Energy-Architecture Tuning for ECC-based RFID tags

Deepak Mane and Patrick Schaumont

Secure Embedded Systems
Center for Embedded Systems for Critical Applications
Bradley Department of ECE
Virginia Tech, Blacksburg, VA 24061, USA
{mdeepak,schaum}@vt.edu

Abstract. The implementation of Elliptic Curve Cryptography (ECC) on small microcontrollers is challenging. Past research has therefore emphasized performance optimization: pick a target architecture, and minimize the cycle count and footprint of the ECC software. This paper addresses a different aspect of resource-constrained ECC implementation: given the application profile, identify the most suitable architecture parameters. At the highest level, an application profile for ECC-based RFID tags is defined by the required security level, signature generation latency and the available energy/power budget. The target architecture parameters of interest include core-voltage, core-frequency, and/or the need for hardware acceleration. The paper brings two contributions to this complex design space exploration problem. First, we introduce a prototype setup for the precise energy measurement of a microcontroller-based ECC implementation. Second, we present a methodology to derive and optimize the architecture parameters starting from the application requirements. We demonstrate our methodology on a MSP430F5438A microcontroller, and present the energy/architecture design space for 80-bit and 128-bit security-levels, for prime field curves `secp160r1` and `nistp256`.

Keywords: Public Key Cryptography, Elliptic Curves, RFID, Energy Harvesting, Throughput, Digital signatures.

1 Introduction

There are appealing advantages to using public-key cryptography (PKC) in RFID applications that require authentication. Indeed, PKC-based authentication significantly simplifies the distribution of cryptographic keys, resulting in a more scalable solution. The challenge of using ECC in the RFID environment is how to deal with the high computational cost associated with ECC algorithms relative to the capabilities of the RFID platform. Fortunately, this question has been reasonably well solved, and Table 1 shows some of the more recent achievements. It is fair to state that a security level of 80 bit (ECC curves of at least 160 bit) is within reach, with sub-second latency, in small footprint applications

Table 1: Recent ECC implementations for RFID

Ref	Field/Curve	Target Hardware or Software	Time (seconds)	Operation	Resource
[11]	$GF(2^{163})$	HW, 0.13μm 100Khz	0.244	Point Mult	12,506 GE
[9]	$GF(2^{163})$	HW, 0.18μm 106Khz	0.279	Point Mult	11,904 GE
[12]	`secp160r1`	SW, MSP430F1611 8MHz	8.54	ECDSA sign	13,520 bytes code
[21]	`secp160r1`	SW, MSP430F1611 8MHz	1.1	Point Mult	NA
[17]	`nistp192`	SW, MSP430F2131 6.7MHz	1.6	Point Mult	16,060 bytes code
[7]	`secp160r1`	SW, MSP430F5529 25MHz	0.068	ECDSA sign	24,000 bytes code

(i.e. 15 KGate, 100 KHz hardware implementations; or 16 bit, 8-MHz software implementations).

To achieve these results, the authors of the designs in Table 1 need to use advanced algorithmic transformations and optimizations. Furthermore, they also make use of technology-specific features. Software implementations assume certain amounts of flash and RAM memory, or microcontroller features such as a hardware multiplier. Hardware implementations assume a target cell library of specific performance and feature size.

The assumption of a specific target architecture at the start of the design is typical for contemporary digital design methods. The designs in Table 1 are no exceptions. On the other hand, it is much harder for ECC designers to make clear commitments to application constraints such as the available energy budget and the required authentication latency. This is understandable: EDA tools do a poor job at estimating system performance and energy consumption. It is easier to optimize an implementation on a given target and then evaluate its characteristics on a prototype or using low-level simulation.

In this paper, we discuss the ECC RFID design problem by considering how to meet design requirements, rather than how to obtain the fastest ECC point multiplication. The main motivator for this is that the power source in the RFID environment is truly unique, and that this design problem deserves *a more holistic approach which considers energy source as well as energy consumer.*

Indeed, depending on the power source, RFID designs are either energy-constrained or else power constrained. They also have to optimize application throughput (the time taken to complete a single signature) with respect to the available energy budget. We note that the requirements on energy and power depend on the type of RFID.

