NEOFog: Nonvolatility-Exploiting Optimizations for Fog Computing

Kaisheng Ma, Xueqing Li, Mahmut Taylan Kandemir, Jack Sampson, Vijaykrishnan Narayanan
Dept. of Computer Science and Engineering, The Pennsylvania State University
{kxm505,lixueq,kandemir,sampson, vijay}@cse.psu.edu

Jinyang Li, Tongda Wu, Zhibo Wang, Yongpan Liu
Dept. of Electronic Engineering, Tsinghua University
{lijy15,wtd14,wzb13}@mails.tsinghua.edu.cn
ypliu@tsinghua.edu.cn

Yuan Xie
Dept. of Electrical and Computer Engineering, University of California at Santa Barbara
yuanxie@ece.ucsb.edu

Abstract

Nonvolatile processors have emerged as one of the promising solutions for energy harvesting scenarios, among which Wireless Sensor Networks (WSN) provide some of the most important applications. In a typical distributed sensing system, due to difference in location, energy harvester angles, power sources, etc. different nodes may have different amount of energy ready for use. While prior approaches have examined these challenges, they have not done so in the context of the features offered by nonvolatile computing approaches, which disrupt certain foundational assumptions. We propose a new set of nonvolatility-exploiting optimizations and embody them in the NEOFog system architecture. We discuss shifts in the tradeoffs in data and program distribution for nonvolatile processing-based WSNs, showing how nonvolatile processing and non-volatile RF support alter the benefits of computation and communication-centric approaches. We also propose a new algorithm specific to nonvolatile sensing systems for load balancing both computation and communication demands. Collectively, the NV-aware optimizations in NEOFog increase the ability to perform in-fog processing by 4.2X and can increase this to 8X if virtualized nodes are 3X multiplexed.

ACM Reference Format:
Kaisheng Ma, Jinyang Li, Tongda Wu, Zhibo Wang, Xueqing Li, Yongpan Liu, Yuan Xie, Mahmut Taylan Kandemir, Jack Sampson, and Vijaykrishnan Narayanan. 2018. NEOFog: Nonvolatility-Exploiting Optimizations for Fog Computing. In *Proceedings of 2018 Architectural Support for Programming Languages and Operating Systems (ASPLOS'18).* ACM, New York, NY, USA, 15 pages. https://doi.org/http://dx.doi.org/10.1145/3173162.3177154

1 Introduction

As the integration of a new technology percolates through systems of increasing complexity, new optimization opportunities are consistently found in its wake. For example, research developments in non-volatile elements, such as ReRAM [3], STT-RAM [78], and PCRAM [16], have led to use patterns for nonvolatile memories [82] distinct from their volatile counterparts and enabled nonvolatile logic-compatible elements such as NV-DFFs [39, 64] and non-volatile SRAM [9] that can support distributed on-chip backup and restore operations. These developments, in turn, have led to the exploration of nonvolatile processor architectures [42, 50] (NVPs) that rely on these integrated nonvolatile circuits to provide new guarantees for intermittently powered execution. As an increasing number of NVPs have been proposed and several have now been fabricated [28, 43, 69, 79, 91] with varying feature sets, new opportunities will arise as components of existing multi-node systems are replaced by their nonvolatile counterparts.

Energy-harvesting wireless sensor networks are a major sub-domain of the *Internet of Things* (IoT), and many such systems have already been successfully deployed. The applications that make use of this paradigm are diverse, including area monitoring, e.g., the position of the enemy; environmental monitoring; industrial monitoring; medical and health-care monitoring; traffic control systems; underwater acoustic sensor networks; and near-body wearable device networks. Despite their diversity, most systems operating on energy harvested from ambient power sources share a common core design pattern in their operation as *normally-off systems* (NOS). First, each node is designed with a large super-capacitor or rechargeable battery capable of storing enough energy to perform at least one complete unit of work. Then, in deployment, the node waits, harvesting energy, until this energy storage device is sufficiently charged before starting a simple micro controller (MCU) to oversee the collection of a data sample from the sensor, and, after very limited processing, transmits the sampled data. While there

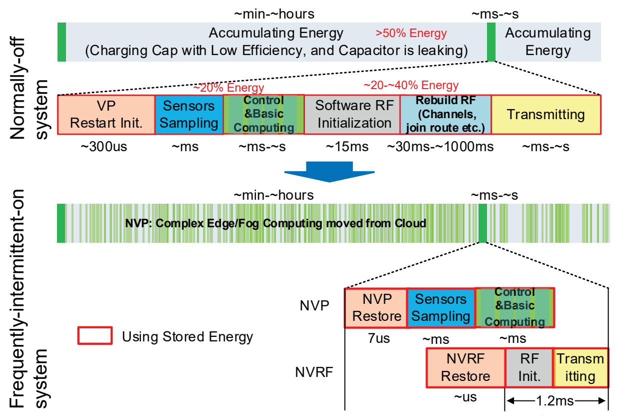

Figure 1. Optimizing from normally-off system (NOS) to frequently-intermittent-on system (FIOS)

is increasing research interest in moving more computation to sensing platforms [17, 18, 32, 37, 56, 57, 73], these systems have traditionally avoided relying on complex local computing in energy harvesting nodes based on the following two assumptions: 1) energy harvesting power is unreliable, so shorter work periods are preferable for reliability and cost/form-factor reductions in necessary energy-storage capacity, and 2) the net effect of redundancy and recomputation mechanisms for circumventing the unreliability of energy income is that both computation and communication at the sensor node are energy expensive, constraining both work done at data collection and the topologies of the deployed networks. Table 1 demonstrates examples of currently deployed energy-harvesting sensor applications built on this model.

However, advances in both hardware [48, 49] and software [12, 36, 44, 45, 56, 61, 90] management of integrated nonvolatile resources to more effectively compute and communicate [80] in intermittent power environments warrant reconsidering these high-level assumptions. Computation under unreliable power supply is now relatively reliable on processors with managed, integrated non-volatile storage (e.g. NVPs). In addition, NVPs have been shown to make better forward progress than their volatile counterparts given the same incoming power, i.e., NVPs are not only more reliable, but are also more efficient than their predecessors (if only in unstable power environments). Recent works that look beyond the processor to leverage nonvolatility in the communication path of these platforms have shown that acceleration [80] of the initialization of the RF module through the introduction of a nonvolatile RF controller (NVRF) can substantially reduce the time and energy cost for data transmission. Collectively, these advances have weakened prevailing assumptions about the cost and reliability of computation and communication at the sensor node level. Thus, it is likely that the traditional optimizations for collections of such nodes that aim to limit on-node computation as a first-order goal may no longer be effectively capitalizing on the current opportunities within these distributed systems, and new algorithmic and system level redesigns should be explored for the next generation energy harvesting based wireless sensor network systems.

To understand the potential changes stemming from adopting nonvolatile nodes, we have deployed and measured various real NVP-based systems. To explore how utilizing the node level nonvolatile features affects system tasks, we introduce optimizations that leverage NVP and NVRF efficiency and reliability to trade increased computation for reduced transmission, changing the system from a *normally-off system* (NOS) to a frequently-intermittently-on system (FIOS), as shown in Figure 1. In addition, we propose an efficient load balancing approach for NV-mote chains under certain wireless protocols. We then explore new optimizations, driven by realistic user requirements [14, 20, 21], that leverage the features of the platforms with both NVPs and NVRF (NV-motes), specifically NVRF state-share among nodes, to allow adding node virtualization to improve quality-of-service (QoS). Analyzing the benefits of the new approaches, we combine these discussed features to propose the **NEOFog** architecture for the next generation fog computing.

This paper demonstrates that, by exploiting node-level nonvolatility, the NEOFog approach can provide new optimization opportunities and improve system performance and efficiency due to the following contributions:

- Introducing nonvolatility into nodes improves the computation efficiency and reduces the energy for data transmission. Accordingly, we optimize the programs from RF-energy-dominating to computation-intensive, and from normally-off to frequently-intermittently-on, so as to better utilize the new opportunities brought by nonvolatility.
- Considering (i) the imbalance and time-varying income power and stored energy level of each node and (ii) different energy requirements of different workloads, we propose a distributed load balancing algorithm specially optimized for "unstable power supply" in an intra-chain level to balance the loads and further increase the computation done in the fog. We show that it can increase the amount of data processed in-fog by 1.9X to 2.1X compared to a system that employs a traditional load balancing algorithm.
- We propose architectural support for improving QoS via increasing nodes virtualization and slotted time-division multiplexing enabled by the NVRF in a fashion not previously viable. A nearly 2X QoS improvement is observed for a 3X multiplexing in low energy income situations.

The remainder of the paper is organized as follows. Section 2 provides an overview of energy-harvesting platforms and introduces the new features brought by integrating nonvolatility into different parts of the system. Section 3 presents the technical details of our proposed NEOFog architecture. Section 4 describes our simulation infrastructure, and Section 5 gives results from our experimental evaluations. We discuss the relevant prior work in Section 6 and conclude in Section 7.

Existing System	Energy Source	Sensors	Network Topology	Transmitted Data
Bridge Health Monitor[17, 18]	Solar, Piezoelectric	Accelerometers, piezo-sensors	Zigbee Chain Mesh	Raw sampled data
Wearable UV Meter[32, 37]	Solar	UV sensor	Star	Raw data
Joint-less Railway Temp. Monitor[22]	Solar	Multiple temperature sensors	Zigbee Chain Mesh, GPRS	Raw uncompressed data
Machine Health Monitor [34, 83]	Piezoelectric, thermal, RF	3-axis accelerometer, vibration sensors, temperature	Star, bus or tree	Raw data
RF Powered Camera[56, 57, 73]	RF Source, WiFi	Image sensor	Point-to-point backscatter	Raw image pixels

Table 1. Functionality and components of current energy harvesting WSN system

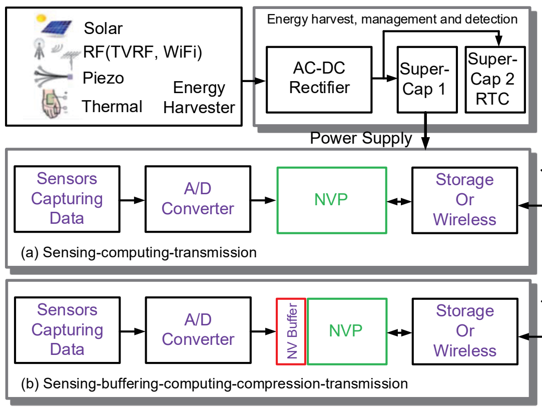

Figure 2. NVP-based energy-harvesting system organization.

2 System Model

In this section, we first introduce a typical energy harvesting system. We then explain, step by step, how nonvolatility changes the processor, buffer, and RF. Lastly, we discuss network topology and construction.

