Reducing Static Energy of Cache Memories via Prediction-Table-less Way Prediction

Akihito Sakanaka[1] and Toshinori Sato[1,2]

[1] Department of Artificial Intelligence
Kyushu Institute of Technology
[2] Precursory Research for Embryonic Science and Technology
Japan Science and Technology Corporation
toshinori.sato@computer.org

Abstract. Power consumption is becoming one of the most important constraints for microprocessor design in nanometer-scale technologies. Especially, as the transistor supply voltage and threshold voltage are scaled down, leakage energy consumption is increased even when the transistor is not switching. This paper proposes a simple technique to reduce the static energy. The key idea of our approach is to allow the ways within a cache to be accessed at different speeds. We combine variable threshold voltage circuits with way prediction technique to activate only the way which will be referred, and propose a simple prediction mechanism which eliminates history tables. Experimental results on 32-way set-associative caches demonstrate that any severe increase in clock cycles to execute application programs is not observed and significant static energy reduction can be achieved, resulting in the improvement of energy-delay product.

1 Introduction

Power consumption is becoming one of the most important concerns for microprocessor designers in nanometer-scale technologies. Until recently, the primary source of energy consumption in digital CMOS circuits has been the dynamic power that is caused by dynamic switching of load capacitors. The trend of the reduction in transistor size reduces capacitance, resulting in less dynamic power consumption. Microprocessor designers have relied on scaling down the supply voltage, resulting in further dynamic power reduction[8, 9, 13]. In addition, many architectural technologies to reduce power have been proposed by reducing the number of switching activities[7, 11, 15]. To maintain performance scaling, however, threshold voltage must also be scaled down with supply voltage. Unfortunately, this increases leakage current exponentially. The International Technology Roadmap for Semiconductors (ITRS) predicts an increase in leakage current by a factor of two per generation[18]. Borkar estimates a factor of 5 increases in leakage energy in every generation[1].

Many of techniques[3, 10, 12, 17] proposed to address this problem have focused on cache memory that is a major energy consumer of the entire system

because leakage energy is a function of the number of transistors. For example, the Alpha 21264 and the StrongARM processors use 30% and 60% of the die area for cache memories[15]. Current efforts at static energy reduction have focused on dynamically resizing active area of caches[3, 10, 12, 17]. These architectural techniques require additional circuits to control cache activities. The control circuitry including history tables consumes power, and thus these dynamic approaches are not suitable for embedded processors due to the area and power overhead and design complexity. In order to solve the problem, we propose a simple technique for leakage energy reduction.

The organization of the rest of this paper is as follows: Section 2 discusses the motivation of our work. Section 3 presents our concept to reduce leakage energy in caches, and explains a prediction-table-less way prediction mechanism. Section 4 presents experimental results and discussion on the effectiveness of our approach. Finally, Section 5 concludes the paper.

2 Motivations

2.1 Leakage Energy

Power consumption in a CMOS digital circuit is governed by the equation:

$$P = P_{active} + P_{off} \tag{1}$$

where P_{active} is the active power and P_{off} the leakage power. The active power P_{active} and gate delay t_{pd} are given by

$$P_{active} \propto f C_{load} V_{dd}^2 \tag{2}$$

$$t_{pd} \propto \frac{V_{dd}}{(V_{dd} - V_t)^\alpha} \tag{3}$$

where f is the clock frequency, C_{load} the load capacitance, V_{dd} the supply voltage, and V_t the threshold voltage of the device. α is a factor dependent upon the carrier velocity saturation and is approximately 1.3–1.5 in advanced MOSFETs[6]. Based on Eq.(2), it can easily be found that a power-supply reduction is the most effective way to lower power consumption. However, Eq.(3) tells us that reductions in the supply voltage increase gate delay, resulting in a slower clock frequency, and thus diminishing the computing performance of the microprocessor. In order to maintain high transistor switching speeds, it is required that the threshold voltage is proportionally scaled down with the supply voltage.

