Graceful Performance Modulation for Power-Neutral Transient Computing Systems

Domenico Balsamo, Anup Das, *Member, IEEE,* Alex S. Weddell, *Member, IEEE,* Davide Brunelli, *Member, IEEE,* Bashir M. Al-Hashimi, *Fellow, IEEE,* Geoff V. Merrett, *Member, IEEE,* and Luca Benini, *Fellow, IEEE*

***Abstract*—Transient computing systems do not have energy storage, and operate directly from energy harvesting. These systems are often faced with the inherent challenge of low-current or transient power supply. In this paper, we propose "power-neutral" operation, a new paradigm for such systems, whereby the instantaneous power consumption of the system must match the instantaneous harvested power. Power neutrality is achieved using a control algorithm for dynamic frequency scaling (DFS), modulating system performance gracefully in response to the incoming power. Detailed system model is used to determine design parameters for selecting the system voltage thresholds where the operating frequency will be raised or lowered, or the system will be hibernated. The proposed control algorithm for power-neutral operation is experimentally validated using a microcontroller incorporating voltage threshold-based interrupts for frequency scaling. The microcontroller is powered directly from real energy harvesters; results demonstrate that a power-neutral system sustains operation for 4–88% longer with up to 21% speedup in application execution.**

***Index Terms*—Dynamic Frequency Scaling, Transient Computing, Energy Harvesting, Graceful Performance Modulation.**

I. Introduction

The Internet-of-Things (IoT) is a networking paradigm, providing wide-ranging opportunities by allowing the internet to encompass a large number of smart objects through standard communication protocols, providing information and services to end users [1]. Fundamental to this vision are networked autonomous devices, collections of tens to thousands of nodes organized into cooperative networks [2]. Each device is equipped with sensors to detect physical phenomena such as light, heat, pressure etc.; microcontrollers, CPUs, DSPs etc. to process data; memory for data storage; and finally, radio-frequency (RF) transceivers for communication. Rapid technological advances are driving the widespread deployment of IoT devices for numerous application areas including environmental, medical, military, transportation, crisis management, homeland defense, and smart spaces [3].

Scaling of CMOS device geometry has far outpaced the scaling of energy densities in batteries, meaning that the power supply is often the largest and most expensive part of IoT sensor nodes. This determines the operating life, defined as the **lifetime** of an IoT sensor node [4]. The high cost and disruption associated with replacing the batteries of IoT nodes is limiting their large scale deployment. For several years, researchers have tried to reduce the power consumption of sensor nodes using various design techniques [5]. An alternative approach which addresses the finite lifetime of battery-operated sensor nodes is energy harvesting [6], which is defined as the process of harnessing electrical energy from alternative sources such as light, wind, heat, vibration, and movement, and using it to power sensor nodes. However, energy harvesting sources can be volatile, meaning that a steady power supply cannot be relied upon [7]. A conventional sensor node will turn off when its supply voltage falls below the minimum operating voltage. To mitigate this, several approaches may be used (Section II).

The conventional solution is to equip energy-harvesting autonomous devices with an energy buffer in the form of a supercapacitor or rechargeable battery [8]. The design objective is to match in the long-term the energy consumption with the available energy, achieving longer lifetimes. These approaches are known as **energy-neutral** solutions, as they attempt to balance the long-term energy consumed with the stored energy [9]–[11]. Techniques for adjusting the activity levels of battery-powered systems include sleeping or powering-down parts of the system, or modifying the supply voltage and/or frequency of the processor [12]. Dynamic voltage scaling (DVS) is where the supply voltage of the system is adjusted to the minimum to maintain the required performance level; dynamic frequency scaling (DFS) is where the clock frequency of the system is modulated for the required performance; or a combination of dynamic voltage and frequency scaling (DVFS) involves both parameters (voltage and frequency) being adjusted to maximize efficiency. Conventionally, these parameters are modulated to minimize the power drawn by the system while still meeting the required deadlines or other performance demands.

Even though storing energy is convenient for compensating temporary deficiencies in harvested power with the energy stored in periods of abundance, energy storage in the form of batteries pose pollution and sustainability issues, while supercapacitor-based solutions increase sensor node dimensions. As an illustration, two AA-sized batteries on a Crossbow Telos mote occupy over half of its overall volume [13]. In

Manuscript received Mmmmmm dd, yyyy. This work was supported by EPSRC Grants EP/L000563/1 and EP/K034448/1 (the PRiME Programme www.prime-project.org). It was also supported by a Telecom Italia s.p.a. PhD grant and PHIDIAS (EU 7th Framework Programme CA 318013).

D. Balsamo, A. Das, A. S. Weddell, B. M. Al-Hashimi and G. V. Merrett are with the Pervasive Systems Centre, Electronics and Computer Science, University of Southampton, Southampton, UK SO17 1BJ. E-mail: {dominico.balsamo,a.k.das,asw,bmah,gvm}@ecs.soton.ac.uk. D. Brunelli is with the Department of Industrial Engineering, University of Trento, Italy. L. Benini is with Department of Electrical, Electronic and Information Engineering Guglielmo Marconi (DEI), University of Bologna, Italy.

Experimental data used in this paper can be found at DOI:10.5258/SOTON/386876 (http://dx.doi.org/10.5258/SOTON/386876)

systems with constrained dimensions, such as implantable bio-sensors for medical applications and wearable consumer devices [14], assisted living and health-care [15], home and building automation [16], and RFID applications [17], the inherent lack of storage may limit their effectiveness. It is desirable to reduce the size of the energy harvester and the storage as far as possible, to limit the cost and dimensions of sensor systems.

An alternative approach is to operate a sensor node directly from an energy harvesting source without dedicated energy storage. However, the drawback of this approach is that, the power supply is typically varying (transient), computation will frequently be interrupted and reset. This recently proposed computing paradigm is termed **transiently-powered computing** [18], and systems supporting transient computations are termed as **transient computing systems**. To deal with the unstable output from energy harvesters, the system state is saved and hibernated as the supply voltage falls below the minimum operating voltage, and resumes operation from that point when the power supply recovers [19]. Other approaches include checkpointing at regular intervals [20] [21], and optimal energy allocation for data communication using information about existing and future power availability [22]. However, for some application scenarios, continuous operation, even with performance degradation, is desirable over a fail-stop approach where operation suspends on power failure. It could also avoid some of the overheads of the other approaches.

In this paper we propose **power-neutral** operation, a new paradigm for low-current and transiently-powered IoT devices: instead of using additional energy storage or suspending operation on power failure, power-neutral IoT devices aim to operate uninterruptedly through dynamic frequency scaling (DFS), matching their instantaneous power consumption to that of the their energy harvesting supply. Figure 1 shows the envisaged power-neutral behavior of a system (shown in red) in response to a rectified sinusoidal signal (shown in blue). An existing system [19] responds to this input signal by hibernating and restoring every time the input voltage passes a threshold (2.2 V). This response is shown in the figure in green. As seen in the figure, power neutral behavior offers advantages over the existing system by continuing operation in a dynamic scenario without multiple hibernation and restore events. Power neutrality is achieved using a control algorithm which scales the microcontroller frequency adaptively in response to changes in the supply voltage. Specifically, the algorithm gradually increases the operating frequency when the input voltage increases, thereby improving system performance. On the other hand, when the input voltage decreases, the algorithm gradually reduces the operating frequency, modulating the performance. Only when the input power is too low to sustain computation even at the lowest operating condition, a snapshot of the system is stored in the Ferroelectric RAM (FRAM), and the microcontroller is hibernated. Thus, through power-neutral operation, we achieve the following:

- for a scenario where the supply is insufficient to start operation at the highest performance level, power-neutrality allows a microcontroller to start with reduced performance and gracefully modulate it in response to changes in the input power source.
- for a dynamic scenario where the input power is transient, power-neutrality allows continuous operation without frequent hibernation and restore operations (as in existing systems), thus saving energy and improving performance.

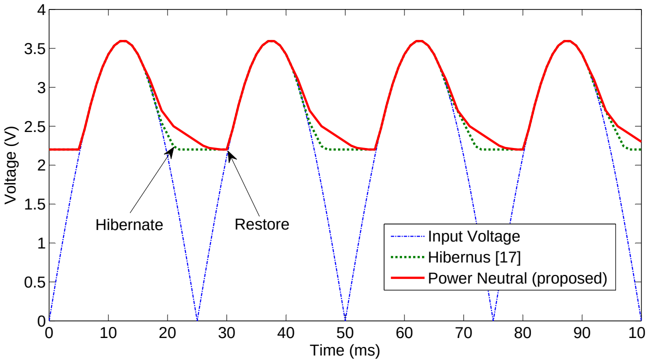


Fig. 1: Dynamic behavior of microcontroller in response to a fully rectified sinusoidal signal.