- Active RFID are powered from a battery source, and they have to minimize the energy consumed per signature as this will maximize the battery lifetime.
- Passive RFID are powered through an RF source, and they have to minimize the time required per signature while matching the available power budget.
- Passive RFID, powered through an energy harvesting mechanism that includes an energy store, have to minimize the energy used per signature

as well, since this allows uninterrupted RFID operation. Furthermore, the energy needed for the desired application throughput has to match the average energy influx in the harvester.

The question addressed in this paper is: how can we select architecture parameters such that we meet these design requirements, including energy budget and application throughput? We provide an empirical answer to this question by presenting the energy/latency characteristics of an RFID doing ECDSA key generation, signature generation and signature verification. We present our results for a microcontroller target, a MSP430F5438A from Texas Instruments. We evaluate the energy characteristics of signatures at 80-bit and 128-bit security level. We examine multiple architecture configurations: multiple frequencies, multiple core voltages, and with/without use of the MSP430's hardware multiplier. For each of these configurations, we carefully measure the required energy by tracking the MSP430 core current at high speed. The resulting curves then allow us to determine, for a given security level and energy budget, the most appropriate core frequency and voltage level.

In a nutshell, our results are as follows. Increasing the MSP430 core frequency always reduces the energy consumed per signature. This is because energy consumed by leakage (i.e. static power dissipation) becomes proportionally less important as the runtime of the application decreases. Hence, in a given energy harvesting method, it is always better to wait as long as possible before initiating ECC computations and, once sufficient energy is available, complete them as quickly as possible at the highest possible operating frequency. A second observation is that the impact of security level on energy budget is significant. For an MSP430 without a hardware multiplier, our prototype needs roughly six times as much energy per signature at the 128 bit security level compared to the 80 bit security level. When a hardware multiplier can be used, the difference is roughly two times. A third observation is that architecture specialization matters. Under constant security level, a hardware multiplier reduces the energy consumption by almost 8 times. Voltage scaling results in an additional gain factor of 2 in energy.

The remainder of the paper is organized as follows. In the next section, we describe the background related to power and energy in the digital electronics. In Section 3, we review the target ECC design measured using our method. Next, we introduce the target platform for our experiments. Section 5 describes a setup that can be used for precise energy measurement of RFID applications. The resulting energy/throughput curves are presented in Section 6, and applied in a methodology in Section 7. Section 8 concludes the paper.

2 Background: Power and Energy in Digital Electronics

Power dissipation in modern digital electronics has two major components: static power dissipation defined by static leakage current, and dynamic power consumption, defined by circuit activity. The static power dissipation depends, in first

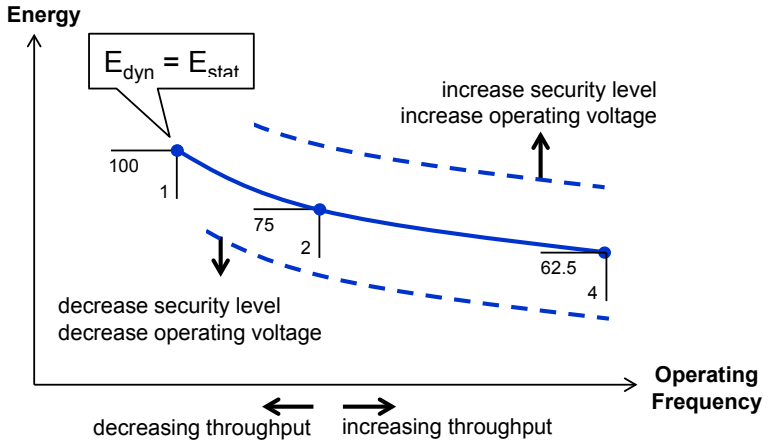

Fig. 1: Expected Energy dissipation as a function of frequency

order, on the operating voltage of the circuit and the size of the circuit. Dynamic power dissipation depends, in first order, on the operating frequency of the circuit, the size of the circuit, and the square of the operating voltage.

