An Evaluation of Early Activity Completion Detection Algorithms for Low-Power Peripheral Devices in Embedded Systems

Daniel Ross Moore
Center for Efficient, Scalable and Reliable Computing
Dept. of Electrical and Computer Engineering
North Carolina State University, Raleigh, USA
e-mail: drmoore2@ncsu.edu

Alexander G. Dean
Center for Efficient, Scalable and Reliable Computing
Dept. of Electrical and Computer Engineering
North Carolina State University, Raleigh, USA
e-mail: agdean@ncsu.edu

***Abstract*—Embedded peripheral devices such as memories, sensors and communications interfaces are used to perform a function external to a host microcontroller. The device manufacturer typically specifies worst-case current consumption and latency estimates for each of these peripheral actions. Peripheral Activity Completion, Estimation and Recognition (PACER) is introduced as a suite of algorithms that can be applied to detect completed peripheral operations in real-time. By detecting activity completion, PACER enables the host to exploit slack between the worst-case estimate and the actual response time. These methods were tested independently and in conjunction with Intra-Operation Dynamic Voltage Scaling (IODVS) on multiple common peripheral devices. For the peripheral devices under test, the test fixture confirmed decreases in energy expenditures of up to 80% and latency reductions of up to 67%.**

Keywords-embedded systems; energy aware embedded computing; embedded profiling; embedded performance analysis; Dynamic Voltage Scaling (DVS); low-power; low-energy; wireless sensor node (WSN); adaptive embedded systems.

I. INTRODUCTION

Embedded systems are often constrained by timing and energy budgets because both factors affect the resultant cost and size of the system. Peripheral devices external to the microcontroller (MCU) such as those shown in Figure 1 can play a significant role in system-wide energy consumption. PACER [1] decreases dynamic power consumption and latency by exploiting slack between actual versus worst-case operation time. This contrasts with other methods that focus on decreasing static power usage of peripherals [2] [3] [4].

Device manufacturers derive and specify the worst-case operation duration by summing exacerbating factors including age, temperature and voltage. Using the worst-case operation time as a naïve guideline, the worst-case energy consumption of a given operation is characterized by (1).

$$E_{op-wc} = \int_0^{t_{op}} P_{op}(t)dt + \int_{t_{op}}^{t_{slack}} P_{slack}(t)dt \qquad (1)$$

Where t_{op} and P_{op} are the time and power comprising the actual operation while t_{slack} and P_{slack} are the time and power comprising the period between operation completion and the worst-case execution time.

Most peripheral devices provide a mechanism for signaling that operations completed earlier than the maximum. However, using these mechanisms results in sub-optimal power performance. For example, a common method of detecting write completion on external non-volatile memory relies on polling a status register. Performing this signaled method has power and energy consequences:

$$P_{overhead} = P_{MCU} + P_{MCD} + P_{Comm} + P_{Match} + P_{Dev} \qquad (2)$$

- P_{MCU}: MCU must be active while polling
- P_{MCD}: MCU communications driver must be active
- P_{Comm}: Communications incurs $P = cfV_{dd}{}^2$ penalty
- P_{Match}: MCU and device voltages must be matched.
  - Neither can use dynamic voltage scaling
- P_{Dev}: Device communications driver must be active

$$E_{op-sig} = \int_0^{t_{op}} \big(P_{op}(t) + P_{overhead}(t)\big)dt \qquad (3)$$

The components of $P_{overhead}$ are highly variable between microcontrollers, systems and devices. The signal may involve protocol-level communication or it may be as simple as an interrupt pin and that signal may traverse PCB traces with considerable capacitance. E_{op-} can exceed E_{op-wc}.

Both interface methods incur a power penalty and the naïve worst-case method also incurs a latency penalty. As the energy cost of computation continues to decrease in modern microcontrollers, it becomes more rewarding to use onboard intelligence to minimize the impact of power and latency penalties. PACER develops adaptive timing, current usage and charge consumption heuristics for estimating or recognizing early completion of peripheral operations, thus reducing total latency and energy consumption.

The prediction is verified in real-time against the actual state and the heuristic is updated with the results. In this fashion, the algorithms are resistant to variations in behavior that may occur across the lifecycle of the device. PACER is evaluated against a variety of embedded peripherals and is shown to significantly decrease both energy consumption and latency of peripherals with minimal computational overhead.

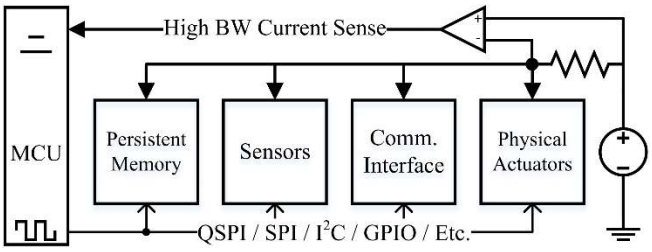

Figure 1: Typical Embedded System with Device Current Feedback

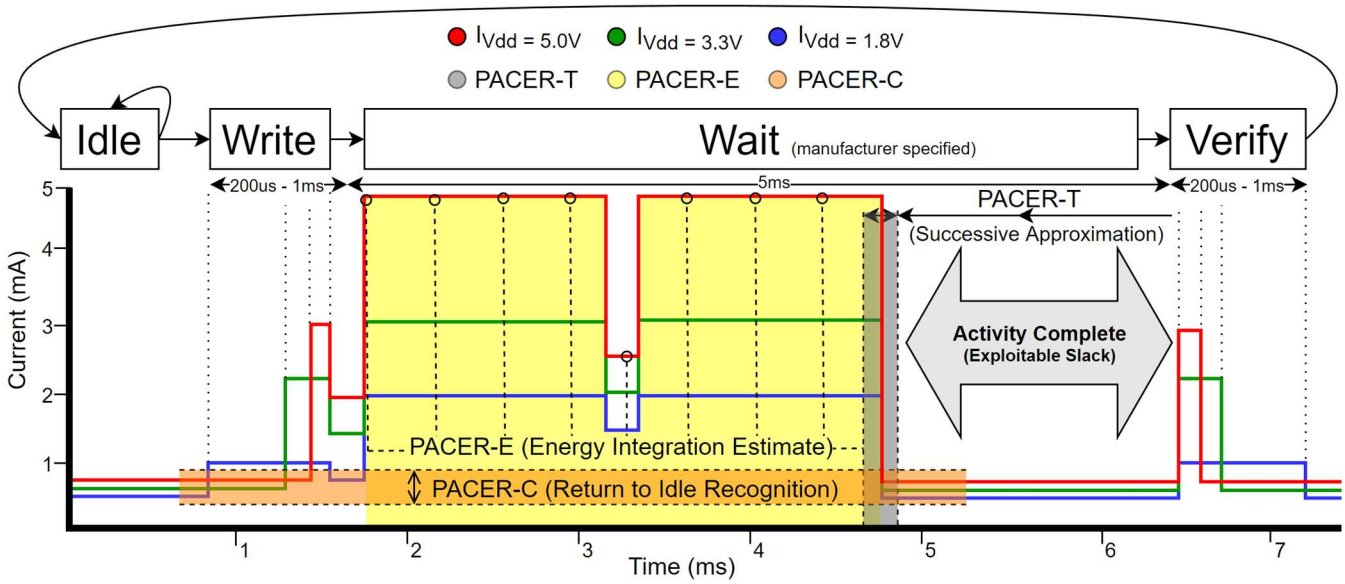

Figure 2: A Typical External Memory Transaction with IODVS and PACER

Figure 2 shows the current profile for the common peripheral operation of writing a page to EEPROM. The manufacturer-specified mandatory wait period is 5ms, beginning about the 1.75ms mark. As the device transitions through the Idle → Write → Wait → Verify states, it can be inferred from the current profile that the operation completed by the 4.75ms mark and that it was not necessary to delay until approximately 6.5ms per the specification. This 1.75ms differential is slack that can be exploited to decrease latency.

