Nonvolatile Processor Architectures: Efficient, Reliable Progress with Unstable Power

Kaisheng Ma
Xueqing Li
Pennsylvania State University

Karthik Swaminathan
IBM T.J. Watson Research Center

Yang Zheng
Pennsylvania State University

Shuangchen Li
University of California, Santa Barbara

Yongpan Liu
Tsinghua University

Yuan Xie
University of California, Santa Barbara

John (Jack) Sampson
Vijaykrishnan Narayanan
Pennsylvania State University

Nonvolatile processors (NVPs) are a promising solution for energy-harvesting scenarios in which the available power supply is unstable and intermittent. This article explores the design space for an NVP across different architectures, input power sources, and policies for maximizing forward progress in a framework calibrated using measured results from a fabricated NVP. The authors propose a heterogeneous microarchitecture solution that efficiently capitalizes on ephemeral power surpluses.

•••••• To handle unstable power conditions, such as power from ambient Wi-Fi signals, solar panels, and human movement piezo-electronic energy harvesters (see the "Energy-Harvesting Systems" sidebar), traditional processors need a large energy-storage device such as a supercapacitor to conservatively accumulate sufficient energy for task completion before that task starts. Otherwise, a task failure could occur at every power-supply emergency. Although this approach is viable for certain deployments, both form factor (the energy-storage device's mass and/or size) and conversion efficiency can preclude fully conservative solutions. Existing solutions to power instability mainly consist of checkpointing techniques to store the intermediate task computation states to external nonvolatile memory (NVM) storage, such as flash, before power failures occur. However, reset and rollbacks with communication to external data storage result in high performance and power costs.[1,2]

Unlike traditional processors, nonvolatile processors (NVPs) can leverage their nonvolatility feature to ensure forward progress without relying on additional durable storage to preserve processor state.[1–4] An NVP is a processor with built-in NVM and the facilities to back up all state on the chip to these memories when a power failure occurs and to restore the processor state when power returns. Importantly, an NVP's instruction-level backup and recovery operations can be built transparent to programmers and compilers, making it compatible with many design automation techniques and more energy efficient than existing solutions. Although NVPs intuitively offer simpler minimum forward progress guarantees, recent works[3–5] have shown that NVPs offer superior overall forward progress—that is,

Energy-Harvesting Systems

Recent developments in body-area networks and the Internet of Things (IoT) markets have led to a profound proliferation of power-constrained and form-factor-constrained devices. For many of these devices, their intended deployment conditions or durations make battery-powered operation difficult or impractical. With technological improvements in both material types and computation efficiency, energy-harvesting systems[1–6] have become a plausible alternative to battery-powered operation, in that the average power available for harvesting-powered devices is now sufficient to perform meaningful computation. Figure A shows a sampling of such energy-harvesting IoT devices[7,8] and highlights their diversity in energy source (RF, piezoelectric,[9,10] thermal,[11] and solar[12]), form factor, application space, and power needs. Although such devices are well-suited for certain niche applications, they do not present a competitive alternative for the complex computations required by more mainstream applications. This is because of the difficulty in using a short-term power budget that is both highly variable and deeply unpredictable.

Figure B shows the power traces for four typical ambient energy sources that could be harvested to power an embedded system: namely, solar energy and energy due to RF radiation, piezoelectric

(1)

(2)

Temporomandibular joint (jaw joint)

Ear canal deformation

Jaw joint movement

(3)

(4)

Figure A. A sampling of energy-harvesting Internet of Things devices. (1) A Wi-Fi-powered camera, with demonstration system mounted on an industrial gas cylinder, monitoring a pressure gauge.[7,8] (2) In-shoe piezoelectric devices[9] and a piezoelectric ear canal motion energy harvester.[10] (3) Thermoelectric generator.[11] (4) Solar leaves.[12]

effect, and thermal gradients. The RF energy is obtained by measuring the power of the frequency spectrum from a TV station, the piezoelectric energy is measured through devices fixed on a bike, the thermal energy is generated from characterizations described by Romain Grezaud and Jerome Willemin,[13] and the solar trace is obtained using data from the Measurement and Instrumentation Data Center (www.nrel.gov/midc). Although all of these sources are nearly ubiquitously available, there are several drawbacks in relying on ambient sources of energy for computing purposes. Most harvesters of these energy sources operate at relatively low conversion efficiencies, because only a small fraction of the total transmitted power can be tapped. In addition, a common issue across harvesting systems is unstable input power, because external factors could cause a supply disruption. For instance, ambient RF or Wi-Fi power can vary arbitrarily, according to power source, frequency, distance from the transmitter, height, obstacles, external electromagnetic signals, and other factors[3]; Figure B1 showcases this phenomenon, with instantaneous power levels that can vary by orders of magnitude over even very short timescales. To understand how these sources' properties will enable and limit various aspects of energy-harvesting systems, we classify power sources according to three primary characteristics: signal magnitude, signal strength variability, and power drop-out frequency (intermittency).

With respect to signal variability, we observe substantial variation in power, even over a few milliseconds, for RF in Figure B1 with the ratio between the maximum and minimum power over this period around 250 times. The piezoelectric power is more stable than RF with just some short power loss in Figure B2. The thermal power, shown in Figure B3, is even more stable, because of the gradual nature of temperature variation. Variation in solar power, seen in Figure B4, depends on the weather conditions and orientation of the solar cell.

