing and wireless communication systems makes power dissipation a major concern in modern designs [2]. Leakage power dissipation constitutes an increasing fraction of the total power in modern semiconductor technologies. Designing power efficient products will require consideration of leakage power in the earliest phases of design [1].

This paper presents bitwidth optimization focusing on leakage power reduction for system level design, while providing adequate performance level. The power dissipation of the whole system not only dynamic power but also leakage power is drastically reduced by tuning the parameters of processors and memories tailored for the applications.

The rest of this paper is structured as follows: the next section 2 presents leakage power reduction technique using bitwidth optimization, estimation models for leakage power dissipation are also presented in this section. Experiments are described in section 3. Finally, section 4 concludes our work.

2. Leakage Power Reduction

2.1 A Leakage Power Model

We have proposed bitwidth optimization for dynamic power reduction in [8]. Dynamic power dissipation is described by the familiar formula, $P_{dyn} = \frac{1}{2} \cdot C \cdot V_{cc}^2 \cdot f$, C is the capacitance of switching nodes, which is roughly proportional to the number of switching devices. V_{cc} is the supply voltage, and f is the effective operating frequency (frequency times activity factor). Our technique reduces dynamic power by reducing C and f.

This paper focuses on leakage power reduction. Leakage power dissipation is equal to the product of the supply voltage and the leakage current. To deal with the leakage power at system level, we start addressing leakage power model from a simple equation for estimation in [3],

$$P_{static} = V_{cc} \cdot N \cdot k_{design} \cdot \hat{I}_{leak} \quad (1)$$

where V_{cc} is the supply voltage, N is the number of transistors, k_{design} is a design dependent parameter, and $\hat{I}_{leak}$ is a technology dependent parameter.

The model suggests different ways in which leakage power may be reduced. One obvious technique that can be employed is to reduce the total number of devices. In this paper, we present bitwidth optimization to reduce the number of devices(N) by tuning datapath width of processors for each application, to try to ease the problem of leakage power. Because reducing the number of devices directly reduces the switching capacitance(C), our technique also reduce dynamic power dissipation as well.

2.2 Design Platform

We have developed a design platform for embedded core-based systems, which consists of a variable configuration processor called Bung-DLX [4], a multi-precision retargetable compiler called Valen-C retargetable compiler [5], a variable size analyzer [6] and a cycle-based simulator [7].

2.3 Bitwidth Optimization

Optimizing bitwidth of datapath for each given application is an effective technique to reduce leakage power dissipation of the whole embedded systems. The leakage power reduction problem is formulated as *to minimize* $P_s(w)$*, subject to* $Cycle(w) \leqq C_{cst}$.

minimize	$P_s(w)$	
subject to	$Cycle(w) \leqq C_{cst}$	Leakage power $P_s(w)$
	$Area(w) \leqq A_{cst}$	

Manuscript received March 12, 2003.
Submmit to Special Section on
"VLSI Design and CAD Algorithms"
Manuscript revised June 12, 2003.

Input:
source program : AP
(variables : $x_i \in X = \{x_1, x_2, ..., x_n\}$, $1 \leqq i \leqq n$)
input data : D_{in}
the constraint of cycles : C_{cst}
Variable:
datapath width $w_i \in W$=$\{w_1$,w_2,...$w_n\}$
Output:
execution cycles $c_i \in C = \{c_1$,c_2,...$c_n\}$
the minimal leakage power dissipation P_{sMin} when $c_k \leqq C_{cst}$
the datapath width w_k when $P_{s_k} = P_{sMin}$
Phase 1 : Variable Size Analysis
{
$analyzer \leftarrow (AP, X, D_{in})$
$return(EWd = \{EWd(x_1), EWd(x_2), ..., EWd(x_n)\})$
}
Phase 2 : Define Design Parameters for Bung-DLX
datapath width w_i
the number of registers n_i
Phase 3 : Valen-C program
variable declaration of bit width $EWd(x_i)$
compile the Valen-C source program for customized Bung-DLX at w_i
Step 4 : Bitwidth Optimization
for $W \neq \emptyset$
{
estimate the execution cycles c_i
estimate the leakage power dissipation P_s
$P_{sMin} \leftarrow$ the minimal leakage power when $w_i = w_k$ under $c_k \leqq C_{cst}$
}
$return(P_{sMin}, w_k, c_k)$

Fig. 1 Pseudo code of the algorithm for leakage power reduction

and cycle $Cycle(w)$ are functions of datapath width w, C_{cst} is the constraint on the execution cycle.