There are a wide variety of peripheral devices with a correspondingly wide variety of completion determinism and current profiles. PACER introduces three methods by which the host MCU can estimate or detect early completion of peripheral operations while also minimizing computational overhead. Devices with highly deterministic timing respond best to the timing heuristic while those with variable timing respond best to current or charge heuristics. Through low-overhead early completion detection, PACER is able to decrease both latency and system-wide energy consumption.

II. Related Work

Intra-Operation Dynamic Voltage Scaling [5] (IODVS) has been shown to significantly reduce the energy consumption of embedded peripherals (Flash, EEPROM, sensors, etc.) during their voltage-independent states. These states typically occur during mandatory delay periods while the peripheral completes a specified operation. When implementing IODVS, the host MCU and peripheral devices are placed on different voltage domains throughout the course of the voltage-independent state. Because of this, it is not possible for the MCU to poll the peripheral device for operation completion. Polling is also shown to be a rather costly operation due to (2) and (3). Without the ability to communicate to the peripheral device, PACER is necessary to achieve minimal operation latencies.

A. *Timing Heuristic*

Peripheral operations can vary in their latency or completion times due to a number of factors. Temperature can significantly affect the completion time for peripherals with deterministic timing requirements such as DRAM [6]. Device aging can also affect timing due to a number of issues resulting from fundamental semiconductor physics [7]. Furthermore, some devices simply have non-deterministic completion times due to features such as MMUs and caches that are implemented in various data storage devices like Micro-SD cards, or age and wear as they effect FLASH storage timing.

Because the latency can vary significantly between operations, it is necessary to develop a timing heuristic that can adapt to slowly changing effects like age and temperature as well as rapidly changing factors like cache hits and misses. Adaptive delay estimation is not a new problem [8] and research continues to compensate for non-deterministic delay with different approaches for wireless communications, control systems and mass storage latency [9].

B. *Energy Heuristic*

For devices with highly variable timing and dynamic current consumption characteristics, integrating the current consumption of the device throughout an operation can allow for better detection of completion. Some operations can be characterized by the amount of charge necessary to complete them. This technique is referred to as “coulomb counting” and is a common technique used to determine the state of charge in rechargeable batteries [10].

C. *Current Heuristic*

The completion of some peripheral operations are easily detectable by their current consumption profile. These devices have a distinct and deterministic current profile that can be characterized and used to estimate the moment when an operation completes.

Simple and differential power analysis (SPA and DPA) attacks are performed by monitoring device current consumption with very fine grained detail. These attacks seek to undermine encryption techniques by monitoring the current consumption of the processor and detecting the moment at which the processor executes a branch operation [11]. The attacks have been performed on an ARM Cortex MCU using AES and required an extensive measurement setup to accomplish [12]. PACER is inspired by this previous work using fine-grained in-circuit current measurement and fortunately benefits from much more lenient sampling requirements.

III. METHODS

A. Timing Heuristic PACER-T

Some peripheral operations exhibit highly deterministic timing qualities. Such a device is likely to be internally clocked and the operation is waiting for some number of clock cycles to expire before signaling that the operation completed. Such operations are typified Figure 3 in that neither the total energy consumed, nor the profile of that consumption are necessary to predict completion. Regardless of the power profile, the operation always completes within a deterministic window of time. Erases and write operations to EEPROM and flash are typical examples of this behavior.

PACER-T uses the successive approximation algorithm shown in (4) to determine the optimal delay for an operation. The algorithm begins by executing an operation with the amount of delay specified in the device datasheet. After each iteration, if the operation completed earlier than predicted (Pass), then the amount of delay is halved. Otherwise, the operation was ongoing (Fail) and the next delay is increased by half the distance to the last previously successful operation.

$$\begin{aligned} Pass: &\begin{cases} T_{upper} = T_{lower} \\ T_{lower} = T_{lower} - \dfrac{(T_{upper} - T_{lower})}{2} \end{cases} \\ Fail: &\begin{cases} T_{lower} = T_{lower} + \dfrac{(T_{upper} - T_{lower})}{2} \end{cases} \\ Initial\ Conditions: &\ T_{upper} = T_{worst-case}, T_{lower} = 0 \end{aligned} \tag{4}$$

The algorithm is executed online and provides the tightest possible timing. Upon expiration of the predicted wait period, if the device status register indicates that the operation is still occurring, then the algorithm has yielded an early prediction and it is appropriate to continue to wait. This would be considered the 'Fail' case of (4) and future estimates are increased. Otherwise, if the device status register indicates that the operation is complete, then the algorithm has yielded a late prediction and it is appropriate to reduce future estimates.

B. Energy Heuristic PACER-E

Operations that consume a deterministic amount of energy are better characterized by PACER-E. For example, the operation might involve the charging of a storage element such as an inductor or capacitor. In any case, a certain amount of energy is required to complete the operation and once that energy requirement has been satisfied, the peripheral device considers the operation to be complete. Figure 4 is an example of an energy bound operation.

The energy based heuristic was performed similarly to PACER-T in that successive approximation is used. The system multiply-accumulates voltage and current samples fed to the peripheral device. When the digital integration has reached the test value, the operation is 'complete' and checked for correctness. The mechanics of (4) are applied to PACER-E, except that all T limits are replaced with E energy limits. PACER-E is slightly less precise than the timing based algorithm due to the time required to both sample and perform the digital integration necessary for threshold checking.

The energy consumed throughout a test is calculated using the fundamental relationship shown in (5). The results were calculated offline via (6) and (7), where S is the state of the device, and T_s is the sampling period.

$$P = VI = \frac{E}{t} \tag{5}$$

$$E_s = \sum_{n=0}^{N-1} V_n I_n T_s \tag{6}$$

$$E_{total} = \sum_{S_0}^{S_{n-1}} E_s \tag{7}$$

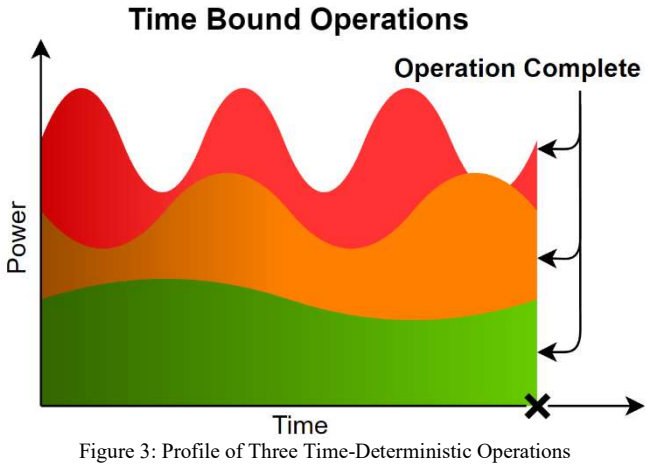

Figure 3: Profile of Three Time-Deterministic Operations

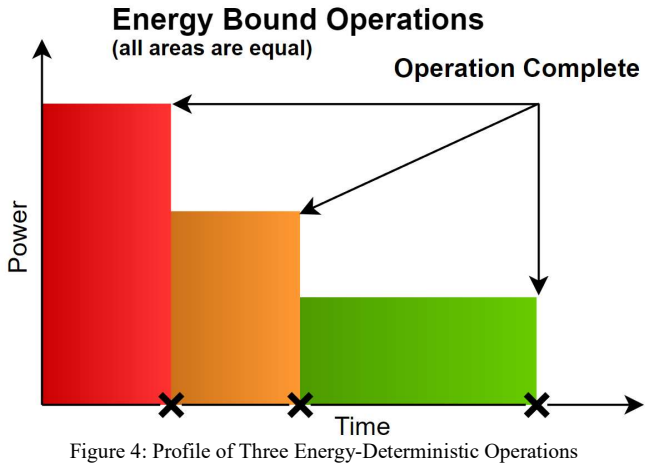

Figure 4: Profile of Three Energy-Deterministic Operations

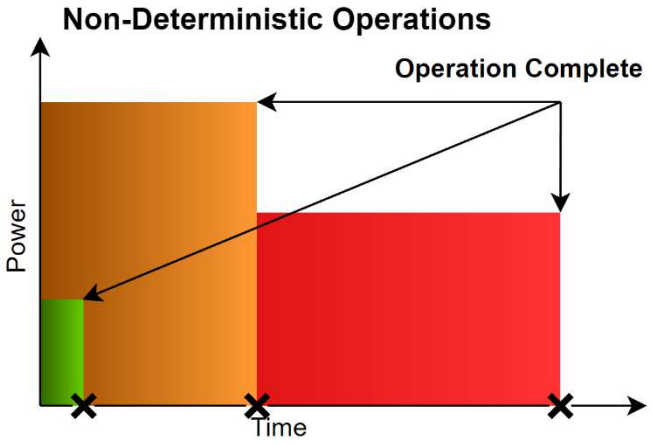

Figure 5: Profile of Three Non-Deterministic Operations

C. *Current Heuristic PACER-C*

Some operations cannot reliably be defined in terms of time nor energy. One example of a non-deterministic operation would be communications tasks performed by Ethernet or wireless devices that have non-deterministic transmission latencies. Another example would be memory devices that incorporate an onboard memory hierarchy. In such devices, operations are affected by cache latencies.