We analyze what happens to the energy dissipation for a fixed workload, such as signature verification, under varying operating conditions. In the following, K and C are technology constants, α is an application-dependent activity factor, T_{cycle} is the clock cycle period, f_{cycle} is the operating frequency, and n is the workload cycle budget. The energy dissipation per workload has a static and a dynamic component.

$$E_{dyn} = P_{dyn}.T_{alg} = \alpha.C.V^2.f_{cycle}.T_{cycle}.n \quad (1)$$

$$E_{stat} = P_{static}.T_{alg} = K.V.T_{cycle}.n \quad (2)$$

The total energy dissipation thus equals

$$E_{tot} = E_{dyn} + E_{static} = n.[\alpha.C.V^2 + K.V.T_{cycle}] \quad (3)$$

This formulation leads to the following assessment. If the clock frequency increases, the total energy per workload will decrease: the dynamic energy remains constant, while the static energy decreases. Furthermore, if the operating voltage decreases, the total energy per workload will decrease as well. Finally, if the security level of the design increases (from 80 bit to 128 bit, for example), the cycle budget n will increase, and the total energy per workload will increase as well. This analysis is captured by Figure 1. The leftmost point on the curve represents a design where the static and dynamic parts of the energy per workload are equal. As the operating frequency increases, the total energy decreases as well. We note that this figure is a theoretical model: it ignores overhead for clock generation, and for transitions between power modes.

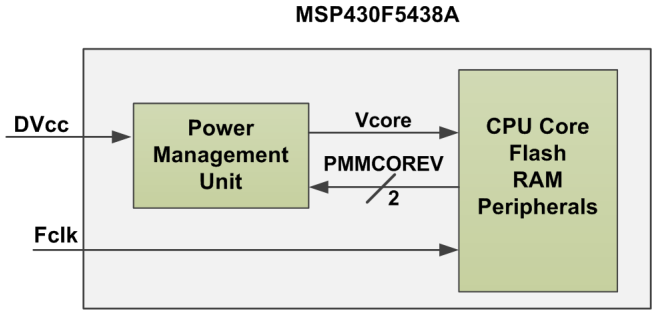

Fig. 2: Voltage and Frequency Scaling

3 Target Protocol: ECDSA in `secp160r1` and `nistp256`

The driving application for our measurements is ECDSA key generation, signature generation, and signature verification. Rather than developing our own implementation from scratch, we use the RELIC library with support for the MSP430 and 32-bit hardware multiplier [16]. We implement two prime-field curves, `secp160r1` and `nistp256`. The scalar multiplication is done with a left-to-right window-3 NAF multiplication, and using Jacobian Projective Coordinates. The field operations are basic Comba multiplication and squaring, with Montgomery reduction. SHA-1 is used for hashing and as a pseudorandom generator. We used two implementation variants for each of the curves: one which uses a 32-bit hardware multiplier (using RELIC's `msp-asm` backend), and a second one which emulates multiplication in software (using RELIC's `easy` backend).

Our standard testbench goes through ECDSA key generation, ECDSA signature generation of a fixed digest, and ECDSA signature verification. The execution time, as well as the energy, is measured for each of these steps separately. Our performance and energy numbers only cover the computations, and they don't include initialization overhead, or overhead from data communications.

4 Target Platform: MSP430F5438A

We use MSP430F5438A [1] as our prototyping platform. The MSP430F5438A is an ultra-low power Reduced Instruction Set Computer (RISC) from Texas Instruments, optimized for low-resource applications [2]. The architecture combines five different low power modes suitable for low power battery operation. The MSP430F5438A features a 16-bit CPU, 256KB flash, 16KB SRAM, up to 25 MHz CPU clock and 16 working registers with 12 available as general purpose registers. It also supports a 32 bit hardware multiplier.

Figure 2 shows the internal energy management architecture of the MSP430F5438A. In general, `VCore` supplies the CPU, memories (flash and RAM), and the digital modules, while `DVcc` supplies the I/Os and all analog modules. The internal

Table 2: Recommended PMMCOREV and DV_{CC} settings for Selected f_{sys}

f_{sys}(max) (MHz)	Minimum DVCC	Minimum PMMCOREV[1:0]
8	1.8V	00
12	2.0V	01
20	2.2V	10
25	2.4V	11