2.1 Traditional wait-compute systems

System components Figure 2(a) illustrates a block diagram for a typical NVP-based WSN node powered by energy harvesting. There are four types of ambient energy that are widely available and relatively easy for commodity systems to harvest: 1) solar, via photovoltaic cells; 2) RF via antennas; 3) piezoelectronics, via vibrations of either the substrate or entity to which the harvester is attached; and 4) thermal energy, via thermal gradients across a thermoelectric. All these energy sources lack stability, varying with different locations, angle, etc. of the node's dynamic environment. Front-end circuit design is specific to the AC or DC characteristics of the input source. Capacitors are adopted to temporarily store the harvested energy. The rest of the node consists of data sensing, ADC, NVP, storage, as well as transceiver.

In our platform, two super-capacitors are employed, one for powering the real-time clock chip, which is utilized to synchronize the wireless communication, and another one for powering the rest of the node. The real-time clock super-capacitor has a higher charging priority because if it loses power entirely, then, when the system recovers, resynchronizing with the logical time slots in the network imposes large overheads compared to normal state restoration.

Wait-compute system timing and energy Figure 1 (upper) shows the typical execution pattern for a NOS, in which most of the time is spent accumulating energy into the capacitor. The system starts only when there is sufficient energy stored in the capacitor. System activation begins with processor initialization for about 300us, followed by sensor sampling under the control of the processor, then activation of the RF parts, whose initialization penalty is large, and finally raw data are sent out.

One version of the WispCam project [57], powered by RF, is a typical system example using this approach. In the described deployment, the system first accumulates energy for 15 minutes (5m away from the RF source), and then starts the system for three seconds. Of the three seconds system-on time, only 115ms is spent for data sampling, and the rest is for data transmission under the control of the processor. Due to long charging time with capacitor leakage, as well as low charging efficiency, more than half of the energy income is wasted. Sensing consumes around 20% energy, and data transmission and computation consume 20-40%, even though WispCam uses the backscatter wireless technology, which is extremely energy efficient. Similar properties have been observed in our own observations on the systems shown in Table 1, wherein the RF parts, even using Zigbee, dominate the energy consumption. The dominance of communication motivates our study of leveraging the improved computation efficiency of an NV-mote to shift the costs from communication domain to computation domain and adopt a FIOS rather than NOS paradigm.

2.2 Nonvolatility in NV-motes

NVP Nonvolatile processors [43, 58, 62, 85] have emerged as a promising solution for intermittent unstable power under various energy scenarios. Through distributed fast and efficient nonvolatile logic including NVSRAM, NVFF, etc. [9, 39, 64], the computation state within the processors are backed up with an on-chip capacitor and restored when power recovers. NVPs can still achieve forward progress under power failure frequencies which are as high as 100kHz [43], belying the notion that an unstable power supply produces unreliable task completion, even if ensuring timeliness remains challenging.

The FIOS approach can improve system efficiency over NOS by reducing the inefficiencies involved with charging and discharging an energy storage device at the cost of the overheads of supporting backup and recovery in an NVP. Prior research [47] has indicated that replacing a volatile processor operating as a NOS with an NVP and a FIOS approach can increase forward progress by 2.2X to 5X (depending on the power profile at hand), making this an appealing set of tradeoffs.

Beyond the basic NVP operations seen in fabricated designs, dynamic policies have been proposed to further improve the forward progress of NVPs [48, 49]. In this work, we assume that the NVP implements the Spendthrift [49] frequency scaling and resource allocation architecture to provide efficient conversion of incoming energy into completed work.

NVBuffer To ensure consistent data transmission between the sensors and the NVP, Figure 2(b) shows the modification to the nodes to incorporate a nonvolatile buffer (NVBuffer) to guarantee asynchronous data transmission. NVBuffer, which is usually implemented as an NV FIFO, also provides a raw data buffer for load balance between nodes.

NVRF Traditional nodes operating as NOS require reinitializing the radio frequency (RF) transceiver before transmission, as all configuration and data in the transceiver are lost when a power failure occurs. Traditional software-based re-initialization leads to large overheads due to a long initialization delay between the host processor (whether VP or NVP) and the RF module, which consists of a high performance baseband core (about 10mW in ML7266 or ML7396 zigbee chipset) for computing wireless protocols and logic and analog parts exhibiting high power draw in TX or RX mode (between 30-100mW). Figure 3(a) shows the delay path in conventional software RF based initialization. The data stored in NVM (note that, in a traditional VP, this is a separate flash module) are taken out through the bus to the processor that, after some processing of the data, sends the processed initialization data back through the bus to the public SPI interface, and then to the RF transceiver. Measurement shows that the whole initialization process can take as much as 33ms, during which the RF module dissipates enormous energy.

Nonvolatile radio frequency controllers (NVRF) [80] represent an attempt to employ a hardware-controlled nonvolatile IO interface to support fast and efficient initialization of a specified peripheral. By offloading the RF initialization task to the NVRF controller to speed up the long delay between processor and peripheral, 27X speedup is observed (1.2ms) [80]. In fabricated NVRF, the NVRF controller stores the configuration of RF chips in a nonvolatile register file and initialize the RF chip under the control of a finite state machine following special protocols of RF chips; both ML7266 and ML7396 are supported. Figure 3(b) diagrams NVRF initialization. Data from the NV registers where the configuration information and the latest transmission data are stored are sent through direct memory access with special designed SPI interface to the RF transceiver. In addition, NVRFs can self-reinitialize once configured by the processor, which totally frees the processor from controlling the RF during many tasks. Even during the periods where the RF is not activated in FIOS, the processor can store the data to be sent and any associated configuration information into the NV data buffer and control registers. Once the NVRF is triggered by a timer or through a control signal from the processor, it sends data out without further intervention.

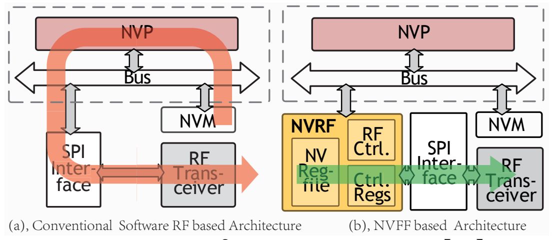

Figure 3. Software RF v.s. NVRF [80]

In summary, the presence of an NVRF controller speeds up initialization by 27X through quick and processor-independent response in a direct nonvolatile memory access (DNVMA) fashion. Reducing the stand-by time of the RF module, whose stand-by power is enormous, saves substantial energy, and prior measurements [80] indicate that NVRF achieves a 6.2X throughput advantage over software-based RF control.

2.3 Network Topology and Construction

A problem inherent with unstable power supplies is coordinating both sending and receiving nodes to be active at the same time. To address this, we employ a real-time clock (RTC) powered by a super-capacitor as shown in Figure 2(a). The RTC coordinates a common notion of time such that the synchronized sets of sending and receiving nodes can be coactive. The RTC wakes up once in every predefined interval, and as a result, once synchronized, all the nodes in the network with sufficient energy would wake up at the same time to transmit and receive data, and to again synchronize the RTC. For those nodes without sufficient energy to wake up at the RTC-indicated time, they will wakeup at a multiple of the RTC-indicated time rather than when they first accumulate sufficient energy. If, however, a node exhausts all its stored energy, meaning that even the RTC is desynchronized, the node will wake up whenever it has sufficient power in order to attempt to re-connect and synchronize with the whole cluster network. Another possible solution for the root problem would be to use an RF wakeup sensor [25, 35], but this has not been implemented in our work. In this work, we focus on load imbalance of energy income among nodes and the computation requirements of the whole system, which, while in some cases changing how and when the network is used, do not introduce new network architectures.

With respect to network topologies, there are many prior works discussing various self-organized cluster networks [15, 23, 29, 30, 33, 59, 60, 71, 76]. It has been shown that nodes preferring to communicate primarily with others in physical proximity can save transmission energy, and relay transmission can also save energy. We regard this as the MAC layer, which is transparent to programmers and schedulers. The most often used network topologies include star, bus, tree and mesh, as can be seen in the systems in Table 1. Although a mesh topology is adopted in the bridge monitoring and joint-less railway temperature monitoring systems, the network works like a chain mesh [18] due to the physical locations of the nodes along the railway or bridge. Our proposed intra-chain load balancing and inter-chain node virtualization algorithms specifically optimize these chain mesh systems.

3 Distributed Fog Computing

In this section, we explore various approaches for supporting and enhancing distributed fog computing on NVP-based systems powered by energy-harvesting. We group our approaches into (1) *Node* level optimizations for increasing compute capability, (2) *Intra-chain* level distributed fog computing load balancing schemes, and (3) *Inter-chain* level optimizations to enable leveraging slotted time-multiplexing node virtualization for QoS.

3.1 Node Level - Reoptimizing for an NV-mote

Integrating nonvolatility into nodes or their individual components, and managing hardware resources accordingly, has been introduced in a wide range of prior efforts [24, 38, 46, 66, 81, 86–88]. In this work, we focus on the system-level integration of NV-motes and, at the single node level, consider optimizations on the execution of system work sequences to better match the FIOS paradigm rather than the previous NOS paradigm – thereby, making better use of the features offered by an NV-motes nonvolatile components. Specifically, traditional programing in energy-harvesting WSN deployments tries to limit reliance on local computation. However, recent advances in handling intermittency make computation functionally reliable [4, 5, 10–13, 42, 44, 46, 50, 55, 74, 75] and more efficient [48, 49], while NVRF [80] and backscatter [27, 41] techniques significantly reduces the portion of energy contributed by the RF module. These shifts can be exploited by altering the work sequences performed during the active periods and by increasing the amount of local processing performed, such as tasks off-loaded from cloud, data compression and merging, in order to limit the communication costs. An effective node-level design for NV-motes should follow the rules of clearly splitting the computation energy source and policy to improve the efficiency of the computation.

Take an actually deployed "Bridge Health Monitor" system in Table 1 for example, a naive sensing-computing-transmission bridge cable node samples and transmits raw acceleration, etc. data of about 300-500MB per day. Then, these data are analyzed in-cloud, performing noise removal, FFT, etc. to produce strength models in order to monitor the strength of bridge cable, which can be calculated through harmonic and vibration. Each step is called a "task" in the paper, and all bridge cable nodes run homogeneous tasks in the actual scenario.

To improve the efficiency of the computation, we can shift computation from the cloud to the fog that includes combination of 3-direction acceleration into one cable-vertical direction vibration, noise removal, FFT, strength calculation in three different bridge structure-specialized models, temperature and humidity noise removal, temperature and humidity compensation of each model's results, and calculation of the average strengths and data compression. Because the processed data are only the strength data with less variation than original acceleration etc. raw data, the compression has a good compression ratio. So the local-computing can dominate the computing time and energy rather than compression.