PACER-C provides recognition that the operation is complete by measuring the idle current usage of the device before the operation begins and marking the operation as complete after the current returns to idle. In order to accommodate operations where the current returns to idle and yet the operation has not yet completed, the algorithm incorporates both a minimum latency and an idle current percent threshold to mark the operation as complete.

Algorithm 1: *PACER-C*

1: ICT = (Idle Current Measurement) * threshold
2: Execute Peripheral Device Operation
2: **While (**t < Minimum Latency) and (I > ICT) **then**
3: I = New Current Measurement
4: **End While**

PACER-C is described by Algorithm 1 and begins by taking a sample of the device input current while idle. Next, the operation is executed and the algorithm waits for a minimum latency period to expire. The operation is considered complete after the output current returns to the threshold percentage of its previous state. The threshold for all following experiments was set empirically at 110%.

PACER-C is the most basic method to determine in real time if an operation is complete and may also be prone to false positives in some cases. There are many more advanced algorithms that can suit the purpose such as a multi-layer perceptron that is used in neural networks that could be used to identify features in real-time. It is notable however, that reducing the complexity of the detector is important so that the algorithm can ensure that it is keeping pace with incoming samples. Naturally, more complex algorithms could be accommodated by a more powerful host microcontroller.

IV. MATERIALS

PACER and IODVS are implemented on an STM32F429 MCU supported by the STMicroelectronics DISCO board and hosted by the PRIME (Precise Real-Time In-Circuit Micro-EMS) assembly. The board provides 64MB of SDRAM which allows for simultaneous sampling throughout the test suite at very high speed. All experiments were sampled at 1MSPS and the SDRAM allowed any individual experiment to last up to 1 full second. All of the analog conversions as well as the device state sampling were performed via DMA. Therefore, the test fixture is expected to have had no impact on the operation under test.

The PRIME assembly, shown in Figure 6, hosts a variety of peripherals (labelled in red as DUT: Devices Under Test) that are common in embedded designs. The board provides access to Bluetooth, Wi-Fi and a Si1143 proximity detector. PACER was evaluated on NAND and NOR FLASH memories, as well as a commercial EEPROM, temperature / humidity sensor and four independent Micro-SD cards.

At 1 MSPS and 4 channel measurements and 2 bytes per sample, each test can result in up to 8 megabytes of data. Because repeatability is so important, each test was run 50 times. Therefore, bandwidth became a limiting factor and a Hi-Speed (480Mbps) USB module was added to the board to allow for rapid development. Operating as a virtual communications port and using MCU parallel bus, actual bandwidth was realized at approximately 120Mbps.

Each of the peripheral devices under test has some method of determining if an operation completed successfully. For the memory devices, a simple read-back verification is sufficient to determine correctness and is a common practice among embedded designs. The temperature and humidity sensor provides a status bit indicating if an operation is in progress, thus indicating that a requested operation has not yet completed.

Power is provided and voltage is modulated to each individual device on the domain using independently configurable power supplies. The ASDM-300F module shown in orange on Figure 6 provides a high-efficiency buck power supply, followed by a linear regulator with a high ripple-rejection ratio. A high-precision and clean power

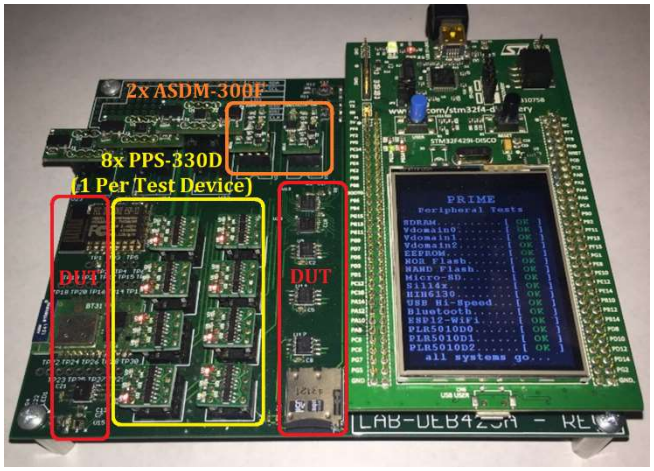

Figure 6: PRIME (Precise Real-Time In-Circuit Micro-EMS)

supply is important because PACER uses the current profile to make real-time decisions. If the power supply outputs a significant amount of noise, then it becomes difficult to acquire signal and determine activity completion in real-time.

The ASDM-300F is also outfitted with a dual current measurement circuit using the Maxim MAX4377HAUA+. This circuit allows the host to measure both the input and output current of the power supply with high analog bandwidth. Figure 8 shows peripheral current output (PIOx) and peripheral current input (PIIx) along with peripheral voltage (PVx) routed to their associated ADCs on the MCU. Note that all signals are buffered to provide high-driving capability and therefore a fast response time.

Ultimately, these measurements are used to determine activity completion with the PACER-E and PACER-C algorithms. It is important to note the gain-bandwidth product of the amplifier. High frequency content will be attenuated to some degree and the actionable data output would be of higher quality if a higher frequency device were available.

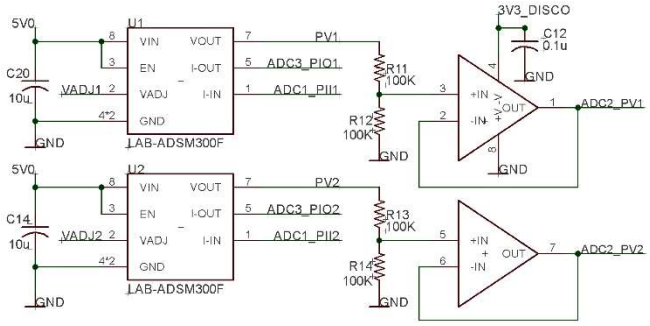

Figure 8: Voltage and Current Measurements from ASDM-300F

While measuring and classifying activity completion, it is important that each device be analyzed independently. The PPS-330D shown in yellow on Figure 6 allows the host to switch the voltage domain of an individual peripheral to any one of three domains, or disconnect the device entirely

PPS-330D devices are connected to each peripheral, and while a peripheral is under test, the remaining devices are switched to an alternate voltage domain. Thus, each device is independently classified in-system without physically removing other devices that may affect current measurements. Once the devices are characterized, then their individual contributions to the power supply current output can be deduced through superposition. The PPS-330D is convenient for initial profiling, but unnecessary for a streamlined implementation. The ASDM-300F is necessary for an IODVS implementation, but PACER-E and PACER-C only require the current measurement component.

V. Results

Initial IODVS results were repeated so as to establish a baseline with which to compare the results of PACER. Previous experiments required the results to be averaged many times over. The PRIME assembly provides high enough signal to noise ratio that averaging multiple test results is unnecessary and a simple 50-sample moving average provides enough filtering while maintaining a quick response time.

