

UCLA

UCLA Electronic Theses and Dissertations

Title

Energy Efficient Computing with the Low Power, Energy Aware Processing (LEAP) Architecture

Permalink

<https://escholarship.org/uc/item/5g57d836>

Author

McIntire, Dustin Hale

Publication Date

2012

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

**Energy Efficient Computing with the Low
Power, Energy Aware Processing (LEAP)
Architecture**

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Electrical Engineering

by

Dustin Hale McIntire

2012

© Copyright by
Dustin Hale McIntire
2012

ABSTRACT OF THE DISSERTATION

**Energy Efficient Computing with the Low
Power, Energy Aware Processing (LEAP)
Architecture**

by

Dustin Hale McIntire

Doctor of Philosophy in Electrical Engineering

University of California, Los Angeles, 2012

Professor William J. Kaiser, Chair

Recently, a broad range of ENS applications have appeared for large-scale systems, introducing new requirements leading to new embedded architectures, associated algorithms, and supporting software systems. These new requirements include the need for diverse and complex sensor systems that present demands for energy and computational resources as well as for broadband communication. To satisfy application demands while maintaining critical support for low energy operation, a new multiprocessor node hardware and software architecture, Low Power Energy Aware Processing (LEAP), has been developed. In this thesis we described the LEAP design approach, in which the system is able to adaptively select the most energy efficient hardware components matching an applications needs. The LEAP platform supports highly dynamic requirements in sensing fidelity, computational load, storage media, and network bandwidth. It focuses on episodic operation of each component and considers the energy dissipation for each platform task by integrating fine-grained energy dissipation monitoring and sophisticated power control scheduling for all subsystems, including sensors. In

addition to the LEAP platforms unique hardware capabilities, its software architecture has been designed to provide an easy to use power management interface, a robust, fault tolerant operating environment, and to enable remote upgrade of individual software components.

Current research topics such as mobile computing and embedded networked sensing (ENS) have been addressing energy efficiency as a cornerstone necessity, due to their requirement for portability and long battery life times. This thesis discusses one such related project that, while currently directed toward ENS computing applications, is generally applicable to a wide ranging set of applications including both mobile and enterprise computing. While relevant to many applications, it is focuses on ENS environments necessitating high performance computing, networking, and storage systems while maintaining low average power operations.

The dissertation of Dustin Hale McIntire is approved.

John Wallace

Majid Sarrafzadeh

Gregory Pottie

William J. Kaiser, Committee Chair

University of California, Los Angeles

2012

This thesis is dedicated to my family and friends.

TABLE OF CONTENTS

1	Introduction	1
1.1	Motivation	3
1.2	Thesis Contributions	5
1.3	Outline	6
2	Background and Motivation	8
2.1	Defining Energy Efficiency	8
2.2	Energy Tradeoffs	9
2.2.1	Processing Resources	11
2.2.2	Communications Resources	15
2.2.3	Storage Resources	16
2.3	Prior Work	18
2.3.1	Established Power Management Solutions	18
2.3.2	Related Research	23
2.4	The Case for Energy Accounting	36
2.4.1	Dynamic power profiles	37
2.4.2	Energy Correlation Effects	38
3	A New Approach: LEAP	46
3.1	Hardware Enablement	48
3.1.1	Power Supply Characteristics	49
3.1.2	Power Measurement Sensor	51

3.1.3	Data Conversion	57
3.1.4	Power Switching	61
3.1.5	Energy Accounting	62
3.2	LEAP Enabled Software	100
3.2.1	Operating System Integration	100
3.2.2	Energy Profiling Tools	105
3.2.3	Power Scheduling Tools	110
4	LEAP Application: GeoNET	113
4.1	GeoNET Design	116
4.1.1	Hardware Architecture	116
4.1.2	Software Architecture	125
5	Field Experiments	137
5.1	El Centro	138
5.1.1	Data Collection Efficiency	139
5.1.2	Data Archiving Efficiency	140
5.1.3	Detection Efficiency	141
5.2	Stunt Ranch	146
5.2.1	Detection Efficiency	147
6	LEAP Applications and Partnerships	154
6.1	LEAP Applications and Markets	154
6.2	LEAP Inspired Products, Research, and Partnerships	155