Figure 4 shows differences in the sequences of tasks performed in a completely volatile node, a node with a nonvolatile processor, and a NEOFog NV-mote. The volatile node (VP) and nonvolatile node operate as NOS whose supporting front-end circuits are designed with a single super capacitor for energy storage, as shown in Figure 5(a). The system is active only when there exists sufficient stored energy to perform all the tasks in the sequence at the top of Figure 4. This includes using the processor to initialize the volatile RF using software. In the VP, this takes about 15-100ms and building the connection requires 30ms-1s before data can be transmitted. Replacing a VP with NVP can bring 2.2X-5X forward progress benefits [47]. The NOS NVP's startup time is much shorter, 32us, compared to VP. Due to the direct restore of the RF states with the help of NVP, the data transmission time reduces to 33ms. However, the NVP benefits were not maximized in this wait-compute scheme [47].

To support the FIOS operation, the front-end circuits need to be enhanced, as shown in Figure 5(b). More specifically, adding SW1 allows a direct source-to-load unstable power channel to the NVP. These front-end circuit design concepts were originally proposed by Wang et al. [77], improving the front-end efficiency to 90%, and further developed by Sheng et al. [70] taking into account node-level considerations. Further investigations along similar directions include optimizations for rectifier [8] and mix-source design [52]. Our work leverages, but does not substantially advance, the front-end design literature. However, we do shift task sequencing and power source dependency to better utilize the direct source-to-load advantages of Figure 5(b). To optimize for the FIOS operation, we propose the sequence shown at the bottom of Figure 4. Application computations off-loaded from the cloud and other complex local programs are executed in an intermittent fashion (dashed-line boxes), increasing their effective computation efficiency via a leaner front-end conversion ratio, while the other system activities (red-line boxes) are still powered from the capacitor. NVBuffers are designed to sit between the sensors and the NVP, as well as within the NVRF, to guarantee reliable asynchronous data transfers. By adopting more computation locally, like local processing and complex data compression, lower data transmission can be expected. By this means, the proposed FIOS clearly divides data sample/transmission and computing, and sufficiently utilizes the high efficiency feature brought by NVP via the support of direct-dual-channel front-end circuits. To the best of our knowledge, this is the first work that addresses system-level rearchitecting to benefit from NVP+NVRF at node level.

3.2 Intra-chain Level - Load Balancing in FIOS

At any given time, there can be substantial variation in both energy income and reserves among the nodes in the network due to variations in the physical deployment and environment, as well as history effects. Local efficiency optimizations that perform frequency and resource scaling in the NVP [49] can further exacerbate these variations. Similarly, any task

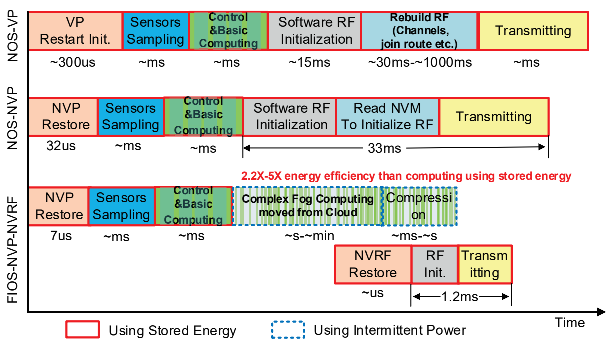

Figure 4. Time details of node level.

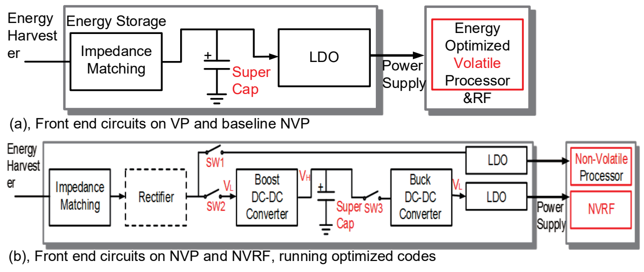

Figure 5. Front-end circuits for wireless nodes.

or resource heterogeneity at the node level (e.g., differences in node-level provisioning across generations in a network with nodes of different deployment ages or data-dependent or position-dependent software execution) would also amplify variations in energy reserve levels over a given time period. Considering the imbalances stemming from both unbalanced load distribution and varying node-level energy-constrained computation capabilities, *we desire to design a network balance method to make the nodes that harvest ample energy do more computation than nodes with limited energy, and allocate the tasks to the most efficient rather than inefficient nodes.*

Efficiency-Oriented Load Balance for FIOS An unstable power supply presents challenges for load balancing in an energy harvesting system. Since the nodes may or may not have sufficient energy to start, traditional top-down load-balancing approaches can fail to reach regions of the network. Specific to energy harvesting systems, the computational efficiency varies substantially depending on local energy income, and thus, even when scheduling uniform tasks to nodes with uniform hardware, the dynamic computational efficiency of the target nodes must be an optimization goal for energy harvesting networks (in addition to load balance). Moreover, the capabilities of a given node are time-varying at fine granularities and may shift after scheduling. Distinct from similar scheduling scenarios in data centers, each node generates its own inputs through distributed sensing of raw data and application code is often small enough to be pre-distributed to all nodes. Due to these different distributed system level requirements, a new scheduling algorithm and supporting architecture is deployed to fit into the specific needs of FIOS.

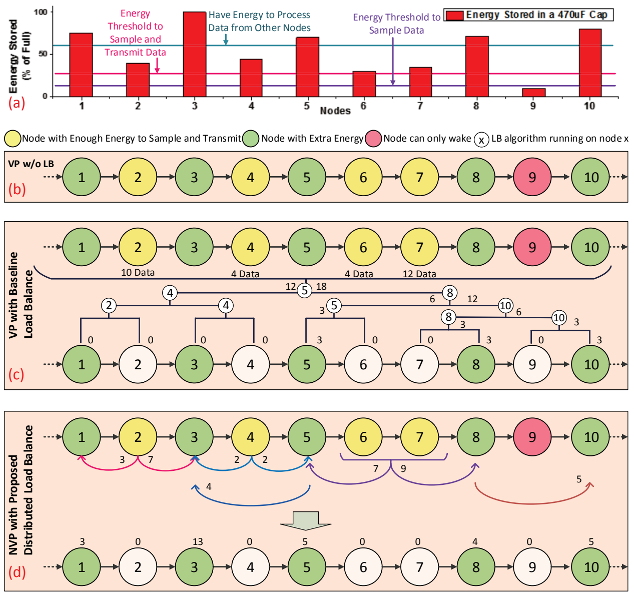
Energy Stored (% of Full)
Energy Threshold to Sample and Transmit Data
Have Energy to Process Data from Other Nodes
Energy Threshold to Sample Data
Energy Stored in a 470uF Cap
Nodes
Node with Enough Energy to Sample and Transmit
Node with Extra Energy
Node can only wake
LB algorithm running on node x
VP w/o LB
VP with Baseline Load Balance
10 Data
4 Data
12 Data
NVP with Proposed Distributed Load Balance

Figure 6. Illustration of distributed load balance algorithms

Distributed Load Balancing Algorithm Figure 6 demonstrates several approaches to load balancing. Figure 6(a) shows the stored energy thresholds that separate the different classes of potential actions a node could take in the next interaction phase: Yellow stands for nodes with enough energy to sample and transmit data; Green stands for nodes with extra energy beyond yellow ones; Red nodes do not have sufficient energy to communicate. Nodes with stored energy level lower than "energy_threshold_to_sample_data" are effectively dead for the current sampling period, so they are not shown in Figure 6(b). Absent load balancing, efficiency is very low, as can be observed in Figure 6(b) – the available energy and energy required are unbalanced. Volatile nodes with baseline up-down multi-level tree load balance may not fully alleviate energy imbalances: Figure 6(c) shows an up-down binary scheduling that is only partly achieved (left 12 tasks are all missed) when the assigned node 4 running parts of the load balance is low on stored energy.

We propose a distributed load balancing algorithm tailored for energy harvesting scenarios. Based on available energy at each node, a node determines how much extra energy it has available for any tasks beyond what it expects processing its own raw data will require. The available energy as well as NVP configuration (frequency and resource state for the Spendthrift policy [49]) are shared with other nearby nodes in the local network chain. For example, node 4 can know states of its left node 3 before touching another energy hungry node 2, and node 5 on the right in the first round. Based on the energy available to left or right nodes, node 2 estimates the execution time of its tasks to be distributed, and builds an array $\langle a_1, a_2, \ldots, a_n \rangle$ and array $\langle b_1, b_2, \ldots, b_n \rangle$ to stand for shortest time running the same tasks on either best efficiency nodes on the left side or right individually. Then, Algorithm 1 is called to compute an "assignment result".

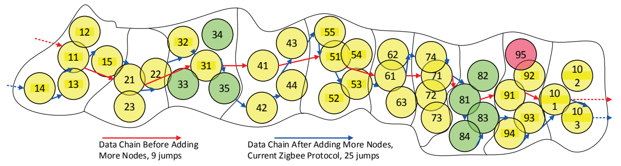
Data Chain Before Adding More Nodes, 9 jumps
Data Chain After Adding More Nodes, Current Zigbee Protocol, 25 jumps

Figure 7. Naive density increase does not boost Zigbee QoS

Based on the result, for instance, two tasks from node 4 are assigned to node 3, and another two to node 5. A second call to Algorithm 1 may happen when the assigned tasks require more energy than one node has already stored or beyond *MAXTIME* - load balance call interval. In our example, node 8 is over assigned, a second call distributes 5 tasks to node 10. While, in the worst-case, several distribution rounds may be required to achieve a balanced load and optimality is not guaranteed, this distributed bottom-up algorithm is expected, in practice, to produce fewer, and more local, data transmissions. Note that if load balance algorithm is interrupted, no load balance will take place at that region. This failure affects performance, but not functionality of the network, and is modeled in our simulation framework. After balancing, the data transmission begins, tasks are computed, and results are transmitted at during the next power-on period.

3.2.1 Node Implementation of Load Balancing

Let us define $OPT(i,k)$ as the first k tasks that can be finished within time i by the most efficient node on the left and within time $OPT(i,k)$ on the right. The specific task indexed as k can be finished by either left or right. If the k_{th} task is finished by left, then the right's total time does not change; thus, we have:

$$OPT(i,k) = OPT(i - left[k], k-1) \tag{1}$$

On the other hand, if the k_{th} task is finished by right, then the left's total time does not change; thus, we have:

$$OPT(i,k) = OPT(i, k-1) + right[k] \tag{2}$$

By combining both the situations, we have:

$$\begin{aligned} OPT(i,k) = min(&OPT(i - left[k], k-1), \\ &OPT(i, k-1) + right[k]) \end{aligned} \tag{3}$$

Algorithm 1 is implemented as an interrupt-driven program in the node software. It is a dynamic-programing approach with three steps: build the table, find the minimum time required to finish all the tasks, and generate the assignment. The algorithmic complexity of the approach is $O(n * MAXTIME)$, which is task number*load balance call interval.