Contributions: the key contributions of this work are:

1) A methodology for power-neutral operation through dynamic frequency scaling, responding to changes in the power supply (Section III);
2) Mathematical formulation of power-neutral system behavior for determining design parameters of the control algorithm (Section IV); and
3) Validation of the methodology with real energy harvesters on a low-power microcontroller (Section V), demonstrating lifetime and performance improvement.

Experiments conducted on a Texas Instrument's microcontroller with a real photovoltaic cell and wind energy harvester demonstrate that the power-neutral system sustains operation for an additional 4–88% of the time (as compared to existing systems) with up to 21% speedup in application execution.

The remainder of this paper is organized as follows. Related works on transient computing techniques is provided in Section II. The control strategy leading to power-neutral operation is described in Section III. This is followed by mathematical modeling of the system in order to determine design parameters for experiments in Section IV. Results using synthetic and real energy harvesters are presented in Section V. The paper is concluded in Section VI.

II. Related Works

Wireless distributed sensor networks have gained importance in recent years due to the widespread deployment of Internet-of-Things (IoT) devices for numerous application areas including environmental, medical, military, transportation, crisis management, homeland defense, and smart spaces [3] [23]. To maximize a sensor node's active lifetime post deployment, power management techniques are typically used to match the functionality of a system and its peripherals to the demands of its workload [12] while consuming the least power. In [24], the authors propose a new supply voltage scaling technique termed passive voltage scaling (PVS) that eliminates the need for DC-DC converters. The idea behind this approach is to extend the lifetime of a wireless sensor node by scaling the frequency of operation as a function of battery voltage. There are two major limitations of this work. First, this technique implements an open loop control system, which results in stability concerns when applied to transient supplies associated with energy harvesters. Second, this technique is not able to adapt the voltage level in response to the workload, independent of the battery state.

A DC-DC converter aware optimal power management technique is proposed in [25] which aims to achieve optimal system performance under energy harvesting constraints through dynamic voltage scaling. A system-wide power management approach is proposed in [26] that maximizes the energy reserve for all nodes by modulating frequency of operation, ensuring highly resilient performance under emergency and fault-driven situations. These approaches generally assume that power is provided by a battery, and that energy should be conserved where possible while meeting the objectives of the application. However, this is complicated by systems which can generate electrical power from their environment through energy harvesting [6].

Traditionally, sensor nodes powered directly by energy harvesting lose their state upon power loss, restarting computation from the beginning on power recovery. To address this, a number of approaches have been recently proposed. Checkpointing can be used [20] to periodically store system state in non-volatile memory (e.g. flash); when power recovers, the system will restore the most recent checkpoint and continue operation. Disadvantages of this approach include the fact that many checkpoints will be taken (most of which will be redundant), and that space must be reserved in non-volatile memory for two complete checkpoints in case a power interruption occurs whilst one is being taken. To overcome this, a new technique, known as Hibernus [19], was recently reported which delivers stable operation for intermittently powered systems, saving state to non-volatile memory as the supply fails. The technique monitors the external supply using a voltage comparator; when the external voltage drops below a pre-defined threshold, a snapshot of the system's volatile memory (RAM and processor registers) is stored in the non-volatile memory. Upon recovering power, the microcontroller restores these data, allowing the system to resume operation. The major benefit of Hibernus is that only one checkpoint is saved per power interruption, so the overheads of the scheme are much lower than for other reported works. A similar approach is recently proposed in [27] that shuts down or hibernates the microcontroller in response to the external power availability.

This work differs from the existing work [19] by exploiting the dynamic frequency scaling capabilities of the microcontroller to gracefully scale its operating frequency up or down in response to its power status. An additional safeguard is that it can hibernate when the supply voltage falls below a predefined threshold, enabling our proposed approach to be used alongside the existing checkpointing techniques [19], [20], [27]. The main benefit of the proposed approach is that, by avoiding the overheads of hibernation (as in the case of [19]) by gracefully scaling the operating frequency, we can achieve power-neutral operation while maximizing the system's active time and throughput.

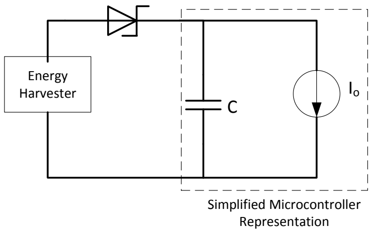


Fig. 2: Example system architecture.

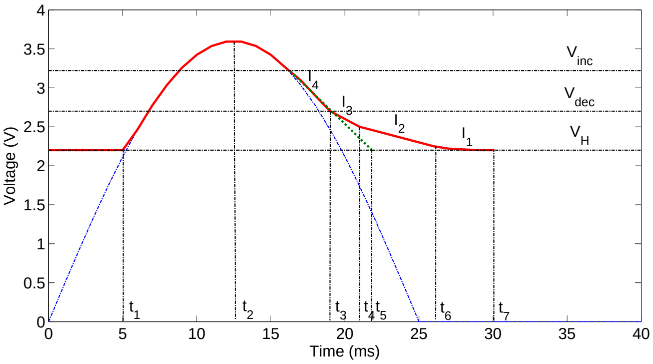


Fig. 3: Behavior with a transient input (in blue). The response to this input for existing system [19] and power-neutral system is shown in green and red, respectively.

III. Power-Neutral Operation

A. Control Strategy

For an explanation of the power-neutral concept, we consider an example system architecture, Figure 2, where the output from an energy harvester (EH) is half-wave rectified and used to power an autonomous device. In this scenario, the input EH is first applied and then discontinued, demonstrating how the microcontroller responds to this transient source. This behavior is shown in Figure 3, which plots the input voltage (in blue), the typical microcontroller response (in green) and the ideal response under our proposed power-neutral control scheme (in red), where the system gracefully modulates the system's performance dependent on its power status. Additionally, the system is assumed to have the capability to hibernate when the voltage falls to a critical level (as with *Hibernus* [19]), i.e. as soon as the decoupling capacitor's voltage crosses the hibernation threshold (V_H), the microcontroller saves its state and suspends operation. Two additional thresholds – frequency up-scaling threshold V_{inc} and frequency down-scaling threshold V_{dec} are defined.

B. Power-Neutral Dynamic Frequency Scaling

The flowchart in Figure 4 describes the behavior of the proposed approach, from when the microcontroller is powered on. The microcontroller remains in low power mode until the supply voltage reaches the restore threshold V_R. At this time,

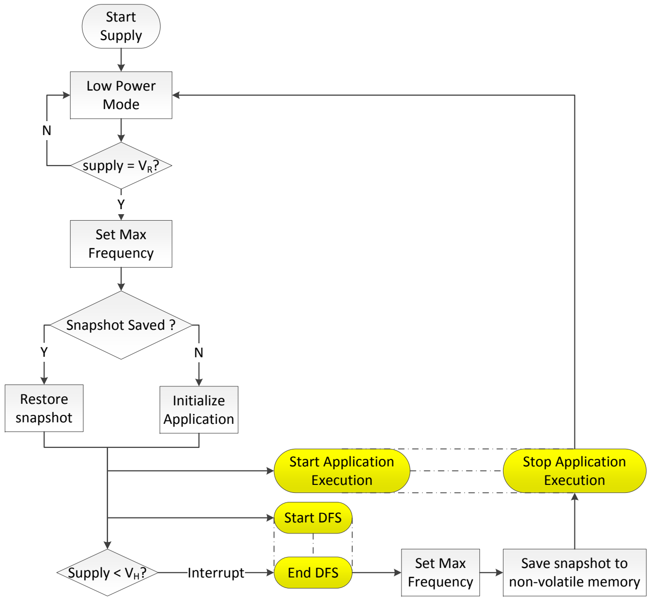


Fig. 4: Power-neutral system operation shown as flowchart.

