The Phoenix Processor: A 30pW Platform for Sensor Applications

Mingoo Seok, Scott Hanson, Yu-Shiang Lin, Zhiyoong Foo, Daeyeon Kim, Yoonmyung Lee, Nurrachman Liu, Dennis Sylvester, David Blaauw
University of Michigan, Ann Arbor, MI

Abstract

An integrated platform for sensor applications, called the Phoenix Processor, is implemented in a carefully-selected 0.18µm process with an area of 915x915µm^2, making on-die battery integration feasible. Phoenix uses a comprehensive sleep strategy with a unique power gating approach, an event-driven CPU with compact ISA, data memory compression, a custom low leakage memory cell, and adaptive leakage management in data memory. Measurements show that Phoenix consumes 29.6pW in sleep mode and 2.8pJ/cycle in active mode.

Introduction

Form-factor is a critical concern for wide applicability and cost effectiveness in sensor systems, especially in medical applications. In this work, we explore the development of a sensor platform, called the Phoenix Processor, which will occupy only 1mm^3 when coupled with an on-die battery. To ensure multi-year lifetime for the given platform size, average power consumption must be reduced to tens of pW [1], which marks a ~4000X reduction over previous sensor designs [2]. Recent work [2-6] has explored aggressive V_{dd} scaling for reducing active energy but has overlooked the power consumed during idle periods, which can be >99% of the lifetime. In addition to aggressive voltage scaling, Phoenix leverages a comprehensive sleep strategy, including the intentional selection of an older, low leakage technology, a unique power gating approach, an event-driven CPU with compact instruction set, data memory compression, a custom low leakage memory cell, and adaptive leakage management in the data memory. A test chip, including a sensor and timers, was fabricated in an area of 915x915µm^2 in a 0.18µm process. Phoenix consumes 29.6pW in sleep mode and 2.8pJ/cycle in active mode at V_{dd}=0.5V. For a typical sensor application that runs 2000 instructions every 10 minutes, average power consumption is 39pW, which will enable integration of an on-die battery in a volume of 1mm^3 while ensuring >1 year lifetime [1].

System Overview

As shown in Fig 1(a), Phoenix is a modular system with the CPU, memories (DMEM, IMEM, IROM) and power management unit (PMU) serving as parents of the system bus and peripherals (timer, sensor) acting as children. The accommodations made for sleep mode are best understood by exploring typical system operation (Fig 2(a)). The system begins in sleep mode (A), where 65-87% of all transistors are power gated using footers (depending on the retentive size of DMEM). In this mode, the PMU, a 2-bit FSM (Fig 3), remains awake and acts as the parent of the system bus. All gates in the PMU use stacked high-V_{th} (HVT) devices with V_{th}~0.7V to minimize leakage. In addition to the PMU, IMEM cells and valid DMEM cells remain awake to retain data. As shown in Fig 2(c), measurements at V_{dd}=0.5V reveal that total power is limited to 29.6pW in sleep mode with half DMEM retention. Note in Fig 2 that IMEM and DMEM draw 89% of sleep mode power.

After a programmable sleep time (e.g., 10min), a 0.9pW timer similar to [7] raises an interrupt on the system bus using the asynchronous protocol. In response the PMU initiates a short wake sequence and relinquishes control of the bus to the CPU (B). The CPU then runs a routine of ~2000 instructions to query the sensor for data and process the acquired data (C). During this routine, the CPU decompresses a block of DMEM and places it in the cache (which is part of the register file), requests a sensor measurement over the bus, processes the returned data, compresses the cache contents and stores it to DMEM, and finally issues a sleep request. The PMU then regains control of the bus and gates system power (D,E). Fig 2(b) shows measured energy and frequency characteristics for Phoenix in active mode across V_{dd}. At V_{dd}=0.5V, the system operates at 106kHz and consumes 2.8pJ/cycle, with 88% of energy consumed by the CPU. Phoenix effectively operates at V_{dd}=384mV due to a non-zero virtual ground.

