Putting the Software Radio on a Low-Calorie Diet

Prabal Dutta[†], Ye-Sheng Kuo[†], Akos Ledeczi[‡], Thomas Schmid[†], and Peter Volgyesi[‡]

[†]Computer Science & Engineering Division
University of Michigan
Ann Arbor, MI 48109
{prabal,samkuo,thschmid}@eecs.umich.edu

[‡]Institute for Software Integrated Systems
Vanderbilt University
Nashville, TN 37235
{akos.ledeczi,peter.volgyesi}@vanderbilt.edu

ABSTRACT

Modern software-defined radios are large, expensive, and power-hungry devices and this, we argue, hampers their more widespread deployment and use, particularly in low-power, size-constrained application settings like mobile phones and sensor networks. To rectify this problem, we propose to put the software-defined radio on a diet by redesigning it around just two core chips – an integrated RF transceiver and a Flash-based, mixed-signal FPGA. Modern transceivers integrate almost all RF front-end functions while emerging FPGAs integrate nearly all of required signal conditioning and processing functions. And, unlike conventional FPGAs, Flash-based FPGAs offer sleep mode power draws measured in the microamps and startup times measured in the microseconds, both of which are critical for low-power operation. If our platform architecture vision is realized, it will be possible to hold a software-defined radio in the palm of one's hand, build it for $100, and power it for days using the energy in a typical mobile phone battery. This will make software radios deployable in high densities and broadly accessible for research and education.

Categories and Subject Descriptors

B.4.1 [**Input/Output and Data Communications**]: Data Communications Devices

General Terms

Design, Performance

Keywords

Software defined radios, wireless sensor networks, energy-efficiency, low-power communications

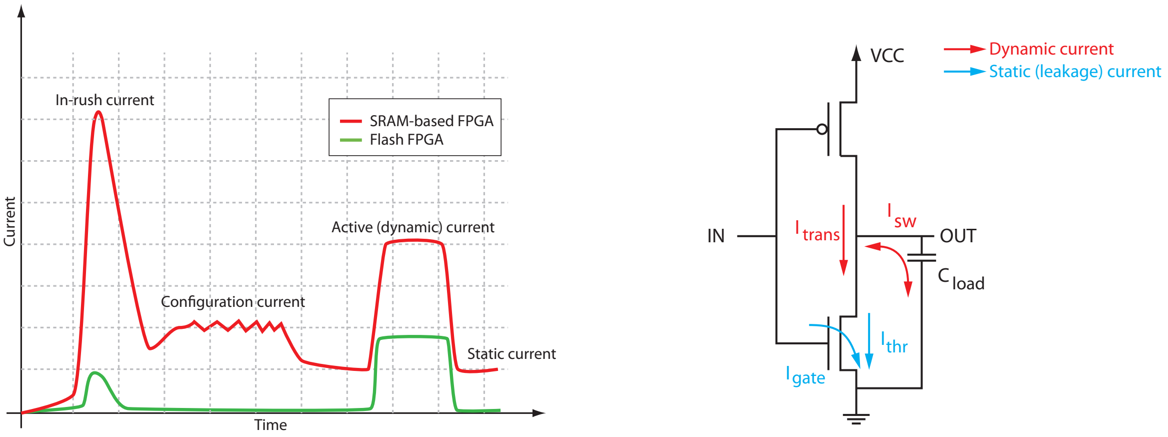
Hotnets '10, October 20–21, 2010, Monterey, CA, USA.

1. INTRODUCTION

Traditional software-defined radio (SDR) architectures offer high performance, which has led to an impressive array of applications. But modern SDR platforms are also *large*, *expensive*, and *power-hungry systems*, making them ill-suited to low-power, high-density, and mobile settings. Their large size results principally from their modular architecture. Separate boards implement the radio front-end electronics, the field-programmable gate array (FPGA) glue logic, and the baseband and protocol processing back-ends, the latter using a high-speed digital signal processor (DSP) or single- or multi-core general-purpose processor (GPP). These different boards are interconnected using fast or wide buses. Their modular architectures and high-performance analog front-ends and digital back-ends also contribute to their high cost. And, their high power draws are due to the lack of power gating on the radios and processors, and the difficulty of duty cycling RAM-based FPGAs.