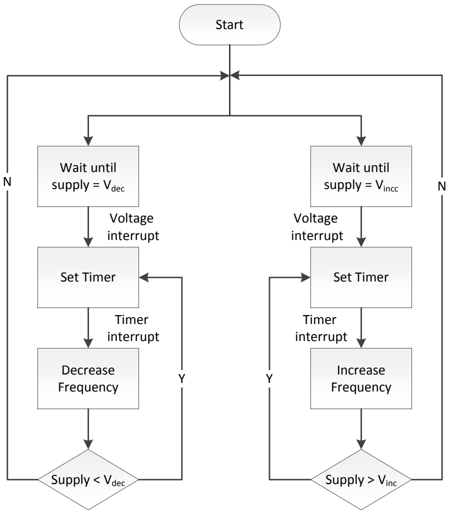


Fig. 5: Control strategy for dynamic frequency scaling.

the microcontroller is set at the highest frequency[1] (f_{N_f}). A check is performed to determine whether a snapshot of the system state has previously been saved (i.e. whether the system has been active and successfully hibernated). If there is a saved snapshot, it is restored; otherwise the system is initialized. Following this, the DFS control algorithm is started simultaneously with the application. As long as the supply voltage is above the hibernate threshold V_H, the application execution is triggered along with the DFS control algorithm to scale the operating frequency dynamically. When the supply voltage drops to V_H, an interrupt is generated which stops the DFS algorithm. The frequency is set to its maximum value in order to save the snapshot. The application execution is terminated and microcontroller settles to a low power mode and waits for the supply voltage to raise again to V_R. It is to be noted that if the supply system stays in low-power mode for a fixed amount of time, the system shuts down.

The DFS control algorithm is shown as a flowchart in Figure 5. The algorithm waits for the supply voltage to be equal to V_{inc} or V_{dec}. If the supply voltage is equal to V_{dec}, a voltage interrupt is generated which sets a timer to count N_{wait} number of cycles. When this timer expires, another interrupt is generated and the frequency is decremented. A check is performed to determine if the supply voltage is lower than V_{dec}. If true, the timer is set again and process is repeated. If, however, the frequency is above V_{dec} (the check evaluates to false), the control is initialized to the beginning of the flow, to be interrupted when the supply voltage is equal to V_{dec} or V_{inc}. A similar but opposite control strategy is implemented when the supply voltage equals V_{inc}.

IV. System Modeling

A. Control Analysis

As shown in Figure 3, an example microcontroller (response shown in green) suspends at time t_5. The time duration for which the microcontroller is active is $(t_5 - t_1)$. Mathematically, this duration is determined as follows. During the duration $(t_2 - t_1)$, the on-board capacitance charges to the peak voltage V_{max} using the input source and this time is dependent on the input source (shown in blue). During the time duration $(t_5 - t_2)$, the capacitor discharges due to the application execution, which can be represented using a constant current sink I_4. It is to be noted that this current I_1 is dependent on the operating frequency of the microcontroller. For *Hibernus*, this is the highest frequency f_{N_f}. Using this, the current-voltage relation of a capacitor can be solved to determine the time duration as

$$V_H = V_{max} - \frac{I_4 \cdot (t_5 - t_2)}{C} \text{ or } t_5 - t_2 = \frac{C\,(V_{max} - V_H)}{I_4} \qquad (1)$$

In the considered scenario (Figure 3), the microcontroller is powered by an energy harvester and the current it draws is dependent on the application and the operating frequency. The capacitance, C, is the on-board capacitance, which is mainly comprised of decoupling capacitors. In the following subsections, we analyze the power-neutral behavior of the microcontroller to determine the design parameters for the proposed control algorithm.

B. Threshold-based Control

In the proposed approach, a power-neutral system needs to change its frequency adaptively in response to the input source. To achieve this, we define two voltage thresholds – V_{inc} and V_{dec}. In its simplest form (discussed in details in the previous section), the frequency is increased if the input voltage (the voltage at the output of the Schottky diode) increases above V_{inc}; and the microcontroller decreases the frequency of operation if the input voltage is lower than V_{dec}. As discussed before, the system's power-neutral behavior is

[1] In this work, we used the highest frequency to hibernate and restore, as this is the most energy efficient frequency (Section V)

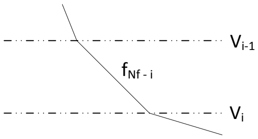


Fig. 6: An example of non-uniform voltage interval.

demonstrated only for the decreasing voltage scenario and is plotted in red in the figure. As can be seen, the discharge characteristics of the decoupling capacitor change over time and the overall duration for which the microcontroller operates is $(t_7 - t_1)$. The improvement in the operating lifetime of the microcontroller as compared to the typical behavior is $(t_7 - t_5)$. The time duration $(t_7 - t_1)$ can be determined as follows. As before, the time $(t_2 - t_1)$ is the time of charging the capacitance to the peak voltage V_{max}. Following this, the microcontroller operates at the highest frequency until the input voltage reaches V_{dec}. This time is

$$t_3 - t_2 = \frac{C\,(V_{max} - V_{dec})}{I_4} \tag{2}$$

Next we determine the time interval $(t_7 - t_3)$ for a general case with N_f frequency points[2]. There are two choices for selecting the interval for successive frequency scaling:

- **Fixed voltage interval:** i.e., the voltage interval $(V_{dec} - V_H)$ is divided into $(N_f - 1)$ equal intervals. At each of these intervals, the frequency of the microcontroller is scaled down progressively.
- **Non-uniform voltage interval:** i.e., the voltage interval $(V_{dec} - V_H)$ is divided into $(N_f - 1)$. However, the lengths of these intervals are not equal. As before, the frequency of the microcontroller is scaled down progressively at these intervals.

In this work, we use variable voltage intervals determined using a fixed number of clock cycles. In other words, the microcontroller scales its operating frequency and waits for a fixed number of clock cycles, before checking the voltage and scaling the frequency again. Figure 6 shows one of the non-uniform voltage intervals. The frequency (scaled down from the highest frequency f_{N_f}) for this interval is shown in the figure. Assuming that the microcontroller waits for N_{wait} clock cycles before scaling the frequency further, the time interval (t_{wait}) for these N_{wait} cycles is

$$t_{wait}(i) = \frac{N_{wait}}{f_{N_f - i}} \tag{3}$$

We are interested in finding the bounds on this time t_{wait} in order to determine the optimum value of N_{wait}. The minimum wait time is given by the highest operating frequency N_f, i.e., $t_{wait}^{min} = \frac{N_{wait}}{f_{N_f}}$. The lower bound on t_{wait}^{min} is determined by the frequency switching overhead t_{so} i.e., $t_{wait}^{min} > t_{so}$. The upper bound is determined by the time interval $(t_5 - t_2)$ which is the time taken by a microcontroller to hibernate, after supply is cut-off i.e.,

$$t_5 - t_2 = \frac{C(V_{max} - V_H)}{I_{N_f}} \tag{4}$$

In other words, if the time t_{wait}^{min} is greater than the time $(t_5 - t_2)$, there remains no scope for frequency scaling and the power-neutral system behaves like a traditional one, offering no lifetime improvement. To summarize, the minimum wait time is bounded by

$$t_{so} < t_{wait}^{min} < t_5 - t_2 \tag{5}$$

Using Equation 4, the above equation can be re-written as

$$t_{so} < \frac{N_{wait}}{f_{N_f}} < \frac{C(V_{max} - V_H)}{I_{N_f}} \tag{6}$$

The value of the frequency switching overhead t_{so} is justified in Section V-F.

We introduce a design parameter $\mathcal{L}$ such that

$$\frac{N_{wait}}{f_{N_f}} = \mathcal{L} \cdot t_{so} \text{ or equivalently } N_{wait} = \mathcal{L} \cdot t_{so} \cdot f_{N_f} \tag{7}$$

where $\mathcal{L} \geq 1$.

Having defined the wait time, the time interval $(t_7 - t_2)$ can be written as

$$\begin{aligned} t_7 - t_2 &= t_3 - t_2 + \sum_{i=1}^{N_f - 1} \left[t_{wait}(i) + t_{so} \right] \qquad (8) \\ &= \frac{C\,(V_{max} - V_{dec})}{I_{N_f}} + \sum_{i=1}^{N_f - 1} \left[\frac{N_{wait}}{f_{N_f - i}} + t_{so} \right] \end{aligned}$$

where I_4 is written as I_{N_f} for the general case and t_{so} is the frequency switching overhead. Typically, the frequency switching overhead depends on the frequency levels between which the switching takes place. However, as a first-order approximation to the problem formulation, t_{so} is the maximum overhead of switching between any pair of frequencies.