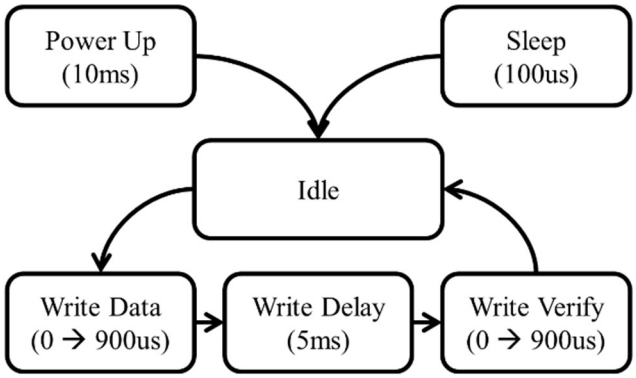

Figure 7: EEPROM Write State Transition Diagram

A. MCP25AA512 EEPROM

The Microchip EEPROM is specified by the manufacturer for a 5ms mandatory wait period following the write command and data. The full state transition diagram is shown in Figure 7. Upon receiving a write command and associated data, the delay begins and the operation completes after a verification stage. This operation is highly deterministic with respect to time, energy and current profile.

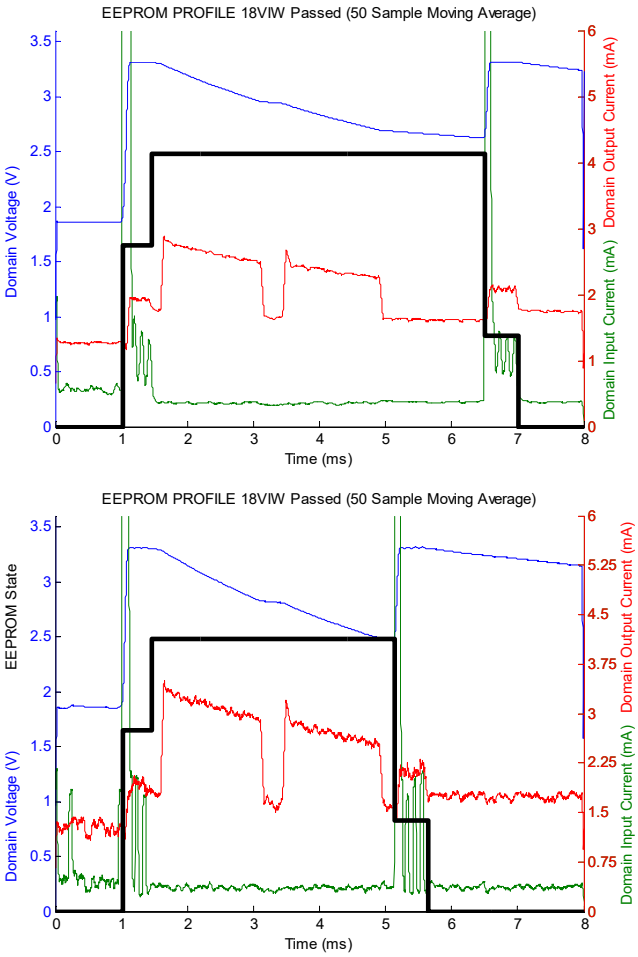

Figure 9: EEPROM Write Cycle Using IODVS and PACER-T

TABLE I. MCP25AA512 EEPROM PACER RESULTS

Stage	Latency Results (ms)				
	Control	*PACER-T*	*Diff.*	*PACER+IODVS*	*Diff.*
Wait	5.05	3.51	30.5%	3.51	30.5%
All	5.98	4.44	25.7%	4.44	25.7%
	Energy Results (uJ)				
Wait	46.84	37.89	19.1%	27.85	40.5%
All	53.05	43.91	17.2%	32.40	38.9%

The current waveform of Figure 9 shows that the EEPROM write operation begins at t=1.5ms and indicates completion at approximately t=5ms instead of t=6.5ms as specified by the manufacturer. After applying the PACER-T algorithm, it is indeed true that the operation was complete at the 5ms mark, thus reducing the wait latency by 30.5%.

Due to the deterministic nature of the operation, all algorithms identified activity completion with high accuracy as summarized in TABLE I. The PACER-E and PACER-C algorithms were also successful in identifying activity completion with latency reductions of 25.7% and 23.2%, respectively. The two algorithms do require additional computation to integrate or otherwise observe the current waveform. Given identical performance, PACER-T is the best choice in this application.

B. Numonyx M25PX16 NOR Serial Flash

NOR flash modules sacrifice byte-wise modification for overall capacity. The M25PX16 presents 16MBits of capacity in a small package, but the host must erase sub-sectors of flash (4K) to write pages of flash (128B). To perform a read-modify-write operation, the host must read the contents of a sub-sector, modify the contents locally, erase the sub-sector in flash and finally write the modified contents back to the flash on a page-by-page basis. The complete state transition diagram is shown below in Figure 10.

Both the sub-sector erase and page write have a worst-case delay specified by the manufacturer. PACER algorithms were run against both operations to find the comprehensive result. Although specified for 150ms, the current waveform indicates

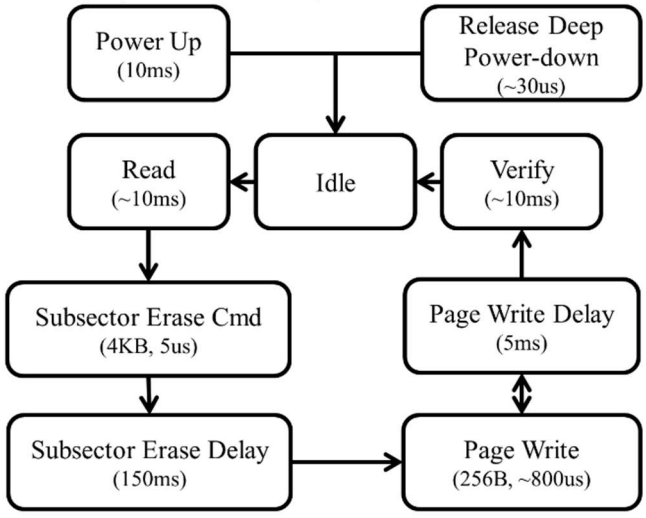

Figure 10: Serial Flash Write State Transition Diagram

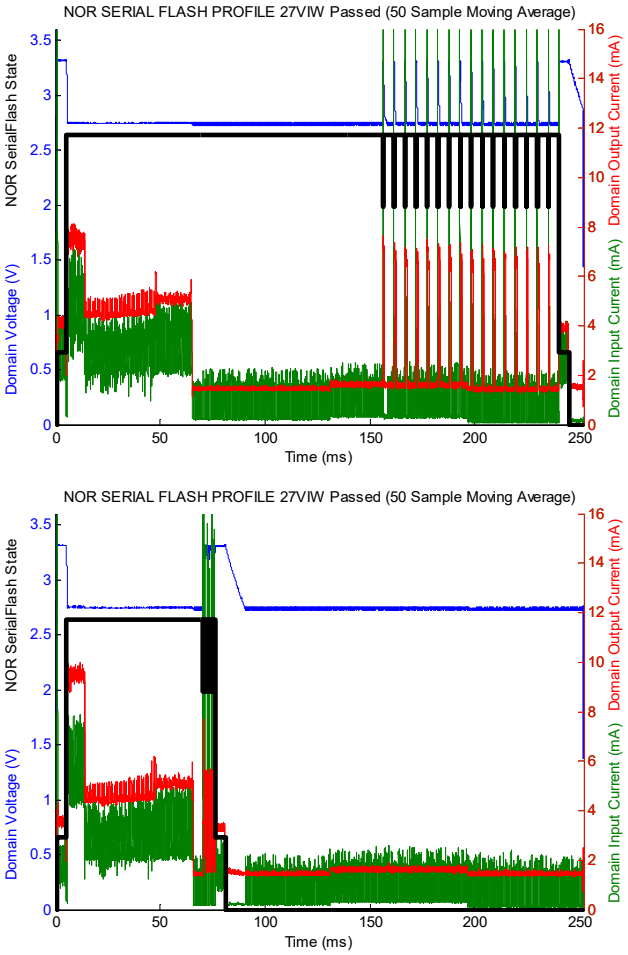

Figure 11: NOR Serial Flash Write-Cycle using IODVS and PACER-T

that the sub-sector erase completed approximately 65ms after it begins. Page writes are specified for a worst-case completion time of 10ms but through the application of PACER-T, they complete much faster as shown in TABLE II. The wait figure is the total amount of time spent waiting for the erase and the aggregate amount of time for each page write. The PACER-T algorithm delivered a 70% decrease in wait latency which yielded a 38.9% decrease in overall energy consumption. The worst-case manufacturer specification appears to be very pessimistic, although may be appropriate across both process and temperature variables.