Another feature is the intermittency frequency, which influences how soon the power drops below a viable threshold, as indicated by the annotations in Figure B1. The intermittency frequency strongly influences backup and recovery overheads. Sources with periodic behavior, as in Figure B2, favor prediction of power loss and enable efficient scheduling of tasks, whereas less predictable sources similar to Figure B1 must consider more conservative approaches or minimize the cost of mispredictions.
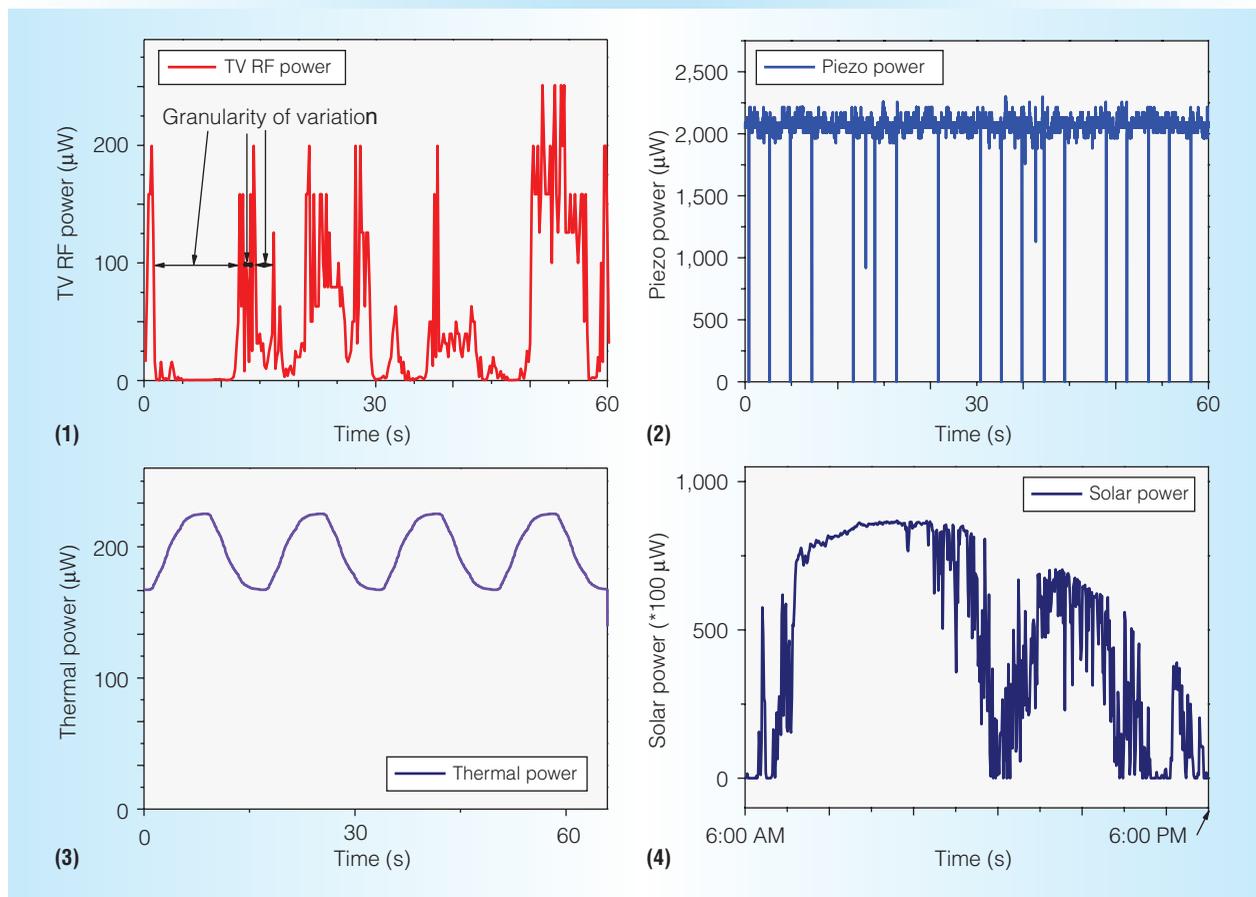
Figure B. Power traces. (1) TV station RF, (2) piezoelectric, (3) thermal, and (4) solar. Sample time for each figure is 0.33 μs.

References

1. K. Ma et al., "Architecture Exploration for Ambient Energy Harvesting Nonvolatile Processors," *Proc. Int'l Symp. High-Performance Computer Architecture*, 2015. pp. 526–537.
2. K. Ma et al., "Nonvolatile Processor Architecture Exploration for Energy-Harvesting Applications," *IEEE Micro*, vol. 35, no. 5, 2015, pp. 32–40.
3. Y. Liu et al., "Ambient Energy Harvesting Nonvolatile Processors: From Circuit to System," *Proc. 52nd Ann. Design Automation Conf.*, 2015, article 150.
4. Y. Liu et al., "A 65nm ReRAM-Enabled Nonvolatile Processor with 6X Reduction in Restore Time and 4X Higher Clock Frequency Using Adaptive Data Retention and Self-Write-Termination Nonvolatile Logic," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2016, pp. 84–85.
5. X. Li et al., "RF-Powered Systems Using Steep-Slope Devices," *Proc. IEEE 12th Int'l New Circuits and Systems Conf.*, 2014, pp. 73–76.
6. H. Liu et al., "Tunnel FET RF Rectifier Design for Energy Harvesting Applications," *IEEE J. Emerging and Selected Topics in Circuits and Systems*, vol. 4, no. 4, 2014, pp. 400–411.
7. V. Talla et al., "Powering the Next Billion Devices with Wi-Fi," arXiv preprint, 2015; arXiv:1505.06815.
8. S. Naderiparizi et al., "WISPCam: A Battery-Free RFID Camera," *Proc. IEEE Int'l Conf. RFID*, 2015, pp. 166–173.
9. N.S. Shenck et al., "Energy Scavenging with Shoe-Mounted Piezoelectrics," *IEEE Micro*, vol. 21, no. 3, 2001, pp. 30–42.
10. A. Delnavaz and J. Voix, "Energy Harvesting for In-Ear Devices Using Ear Canal Dynamic Motion," *IEEE Trans. Industrial Electronics*, vol. 61, no. 1, 2014, pp. 583–590.
11. S.J. Kim et al., "A Wearable Thermoelectric Generator Fabricated on a Glass Fabric," *Energy & Environmental Science*, vol. 7, no. 6, 2014, pp. 1959–1965.
12. "Solar Power from Energy-Harvesting Trees," video, 16 Feb. 2015; http://youtu.be/_QswunfBC8U.
13. R. Grezaud and J. Willemin, "A Self-Starting Fully Integrated Auto-Adaptive Converter for Battery-Less Thermal Energy Harvesting," *Proc. IEEE 11th Int'l New Circuits and Systems Conf.*, 2013; doi:10.1109/NEWCAS.2013.6573612.