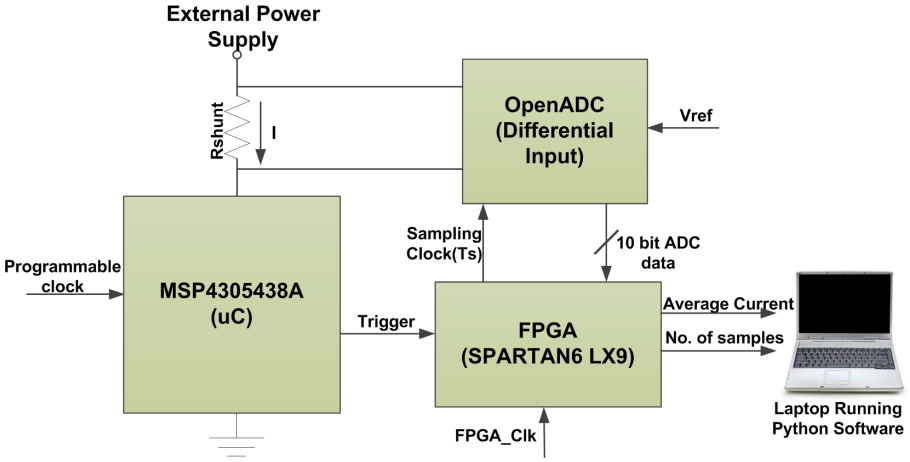

Fig. 3: Energy Measurement Setup Block Diagram

core voltage of the MSP430F5438A, `VCore`, needs to be adjusted as a function of the desired operating frequency. The `VCore` output is programmable in four steps, to provide only as much power as is needed for the speed that has been selected for the CPU. We configure `VCore` voltage by writing register bits `PMMCOREV[1:0]`. Table 2 shows recommended `PMMCOREV` settings and minimum external power supply voltage for different frequency ranges. We make use of these programmable `VCore` levels to optimize the energy efficiency of ECDSA on the MSP430.

5 High Resolution Energy Measurement

Figure 3 depicts the block diagram of energy measurement setup used in our experiments. The average current consumed by a microcontroller during a particular interval of time is measured by integrating the immediate current. The current is measured by means of the voltage drop over a shunt resistor on the microcontroller `Vcc` line. To sample the voltage drop, we use OpenADC [14] with a Spartan FPGA [3], in place of a traditional high-speed oscilloscope setup. OpenADC is a custom ADC board with a 10-bit A/D converter that supports

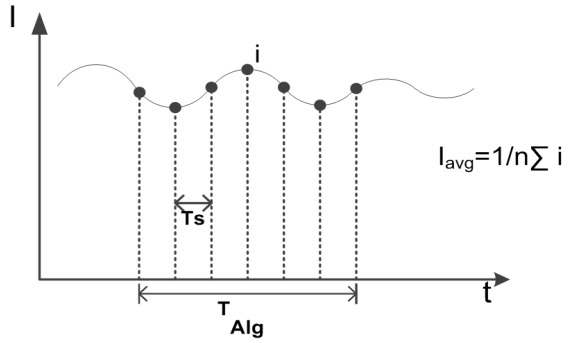

Fig. 4: Average current measurement

differential inputs and an adjustable reference voltage. The FPGA takes care of sample accumulation and sample counting.

The integration period is defined by means of trigger signals created by the microcontroller. This requires a simple instrumentation of the software application. The clock frequency of the A/D converter can be independently chosen of the microcontroller clock.

Procedure to measure average current: At a desired time, the microcontroller triggers the FPGA to start sampling OpenADC data (Figure 4). Once trigger is asserted, FPGA accumulates ADC samples until trigger line is re-asserted. The accumulator represents the average current consumed by a function being executed on a microcontroller. It also measures number of samples collected during trigger window to derive the execution time of that function. The FPGA design then provides average current and number of samples collected to a python script running on laptop (PC). A Python script uses this data and calculates the average energy consumed by an MSP430 function as follows.

$$\begin{aligned} \text{Energy} &= V_{cc}.I_{avg}.T_{Alg} \\ &= V_{cc}.\frac{Accm.V_{ref}}{2^n.N_s.R}.N_s.T_s \\ &= V_{cc}.\frac{Accm.V_{ref}}{2^n.R}.T_s \end{aligned} \tag{4}$$

$$\text{Cycle count} = N_s.T_s.F_{cpu} \tag{5}$$

where V_{cc} is supply voltage of the microcontroller, $Accm$ is FPGA accumulator value, V_{ref} is ADC reference voltage, N_s is number of samples collected during trigger window, T_s is sampling period, 2^n is resolution of ADC where n is 10 bit, R is a shunt resistor value and F_{cpu} is microcontroller frequency. Our Energy

