Enhancing Forward Progress of Transient Systems by Optimising Capacitor Size

Jie Zhan, Alex S. Weddell, *Member, IEEE,* and Geoff V. Merrett, *Member, IEEE*

***Abstract*—Batteryless energy-harvesting devices are promising for a sustainable IoT. Forward progress in such devices is maintained by transient computing, where volatile computing state is saved into and restored from nonvolatile memory before and after power failures. Current transient computing approaches typically minimise energy storage to reduce device dimensions and recharging time. However, minimising energy storage results in expensive state-saving and -restoring overheads and impedes forward progress, whereas provisioning more energy storage requires more space and longer recharging time though improves forward progress. In this paper, we develop a theoretical transient computing model to define the relationship between forward progress and energy storage capacity. Using this model, we show that sizing energy storage capacity can improve forward progress by up to 64.9%. We further propose a process of identifying the proper energy storage capacity which trades off forward progress, dimensions, and recharging time. We integrate the model into a photovoltaic-based energy-harvesting device framework to demonstrate the sizing process given various real-world light source datasets. Following this process, we show that sizing energy storage can gain up to 43.3% annual mean forward progress. Also, the sizing process achieves 98.3% of the maximum forward progress while saves 71.7% capacitor volume and 83.8% recharging time. The model is experimentally validated based on an MSP430FR6989 microcontroller, showing 0.5% mean absolute percentage error. The experimental results also show that a suggested 43 μF capacitor improves forward progress by up to 30.4% compared an on-board 10 μF one.**

***Index Terms*—intermittent computing, transient computing, energy harvesting, energy storage, forward progress, batteryless, wireless sensor networks, internet of things.**

I. Introduction

INTERNET of Things (IoT) devices are becoming ubiquitous, with tens of billions of devices to be installed at possibly hard-to-reach locations [1]–[4]. Conventionally, these devices are battery-powered, and thus have constrained lifespan and require impractical replacement work. To circumvent the limited battery lifespan, energy-harvesting IoT devices becomes a solution.

Environmentally harvested power is intrinsically variable and intermittent [5]. Traditionally, large energy storage, in forms of rechargeable batteries or supercapacitors, is adopted to buffer the variable harvested power and provide reliable power supply [6]–[11]. Unfortunately, such types of energy storage may raise pollution concerns, increase cost and device dimensions, and still have limited lifespan, yet using an energy-harvesting supply without energy buffering hinders execution by frequent power failures.

Recently, *transient computing* (also known as *intermittent computing*) has been developed in research with the aim of removing large energy storage while maintaining execution despite frequent power failures [12]–[17]. During active periods, transient computing systems save system volatile computing state, e.g. CPU register data, RAM data, into nonvolatile memory (NVM) either at pre-installed points (*static methods*) or when the supply is about to fail (*reactive methods*) [18]. The volatile state is lost when the supply voltage drops below the minimum operating threshold, while the saved state in NVM is retained. The saved state is restored from NVM when the supply voltage recovers to a restore threshold, and then the execution continues from the last saved point. In transient computing, forward progress denotes the effective program progress, as opposed to re-executed progress, lost progress, and the progress of state-saving and -restoring operations [19], [20]. The amount of forward progress directly determines application performance, e.g. program iteration rate or task completion time.

Current transient computing approaches typically adopt only a minimum amount of energy storage which is just enough for the most energy-expensive atomic operation[1] with the goal of minimising device dimensions and recharging time [13], [14], [21]–[25]. However, in exploration we found that, a device with minimum energy storage has to frequently wake up, restore state, execute shortly, save state, and halt, consuming much energy in saving and restoring operations. Provisioning more energy storage prolongs the operating cycles, reduces the state-saving and -restoring overheads, and hence, improves forward progress; however, larger energy storage also increases leakage current, occupies more space, and requires longer recharging time. The relationship between energy storage capacity and forward progress remains undefined, and how to size energy storage to improve forward progress while balance volume and recharging time is still a challenge.

In this paper, we develop a theoretical model of reactive transient computing to estimate forward progress. Taking advantage of the theoretical model, we explore the effect of energy storage capacity on forward progress with respect to supply current and volatile state size. We further propose a process of identifying the proper energy storage size for deploying energy-harvesting transient computing (EHTC) devices, which improves forward progress while balances dimensions and recharging time. We integrate the reactive transient computing model into a photovoltaic-based (PV-based) EHTC device framework. We demonstrate the sizing process with the

Software of this work will be open-source upon acceptance of this paper.

[1]Atomic operations in transient computing denote operations that should be completed in one continuous period. If an atomic operation is interrupted by a power failure, it should be re-executed rather than resumed. One example of atomic operations is saving and restoring volatile state.

framework given various real-world indoor and outdoor light source datasets.

In summary, the main contributions of this paper are:

- A theoretical model of reactive transient computing to estimate forward progress.
- An exploration with the model, which analyses the energy storage sizing effect on forward progress with respect to supply current and volatile state size, showing up to 64.9% forward progress improvement.
- A sizing process that identifies the optimal energy storage capacity in deploying EHTC devices, which trades off forward progress, capacitor volume, and recharging time.
- A demonstration of the sizing process with the theoretical model integrated into a PV-based EHTC device framework under various real-world light source datasets, where results show that sizing energy storage can improve annual mean forward progress by 7.8-43.3% compared to a minimum one, and the suggested capacity achieves 98.3% of the maximum forward progress while saves 71.7% capacitor volume and 83.8% recharging time.
- Experimental validation of the theoretical model, which shows high accuracy with 0.5% mean absolute percentage error (MAPE), and a 43 μF capacitor suggested by the sizing process improves forward progress by up to 55.2% and 30.4% compared to a theoretical minimum 6.2 μF one and an on-board 10 μF one across various levels of supply current.

The rest of this paper is organized as follows. Background and related work on transient computing and its modelling approaches are provided in Section II. The device framework and the theoretical model are illustrated in Section III. The sizing process is presented in Section IV. Model configuration and simulation setup are explained in Section V. Design exploration and demonstration of the sizing process are presented in Section VI. The theoretical model is experimentally validated in Section VII. Finally, Section VIII concludes this paper.

II. Background and Related Work

A. Transient Computing

The goal of transient computing is to maintain progress despite frequent power failures and insufficient power supply, such that the devices can operate even directly powered by energy harvesters without large energy storage. The basic mechanism is to back up volatile computing state (lost if the supply fails) as nonvolatile state (preserved if the supply fails) during active periods; when the supply recovers, the lost volatile state is restored from the saved nonvolatile state to resume execution. In principles, approaches in transient computing can be classified as *static* and *reactive*.

1) Static Transient Computing: Static approaches save volatile state into NVM at compile-time pre-installed points in a program, either by inserting checkpoints or decomposing a program into atomic tasks. After power failures, the execution resumes from the last saved checkpoint or the last task boundary. Advantages in static approaches include eliminating additional hardware and ensuring operation atomicity. However, the rollback of progress intrinsically causes code re-execution, which introduces issues on memory consistency and wastes energy on both lost progress and re-executed progress due to power failures. Also, if the code between two successive checkpoints or task boundaries exceeds the amount that the energy storage can guarantee, the progress may never proceed due to insufficient power input.

2) Reactive Transient Computing: In contrast to static approaches, reactive approaches save volatile state in NVM when supply is about to fail by monitoring supply voltage. Specifically, reactive transient computing saves volatile state and enters a low-power mode (execution halted) if supply voltage falls below a save threshold. This save threshold should be set high enough to save volatile state before power fails. By entering the low-power mode, reactive approaches avoid re-execution, and hence, typically make more forward progress than static approaches. In a comparison between static [26] and reactive [13] approaches, the reactive approach achieves a 2.5× mean speedup in computational workloads [27]. Therefore, reactive transient computing is the focus in this work.