The overview of our leakage power reduction algorithm is described in Figure1. In the initial design phase of our approach, we design a system with a variable configuration processor(Bung-DLX), data RAMs, instruction ROMs and logic circuits. Then we analyze the effective bitwidth of each variable ($EWd(x_i)$) in a given application program (AP). After that, using the results of analysis, we rewrite the application program in Valen-C language, in which we specify the word length of each variable satisfying accurate computation to reduce leakage power consumed by redundant bits in the application program. After verifying the functionality of the initial design, we modify several design parameters of the variable configuration processor, including the datapath width w_i, the number of registers and the instruction set. We can tune up the variable configuration processor to minimize the leakage power dissipation (P_{sMin}) while satisfying the system performance constraints($c_k \leqq C_{cst}$).

Since in many cases, high-level specifications are devoted to describe functionalities of target systems rather than implementation details, they often contain a lot of redundancies such as duplicated computations and never executed code. Therefore, the specifications must be optimized to remove the redundancies for power-efficient design. Some redundancies are introduced in size of variables. For example, in C programs, a variable whose value is between 0 and 1000 is often declared as the *int* type, i.e., usually 16 or 32 bits depending on target processors, and then some upper bits are useless. C language provides three integer sizes, declared using the keywords *short*, *int* and *long*. The compiler designer determines the sizes of these integer types. We present Valen-C language, by which programmers can explicitly specify the required bitwidth of each integer data type, so it becomes possible to reduce the leakage power of the datapath and the data memory, which is dissipated by the redundant bits. As a result, specifying the bitwidth required for each variable and changing the datapath width have a significant role in reducing the processor and the data memory size of a system. Therefore it also affects the power dissipation of the system.

The value of the datapath width can be tuned in accordance with the characteristics of target system to deliver most suited processor. Designers can reduce the datapath width until the single precision point (SPP) without performance loss. SPP is the processor datapath width, which is equal to the bitwidth of the biggest variable in a program. It is the smallest datapath width at which all instructions can remain single-precision. The datapath width of a processor strongly affects the power dissipation of the whole system including the processor, data memories and instruction memories, it also affects the execution cycles of a given task, i.e., narrowing the datapath width less than SPP will cause the increase of execution cycles because of multiple-precision operations. So trade-offs exist between datapath width and execution cycles. Therefore, for a given target system, trading off the power dissipation and performance is an important work for bitwidth optimization.

2.4 Estimation Models for Leakage Power

This section presents leakage power models to give the criteria for leakage power reduction. We assume that the target system consists a processor, a data RAM and an instruction ROM. The variable configuration processor Bung-DLX is used. The total leakage power dissipation, P_{sTot} of a system, is estimated as the summation of leakage power consumed by the processor (P_{sProc}) and memories (P_{sMem}).

$$P_{sTot} = P_{sProc} + P_{sMem} \tag{2}$$

P_{sProc} and P_{sMem} are estimated separately. We built the leakage power dissipation model of Bung-DLX generated by HITACH 0.5 μm CMOS technology and the power dissipation models of memories generated by Alliance CAD System Ver.4.0 with 0.5 μm double metal CMOS technology.

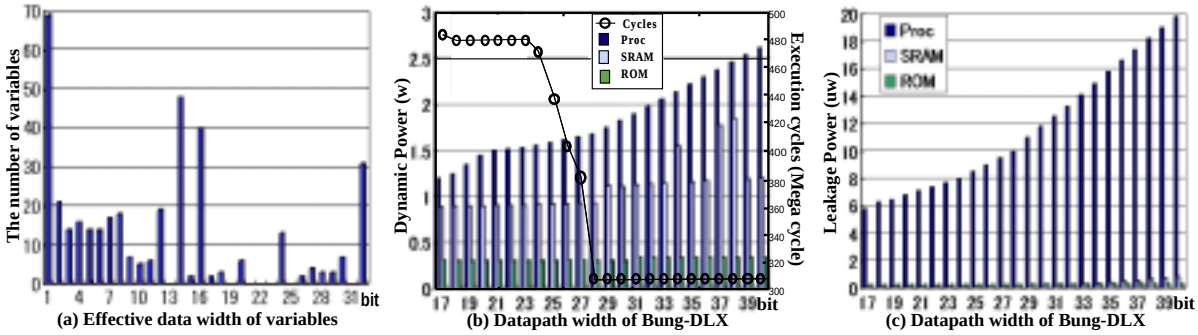
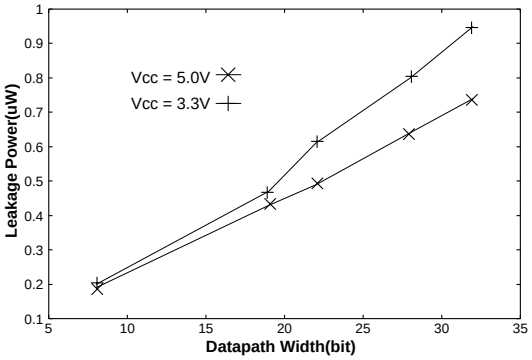