conversion of incoming energy into useful work—than other approaches.

The concept of a processor or microcontroller with integrated nonvolatility is not new. Commercial products from Texas Instruments[6] and chips from academic sources[2] have been produced with nonvolatile retention features. However, our previous work[3] and its extensions[4] represent the first exploration into the architectural design space of NVPs; the interactions among microarchitectures, backup policies, and harvesting technologies; and the design and management of an energy-harvesting system that maximizes conversion of incoming ambient energy into useful work in an environment where power emergencies can occur with frequencies in the tens of instructions. Our work aims to provide forward guidance for an increasingly battery-less Internet-of-Things future, and we have highlighted key differences between design optimizations for harvesting and battery-powered systems.

Our models have been validated against fabricated NVP hardware, taking into account system-level effects of power instability beyond the processor itself. NVP research is fundamentally cross-layer in nature, because power instability affects every aspect of the system, from application quality of service to delays in phase-locked-loop stabilization to the interaction of capacitor sizing, rectifier efficiency, and the ability to make predictions regarding whether continued execution during a power emergency is likely to improve or degrade the rate of progress. We show how, even in this extremely power-limited environment, aggressive architectures operating beyond the maximum efficiency point can actually be beneficial in scenarios where peak power can be vastly higher than average, and we show the potential of predictive mechanisms to more aggressively exploit whatever incoming energy does still arrive during a power emergency and to select among microarchitectures in a heterogeneous design to maximize forward progress. In this article, we summarize our key findings and approaches to exploring the tradeoffs in the design space of NVP architectures and policies.

Architectural and Backup Policy Codesign Space

Even the simplest processor microarchitecture has multiple possible policies for what state to back up and when. Here, we outline the space of microarchitectures and
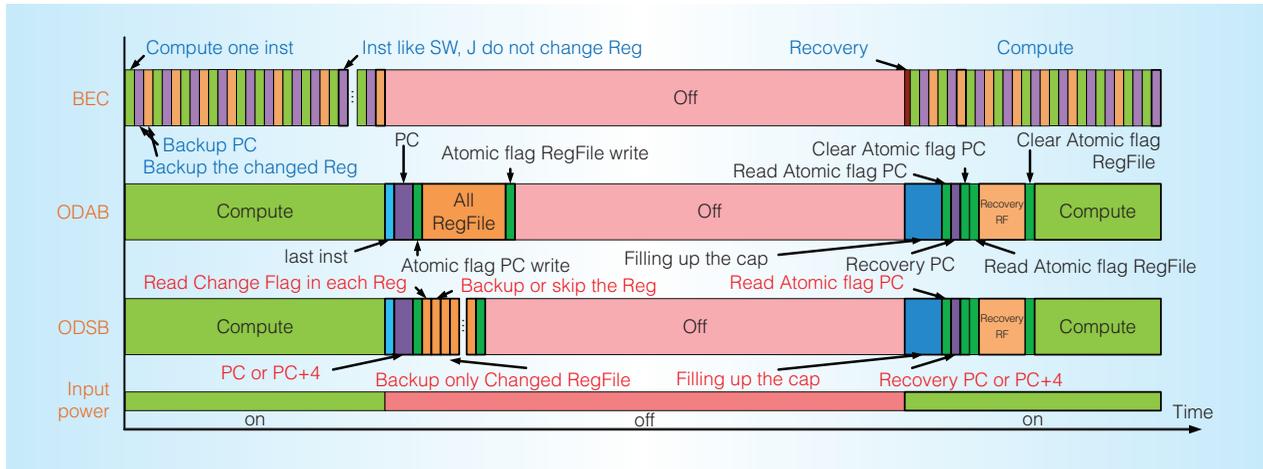
Figure 1. Runtime components for nonpipelined (NP) configuration schemes.

associated policies we considered. We focused on determining which architectural configurations were best suited to optimally use the available power and energy by maximizing processor performance under different energy constraints. Hence, depending on the energy harvested, we analyzed various parameters, such as the number of pipeline stages, data to be backed up, and frequency of backups.

Nonpipelined Configuration

In the absence of any pipeline stages, the entire processor state can be characterized by a single instruction state. In addition to the architecture, there are also tradeoffs between the energy consumed in backing up and recovering the data and the overall performance. We explore these tradeoffs by choosing which data to save and where and when to save them for three architectures of gradient complexity.

The first policy, *backup every cycle* (BEC), employs an NVM register file, or else both the contents of a volatile Regfile and its counterpart nonvolatile location need to updated every cycle. As Figure 1 shows, only the program counter (PC) and a few registers are written into the Regfile every cycle. Some instructions, such as StoreWord and Jump, do not require any further Regfile write.

The second policy, *on-demand all backup* (ODAB), differs from the previous solution in that all RegFile entries must be backed up only in the event of a reduced power state. In the last policy, *on-demand selective backup*

(ODSB), only data that has changed since the last backup is updated in NVM.