7 Conclusion	162
A Hardware Design Files	164
A.1 RefTek RT619	164
B LEAP2 Design Documents	184
B.1 Hardware Schematics and PCB Layout	184
B.1.1 EMAP2	184
B.1.2 HPM	197
B.1.3 SIM	207
B.1.4 MPM	212
References	218

LIST OF FIGURES

2.1	ENS system configured with example sensor array	9
2.2	Graph of power contributions of various laptop subsystems [MV04a].	10
2.3	Notional computation and communications requirements by application	11
2.4	Relative energy efficiency, execution speed, and efficiency-speed product calculations for each of the three benchmark algorithms on each processor architecture	13
2.5	Comparison of radio chipsets with equal link margins at 10^{-6} BER showing relative energy efficiency per bit, throughput, and efficiency-throughput	16
2.6	Comparison of storage media showing relative energy efficiency per read, write, throughput, and efficiency-throughput	18
2.7	ACPI architecture diagram (ACPI Specification v3.0b)	20
2.8	OnNow architecture diagram (Microsoft Corp.)	22
2.9	Wakeup information display from PowerTOP	29
2.10	Power Tutor graphical display	30
2.11	Qualcomm's TREP power profiling	34
2.12	PXA270 power dissipation over time, when running the <i>dcache</i> test program, for six CPU speeds measured by the EMAP ASIC on LEAP2	38
2.13	CPU, SDRAM, NAND and NOR power draw over time when copying a 7Mb file from NAND to NOR using JFFS2.	42

2.14	Power profile when writing directly from user space utilities. . . .	43
2.15	Typical PC hardware architecture [Wik07]	44
2.16	Energy correlation effects showing CPU power increase due to L1 cache hit	45
2.17	Traditional sampling methods: oscilloscope power profiling setup measuring the UUT (a) and fuel gauge circuit (b)	45
3.1	Applications development using a LEAP enabled design flow . . .	47
3.2	Energy-aware architecture components: power sensors, power switches, analog conversion, energy accounting	48
3.3	Typical set of ENS subsystems included the in-situ measurement in LEAP architecture	49
3.4	Power profile of Intel Core 2 Quad CPU Q6600 2.4GHz	50
3.5	Power filter formed through bypass capacitors and trace inductance	51
3.6	Resistive current sensor types a) high-side b) low-side and typical PCB power plane regions	52
3.7	ADS7888 ADC power consumption over a range of sampling fre- quencies	60
3.8	Parasitic diode affect when driving I/O from a powered domain to an unpowered domain ($V_{in} > V_{dd}$)	62
3.9	Original LEAP platform hardware architecture	64
3.10	EMAP ASIC architecture block diagram with bus interconnect . .	65
3.11	Schematic of GPIO pin logic	68
3.12	Programmable interrupt controller logic	69

3.13	Wishbone to Host Controller read cycle waveforms	70
3.14	Data collection state machine showing energy measurement method	74
3.15	ASC state machine for the TLV2548 (a) and waveform example of multiple ASC modules sampling in parallel on EMAP2 (b)	77
3.16	Power Management Scheduler architecture	78
3.17	Power Management Scheduler table entry	80
3.18	Power Management Scheduler state logic	81
3.19	Power comparison of four energy measurement solutions: (a) Total sampling power by number of channels (b) Total sampling power by number by sample rate and (c) applications processor sample read back time	86
3.20	Comparison of four energy measurement solutions when including applications processor power, assuming 1W operation with 100 sample reads/sec	87
3.21	Photo and block diagram of the LEAP2 platform	89
3.22	Photo and block diagram of the energy management and account- ing (EMAP2) module	90
3.23	Measurement accuracy of the EMAP2 current sensors and ADC for the following channels: (a) HPM CPU (b) HPM SDRAM (c) HPM NAND	92
3.24	CPU energy overhead when reading from the EMAP ASIC on EMAP2: (a) CPU sampling energy by number of sampled channels (b) by sample rate	93
3.25	Photo and block diagram of the host processor module (HPM) . .	95