On the other hand, the leakage power can be given by

$$P_{off} = I_{off} V_{dd} \tag{4}$$

where I_{off} is the leakage current. The subthreshold leakage current I_{off} is dominated by threshold voltage V_t in the following equation:

$$I_{off} \propto 10^{-\frac{V_t}{S}} \tag{5}$$

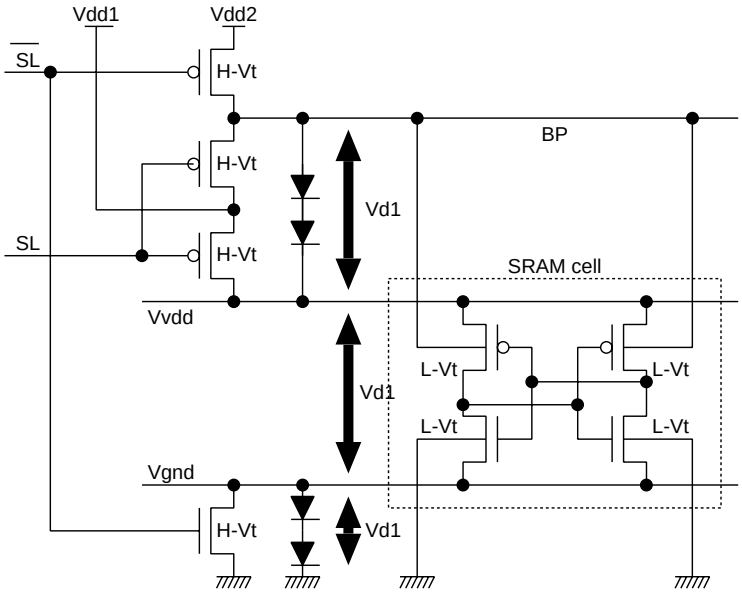

Fig. 1. ABC-MOS Circuit

where S is the subthreshold swing parameter and is around 85mV/decade[18]. Thus, lower threshold voltage leads to increased subthreshold leakage current and increased static power. Maintaining high transistor switching speeds via low threshold voltage gives rise to a significant amount of leakage power consumption.

2.2 Variable Threshold Voltage CMOS Circuits

There are several circuits proposed to reduce leakage current by dynamically raising the threshold voltage, for example by modulating backgate bias voltage[14, 16]. Figure 1 shows ABC-MOS circuit[16]. The four transistors with high threshold voltage (denoted as `H-Vt`) become the switch that cut off leakage current. During normal operation, when sleep signal `SL` is deasserted, the virtual source line, `Vvdd`, and the back gate bias line, `BP`, are set to appropriate power supply, `Vdd1`. During sleep mode, when `SL` is asserted, P-well is biased using the alternative power supply, `Vdd2`, at a higher voltage level. Diodes perform backgate bias effect and thus threshold voltage of all transistors is increased. This reduces leakage current, which flows from `Vdd2` to the ground. As shown in Table 1, Hanson et al.[5] report that leakage current in sleep mode is reduced by a factor of 160 compared with that in normal mode. VT-CMOS[14] also controls the backgate bias to reduce leakage current in sleep mode by rising up threshold voltage.

Table 1. Impact of V_t on Leakage Current

Mode	I_{off} (nA)
Sleep	12
Normal	1941

2.3 Related Works

Current efforts at static energy reduction have focused on dynamically resizing active area of caches[3, 10, 12, 17]. These architectural techniques employ circuit techniques. VT+ADR cache[3] and SA cache[10] use VT-CMOS[14] and ABC-MOS[16], respectively. Decay cache[12] and DRI cache[17] use gated-V_{dd}[17], which shuts off the supply voltage to SRAM cells to reduce leakage current. The circuit technique has a disadvantage. Gated-V_{dd} loses the state within the memory cell in the sleep mode. Thus, additional cache messes might occur, resulting in an additional dynamic power consumption. In contrast, VT-CMOS and ABC-MOS can retain stored data in the sleep mode. However, these architectural techniques require additional circuits to control cache activities. The control circuitry, especially large cache-structured history table, consumes power, however, most studies did not consider its effect. In summary, these dynamic approaches are not suitable for embedded processors due to the area and power overhead and design complexity.

3 Prediction-Table-less Way Prediction

In this paper, we propose a simple technique for leakage energy reduction. We combine the variable threshold voltage circuit with way prediction to limit the number of ways in normal mode. Way prediction is a technique for set-associative cache, which reduces conflict misses and maintains the hit speed of direct-mapped cache. It predicts the way within the set of the next cache access. A hit prediction means short access latency, since only a single way is speculatively referred. However, a miss results in checking the remaining ways in subsequent cycles and thus in long access latency. A dynamic power reduction technique based on way prediction has been proposed[7], and it employs a way prediction policy based on the most recently used (MRU) algorithm.