3.3 Inter-chain Level - Node Virtualization for Enhanced QoS (NVD4Q)

Naively increasing node count and density will not improve QoS. Figure 7 shows one example of zigbee protocol. When there are only 10 nodes (Node 11, 21, ... , 101) in the system, the network topology is as the red line. When the node density in the area increases 4x, naive zigbee protocol opts for locality in transmission distance and increases the number of jumps from 9 to 25 when transmitting data from node 11 to 101.

In order to improve the system performance while using the existing RF protocol (zigbee in this work), we propose a slotted time-multiplexing node virtualization for QoS algorithm (NVD4Q), shown in Algorithm 2, that leverages the capabilities of the NVRF to support new behaviors. When a new node is added to an existing network, it will first open its NVRF to search for the closest node and then clone that node's NVRF state, as shown in Figure 3(b). It will then synchronize its timer with the network. However, rather than

Algorithm 1: Distributed Load Balancing

```
Input: Time cost if n task running on the nodes on the left
       ⟨$a_1, a_2, \ldots, a_n$⟩; Time cost if n task running on the nodes on
       the right ⟨$b_1, b_2, \ldots, b_n$⟩; $MAXTIME$: load balance call
       interval;
Output: Out ⟨$o_1, o_2, \ldots, o_n$⟩: task assignment to either nodes on
        the left or right
$n \leftarrow Sizeof(a)$
$p \leftarrow Zeros(MAXTIME, n)$
$sa \leftarrow 0$
#build the table
for $k = 1 \rightarrow n$ do
    $sa+ = a[k]$
    for $i = 1 \rightarrow sa$ do
        $p[i,k] = p[i,k-1] + b[k]$
        if $i \geq a[k]$ then
            if $p[i,k] < p[i-a[k], k-1]$ then
                $p[i,k] = p[i,k]$
            else
                $p[i,k] = p[i-a[k], k-1]$
#find the minimum time
$minTime \leftarrow \infty$
for $i = 1 \rightarrow sa$ do
    if $p[i,k] \geq i$ then
        $temp = p[i,k]$
    else
        $temp = i$
    if $minTime > temp$ then
        $minTime = temp$
        $AtimeFinal = i$
        $BtimeFinal = p[i,k]$
#generate the assignment output
$i \leftarrow AtimeFinal$
for $k = n \rightarrow 1$ do
    if $i \geq a[k]$ then
        if $p[i,k-1] + b[k] < p[i-a[k], k-1]$ then
            $Out[k] = "right"$
        else
            $Out[k] = "left"$
    else
        $Out[k] = "right"$
return $Out$
```

Algorithm 2: Node virtualization for QoS Algorithm

```
1  Always open the NVRF
2  Find the closest node through NVRF
3  Copy its states of NVFF in NVRF controller and NVM
4  Synchronize the timer, then turn off NVRF
5  while Timer reaches a pre-set time do
6      Update or not update wake-up interval time
7      Start NVRF to send or receive data
8      if Received special command
9      then
10         Go to line 2
11     Shut down NVRF w/o backup of NVRF control states
```

waking up every tick of the RTC timer, it will receive an initial (phase) offset in ticks (unique among the clones of the same node) and a pre-set tick count between activations (common among all the clones) which are only updated when requested by software. Software can thereby manage the effective sampling frequency of the collection of the cloned nodes in order to match the sampling frequency needed by the particular application for the single logical node that the cloned nodes implement via time-multiplexing based on the number of clones for a given node. Membership to a set of clones is updated at a programmer-defined frequency specific to both the application and deployment scenario. For instance, a bridge monitor system may have nearly permanent clone set memberships, while a mountain sliding monitoring system may update at a low frequency and sensor nodes on moving objects would frequently request network reconstruction, including re-association of clones.

With this NVD4Q, we expect to achieve the behaviors shown in Figure 8. Collectively, these behaviors are expected to increase the number of samples processed by each logical node. These include: At each wake-up period, only nodes with a common phase (10 in the example) wake up. Nodes in chain 1 to 5 wake up consecutively, but chain 6's nodes are totally different from chains 1 to 5. From the network's perspective, the network structure and information does not change during power off period; so, because of the NVRF, there is no reconstruction penalty required for the network. Note that each node continuously accumulates energy in its own super capacitor, and as a result, chances that one node runs out of energy become lower. Each node in a collection of clones activates less frequently than a baseline node (by a factor equal to the number of clones), and hence, the energy to be dissipated is smaller.

4 Simulation Methodology

We built and measured real WSN prototype platforms to quantify the energy distribution of energy harvesting and NVP-based WSN systems. Each node consists of an NVP running at 1MHz, consuming 0.209mW, sensors specific to different applications, and a Zigbee based RF module with a data rate of 250kbps, consuming an average of 89.1mW when

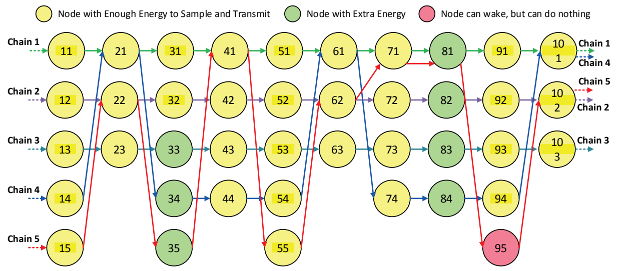

Figure 8. NVD4Q Node virtualization for QoS expected effects

transmitting. The power varies depending on the configuration of the TX strength, ranging from 72mW to 102mW. The RF module, when working in the RX mode, typically consumes 72mW.

Our simulation framework consists of two parts. The first part is a node-level functional simulator, at the core of which is a modified 8051 RTL [43], which has been calibrated with physically fabricated nodes [50]. The node-level functional simulator also includes two parts, one matlab/python part starts the nodes' simulation, calculating the input power, energy, stored energy, efficiency etc. and calls the Modelsim Verilog part (step by cycle) to run function simulation. The node-level simulation is cycle-accurate, and our modeling of NVRF is calibrated against a fabricated device [80]. Capacitor size selection and according policy are modeled in work [48]. Power and stored energy sampling supporting circuits (including ADC's power) and penalty are also modeled [49] with more features in sensors such as accelerometer LIS331DLH, image sensor LUPA1399, temperature sensor TMP101, etc. For example, for TMP101, the initialization costs 566ms, and one time sampling costs 0.283ms. For Zigbee chip with volatile RF and traditional method, ML7266 initialization costs 531ms (Host MCU@1MHz), and data transmission of N Bytes costs $(255 + 1.44 * N + 0.032 * N)$ms, while zigbee chip ML7266 with NVRF module only costs 28ms as initialization and $(1.74(NVRF\ start) + 0.156 + 0.216 * N + 0.032 * N)$ms as data transmission. The measured node was built with the FIOS mode, not directly replacing WISP with NVP.

The second part of our framework is a WSN system-level simulator. It starts thousands of node simulators at a time, and provides WSN system-level status. The front-end circuit efficiencies, stored energy level, etc. at system level are modeled in previous work [47]. The time running one specific program is simulated in Verilog to determine clock cycle counts. RF energy is computed as the integration of power over that specific time period considering active, idle, and OFF modalities (89.1mW at TX and RX, 14.93mW at IDLE).

Various failures are modeled in our simulation framework. First, at a node level, two kinds of failures are modeled: node failures caused by depletion of energy, and packet failures caused by wireless transmission affected by weather and channel interference. Transmission success parameters are from an experiment with constant-light powered 3-mote point-hop-router transmission (A⟶B⟶C) mounted on top of a building, with a distance of 10-15m and 10 days consecutive data collection. Packet loss rate 0.75% was observed (totally 14400 A→C packets were expected), mainly affected by weather, especially rain. Packet transmission success rate between two sufficiently powered nodes was therefore modeled as 99.25%.

Second, at intra-chain network level, for a 3-mote transmission example (A→B→C), when B fails to start due to energy shortage, âĂIJorphan_scanâĂİ function in Zigbee stack is called in A to broadcast, C sends unicast to A to confirm ("Scan_confirm" status is success), following with an update of "AssociatedDevList". So, A→C. When B recovers, B broadcasts, A adds B in its "AssociatedDevList" and removes C, C join B, and finally A→B→C. At inter-chain level, the "AssociatedDevList" as well as other info are duplicated within virtual nodes to avoid rebuilding the network structure. "RSSI" which exists in every data packet, is the signal strength, which is used to find the closest neighbors. All these latencies and energy are measured in real ML7266 NVRF based nodes, and are then modeled.

To test the effectiveness of the whole simulation framework, the deployed "bridge health monitor system" and "wearable UV meter system" in Table 1 are used to validate the baseline NVP simulations. Other systems with different sensors and functions are also built, whose measured results are presented in Table 2.

Due to node count limitations in currently deployed systems using fabricated NVPs and NVRF chips, as well as the fact that the proposed load balancing and virtualization for QoS (NVD4Q) policies need some architectural-level support that is not yet present in the fabricated devices, the load balance and the NVD4Q results are simulated. Our simulator runs thousands of single-node simulators simultaneously (1000 for intra-chain simulation, and 1000 to 5000 for inter-chain simulation). Each node has different power inputs. The communication is mimicked by direct data transmission under a certain successful transmission possibility through virtual buffers among nodes. Of the simulated thousands of nodes, 10 consecutive nodes' information is shown as the presented example in the paper for simplicity.

5 Results and Discussion

In this section, we evaluate the impact of our individual schemes on WSN quality and then quantify the contributions due to individual techniques employed. Further, we combine them together and evaluate under both dependent (time and location correlated) and independent power trace scenarios. To quantify WSN output quality, we employ the following metrics: counts of node wakeups, successfully processed samples, and samples processed in the fog. For total data processed in the fog, the data compression, decompression and other post-optimized operations not present in the baseline system are not included. Only tasks offloaded from the cloud are considered as data processing (i.e., work that would otherwise have been done at the cloud).

5.1 Single Node Energy Distribution

Table 2 shows the measured parameters of each system. Five diverse deployments are examined, including bridge health monitoring, UV meter (data capturing only), temperature, acceleration sensing, and heartbeat signal pattern matching. Two strategies are deployed: naive sensing-computing-transmission and sensing-buffering-computing-compression-transmission. In the former strategy, the single node starts itself to sense the data, which are normally of relatively small size. Since the computation involves mostly sensing and simple data processing, the instruction counts in Table 2 are also light. After processing the captured data, the node performs the transmission sequence, including starting the RF module, initialization, and channel setup. The computation ratio indicates the percentage of NVP energy consumption among all node components. We observe that, for some applications, computation does not play an essential part in energy consumption as the transmission energy dominates. For some other applications; however, especially in the pattern matching, the NVP computation is substantial, consuming up to 59.5% of node energy.