Fig. 3 Results of MPEG-2 video decoder



$DatapathWidth$	$V_{cc} = 5.0V$		$V_{cc} = 3.3V$	
	P_{sProc}	$Saving$	P_{sProc}	$Saving$
32bit	0.74	-	0.95	-
28bit	0.64	13.5%	0.80	15.8%
22bit	0.49	33.8%	0.61	35.8%
19bit	0.43	41.9%	0.47	61.1%
8bit	0.19	74.3%	0.20	79.0%

Fig. 2 Leakage power of Bung-DLX (μw)

P_{sProc} is obtained by using Synopsys Power Compiler. The power dissipation in static CMOS circuitry can be divided into static(leakage), dynamic and short-circuit power. Here we just focus on leakage power (P_{sProc}). After several simulations, we obtained the empirical power model at several datapath widths for supply voltage V_{cc} of $5.0V$ and $3.3V$ respectively, shown in the table of Figure2, where power savings, $Saving$ are got by comparing to the leakage power dissipation of the 32bits processor.

P_{sProc} can be described as follows:

$$P_{sProc} = \sum_{\forall cells_k} P_{CellLeakage_k} \tag{3}$$

P_{sProc}: Total leakage power of a processor
$P_{CellLeakage_k}$: Leakage power of each cell k
P_{sMem} is estimated as follows:

$$P_{sMem} = P_{sROM} + P_{sSRAM} \tag{4}$$

where
P_{sMem} : Total leakage power of memory
P_{sROM} : Leakage power of ROM
P_{sSRAM} : Leakage power of SRAM

The power of P_{sROM}, P_{sSRAM} is obtained from the SPICE simulation of several memories with different configurations. As the result, we have obtained the estimation models as follows:

$$P_{sROM} = 18.1 * b * \sqrt{N_{words}} + 0.08[pw] \tag{5}$$
$$P_{sSRAM} = 231.0 * b * \sqrt{N_{words}} + 1.1[pw] \tag{6}$$

Where b is the bit width of the memory and N_{words} is the number of words.

3. Experiments

This section reports some experimental results concerning the use of our technique to reduce power dissipation based on several real applications. We report both dynamic power and leakage power results.

In the experiments, we assumed the target system, a SOC chip, which consists a Bung-DLX processor, a ROM and a SRAM. The ROM and the SRAM are used as instruction memory and data memory respectively. For simplicity, we assumed that no other core is integrated in the SOC chip.

Four real embedded applications are used as benchmarks, which are Lempel-Ziv algorithm, ADPCM encoder, MPEG-2 AAC audio decoder, and MPEG-2 video decoder. The cycle count is used to evaluate performance obtained by using the simulator [7]. For dynamic power estimation, the power models in [8] are used, and for leakage power estimation, models in section 2.4 are used.

Figure 3 shows the estimation results for MPEG-2 video decoder. We analyzed the C source program of MPEG-2 video decoder from the MPEG Software Simulation Group, using our developed variable size analyzer and got the variable size analysis results of effective data width depicted in Figure3(a). This figure shows that there are a lot of variables having many

Application	No. Opt. Dynamic Power(mw)				Opt. Dynamic Power(mw)				Savings
	P_{dProc}	P_{dSRAM}	P_{dROM}	P_{dTot}	P_{Proc}	P_{dSRAM}	P_{dROM}	P_{dTot}	
Lempel-Ziv	645.8	330.2	66.98	$1.04w$	208.3	184.6	31.4	424.3	59.2%
ADPCM	155.5	82.5	118.5	356.5	77.5	51.28	70.3	199.1	44.2%
Mpeg2AAC	589.3	247.6	337.23	$1.17w$	503.6	194.5	301.7	999.8	14.5%
Mpeg2Video	$2.05w$	$1.16w$	349.68	$3.56w$	$1.68w$	930.72	305.97	$2.91w$	18.3%