For instance, Rice University's Wireless Open-Access Research Platform (WARP) uses the Xilinx Virtex-II Pro FPGA and costs $9,750 [2]. The Agile Radio project at Kansas (KUAR) uses an embedded PC platform augmented with the same FPGA for its baseband processor [28]. Rutgers' WINC2R SDR platform uses an embedded CPU and two high-end Xilinx FPGAs, one for baseband processing and one for the network layer [27]. Ettus Research's USRP uses an Altera FPGA for low-level hardware control and an x86 CPU for baseband signal processing, and costs $700 in its base configuration (the USRP2 costs $1400) [1]. MSR's SORA platform uses a Xilinx Virtex-5 FPGA for low-level hardware control and DMA-transfers between the cores of a processor, and it uses a PC host, a PCIe FPGA daughterboard, and a WARP radio board [40]. However, none of these systems are small, low-cost, or energy-efficient.

These observations raise the question of whether it is possible to build a small, cheap, and low-power SDR platform. Berkeley's Waldo platform, measuring just under 2.5" by 3.5", and costing approximately $400, provides some evidence that a *small* and *less expensive* SDR platform may be feasible [23]. Waldo, based on a Xilinx Spartan 3 FPGA, and built to support RF time-of-flight (ToF) ranging, is not quite *inexpensive* or *low-power*.



(a) Startup and operating power SRAM and Flash FPGAs.

(b) Static and dynamic currents inside the CMOS buffer.

Figure 1: Power dissipation in SRAM and Flash FPGAs. SRAM-based FPGAs experience much high powerup, configuration, and static power than their Flash-based counterparts due to the use of an SRAM-based interconnect matrix. Both types of FPGAs draw dynamic power in proportion to their activity factors. For low activity factors, SRAM static (leakage) can dominate the dynamic (active) currents, making SRAM-based FPGAs ill-suited to low-power operation.

Waldo eschews modularity and generality, and it offers small size and modest cost, through system integration and application focus. An SDR platform targeted to mobile communications and sensor networks could benefit from a similar platform specialization, for example by using highly-integrated radio front-ends and processing back-ends that are tailored to needs of short-range, personal area networks. Modern transceivers, for example, integrate almost all radio front-end functions, including RF-to-baseband pathways, an RF RX/TX switch, power amplifiers, oscillator circuitry, digital frequency synthesizers, and baseband/control interfaces, into a single chip that draws very low sleep power and can wake up rapidly when needed. When coupled with mixed-signal FPGAs that integrate ADC/DAC, FPGA, and CPU blocks into a single device, most of the elements of a lightweight SDR can be found in just two chips, leaving mainly the issue of power.

Mainstream (SRAM-based) FPGAs draw power in four distinct modes, as Figure 1 shows. In the *powerup* mode, the device (look-up tables, interconnect, I/O pads) must be configured, which requires initial charging of the distributed capacitances and causes a significant in-rush current to flow. Proper power sequencing can mitigate this problem, but cannot eliminate it completely. Then, the FPGA enters a *configuration* mode, which consists of shifting in a several megabits long configuration bitstream. This stage also has a significant current drain and experiences non-trivial delay before the FPGA is ready for use. Finally, during normal operation, the SRAM-based FPGAs dissipate power in two different ways, during *active* operation and through *static* leakage. It is this static leakage that dominates at low activity factors, and makes SRAM-based FPGA ill-suited to a low-power software radio platform.

Since static leakage dominates, traditional approaches to low-power operation become impossible. Frequency scaling, for example, addresses dynamic power so simply lowering or suspending the internal clocks cannot achieve truly low-power operational states on these devices. Voltage scaling, or power gating in the extreme, where the supply is turned off, is not an effective solution either, and may create additional problems as well. First, by turning off power to the device, the memory contents will be lost. Second, the in-rush and reconfiguration currents are incurred on every subsequent powerup following a power down. These startup costs could easily render any savings from the sleep mode moot. Finally, latency incurred due to reconfiguration prevents fast or real-time wake-up techniques that are popular in low-power MAC protocols from being used. Therefore, duty cycling is limited to long sleep cycles and cases in which the discarding of the application state can be tolerated.

2. RECIPE FOR A LOW-CALORIE DIET

It is well-known that the radio dominates the system power budget in many low-power wireless deployments. For example, in the Great Duck Island sensor network deployment, 84.4% of the node-level power budget (totaling 1,172 μW), went to radio operation [38] and in the Wireless Soil Ecology deployment, 97.8% of the power budget (totaling 994 μW) supported radio operation [39]. If radio operation could be made more energy-efficient, deployment lifetimes would be extended and data collection volumes improved.