In this study, we use way prediction for leakage energy reduction. Based on every prediction, only one way, which will be referred in the next cache access, is activated. Remaining ways are set to sleep mode. In other words, only one way consumes large static energy and the other ways do negligible one. We name the cache *non-uniform-access-latency (NUAL) cache*. NUAL cache based on way prediction resembles VT+ADR cache based on address prediction[3] and SA cache based on block prediction[10]. However, while they require large history table to predict the region which will be accessed next, NUAL cache eliminates the history table as explained next.

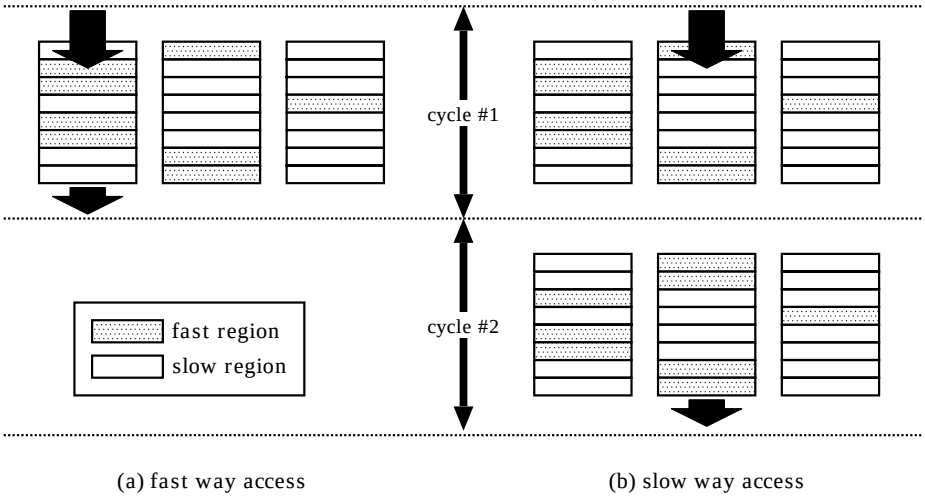

Fig. 2. Non-Uniform Access-Latency Cache

We use least recently used (LRU) bits that is already implemented in every set-associative cache. The LRU bits are utilized to determine which way is replaced for the next cache refill process, and keep the order of way accesses. According to the LRU bits, least recently used ways go to sleep mode. The remaining ways stay in normal mode. As you can easily find, no history table is required for NUAL cache. While we use a 3-way set-associative cache for explanation, our proposal is applicable to any set- and full-associative caches. In this explanation, in every set, two least recently used ways go to sleep mode, and only one way is in normal mode. We call the way in normal mode *the primary fast way* and those in sleep mode *slow ways*. When a cache access refers the primary fast way, the datum is provided in 1 cycle as shown in Fig.2(a). The primary ways are in gray. In contrast, it refers a slow way, activating the slow way requires 1 cycle and the datum is provided in the following cycle, as shown in Fig.2(b).

We name the NUAL cache presented in Fig.2 *fine-grained* NUAL cache, because activating or inactivating is each set basis. This requires a backgate bias control circuit per way in every set. To reduce the number of the backgate bias control circuits, it is possible to make the activate policy way basis. This requires an additional N-bit LRU register for N-way set-associative cache and only N backgate bias control circuits. We name this type of NUAL cache *coarse-grained* NUAL cache. The behavior of coarse-grained NUAL cache is explained in Fig.3. When a cache access refers the primary fast way, the datum is provided in 1 cycle as shown in Fig.3(a). There is no difference from fined-grained NUAL cache. In contrast, it refers a slow way, activating the slow way requires 1 cycle and the datum is provided in the following cycle, as shown in Fig.3(b). In this case, all