Table 3: Cycle count of ECDSA operations on the MSP430F5438A

Operation	secp160r1		nistp256	
	w/o hardware multiplier	with hardware multiplier	w/o hardware multiplier	with hardware multiplier
Key Generation	19,343,970	1,796,499	124,427,469	5,225,820
Signing	19,141,737	2,372,103	124,942,312	6,408,792
Verification	57,621,281	57,48,345	346,290,644	15,584,937

Table 4: Code size of the implementation of ECDSA on the MSP430F5438A

	secp160r1		nistp256	
	w/o hardware multiplier	with hardware multiplier	w/o hardware multiplier	with hardware multiplier
Flash Bytes	27,134	28,168	28,138	32,234
RAM Bytes	1,074	1,074	1,542	1,542

formula assumes that the voltage supply at the microcontroller input is constant, in other words, that the voltage drop over the shunt resistor is negligible with respect to V_{cc}.

The above formula shows that the resolution of energy measurements increases with low reference voltage of ADC, high sampling frequency and with high resolution of ADC. We use sampling rate of 20MHz for operating frequencies below 15MHz and 30MHz for operating frequencies above 15MHz. In our experiments, ADC reference voltage is 0.5V and shunt resistor is 100Ω. This setup can be used to measure the energy consumption of any device provided that it has the facility to insert a resistor in series with power supply. Also, the device should be able to generate a trigger signal to activate the FPGA accumulator.

6 Results

In this section, we present experimental results of our energy measurements under different architecture configurations. We measure the energy consumption of ECDSA key generation, signature generation and signature verification for different operating voltages and frequency settings. We also measure the runtime for each operation in order to obtain the throughput. We use the `gcc 4.6.3` cross-compiler for MSP430 family of microcontrollers. Table 3 shows cycle counts for different ECDSA operations, and Table 4 shows the footprint of the implementations.

The graphs show the energy/throughput characteristics for signature generation.

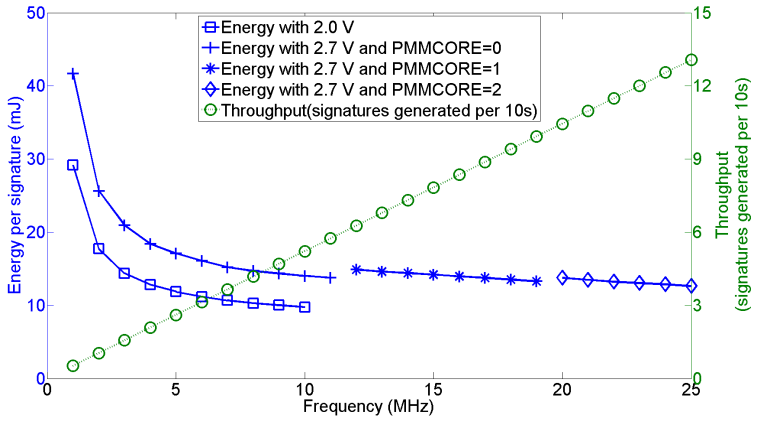

Fig. 5: Energy consumption for `secp160r1` without hardware multiplier

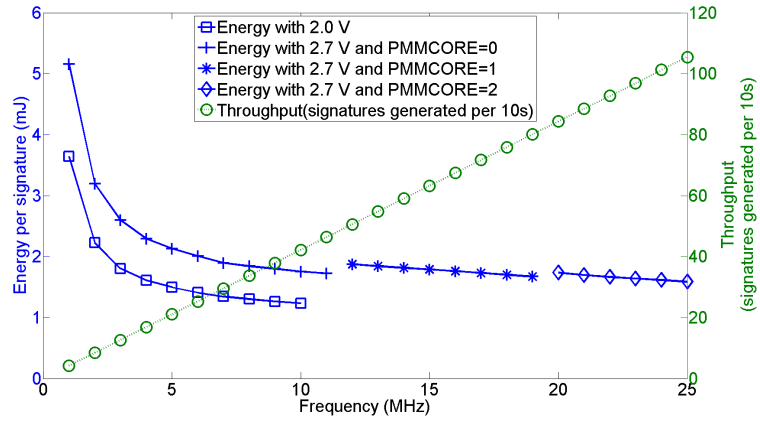

Fig. 6: Energy consumption for `secp160r1` with hardware multiplier

Figure 5 and 6 show energy consumption for 80-bit(`secp160r1` curve) security level, without and with hardware multiplier, respectively. We analyze the effect of different operating voltage on energy consumption. In our experiments, we found that if the microcontroller is operated at 2.0V instead of 2.7V, it saves almost 1.4 times energy consumption. Reducing operating voltage reduces dynamic power consumption because the circuit will have smaller voltage swings during switching. Also the static power consumption reduces because the leakage current reduces. Further, the use of the hardware multiplier results in almost 8 times energy reduction. This is because the hardware multiplier accelerates the signing operation almost by 8 times.