Using Equations 4 and 8, the improvement in microcontroller lifetime is

$$\begin{aligned} t_7 - t_5 &= (t_7 - t_2) - (t_5 - t_2) \qquad (9) \\ &= \frac{C\,(V_{max} - V_{dec})}{I_{N_f}} + \sum_{i=1}^{N_f - 1} \left[\frac{N_{wait}}{f_{N_f - i}} + t_{so} \right] \\ &\quad - \frac{C(V_{max} - V_H)}{I_{N_f}} \\ &= \sum_{i=1}^{N_f - 1} \left[\frac{N_{wait}}{f_{N_f - i}} + t_{so} \right] - \frac{C\,(V_{dec} - V_H)}{I_{N_f}} \end{aligned}$$

To verify the validity of the simplified microcontroller representation (Figure 2), we need to look at the dynamics of microcontroller's power demands. To this end, we conducted a set of experiments on the experimental platform executing the three example applications at frequencies ranging from 1 MHz to 8 MHz. Figure 7 plots the results for the FFT, CRC and RSA applications. As can be seen, at a given frequency the current drawn by these applications are similar (within 0.1%). Additionally, the current drawn by the microcontroller appears to be linearly dependent on its frequency of operation; this is consistent with the datasheet. Although the current values reported in this figure are averaged across the execution of

[2] In Figure 3 only four frequency points are shown for demonstrations, i.e., $N_f = 4$.

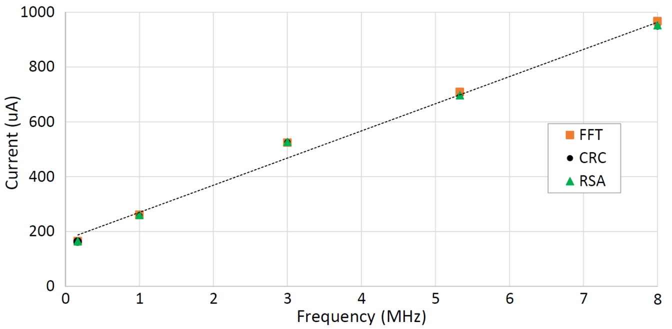


Fig. 7: Current consumed by the experimental platform at different frequencies and executing different applications.

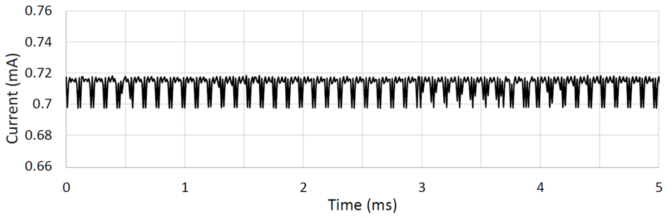


Fig. 8: Current drawn during execution of FFT at 5.33 MHz.

these applications, the variation within execution is insignificant. To illustrate this, Figure 8 plots the current drawn by the microcontroller during FFT execution at 5.33 MHz. The maximum current variation is 20 μA which is less than 3% of the mean current of 0.715 mA during the FFT execution at this frequency. For all our analysis in Section IV, this variation is ignored and the microcontroller is represented using a constant current load for a given application at a particular frequency, as shown in Figure 2.

As seen from Figure 7, the frequency of operation and the current drawn by a microcontroller can be represented using a linear equation of the form

$$I_i = I_o + k \cdot f_i \tag{10}$$

Substituting this in Equation 9, the lifetime improvement is determined by

$$t_7 - t_5 = \sum_{i=1}^{N_f-1} \left[\frac{k \cdot N_{wait}}{I_{N_f-i} - I_o} + t_{so} \right] - \frac{C\,(V_{dec} - V_H)}{I_{N_f}} \tag{11}$$

C. Threshold and Interval Definition

Referring back to Figure 6, the boundary conditions for the voltage levels are $V_0 = V_{dec}$ and $V_{N_f-1} = V_H$. The intermediate voltage values (V_i) can be determined using the capacitor voltage current relation using a constant current sink model (Figure 2) as

$$V_i = V_{i-1} - \frac{I_{N_f-i} \cdot t_{wait}(i)}{C} \tag{12}$$

Using the boundary conditions, the interval $V_{dec} - V_H$ can be represented as

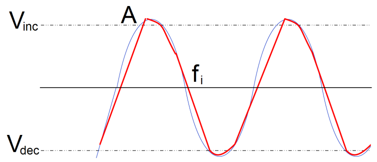


Fig. 9: Voltage response of the microcontroller (red) to an input voltage (blue) oscillating between V_{inc} and V_{dec}.

$$\begin{aligned} V_{dec} - V_H = V_0 - V_{N_f-1} \quad &\geq \sum_{i=1}^{N_f-1} (V_{i-1} - V_i) \\ &\geq \sum_{i=1}^{N_f-1} \frac{I_{N_f-i} \cdot t_{wait}(i)}{C} \end{aligned} \tag{13}$$

The minimum value of V_{dec} can thus be determined. To determine V_{inc}, we consider an EH source with an oscillating nature as shown in Figure 9. As discussed previously, every time the input voltage increases above V_{inc}, the frequency is incremented one step, whereas if the voltage decreases below V_{dec}, the frequency is decremented to the next lower level. Assuming that the frequency of the microcontroller is incremented to f_i at point A in the figure when the input voltage crosses V_{inc}, the time taken by the microcontroller voltage to drop to V_{dec} in response to the EH source is

$$t = \frac{C \cdot (V_{inc} - V_{dec})}{I_i} \tag{14}$$

where I_i is the current drawn by the microcontroller at frequency f_i. The least time taken t_{min} is

$$t_{min} = \frac{C \cdot (V_{inc} - V_{dec})}{I_{N_f}} \tag{15}$$

For stable operation, this minimum time has to be greater than the worst-case frequency switching overhead t_{so}. We choose this to be a few orders of magnitude higher. The lower bound for V_{inc} is t_{so}. To determine the upper bound of V_{inc}, we need to consider the scenario with increasing supply voltage. Using similar reasoning as before, it can be shown that the upper bound of V_{inc} is $V_{max} - \sum_{i=1}^{N_f-1} \left[\frac{N_{wait}}{f_{N_f-i}} + t_{so} \right]$, i.e.,

$$V_{dec} + \frac{I_{N_f} \cdot t_{so}}{C} < V_{inc} \leq V_{max} - \sum_{i=1}^{N_f-1} \left[\frac{N_{wait}}{f_{N_f-i}} + t_{so} \right] \tag{16}$$

D. Performance Analysis

The performance of a microcontroller is measured in terms of the number of instructions executed in the time interval between restoring a snapshot and hibernating. To estimate this, we let `IPC` denote the instructions executed by the microcontroller per clock cycle and f_{source} denote the frequency of a sinusoidal signal powering the the microcontroller. For the existing system [19], the time for which the microcontroller is active is the sum of the time to charge the decoupling capacitor to its peak value and the time to discharge. The charging time is approximately

$$t^{EN}_{charge} = t_2 - t_1 = \frac{1}{4 \cdot f_{source}} - \frac{\sin^{-1}\left(\frac{V_H}{V_{max}}\right)}{2\pi \cdot f_{source}} \tag{17}$$

where the first term in the above equation is the rise time of the input source to its peak value V_{max}. The second term, which represents the time the input signal takes to reach the threshold voltage V_H, is discounted from the first term to highlight the fact the microcontroller starts its operation when the supply voltage reaches the threshold V_H. The microcontroller always operates at the highest frequency f_{N_f} and therefore the discharge time is given by Equation 1.

$$t^{EN}_{discharge} = t_5 - t_2 = \frac{C\,(V_{max} - V_H)}{I_{f_{N_f}}} \tag{18}$$

The number of instructions executed in the duration for which the microcontroller is on, is given by

$$C^{EN} = \left(t^{EN}_{charge} + t^{EN}_{discharge}\right) \cdot f_{N_f} \cdot \texttt{IPC} \tag{19}$$

On the other hand, the power-neutral system scales its frequency adaptively. As before, $t_{wait}(i)$ denotes the time a power-neutral system spends before scaling up/down the frequency. We determine the average throughput obtained by this microcontroller when the supply voltage is suddenly cut off. As discussed in the previous section, when the supply is cut off, the microcontroller uses the highest frequency until the voltage drops to V_{dec}. This time interval $(t_3 - t_2)$ is given by Equation 2. Thereafter, frequency is scaled every $t_{wait}(i)$ interval. It is to be noted that once scaled down to the lowest frequency f_1, the microcontroller continues to operate at this frequency until the frequency drops to V_H. We therefore have to determine the time interval with lowest frequency of operation separately. To determine this interval, we determine the voltage drop until the point when the frequency is scaled to the lowest value f_1. This is given by

$$V_{N_f-2} = V_{dec} - \sum_{i=1}^{N_f-2} \frac{I_{N_f-i} \cdot t_{wait}(i)}{C} \tag{20}$$