3.26	Memory map of the HPM resources from the PXA270 CPU	96
3.27	Photo and block diagram of the Sensor Interface Module (SIM)	98
3.28	Photo and block diagram of the MiniPCI Module (MPM)	99
3.29	Block diagram of the developed Linux energy registry	103
3.30	Etop screenshot displaying per-subsystem power and energy information as well as per-process energy consumption.	106
3.31	LTT screenshot displaying NOR flash write information as well as per-process energy consumption.	109
3.32	Example commands from <i>emap-client</i> demonstrating scheduling capability	111
4.1	Map of aftershock activity near Japan as of March 31, 2011	114
4.2	GeoNET (RefTek RT155) product photo (a) and internal bays (b)	116
4.3	RT619 module photo	118
4.4	GeoNET RT614 Analog module performance measurements: (a) PSD of the noise floor from DC to 100Hz and (b) dynamic range of a 17.08Hz tone	120
4.5	RT155 Backbone Serial Data Protocol	121
4.6	RT619 architecture highlighting additional GeoNET capabilities	122
4.7	GeoNET FPGA architecture highlighting additional sensing capabilities	123
4.8	Diagram of the Geonet FPGA's Data Acquisition Unit	124
4.9	Diagram of RT619 software applications	127

4.10	Example of email report generated during El Centro deployment demonstrating display of distributed detection from two units . . .	132
5.1	Locations of the B003 and B004 units installed in El Centro, CA .	138
5.2	Photo of B003 and B004 unit installations at El Centro test site .	139
5.3	Energy consumed by adlogger to capture and store different format types and file systems	140
5.4	Energy consumed using different file systems and compression for data archiving	142
5.5	Energy consumed for several event detection methods	144
5.6	Energy contribution in joules from the component software applications used for event detection reporting	145
5.7	Locations of the two units installed at Stunt Ranch	147
5.8	Photo of B003 unit installation at Stunt Ranch facility	148
5.9	Solar charging and discharging over a one week time period . . .	151
5.10	Battery state during July and November periods	152
5.11	Energy consumed using CPU and FPGA data collection methods	153
6.1	Photo of MicroLEAP platform	156
6.2	Photo of the LEAP server project and RTDEMS block diagram .	157
6.3	Photo of LEAP enabled LabVIEW components	158
6.4	DARPA ADAPT module attached to expansion carrier board. . .	159
6.5	Photo of the Agile Communications NetRAT handheld computing device	160

LIST OF TABLES

2.1	Processor Types for Energy Benchmarks	13
2.2	Radio chipsets used for energy efficiency calculations	16
2.3	Storage media types used for energy efficiency calculations	17
3.1	Power Domains with operating range, Sense Resistors, Amplifier Gain, and Measurement Efficiency	57
3.2	Power Switching Considerations	61
3.3	EMAP Wishbone SoC bus addresses	66
3.4	Clock Manager domain frequency bounds for an Actel IGLOO FPGA derived from place and route timing reports	67
3.5	EMAP Energy Accounting Controller Registers	73
3.6	CAC rollover periods for data width and sample rates	75
3.7	Quicklogic QL8250 and Actel IGLOO AGL600V2 implementation details	83
3.8	Implementation details of energy measurement solutions	85
3.9	EMAP2 operating states and power consumption	90
3.10	PXA270 operating frequencies and operating power	97
3.11	SIM operating states and power consumption	98
3.12	Qualitative comparison of <i>Etop</i> and <i>Endoscope</i>	108
4.1	GeoNET design requirements list	117
4.2	GeoNET configuration of the RT155's modular bays.	118

5.1	GeoNET Energy Efficiency during El Centro experiment assuming one valid event per day	146
5.2	GeoNET Energy Efficiency during Stunt Ranch experiment assum- ing one valid event per day	148

ACKNOWLEDGMENTS

This thesis would not have been possible without the assistance and guidance of many individuals. First and foremost, I would like to thank my advisor Professor William Kaiser, who has provided me guidance and mentorship over many years and to whom I am eternally grateful. His broad academic knowledge and practical knowhow have led to new ideas and research directions, and I am sincerely grateful for their time and effort. I would also like to thank Professor Gregory Pottie, Professor Deborah Estrin, Professor Majid Sarrafzadeh, and Professor John Wallace for agreeing to serve on my Ph.D. committee.