N-Stage Pipeline (In Order)

Owing to the increase in processor circuit complexity and activity factor in an *n*-stage pipeline processor, the power threshold of this design in energy-harvesting systems is higher than that of the nonpipelined (NP) case.

We consider two strategies for backup in the pipelined design (see Figure 2). In the first scheme, shifted PC and volatile flip-flop (SPC/VFF), a shifter buffer is designed to remember the PC value in each pipeline stage. The unfinished PC to be backed up would then be in the data memory stage. In the second scheme, nonvolatile flip-flops (NVFF), the PC and RegFile are automatically backed up through NVM flip-flops in the instruction fetch/instruction decode (IF/ ID) pipeline stages.

Out-of-Order Processors

Although OoO processors are less frequently considered for low-power deployments because of their lower efficiency, the need of batteryless harvesting systems to greedily consume power when it is present can still make them a competitive option. Because of its higher activation requirements, an OoO processor will be less frequently active than the other two datapath designs, and it also contains more state to consider saving during power emergencies. We considered several policies for an OoO processor.
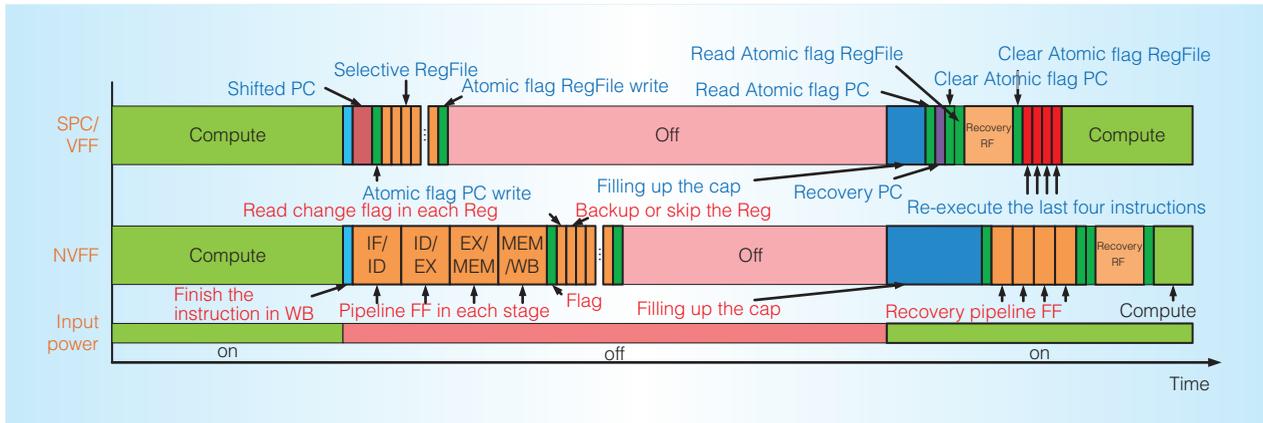
Figure 2. Runtime components for five-stage pipelined (5-SP) schemes. (SPC: shifted PC; NVFF: nonvolatile flip-flops; VFF: volatile flip-flops.)

Minimum-state resource backup solution (MinR). MinR backs up the minimal number of bits required to preserve functionality across power interruptions, including first uncommitted PC at the head of the reorder buffer (ROB), architectural RegFile, and map table. However, before backing up, some extra operations are needed to achieve a consistent state.

Low-latency backup solution (LLB). Rather than back up only the first uncommitted PC, this solution backs up the entire ROB, the instruction queue (IQ), ARegFile, map table, and PRegFile. Although LLB has more structures to be backed up than MinR, it can sometimes be more energy efficient because of the extra work required in MinR both pre-backup and post recovery.

Middle-level backup solution (MLB). Instead of using extra recovery time and energy to restore the ready table and free list in the LLB, MLB backs up the ready table and free list as well.

Min-state-lost backup solution. In this solution, all the structures are backed up, including the branch history buffer (BHT) and branch target buffer (BTB).

Incremental backup. This scheme combines two key insights. During power emergencies, power income often is still greater than zero, and if substantial capacitor energy remains after the minimum amount of state has been preserved, the system can continue backing up progressively less-essential microarchitectural

state to preserve performance after recovery, despite having a capacitor provisioned only for minimal backup margins. This could delay recovery from shorter power emergencies, because the capacitor might be more drained than it otherwise would have been.

Architecture Selection

Which nonvolatile architecture provides the best forward progress for a particular application scenario depends on various factors. The input power and the stability of the power supply are two key elements that impact the choice. In addition, the application's computational complexity and performance requirements are also important.

To evaluate our different design strategies, we developed a simulation framework for modeling both execution and power emergencies. We validated this framework, especially the system-level effects of power emergencies, against a fabricated NVP.[7] Our framework models energy consumption based on synthesized versions of all three pipelines (including an OoO design from Fabscalar[8]) in Synopsys Design Compiler with a 45-nm TSMC low-power library.