Figure 7 and 8 show energy consumption for 128-bit(`nistp256` curve) security level, without and with hardware multiplier respectively.

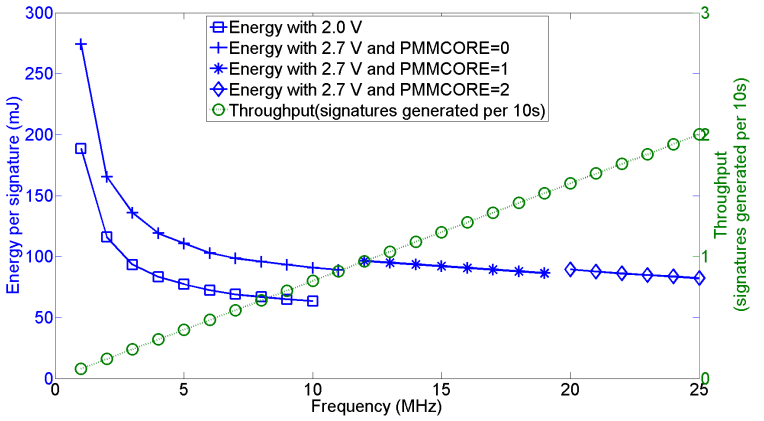

Fig. 7: Energy consumption for `nistp256` without hardware multiplier

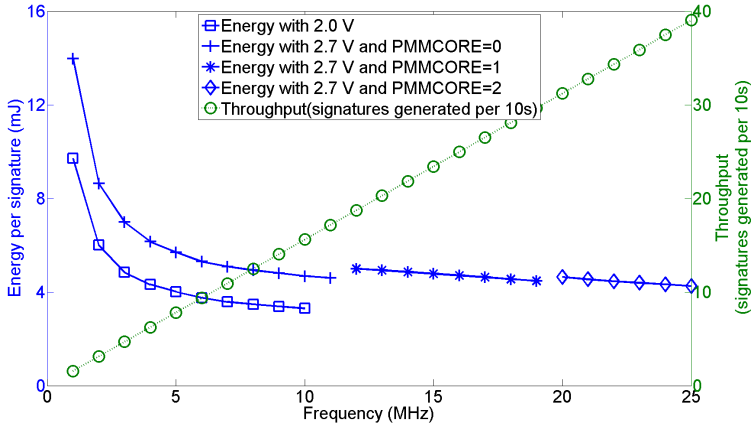

Fig. 8: Energy consumption for `nistp256` with hardware multiplier

The curves for operation at 2.7V is discontinuous at 12MHz and 20MHz. The discontinuities are caused by reprogramming of the power management system of the CPU.

7 Methodology for Architecture-Energy Tuning

Finally, we show how the energy measurement method can be applied to meet the design requirements. Figure 9 shows a typical energy harvesting setup where energy is harvested and stored. When sufficient energy is available, the application can execute. In the following examples, we demonstrate how our energy throughput curves can be used for system dimensioning. We consider two cases. In a first case, start from an energy constraint and derive the achievable perfor-