I would also like to thank Professor Paul Davis and Igor Stubalo in the Geology Department for providing their time and support in the development of the GeoNet system. In addition the people at Reftek, specifically Phil Davidson and Bruce Elliot have also provided much assistance in the development of a now production ready seismic detection system. I would also like to express my utmost gratitude to many current and former colleagues in the ASCENT research lab whose experiences and sight have been critical in the culmination of this thesis. I would like to specifically mention Thanos Stathopoulous, Winston Wu, Lawrence Au, Brett Jordan, Timothy Chow, Digvijay Singh, and Amarjeet Singh for their help.

Finally, my sincere thanks to my wife Tamra and three children Jake, Cassidy, and Lauren for their limitless patience and support throughout my graduate study.

This thesis contains texts and figures from papers listed in my publication section. This work is supported by National Science Foundation under grant CNS-0453809 and ANI-0331481.

VITA

1974	Born, Eau Claire, WI
1996	B.S., Electrical Engineering Stanford University Palo Alto, California
1996-2000	Member of Technical Staff III Rockwell Science Center Thousand Oaks, CA
1999	Graduate Certificate in Networked Systems Stanford University Palo Alto, California
2000-2008	Systems Architect Sensoria Corporation Redwood City, CA
2004	M.S., Electrical Engineering University of California, Los Angeles Los Angeles, California
2008-current	CTO Arrayent Corporation Redwood City, CA

PUBLICATIONS

Dustin McIntire, Thanos Stathopoulos, Sasank Ready, Thomas Schmidt, and William J. Kaiser. Energy Efficient Sensing with the Low Power, Energy Aware Processing (LEAP) Architecture. *ACM Transactions on Embedded Computing Systems*, Volume 11, Issue 2, July 2012.

Thanos Stathopoulos, Dustin McIntire, William J. Kaiser. The Energy Endoscope: Real-Time Detailed Energy Accounting for Wireless Sensor Nodes. *Proceedings of the 7th International Conference on Information Processing in Sensor Networks*, IPSN 2008, pages 383-394, April 2008.

L.K. Au, W.H. Wu, M.A. Batalin, D.H. McIntire, and W.J. Kaiser. MicroLEAP: Energy-aware Wireless Sensor Platform for Biomedical Sensing Applications. In *Biomedical Circuits and Systems Conference, 2007. BIOCAS 2007*. IEEE, pages 158-162, November 2007.

Dustin McIntire, Thanos Stathopoulos, William J. Kaiser. etop: sensor network application energy profiling on the LEAP2 platform. *Proceedings of the 6th International Conference on Information Processing in Sensor Networks*, IPSN 2007, pages 576-577, April 2007.

Thanos Stathopoulos, Martin Lukac, Dustin McIntire, John Heidemann, Deborah Estrin, William J. Kaiser. End-to-End Routing for Dual-Radio Sensor Networks. *26th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies*. INFOCOM 2007. pages 2252-2260, 2007.

Dustin McIntire, Kei Ho, Bernie Yip, Amarjeet Singh, Winston H. Wu, William J. Kaiser. The low power energy aware processing (LEAP) embedded networked sensor system. Proceedings of the Fifth International Conference on Information Processing in Sensor Networks, IPSN 2006, IEEE, pages 449-457, April 2006.

Maxim Pedyash, Victor Panov, Nikolai Romanov, Dustin McIntire, Gary Gould, Loren Clare, Jonathan Agre, Allen Twarowski, and James Duncan. Modular wireless integrated network sensor (WINS) node with a dual bus architecture. US Patent 2004021788, November 2004.