The above equation is determined using Equations 12 and 13. The voltage interval for which the lowest frequency is active is $V_{N_f-2} - V_H$. The time taken for this voltage drop across the decoupling capacitor using the constant current sink of I_1 (Figure 2) corresponding to the lowest frequency of f_1, is given by

$$t_{last} = \frac{C(V_{N_f-2} - V_H)}{I_1} \tag{21}$$

The number of instructions executed by the microcontroller in the time for which the decoupling capacitor discharges from V_{max} to V_H is given by

$$\begin{aligned} C^{PN}_{discharge} \quad = \quad & \frac{C\,(V_{max} - V_{dec})}{I_{N_f}} \cdot f_{N_f} \cdot \texttt{IPC} + \sum_{i=2}^{N_f-1} t_{wait}(i) \cdot f_i \cdot \texttt{IPC} + \\ & \frac{C(V_{N_f-2} - V_H)}{I_1} \cdot f_1 \cdot \texttt{IPC} \end{aligned} \tag{22}$$

Due to symmetrical operation of the power neutral microcontroller during charging and discharging, the number of instructions executed during charging is given by

$$\begin{aligned} C^{PN}_{charge} \quad = \quad & \frac{C\,(V_{inc} - V_H)}{I_1} \cdot f_1 \cdot \texttt{IPC} + \sum_{i=2}^{N_f-1} t_{wait}(i) \cdot f_i \cdot \texttt{IPC} + \\ & \frac{C(V_{max} - V_{N_f-2})}{I_{N_f}} \cdot f_{N_f} \cdot \texttt{IPC} \end{aligned} \tag{23}$$

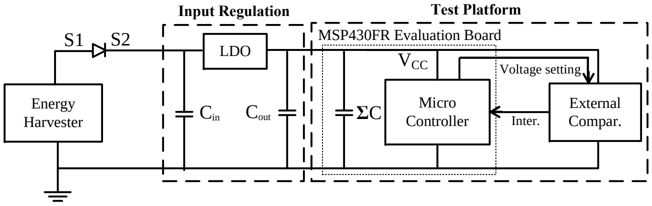


Fig. 10: Experimental setup.

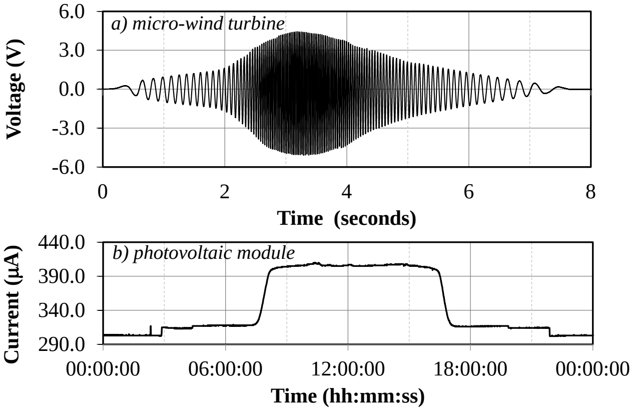


Fig. 11: Dynamics of (a) wind, and (b) PV harvesters.

The total number of instructions executed by a power-neutral system is given by

$$C^{PN} = C^{PN}_{charge} + C^{PN}_{discharge} \tag{24}$$

The comparison of the performance C^{EN} and C^{PN} depends on the input source used. In Section V, we provide a comparison of these for a range of input frequencies.

V. Results and Discussion

In this section we present a thorough validation of our approach starting with a description of the test setup, the energy harvesters, and applications.

A. Experimental Setup

Figure 10 shows the test setup for all our experiments. The microcontroller is a TI MSP430FR5739, which incorporates ferroelectric RAM (FRAM) nonvolatile memory and a 16-bit RISC CPU. The microcontroller is operative from 2.0 V; an external low drop-out regulator (LDO) restricts the supply voltage to a maximum of 3.6 V. An external comparator circuit is used to detect two separate thresholds simultaneously and assert digital inputs when the supply exceeds them. This is used to detect when V_H, V_{inc} or V_{dec} are crossed.

To demonstrate the power-neutral behavior with transiently-powered energy-harvesting sources, all external peripherals are disabled. The frequency of the CPU clock `MCLK` is adjusted using the digitally controlled oscillator (DCO). Two registers `DCOCTL` and `BCSCTL1` are programmed to set any frequency in the range of 0.167 MHz to 24 MHz. To demonstrate the operation of the proposed control algorithm, four frequencies

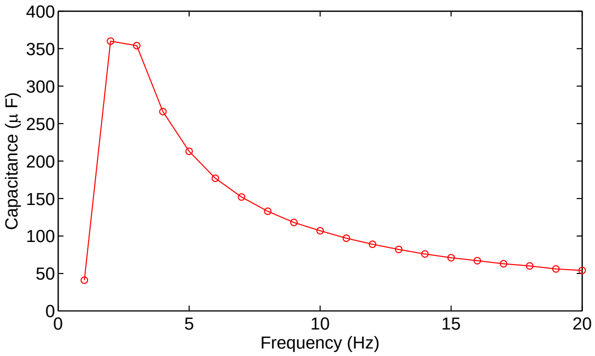


Fig. 12: External capacitance needed to sustain operation.

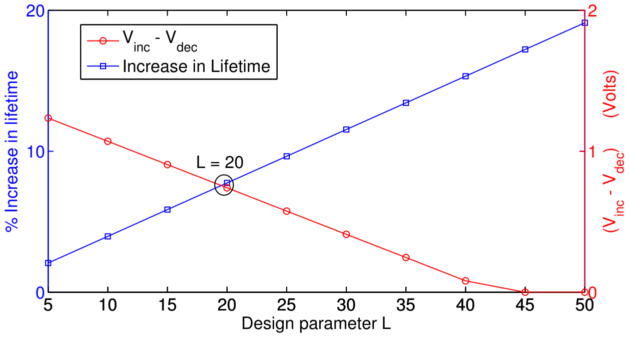


Fig. 13: Selecting the design parameter $\mathcal{L}$.

are selected: 1 MHz, 3 MHz, 5.33 MHz, and 8 MHz. It is to be noted that below 1 MHz, the quiescent component dominates the current consumption and there would be little benefit in extending the operating frequency below 1 MHz. Similarly, the maximum frequency of FRAM operation is 8 MHz. To avoid having wait states while accessing FRAM, the maximum frequency of the CPU is also limited to 8 MHz.

For validation of the proposed control scheme, we have used three sources: a sinusoidal voltage from a bench signal generator, and two real energy-harvesting sources (a photo-voltaic cell (PV) and a micro wind-turbine). Example dynamics of the energy harvesters are shown in Figure 11.

It is important to note that the microcontroller's CPU is ultra-low power[3] but less powerful than their application processor counterparts. For this reason, the range of executable applications is limited. We therefore demonstrate power-neutral behavior using three standard microcontroller applications – FFT, CRC, and RSA, which are the applications used in [19] [20] [21] and are representative workloads for low-power resource-constrained systems. The FFT application analyses three arrays, each holding 128 8-bit samples of tri-axial accelerometer data. The CRC application calculates a 16-bit CRC of a message using polynomials. The RSA application performs 64-bit encryption on 128 characters. In a recent study [18], the technique of [19] is compared with two previously proposed transient computing approaches [20] [21]; results demonstrate that [19] outperforms the other two techniques. In this paper we have therefore, compared our work with [19].

B. Capacitance Requirement of Energy-Neutral Systems

Figure 12 plots the value of the external capacitor required to sustain continuous operation if a buffered ”energy neutral” system is used, as the frequency of the input sinusoid is increased from 1 Hz to 20 Hz. This is the frequency range of a typical wind turbine energy harvester. As seen from this figure, the peak capacitance requirement of the system is 360μF. It is to be noted that if the capacitor is not sufficient to sustain continuous operation at a particular input frequency, the microcontroller resets and restarts computation every time the the voltage value drops below the minimum working voltage of the microcontroller. On the other side, a transient computing system does not require any external capacitance and operates directly from an energy harvester source using the parasitic and on-board decoupling capacitance (approximately 19μF for our experimental platform). For transient computing systems, execution is restored from the point of suspension using hibernate and restore through FRAM [19]; and lifetime is extended through dynamic frequency control (our approach).

C. Selection of Design Parameters

Figure 13 plots the improvement in lifetime of a power-neutral system as compared to existing one [19], as the design parameter $\mathcal{L}$ (Equation 7) is varied from 5 to 50 for the FFT application. Also plotted in the same graph, the difference $V_{inc} - V_{dec}$ as a function of $\mathcal{L}$. As can be seen, the microcontroller lifetime increases with an increase in $\mathcal{L}$, signifying that higher $\mathcal{L}$ is better. On the other hand, with increasing $\mathcal{L}$, the difference between the two voltage levels decreases. From Equation 15, it can be concluded that the lower the design parameter $\mathcal{L}$, the higher the difference between V_{inc} and V_{dec} and therefore, better it is for system stability. Results for the CRC and the RSA application are similar to the FFT application. We therefore select the cross-over point with $\mathcal{L} = 20$ and the remaining design parameters are selected using Equations 2-15, such that the system's lifetime is maximized. Results are summarized in Table I.