The other approach considered employs sensing-buffering-computing-compression-transmission, except when there is a real-time request from a control node. Instead of immediately processing and transmitting the captured data, it is buffered in a 64kB nonvolatile memory buffer designed to support this policy, as shown in Figure 2(B). Once the buffer is full, it triggers an interrupt of the NVP to process the buffered data. If the node lacks energy to process or send the buffered data out, the sampled data are discarded. "Incidental Computing" techniques [47] have been proposed to mitigate this. Two observations are noticed. Firstly, we find that, for some applications, more complex data processing requires data from neighboring samples in time – e.g., the pattern matching in heartbeat monitoring. Thus, buffering and operating on the local sequence as a batch reduces data to be transmitted. Secondly, compression (bzip or jpeg depending on application) before data transmission can reduce the data size to $3\% - 14.5\%$ of its original. The many repeated patterns in data, especially in that sensed by WSNs, foster high data compression rates. However, compression is not free; it requires a large amount of computation energy as shown in Table 2. To compute the total energy savings, we use the following formulas:

$$E_{naive} = (E_{naive}^{comp} + E_{naive}^{trans}) * 64k \tag{4}$$

$$E_{new} = E_{new}^{comp} + E_{new}^{trans} \tag{5}$$

$$Energy\ saved\ ratio = (E_{new} - E_{naive})/E_{naive} \tag{6}$$

Note that the naive method repeatedly samples a single data item and sends it out, while the new method samples and buffers until 64k data, then processes them together. Although compression costs energy, total energy is saved through less energy spent in RF initialization and smaller data size to transmit. Through the optimization of the buffered strategy, we observe that computation energy dominates across applications, ranging from 91.5% to 98.5%.

Method	*Naive sensing-computing-transmission*				*Sensing-buffering-complex local computing-compression-transmission*			*Comparison*
App.	**Inst. NO.**	**Compute energy(nJ)**	**TX energy(nJ)**	**Compute ratio**	**Compute energy(mJ)**	**TX energy(mJ)**	**Compute ratio**	**New method energy saved**
Bridge Health	545	1366.86	22809.6	5.65%	81.7	6.95	92.2%	-55.2%
UV Meter	460	1153.68	5702.4	16.8%	108.3	6.8	94.1%	-48.8%
WSN-Temp.	56	140.448	5702.4	2.4%	75	6.99	91.5%	-57.1%
WSN-Accel.	477	1196.316	17107.2	6.53%	83.6	6.59	92.7%	-54.9%
Pattern Matching	1670	4188.36	2851.2	59.5%	345.1	5.39	98.5%	-24.1%

Table 2. Measured energy distribution on different platforms using two different strategies. Parameters in the table stand for parameters and energy during two-time data transmission interval.

Compared to the naive strategy, the buffered strategy can save 24.1% – 57.1% total energy. In the actual deployed systems, physical buffers are present, but both strategies are utilized to balance between energy savings and fast response times, although the buffered strategy dominates running times.

5.2 Performance of Distributed Load Balancing

Figure 9 shows the stored energy level of three consecutive nodes in a chain when powered with a solar panel in the daytime. As can be seen in Figure 9(top), the stored energy level of VP node 1 without load balancing during the 0-50 min period indicates that the capacitor was frequently full, meaning further energy was rejected by the system. Using an NVP node with baseline load balancing reduces the stored energy by balancing the loads among nodes. Employing the proposed distributed load balance further reduces the stored energy level. A similar situation can also be observed for Nodes 2 and 3, indicating that both load balancing approaches succeed in avoiding the capacitor overflow scenarios seen without load balancing by moving work to neighboring nodes.

5.2.1 Performance with Independent Power Income

Since the load balance is intra-chain, the power income level among nodes plays an important role for the intra-chain load balance. The more stored energy variation among the nodes, the more impact from the proposed algorithm. We use solar power traces from a forest to mimic deployment scenarios. With winds, depending on whether the leaves are moving, the power variation among nodes is large, and effectively independent.

We consider a forest fire monitoring system. Such a system may have as many as 1k to 1M nodes, but we present data for only the 10 nodes of one chain. Given uninterrupted power and no failures, these 10 nodes could ideally deliver 15000 data packages in 5 hours. Each power trace is a synthetic trace generated from data collected by a single energy-harvesting node in both tree-covered and open environments under different time and weather conditions. Individual node power incomes are generated by concatenating sequences from the measured traces in random order. The computation performed when performing in-fog offload is a reconstruction kernel for a volumetric map based on point samples.

Figure 10 shows the simulated number of wakeups, total processed data packets, and in-fog processed packets for five different power traces over three different systems. The VP nodes do not perform fog-offloading, and without load balancing, despite 13656 node wakeups (1344 node failures due to energy depletions), only capture and transmit 2664 packets. With a higher activation threshold, NVP nodes running the baseline load balance only exhibit 12383 wakeups, but improve the total packets processed to 3236 and increase in-fog processing to 3045. NEOFog wakeups are similar to the baseline NVP, but substantially improve the total packets processed to 5582, 37% of the ideal of 15000, and increase in-fog processing to 5018 packets.

5.2.2 Performance with Dependent Power Income

To explore situations where the power observed by each node is strongly correlated with its neighbors, we consider the power observed from daytime solar traces at fixed locations on bridges in a bridge monitoring system. Each power trace is a synthetic trace generated from the same 5-hour period from 5 different days of measured bridge data [1]. Individual node power incomes are generated by applying 30% random variance to the measured data traces. The computation performed when performing in-fog offload is based on the structural health monitoring algorithms in [7, 84]. Again, under an ideal power and communication scenario, the maximum number of processed packets would be 15000.

Figure 11 shows the simulated wakeups, processed packets, and in-fog processed packets for bridge monitoring under a set of highly dependent power traces. In a system with dependent power profiles, stored energies exhibit lower variance, and thus the load balance scheme is activated less frequently. The amount of energy required to transmit data from one node to another decreases, partially compensating for reductions in actual load-balancing. This is why the dependent values are within 10% of those for the independent power traces.

For the dependent power traces in the bridge monitoring scenario, similar to the prior forest scenario, the VP nodes do not perform fog offloading, and without load balancing, despite still performing 13886 node wakeups, only captures and transmits 2494 packets. With a higher activation threshold, NVP nodes running the baseline load balance again exhibits reduced wakeup counts of around 12859, but improves the

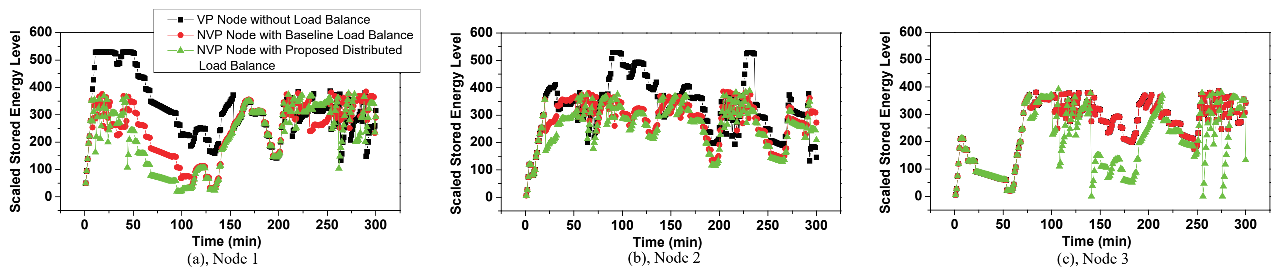

Figure 9. Stored energy level of 3 consecutive nodes in a chain

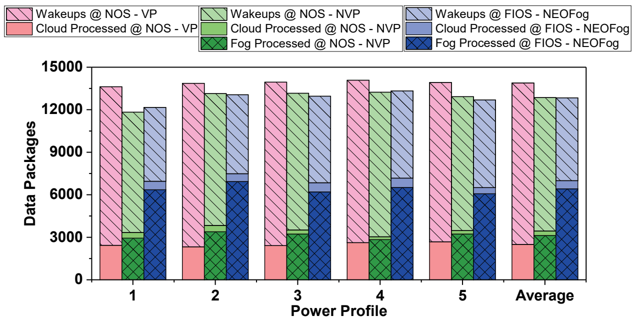

Figure 10. Total data packages captured by the nodes in the network, and total data packages processed by the fog, when the power traces are ample and independent

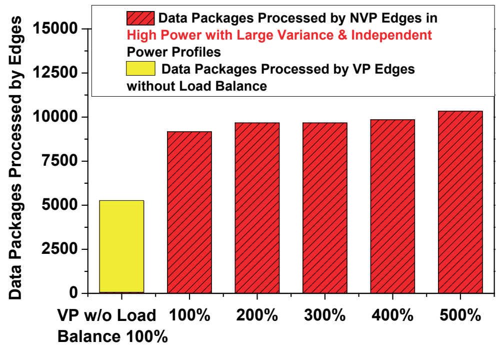

Figure 12. Increasing node multiplexing in an environment with high power with large independent variance

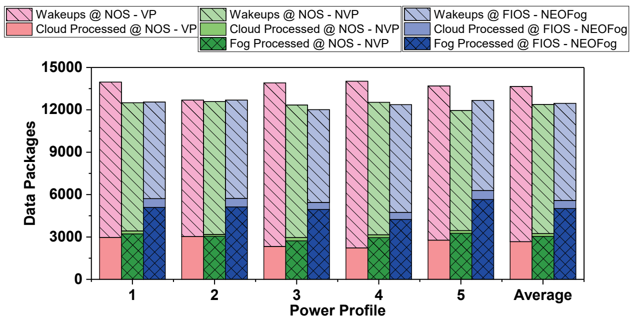

Figure 11. Total data packages captured by the nodes in the network, and total data packages processed by the fog, when the power traces are ample and dependent

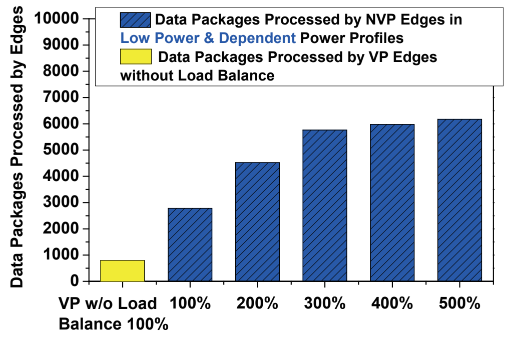

Figure 13. Increasing node multiplexing in an environment with very low power and dependent variation

total packets processed to 3439 and increases in-fog processing to 3126. NEOFog wakeups remain similar to the baseline NVP, but substantially improve the total packets processed to 6990, 46.6% of the ideal of 15000, and increases in-fog processing to 6418 packets.

If we compare the NEOFog system with nodes running distributed load balance to VP nodes without load balance and NVP nodes with baseline load balance, the average gains of 2.8X and 2.0X gains are seen in the total network output, respectively, for average gains for a high variance, independent deployment and 2.1X and 1.7X gains are seen for a dependent power scenario. While this indicates that the proposed load balance is less effective in dependent power conditions, it still vastly outperforms the baseline and sans-balancing systems.