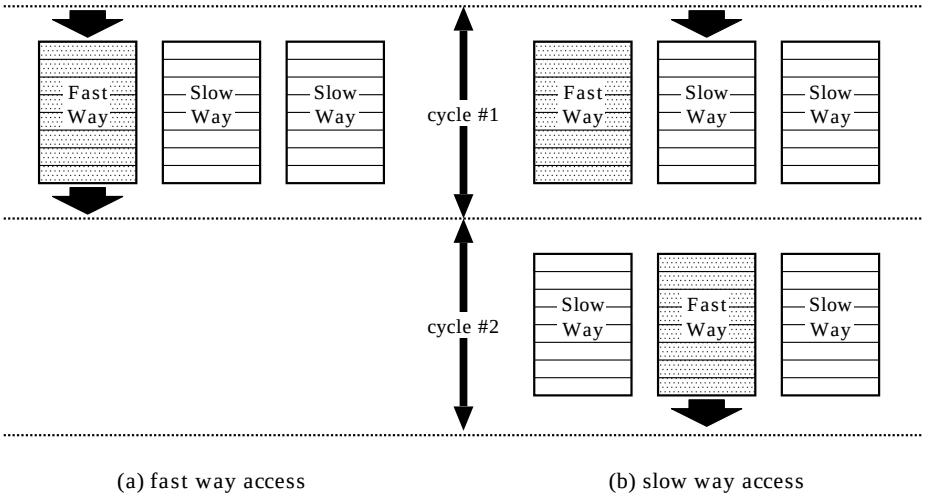

Fig. 3. Coarse-Grained NUAL Cache

ways regardless of sets are activated, different from fine-grained cache. As you can find, in coarse-grained NUAL cache, it is impossible to optimally predict the next way for every set, and thus average access latency may be longer than that of fine-grained NUAL cache. In the next section, we will evaluate this.

4 Experimental Results

4.1 Simulation Methodology

We implemented our simulator using the SimpleScalar/ARM tool set[2]. We use ARM instruction set architecture in this study. The processor model evaluated is based on Intel XScale processors[8]. 4KB, 4B block, 32-way set-associative L1 instruction and data caches are used. They have a load latency of 1 cycle and a miss latency of 32 cycles. The non-uniform-access-latency 32-way set-associative L1 instruction and data caches consist of one primary fast way and remaining 31 slow ways. It has the same load latency of 1 cycle when the requested data is placed in the primary way and otherwise has a latency of 2 cycles, 1 cycle of which is required to activate the sleeping the way[5, 10] and another cycle of which is for reading the way. We also evaluate caches whose load latency is always 2 cycles for comparison. The replacement policy is based on LRU. No L2 cache is used. A memory operation that follows a store whose data address is unknown cannot be executed.

The MiBench[4] is used for this study. It is developed for use in the context of embedded, multimedia, and communications applications. It contains image processing, communications, and DSP applications. We use original input files

Table 2. Benchmark programs

program	input set
FFT	Fast Fourier Transform
IFFT	Inverse FFT
Rawcaudio	ADPCM encode
Rawdaudio	ADPCM decode
Toast	GSM encode
Untoast	GSM decode
CRC	32-bit Cyclic Redundancy Check

provided by University of Michigan. Table 2 lists the benchmarks we used. All programs are compiled by the GNU GCC with the optimization options specified by University of Michigan. Each program is executed to completion.

4.2 Energy Parameters

To use data from Hanson et al.'s[5], we assume a 70nm process technology, a 2.5GHz clock frequency, a 0.75V supply voltage, and a 110C operating temperature. To compute total energy reduction, we compute the leakage energy using the numbers shown in Table 1 and the number of clock cycles to execute each program. However, since Hanson et al.[5] reports the energy required to awake every way is only 50 fJ per bit, we ignore the energy in our evaluation.

4.3 Results

Figures 4 and 5 shows way hit ratio in the cases of coarse- and fine-grained NUAL caches. The hit means that a cache access refers the primary fast way. The left bar is for the instruction cache, and the right is for the data cache. In the case of the coarse-grained NUAL cache, instruction and data caches suffer approximately 20% and 50% miss ratios, respectively. Every miss requires additional latency to activate the way. This might occur severe performance loss. In contrast, both fine-grained NUAL instruction and data caches achieve 90% of the way hit ratio for most programs.