TABLE I: Design parameters.

Parameter	**Value**	**Comments**
V_H	2.2 V	constraint of the microcontroller
$\mathcal{L}$	20	based on trade-off in Figure 13
t_{so}	58.6μs	refer to Section V-E
N_{wait}	9376	using Equation 7
V_{dec}	2.53V	using Equation 13
V_{inc}	3.27V	using Equation 15

To determine the impact of the parameter $\mathcal{L}$ on different design aspects, we have conducted an experiment with the

[3] Since battery replacement is often not possible for reasons of access and cost, ultra-low power microcontrollers, such as the TI MSP430FR5739, with energy harvesting are especially suited for low-power and energy-sensitive applications, including energy metering, building automation, security and portable medical or fitness equipment. Such applications need to operate for as long as possible without external power. We have therefore used this microcontroller for validation of our approach.

FFT application and the results are summarized in Table II.

TABLE II: Impact of changing the parameter $\mathcal{L}$.

$\mathcal{L}$	$\mathbf{V_{dec} - V_H}$	$\mathbf{V_{inc} - V_{dec}}$	**time spent**	**% hibernate overhead**	**% input period**
10	0.16V	0.92V	0.01s	40	12
20	0.33V	0.75V	0.03s	20	25
30	0.49V	0.58V	0.04s	13	37
40	0.66V	0.40V	0.05s	10	50
50	0.82V	0.23V	0.06s	8	62

As seen from this table, the difference between $V_{dec} - V_H$ increases with an increase in the parameter $\mathcal{L}$ as expected according to Equation 13. This voltage difference determines the time spent by the system in the lowest frequency as reported in column 4. This time is expressed as a percentage of the hibernation time and input signal period as reported in columns 5 and 6 respectively. As can be seen, the fraction of the time for the system to hibernate reduces with an increase in the time spent in the lowest frequency. However, the percentage of time of the input period increases. This effectively means that as the system spends more time at the lowest frequency, there is much less scope for graceful performance modulation.

D. Hibernate and Restore

An important aspect of the proposed algorithm (Figure 4) is that the microcontroller's operating frequency is set to its maximum (i.e. 8 MHz in this case) when hibernating and restoring. This is because this operating condition is the most energy-efficient of all the four frequencies. To explore this, Table III reports the time overhead, the power and the energy consumption for storing and restoring a snapshot from FRAM. As can be seen, the energy overhead for a snapshot restore is 3.56 μJ at 8 MHz, which is 3x lower than the energy overhead for restoring a snapshot at 1 MHz. Similarly, the energy overhead for hibernation at 8 MHz is 2.9x lower than that at 1 MHz. Clearly, it is most energy efficient to use 8 MHz, and is therefore selected as the frequency for restore and hibernate in our control algorithm.

TABLE III: Energy Overhead for Hibernate and Restore

Frequency	**Restore**			**Hibernate**		
	Time	Power	Energy	Time	Power	Energy
1 MHz	14.8 ms	0.73 mW	10.75 μJ	15.2 ms	0.58 mW	8.8 μJ
3 MHz	5.00 ms	1.43 mW	7.15 μJ	5.2 ms	1.16 mW	6.02 μJ
5.33 MHz	2.85 ms	1.93 mW	5.50 μJ	2.9 ms	1.56 mW	4.53 μJ
8 MHz	1.35 ms	2.63 mW	3.56 μJ	1.40 ms	2.13 mW	2.99 μJ

E. DFS Overhead

To quantify the DFS switching overhead, Figure 14 shows the overhead of switching from 1 MHz to 8 MHz and 8 MHz to 1 MHz (channel `FREQ`). To set a particular frequency, the corresponding clock divider register is first unlocked; the setting is written into the register; and the register is locked again (channels 0 and 1, respectively for switching from 1 MHz to 8 MHz and vice versa). Specifically, channel 0 is set when the clock divider register is unlocked (indicated by marker with letter `a`) and is reset when the clock divider register is locked again (indicated by marker with letter `b`). The clock frequency changes within this interval. As shown in the figure, time from unlocking the register to that when the clock frequency changes, with the change propagated to the output, measuring the overhead of switching from 1 to 8 MHz and is close to the time between marker a and b, which is 58.6 μs. For practical reasons, this is used as the switching overhead. A similar trend can be observed for switching from 8 to 1 MHz, where the time between markers c and d is 20μs.

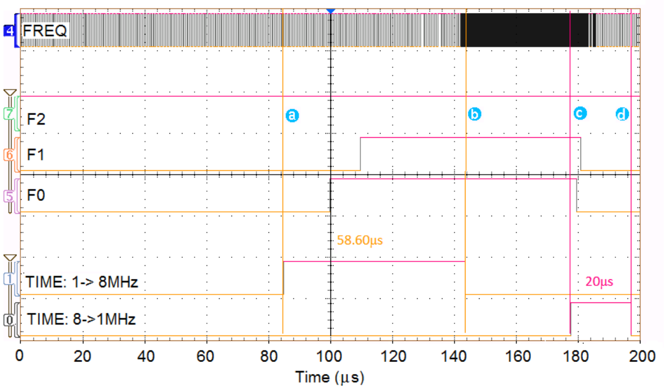


Fig. 14: Overhead of switching from 1 to 8 and 8 to 1 MHz.

Table IV reports the time taken for switching between the four selected frequency levels, both during up-scaling and down-scaling. As can be seen, the time taken for switching depends on the frequency levels between which the switching takes place. For deriving the voltage levels for our design, we have used the worst-case frequency switching overhead of 58.6 μs. This choice addresses two objectives: the worst-case overhead gives an upper bound on the energy overhead of the proposed approach, and the worst-case frequency switching overhead also guarantees graceful system performance operating at the highest frequency.

F. System Response

1) Sinusoidal Input Source: Figure 15 shows the response of a power-neutral system to a sinusoidal source of 0.5 Hz. The input to the microcontroller is shown in blue. This is the voltage applied to the input of the Schottky diode in

TABLE IV: Average frequency switching time.

Frequency Range	**Scaling Up**	**Scaling Down**
1.00 MHz ↔ 3.00 MHz	53.70 μs	39.75 μs
1.00 MHz ↔ 5.33 MHz	62.15 μs	24.15 μs
1.00 MHz ↔ 8.00 MHz	58.60 μs	20.00 μs
3.00 MHz ↔ 5.33 MHz	32.95 μs	13.35 μs
3.00 MHz ↔ 8.00 MHz	28.95 μs	11.35 μs
5.33 MHz ↔ 8.00 MHz	12.95 μs	10.95 μs

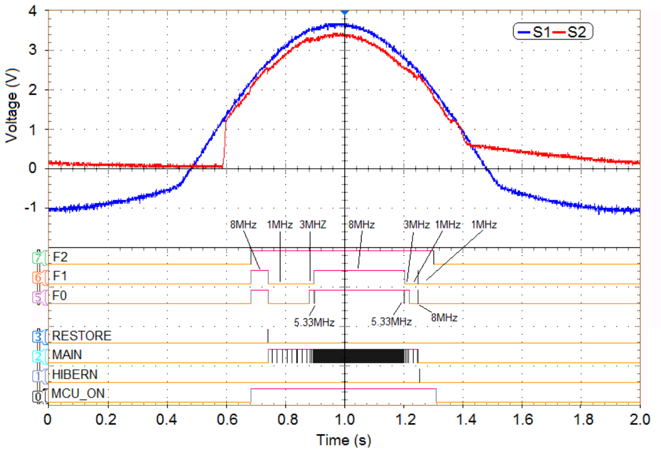


Fig. 15: Microcontroller response to sinusoidal input.

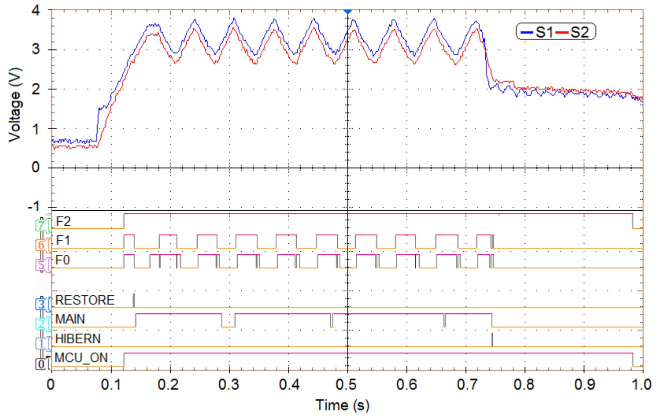


Fig. 16: Microcontroller response to PV module.