5.3 NVD4Q - Virtualization for QoS Results

We consider a mountain sliding monitoring system as our running example for NV4DQ, in which solar-powered nodes are randomly distributed in the area to be monitored via aerial dispersion. Some of them will have excellent sun exposure, while others may fall into grass or shrubs or be poorly oriented. On a sunny day (which can be regarded as high power with large variance), the network collects around 12000 samples, and a traditional VP system without load balancing can process about 5000 data packages in-fog, as shown in Figure 12. The NVP nodes with the proposed load balancing can process around 9500 data packages, almost double the baseline. Since the in-fog processing rate is already very high, the NV4DQ approach provides minimal gains.

However, the highest concern for the events the system is designed to monitor, slides, occurs during heavy rains when solar energy is very limited. We show the results for operation during inclement weather in Figure 13. A traditional VP system without load balancing can process only 725 data packages in-fog, while the NEOFog system with identical node count can process 2800 packages in-fog. Increasing the node count and multiplexing rate when using NV4DQ substantially improves the in-fog processing rate, up until 300% multiplexing is reached, by allowing longer times for each node to accumulate the less-available energy and from less frequent startup overheads for each node. Benefits saturate at 3X multiplexing for this experiment, because total successful sampling under the reduced power conditions reduces to 8000.

6 Related Work

Nonvolatility in Processors. By designing distributed nonvolatile logic or elements at a microarchitectural level, computation state can be checkpointed before power outages occur with only on-chip energy storage and without programmer intervention [6, 53, 54, 63, 67]. Various materials can implement the nonvolatile features, e.g., FeRAM based NVPs [28, 79, 91], ReRAM based NVPs [43], and MRAM (magnetoresistive random access memory) based NVPs [69]. Beyond these, there are other types of NVPs [26, 58, 72], and NV associative processors [51] as well. NVP efficiency has been improved by frequency scaling [48] and dynamic resource control [49] for NVPs. Many cross-layer works leveraging integrated non-volatility to address intermittency have also been explored, including, OS and high-level synthesis approaches [55, 75], programming language and compiler approaches [12, 66], HW/SW approaches [24], and software-based approaches [4, 5, 74]. While this work relies on the existence of NVPs, it focuses on how to benefit from their use at system level rather than the design of a new NVP.

Nonvolatility in RF control and other low-power RF technologies. Nonvolatile RF control was first proposed by Wang et al. [80], and it boosts the startup speed of a commercial Zigbee module by 27X. Other point-to-point techniques like backscatter [27, 41] are also promising techniques to reduce RF energy in nodes, and there are many protocol implementations that aim to implement lower-power RF communication [31, 40, 65, 68]. Any of these protocols would reap further benefits from the NVRF approach. In this work, we emphasize how to develop new network-level strategies that exploit the ability to clone NVRF states and its superior latency and throughput for cheap node virtualization via time-multiplexing without changing the network protocols.

Load Balance on WSN Load balancing on WSNs, aims to match the work to be done with the nodes that have sufficient energy to perform it. Many prior load balance methods have been proposed, including weight-based approaches [19, 89], package forwarding [33], pseudo-sink protocol [59], and dynamic route [15] balancing. Some works use partitioned clusters for load balance [29, 60, 71]. A decentralized routing algorithm, known as a game theoretic energy balance routing protocol is proposed by Abd et al. [2]. All these approaches target battery-powered devices rather than energy-harvesting nodes with highly unstable power supply. Our proposed balancing algorithm is specifically designed for high failure rates, even during the balancing algorithm itself. To this end, it eschews global communication and optimization, and aims instead to reduce the scope of the shared information and the associated transmission costs.

Virtualization for QoS Wajgi et al. [76] proposes backup nodes for the cluster head after it triggers a threshold. Other works also propose inter-cluster level optimizations by optimizing the protocol and network topology [23, 30]. In contrast, our proposed NVD4Q policy shares states in NV elements in NVRF, and thus, the (virtual) network topology does not change. Note that NVD4Q is not a load balance at inter-chain level. Rather, it enhances the QoS via each (physical) node having more time to accumulate energy before communicating, enhancing the success rate of each virtual node.

7 Conclusion

The maturation of NVPs and other integrated nonvolatile elements brings new optimizing opportunities at system level design as prior assumptions regarding the key tradeoffs and design principles for energy-harvesting systems are challenged. To address this, in this paper, we revisit the core operating paradigm of normally-off systems, and instead, reoptimize for the frequently-intermittently-on systems enabled by NVPs and NVRFs. We show that integrating nonvolatility into nodes can improve the amount of computation offloaded to the fog, rather than performed in the cloud, and that applying NVP-specific distributed load balancing can further increase fog computing capability. We also provide a node virtualization technique that exploits NVRF advantages to increase fog computing QoS in lower power-income conditions. Collectively, these optimizations increase fog-offload capabilities by 4.2X at baseline deployment node count and up to 8X at 3X multiplexing.

Acknowledgments

We would like to thank the anonymous reviewers and our paper shepherd, Prof. Vijay Janapa Reddi at University of Texas at Austin, for providing constructive suggestions and insightful feedback. This work was supported in part by NSF ASSIST, NSF expeditions of Visual Cortex on Silicon, and also by LEAST, one of the six SRC STARnet Centers, sponsored by MARCO and DARPA, as well as NSF 1213052/1500848/1719160/1409095/1626251/1439057. This work was also supported in part by NSFC Grant 61674094 and the Beijing Innovation Center for Future Chip.

References

[1] 2016. Measurement and Instrumentation Data Center (MIDC). (2016). http://www.nrel.gov/midc/.

[2] Mehmmood A Abd, Sarab F Majed Al-Rubeaai, Brajendra Kumar Singh, Kemal E Tepe, and Rachid Benlamri. 2015. Extending wireless sensor network lifetime with global energy balance. *IEEE Sensors Journal* 15, 9 (2015), 5053–5063.

[3] Hiroyuki Akinaga and Hisashi Shima. 2010. Resistive random access memory (ReRAM) based on metal oxides. *Proc. IEEE* 98, 12 (2010), 2237–2251.

[4] Domenico Balsamo, Alex S Weddell, Anup Das, Alberto Rodriguez Arreola, Davide Brunelli, Bashir M Al-Hashimi, Geoff V Merrett, and Luca Benini. 2016. Hibernus++: a self-calibrating and adaptive system for transiently-powered embedded devices. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 35, 12 (2016), 1968–1980.

[5] Domenico Balsamo, Alex S Weddell, Geoff V Merrett, Bashir M Al-Hashimi, Davide Brunelli, and Luca Benini. 2015. Hibernus: Sustaining computation during intermittent supply for energy-harvesting systems. *IEEE Embedded Systems Letters* 7, 1 (2015), 15–18.

[6] Paul Bogdan, Miroslav Pajic, Partha Pratim Pande, and Vijay Raghunathan. 2016. Making the Internet-of-things a Reality: From Smart Models, Sensing and Actuation to Energy-efficient Architectures. In *Proceedings of the Eleventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES '16)*. ACM, New York, NY, USA, Article 25, 10 pages.

[7] Fernando Cerda, Siheng Chen, Jacobo Bielak, James H Garrett, Piervincenzo Rizzo, and Jelena Kovacevic. 2014. Indirect structural health monitoring of a simplified laboratory-scale bridge model. *Smart Structures and Systems* 13, 5 (2014), 849–868.

[8] I. Chaour, S. Bdiri, A. Fakhfakh, and O. Kanoun. 2016. Modified rectifier circuit for high efficiency and low power RF energy harvester. In *2016 13th International Multi-Conference on Systems, Signals Devices (SSD)*. 619–623.

[9] Pi-Feng Chiu, Meng-Fan Chang, Shyh-Shyuan Sheu, Ku-Feng Lin, Pei-Chia Chiang, Che-Wei Wu, Wen-Pin Lin, Chih-He Lin, Ching-Chih Hsu, Frederick T Chen, Keng-Li Su, Ming-Jer Kao, and Ming-Jinn Tsai. 2010. A low store energy, low VDDmin, nonvolatile 8T2R SRAM with 3D stacked RRAM devices for low power mobile applications. In *VLSI Circuits (VLSIC), 2010 IEEE Symposium on*. IEEE, 229–230.

[10] Alexei Colin, Graham Harvey, Brandon Lucia, and Alanson P Sample. 2016. An energy-interference-free hardware-software debugger for intermittent energy-harvesting systems. *ACM SIGPLAN Notices* 51, 4 (2016), 577–589.

[11] Alexei Colin, Graham Harvey, Alanson P Sample, and Brandon Lucia. 2017. An Energy-Aware Debugger for Intermittently Powered Systems. *IEEE Micro* 37, 3 (2017), 116–125.

[12] Alexei Colin and Brandon Lucia. 2016. Chain: tasks and channels for reliable intermittent programs. In *Proceedings of the 2016 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications*. ACM, 514–530.

[13] Alexei Colin, Alanson P Sample, and Brandon Lucia. 2015. Energy-interference-free system and toolchain support for energy-harvesting devices. In *Proceedings of the 2015 International Conference on Compilers, Architecture and Synthesis for Embedded Systems*. IEEE Press, 35–36.

[14] Brian D Collins and Randall W Jibson. 2015. *Assessment of existing and potential landslide hazards resulting from the April 25, 2015 Gorkha, Nepal earthquake sequence.* Technical Report. US Geological Survey.

[15] Yaping Deng and Yaming Hu. 2010. A load balance clustering algorithm for heterogeneous wireless sensor networks. In *E-Product E-Service and E-Entertainment (ICEEE), 2010 International Conference on*. IEEE, 1–4.

[16] Xiangyu Dong, Naveen Muralimanohar, Norm Jouppi, Richard Kaufmann, and Yuan Xie. 2009. Leveraging 3D PCRAM technologies to reduce checkpoint overhead for future exascale systems. In *High Performance Computing Networking, Storage and Analysis, Proceedings of the Conference on*. IEEE, 1–12.

[17] Charles R Farrar and Keith Worden. 2012. *Structural health monitoring: a machine learning perspective.* John Wiley & Sons.

[18] Andrew Gastineau, Tyler Johnson, and Arturo Schultz. 2009. Bridge Health Monitoring and Inspections–A Survey of Methods. (2009).

[19] Gaurav Gupta and Mohamed Younis. 2003. Load-balanced clustering of wireless sensor networks. In *Communications, 2003. ICC'03. IEEE International Conference on*, Vol. 3. IEEE, 1848–1852.

[20] Fausto Guzzetti, Alberto Carrara, Mauro Cardinali, and Paola Reichenbach. 1999. Landslide hazard evaluation: a review of current techniques and their application in a multi-scale study, Central Italy. *Geomorphology* 31, 1 (1999), 181–216.