William Merrill, Lewis Girod, Brian Schiffer, Dustin McIntire, Guillaume Rava, Katayoun Sohrabi, Fredric Newberg, Jeremy Elson, William J. Kaiser. Dynamic Networking and Smart Sensing Enable Next-Generation Landmines. IEEE Pervasive Computing, Vol. 3, No. 4, pages: 84-90, April 2004.

Victor S. Lin, Dustin McIntire, and Charles Chien. Robust Adaptive Error Control. IEEE Wireless Communications and Networking Conference, 2000. WCNC 2000, IEEE, pages 644 - 648.

CHAPTER 1

Introduction

A broad range of embedded networked sensor applications have appeared for large-scale systems focused on the most urgent demands in environmental monitoring, security, and other applications [EPS01, HRA07, OAB03]. These applications not only introduce important demand for Embedded Networked Sensing (ENS) systems, they also introduce fundamental new requirements that lead to the need for new ENS embedded architectures, associated algorithms, and supporting software systems. New requirements include the need for diverse and complex sensor systems that present demands for energy and computational resources as well as for broadband communication. Sensing resources include imaging devices, chemical and biological sensors, and others [ABF04]. Computational system demands for new ENS applications now include image processing, statistical computing, and optimization algorithms required for selection of proper sensor sampling [BSY04].

In addition to environmental monitoring, biomedical systems such as wireless body networks (WBN) are becoming increasingly important as a means to address rising health care costs [AWB07]. These WBN systems collect real-time patient physiologic data to improve ailment diagnosis accuracy and decrease patient recovery time. Patients are equipped with small, body-worn sensing systems to collect and analyze biological signals and in some cases transmit alert information wirelessly to trained medical personnel [WBB08]. These new biomedical

platforms require long battery lifetimes, while also requiring substantial sensing, processing, and networking capabilities.

The emergence of smart phone technology has revolutionized mobile computing by providing desktop capabilities in small portable devices. Users now demand platforms enabling rich media content and diverse sets of applications. These include computing and networking intensive applications such as streaming high definition video and real-time gaming capabilities. At the same time smart phone package sizes and battery density have remained nearly constant while users expect ever longer battery lifetimes.

The situation for enterprise computing remains critical. The total data center power consumption from servers, storage, communications, cooling, and power distribution equipment accounts for between 1.7% and 2.2% of total electricity use in the U.S. in 2010 [Mar11]. This is the equivalent to 5.8 million households, or 5% of the entire housing stock. While the rate of growth has slowed in the past few years, increased reliance upon cloud computing may accelerate the trend toward larger data centers and higher energy requirements.

Worldwide energy use in general purpose computing continues to increase as well. The number of computers in use worldwide continues to grow, from 18.7 per 1000 people in 1990 to 217.1 per 1000 people in 2011, reaching 2 billion total by 2014 [Bri08]. At the same time, the power required by an individual computer has remained nearly constant over the last 20 years [Col03]. This results in an increasing demand for worldwide energy resources. However, much of this new energy demand is due to waste. The U.S. Department of Energy reports that an average PC wastes up to 400 kilowatt-hours of electricity annually [RGK02].

It is critical that future computing platforms, across all application domains, consider energy efficiency as a primary design objective. This thesis proposes a

solution that enables run time optimization of performance under energy constraints. This will be accomplished through the development of a new hardware and software architecture including a custom ASIC solution that together provide the first task-resolved and component-resolved measurements of energy dissipation integrated into an embedded platform. This new architecture is then verified by applying the design in a state of the art environmental sensing system. This sensing system is validated in several long term field deployments and the results from these experiments are given.