The input signal characteristics play a major role in determining the optimal design, as our experiments with Wi-Fi power trails under different environmental conditions made clear. Figure 3 demonstrates the performance of the various backup schemes when home and office Wi-Fi sources are used for harvesting energy. For the home environment, a nonpipelined ODSB architecture
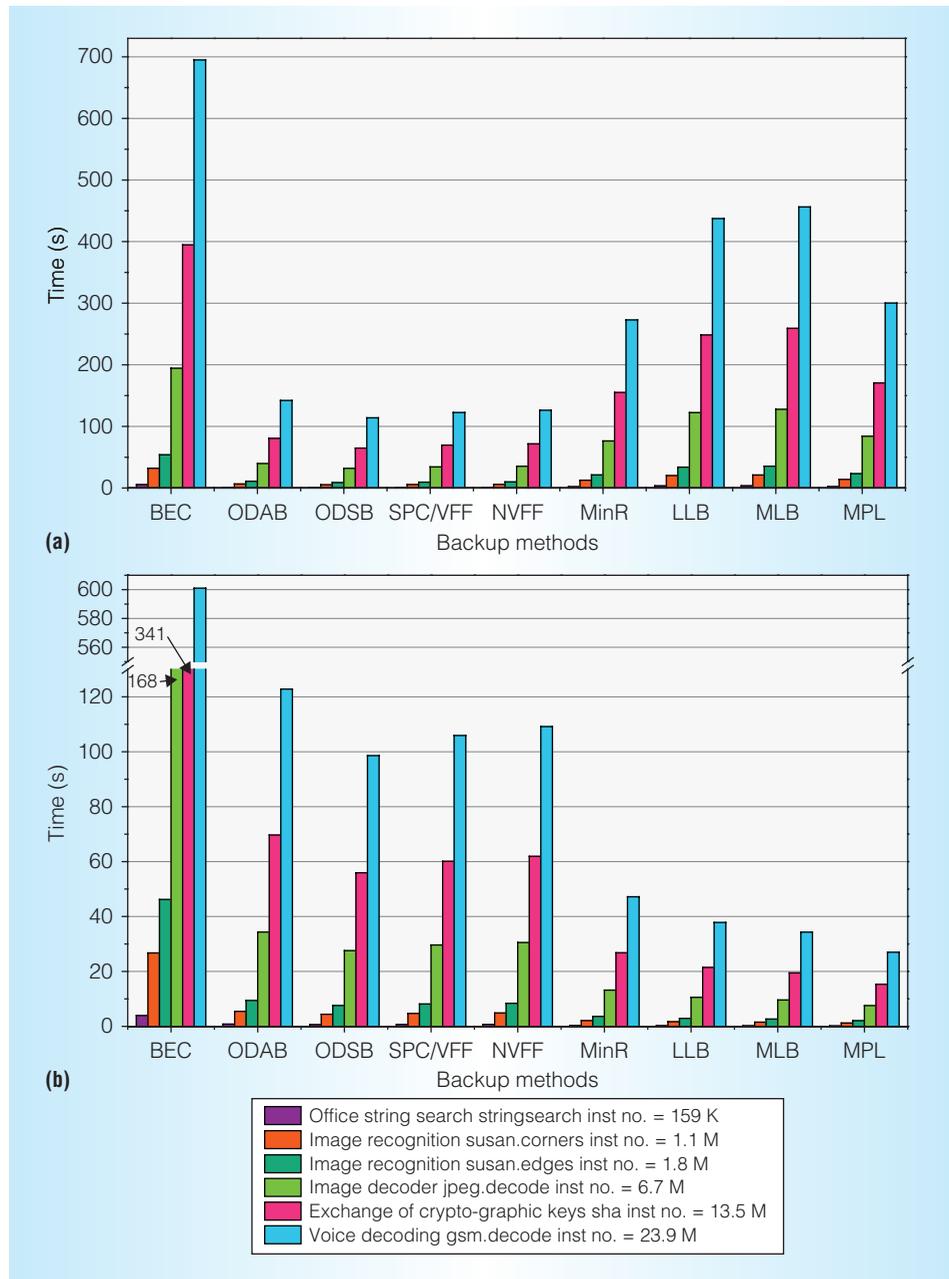
Figure 3. Execution time when harvesting Wi-Fi energy. (a) Home environment (with one router as the strongest and the others with relatively weak signals). (b) Office environment (with multiple routers of similar signal strength).

performs best, whereas in the office environment, the more complex OoO processor is desirable. This is because the home Wi-Fi signal typically comprises a single router, whereas the office environment usually comprises signals from more routers. A disturbance in the signal would result in input power going to almost zero in the home environment; thus,

the simplest design with the lowest power threshold is preferred. In contrast, in the office environment, the additional routers continue to supply input power at a relatively similar strength in an uninterrupted fashion, allowing for more complex architectures.

Here, we briefly summarize our findings when comparing architecture and policy pairs.
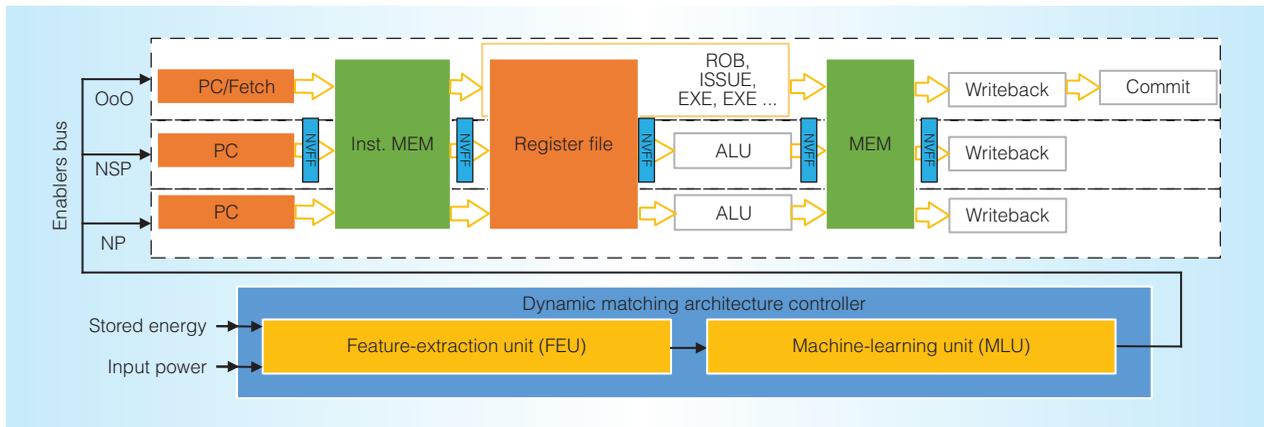
Figure 4. Dynamic matching architecture diagram with a machine-learning-based controller in the control path.