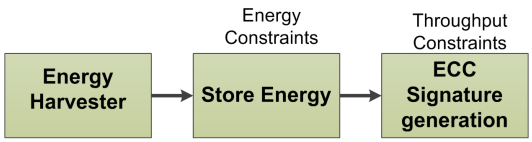

Fig. 9: Energy harvesting system

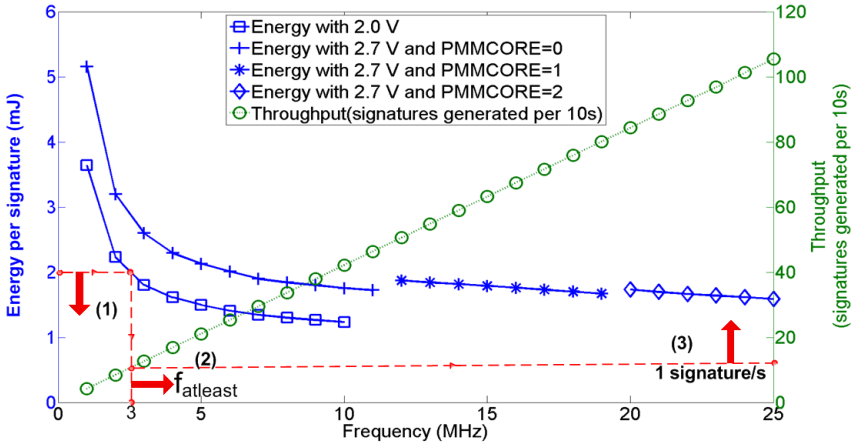

Fig. 10: Architecture tuning for energy constrained system

mance, expressed as the latency to complete one signature. This case is relevant when we use an energy store of a given dimension, and we would like to evaluate what system performance can be achieved. In the second case, we start from a desired application performance, and we derive the required energy (and thus the required energy store). Since there are multiple energy/throughput curves (at different security levels, and at different architecture configurations), we propose that this evaluation is done concurrently over all curves available, in order to analyze the design space. In the example discussed below, however, we will focus on a single security level and a microcontroller with a hardware multiplier.

Figure 10 shows the energy curve for the 80-bit security level. We assume an energy limit of 2 mJ per signature, and we assume a 2V operating level. This limit sets a *minimum* operating frequency for the microcontroller. The 2 mJ energy level requires a 3MHz operating frequency, which enables a signature to complete in a one second. If we increase the core voltage to 2.7V, the minimum operating frequency at 2mJ per signature will increase to 9MHz, and the signature will be done in 0.25 second.

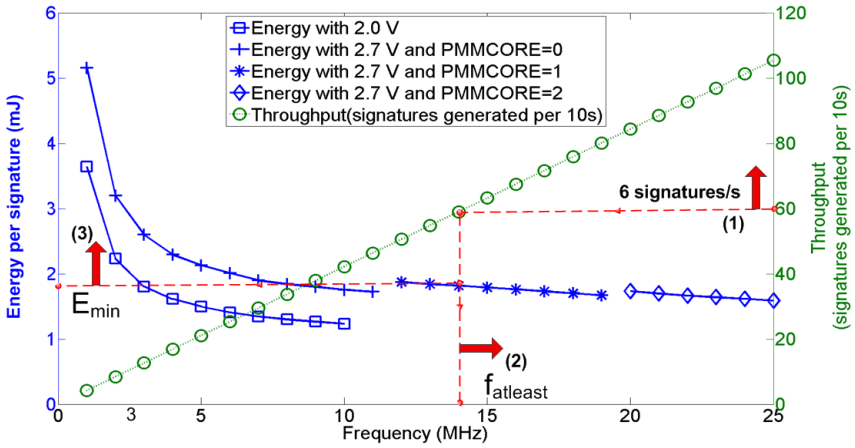

Fig. 11: Architecture tuning for throughput constrained system

A second example is to start from the required signature throughput. The example shown in Figure 11 needs to complete a signature in 1/6 of a second. We use our graphs to decide the operating frequency and to find required energy per signature. First, we note that the system needs to operate at least at 14MHz. At that frequency, only the 2.7V core voltage mode is available. Under this setting, the corresponding energy per signature is 1.9mJ.

8 Conclusion

We demonstrated the importance of energy-architecture tuning for RFID. The complex energy-provisioning environment of these applications requires a holistic approach that not only considers performance optimization, but also the energy and/or power needs. We analyzed and quantified, for two different security levels, the impact of several architecture optimization techniques including voltage scaling, frequency scaling, and the use of a hardware multiplier. Using the analysis presented in this paper, we can now investigate the design of a secure RFID with integrated energy harvesting.