Application	No. Opt. Leakage Power(μw)				Opt. Leakage Power(μw)				Savings
	P_{sProc}	P_{sSRAM}	P_{sROM}	P_{sTot}	P_{sProc}	P_{sSRAM}	P_{sROM}	P_{sTot}	
Lempel-Ziv	3.8	$36.7nw$	$59.7nw$	3.90	1.28	$19.8nw$	$24.9nw$	1.32	66.2%
ADPCM	0.95	$8.8nw$	0.11	1.06	0.48	$5.7nw$	$61.7nw$	0.55	48.1%
Mpeg2AAC	3.1	$28.3nw$	0.30	3.43	2.3	$21.5nw$	0.26	2.59	21.5%
Mpeg2Video	13.3	0.21	0.31	13.82	10.0	0.18	0.27	10.4	24.7%

Fig. 4 Power savings for benchmarks

unused bits in MPEG-2 decoder, which originally declared as "int" type. We got average 39% reduction of bits from the variable size analysis. The dynamic power dissipation and execution cycles are described in Figure 3(b), and the estimation results of leakage power shown in Figure 3(c). From these figures, we can get the optimal datapath width, 28bits for MPEG-2 video decoder without performance loss. Optimal datapath width is the datapath width where the whole system has the minimization power dissipation without performance penalty.

Figure4 shows the results of the experiments employed our technique for the benchmarks. In the first table of Figure4 for dynamic power, column $No.Opt.DynamicPower$ including four columns, which show the results for original designs without using our technique. Column P_{dProc} shows the dynamic power dissipation of processor Bung-DLX, column P_{dSRAM} is dynamic power of SRAM, column P_{dROM} is the dynamic power of ROM, and column P_{dTot} shows the total dynamic power dissipation. The next four columns show the results using our optimization technique($Opt.$). The last column $Savings$ shows the dynamic power reduction compared to the designs without using our technique($No.Opt.$).

The results of leakage power obtained are listed in the second table of Figure 4. Column $No.Opt.LeakagePower$ show the results for original designs without using our technique. Column P_{sProc} shows the leakage power for processor Bung-DLX, column P_{sSRAM} for SRAM, column P_{sROM} for ROM, and column P_{sTot} shows the total leakage power dissipation. The next four columns show the results using our optimization technique($Opt.$). The last column $Savings$ shows the leakage power reduction compared to the designs without using our technique($No.Opt.$).

4. Conclusions

In this paper, we have proposed a system-level design technique focusing on leakage power reduction, which can suit the complexity of embedded systems and stringent time-to-market constraints. We have demonstrated power savings without performance penalty average about 40.1% of leakage power, and 34.1% of dynamic power, which based on a number of real embedded applications. If given a power constraint, within tolerable performance loss, this technique can further reducing power consumption.

5. Acknowledgments

This work has been supported by the Grant-in-Aid for Creative Scientific Research No.14GS0218 of the Ministry of Eductaion, Science, Sports and Culture (MEXT) from 2002 to 2006. We are grateful for their support.

References

[1] Takayasu Sakurai, "Minimizing Power across Multiple Technology and Design Levels", International Conference on Computer Aided Design (ICCAD), pp. 24-27, 2002.

[2] S. Borkar, "Low Power Design Challenges for the Decade", Proc. of Asia South Pacific Design Automation Conference(ASPDAC), pp. 293-296, 2001.

[3] J. Adam Butts and Gurindar S. Sohi, "A Static Power Model for Architects", Proc. of the 33th Annual International Symposium on Microarchitecture, pp. 191-201, 2000.

[4] F. N. Eko, A. Inoue, H. Tomiyama, H. Yasuura, "Soft-Core Processor Architecture for Embedded System Design", IEICE Trans.on Electronics, Vol.E81-C, No.9, pp. 1416-1423, 1998.

[5] A.Inoue, H.Tomiyama, T.Okuma, H.Kanbara and H.Yasuura, "Language and Compiler for Optimizing Datapath Width of Embedded Systems", IEICE Trans. Fundamentals, Vol. E81-A, No.12, pp. 2595-2604, Dec. 1998.

[6] H. Yamashita, H. Yasuura, F. N. Eko, and Yun Cao, "Variable Size Analysis and Validation of Computation Quality", Proc. of Workshop on High-Level Design Validation and Test, pp.95-100, 2000.

[7] E. N. Eko and H. Yasuura, "A Cycle-Accurate Simulator Toolkit for Soft-Core Processors", Proc. of Asia Pacific Conference on Chip Design Languages, pp. 11-16, 1999.

[8] Yun Cao, Hiroto Yasuura, "A System-level Energy Minimization Approach Using Datapath Width Optimization", Proc. of International Symposium on Low Power Electronics and Design(ISLPED), pp. 231-236,2001.