Figure 6 shows relative processor performance. For each group of 4 bars, the first one (see from left to right) indicates the performance which has the conventional set-associative cache, whose load latency is always 1 cycle (denoted by `Fast`). The middle two bars indicate those of the coarse- and fine-grained NUAL caches (denoted by `Coarse` and `Fine`), respectively, and the right indicates that of the slow conventional cache, which has a load latency of 2 cycles (denoted by `Slow`). Every result is normalized by that of the `Fast` model. We can find the model with the `Fine` NUAL cache has no considerable difference from that with the conventional fast and energy-hungry `Fast` cache. In addition, it has significant performance gain over that with slow and energy-efficient `Slow` cache. In

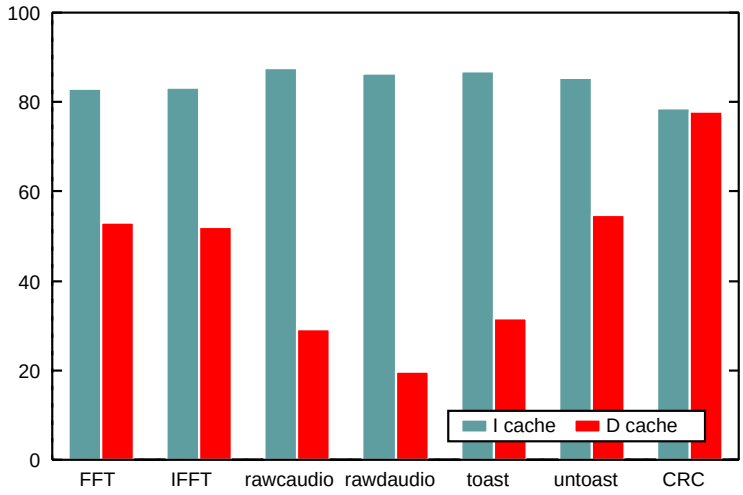

Fig. 4. %Way Hit Ratio (coarse-grained)

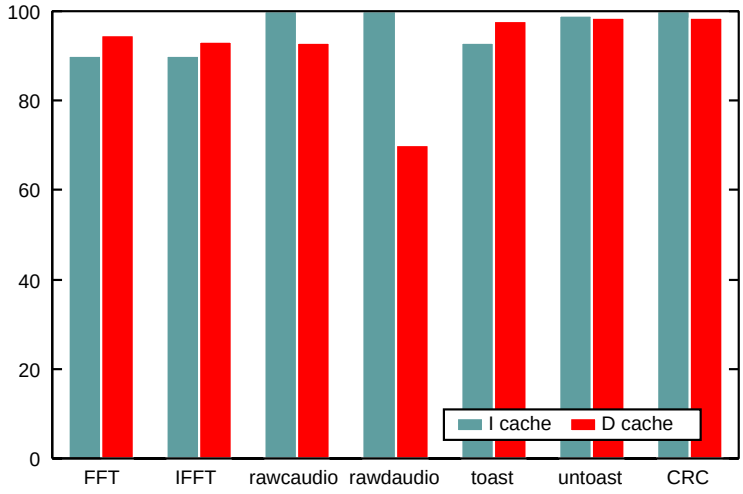

Fig. 5. %Way Hit Ratio (fine-grained)

contrast, the `Coarse` NUAL cache suffers approximately 10% performance loss, as we expected due to relatively large miss ratio (see Fig. 4).

The static energy consumed in NUAL cache is reduced by approximately the factor of 30, since only 1 of 32 ways is activated and we do not consider the energy required to awake every way. In other words, we only consider leakage

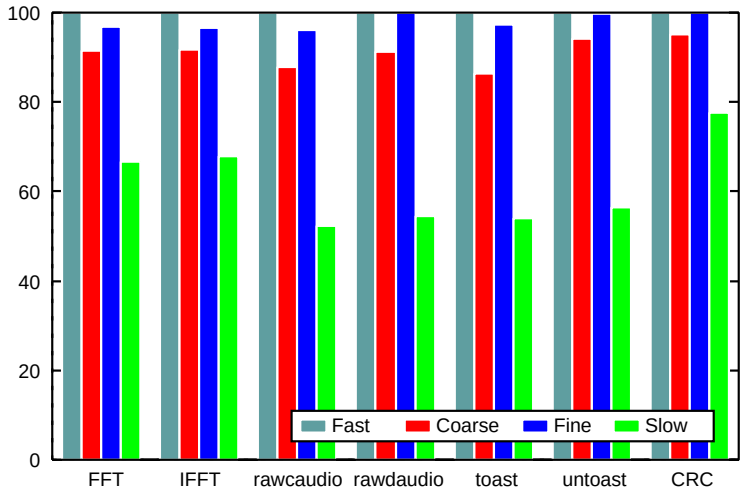

Fig. 6. %Processor Performance

energy. Based on the discussion in Section 4.2, in the case of the processor model with the fine-grained NUAL cache, we calculate the leakage energy and find over 96% reduction of energy-delay product due to leakage energy for all benchmark programs.