Sleep Strategies

One of the critical pieces of our sleep strategy is the unique approach to power gating (Fig 4). Traditional power gating uses a wide HVT footer to minimize the voltage drop across the footer and thus maintain performance. Additionally, a HVT footer is attractive since it gives a dramatic leakage reduction compared to a medium-V_{th} (MVT) footer for minimal delay penalty. Our approach for low V_{dd} power gating differs in two ways: 1) we use a MVT footer since on-current for a HVT footer is exponentially smaller at low V_{dd}, making HVT footers infeasible, and 2) our aim is to minimize energy rather than maintain performance [8], so footer size is set to only 0.66µm (0.01% of total effective NFET width) with L=0.50µm. Due to its small size, the footer develops a voltage drop of 116mV in active mode (frequency implications shown in Fig 5), elevating the energy-optimal V_{dd} to 0.5V. (In contrast to the optimal value of 0.36V reported for the un-gated design in [2]) The reduced leakage of the small footer (Fig 6) easily offsets the slight increase in active energy due to the power consumed across the footer (Fig 7). Total energy is reduced by 2.5X compared to a somewhat larger footer of 28µm and by 4 orders compared to a design without a footer (Fig 8). Additionally, the elevated optimal V_{dd} aids in robust SRAM design.

While the CPU (Fig 9(a)) largely impacts active mode power, it also plays an important role in sleep mode. An event-driven operating system initiates computation only in response to interrupts raised by peripherals, ensuring that the system defaults to sleep mode. Since IMEM is a major source of sleep mode power, the instruction set was chosen to minimize IMEM footprint using a minimum group of basic instructions while also including support for compression, interrupt and sleep functionalities. To limit instruction width to only 10 bits, common instructions use flexible addressing modes while less common instructions use implicit operands. Additionally, the 64-word IMEM is supplemented with a low leakage 128-word IROM that contains common functions.

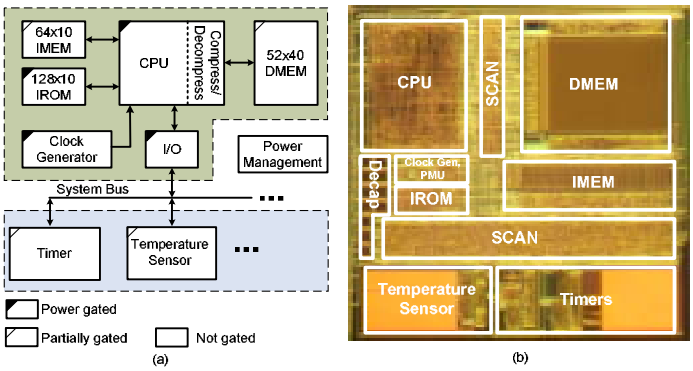
Hardware support for compression (Fig 9(c)) was included in the CPU to minimize the DMEM footprint in sleep mode and to maximize memory capacity. A virtual data memory space of 512B is mapped to the 256B DMEM using Huffman encoding with a fixed dictionary for a maximum compression ratio of 50%. DMEM is divided into 2 partitions: statically and dynamically allocated (Fig 9(b)). Each group of 16B (a block) in virtual memory is given one line in the statically allocated partition. If a write to memory causes a block to overflow its statically allocated entry, overflow data is written to an entry in dynamically allocated memory whose location is noted by a pointer in the statically allocated entry. A 52b free-list, which is visible to the CPU, monitors the usage of entries in both memory partitions.

Since SRAM can dominate total energy consumption, we place emphasis on low leakage SRAM design. Both IMEM and DMEM use the bitcell shown in Fig 10. The cross-coupled inverters and access transistors use HVT devices, while stack forcing and gate length biasing are used to further reduce leakage and improve subthreshold swing. Measurements show that a single bitcell (~40µm^2, which is acceptable for sensor applications) consumes 10.9fW while retaining data. To enable robust low V_{dd} operation, the proposed cell includes a MVT read buffer similar to [9]. The MVT read buffer also enables single-cycle read-out despite the aggressive use of HVT devices elsewhere. This is useful for the IMEM, where a read occurs every cycle. Since the write operation in DMEM is slow relative to the MVT CPU, write operations are asynchronous. Write completion is determined by reading the contents of the row being written and comparing to the write data. Read is single-ended, so a replica delays the write completion signal to guarantee that both sides of the cell have been written correctly.