Prior work [12], has shown the dependence of radio duty cycle on several key factors. For asynchronous low-power MACs, like B-MAC [32], the achievable duty cycle is highly dependent on the latency of *radio startup*. For synchronous low-power MACs, like SCP [42], the duty cycle is heavily dependent on *clock stability*. And, for low-power, *receiver-initiated* MACs, the achievable duty cycle also depends on being able to *quickly load and unload data* from the radio and *efficiently poll neighbors for pending traffic* [29, 37]. Unfortunately, modern radios lack the needed hardware support to fully implement and evaluate these ideas. With proper hardware support, analysis suggests that the radio power draw could be reduced by one to two orders of magnitude [14].

Unlike monolithic radio transceivers, in a software radio platform, the "radio" is actually distributed across multiple, distinct integrated circuits. Therefore, it is not enough to simply make the radio front-end low-power. Rather, both the radio front-end *and* the processing back-end, typically implemented by a CPU or FPGA, must be low-power. The challenge lies in making traditional FPGAs draw less power.

To sidestep the power problems inherent in SRAM-based FPGAs, we propose to use a new kind of FPGA, based on Flash memory, that has become available recently. First, by using Flash-based switches rather than SRAM-based ones, the aggregate static power draw is greatly reduced (to just a few *microamps*). Second, since Flash memory retains its state when power is removed, there is no need for loading the configuration information, which results in very fast powerup times (typically just a few *microseconds*). Third, no external configuration flash is required, which eliminates a component and its associated footprint, cost, and power.

These devices are therefore excellent candidates for implementing low-power wireless platforms with efficient duty cycling while keeping most of the benefits of the FPGA approach. One programmable logic manufacturer (Actel) has shifted its whole product line towards low-power applications targeting mobile, embedded, and control applications. Their IGLOO device family is built on Flash and consists of small to midsize devices capable of integrating a soft ARM Cortex-M1 (implemented using the FPGA fabric) with other IP cores on a single die. Efficient duty cycling is directly supported by Actel's FlashFreeze technology which suspends all internal clock domains and puts all I/O pins in a high-impedance state, effectively removing the device from the circuit board. Since the core remains powered, because of the negligible static power drain, the current state of the application is preserved and the device is capable of waking up almost instantly. Actel's most recent offering, called SmartFusion [4], integrates the Flash-based FPGA with a hard ARM Cortex-M3 processor (implemented using dedicated silicon), analog signal conditioning, flash memory, SRAM, and an Ethernet MAC – an ideal platform around which to build a software radio.

The final element of a low-power software radio must be measurement. Since much of the research agenda in low-power systems is focused on *achieving low-power operation*, the hardware platform should offer integrated energy metering. Wireless sensornets, perhaps more than any other area of systems and networking research, has been driven by low-power design goals. Yet, with few exceptions, publications do not report empirical energy consumption results even though hardware [6, 13, 19], software [11, 24, 43], and simulators [22, 31, 36] for energy metering and estimation exist. Of the few testbeds that do offer some metering, two offer just a single instrumented node [41, 16] and one offers 42 metered nodes out of a total of 102 nodes, but the metered nodes offer relatively low resolution [18, 21]. Given the range of options, it is surprising that relatively few publications report empirical results, choosing instead proxies like "messages sent." We propose to address this problem by integrating comprehensive hardware and software support for energy metering into the platform [13, 15].

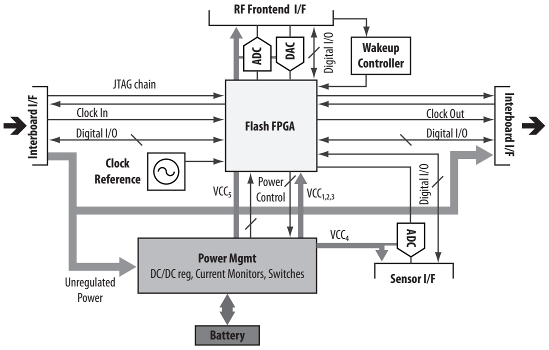


Figure 2: Proposed hardware architecture for low-power ad-hoc wireless communication research.