B. Related Work on Modelling Transient Computing

A few models about transient computing have been proposed to explore design space. Su et al. [28] provide a model to explore the impact of sizing energy harvester and energy storage on a dual-channel solar-powered nonvolatile sensor node, but their exploring range of supercapacitors spans from 10F to 10^4F, which is significantly larger than a typical scale of energy storage (μF to mF scale) in transient computing. EH model [29], [30] explores forward progress with different parameter settings in transient computing, but the model only considers a single active period without considering inactive periods that scale down the effective forward progress, and hence, the model cannot predict the effective forward progress. Jackson et al. [31] also present a modelling approach to explore energy storage effect, where they suggest that novel batteries are preferable in order to offer reliable task completion and to improve energy utilization. However, they do not provide guidance on how to design energy storage, and their model lacks a detailed state-saving and -restoring mechanism in transient computing.

We provide a modelling approach to estimate forward progress of an EHTC device in real-world deployment, considering energy source conditions as well as the forward progressing behaviours transient computing. This model can be used to explore the effect of energy harvester and energy storage sizing on forward progress. Further, we provide a process of sizing energy storage in transient systems.

III. Modelling Transient Computing Devices

The goal of modelling is to estimate forward progress of EHTC devices in real-world deployment. In this section, we introduce the framework for general transient computing devices, and then present a load model of reactive transient computing for estimating forward progress based on its theoretical behaviours. In this paper, without being restricted to a specific workload, **we describe forward progress as the ratio of the effective execution time to the total time**, based on

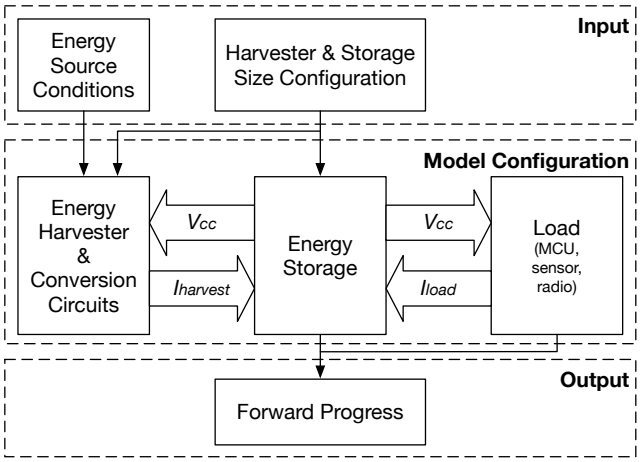


Fig. 1. Model framework of EHTC devices for estimating forward progress in real-world deployment.

TABLE I
MODEL PARAMETERS OF REACTIVE TRANSIENT COMPUTING

Input Parameters	
I_{harv}	Energy harvester current supply
C	Energy storage capacitance
Configuration Parameters	
I_{exe}	Execution current consumption
I_{sleep}	Sleep current consumption
I_r	Restore current consumption
I_s	Save current consumption
I_{leak}	Leakage current consumption
V_r	Restore voltage threshold
V_s	Save voltage threshold
T_r	Restore time overhead
T_s	Save time overhead
Output Parameter	
α_{exe}	Effective execution time ratio (forward progress)

an assumption that application progress is linear to effective execution time. The model is then used to explore the effect of energy storage capacity on forward progress.

A. Model Framework

The model framework is shown in Fig. 1. The model is driven by energy source conditions as a function of time. The configuration of this model includes three modules, *Energy Harvester and Conversion Circuits*, *Energy Storage*, and *Load*, which can be individually specified to represent the target platform. The three modules communicate by their voltage and current flows. This model outputs the time distribution of estimated forward progress over the test period of energy source input.

1) Model Inputs: This model takes two inputs, a) energy source conditions, and b) the size configuration of energy harvester and energy storage. The unit of energy source conditions should be consistent with the unit of the *Energy Harvester and Conversion Circuits* module. In this model, we assume that the energy source is equally distributed in the deployed space, such that increasing the size of energy harvester leads to more exposure to ambient energy, and vice versa. Note that the size configuration of energy harvester configures actual dimensions, e.g. PV panel area, while the one of energy storage configures capacity.

2) Model Configuration: Due to the variety of each module, the model configuration depends on the techniques implemented for each module.

Energy Harvester and Conversion Circuits. The energy harvester model transduces energy source conditions into electrical power. Typically, the power harvested from the energy harvester should be conditioned to provide a suitable voltage for charging the energy storage and supplying the load efficiently. However, in transient computing, such conversion circuits may be removed to increase power efficiency, using only a diode to prevent current backflow. The energy harvester and conversion circuits can be modelled together as a module because they are usually coupled and integrated.

Energy Storage. Energy storage in transient computing devices is usually in the form of a µF-scale capacitor. The energy storage provides the minimum length of an execution period, and should ensure the most energy expensive operation. Completes?

Load. The load module includes all the power consumers in a transient computing device, such as a microcontroller, sensors, and a radio. Techniques in transient computing dominate how the load consumes power and makes forward progress. As mentioned in Section I and Section II, transient computing approaches can be classified as *static* and *reactive*. Different approaches require different models for estimating forward progress.

B. Forward Progress of Reactive Transient Computing

We present a theoretical method to understand and model the forward progress in reactive transient computing given a constant current supply. This ~~model~~ includes *Energy Storage* and *Load* modules in the model framework. The parameters of this model are listed in TABLE I. The model inputs, i.e. I_{harv} and C, are related to the size configuration of energy harvester and energy storage respectively. The model output is α_{exe}, the mean forward progress when a constant current supply is applied. In the model we assume that all the parameters maintains constant mean values.

We use I_{in} to denote usable current input for equation simplicity where:

$$I_{in} = I_{harv} - I_{leak} \tag{1}$$

The capacitor leakage effect is discussed at the end of in this section.

1) Operating Modes of Transient Computing Devices: The computing behaviours of reactive transient computing devices can be classified as three operating modes given a spectrum of supply current. As shown in Fig. 2, an empirical experiment demonstrates the three operating modes. These three modes are divided by the relationship between the system current input and current consumption. We denote the three modes as *Off*, *Switch*, and *On*.

Off mode: When $I_{in} < I_{sleep}$, the device cannot wake up. The supply voltage V_{cc} cannot be charged up to the restore

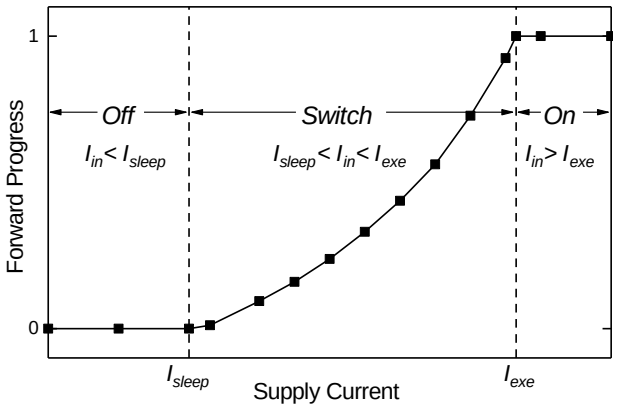


Fig. 2. Operating modes of reactive transient computing devices according to supply current and forward progress.

threshold V_r to start execution, so the device stays inactive. The sleep current I_{sleep} denotes the current consumption when the device waits for supply voltage to recover above the restore threshold, and hence, includes the consumption of voltage monitoring circuits.

Switch mode: When $I_{sleep} < I_{in} < I_{exe}$, the device executes transiently. V_{cc} can be charged up to V_r and the device starts execution. However, the energy in energy storage is then consumed by the load as $I_{in} < I_{exe}$, causing V_{cc} to drop below the save threshold V_r, where the device saves its state and sleeps. The device sleeps until V_{cc} charges to V_r again and then resumes the execution. In this mode, V_{cc} oscillates approximately between V_r and V_s, 'switching' on and off the execution. In general, higher I_{harv} leads to more forward progress in this mode, but the exact relationship between I_{harv} and forward progress requires further analysis.