TABLE II. M25PX16 NOR SERIAL FLASH PACER RESULTS

Stage	Latency Results (ms)				
	Control	*PACER-T*	*Diff.*	*PACER+IODVS*	*Diff.*
Wait	231.57	69.47	70.0%	66.92	71.1%
All	243.87	82.45	66.2%	80.26	67.1%
	Energy Results (uJ)				
Wait	2138.3	1212.0	43.3%	1029.52	51.9%
All	2277.0	1392.0	38.9%	1158.26	49.1%

C. *Microchip SST26VF016B Serial NAND Flash*

The SST26 serial flash module uses NAND-like control logic to provide higher capacity and lower latency than the NOR serial flash. However, the device sacrifices the random-access timing benefit of NOR flash. The serial flash module must therefore read an entire page of flash into a local buffer before providing read data to the host. This can result in non-deterministic read and write access times.

Despite the core logic differing from the M25PX16, PACER-T still performed the best. Application yielded a 66.6% decrease in aggregate wait times and a 17.8% decrease in energy consumption due to waiting as shown in TABLE III.

TABLE III. SST26VF016B NAND SERIAL FLASH PACER RESULTS

Stage	Latency Results (ms)				
	Control	***PACER-T***	***Diff.***	***PACER+IODVS***	***Diff.***
Wait	57.61	19.26	66.6%	19.27	66.6%
All	71.28	32.94	53.8%	32.95	53.8%
	Energy Results (uJ)				
Wait	1053.0	806.2	23.8%	584.87	44.5%
All	1247.9	997.26	17.8%	801.95	35.7%

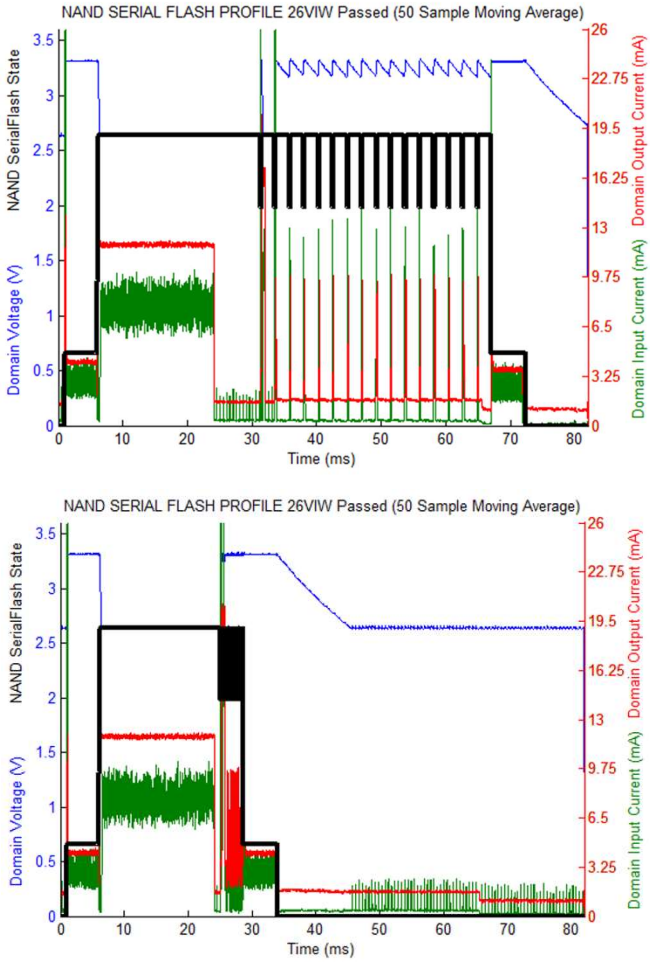

Figure 12: NAND Serial Flash Write Cycle Using IODVS and PACER-T

D. *An assortment of Micro-SD Memory Cards*

Onboard caches and memory management units cause the write operation of Micro-SD cards to have non-deterministic timing. In this case, PACER-C is the only algorithm that can reliably detect when the operation is finished. As with all memory tests, writes were performed with random data to random addresses throughout the memory space and so the cache performance is thoroughly exercised.

Figure 13 shows the massive power and latency difference between a cache miss and a cache hit. The cache miss has an overall response time of about 150ms, while the cache hit has a response time of about 1ms. It is important to recall that PACER-C is effective in this situation because it samples current on the domain at 1us intervals and makes decisions with a 50 sample moving average.

The SD-Card protocol is polling-based, as shown in the state transition diagram of Figure 14. There can be significant power-up and initialization delays. This prevents traditional dynamic power management (disabling the peripheral when not in use) techniques from being effective. Although constant polling is a good way to minimize latency, it results in extreme power increases. The analysis uses worst-case execution time (WCET) as the control group.

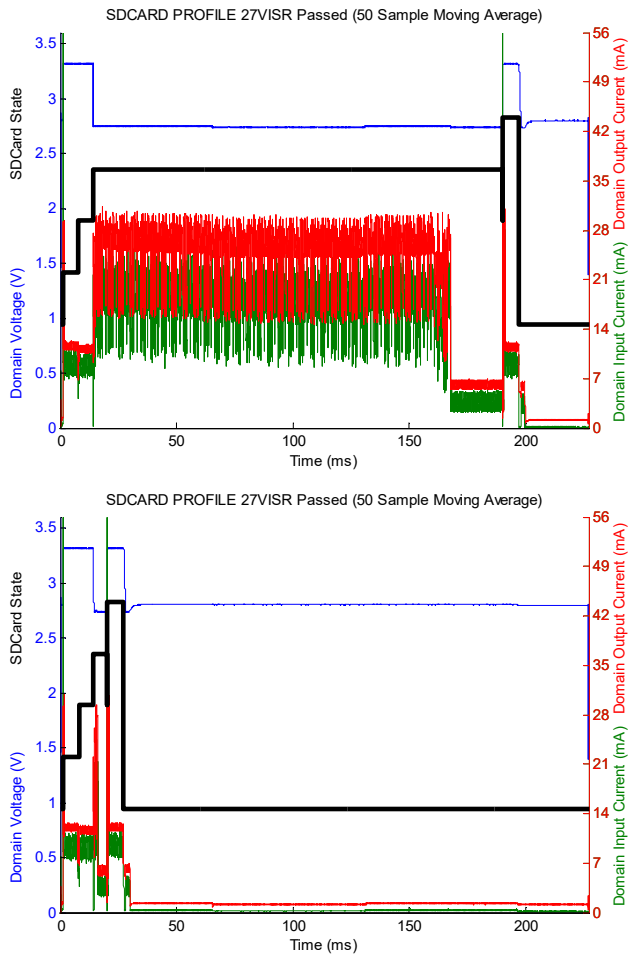

Figure 13: A Micro-SD Card Cache Miss and a Cache Hit

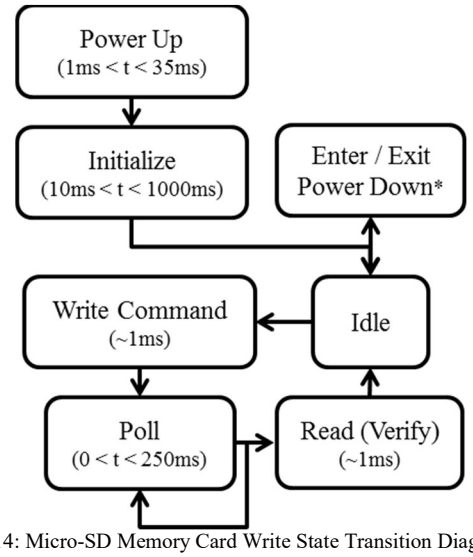

Figure 14: Micro-SD Memory Card Write State Transition Diagram

The WCET for each Micro-SD Card was determined to be the longest-observed operation duration for the measurement set under consideration. This WCET typically corresponds with a cache miss on the device.