Figure 10. The voltage at the input of the decoupling capacitor of the microcontroller (i.e., at output of the Schottky diode) is shown in red in the figure. Channel `F0:F1:F2` represents the frequency of the the microcontroller (e.g., `F0:F1:F2` = "101" implies that the frequency is set to 5 MHz); channel `RESTORE` is set when a snapshot is restored; channel `MAIN` is the main clock (`MCLK`) to the microcontroller; channel `HIBERN` is set when the hibernate routine is invoked; and finally channel `MCU_ON` is the set when the microcontroller is in operation. As seen from the figure, when the supply is applied to the microcontroller, the voltage across the Schottky diode starts to increase. When the voltage crosses the restore threshold (2.2 V), the frequency is ramped up gradually to the highest frequency of 8 MHz. This can be observed from the changes to the frequency selection channels `F0-F2`. The microcontroller continues to operate at the highest frequency, until the voltage drops to V_{dec}, when the microcontroller starts decreasing the frequency until it reaches the hibernate threshold, at which point the microcontroller saves its state to FRAM. The lifetime of the microcontroller is the time interval between the restore and the hibernate.

2) Photovoltaic Cell and Wind Harvester: To verify the system operation using real energy harvesters, experiments are conducted on the platform by using a photovoltaic cell and micro wind turbine. Figure 16 plots the response of the microcontroller to a photovoltaic (PV) energy harvester which generates a current of approximately 450 μA. As seen from the figure, the microcontroller switches between the four frequencies in response to the input source. This demonstrates the power-neutral behavior of the microcontroller, scaling frequency and modulating performance in response to the dynamics of the input source. A similar trend is observed in the wind energy harvester; the response is shown in Figure 17. It is important to note that the system operates continuously, without hibernate and restore, in response to photovoltaic cell and micro wind turbine energy harvesters.

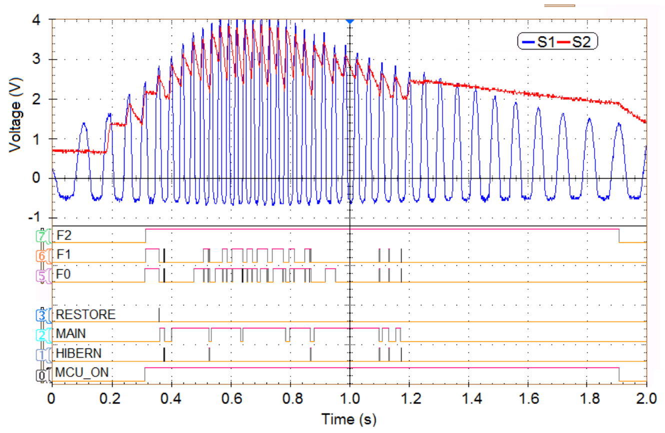


Fig. 17: Microcontroller response to wind energy harvester.

TABLE V: FFT instructions executed in one period of sine input.

Input Frequency (Hz)	Existing System [19]	Power-Neutral System
1	2.77×10^6	2.69×10^6
2	1.61×10^6	1.53×10^6
5	0.91×10^6	1.04×10^6
10	0.68×10^6	1.04×10^6
20	0.57×10^6	1.04×10^6

G. Performance Improvements

Table V reports the number of instructions executed by the existing system [19] and the power-neutral one, for an input sinusoidal signal. Results are reported for five frequencies – 1 Hz to 20 Hz, encompassing the frequency range for most commonly used energy harvesters. As can be seen, for input frequencies of 1 Hz and 2 Hz, the existing system executes an average 4% more instructions than the power-neutral system. However, for all other frequencies, the power-neutral system outperforms the existing system by executing 13-82% (mean 50%) more instructions per clock cycle.

To compare the execution time of the chosen applications, an experiment is conducted on a microcontroller powered with a constant low current source. The number of times the microcontroller hibernated during the execution and the total time taken to execute the application are recorded and are reported in Table VI for the existing system [19] and the

TABLE VI: Execution time with constant current sources.

Application	Current (μA)	Existing System [19]		Power-Neutral System	
		# Hibernate	Time (ms)	# Hibernate	Time (ms)
FFT	200	18	1960	12	1551
	400	13	940	0	608
	600	9	306	0	286
	800	5	202	0	197
CRC	200	5	407	1	338
	400	2	206	0	175.2

proposed power-neutral system. The current values used for this experiment are reported in Column 2 for the application in Column 1. As can be seen, the existing system takes 1960 ms to execute the FFT application with a current source of 200 μA. In doing so, the system hibernates 18 times. For the same input current, the power-neutral system takes 1551 ms (improvement of 21%) to complete FFT executions, hibernating only 12 times. This demonstrates the performance improvement of the power-neutral system over [19]. A power-neutral system also reduces the energy overhead associated with hibernate and restore by reducing the number of hibernates. It is interesting to note that with larger current source, the existing system sustains its operation for a longer duration due to higher charge stored in the decoupling capacitor and thus, requiring fewer hibernate and restore operations. For same current values current values, the power-neutral system does not need to hibernate (and restore) and therefore, the corresponding entries are marked as 0. Beyond 800 μA, the existing system also sustains its operation for the entire duration, requiring no hibernate (or restore) and are therefore not included in the table.

A similar trend is observed for the other applications. To summarize, the power-neutral system can provide a speedup of up to 21%, by sustaining operation with reduced frequency, avoiding the need for multiple hibernate and restore operation.

H. Improvement of IoT Device Lifetime

Table VII reports the lifetime (measured as the absolute time from the end of the restore routine to the start of the hibernate routine) of the microcontroller in response to a sinusoidal source for one period of operation. For demonstration, five frequencies are selected from 1 Hz to 20 Hz. The result using the proposed power-neutral DFS approach is compared with the existing system [19]. The results are reported both as the absolute operating time (in ms) and as a percentage of the period of the sinusoidal source. As can be seen for both systems, the lifetime decreases with an increase in frequency. This is because the increasing frequency has a shorter period; the lifetime which represents a part of this time period also reduces. In comparison to the existing system, the power-neutral system increases the lifetime by 4% at 1 Hz (241.5 vs. 232.3 ms). However, at higher frequencies (20 Hz), the lifetime improvement is up to 88%. It is to be noted that with an increase in the frequency of operation, the fraction of time for which the microcontroller is active within a clock period increases, both for the existing system and the power-neutral system. However, the improvement for the power-neutral system is higher; because the power-neutral system adaptively modulates the frequency, thereby modulating the discharge characteristics of the on-board decoupling capacitance. Since the time spent at each frequency step is independent of the input frequency, the improvement of active time of the power-neutral system increases with frequency. This demonstrates that the power-neutral approach significantly improves the operational lifetime of systems.

TABLE VII: Microcontroller lifetime at different frequencies.

Input	Existing System [19]		Power-Neutral System		
Frequency (Hz)	Time (ms)	Percentage	Time (ms)	Percentage	Improvement
1	232.3	23%	241.5	24%	3.9%
2	114.2	23%	124.0	25%	8.6%
5	50.29	25%	66.85	33%	32.9%
10	24.50	25%	44.23	44%	80.5%
20	16.95	34%	31.82	64%	87.7%

VI. CONCLUSION

We have proposed a new paradigm for transiently-powered computing, called **power-neutral** operation, in which a microcontroller's frequency is dynamically adapted in response to the input power source. Power neutrality was achieved using a control strategy which scales the microcontroller frequency adaptively in response to changes in the supply voltage. Specifically, the control strategy gradually increases the operating frequency when the input voltage increases, thereby improving system performance. An analytical model was developed for the power-neutral system to determine the voltage thresholds for experiments. A detailed experimental validation was performed to demonstrate the power-neutral behavior in response to a sinusoidal voltage from a bench signal generator, and two real energy-harvesting sources (photovoltaic cell and micro wind-turbine). Results using standard applications demonstrated that a power-neutral system can extend its lifetime by 4–88% at different frequencies, with up to a 21% speedup in application execution, by sustaining operation with reduced frequency.

REFERENCES

[1] H. Jayakumar, K. Lee, W. S. Lee, A. Raha, Y. Kim, and V. Raghunathan, "Powering the internet of things," in *Proc. of Int. Symp. on Low Power Electronics and Design*, 2014.