On mode: When $I_{in} > I_{exe}$, the device execute constantly as the supply voltage V_{cc} never fails. The excessive power either dissipates through circuits or overcharges V_{cc}. An overcharged V_{cc} may affect harvesting efficiency (due to poor impedance matching) and reduce I_{harv}, such that current input and consumption are in equilibrium.

2) Formulating Forward Progress: We aim to calculate how much forward progress is made in relation to energy storage capacity and current input through a theoretical analysis. In the following analysis, we focus on the Switch mode ($I_{sleep} < I_{in} < I_{exe}$), as the forward progress of the On mode and the Off mode is straightforward (i.e. 1 in the On mode and 0 in the Off mode).

In the Switch mode, as shown in Fig. 3, the device goes through four intervals in turn, i.e. charging, restoring, executing, and saving, with current consumption of I_{sleep}, I_r, I_{exe}, and I_s in each interval respectively.

Let V_{pr} (post-restore) and V_{ps} (post-save) denote the voltage after restoring and saving operations. V_{pr} and V_{ps} can be calculated as:

$$V_{pr} = V_r + \frac{T_r(I_{in} - I_r)}{C} \tag{2}$$

$$V_{ps} = V_s + \frac{T_s(I_{in} - I_s)}{C} \tag{3}$$

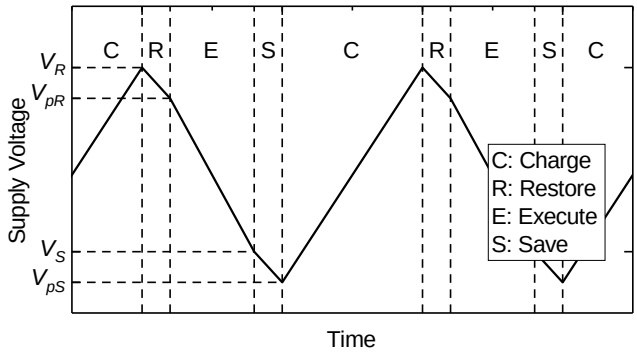


Fig. 3. Operating cycles in the Switch mode.

With Equation (2), the time spent on useful execution T_{exe} in one operating cycle can be expressed as:

$$\begin{aligned} T_{exe} &= \frac{C(V_{pr} - V_s)}{I_{exe} - I_{in}} \\ &= \frac{C(V_r - V_s) + T_r(I_{in} - I_r)}{I_{exe} - I_{in}} \end{aligned} \tag{4}$$

In Equation (4), $C(V_r - V_s)$ represents the amount of energy in the storage capacitor for restoring and executing. $T_r(I_{in} - I_r)$ represents the restoring cost. $I_{exe} - I_{in}$ is the current consumption for execution.

Also, with Equation (3), the charging interval can be described as:

$$\begin{aligned} T_{charge} &= \frac{C(V_r - V_{ps})}{I_{in} - I_{sleep}} \\ &= \frac{C(V_r - V_s) - T_s(I_{in} - I_s)}{I_{in} - I_{sleep}} \end{aligned} \tag{5}$$

Then, with Equation (4) and (5), the period of an operating cycle is:

$$\begin{aligned} T_{period} = &\ T_{charge} + T_r + T_{exe} + T_s \\ = &\ \frac{C(V_r - V_s) + T_s(I_{in} - I_s)}{I_{in} - I_{sleep}} + \\ &\ \frac{C(V_r - V_s) + T_r(I_{exe} - I_r)}{I_{exe} - I_{in}} \end{aligned} \tag{6}$$

Finally, with T_{exe} and T_{period} calculated in Equation (4) and (6), we obtain the percentage of time spent on effective execution (forward progress) in the Switch mode:

$$\alpha_{exe} = \frac{T_{exe}}{T_{period}}, \quad I_{sleep} < I_{in} < I_{exe} \tag{7}$$

Higher α_{exe} leads to more time spent on forward progress. As $d\alpha_{exe}/dI_{in}$ is positive, higher harvested current I_{harv} leads to more forward progress. To find out the energy storage effect on α_{exe}, we need to analyze $d\alpha_{exe}/dC$. Here, if we assume the leakage current maintains constant when storage capacity increases, α_{exe} keeps increasing with C to approach $(I_{in} - I_{sleep})/(I_{exe} - I_{sleep})$, which is an ideal case where restore and save overheads are zero. However, in an electrolyte capacitor, the leakage current typically increases with storage capacity C in the following relationship:

$$I_{leak} = kCV_{cc} \tag{8}$$

where k is a constant normally in a range from 0.01 to 0.03 ($\frac{A}{F \cdot V}$). Therefore, referring to Equation (1), dI_{in}/dC is $-kV_{cc}$, which is negative. Hence, considering capacitor leakage increases with capacitance, there is an optimal capacity that leads to the maximum forward progress α_{exe}.

To include the Off and On modes, the forward progress given all supply levels is presented as:

$$\alpha_{exe} = \begin{cases} 0 & , \quad Off\,(I_{in} < I_{sleep}) \\ \dfrac{T_{exe}}{T_{period}} & , \quad Switch\,(I_{sleep} < I_{in} < I_{exe}) \\ 1 & , \quad On\,(I_{in} > I_{exe}) \end{cases} \tag{9}$$

where T_{exe} and T_{period} are calculated by Equation (4) and (6) respectively.

IV. Sizing Process of Energy Storage

Leveraging our model, a process of identifying the optimal capacitor size that improves forward progress and balances the side effects, i.e. increased dimensions and recharging time, is presented. The process follows the steps below.

S1- Determine design specifications: Derive specifications according to a target application, such as minimum forward progress, minimum energy storage capacity, maximum capacitor dimensions, and maximum recharging time under certain energy source conditions.

S2- Configure model: Configure the model as explained in Section III according to the target platform and application, and collect energy source conditions of the environment where the device is deployed (demonstrated in Section V).

S3- Size Energy Harvester With the configured model, run tests with the minimum capacitance, and find the energy harvester size to ensure the minimum forward progress under the given energy source conditions.

S4- Optimise capacitor size: Generate the relationship between forward progress and capacitance. Evaluate the optimal capacitance by balancing the side effects of capacitor volume and recharging time with forward progress in a cost function. The cost function is given in Euqation (10):

$$f = \frac{\alpha_{exe}}{k_1} - (\frac{v_{cap}}{k_2})^2 - (\frac{T_{recharge}}{k_3})^2 \tag{10}$$

where v_{cap} represents capacitor volume and $T_{recharge}$ represents recharging time. k_1, k_2, and k_3 are linear scalers, which are empirically determined according to design specifications. The negative side effects are calculated in quadratic forms so as to punish high values heavier. We only consider the above three factors in this paper to size energy storage, but other factors (e.g. dimensions of energy harvesters) can be certainly included to consider a broader scenario.

We demonstrate this process to optimise the capacitor size in the next two sections (Section V and Section VI). We do not determine design specifications with specific numbers as we do not target a specific application, but we discuss these design specifications.

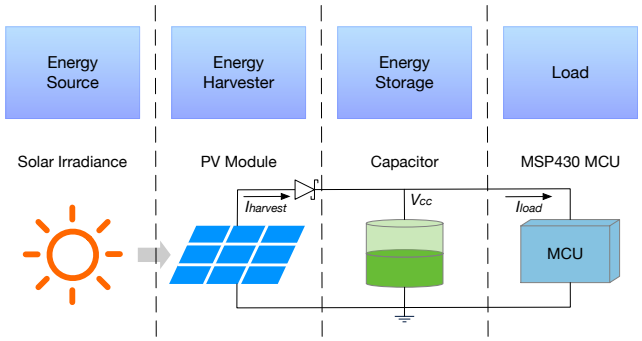


Fig. 4. Model configuration of a PV-based EHTC device.