Figure 15 helps to describe the performance differences shown in TABLE IV. The control delay is set to the WCET delay for each characterization, PACER-C allows the host to react to those operations deviating considerably from the control. Therefore, the Sandisk and Lexar cards benefitted considerably because they exhibit a bimodal timing distribution. The Swissbit card benefits decisively because of the mostly normal timing distribution. The Kingston card does not benefit as much because write timing exhibits a very low standard deviation. To present complete timing effects, a thorough latency analysis would need to be done on each device. Only energy results are presented here, but they are correlated with overall latency decreases.

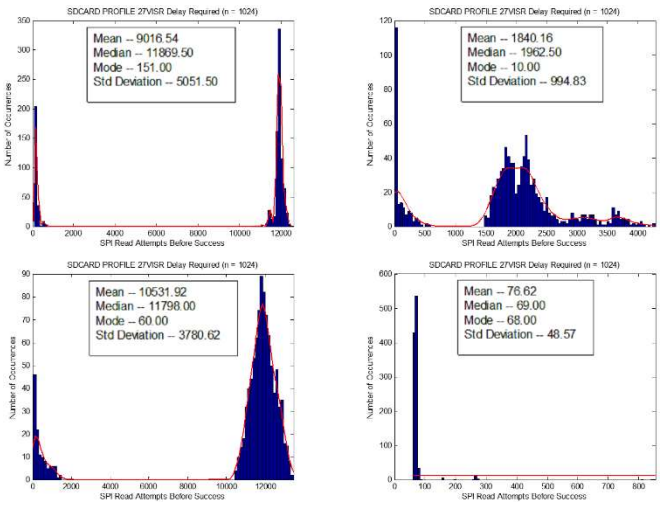

Figure 15: Timing Performance among Tested SD-Cards

TABLE IV. MICRO-SD CARD PACER RESULTS

Stage	Energy Results (uJ)				
	Control	*PACER-C*	*Diff.*	*PACER+IODVS*	*Diff.*
Sandisk	17066	15198	10.9%	11848	30.6%
Lexar	22707	21428	5.6%	16977	25.24
Swissbit	2763	914	66.9%	554	80.0%
Kingston	942	933	0.9%	897	4.8%

The Sandisk, Lexar, Swissbit and Kingston cards were manufactured in 2007, 2008, 2014 and 2015, respectively. It is reasonable to infer that the technology within the card advanced considerably within that time and will continue to do so in the future. Another key-takeaway from the statistical distribution of write operations on random memory locations is that the associativity appears to be different between the cards. The Lexar and Sandisk cards appear to be set-associative while the Swissbit appears to be fully associative. The standard deviation and response time was so low for the Kingston card that it cannot be determined what type of cache structure is used internally.

E. Honeywell HIH-6130 Temperature / Humidity Sensor

The Honeywell HIH-6130 communicates via the I^2C bus. Therefore, the device must only match the host voltage during the host-read states which is convenient for IODVS application. As shown in the state transition diagram of Figure 16, the host requests the sensor to take a measurement and then waits the manufacturer-specified 45ms for the measurement to complete. Finally, the host retrieves the completed measurement. PACER-E demonstrated the best performance among the algorithms, perhaps because of the capacitive nature of the peripheral ADC. The effects are shown in Figure 17 and the numeric results are presented in TABLE V.

The PACER-T algorithm also produced impressive results with a wait latency of 31.66ms and wait energy of 254.14uJ. Compared with PACER-E, the result corresponds with a slightly increased latency of 0.5% and slightly increased energy consumption of 4.3%. For some applications, the simplicity of the PACER-T implementation may be preferable when compared to the best performing PACER-E algorithm.

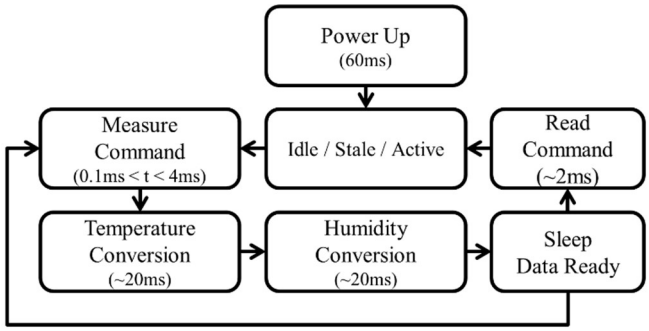

Figure 16: HIH-6130 State Transition Diagram

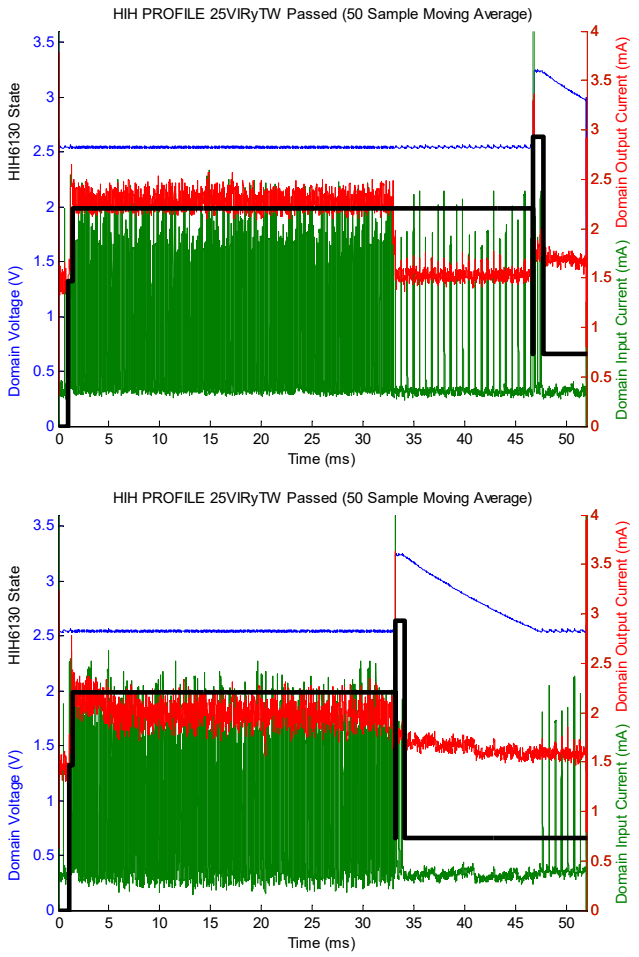

Figure 17: HIH-6130 Measurement Cycle Using IODVS and PACER-E

TABLE V. HONEYWELL HIH-6130 PACER RESULTS

Stage	Latency Results (ms)				
	Control	*PACER-E*	*Diff.*	*PACER+IODVS*	*Diff.*
Wait	45.27	31.45	66.6%	19.27	66.6%
All	45.99	32.17	53.8%	32.95	53.8%
	Energy Results (uJ)				
Wait	325.95	240.29	26.3%	169.62	48.0%
All	330.50	245.39	25.8%	173.89	47.4%

VI. SYSTEM COST / BENEFIT ESTIMATION

Application of the PACER algorithms results in moderate to extreme decreases in peripheral response time and power consumption. These algorithms do require that the application MCU perform some computation before and during the peripheral operation. All of the algorithms require a high-precision (hardware) timer and PACER-E/C require active analog to digital conversion (ADC).

Equation (1) of the introduction demonstrates the difficulty in accurately estimating the power consumption of a polling scenario. Fortunately, keeping the control group as the WCET does allow for an estimate of overall impact.