Of the NP policies, ODSB is the most energy-efficient strategy when the source is relatively stable (for example, solar energy). Compared to ODAB, ODSB can reduce the backup energy penalty by 69 percent with only 0.002 percent area overhead. Although BEC is not the most energy efficient, with very weak sources such as Wi-Fi, it does not require the time to accumulate energy in the capacitor to ensure sufficient backup energy is available. Thus, it is viable when the power failures are extremely frequent (less than 1 in 10 cycles), which rarely happens even in Wi-Fi sources.

For pipelined processors, we find that SPC/VFF requires 11 percent less time and 57 percent less energy than NVFF. An extra four clock cycles are needed to reexecute the last four instructions that are lost from the latter pipeline stages after recovery (we regard this as part of the recovery time penalty). However, only backing up one PC with a small shifter allows a smaller backup capacitor with lower leakage to be sufficient for SPC/VFF, which in turn affects the power threshold. In this case, SPC/VFF will also be able to outperform NVFF after several repeated instructions.

The OoO processor's viability and backup preference depend highly on the power profile, offering both substantial speedups and large slowdowns for different power input traces. In particular, the OoO processor backup policies are sensitive to the frequency of power emergencies. For the traces examined, the incremental backup approach was highly promising.

Smart Matching Architecture

The results of the previous section highlight how different architectures can achieve superior progress depending on the application and power profile features. This variable affinity also exists at finer temporal granularity within a power profile. To exploit this, we propose using a dynamic heterogeneous architecture with NP, n-stage pipelined, and OoO microarchitecture cores to fit different scenarios (see Figure 4).[9] The NP datapath has the lowest energy per instruction at our minimum considered frequency, giving it the lowest turn-on threshold and making it suitable for periods with minimally viable power incomes. For our NSP design, we employ pipelining not to achieve a higher frequency, but to achieve the same frequency as the NP design at a lower voltage. Finally, the OoO processor's greater performance can lead to greater overall progress despite less-frequent activation opportunities, making all three designs plausible selections for being the most suitable core for executing a portion of a program given varying input power.

At one time, only one microarchitecture is active, and a dynamic matching architecture controller selects the active core. The controller comprises two primary components: a feature extractor that accumulates the recent history of input power and power emergencies into a signature, and a prediction module that maps this signature to a selection. Because of the predictor's relatively high complexity compared to the simple NP and NSP datapaths, this controller is activated to

perform a prediction only once every 200 ms, thus limiting prediction overhead. The controller predicts if and when the microarchitecture should be switched, for example, from NP to OoO due to underexploitation of incoming power, or from OoO to NP due to excessive frequency of backup events. If the nonvolatile storage is augmented with double buffering, the same controller can also be used to make predictions on whether power emergencies will resolve before stored energy is depleted, eliding a backup operation. If the controller changes the current microarchitecture setting, it will need to take several steps to finish such a switching operation, and the transition overheads vary depending on which transition is occurring.

Switching among architectures requires several steps to be carried out under the control of the dynamic matching architecture controller. The complexity of the switch varies depending on which transition is occurring. For example, the NP to NSP or OoO transition proceeds through the following steps:

1. The controller ensures that there is enough energy storage to guarantee successful microarchitecture switching.
2. The controller gates the clock signal and waits for longer than one normal clock cycle to make sure that one instruction is finished for NP.
3. The PC indicating the next instruction address in instruction memory is shared from NP to NSP or OoO.
4. The register file is volatile and has already been updated by NP. The control signals of register files are now handed over from NP to NSP or OoO.
5. The data memory is nonvolatile and handed over from NP to NSP or OoO.
6. The PC part for NP, the arithmetic logic unit part for NP, and the writeback part for NP are all supply gated to avoid leakage.

The process of switching from NSP to NP or OoO is similar, albeit with a higher initial energy threshold. Key differences include potential reexecution of software if it is the oldest incomplete instruction and slight differences in PC backup. The NSP PC tracking hardware (SPC/VFF) can be integrated into both the NP and NSP pipelines to simplify transitions between the two datapaths.

Switching from OoO is more complicated. The minimum energy required for a switch is much higher than that of NP and NSP, because OoO needs to restore the original states changed by the uncommitted instructions. We resume on uncommitted the PC at the head of ROB and squash all other instructions regardless of status, leveraging existing branch misprediction pathways. When returning to OoO from NP or NSP, performance could initially be substantially lower than that observed during the preceding OoO execution period. This is because some metadata related to performance rather than correctness (for example, BHT and/or BTB) is lost.

Result and Discussion

Figure 5 compares a baseline execution of the NP core at 32 KHz against the dynamic matching architecture. For the same power profile, the forward progress of NSP is 1.08 times that of NP, and the forward progress of OoO is 1.32 times that of NP. We also test other power profiles and observe that the forward progress ratio of OoO to NP varies from 2.55 to 0.14 times. Because OoO requires a higher power and energy threshold, this ratio is highly dependent on the power sources and profiles. The power profile shown in Figure 5a is the power input sampled every 0.2 seconds per point in the ambient Wi-Fi environment. The Wi-Fi's variation is large because of the multiple channel effect, data transformation, obstacle movement, signal refraction, and reflections. The maximum temporal power can be 300 times larger than the minimum power. We also compare with a fixed OoO core, but we show only the NP baseline for clarity. The NP baseline shows continuous forward progress due to a lack of backup events for this particular power trace. Note, however, that system execution does not begin until there is sufficient stored energy for a successful backup.