V. Model Configuration

This section explains how this model framework is configured for design exploration in a PV-based EHTC device. Energy harvester and energy storage are modelled by existing models of PV cells and capacitors. We use a converter-less architecture where there is only a diode at the energy harvester output for preventing current backflow. The load is modelled by the forward progress formulation in Section III-B, with parameters profiled on a TI MSP430FR6989 MCU. A diagram of the configured model is shown in Fig. 4.

A. Energy Source and Energy Harvester

We import NREL outdoor solar irradiance data [32] and EnHANTs indoor irradiance data [33] as the energy source conditions input. To convert irradiance to energy harvester power output, we adopt a datasheet-based PV cell model [34] developed recently. This model takes the parameters available in common PV cell datasheets, so it is easily reconfigured to suit different PV cells. According to this model, the output current I_o of a PV cell can be described as:

$$I_o = \frac{G}{G_{ref}} I_{sc}(1 - (1 - \frac{I_{mpp}}{I_{sc}})^{\frac{V_o - V_{oc}}{V_{mpp} - V_{oc}}}) \tag{11}$$

where V_o is the output voltage of the PV cell, G is the ambient irradiance, G_{ref} is the reference irradiance (normally $1000W/m^2$), and I_{sc}, V_{oc}, I_{mpp}, V_{mpp} are respectively short-circuit current, open-circuit voltage, and the current and voltage at maximum power point (MPP) given the reference irradiance. V_o and G are dynamic at run time, while other parameters in this model are constant.

A PV panel is an array of PV cells, which amplifies voltage and current output by connecting PV cells in series or parallel. In a PV panel, the open-circuit voltage is proportional to the number of cells in series, and the short-circuit current is proportional to the area of each cell and the number of cells in parallel. We refer to a commercial solar cell [35] for PV cell properties as shown in TABLE II. We set four cells in series (with V_{oc} = 3.56V) to match the operating voltage of the MCU (maximum 3.6V), and model energy harvester sizing by scaling the cell area. A Schottky diode is connected to the energy harvester output in order to prevent current backflow.

TABLE II
PV CELL MODEL PROPERTIES

Parameter	Value
Open-Circuit Voltage	0.89 V/cell
Short-Circuit Current	14.8 mA/cm^2
MPP Voltage	0.65 V/cell
MPP Current	12.1 mA/cm^2

TABLE III
PROFILED MCU PARAMETERS

Parameter	Value
I_{exe}	887 µA
I_{sleep}	26 µA
I_r	971 µA
I_s	811 µA
T_r	1.903 ms
T_s	1.880 ms

B. Energy Storage

The energy storage is represented as an ideal capacitor with current leakage. The terminal voltage of this buffering capacitor is directly applied to the load, so this capacitor is modelled as:

$$C\frac{dV_{cc}}{dt} = I_{harv} - I_{load} - I_{leak} \quad (12)$$

where I_{load} is the current consumption of the load. The leakage current I_{leak} in electrolytic capacitors demonstrates complex physical phenomenon. In the model exploration, we refer to an off-the-shelf tantalum capacitor [36], with an empirical I_{leak} in relation to capacitance C, rated votlage V_{rated}, and terminal voltage V_{cc} [37]:

$$I_{leak} = 0.01CV_{rated}R_i \ (A) \quad (13)$$

$$R_i = 0.05 \times 20^{\frac{V_{cc}}{V_{rated}}} \quad (14)$$

where R_i is ratio of the actual current leakage at V_{cc} to the current leakage at V_{rated}. V_{rated} is chosen to be 10 V so as to operate the load (typically $<$ 4.0 V) around 25-40 % of the rated voltage for low leakage design [37].

C. Load

We implemented a reactive transient computing approach [14] on a TI MSP430FR6989 microcontroller. We profiled the model parameters by setting the MCU clock frequency at 8 MHz, running Dijkstra path finding algorithm with 1696B RAM usage in total. The supply voltage monitoring circuits use the MCU internal comparator and an external 3 MΩ voltage divider. The restore and save voltage thresholds are set as V_r = 2.4 V and V_s = 2.1 V respectively. The MCU shutdown voltage is V_{off} = 1.8 V.

The profiled parameters are shown in TABLE III. We measured the current consumption with a range of supply voltage. In experimental measurements, the variation of I_{sleep} between V_{off} and V_r is 2%, and the variation of I_{exe} between V_s and 3.3 V is 1.5%. I_{exe} also has a run-time variation of -2.4% to +2.8% with constant supply voltage due to a variable memory access rate. We omit such negligible variations and use the mean of I_{exe} and I_{sleep} in the model. I_r and I_s are measured at V_r and V_s respectively. Given the voltage thresholds and the current consumption, the minimum energy storage to guarantee save and restore operations is 6.2 µF.

Although we define the parameters in the model, however, these parameters can be tuned to explore different load characteristics. For example, T_r and T_s can be tuned to model different volatile state sizes.

Before the simulation, we developed and explored two simulation processes: (a) simulate system state with a fine-grained time step, and (b) sort energy source conditions into a distribution in time lengths and process the distribution. Process (b) shows a MAPE of only 0.892% compared to Process (a), while reducing simulation time significantly (e.g. from several hours to several seconds in a one-year test). Hence, we use Process (b) in the following tests.

VI. ENERGY STORAGE SIZING

This section first shows an exploration of how to improve forward progress by sizing energy storage with respect to supply current and volatile state size (Section VI-A), and then demonstrates the sizing process of energy storage in real-world energy source conditions. The sizing method finds the suitable energy harvester size that achieves a target forward progress and explores the energy storage sizing effect (Section VI-B), with a trade-off in forward progress, capacitor volume, and recharging time (Section VI-C).

A. Sizing Energy Storage to Improve Forward Progress

1) Impact of Supply Current: Increasing energy storage capacity improves forward progress by prolonging the period of operating cycles and reducing the number of restore and save operations compared to the minimum energy storage. The relationship of forward progress and energy storage capacity is plotted given a range of constant current supply in Fig. 5. Forward progress increases as energy storage capacity increases beyond the minimum level (6.2 µF in this case). However, this improvement is offset by increased capacitor leakage, which leads to an optimal storage capacity.

When capacity is less than the minimum (in Fig. 5, the area on the left of the dashed line), the execution may still progress given that the current supply keeps providing energy during execution. However, if the restore or save operation cannot be completed even with the energy input during execution, the forward progress directly falls to zero as the volatile state cannot be preserved completely. There is a steep downfall because the implemented control algorithm enters the low-power mode with volatile state retained after a save operation, and hence, as long as the supply voltage recovers to the restore threshold without a power failure (which is achieved by the constant current supply), the energy budget used for restoring state at first is then used for effective execution in the following operating cycles.

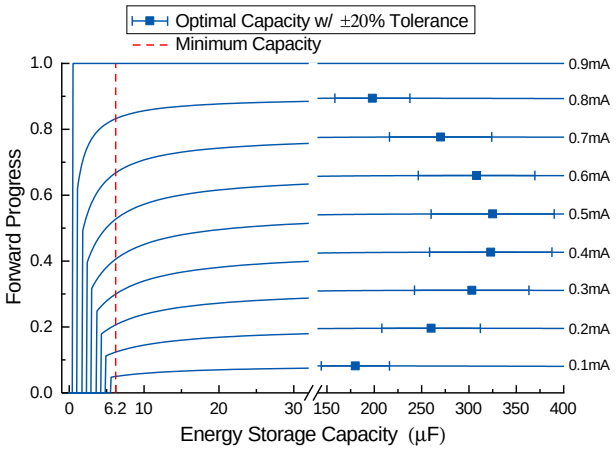


Fig. 5. Forward progress against energy storage capacity given different levels of constant supply current.