TABLE VI. ALGORITHM RESOURCE REQUIREMENTS

	CPU Usage	Timer	ADC	DMA,
Waiting	0%	Active	Disabled	Inactive
PACER-T	<1%	Active	Disabled	Inactive
PACER-E	5%	Active	Active	Active
PACER-C	3%	Active	Active	Active

TABLE VII. MODERN MICROCONTROLLER CHARACTERISTICS

MCU	STM32F730	STM32F401	Apollo2
Frequency (MHz)	216	84	48
CPU Current (mA)	90	11.5	0.432
ADC (mA)	1.6	1.6	0.541
DMA (mA)	3.1	1.7	N/A

The resource requirements for implementing both the control group and the various PACER algorithms are shown in TABLE VI. Performance and power estimates of required peripherals among a few modern MCUs are shown in TABLE VII. The most important aspect of this data is the massive increase in CPU energy expenditure in order to enable high-speed analog sampling.

The PACER-E and PACER-C tests were completed with a 1us sampling period. The ADC results were transferred via DMA to a buffer where they were arithmetically manipulated in order to achieve the current and energy results as required by the algorithm. The Apollo2 MCU does not have a DMA unit onboard and therefore the sampling period would need to be reduced and CPU energy expenditure would rise. Fortunately, due to the sub-threshold switching nature of that MCU, the expected utilization increase would not result in substantially higher overall energy expenditure.

Primarily, these tables indicate that PACER-T is generally the best algorithm to implement from a system-level evaluation. For non-deterministic peripherals such as the Micro-SD cards, only the PACER-C algorithm was able to detect early completions. In this case, a balance needs to be achieved between the energy costs of sample rate and peripheral latency/power reductions.

VII. EXTENDED ALGORITHM RESULTS

Previous sections presented only the result of the best algorithm. Complete results are presented in this section for each peripheral against each and every algorithm.

There is an exception for the Micro-SD cards because those devices are non-deterministic and therefore the results presented in that section are already comprehensive.

Note that the abridged results of the previous sections have had the final idle-phase of each test removed because that additional power consumption and latency is an artifact of the test fixture rather than real-world application. That aspect is presented here for completeness.

TABLE VIII. EEPROM OPERATION ENERGY

State	Control	IODVS	PACER-T	PACER-T + IODVS	PACER-E	PACER-E + IODVS	PACER-C	PACER-C + IODVS
Idle	6.03	2.47	6.04	2.52	6.06	2.47	6.13	2.48
Writing	2.71	2.55	2.73	2.51	2.69	2.50	2.80	2.58
Waiting	46.84	33.85	37.89	27.85	33.59	25.06	35.47	26.24
Read	3.50	3.48	3.29	2.04	3.51	3.49	3.55	3.21
Idle	5.89	5.76	15.30	13.42	14.97	14.16	14.50	14.03
Active Total	*64.97*	*48.11*	*65.24*	*48.34*	*60.82*	*47.69*	*62.44*	*48.54*
Delta	**0.00%**	**-24.83%**	**-17.24%**	**-38.93%**	**-25.00%**	**-41.46%**	**-21.17%**	**-39.62%**

TABLE IX. EEPROM OPERATION LATENCY

State	Control	IODVS	PACER-T	PACER-T + IODVS	PACER-E	PACER-E + IODVS	PACER-C	PACER-C + IODVS
Idle	1.025	1.025	1.026	1.025	1.026	1.025	1.038	1.038
Writing	0.429	0.43	0.429	0.429	0.429	0.43	0.429	0.429
Waiting	5.045	5.044	3.508	3.507	3.54	3.508	3.603	3.653
Read	0.506	0.508	0.506	0.507	0.506	0.506	0.507	0.508
Idle	0.994	0.992	2.53	2.531	2.498	2.53	2.422	2.371
Active Total	*5.98*	*5.982*	*4.443*	*4.443*	*4.475*	*4.444*	*4.539*	*4.59*
Delta	**0.00%**	**0.03%**	**-25.70%**	**-25.70%**	**-25.17%**	**-25.69%**	**-24.10%**	**-23.24%**

TABLE X. NOR SERIAL FLASH OPERATION ENERGY

State	Control	IODVS	PACER-T	PACER-T + IODVS	PACER-E	PACER-E + IODVS	PACER-C	PACER-C + IODVS
Idle	6.04	4.66	6.05	4.61	6.07	4.65	6.15	4.63
Reading	49.99	50.08	58.08	50.21	50.13	49.89	49.87	50.14
Erase	0.95	0.96	0.91	0.94	0.96	0.97	0.99	0.95
Total Write	39.17	36.31	63.05	41.26	37.68	20.40	37.36	17.81
Total Wait	2138.32	1713.89	1211.99	1029.52	1501.73	1178.32	1319.34	1040.74
Reading	48.60	48.78	51.91	31.72	48.59	33.76	48.47	42.02
Idle	37.05	33.13	929.24	711.59	689.63	576.83	859.97	657.46
Active Total	*2277.02*	*1854.68*	*1391.98*	*1158.26*	*1645.17*	*1287.98*	*1462.18*	*1156.28*
Delta	**0.00%**	**18.55%**	**38.87%**	**49.13%**	**27.75%**	**43.44%**	**35.79%**	**49.22%**

TABLE XI. NOR SERIAL FLASH OPERATION LATENCY

State	Control	IODVS	PACER-T	PACER-T + IODVS	PACER-E	PACER-E + IODVS	PACER-C	PACER-C + IODVS
Idle	1.04	1.04	1.04	1.04	1.04	1.04	1.05	1.05
Reading	4.27	4.27	4.27	4.27	4.27	4.27	4.27	4.27
Erase	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08
Total Write	3.31	3.32	3.31	3.32	3.31	3.32	3.31	3.32
Total Wait	231.57	231.57	69.47	66.92	104.81	120.17	80.15	79.06
Reading	4.64	4.64	4.27	4.63	4.64	4.73	4.64	4.64
Idle	7.09	7.08	169.55	171.74	133.86	118.38	158.50	159.57
Active Total	*243.87*	*244.92*	*82.45*	*80.26*	*118.14*	*133.62*	*93.50*	*92.43*
Delta	**0.00%**	**-0.43%**	**66.19%**	**67.09%**	**51.55%**	**45.21%**	**61.66%**	**62.10%**

TABLE XII. NAND SERIAL FLASH OPERATION ENERGY

State	Control	IODVS	PACER-T	PACER-T + IODVS	PACER-E	PACER-E + IODVS	PACER-C	PACER-C + IODVS
Idle	5.62	4.12	5.54	4.07	5.58	4.08	5.64	4.14
Reading	71.49	71.57	72.58	71.52	71.45	72.10	71.54	71.39
Erase	1.58	1.59	1.53	1.58	1.55	1.49	1.55	1.52
Total Write	51.88	48.14	75.40	70.70	73.01	70.19	51.63	44.66
Total Wait	1052.98	806.15	802.63	584.87	817.59	596.84	887.28	670.32
Reading	69.97	69.81	73.11	73.27	72.90	73.11	72.75	72.85
Idle	52.31	44.66	249.37	187.25	234.28	158.54	149.37	133.95
Active Total	*1247.90*	*997.26*	*1025.25*	*801.95*	*1036.50*	*813.72*	*1084.76*	*860.73*
Delta	**0.00%**	**-20.08%**	**-17.84%**	**-35.74%**	**-16.94%**	**-34.79%**	**-13.07%**	**-31.03%**

TABLE XIII. NAND SERIAL FLASH OPERATION LATENCY

State	Control	IODVS	PACER-T	PACER-T + IODVS	PACER-E	PACER-E + IODVS	PACER-C	PACER-C + IODVS
Idle	1.04	1.04	1.04	1.04	1.04	1.04	1.05	1.05
Reading	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
Erase	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.08
Total Write	3.31	3.32	3.31	3.31	3.31	3.32	3.31	3.31
Total Wait	57.61	57.62	19.26	19.27	25.92	25.92	36.19	36.20
Reading	5.28	5.28	5.29	5.29	5.28	5.28	5.28	5.29
Idle	9.68	9.67	48.02	48.01	41.37	41.36	31.09	31.07
Active Total	*71.28*	*71.29*	*32.94*	*32.95*	*39.59*	*39.60*	*49.86*	*49.87*
Delta	**0.00%**	**0.01%**	**-53.79%**	**-53.78%**	**-44.46%**	**-44.45%**	**-30.05%**	**-30.03%**