3. SYSTEM ARCHITECTURE

We propose to build a small, low-cost, and low-power SDR around a 2.4 GHz radio front-end and a flash-based, mixed-signal FPGA-based back-end, along with a stable time-base, as shown in Figure 2. Additional components include a power supply, wakeup controller, oscillators, and optional high-speed ADC/DAC and expansion ports.

The radio, a Maxim MAX2831, integrates almost all traditional RF functions, including RF-to-baseband pathways, an RX/TX switch, power amplifiers, oscillator circuitry, digital frequency synthesizers, baseband/control interfaces, and antenna diversity, into a single chip that draws very low sleep power and can wake up rapidly when needed. When coupled with a mixed-signal FPGA, a SmartFusion A2F200M3F, that integrates an analog compute engine (ACE) with 12-bit ADC and DAC, an FPGA with 200K gates, and an ARM Cortex-M3 processor with 64K SRAM and 256K Flash, into a single device, most of the elements of a lightweight software radio can be found in just two chips that can be connected directly for low baseband modulation rates – approximately 100K symbols per second – or augmented with inexpensive, high-speed ADC/DACs for higher data rates. Table 1 shows that the cost of the major platform components is $64, leaving $36 for the power supply and discrete parts.

Desc	Mfg	Part	Size	Cost
FPGA	Actel	A2F200M3F	17 mm x 17 mm	$40
Radio	Maxim	MAX2830	7 mm x 7 mm	$4
OSC	Maxim	DS32kHz	11 mm x 11 mm	$4
PCB	4PCB	4 layer	38 mm x 63 mm	$5
ADC	ADI	AD9288	9 mm x 9 mm	$6
DAC	Maxim	MAX5189	6 mm x 10 mm	$5

Table 1: Major components of the proposed design. The radio and FPGA cost less than $50. An external ADC/DAC could offer 40 Msps/40 MHz, nearly 67x faster than the FPGA's integrated ADC/DAC, for $11.

4. LOW-POWER PROCESSING

The real strength and novelty of the proposed platform lies in the FPGA fabric and its memory-mapped access by the hard ARM Cortex-M3 processor, since key elements of the physical and media access layers of the radio communication can be implemented as IP cores. Also, the FPGA fabric can be populated based on the demands of the application, and could also include typical microcontroller accessories like timers, counters, serial bus interfaces, and so on. Finally, high performance signal processing – or other computationally intensive functions – can be accelerated by providing them as Verilog or VHDL IP cores.

At the machine level, the processor would be able to communicate with other peripherals and custom cores via a well-defined register interface, greatly simplifying the design of higher-level software interfaces through the use of thin proxies. These can be used to hide the differences between a software-based implementation and its hardware-based alternative, allowing designers to strike an appropriate balance between implementing an algorithm by time-multiplexing it on a CPU or parallelizing it across the FPGA fabric, depending on power, performance, and latency requirements.

The fine-grained control over the supply nets provides a straightforward way to save power while the FPGA is turned on and the clock networks are active. However, truly power aware applications need to duty cycle the central processing unit also. The Flash technology itself enables very low standby currents and rapid wake-up, but these benefits can be exploited only if supported by efficient duty-cycling logic.

One option is to use an external controller or real-time clock to turn on and off the FPGA. In this case, the FPGA would configure the external logic about when it should be woken up before powering down. The external logic might use additional information as well (e.g. power level, interrupts from other components on the board). The potential benefits would have to be weighed against the additional cost and complexity that such controller logic would require.

The second alternative is to use clock scaling and gating within the FPGA. Using this approach, the FPGA is never suspended completely; a small dedicated IP core controls the I/O buffers and clock networks. Hardware support for this option is lacking today, but on the technology roadmap.

A third option is to extend the RF section and implement a low-power energy detector capable of waking up the rest of the board. This approach requires a mechanism for minimizing the power requirements of the radio-initiated wake-up, since a minimal part of the RF front-end is needed for energy detection, relaxed frequency selection, lower receiver gain, analog comparator(s), and duty cycling. The Actel SmartFusion's integrated analog compute engine offers a number analog measurement, thresholding, and filtering functions that support this option. The new Flash FPGA-based systems and custom RF front-end provide a range of options which were not feasible in previous software radio designs.

5. LOW-POWER COMMUNICATIONS

A radio's listen duty cycle, which is what often dominates the power budget, is proportional to the time needed to poll the channel, which itself is dominated by the radio startup time, which in turn is dependent on the crystal oscillator startup latency. A typical low-power radio will employ a 16 MHz crystal which will require on the order of 1 ms to stabilize due to the relatively high Q-factor needed for stable PLL operation [33]. To reduce the radio startup time, the stabilization period of the oscillator must be reduced.