5 Conclusions

In this paper, we have proposed a simple technique to reduce the static energy consumed in caches. The key idea of our approach is to allow the ways within a cache to be accessed at different speeds and to combine a variable threshold voltage circuit with way prediction. Simulation results showed that any severe increase in clock cycles to execute the application program was not observed and significant static energy reduction could be achieved, resulting in over 96% reduction of energy-delay product due to leakage energy in caches.

References

1. S. Borker, "Design challenges of technology scaling," IEEE Micro, volume 19, number 4, 1999.
2. D. Burger and T. M. Austin, "The SimpleScalar tool set, version 2.0," ACM SIGARCH Computer Architecture News, volume 25, number 3, 1997.
3. R. Fujioka, K. Katayama, R. Kobayashi, H. Ando, and T. Shimada, "A preactivating mechanism for a VT-CMOS cache using address prediction," International Symposium on Low Power Electronics and Design, 2002.

4. M. R. Guthaus, J. S. Ringenberg, D. Ernst, T. M. Austin, T. Mudge, R. B. Brown, "MiBench: A free, commercially representative embedded benchmark suite," Workshop on Workload Characterization, 2001.
5. H. Hanson, M. S. Hrishikesh, V. Agarwal, S. W. Keckler, and B. Burger, "Static energy reduction techniques for microprocessor caches," International Conference on Computer Design, 2001.
6. T. Hiramoto and M. Takamiya, "Low power and low voltage MOSFETs with variable threshold voltage controlled by back-bias," IEICE Transactions on Electronics, volume E83-C, number 2, 2000.
7. K. Inoue, T. Ishihara, and K. Murakami, "Way-predicting set-associative cache for high performance and low energy consumption," International Symposium on Low Power Electronics and Design, 1999.
8. Intel Corp., "Intel XScale technology," `http://developer.intel.com/design/intelxscale/`, 2002.
9. T. Ishihara and K. Asada, "A system level memory power optimization technique using multiple supply and threshold voltages," Asia and South Pacific Design Automation Conference, 2001.
10. T. Ishihara and K. Asada, "An architectural level energy reduction technique for deep-submicron cache memories," Asia and South Pacific Design Automation Conference, 2002.
11. A. Iyer and D. Marculescu, "Power aware microarchitecture resource scaling," Design, Automation and Test in Europe Conference and Exhibition, 2001.
12. S. Kaxiras, Z. Hu, G. Narlikar, and R. McLellan, "Cache-line decay: a mechanism to reduce cache leakage power," Workshop on Power Aware Computer Systems, 2000.
13. A. Klaiber, "The technology behind Crusoe processors," Transmeta Corporation, White Paper, 2000.
14. T. Kuroda, T. Fujita, S. Mita, T. Nagamatsu, S. Yoshioka, F. Sano, M. Norishima, M. Murota, M. Kato, M. Kinugasa, M. Kakumu, and T. Sakurai, "A 0.9V, 150MHz, 10mW, 4mm^2, 2-D discrete cosine transform core processor with variable-threshold-voltage scheme," International Solid State Circuit Conference, 1996.
15. S. Manne, A. Klauser, and D. Grunwald, "Pipeline gating: speculation control for energy reduction," International Symposium on Computer Architecture, 1998.
16. K. Nii, H. Makino, Y. Tujihashi, C. Morishima, and Y. Hayakawa, "A low power SRAM using auto-backgate-controlled MT-CMOS," International Symposium on Low Power Electronics and Design, 1998.
17. M. Powell, S. H. Yang, B. Falsafi, K. Roy, and T. N. Vijaykumar, "Gated-Vdd: a circuit technique to reduce leakage in deep-submicron cache memories," International Symposium on Low Power Electronics and Design, 2000.
18. D. Sylvester and H. Kaul, "Power-driven challenges in nanometer design," IEEE Design & Test of Computers, volume 18, number 6, 2001.