TABLE XIV. HIH-6130 OPERATION ENERGY

State	Control	IODVS	PACER-T	PACER-T + IODVS	PACER-E	PACER-E + IODVS	PACER-C	PACER-C + IODVS
Idle	5.19	3.60	5.28	3.64	5.24	3.62	5.33	3.67
Writing	1.75	0.98	1.76	0.93	1.73	0.98	1.76	1.00
Waiting	325.95	231.17	254.14	120.39	240.29	169.62	223.65	159.00
Idle	0.08	0.05	0.09	0.05	0.12	0.10	0.09	0.05
Reading	2.80	2.64	2.98	3.10	3.37	3.30	2.95	2.92
Idle	28.04	26.18	105.49	83.50	106.34	84.62	105.31	83.60
Active Total	*330.50*	*234.79*	*258.88*	*124.41*	*245.39*	*173.89*	*228.36*	*162.91*
Delta	**0.00%**	**-28.96%**	**-21.67%**	**-62.36%**	**-25.75%**	**-47.39%**	**-30.91%**	**-50.71%**

TABLE XV. HIH-6130 OPERATION LATENCY

State	Control	IODVS	PACER-T	PACER-T + IODVS	PACER-E	PACER-E + IODVS	PACER-C	PACER-C + IODVS
Idle	1.01	1.01	1.01	1.01	1.01	1.01	1.03	1.03
Writing	0.23	0.44	0.23	0.44	0.23	0.44	0.23	0.44
Waiting	45.27	45.27	31.66	31.44	31.45	31.41	31.70	31.60
Idle	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
Reading	0.49	0.95	0.49	0.95	0.49	0.95	0.49	0.95
Idle	4.98	4.32	18.59	18.15	18.80	18.18	18.54	17.98
Active Total	*45.99*	*46.65*	*32.38*	*32.82*	*32.17*	*32.80*	*32.42*	*32.98*
Delta	**0.00%**	**1.44%**	**-29.59%**	**-28.63%**	**-30.05%**	**-28.69%**	**-29.50%**	**-28.28%**

VIII. CONCLUSIONS

Applying the PACER suite of algorithms to a variety of common embedded peripherals resulted in significant reductions to both latency and energy consumption. The PACER-T algorithm performed best against time-bound operations and was very competitive in energy-bound operations. For non-deterministic operations, the PACER-C algorithm performed well. When measured against a median baseline, the algorithm performed even better as the operational latency increased in randomness.

The PACER-T and PACER-E algorithms use successive approximation and the PACER-C algorithm uses a return-to-idle measurement to determine activity completion. It is likely that the performance of both methods could be enhanced further through the application of more complex algorithms. PACER-T could be applied to memory operations with a bimodal delay distribution by first testing for a cache hit and then delaying for a determined cache-miss time. Likewise, the PACER-C algorithm could be modified online so as to identify the current waveform features corresponding varying latencies, thus allowing the MCU to sleep longer.

The designer must be judicious in selecting the appropriate algorithm in order to minimize total energy expenditure. For most cases, the PACER-T algorithm provides sufficient performance improvements without the added computational and peripheral energy costs. For devices with a non-deterministic operation execution time, then PACER-C is the only option. However, the designer can still adjust the sample rate so as to perhaps eliminate the need for DMA and reduce the duty cycle of the ADC. Reducing the sample rate will indeed increase the latency of the algorithm, but the overall implications and design tradeoffs should be handled at the system level.

Combining the best aspects of PACER-T and PACER-C would be an interesting topic for further research. Predicting that a memory access will result in either a cache hit with minimal latency. Failing that, the algorithm could assume a cache miss and delay for a certain amount of time before resuming measurements to determine if current consumption has returned to the idle state. This method would strive to reduce the duty cycle of all three MCU energy consumers, the CPU, the ADC peripheral and the DMA peripheral. The algorithm could also eliminate the need to sample the voltage channel because only the current channel indicates activity.

As the cost of computation in embedded systems continues to decrease, it is natural to devote more computational resources to minimizing system-wide energy consumption and latency. The PACER suite of algorithms use minimal computational resources and are shown to decrease latency by up to 67% and device energy consumption by up to 80% when compared to the naïve worst-case estimate.

REFERENCES

[1] D. Moore and A. Dean, "PACER," in *The Thirteenth International Conference on Systems (ICONS 2018) IARIA*, Apr. 2018, pp 38-45, ISBN: 978-1-61208-626-2.

[2] B. Brock and K. Rajamani, "Dynamic power management for embedded systems [SOC design]," in *Proc. IEEE International [Systems-on-Chip] SOC Conference 2003*, IEEE Press, Sep. 2003, ISBN: 0-7803-8182-3.

[3] C. Kumar, M. Sindhwani and T. Srikanthan, "Profile-based technique for Dynamic Power Management in embedded systems," in *International Conference on Electronic Design (ICED 2008)*, IEEE Press, Dec. 2008, ISBN: 978-1-4244-2315-6.

[4] W. Dargie, "Dynamic Power Management in Wireless Sensor Networks: State-of-the-Art," *IEEE Sensors Journal,* vol. 12, no. 5, pp. 1518 - 1528, 2012, ISSN: 1558-1748.

[5] D. Moore and A. Dean, "Intra-Operation Dynamic Voltage Scaling," in *2015 IEEE 3rd International Conference on Cyber-Physical Systems, Networks, and Applications*, Hong Kong, IEEE Press, Aug. 2015, ISBN: 978-1-4673-7785-0.

[6] D. Lee, Y. Kim, G. Pekhimenko, S. Khan, V. Seshadri, K. Chang and O. Mutlu, "Adaptive-latency DRAM: Optimizing DRAM timing for the common-case," in *IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)*, IEEE Press, Feb. 2015, ISBN: 978-1-4799-8930-0.

[7] S. Sadeghi-Kohan, M. Kamal, J. McNeil, P. Prinetto and Z. Navabi, "Online self adjusting progressive age monitoring of timing variations," in *10th International Conference on Design & Technology of Integrated Systems in Nanoscale Era (DTIS)*, IEEE Press, Apr. 2015, ISBN: 978-1-4799-1999-4.

[8] D. S. S. Etter, "Adaptive Estimatation of Time Delays in Sampled Data Systems," in *IEEE Transactions on Acoustics Speech and Signal Processing (Volume: 29, Issue: 3)*, IEEE Press, Jun. 1981, ISSN: 0096-3518.

[9] S. G. P. A. Z. E. Tarasov V, "Efficient I/O Scheduling with Accurately Estimated Disk Drive Latencies," in *The Proceedings of OSPERT 2012*, 2012.

[10] H. Macicior, M. Oyarbide, O. Miguel, I. Cantero, J. Canales and A. Etxeberria, "Iterative capacity estimation of LiFePO4 cell over the lifecycle based on SoC estimation correction," in *2013 World Electric Vehicle Symposium and Exhibition (EVS27)*, IEEE Press, Nov. 2013, ISBN: 978-1-4799-3832-2.

[11] H. Mahanta, A. Azad and A. Khan, "Power analysis attack: A vulnerability to smart card security," in *International Conference on Signal Processing And Communication Engineering Systems (SPACES)*, IEEE Press, Mar. 2015, ISBN: 978-1-4799-6109-2.

[12] M. Petrvalsky, M. Drutarovsky and M. Varchola, "Differential power analysis attack on ARM based AES implementation without explicit synchronization," in *2014 24th International Conference Radioelektronika*, IEEE Press, Jun. 2014, ISBN: 978-1-4799-3715-8.