We propose to use a digitally-controlled oscillator (DCO) *to drive the crystal close to its resonant frequency on startup*. For maximum effectiveness, the DCO will need to be calibrated against the crystal using standard techniques employed today to calibrate on-chip RC oscillators. One approach is to let the two oscillators run freely, connect their outputs to a pair of counters, apply a common reset and gating signal to the counters, and adjust the DCO frequency up or down depending on whether the DCO runs behind or ahead, respectively, of the primary crystal. Fortunately, the logic to implement such a fast-start oscillator is available on the FPGA.

For synchronous MAC protocols, holding data volume and data rate constant, the minimum achievable duty cycle is linearly dependent on the pairwise clock skew (or rather uncertainty in clock skew) between two nodes [12]. The simplest way to lower the floor on achievable duty cycle is to reduce the clock skew, which in turn requires stabilizing the crystal using a range of simple, complex, or exotic mechanisms [5, 7, 8, 9, 17, 26, 34, 35]. We find, however, that a commercially-available temperature-compensated crystal oscillator (TCXO) like the DS32KHz [3] offers sufficient stability – on the order of ±2 ppm indoors – and draws only 2 μA, *offering a lower bound on the duty cycle of just*

$$DC = 2 \times r_{skew} = 0.0004\%.$$

Many researchers have recently proposed using receiver-initiated link layers to achieve reliable and low-power communications [12], reduce routing state in the network [29], support fast and efficient network wakeup [25], enable collaborative communications [10], and improve throughput under contention [37].

All of these receiver-initiated protocols require a poll or probe operation followed by subsequent data transfer [14]. Unfortunately, modern radios are ill-equipped to support efficient probing and probe filtering. Today these operations require multiple expensive frame transfers between the radio and processor, including probe, data, and timing-critical acknowledgement frames. Without hardware support, today's receiver-initiated protocols are not competitive with their transmitter-initiated counterparts. Our analysis suggests that if the critical data paths for low-level MAC operations can be synthesized into the FPGA, *primitives supporting receiver-initiated operation can be streamlined*, including: (i) a packet filtering framework that can filter based on source, destination, or other (arbitrary) frame fields, or their simple comparisons and conjunctions; and (ii) zero-copy forwarding [30], which reduces overhead in multihop streaming [20], by zero-copy RX/TX FIFO transfers.

6. RESEARCH CHALLENGES

The design, implementation, and use of the proposed platform raises many technical challenges, including architectural ones like integrating the CPU and FPGA subsystems, hardware ones like designing the radio clock and wakeup circuits, software ones like how best to use and leverage existing libraries like GNURadio, and operational ones like how to deploy and manage a large network of software radios. In this section, we briefly discuss some of the questions that this proposal raises, and the challenges that they pose. Fortunately, many of these questions do not need to be addressed immediately. Rather, the integrated yet flexible architecture will enable these questions to be explored in the context of a real working system. In other words, we can let chaos reign until we understand the design space, and then we can rein in the chaos to converge on known good design points.

A number of basic architectural questions are raised by this approach since this design presents a radical departure from the current state-of-the-art. What is the appropriate level of detail exposed by the hardware that applications and protocols require? What is the right interface between a processing core and the radio to support the greatest flexibility, yet ensure software component reuse? How can radio-, time- and sensor-based wake-up be supported to accommodate a range of duty cycles and on-demand wakeup? How can the RF front end wake up as fast as the processing core, and yet minimize false alarms? How can the platform support tightly-coupled synchronous operation of multiple nodes for antenna diversity, time-of-flight, or beamforming research?

Another area rich in research questions is how to interface the CPU and FPGA fabrics in the mixed-signal FPGA. This interface has often been cited as a problem with traditional software radios that required high performance, but where designers are unwilling to pay the programming overhead of Verilog or VHDL based IP cores, or bear the performance cost of slow communication pathways between the RF, FPGA, and CPU subsystems.

In our mixed-signal architecture, the FPGA logic sits *on* the same high-speed bus matrix as the CPU and this bus offers multi-layer communications (i.e. multiple bus masters and slaves can communicate concurrently) with up to 16 Gbps aggregate throughput. With this support, it is now possible to optimistically implement algorithms in traditional languages like C/C++ and offload only performance-critical operations to the FPGA fabric with the speed and ease of memory-mapped I/O via simple register accesses, as needed.