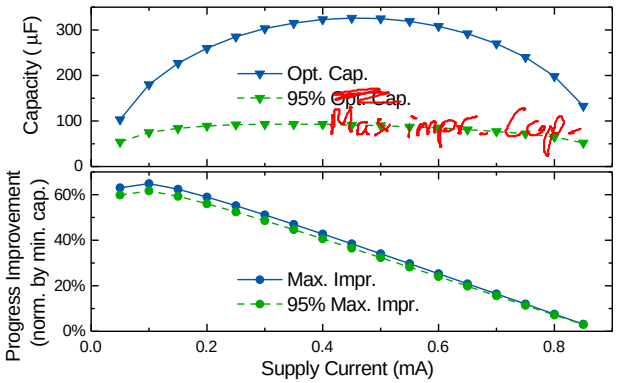


Fig. 6. Maximum forward progress improvement by sizing energy storage given a spectrum of supply current, with the corresponding optimal and sub-optimal (95% of maximum improvement) capacitance.

As a further illustration shown in Fig. 6, the maximum forward progress improvement, compared to using the minimum energy storage, can achieve up to 64.9% with regard to a range of supply current. Typically, capacitors in manufacture has ±20% tolerance of capacitance. This effect of capacitance variations on forward progress is also plotted in Fig. 5, where it is shown to be trivial ($< 0.23\%$ of the maximum values). Correspondingly, the optimal energy storage capacity is also plotted against supply current. It may not be preferable to optimise the energy storage capacitor with the sole goal of maximising forward progress because other concerns exist in increasing capacity, e.g. recharging time and dimensions (latter explained in Section VI-C). As we observe that the progress change is trivial around the optimal storage capacity, we also plot in Fig. 6 the storage capacity which achieves 95% of the maximum improvement, with a significant reduction of capacity (e.g. from 325 μF to 90 μF at 0.5 mA supply and 68.8% mean capacity reduction).

2) Impact of Volatile State Size: The size of volatile state differs across different applications due to their different amounts of RAM usage, and hence, incurs application-specific time overheads for restore and save operations. Although the time overheads are profiled for one application, the model can be tuned to explore other applications with various volatile state sizes. We modelled the volatile state size variations by linearly scaling restore and save time overheads, as the time overheads of restore and save operations are linear to the size of volatile state [18]. We measured time overheads of restore and save operations in the minimum case (64B register data and a 160B stack) and the maximum case (64B register data and 2048B full RAM) respectively as shown in TABLE IV.

TABLE IV
LINEAR SCALING RANGE OF VOLATILE STATE SIZE AND RESTORE/SAVE TIME OVERHEADS

State Size (Registers + SRAM)	Restore Time	Save Time
64B + 160B (lower bound)	232 μs	208 μs
64B + 2048B (upper bound)	2.298 ms	2.274 ms

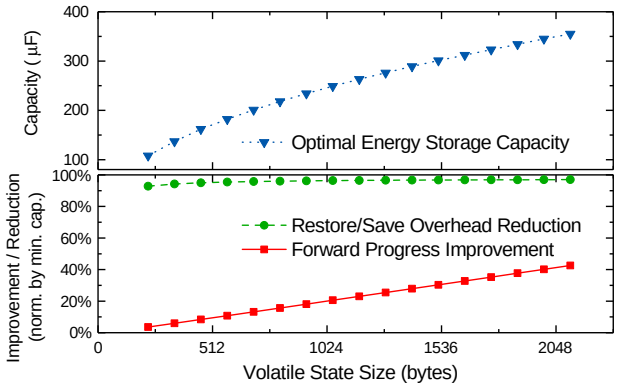


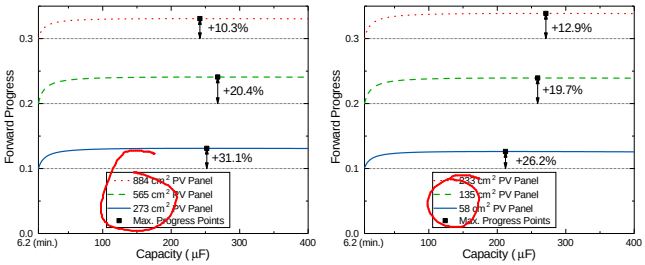
Fig. 7. Impact of RAM usage (linear to restore/save overhead) on sizing energy storage with 0.4 mA current supply.

An example of this exploration on volatile state size is plotted in Fig. 7 with 0.4 mA supply current. The forward progress improvement that can be gained from sizing energy storage increases with the volatile state size, and the optimal storage capacity grows accordingly. The improvement becomes insignificant when the volatile state size is small because the restore and save overheads are already negligible. For example, when the workload uses the least volatile state (the left end point in Fig. 7), the maximum progress improvement is only 3.6% although the restore and save overheads are reduced by 92.9%.

B. Optimising Energy Storage in Deploying EHTC Devices under Real-World Energy Source Conditions

In the practical deployment of EHTC devices, ambient energy source conditions change over time and locations. The energy harvester and energy storage should be configured accordingly in order to achieve desired forward progress across various energy source conditions.

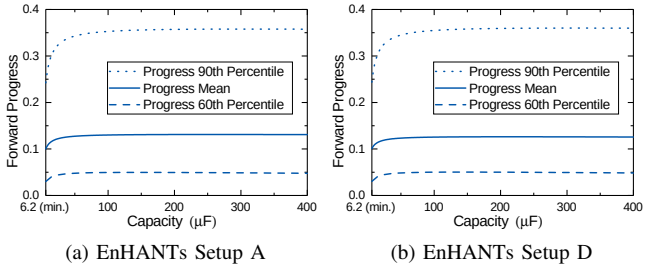
As a step of the sizing process, we use the aforementioned PV-based EHTC device model to find the PV panel area that achieves expected forward progress under different energy source conditions. To encompass different energy environments, four light source datasets across different environments are used in the following tests. To explore a range of target



(a) EnHANTs Setup A (b) EnHANTs Setup D

(c) NREL Denver 2018 (d) NREL Hawaii 2018

Fig. 8. Forward progress improvement by sizing energy storage given different PV panel areas under real-world energy source conditions. The model is able to find the PV panel area required for achieving the target mean forward progress.



(a) EnHANTs Setup A (b) EnHANTs Setup D

(c) NREL Denver 2018 (d) NREL Hawaii 2018

Fig. 9. Time percentiles of forward progress by sizing energy storage with target $\alpha_{exe} = 0.1$ and the corresponding PV panel area listed in Fig. 8. The percentiles start from the 60th as the system is off for around 55% of time due to insufficient energy source.

forward progress, three levels of minimum mean forward progress (i.e. target α_{exe}) are assumed to be 0.1, 0.2, and 0.3. We scale the PV panel area under the minimum energy storage to find the minimum PV panel area that achieves each target α_{exe}. As specified in Fig. 8, the energy harvester size that achieves desired forward progress may span over orders of magnitude given different energy source conditions (from mm^2 for outdoor sources to cm^2 for indoor sources).

We analyse the the sizing effect of energy storage on forward progress given real-world energy conditions. Fig. 8 shows 7.8-43.3% mean forward progress improvement by sizing energy storage under real-world energy conditions. A takeaway is optimising energy storage can either improve forward progress upon a given energy harvester size, or reduce the energy harvester size that achieves the target forward progress. Given strong energy sources (e.g. Denver 2018 and Hawaii 2018 outdoor solar sources), increasing energy harvester size efficiently improves forward progress with minor dimensional overheads, e.g. tens of mm^2; however, given weak energy sources (e.g. EnHANTs Setup A and Setup D indoor light sources), optimising energy storage capacity save tens of cm^2 of PV panel area to achieve the same forward progress.