To further reduce sleep power, the DMEM uses a leakage reduction scheme based on the free-list. The DMEM has 26 footers, with each connected to 2 rows (Fig 10). The choice of 2 rows per footer offers a

good trade-off between high granularity in power gating and footer overhead. These footers are selectively turned off during sleep mode based on the contents of the free-list, reducing DMEM sleep power DMEM from 22.5pW at full retention to 7.5pW at zero memory retention, a 66% reduction.

[1] F. Albano, et al, Journal of Power Source, 170, 2007
[2] S. Hanson, et al, Symposium on VLSI Circuits, 2007.
[3] A. Wang, et, al, International Solid-State Circuits Conference, 2004.
[4] C. Kim, et al, Transactions on VLSI Systems, Vol 11, 2003.
[5] B. Zhai, et al, Symposium on VLSI Circuits, 2006.
[6] B. Calhoun, et al, International Solid-State Circuits Conference, 2005.
[7] Y. Lin, et al, Custom Integrated Circuits Conference, 2007.
[8] M. Seok, et al, Design Automation Conference, 2007.
[9] L. Chang, et al, Symposium on VLSI Technology, 2005.



Chip Area	915µm x 915µm	MVT Device Count	60,332
Technology	0.18µm	HVT Device Count	32,167
V_{th}	0.4V(MVT), 0.7V(HVT)	Target V_{dd}	0.5V

(c)

Figure 1: (a) System diagram (b) Die photo (c) Test-chip specifications

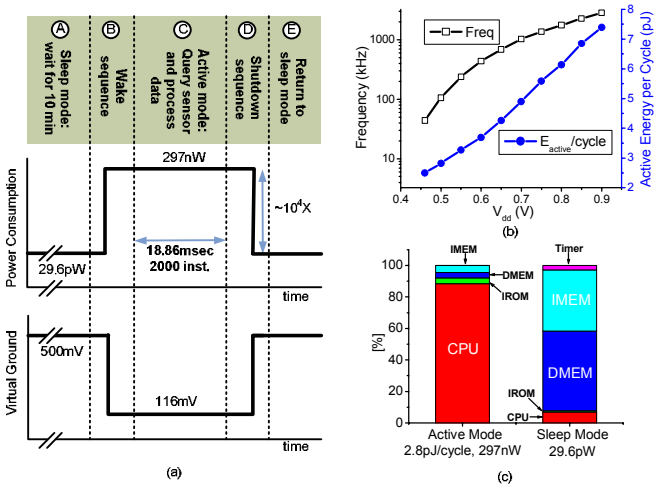


Figure 2: (a) Power and virtual ground profiles for a typical event sequence at Vdd=0.5V (b) Active energy and frequency measurements as functions of Vdd (c) Sleep and active energy/power measurements for all components

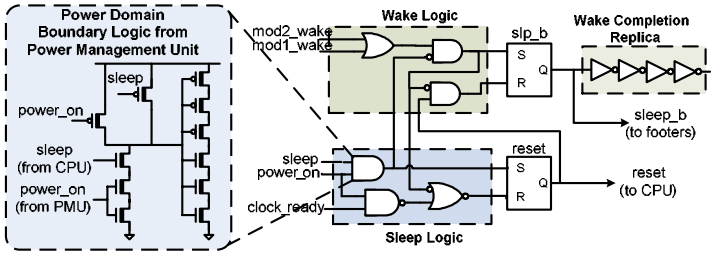


Figure 3: Power management unit diagram with boundary gate highlighted. Boundary gates have skewed P/N strength ratios for implicit level conversion and stacking of retentive inputs to prevent excessive leakage in pull-down stack.