On a more mundane level, networks of standalone software radios, deployed as a large scale testbed, will require monitoring and management support. This operation is complicated by the pair of programmable subsystems within the mixed-signal FPGA: a hard ARM Cortex-M3 processor and a traditional FPGA. Fortunately, the SmartFusion devices offer multiple methods of reprogramming the FPGA from the CPU, providing an important degree of freedom. The mixed-signal FPGA also includes a 10/100 Ethernet MAC which allows a back-channel for programming and data collection.

7. CONCLUSION

Software-defined radios are reconfigurable communication systems that transcend historical boundaries between hardware and software subsystems, physical and logical layers, and analog and digital domains. In so doing, they enable radical new architectures, novel radio designs, and high-performance protocols that are not easy to design, implement, or evaluate using traditionally-layered approaches. Although modern SDR platforms have been used to explore many facets of the wireless design space, their current architectures make it very difficult to explore the *low-power* design space. Their use of SRAM-based FPGAs result in high static and dynamic power draws, their slow startup times are not amenable to rapid duty cycling, their radio front-ends do not support power controls, and their processing requirements place a heavy load on the system. As a result, fertile application areas like mobile phones and sensor networks that could benefit from radical approaches, but which require low-power operation, remain relatively unexplored.

To address this inequity, we propose to create a small, low-cost, and low-power software-defined radio by leveraging emerging technology like highly-integrated radio front-ends and mixed-signal FPGA processing back-ends. This paper argues that a software radio with a footprint of just a few square inches, that costs less than a $100, and is able to operate from a small Lithium battery, is imminently feasible. If successful, this work will be amplified by the new science it will enable. Many of the enabled research areas are currently either completely out-of-reach or exist only in severely limited forms in low-power nodes. Hence, it will be an enabling technology for many high-impact, large scale applications of low-power, ad-hoc wireless networking where high performance and/or precise timing are required, including high-frequency power metering, infrastructureless audio/video streaming, and structural health monitoring.

8. ACKNOWLEDGMENTS

Special thanks to the anonymous reviewers for their insightful and constructive comments. This material is supported in part by the National Science Foundation awards #964120 and #0964592 ("CNS-NeTS"). Additional NSF support was provided under Grant #1019343 to the Computing Research Association for the CIFellows Project.

9. REFERENCES

[1] USRP: Universal Software Radio Peripheral. `http://www.ettus.com`.

[2] WARP: Wireless Open-Access Research Platform. `http://warp.rice.edu/trac`.

[3] DS32kHz: 32.768kHz Temperature-Compensated Crystal Oscillator. `http://datasheets.maxim-ic.com/en/ds/DS32kHz.pdf`, aug 2007.

[4] Actel. Actel SmartFusion FPGA Fabric User's Guide. `http://www.actel.com/documents/SmartFusion_Fabric_UG.pdf`, 2010.

[5] Aebischer, D., Oguey, H., and Kaenel, V. V. A 2.1 mhz crystal oscillator time base with a current consumption under 500 na. *IEEE Journal of Solid-State Circuits, Vol 32* (1997).

[6] Andersen, J., and Hansen, M. Energy Bucket: A Tool for Power Profiling and Debugging of Sensor Nodes. In *Proceedings of the 2009 Third International Conference on Sensor Technologies and Applications* (2009), pp. 132–138.

[7] Benjaminson, A., and Stallings, S. A microcomputer-compensated crystal oscillator using a dual-mode resonator. In *Frequency Control, 1989., Proceedings of the 43rd Annual Symposium on* (May-2 Jun 1989), pp. 20–26.

[8] Bloch, M., Meirs, M., Ho, J., Vig, J., and Schodowski, S. Low power timekeeping. In *Frequency Control, 1989., Proceedings of the 43rd Annual Symposium on* (May-2 Jun 1989), pp. 34–36.

[9] Candelier, V., Caret, G., and Debaisieux, A. Low profile high stability digital tcxo: ultra low power consumption tcxo. In *Frequency Control, 1989., Proceedings of the 43rd Annual Symposium on* (May-2 Jun 1989), pp. 51–54.

[10] Demirbas, M., Soysal, O., and Hussain, M. A singlehop collaborative feedback primitive for wireless sensor networks. In *Proceedings of the 27^{th} IEEE Conference on Computer Communications (INFOCOM)* (2008).