[2] J. A. Stankovic, "Wireless sensor networks." *IEEE Computer*, 2008.

[3] L. Mainetti, L. Patrono, and A. Vilei, "Evolution of wireless sensor networks towards the internet of things: A survey," in *IEEE Int. Conf. on Software, Telecommunications and Computer Networks*, 2011.

[4] Y. Chen and Q. Zhao, "On the lifetime of wireless sensor networks,"*IEEE Commun. Lett.*, 2005.

[5] W. Dargie, "Dynamic power management in wireless sensor networks: State-of-the-art," *IEEE Sensors J.*, 2012.

[6] S. Beeby and N. White, *Energy Harvesting for Autonomous Systems*. Artech House, Incorporated, 2014.

[7] S. Sudevalayam and P. Kulkarni, "Energy harvesting sensor nodes: Survey and implications," *IEEE Commun. Surveys & Tutorials*, 2011.

[8] F. Ongaro, S. Saggini, and P. Mattavelli, "Li-ion battery-supercapacitor hybrid storage system for a long lifetime, photovoltaic-based wireless sensor network," *IEEE Trans. on Power Electron.*, 2012.

[9] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava, “Power management in energy harvesting sensor networks,” *ACM Trans. on Embedded Computing Systems*, vol. 6, no. 4, 2007.

[10] V. Sharma, U. Mukherji, V. Joseph, and S. Gupta, “Optimal energy management policies for energy harvesting sensor nodes,” *IEEE Trans. on Wireless Commun.*, vol. 9, no. 4, pp. 1326–1336, April 2010.

[11] S. Escolar, S. Chessa, and J. Carretero, “Energy-neutral networked wireless sensors,” *Simulation Modelling Practice and Theory*, 2014.

[12] L. Benini, A. Bogliolo, and G. De Micheli, “A survey of design techniques for system-level dynamic power management,” *IEEE Trans. on Very Large Scale Integr. (VLSI) Syst.*, 2000.

[13] J. Polastre, R. Szewczyk, and D. Culler, “Telos: enabling ultra-low power wireless research,” in *Int. Symp. on Information Processing in Sensor Networks*, 2005.

[14] P. Mitcheson, “Energy harvesting for human wearable and implantable bio-sensors,” in *IEEE Conf. on Engineering in Medicine and Biology Society*, 2010.

[15] G. Acampora, D. Cook, P. Rashidi, and A. Vasilakos, “A survey on ambient intelligence in healthcare,” *Proc. of the IEEE*, vol. 101, no. 12, pp. 2470–2494, 2013.

[16] D. Balsamo, G. Paci, L. Benini, and B. Davide, “Long term, low cost, passive environmental monitoring of heritage buildings for energy efficiency retrofitting,” in *IEEE Workshop on Environmental Energy and Structural Monitoring Systems*, 2013.

[17] S. Naderiparizi, A. N. Parks, Z. Kapetanovic, B. Ransford, and J. R. Smith, “Wispcam: A battery-free rfid camera,” in *IEEE Int. Conf. on RFID (RFID)*, 2015.

[18] A. Rodriguez, D. Balsamo, A. Das, A. S. Weddell, D. Brunelli, B. Al-Hashimi, and G. V. Merrett, “Approaches to transient computing for energy harvesting systems: A quantitative evaluation,” in *Int. Workshop on Energy Harvesting & Energy Neutral Sensing Systems*. ACM, 2015.

[19] D. Balsamo, A. S. Weddell, G. V. Merrett, B. M. Al-Hashimi, D. Brunelli, and L. Benini, “Hibernus: Sustaining computation during intermittent supply for energy-harvesting systems,” *IEEE Embedded Systems Letters*, 2015.

[20] B. Ransford, J. Sorber, and K. Fu, “Mementos: System support for long-running computation on rfid-scale devices,” in *Proc. of the Inter. Conf. on Architectural Support for Programming Languages and Operating Systems*, 2011.

[21] H. Jayakumar, A. Raha, and V. Raghunathan, “Quickrecall: A low overhead hw/sw approach for enabling computations across power cycles in transiently powered computers,” in *Inter. Conf. on VLSI Design*, 2014.

[22] C. K. Ho and R. Zhang, “Optimal energy allocation for wireless communications with energy harvesting constraints,” *IEEE Trans. on Signal Process.*, 2012.

[23] S. Kelly, N. Suryadevara, and S. Mukhopadhyay, “Towards the implementation of iot for environmental condition monitoring in homes,” *IEEE Sensors J.*, vol. 13, no. 10, pp. 3846–3853, 2013.

[24] Y. Cho, Y. Kim, and N. Chang, “Pvs: Passive voltage scaling for wireless sensor networks,” in *Proc. of the Int. Symp. on Low Power Electronics and Design*. New York, NY, USA: ACM, 2007, pp. 135–140.

[25] Y. Wu and W. Liu, “Efficient power management for wireless sensor node with wind energy harvesting,” *Int. J. of Sensor Networks*, vol. 12, no. 4, pp. 223–231, 2012.

[26] B. Zhang, R. Simon, and H. Aydin, “Harvesting-aware energy management for time-critical wireless sensor networks with joint voltage and modulation scaling,” *IEEE Trans. on Ind. Informat.*, vol. 9, no. 1, pp. 514–526, 2013.

[27] Y. Wang, Y. Liu, C. Wang, Z. Li, X. Sheng, H. Lee, N. Chang, and H. Yang, “Storage-less and converter-less photovoltaic energy harvesting with maximum power point tracking for internet of things,” *IEEE Trans. on Comput.-Aided Des. of Integr. Circuits Syst.*, 2015.

Domenico Balsamo Dr. Domenico Balsamo received the Master degree in Electronics Engineering from University of Modena and Reggio Emilia, Italy, in 2008. He received his Ph.D. degree in computer engineering in the area of embedded systems from the University of Bologna, Italy, in 2015. He is currently a post-doctoral research fellow at the University of Southampton. His research interest is high-performance and ultra-low power embedded systems.

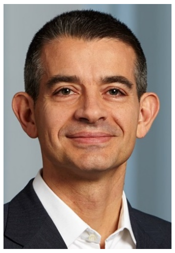
Anup Das Dr. Anup Das received the B.Eng. degree in Electronics and Telecommunication Engineering from Jadavpur University, India, in 2004. He received the Ph.D. degree in computer engineering in the area of embedded systems from the National University of Singapore, in 2014. He is currently a post-doctoral research fellow at the University of Southampton. His research interests include reliability and energy-aware system architecture.

Alex S. Weddell Alex Weddell (GSM’06–M’10) received the Ph.D. degree in Electrical and Electronic Engineering from the University of Southampton, UK, in 2010. He has industrial, consultancy, and project experience in energy harvesting, instrumentation, and wireless sensor networks, and has authored over 30 papers in the area. He is now a Lecturer in the Department of Electronics and Computer Science at the University of Southampton, UK. He has special interests in the areas of energy-aware systems and energy harvesting.

Davide Brunelli Davide Brunelli. (M10) received the M.S. (cum laude) and Ph.D. degrees in electrical engineering from the University of Bologna, Bologna, Italy, in 2002 and 2007, respectively. He has been an Assistant Professor with the University of Trento, Italy, since 2010. His research interests include smart grids and the development of new techniques of energy scavenging for wireless sensor networks and embedded systems.

Bashir M. Al-Hashimi Bashir M. Al-Hashimi (M99-SM01-F09) is a Professor of Computer Engineering and Dean of the Faculty of Physical Sciences and Engineering at University of Southampton, UK. He is ARM Professor of Computer Engineering and Co-Director of the ARM- ECS research centre. His research interests include methods and tools for low-power design and test of embedded computing systems.

Geoff V. Merrett Geoff V. Merrett (GSM06-M09) received the BEng degree (Hons) in Electronic Engineering and the PhD degree from the University of Southampton, UK, in 2004 and 2009 respectively. He was appointed as a Lecturer in energy-efficient electronic systems at the University of Southampton in 2008, and was promoted to Associate Professor in 2014. He has research interests in energy-efficient embedded systems and low-power pervasive computing.



Luca Benini Luca Benini is the chair of digital Circuits and systems at ETHZ and a Full Professor at the University of Bologna. He has served as Chief Architect for the Platform2012/STHORM project in STmicroelectronics, Grenoble. Dr. Benini’s research interests are in energy-efficient system design and Multi-Core SoC design. He is also active in the area of energy-efficient smart sensors and sensor networks for biomedical and ambient intelligence applications.