Figures 5b and 5c show the stored energy, backup operations, and current processor
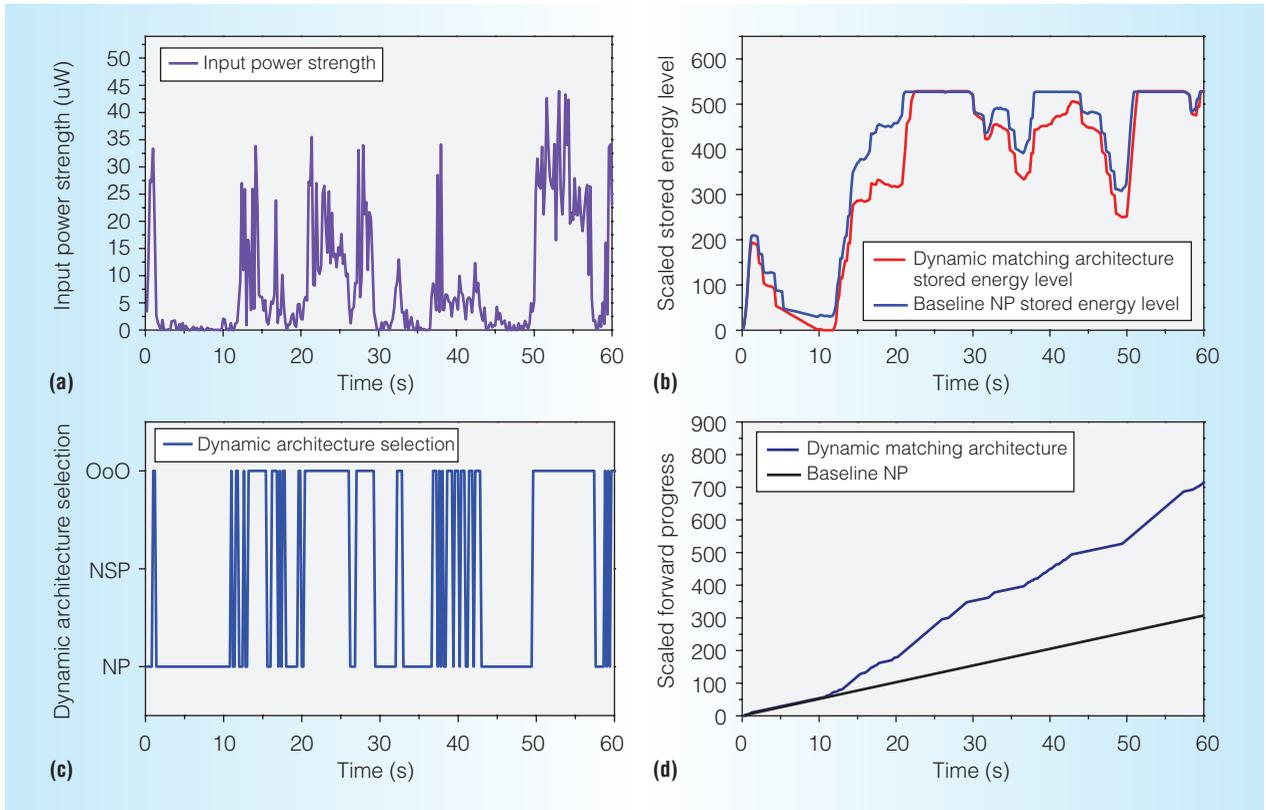
Figure 5. Baseline: a nonvolatile processor with only one NP microarchitecture core simulation result. (a) An example 1-minute power profile in an ambient Wi-Fi environment. (b) Scaled stored energy level in a 470 uF energy storage capacitor. (c) Scaled forward progress simulation results for dynamic matching architecture with nonpipelined (NP), *n*-stage pipelined (NSP), and out of order; the neural network comprises four hidden layers with 30, 10, 10, and 10 neurons in each hidden layer. (d) Stored energy level as inputs and backup number count. (e) Neural network outputs for the microarchitecture core selection. (f) Forward progress result for the dynamic matching architecture.

mode. For this power profile and training result, and the input power profile, NSP is not selected, and there are numerous switches between NP and OoO. As Figure 5b shows, the stored energy level is consumed more aggressively in the dynamic architecture than the baseline. This reduces the percentage of time when the capacitor saturates, which avoids energy losses due to insufficient energy-storage capacity. More aggressive consumption does come with a cost: Compared to the baseline with no backup operation, this dynamic matching architecture needs 13 backup operations. However, the net effect of consumption increase is strongly positive, turning more incoming energy into computation. This significantly increases the forward progress, achieving 2.4 and 1.82 times

the progress of the baseline NP and OoO architectures, respectively.

Nonvolatile-processor-based platforms can be an ideal enabler for the IoT and wearable devices. Some of the explored architectures have been adopted and verified through fabrication: for example, the proposed ODSB solution is applied in second-generation NVP.[2]

In the near future, we will explore how traditional techniques such as dynamic voltage and frequency scaling can be applied to NVP and how should it be adjusted. A hybrid architecture with dynamic resources could also be useful to adapt to variable power profiles. Rather than traditional architecture methods, a machine-learning-based controller proves to

predict in high quality in control path design. Accelerators for machine learning application-level algorithms based on software implementation or hardware implementation can even be merged for both the application and controller. New devices like the Tunnel-FET can also be applied to further reduce the power consumption for NVPs.[10,11] Novel distributed circuits merging both the computation and backup operations can further reduce the backup time and energy.[12]   MICRO

Acknowledgments

References

1. Y. Liu et al., "Ambient Energy Harvesting Nonvolatile Processors: From Circuit to System," *Proc. 52nd Ann. Design Automation Conf.*, 2015, article 150.