[11] Dunkels, A., Oesterlind, F., Tsiftes, N., and He, Z. Software-based on-line energy estimation for sensor nodes. In *Proceedings of the Fourth Workshop on Embedded Networked Sensors (Emnets IV)* (Cork, Ireland, June 2007).

[12] Dutta, P., Culler, D., and Shenker, S. Procrastination might lead to a longer and more useful life. In *HotNets-VI: Hot Topics in Networking* (2007).

[13] Dutta, P., Feldmeier, M., Paradiso, J., and Culler, D. Energy metering for free: Augmenting switching regulators for real-time monitoring. In *Proceedings of the 7th international conference on Information processing in sensor networks* (2008), pp. 283–294.

[14] Dutta, P., Musaloiu-E., R., Stoica, I., and Terzis, A. Wireless ACK collisions not considered harmful. In *HotNets-VII: In Proceedings of the Seventh Workshop on Hot Topics in Networks* (oct 2008).

[15] Fonseca, R., Dutta, P., Levis, P., and Stoica, I. Quanto: Tracking energy in networked embedded systems. In *OSDI'08: In Proceedings of the Eighth USENIX Symposium on Operating System Design and Implementation* (dec 2008).

[16] Gnawali, O., Fonseca, R., Jamieson, K., Moss, D., and Levis, P. Collection tree protocol. In *Proc. of the 7th ACM Conf. on Embedded Networked Sensor Systems (SenSys), Berkeley, CA* (2009).

[17] Habic, D., Pavasovic, A., and Vasiljevic, D. The design of a microcontroller temperature compensated crystal oscillator (0̆3bcctcxo) and automatic compensation line. *Ultrasonics, Ferroelectrics and Frequency Control, IEEE Transactions on 42*, 4 (Jul 1995), 509–516.

[18] Handziski, V., Köpke, A., Willig, A., and Wolisz, A. TWIST: a scalable and reconfigurable testbed for wireless indoor experiments with sensor networks. In *Proceedings of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality* (2006), p. 70.

[19] Jiang, X., Dutta, P., Culler, D., and Stoica, I. Micro power meter for energy monitoring of wireless sensor networks at scale. In *IPSN* (2007), pp. 186–195.

[20] Kim, S., Fonseca, R., Dutta, P., Tavakoli, A., Culler, D., Levis, P., Shenker, S., and Stoica, I. Flush: a reliable bulk transport protocol for multihop wireless networks. In *SenSys'07: Proceedings of the 5th international conference on Embedded networked sensor systems* (2007), pp. 351–365.

[21] Kopke, A., and Wolisz, A. Measuring the node energy consumption in USB based WSN testbeds. In *Proceedings of the 2008 The 28th International Conference on Distributed Computing Systems Workshops* (2008), pp. 333–338.

[22] Landsiedel, O., Wehrle, K., and Gotz, S. Accurate prediction of power consumption in sensor networks. In *EmNets '05: Proceedings of the 2nd IEEE workshop on Embedded Networked Sensors* (2005), pp. 37–44.

[23] Lanzisera, S. *RF Ranging for Location Awareness*. Dissertation, University of California, Berkeley, 2009.

[24] Lorincz, K., Chen, B.-r., Waterman, J., Werner-Allen, G., and Welsh, M. Resource aware programming in the pixie os. In *SenSys '08: Proceedings of the 6th ACM conference on Embedded network sensor systems* (2008), pp. 211–224.

[25] Lu, J., and Whitehouse, K. Exploiting the capture effect for low-latency flooding in wireless sensor networks. In *INFOCOM* (2009).

[26] Lutwak, R., Vlitas, P., Varghese, M., Mescher, M., Serkland, D., and Peake, G. The mac - a miniature atomic clock. In *Frequency Control Symposium and Exposition, 2005. Proceedings of the 2005 IEEE International* (Aug. 2005), pp. 6 pp.–.

[27] Miljanic, Z., Seskar, I., Le, K., and Raychaudhuri, D. The winlab network centric cognitive radio hardware platform. *Mobile Networks and Applications 13*, 5 (oct 2008), 533–541.