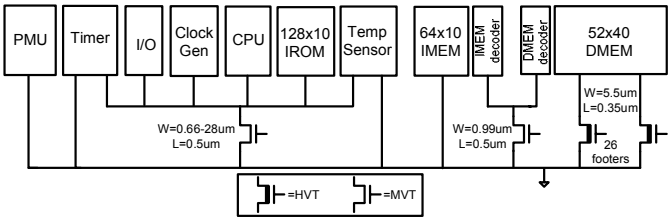


Figure 4: Footer allocation in Phoenix

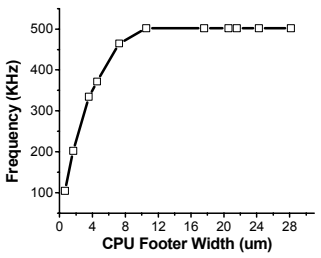


Figure 5: Maximum frequency vs. CPU footer width (CPU only)

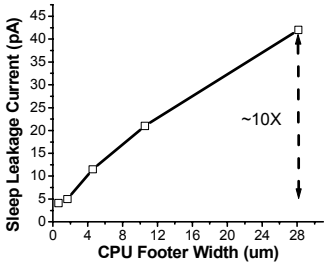


Figure 6: Sleep leakage current vs. CPU footer width (CPU only)

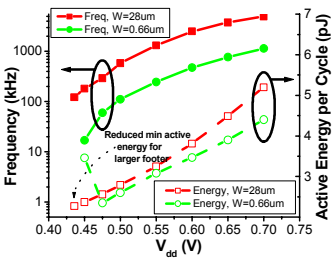


Figure 7: Active energy and frequency vs. Vdd for different footer widths (CPU only)

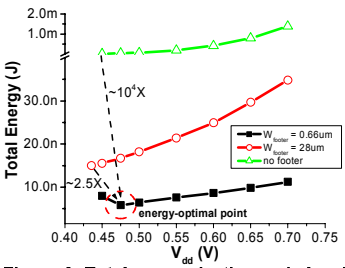


Figure 8: Total energy (active and sleep) vs. Vdd for different footer widths (CPU only)

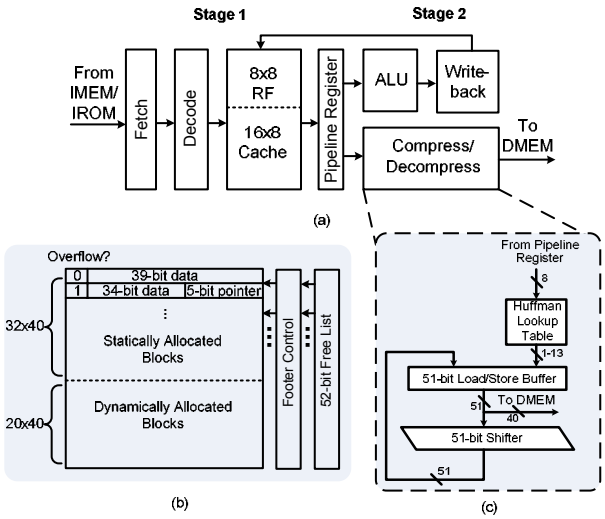


Figure 9: (a) CPU diagram (b) Memory organization for compression (c) Hardware support for compression

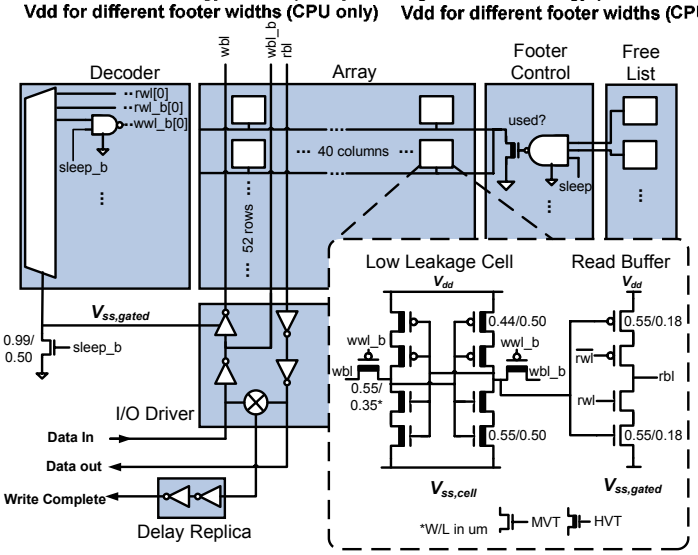


Figure 10: 10.9fW/cell memory architecture. Array diagram highlights footer sharing every 2 rows. Inset figure shows bitcell architecture.