2. Y. Liu et al., "A 65nm ReRAM-Enabled Nonvolatile Processor with 6X Reduction in Restore Time and 4X Higher Clock Frequency Using Adaptive Data Retention and Self-Write-Termination Nonvolatile Logic," *Proc. IEEE Int'l Solid-State Circuits Conf.*, 2016, pp. 84–85.

3. K. Ma et al., "Architecture Exploration for Ambient Energy Harvesting Nonvolatile Processors," *Proc. Int'l Symp. High-Performance Computer Architecture*, 2015, pp. 526–537.

4. K. Ma et al., "Nonvolatile Processor Architecture Exploration for Energy-Harvesting Applications," *IEEE Micro*, vol. 35, no. 5, 2015, pp. 32–40.

5. K. Ma et al., "Nonvolatile Processor Optimization for Ambient Energy Harvesting Scenarios," *Proc. 15th Non-Volatile Memory Technology Symp.*, 2015; www.cse. psu.edu/~xzl3/resources/NVMTS2015-NVP. pdf

6. "MSP430FRxx FRAM Microcontrollers," Texas Instruments, 2016; www.ti.com/lsds/ ti/microcontrollers_16-bit_32-bit/msp/ultra-low_power/msp430frxx_fram/overview.page.

7. Y. Wag et al., "A 3μs Wake-up Time Nonvolatile Processor Based on Ferroelectric Flip-Flops," *Proc. European Solid-State Circuits Conf.*, 2012, pp. 149–152.

8. N.K. Choudhary et al., "FabScalar: Composing Synthesizable RTL Designs of Arbitrary Cores Within a Canonical Superscalar Template," *Proc. 38th Ann. Int'l Symp. Computer Architecture*, 2011, pp. 11–22.

9. K. Ma et al., "Dynamic Machine Learning Based Matching of Nonvolatile Processor Microarchitecture to Harvested Energy Profile," *Proc. IEEE/ACM Int'l Conf. Computer-Aided Design*, 2015, pp. 670–675.

10. X. Li et al., "RF-Powered Systems Using Steep-Slope Devices," *Proc. IEEE 12th Int'l New Circuits and Systems Conf.*, 2014, pp. 73–76.

11. H. Liu et al., "Tunnel FET RF Rectifier Design for Energy Harvesting Applications," *IEEE J. Emerging and Selected Topics in Circuits and Systems*, vol. 4, no. 4, 2014, pp. 400–411.

12. G. Sumitha et al., "Nonvolatile Memory Design Based on Ferroelectric FETs," to be published in *Proc. Design Automation Conf.*, 2016.

Kaisheng Ma is a PhD student in the Department of Computer Science and Engineering at Pennsylvania State University. His research interests include energy-harvesting architectures, machine learning, and neuromorphic computing. Ma received an ME in microelectronics from Peking University. Contact him at kxm505@cse. psu.edu.

Xueqing Li is a postdoctoral researcher in the Department of Computer Science and Engineering at Pennsylvania State University. His research interests include high-performance data converters, wireless transceivers, self-powered nonvolatile systems, and circuits and systems using emerging devices. Li received a PhD in electronics

engineering from Tsinghua University. He is a member of IEEE. Contact him at lixueq@cse.psu.edu.

Karthik Swaminathan is a researcher at the Reliability and Power-Aware Microarchitecture Group at the IBM T.J. Watson Research Center. His research interests include power-efficient and resilient systems, approximate computing, and cognitive architectures. Swaminathan received a PhD in computer science and engineering from Pennsylvania State University. Contact him at kvswamin@us.ibm.com.

Yang Zheng is a software engineer at Google. His research interests include computer architecture design and nonvolatile memory. Zheng received an MS in computer science and engineering from Pennsylvania State University, where he completed the work for this article. Contact him at yxz184@cse.psu.edu.

Shuangchen Li is a PhD student in the Department of Electrical and Computer Engineering at the University of California, Santa Barbara. His research interests include computer architecture, especially emerging nonvolatile memory. Li received an MS in electronic engineering from Tsinghua University. Contact him at shuangchenli@ece.ucsb.edu.

Yongpan Liu is an associate professor in the Department of Electronic Engineering at Tsinghua University. His research interests include nonvolatile computation, low-power VLSI design, emerging circuits and systems, and design automation. Liu received a PhD in electronics engineering from Tsinghua University. He is a member of IEEE; ACM; and the Institute of Electronics, Information, and Communication Engineers. Contact him at ypliu@tsinghua.edu.cn.

Yuan Xie is a professor in the Department of Electrical and Computer Engineering at the University of California, Santa Barbara. His research interests include computer architecture, electronic design automation, and VLSI design. Xie received a PhD in electrical engineering from Princeton University. He is a Fellow of IEEE. Contact him at yuanxie@ece.ucsb.edu.

John (Jack) Sampson is an assistant professor in the Department of Computer Science and Engineering at Pennsylvania State University. His research interests include energy-efficient computing, architectural adaptations to exploit emerging technologies, and mitigating the impact of dark silicon. Sampson received a PhD in computer engineering from the University of California, San Diego. He is a member of IEEE. Contact him at sampson@cse.psu.edu.

Vijaykrishnan Narayanan is a distinguished professor of computer science and engineering and electrical engineering at Pennsylvania State University. His research focuses on energy-efficient computing. Narayanan received a PhD in computer science from the University of South Florida. He is a Fellow of IEEE and ACM. Contact him at vijay@cse.psu.edu.

cn *Selected CS articles and columns are also available for free at http://ComputingNow.computer.org.*