The mean forward progress given target $\alpha_{exe} = 0.1$ is plotted in Fig. 9, with the 60th and 90th time percentiles of forward progress. In all the above datasets, the energy source is absent and the system is off for around 55% of time, so we plot the percentiles from the 60th and the 60th percentile progress is close to zero. The improved progress during those energy-available periods is averaged out by those energy-absent periods, so the actual amount of improvement when the systems operates in Switch mode is higher than the mean.

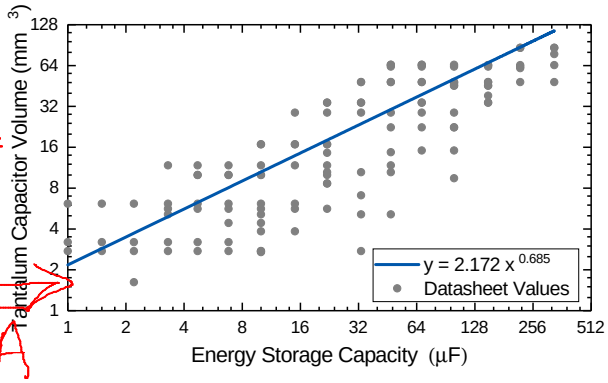


Fig. 10. Tantalum capacitor volume against capacitance [36], [38]–[42].

C. Trading off Forward Progress, Dimensions, and Recharging Time

Although increasing the storage capacity improves forward progress, larger capacitance may impact both dimensions and recharging time. We evaluate the overheads of increased capacitor dimensions and recharging time, and then trade off them with forward progress using a cost function to suggest an optimal capacitor size.

1) Metric of Dimensions: The overhead of capacitor dimensions is evaluated by characteristics of various off-the-shelf tantalum capacitors. We narrow down the range of sample capacitors within a set of characteristics: low-profile, 10V rated voltage, and Surface Mount Device (SMD) package, and select six series of capacitors [36], [38]–[42]. The volume and capacitance of these capacitors are plotted in Fig. 10 according to their datasheets. We use the regression of these data to represent the general capacitance-volume relationship in the sizing process.

2) Metric of Recharging Time: There is a recharging period between two consecutive execution periods in a transient

computing device. During the recharging period, the device saves a volatile state, wait for supply voltage to recover, and restore the state to resume execution, without making forward progress. Applications that require frequent sensing may be negatively affected by such delay. We evaluate and consider this recharging time in our sizing process.

Technically, the actual recharging time between two active periods depends on capacitance, the start and end voltage of recharging, and the current input and consumption. While the end voltage (restore voltage) and current consumption can be set or predicted, the start voltage and current flows vary with energy source conditions. Instead of evaluating the recharging time in complex conditions, we use the time between two successive execution intervals given 0.5 mA supply current in the Switch mode as the metric of recharging time (or recharging ability). Using Equation (4) and (6), we can obtain this recharging period between two successive execution intervals in the Switch mode as:

$$\begin{aligned} T_{recharge} &= T_{period} - T_{exe} \\ &= \frac{C(V_r - V_s) + T_s(I_{in} - I_s)}{I_{in} - I_{sleep}} + T_r \end{aligned} \tag{15}$$

3) Trade-offs: From the previous observations (Fig. 6) we can see that to achieve the optimal progress improvement costs much more storage capacity (mean 3.2×) than to achieve 95% improvement. A trade-off is necessary to improve progress while restricts the overheads of increased capacitor volume and recharging time.

As a part of the proposed sizing process, the cost function Equation (10) is used to trade off forward progress, capacitor volume, recharging time. We configure the function by setting k_1 = 0.2, k_2 = 200, and k_3 = 1, i.e.:

$$f = \frac{\alpha_{exe}}{0.2} - (\frac{v_{cap}}{200})^2 - T_{recharge}^2 \tag{16}$$

where v_{cap} is in mm^3 and $T_{recharge}$ is in second. k_1, k_2, and k_3 are scaling factors, and $\frac{1}{k_1}/\frac{1}{k_2}$ = 1000 does not mean that forward progress is 1000 times more important than capacitor volume.

The effect of the trade-off is plotted in Fig. 11 using Denver 2018 energy source dataset as an example. On average, the sizing process of energy storage achieves 98.3% of the maximum forward progress, while reduces 71.7% capacitor volume and 83.8% recharging time.

Compared to the minimum storage case, the sizing process improves mean forward progress by 5.7-25.3% with energy storage increased from 6.2 µF to 36-46 µF (depending on PV panel area). According to Fig. 10, the closest available capacitance that ensures the minimum storage is 6.8 µF, whereas the closest available capacitance to achieve the optimal capacitance are 33 µF or 47 µF. The minimum volume for these three capacitance are 2.75 mm^3, 2.75 mm^3, and 5.12 mm^3, which means using the optimal capacitance, instead of the minimum one, may not incur dimensional overhead (for 33 µF). For 47 µF, the absolute volume (5.12 mm^3) is still insignificant compared to the device as a whole, e.g. an MSP430FR6989 MCU chip alone occupies 14 × 14 × 1.4 mm^3 (274.4 mm^3). The regressed volume of the above three capacitance values are 8.1 mm^3, 23.8 mm^3, 30.4 mm^3 respectively. Again, such is still insignificant compared to the device as a whole.

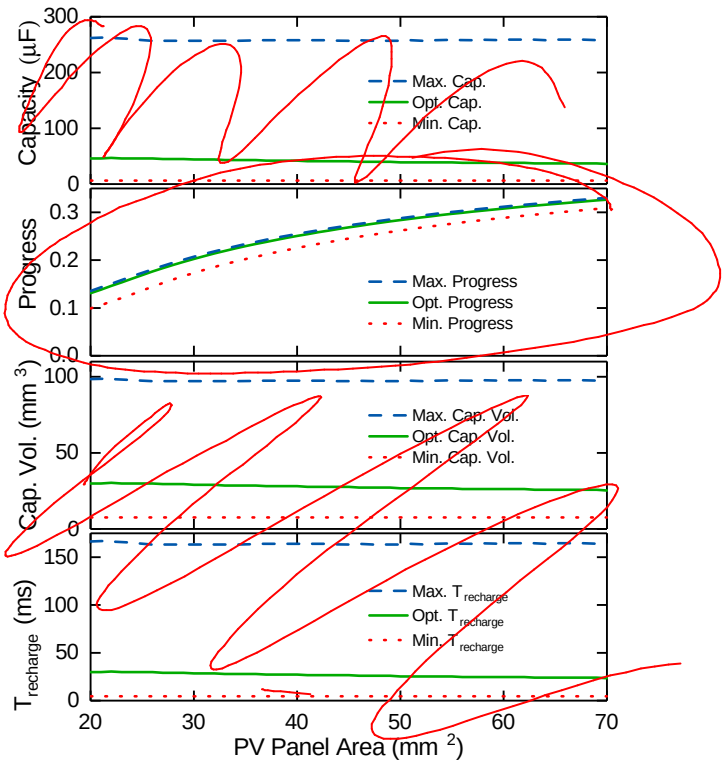


Fig. 11. The sizing process trades off forward progress, capacitor volume, and recharging time. The results are plotted against a range of PV panel area, given Denver 2018 energy source dataset.

VII. Model Validation

This section shows the comparison between the experimental and modelled forward progress to evaluate the accuracy of the proposed reactive transient computing model. As the sizing process suggests the optimal energy storage capacity in 36-46 µF, we select a 43 µF capacity in the experiment as an optimal value of the proposed sizing process for comparison. The forward progress with the optimal capacity is compared to that with a minimum one, an on-board one, and an ideal case given a range of supply current.