References

1. Texas Instruments MSP430F5438A Mixed Signal Microcontroller, `http://www.ti.com/lit/ds/symlink/msp430f5438a.pdf`
2. Texas Instruments MSP430x5xx and MSP430x6xx Family User's Guide 2013, `http://www.ti.com/lit/ug/slau208m/slau208m.pdf`

3. Xilinx Spartan-6 FPGA LX9 MicroBoard, http://www.em.avnet.com/en-us/design/drc/Pages/Xilinx-Spartan-6-FPGA-LX9-MicroBoard.aspx
4. Buettner, M., Greenstein, B., Wetherall., D.: Dewdrop: An Energy-Aware Runtime for Computational RFID. In: Proceedings of USENIX Symposium on Networked Systems Design and Implementation (NSDI). Boston, MA, USA (2011)
5. Buettner, M., Greenstein, B., Wetherall, D.: Dewdrop: an energy-aware runtime for computational rfid. In: Proceedings of the 8th USENIX conference on Networked systems design and implementation. pp. 15–15. NSDI'11, USENIX Association, Berkeley, CA, USA (2011), http://dl.acm.org/citation.cfm?id=1972457.1972478
6. Chae, H.J., Yeager, D.J., Smith, J.R., Fu, K.: Maximalist cryptography and computation on the WISP UHF RFID tag. In: Proceedings of the Conference on RFID Security (July 2007), http://prisms.cs.umass.edu/~kevinfu/papers/chae-RFIDSec07.pdf
7. Gouvêa, C.P.L., López, J.: High speed implementation of authenticated encryption for the msp430x microcontroller. In: LATINCRYPT. pp. 288–304 (2012)
8. Hankerson, D., Menezes, A.J., Vanstone, S.: Guide to Elliptic Curve Cryptography. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2003)
9. Hein, D.M., Wolkerstorfer, J., Felber, N.: Ecc is ready for rfid - a proof in silicon. In: Selected Areas in Cryptography. pp. 401–413 (2008)
10. Hinterwälder, G., Paar, C., Burleson, W.P.: Privacy preserving payments on computational rfid devices with application in intelligent transportation systems. In: Proceedings of the 8th international conference on Radio Frequency Identification: security and privacy issues. pp. 109–122. RFIDSec'12, Springer-Verlag, Berlin, Heidelberg (2013), http://dx.doi.org/10.1007/978-3-642-36140-1_8
11. Lee, Y.K., Sakiyama, K., Batina, L., Verbauwhede, I.: Elliptic-curve-based security processor for rfid. Computers, IEEE Transactions on 57(11), 1514–1527 (2008)
12. Liu, A., Ning, P.: Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks. In: Proceedings of the 7th international conference on Information processing in sensor networks. pp. 245–256. IPSN '08, IEEE Computer Society, Washington, DC, USA (2008), http://dx.doi.org/10.1109/IPSN.2008.47
13. National Institute of Standards and Technology: FIPS 186-3: Digital Signature Standard (DSS) (2009), http://www.itl.nist.gov
14. O'Flynn, C.: OPENADC (2012), http://newae.com/tiki-index.php?page=OpenADC
15. O'Flynn, C.: Power analysis for cheapskates (2012), https://media.blackhat.com/ad-12/O%27Flynn/bh-ad-12-for-cheapskates-o%27flynn-WP.pdf
16. Oliveira, L.B., Aranha, D.F., Gouvêa, C.P.L., Scott, M., Câmara, D.F., López, J., Dahab, R.: Tinypbc: Pairings for authenticated identity-based non-interactive key distribution in sensor networks. Computer Communications 34(3), 485–493 (2011)
17. Pendl, C., Pelnar, M., Hutter, M.: Elliptic curve cryptography on the wisp uhf rfid tag. In: RFIDSec. pp. 32–47 (2011)
18. Raju, M.: T. Instruments White Paper: Energy Harvesting (2008)
19. Ransford, B., Clark, S., Salajegheh, M., Fu, K.: Getting things done on computational rfids with energy-aware checkpointing and voltage-aware scheduling. In: Proceedings of the 2008 conference on Power aware computing and systems. pp. 5–5. HotPower'08, USENIX Association, Berkeley, CA, USA (2008), http://dl.acm.org/citation.cfm?id=1855610.1855615

20. Wang, H., Li, Q.: Efficient implementation of public key cryptosystems on mote sensors (short paper). In: Proceedings of the 8th international conference on Information and Communications Security. pp. 519–528. ICICS'06, Springer-Verlag, Berlin, Heidelberg (2006), http://dx.doi.org/10.1007/11935308_37
21. Wenger, E., Werner, M.: Evaluating 16-bit processors for elliptic curve cryptography. In: CARDIS. pp. 166–181 (2011)