[28] Minden, G., Evans, J., Searl, L., DePardo, D., Petty, V., Rajbanshi, R., Newman, T., Chen, Q., Weidling, F., Guffey, J., Datla, D., Barker, B., Peck, M., Cordill, B., Wyglinski, A., and Agah, A. Kuar: A flexible software-defined radio development platform. In *New Frontiers in Dynamic Spectrum Access Networks, 2007. DySPAN 2007. 2nd IEEE International Symposium on* (April 2007), pp. 428–439.

[29] Musaloiu-E., R., Liang, C.-J. M., and Terzis, A. Koala: Ultra-low power data retrieval in wireless sensor networks. In *IPSN '08: Proceedings of the 7th international conference on Information processing in sensor networks* (Washington, DC, USA, 2008), IEEE Computer Society, pp. 421–432.

[30] Osterlind, F., and Dunkels, A. Approaching the maximum 802.15.4 multi-hop throughput. In *Proceedings of the Fifth ACM Workshop on Embedded Networked Sensors (HotEmNets 2008)* (June 2008).

[31] Perla, E., Catháin, A. O., Carbajo, R. S., Huggard, M., and Mc Goldrick, C. Powertossim z: realistic energy modelling for wireless sensor network environments. In *PM2HW2N '08: Proceedings of the 3nd ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks* (2008), pp. 35–42.

[32] Polastre, J., Hill, J., and Culler, D. Versatile low power media access for wireless sensor networks. In *Sensys'04: In Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems* (nov 2004).

[33] Polastre, J., Szewczyk, R., and Culler, D. Telos: Enabling ultra-low power wireless research. In *IPSN/SPOTS'05: The Fourth International Conference on Information Processing in Sensor Networks: Special track on Platform Tools and Design Methods for Network Embedded Sensors* (apr 2005).

[34] Schmid, T., Charbiwala, Z. M., Shea, R. S., and Srivastava, M. B. Temperature driven time synchronization. *IEEE Embedded Systems Letters 1* (may 2009).

[35] Schmid, T., Friedman, J., Charbiwala, Z., Cho, Y. H., and Srivastava, M. B. Low-power high-accuracy timing systems for efficient duty cycling. In *ISLPED '08: Proceeding of the thirteenth international symposium on Low power electronics and design* (New York, NY, USA, 2008), ACM, pp. 75–80.

[36] Shnayder, V., Hempstead, M., Chen, B.-r., Allen, G. W., and Welsh, M. Simulating the power consumption of large-scale sensor network applications. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems* (2004), pp. 188–200.

[37] Sun, Y., Gurewitz, O., and Johnson, D. B. RI-MAC: a receiver-initiated asynchronous duty cycle mac protocol for dynamic traffic loads in wireless sensor networks. In *SenSys '08: Proceedings of the 6th ACM conference on Embedded network sensor systems* (New York, NY, USA, 2008), ACM, pp. 1–14.

[38] Szewczyk, R., Mainwaring, A., Polastre, J., and Culler, D. An analysis of a large scale habitat monitoring application. In *SenSys'04: Proceedings of the Second ACM Conference on Embedded Networked Sensor Systems* (Nov. 2004).

[39] Szlavecz, K., Terzis, A., Ozer, S., Musaloiu, R., Cogan, J., Small, S., Burns, R., Gray, J., and Szalay, A. Life under your feet: An end-to-end soil ecology sensor network, database, web server, and analysis service. Tech. Rep. Microsoft Technical Report MSR TR 2006 90, Microsoft Research, 2006.

[40] Tan, K., Zhang, J., Fang, J., Liu, H., Ye, Y., Wang, S., Zhang, Y., Wu, H., Wang, W., and Voelker, G. M. Sora: high performance software radio using general purpose multi-core processors. In *NSDI'09: Proceedings of the 6th USENIX symposium on Networked systems design and implementation* (Berkeley, CA, USA, 2009), USENIX Association, pp. 75–90.

[41] Werner-Allen, G., Swieskowski, P., and Welsh, M. Motelab: A wireless sensor network testbed. In *Proceedings of the 4th international symposium on Information processing in sensor networks* (2005), p. 68.

[42] Ye, W., Silva, F., and Heidemann, J. Ultra-low duty cycle MAC with scheduled channel polling. In *SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems* (New York, NY, USA, 2006), ACM, pp. 321–334.

[43] Younis, O., and Fahmy, S. An experimental study of routing and data aggregation in sensor networks. In *In Proceedings of the IEEE International Workshop on Localized Communication and Topology Protocols for Ad Hoc Networks (IEEE LOCAN* (2005), pp. 50–57.