A. Experiment Setup

With the identical platform properties and workload settings in simulation setup, we used the TI MSP430FR6989 development board as the test platform. The on-board capacitance is measured to be 10.0 µF. This is the minimum capacitance that can be experimented on the platform. Additional capacitors were added to provide extra energy storage. The leakage current of these capacitors are less than 10 nA at 3 V, which is negligible compared to the current consumption of the platform (µA current draw), and hence, we omit the leakage of capacitors in the following experiment and model comparison.

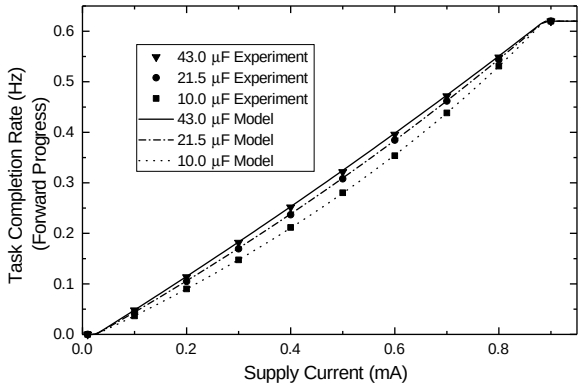


Fig. 12. Model validation by comparing the experimented and modelled task completion rates (forward progress) with different capacitance and supply current.

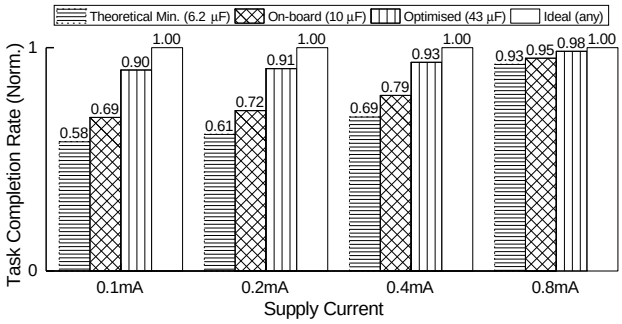


Fig. 13. Experimental comparison of task completion rates (forward progress) among different energy storage capacity.

In this practical experiment, the task completion rate (i.e. tasks completed per second) is used as the metric of forward progress rather than the effective execution time ratio. To gain the task completion rate from the model, we multiply the execution time ratio generated from our model by the completion rate when the device is executing constantly, since the task completion rate is linear to the effective execution time.

B. Validation

1) Model Accuracy: To validate the accuracy of our model, we powered the device with a range of current supply to operate the device in the Switch mode (0.1 to 0.8 mA, 0.1 mA per step), and repeated the tests with three energy storage capacities: a) on-board 10.0 µF capacitance only; b) 21.5 µF measured in total (11.5 µF added); c) 43.0 µF measured in total (33.0 µF added). We compared the actual program completion speed with the one that our model generated. As shown in Fig. 12, the model generated output matches closely with the experimental results with only 0.5% MAPE. Hence, the model is able to accurately estimate forward progress for design exploration.

2) Forward Progress Optimisation: We compared the tested completion speed among a theoretical minimum capacity case (6.2 µF, model generated), an on-board 10 µF, a 43 µF one suggested by the sizing process, and an ideal case. The ideal case switches between the sleep mode and the active mode without overheads of restoring and saving operations (mentioned in Section III-B). As shown in Fig. 13, a reasonably sized energy storage capacity (43µF) improves up to 55.2% and 30.4% more progress compared to the theoretical minimum capacitance and the platform on-board capacitance respectively, and also maintain at least 90% of the ideal forward progress. The forward progress improvement by sizing energy storage is significant when supply is weak and relatively insignificant when supply is strong.

VIII. Conclusion

In this paper, we proposed a model of reactive transient computing to estimate forward progress. Using this model, we explored the sizing effect of energy storage on forward progress in reactive transient computing with respect to supply current and volatile state size, showing up to 64.9% progress improvement under constant current supply and 7.8-43.3% improvement on annual mean forward progress under various real-world energy conditions. We proposed a process of sizing energy storage in deploying EHTC devices, which trades off forward progress, capacitor dimensions, and recharging time. We integrated the model into a PV-based EHTC device framework to demonstrate the sizing process, where results show that the suggested energy storage capacity achieves 98.3% of the maximum forward progress while saving capacitor volume by 71.7% and recharging time by 83.8%. We validated the reactive transient computing model, which demonstrates only 0.5% MAPE on forward progress compared to the experimentally measured values. Experimental results also showed that a reasonbly sized 43 µF capacitor improves forward progress by up to 55.2% and 30.4% compared to a theoretical minimum 6.2 µF one and an on-board 10 µF one across various levels of supply current. We believe that energy storage in transient systems should not be simply minimised or indiscriminately picked. Instead, the design of energy storage plays a significant role in practical deployment of EHTC devices.

References

[1] L. Mainetti, L. Patrono, and A. Vilei, "Evolution of wireless sensor networks towards the internet of things: A survey," in *SoftCOM 2011, 19th International Conference on Software, Telecommunications and Computer Networks*, Sep. 2011, pp. 1–6.

[2] O. Hahm, E. Baccelli, H. Petersen, and N. Tsiftes, "Operating systems for low-end devices in the internet of things: A survey," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 720–734, Oct 2016.

[3] T. Adegbija, A. Rogacs, C. Patel, and A. Gordon-Ross, "Microprocessor optimizations for the internet of things: A survey," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 1, pp. 7–20, Jan 2018.

[4] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 637–646, Oct 2016.

[5] Sravanthi Chalasani and J. M. Conrad, "A survey of energy harvesting sources for embedded systems," in *IEEE SoutheastCon 2008*, April 2008, pp. 442–447.

[6] S. Sudevalayam and P. Kulkarni, "Energy harvesting sensor nodes: Survey and implications," *IEEE Communications Surveys Tutorials*, vol. 13, no. 3, pp. 443–461, Third 2011.

[7] A. Kansal, J. Hsu, S. Zahedi, and M. B. Srivastava, "Power management in energy harvesting sensor networks," *ACM Trans.*

Embed. Comput. Syst., vol. 6, no. 4, Sep. 2007. [Online]. Available: http://doi.acm.org/10.1145/1274858.1274870

[8] B. Buchli, F. Sutton, J. Beutel, and L. Thiele, "Dynamic power management for long-term energy neutral operation of solar energy harvesting systems," in *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys '14. New York, NY, USA: ACM, 2014, pp. 31–45. [Online]. Available: http://doi.acm.org/10.1145/2668332.2668333

[9] P. Wägemann, T. Distler, H. Janker, P. Raffeck, V. Sieh, and W. SchröDer-Preikschat, "Operating energy-neutral real-time systems," *ACM Trans. Embed. Comput. Syst.*, vol. 17, no. 1, pp. 11:1–11:25, Aug. 2017. [Online]. Available: http://doi.acm.org/10.1145/3078631

[10] X. Jiang, J. Polastre, and D. Culler, "Perpetual environmentally powered sensor networks," in *Proceedings of the 4th International Symposium on Information Processing in Sensor Networks*, ser. IPSN '05. Piscataway, NJ, USA: IEEE Press, 2005. [Online]. Available: http://dl.acm.org/citation.cfm?id=1147685.1147765

[11] F. Simjee and P. H. Chou, "Everlast: Long-life, supercapacitor-operated wireless sensor node," in *Proceedings of the 2006 International Symposium on Low Power Electronics and Design*, ser. ISLPED '06. New York, NY, USA: ACM, 2006, pp. 197–202. [Online]. Available: http://doi.acm.org/10.1145/1165573.1165619

[12] B. Ransford, J. Sorber, and K. Fu, "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*, ser. ASPLOS XVI. New York, NY, USA: ACM, 2011, pp. 159–170. [Online]. Available: http://doi.acm.org/10.1145/1950365.1950386

[13] 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 *2014 27th International Conference on VLSI Design and 2014 13th International Conference on Embedded Systems*, Jan 2014, pp. 330–335.