[21] Haoyuan Hong, Wei Chen, Chong Xu, Ahmed M Youssef, Biswajeet Pradhan, and Dieu Tien Bui. 2017. Rainfall-induced landslide susceptibility assessment at the Chongren area (China) using frequency ratio, certainty factor, and index of entropy. *Geocarto International* 32, 2 (2017), 139–154.

[22] Gao Huifang, Ma Kaisheng, and Zhang Wenchao. 2011. The real-time temperature measuring system for the jointless rail. In *Measuring Technology and Mechatronics Automation (ICMTMA), 2011 Third International Conference on*, Vol. 3. IEEE, 902–906.

[23] N Israr and I Awan. 2006. Multi-hop clustering algo. For load balancing in WSN. *International Journal of SIMULATION* 8, 1 (2006).

[24] Hrishikesh Jayakumar, Arnab Raha, and Vijay Raghunathan. 2014. QuickRecall: A low overhead HW/SW approach for enabling computations across power cycles in transiently powered computers. In *VLSI Design 2014, 13th International Conference on Embedded System and 27th International Conference on*. IEEE, 330–335.

[25] Haowei Jiang, Po-Han Peter Wang, Li Gao, Pinar Sen, Young-Han Kim, Gabriel M Rebeiz, Drew A Hall, and Patrick P Mercier. 2017. 24.5 A 4.5 nW wake-up radio with- 69dBm sensitivity. In *Solid-State Circuits Conference (ISSCC), 2017 IEEE International*. IEEE, 416–417.

[26] W. k. Yu, S. Rajwade, S. E. Wang, B. Lian, G. E. Suh, and E. Kan. 2011. A non-volatile microcontroller with integrated floating-gate transistors. In *2011 IEEE/IFIP 41st International Conference on Dependable Systems and Networks Workshops (DSN-W)*. 75–80.

[27] Bryce Kellogg, Aaron Parks, Shyamnath Gollakota, Joshua R Smith, and David Wetherall. 2014. Wi-Fi backscatter: Internet connectivity for RF-powered devices. In *ACM SIGCOMM Computer Communication Review*, Vol. 44. ACM, 607–618.

[28] S. Khanna, S. C. Bartling, M. Clinton, S. Summerfelt, J. A. Rodriguez, and H. P. McAdams. 2014. An FRAM-Based Nonvolatile Logic MCU SoC Exhibiting 100% Digital State Retention at VDD= 0 V Achieving Zero Leakage With < 400-ns Wakeup Time for ULP Applications. *IEEE Journal of Solid-State Circuits* 49, 1 (Jan 2014), 95–106.

[29] Hye-Young Kim. 2016. An energy-efficient load balancing scheme to extend lifetime in wireless sensor networks. *Cluster Computing* 19, 1 (2016), 279–283.

[30] Namhoon Kim, Jongman Heo, Hyung Seok Kim, and Wook Hyun Kwon. 2008. Reconfiguration of clusterheads for load balancing in wireless sensor networks. *Computer Communications* 31, 1 (2008), 153–159.

[31] Y. J. Kim, H. S. Bhamra, J. Joseph, and P. P. Irazoqui. 2015. An Ultra-Low-Power RF Energy-Harvesting Transceiver for Multiple-Node Sensor Application. *IEEE Transactions on Circuits and Systems II: Express Briefs* 62, 11 (Nov 2015), 1028–1032.

[32] Chong-Min Kyung, Hiroto Yasuura, Yongpan Liu, and Youn-Long Lin. 2016. *Smart Sensors and Systems: Innovations for Medical, Environmental, and IoT Applications*. Springer.

[33] Endre László, Kálmán Tornai, Gergely Treplán, and János Levendovszky. 2011. Novel load balancing scheduling algorithms for wireless sensor networks. In *The Fourth Int. Conf. on Communication Theory, Reliability, and Quality of Service, Budapest*. 54–49.

[34] Jay Lee, Fangji Wu, Wenyu Zhao, Masoud Ghaffari, Linxia Liao, and David Siegel. 2014. Prognostics and health management design for rotary machinery systemsâĂŤReviews, methodology and applications. *Mechanical systems and signal processing* 42, 1 (2014), 314–334.

[35] S. H. Lee, Y. S. Bae, and L. Choi. 2016. The design of a ultra-low power RF wakeup sensor for wireless sensor networks. *Journal of Communications and Networks* 18, 2 (April 2016), 201–209.

[36] Hehe Li, Yongpan Liu, Chenchen Fu, Chun Jason Xue, Donglai Xiang, Jinshan Yue, Jinyang Li, Daming Zhang, Jingtong Hu, and Huazhong Yang. 2016. Performance-aware task scheduling for energy harvesting nonvolatile processors considering power switching overhead. In *Design Automation Conference (DAC), 2016 53nd ACM/EDAC/IEEE*. IEEE, 1–6.

[37] Jinyang Li, Yongpan Liu, Hehe Li, Rui Hua, Chun Jason Xue, Hyung Gyu Lee, and Huazhong Yang. 2016. Accurate personal ultraviolet dose estimation with multiple wearable sensors. In *Wearable and Implantable Body Sensor Networks (BSN), 2016 IEEE 13th International Conference on*. IEEE, 347–352.

[38] Qingan Li, Mengying Zhao, Jingtong Hu, Yongpan Liu, Yanxiang He, and Chun Jason Xue. 2015. Compiler directed automatic stack trimming for efficient non-volatile processors. In *Proceedings of the 52nd Annual Design Automation Conference*. ACM, 183.

[39] Xueqing Li, Sumitha George, Kaisheng Ma, Wei-Yu Tsai, Ahmedullah Aziz, John Sampson, Sumeet Kumar Gupta, Meng-Fan Chang, Yongpan Liu, Suman Datta, and Vijaykrishnan Narayanan. 2017. Advancing Nonvolatile Computing With Nonvolatile NCFET Latches and Flip-Flops. *IEEE Transactions on Circuits and Systems I: Regular Papers* (2017).

[40] Zhicheng Lin, Pui-In Mak, and Rui Martins. 2014. 9.4 A 0.5 V 1.15 mW 0.2 mm 2 Sub-GHz ZigBee receiver supporting 433/860/915/960MHz ISM bands with zero external components. In *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2014 IEEE International*. IEEE, 164–165.

[41] Vincent Liu, Aaron Parks, Vamsi Talla, Shyamnath Gollakota, David Wetherall, and Joshua R Smith. 2013. Ambient backscatter: wireless communication out of thin air. *ACM SIGCOMM Computer Communication Review* 43, 4 (2013), 39–50.

[42] Yongpan Liu, Zewei Li, Hehe Li, Yiqun Wang, Xueqing Li, Kaisheng Ma, Shuangchen Li, Meng-Fan Chang, Sampson John, Yuan Xie, Jiwu Shu, and Huazhong Yang. 2015. Ambient energy harvesting nonvolatile processors: from circuit to system. *Proceedings of the 52nd Annual Design Automation Conference* (2015), 150.

[43] Y. Liu, Z. Wang, A. Lee, F. Su, C. P. Lo, Z. Yuan, C. C. Lin, Q. Wei, Y. Wang, Y. C. King, C. J. Lin, P. Khalili, K. L. Wang, M. F. Chang, and H. Yang. 2016. 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. *2016 IEEE International Solid-State Circuits Conference (ISSCC)* (Jan 2016), 84–86.

[44] Brandon Lucia, Vignesh Balaji, Alexei Colin, Kiwan Maeng, and Emily Ruppel. 2017. Intermittent Computing: Challenges and Opportunities. In *LIPIcs-Leibniz International Proceedings in Informatics*, Vol. 71. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[45] Brandon Lucia and Benjamin Ransford. 2015. A Simpler, Safer Programming and Execution Model for Intermittent Systems. In *Proceedings of the 36th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '15)*. 575–585.

[46] Brandon Lucia and Benjamin Ransford. 2015. A simpler, safer programming and execution model for intermittent systems. *ACM SIGPLAN Notices* 50, 6 (2015), 575–585.

[47] Kaisheng Ma, Xuqing Li, Jinyang Li, Yongpan Liu, Yuan Xie, Jack Sampson, Mahmut T. Kandemir, and Vijaykrishnan Narayanan. 2017. Incidental Computing on IoT Nonvolatile Processors. In *The 50th Annual IEEE/ACM International Symposium on Microarchitecture*.

[48] Kaisheng Ma, Xueqing Li, Yongpan Liu, Yuan Xie, John (Jack) Morgan Sampson, and Vijaykrishnan Narayanan. [n. d.]. Dynamic Power and Energy Management for Energy Harvesting Nonvolatile Processor Systems. *Transactions on Embedded Computing Systems* ([n. d.]).

[49] Kaisheng Ma, Xueqing Li, Srivatsa Rangachar Srinivasa, Yongpan Liu, John Sampson, Yuan Xie, and Vijaykrishnan Narayanan. 2017. Spendthrift: Machine learning based resource and frequency scaling for ambient energy harvesting nonvolatile processors. In *Design Automation Conference (ASP-DAC), 2017 22nd Asia and South Pacific*. IEEE, 678–683.

[50] Kaisheng Ma, Yang Zheng, Shuangchen Li, Karthik Swaminathan, Xueqing Li, Yongpan Liu, Jack Sampson, Yuan Xie, and Vijaykrishnan Narayanan. 2015. Architecture exploration for ambient energy harvesting nonvolatile processors. *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)* (2015), 526–537.

[51] Yitao Ma, Sadahiko Miura, Hiroaki Honjo, Shoji Ikeda, Takahiro Hanyu, Hideo Ohno, and Tetsuo Endoh. 2016. A 600-μW ultra-low-power associative processor for image pattern recognition employing magnetic tunnel junction-based nonvolatile memories with autonomic intelligent power-gating scheme. *Japanese Journal of Applied Physics* 55, 4S (2016), 04EF15.

[52] D. Masotti and A. Costanzo. 2015. Start-up solutions for ultra-low power RF harvesting scenarios. In *2015 IEEE MTT-S International Conference on Numerical Electromagnetic and Multiphysics Modeling and Optimization (NEMO)*. 1–3.

[53] G. V. Merrett. 2016. Invited: Energy harvesting and transient computing: A paradigm shift for embedded systems?. In *2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. 1–2.

[54] Geoff V Merrett and Bashir Al-Hashimi. 2017. Energy-Driven Computing: Rethinking the Design of Energy Harvesting Systems. (2017).

[55] Azalia Mirhoseini, Ebrahim M Songhori, and Farinaz Koushanfar. 2013. Idetic: A high-level synthesis approach for enabling long computations on transiently-powered ASICs. In *Pervasive Computing and Communications (PerCom), 2013 IEEE International Conference on*. IEEE, 216–224.

[56] Saman Naderiparizi, Zerina Kapetanovic, and Joshua R Smith. 2016. Battery-free connected machine vision with wispcam. *GetMobile: Mobile Computing and Communications* 20, 1 (2016), 10–13.