[14] 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*, vol. 7, no. 1, pp. 15–18, March 2015.

[15] B. Lucia and B. Ransford, "A simpler, safer programming and execution model for intermittent systems," in *Proceedings of the 36th ACM SIGPLAN Conference on Programming Language Design and Implementation*, ser. PLDI '15. New York, NY, USA: ACM, 2015, pp. 575–585. [Online]. Available: http://doi.acm.org/10.1145/2737924.2737978

[16] Y. Wang, Y. Liu, S. Li, D. Zhang, B. Zhao, M. Chiang, Y. Yan, B. Sai, and H. Yang, "A 3us wake-up time nonvolatile processor based on ferroelectric flip-flops," in *2012 Proceedings of the ESSCIRC (ESSCIRC)*, Sep. 2012, pp. 149–152.

[17] J. V. D. Woude and M. Hicks, "Intermittent computation without hardware support or programmer intervention," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. Savannah, GA: USENIX Association, Nov. 2016, pp. 17–32. [Online]. Available: https://www.usenix.org/conference/osdi16/technical-sessions/presentation/vanderwoude

[18] S. T. Sliper, D. Balsamo, N. Nikoleris, W. Wang, A. S. Weddell, and G. V. Merrett, "Efficient state retention through paged memory management for reactive transient computing," in *Proceedings of the 56th Annual Design Automation Conference 2019*, ser. DAC '19. New York, NY, USA: ACM, 2019, pp. 26:1–26:6. [Online]. Available: http://doi.acm.org/10.1145/3316781.3317812

[19] K. Ma, X. Li, K. Swaminathan, Y. Zheng, S. Li, Y. Liu, Y. Xie, J. J. Sampson, and V. Narayanan, "Nonvolatile processor architectures: Efficient, reliable progress with unstable power," *IEEE Micro*, vol. 36, no. 3, pp. 72–83, May 2016.

[20] K. Ma, Y. Zheng, S. Li, K. Swaminathan, X. Li, Y. Liu, J. Sampson, Y. Xie, and V. Narayanan, "Architecture exploration for ambient energy harvesting nonvolatile processors," in *2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, Feb 2015, pp. 526–537.

[21] D. Balsamo, A. S. Weddell, A. Das, A. R. Arreola, D. Brunelli, B. M. Al-Hashimi, G. V. Merrett, and L. Benini, "Hibernus++: A self-calibrating and adaptive system for transiently-powered embedded devices," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 12, pp. 1968–1980, 2016.

[22] D. Balsamo, A. Das, A. S. Weddell, D. Brunelli, B. M. Al-Hashimi, G. V. Merrett, and L. Benini, "Graceful performance modulation for power-neutral transient computing systems," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 5, pp. 738–749, May 2016.

[23] J. Hester, L. Sitanayah, and J. Sorber, "Tragedy of the coulombs: Federating energy storage for tiny, intermittently-powered sensors," in *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '15. New York, NY, USA: Association for Computing Machinery, 2015, p. 5–16. [Online]. Available: https://doi.org/10.1145/2809695.2809707

[24] D. Balsamo, B. J. Fletcher, A. S. Weddell, G. Karatziolas, B. M. Al-Hashimi, and G. V. Merrett, "Momentum: Power-neutral performance scaling with intrinsic mppt for energy harvesting computing systems," *ACM Trans. Embed. Comput. Syst.*, vol. 17, no. 6, Jan. 2019. [Online]. Available: https://doi.org/10.1145/3281300

[25] K. Maeng and B. Lucia, "Adaptive dynamic checkpointing for safe efficient intermittent computing," in *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*. Carlsbad, CA: USENIX Association, Oct. 2018, pp. 129–144. [Online]. Available: https://www.usenix.org/conference/osdi18/presentation/maeng

[26] K. Maeng, A. Colin, and B. Lucia, "Alpaca: Intermittent execution without checkpoints," *Proc. ACM Program. Lang.*, vol. 1, no. OOPSLA, pp. 96:1–96:30, Oct. 2017. [Online]. Available: http://doi.acm.org/10.1145/3133920

[27] K. Maeng and B. Lucia, "Supporting peripherals in intermittent systems with just-in-time checkpoints," in *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, ser. PLDI 2019. New York, NY, USA: ACM, 2019, pp. 1101–1116. [Online]. Available: http://doi.acm.org/10.1145/3314221.3314613

[28] F. Su, Y. Liu, X. Sheng, H. G. Lee, N. Chang, and H. Yang, "A task failure rate aware dual-channel solar power system for nonvolatile sensor nodes," *ACM Trans. Embed. Comput. Syst.*, vol. 18, no. 4, pp. 33:1–33:21, Jul. 2019. [Online]. Available: http://doi.acm.org/10.1145/3320270

[29] J. San Miguel, K. Ganesan, M. Badr, and N. E. Jerger, "The EH model: Analytical exploration of energy-harvesting architectures," *IEEE Computer Architecture Letters*, vol. 17, no. 1, pp. 76–79, Jan 2018.

[30] J. San Miguel, K. Ganesan, M. Badr, C. Xia, R. Li, H. Hsiao, and N. Enright Jerger, "The EH model: Early design space exploration of intermittent processor architectures," in *2018 51st Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Oct 2018, pp. 600–612.

[31] N. Jackson, J. Adkins, and P. Dutta, "Capacity over capacitance for reliable energy harvesting sensors," in *Proceedings of the 18th International Conference on Information Processing in Sensor Networks*, ser. IPSN '19. New York, NY, USA: ACM, 2019, pp. 193–204. [Online]. Available: http://doi.acm.org/10.1145/3302506.3310400

[32] A. Andreas and T. Stoffel, "NREL solar radiation research laboratory (SRRL): Baseline measurement system (bms); golden, colorado (data)," *NREL Report NO.DA-5500-56488*, 1981.

[33] M. Gorlatova, A. Wallwater, and G. Zussman, "Networking low-power energy harvesting devices: Measurements and algorithms," *IEEE Transactions on Mobile Computing*, vol. 12, no. 9, pp. 1853–1865, Sep. 2013.

[34] S. Vergura, "A complete and simplified datasheet-based model of PV cells in variable environmental conditions for circuit simulation," *Energies*, vol. 9, no. 5, 2016. [Online]. Available: http://www.mdpi.com/1996-1073/9/5/326

[35] Panasonic, "Amorton amorphous silicon solar cells."

[36] AVX, "AVX TAJ low profile series tantalum capacitors," http://datasheets.avx.com/TAJ_LOW_PROFILE.pdf, last accessed: 2 Dec, 2019.

[37] R. Faltus, "Low leakage current aspect of designing with tantalum and niobium oxide capacitors," https://www.avx.com/docs/techinfo/Low_Leakage_Current_Aspect_Designing_Tantalum_Niobium_Oxide_Capacitors.pdf, last accessed: 28 Jan, 2019.

[38] AVX, "AVX TACmicrochip low profile series tantalum capacitors," http://datasheets.avx.com/TAC_LP.pdf, last accessed: 2 Dec, 2019.

[39] ——, "AVX F92 series tantalum capacitors," http://datasheets.avx.com/F92.pdf, last accessed: 2 Dec, 2019.

[40] Vishay, "Vishay 572D series tantalum capacitors," http://www.vishay.com/docs/40064/572d.pdf, last accessed: 2 Dec, 2019.

[41] ——, "Vishay 591D series tantalum capacitors," https://www.vishay.com/docs/40012/591d.pdf, last accessed: 2 Dec, 2019.

[42] ——, "Vishay 592D series tantalum capacitors," http://www.vishay.com/docs/40004/592d.pdf, last accessed: 2 Dec, 2019.