[57] Saman Naderiparizi, Aaron N Parks, Zerina Kapetanovic, Benjamin Ransford, and Joshua R Smith. 2015. Wispcam: A battery-free rfid camera. In *RFID (RFID), 2015 IEEE International Conference on*. IEEE, 166–173.

[58] Naoya Onizawa, Akira Mochizuki, Akira Tamakoshi, and Takahiro Hanyu. 2017. Sudden Power-Outage Resilient In-Processor Checkpointing for Energy-Harvesting Nonvolatile Processors. *IEEE Transactions on Emerging Topics in Computing* 5, 2 (2017), 151–163.

[59] Suat Ozdemir. 2009. Secure Load Balancing via Hierarchical Data Aggregation in Heterogeneous Sensor Networks. *Journal of Information Science & Engineering* 25, 6 (2009).

[60] Vipin Pal, Girdhari Singh, and RP Yadav. 2015. Balanced cluster size solution to extend lifetime of wireless sensor networks. *IEEE Internet of Things Journal* 2, 5 (2015), 399–401.

[61] Chen Pan, Mimi Xie, Yongpan Liu, Yanzhi Wang, Chun Jason Xue, Yuangang Wang, Yiran Chen, and Jingtong Hu. 2017. A lightweight progress maximization scheduler for non-volatile processor under unstable energy harvesting. In *Proceedings of the 18th ACM SIGPLAN/SIGBED Conference on Languages, Compilers, and Tools for Embedded Systems*. ACM, 101–110.

[62] Xiang Pan and Radu Teodorescu. 2014. Nvsleep: Using non-volatile memory to enable fast sleep/wakeup of idle cores. In *Computer Design (ICCD), 2014 32nd IEEE International Conference on*. IEEE, 400–407.

[63] Robert Perricone, Ibrahim Ahmed, Zhaoxin Liang, Meghna Mankalale, X. Sharon Hu, Michael Kim, Chris H.and Niemier, Sachin Sapatnekar, and Jian-Ping Wang. 2017. Advanced spintronic memory and logic for nonvolatile processors. In *International conference and exhibition for the design and engineering of systems-on-chip and embedded systems*.

[64] Jean-Michel Portal, Marc Bocquet, Mathieu Moreau, Hassen Aziza, Damien Deleruyelle, Yue Zhang, Wang Kang, J-O Klein, Y-G Zhang, C Chappert, and W-S Zhao. 2014. An overview of non-volatile flip-flops based on emerging memory technologies. *Journal of Electronic Science and Technology* 12, 2 (2014), 173–181.

[65] Jan Prummel, Michail Papamichail, John Willms, Rahul Todi, William Aartsen, Wim Kruiskamp, Johan Haanstra, Enno Opbroek, Søren Rievers, Peter Seesink, Jan van Gorsel, Harrie Woering, and Chris Smit. 2015. A 10 mW Bluetooth low-energy transceiver with on-chip matching. *IEEE Journal of Solid-State Circuits* 50, 12 (2015), 3077–3088.

[66] Benjamin Ransford, Jacob Sorber, and Kevin Fu. 2011. Mementos: System Support for Long-running Computation on RFID-scale Devices.

In *Proceedings of the Sixteenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS XVI)*. ACM, New York, NY, USA, 159–170. https://doi.org/10.1145/1950365.1950386

[67] Alberto Rodriguez, Domenico Balsamo, Anup Das, Alex S Weddell, Davide Brunelli, Bashir Al-Hashimi, and Geoff V Merrett. 2015. Approaches to transient computing for energy harvesting systems: A quantitative evaluation. In *ENSsys 2015*.

[68] Anith Selvakumar, Meysam Zargham, and Antonio Liscidini. 2015. 13.6 A 600uW Bluetooth low-energy front-end receiver in 0.13 um CMOS technology. In *Solid-State Circuits Conference-(ISSCC), 2015 IEEE International*. IEEE, 1–3.

[69] Sophiane Senni, Lionel Torres, Gilles Sassatelli, and Abdoulaye Gamatie. 2016. Non-Volatile Processor Based on MRAM for Ultra-Low-Power IoT Devices. *ACM Journal on Emerging Technologies in Computing Systems (JETC)* 13, 2, Article 17 (Dec. 2016), 23 pages.

[70] X. Sheng, C. Wang, Y. Liu, H. G. Lee, N. Chang, and H. Yang. 2014. A high-efficiency dual-channel photovoltaic power system for nonvolatile sensor nodes. In *2014 IEEE Non-Volatile Memory Systems and Applications Symposium (NVMSA)*. 1–2.

[71] Saman Siavoshi, Yousef S Kavian, and Hamid Sharif. 2016. Load-balanced energy efficient clustering protocol for wireless sensor networks. *IET Wireless Sensor Systems* 6, 3 (2016), 67–73.

[72] F. Su, Y. Liu, Y. Wang, and H. Yang. 2017. A Ferroelectric Nonvolatile Processor with 46 us System-Level Wake-up Time and 14 us Sleep Time for Energy Harvesting Applications. *IEEE Transactions on Circuits and Systems I: Regular Papers* 64, 3 (March 2017), 596–607.

[73] Vamsi Talla, Bryce Kellogg, Benjamin Ransford, Saman Naderiparizi, Shyamnath Gollakota, and Joshua R Smith. 2015. Powering the next billion devices with Wi-Fi. *arXiv preprint arXiv:1505.06815* (2015).

[74] TI. [n. d.]. CTPL "Compute Through Power Loss" software utility, https://e2e.ti.com/blogs_/b/msp430blog/archive/2015/05/29/what-is-compute-through-power-loss.

[75] Joel Van Der Woude and Matthew Hicks. 2016. Intermittent computation without hardware support or programmer intervention. In *Proceedings of OSDIâĂŹ16: 12th USENIX Symposium on Operating Systems Design and Implementation*. 17.

[76] Dipak Wajgi and Nileshsingh V Thakur. 2012. Load balancing based approach to improve lifetime of wireless sensor network. *International Journal of Wireless & Mobile Networks* 4, 4 (2012), 155.

[77] Cong Wang, Naehyuck Chang, Younghyun Kim, Sangyoung Park, Yongpan Liu, Hyung Gyu Lee, Rong Luo, and Huazhong Yang. 2014. Storage-less and converter-less maximum power point tracking of photovoltaic cells for a nonvolatile microprocessor. In *Design Automation Conference (ASP-DAC), 2014 19th Asia and South Pacific*. IEEE, 379–384.

[78] KL Wang, JG Alzate, and P Khalili Amiri. 2013. Low-power nonvolatile spintronic memory: STT-RAM and beyond. *Journal of Physics D: Applied Physics* 46, 7 (2013), 074003.

[79] Yiqun Wang, Yongpan Liu, Shuangchen Li, Daming Zhang, Bo Zhao, Mei-Fang Chiang, Yanxin Yan, Baiko Sai, and Huazhong Yang. 2012. A 3us wake-up time nonvolatile processor based on ferroelectric flip-flops. In *ESSCIRC (ESSCIRC), 2012 Proceedings of the*. IEEE, 149–152.

[80] Zhibo Wang, Fang Su, Yiqun Wang, Zewei Li, Xueqing Li, Ryuji Yoshimura, Takashi Naiki, Takashi Tsuwa, Takahiko Saito, Zhongjun Wang, Koji Taniuchi, Meng-Fan Chang, Huazhong Yang, and Yongpan Liu. 2017. A 130nm FeRAM-Based Parallel Recovery Nonvolatile SOC for Normally-OFF Operations with 3.9ÃŮ Faster Running Speed and 11ÃŮ Higher Energy Efficiency Using Fast Power-On Detection and Nonvolatile Radio Controlle. In *Proc. Symp. VLSI Circuits (VLSI Circuits)*. C336–C337.

[81] Mimi Xie, Mengying Zhao, Chen Pan, Jingtong Hu, Yongpan Liu, and Chun Jason Xue. 2015. Fixing the broken time machine: Consistency-aware checkpointing for energy harvesting powered non-volatile processor. In *Proceedings of the 52nd Annual Design Automation Conference*. ACM, 184.

[82] Yuan Xie. 2013. *Emerging Memory Technologies: Design, Architecture, and Applications*. Springer Science & Business Media.

[83] Ruqiang Yan and Robert X Gao. 2006. Hilbert–Huang transform-based vibration signal analysis for machine health monitoring. *IEEE Transactions on Instrumentation and measurement* 55, 6 (2006), 2320–2329.

[84] Ruigen Yao and Shamim N Pakzad. 2012. Autoregressive statistical pattern recognition algorithms for damage detection in civil structures. *Mechanical Systems and Signal Processing* 31 (2012), 355–368.

[85] Wing-kei Yu, Shantanu Rajwade, Sung-En Wang, Bob Lian, G Edward Suh, and Edwin Kan. 2011. A non-volatile microcontroller with integrated floating-gate transistors. In *Dependable Systems and Networks Workshops (DSN-W), 2011 IEEE/IFIP 41st International Conference on*. IEEE, 75–80.

[86] Daming Zhang, Shuangchen Li, Ang Li, Yongpan Liu, X Sharon Hu, and Huazhong Yang. 2014. Intra-task scheduling for storage-less and converter-less solar-powered nonvolatile sensor nodes. In *Computer Design (ICCD), 2014 32nd IEEE International Conference on*. IEEE, 348–354.

[87] Daming Zhang, Yongpan Liu, Jinyang Li, Chun Jason Xue, Xueqing Li, Yu Wang, and Huazhong Yang. 2016. Solar power prediction assisted intra-task scheduling for nonvolatile sensor nodes. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 35, 5 (2016), 724–737.

[88] Daming Zhang, Yongpan Liu, Xiao Sheng, Jinyang Li, Tongda Wu, Chun Jason Xue, and Huazhong Yang. 2015. Deadline-aware task scheduling for solar-powered nonvolatile sensor nodes with global energy migration. In *Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE*. IEEE, 1–6.

[89] Han Zhang, Liang Li, Xin-fang Yan, and Xiang Li. 2011. A load-balancing clustering algorithm of WSN for data gathering. In *Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC), 2011 2nd International Conference on*. IEEE, 915–918.

[90] Mengying Zhao, Qingan Li, Mimi Xie, Yongpan Liu, Jingtong Hu, and Chun Jason Xue. 2015. Software assisted non-volatile register reduction for energy harvesting based cyber-physical system. In *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*. EDA Consortium, 567–572.

[91] M. Zwerg, A. Baumann, R. Kuhn, M. Arnold, R. Nerlich, M. Herzog, R. Ledwa, C. Sichert, V. Rzehak, P. Thanigai, and B. O. Eversmann. 2011. An 82 uA/MHz microcontroller with embedded FeRAM for energy-harvesting applications. In *2011 IEEE International Solid-State Circuits Conference*. 334–336.