1.1 Motivation

Conventional embedded networked sensor architecture, based on early sensor node systems, were developed for transducer systems that presented demand for only low energy, relatively low complexity computation, and narrowband communication [ASS02, ADL98, HSW00]. In contrast, the information needed in environmental monitoring and many other sensing applications now requires an investment in energy for sensing, processing, and communication. These requirements were once believed to preclude low average power and long lifetime operation with constrained storage battery energy resources. However, it is important to note (as will be shown here) that new ENS architectures designed to permit selection of processing and communication components for high energy efficiency and that include methods for proper scheduling of operation, can achieve low energy operation. These take advantage of the general characteristic of many applications that demand only intermittent operation of many platform resources and therefore intermittent and infrequent dissipation of energy. Additionally, heterogeneous sets of resources, including computation, communications, and sensing often permit the use of different methods to achieve the same

system level objective often with significantly diverse energy requirements. We will demonstrate that proper scheduling of the higher performance components, though requiring higher peak power operation, may result in a net energy benefit due to higher efficiency operations. The opportunity for efficiency improvement is enhanced by the increased number of heterogeneous hardware resources available to systems today. For example, the latest Qualcomm mobile applications processor (MSM8960), intended for the smart phone market, contains nine separate computational units including three general purpose CPU types, several specialized DSP's, and a mobile GPU, as well as four different wireless communications interfaces [Qua11].

It is important to note that the LEAP architecture introduced as part of this thesis is necessary since, as will be discussed, run-time measurement of energy integrated with the platform is required for optimization of energy and performance for many important applications. A series of critical characteristics for determining platform energy dissipation and sensor node platform performance are in fact only known only at runtime. For example, the dynamic nature of events means that as time evolves, data-dependent processing requirements will demand that platform schedules adapt. As an example, processor resource conflicts resulting from the requirements for both processing sensor data (in order to derive event information) while also maintaining network communication between nodes, presents a resource conflict that can be determined only at runtime due to the inherently unpredictable nature of events.

As a further example, in target detection and tracking, processor task execution time is also data-dependent and will vary with the nature of received sensor signals (including noise and interference sources) and the varying demand for source identification. Also, when presented with the opportunity to select from

multiple wireless interfaces, we require runtime adaptation since available bandwidth and network connectivity, for example, is not known except at runtime.

Finally, since the rate of arrival of events and their priority are also known only at runtime, the platform's operating schedules also must be adaptive and therefore must rely on runtime energy measurement. For example, methods that describe application requirements based upon on hardware power state dependencies [DPN02], are effective only in the restrictive limit where application characteristics are inherently stationary. Of course, this is not typical of many important sensor network applications.

1.2 Thesis Contributions

As part of this thesis, two state-of-the-art embedded networked sensing (ENS) platforms are designed, fabricated, and tested, including a custom application specific integrated circuit (ASIC) providing one of a kind energy accounting and power scheduling features. The first is used to a reference design and to verify the utility of the new design approach and across a broad range of ENS applications. The second refines this design approach for specific, highly demanding, seismic application. In addition, several new software components are composed which provide per-user and per-process energy information, facilitating energy-aware application development. This software architecture is augmented by development of a new operating system facility whereby applications will be able to register energy accounting requests through a centralized energy resource manager. This new energy-aware capability is then applied to ongoing seismic research project, where these new energy aware capabilities are tested and evaluated. In addition, new energy profiling applications are utilized to perform system optimization during operation in multiple field experiments. This includes de-

velopment — and over-the-air deployment — of new applications which reduce average energy usage while not interrupting the seismic detection mission.

The specific contributions of this thesis include:

1. A new hardware and software architecture providing the first task-resolved and component-resolved measurements of energy dissipation allowing *run-time* adaptation to *deployed* systems.
2. Construction of a new, advanced seismic sensing platform based upon this new energy aware architecture. The new platform provides significant advances over the best commercially available systems in several categories including low energy operation, rapid event reporting, and autonomous system operation.
3. Validation of the new architecture through several long term in-field experiments and providing analysis of these deployments which verify the utility of the energy aware approach.

1.3 Outline

The remaining chapters of this thesis are organized as follows:

Chapter 2 discusses the existing state of technology and reviews relevant research and commercial material.

Chapter 3 presents a new embedded system architecture, low power energy aware processing (LEAP). Here we discuss platform design considerations, as well as hardware and software architectures in detail. Existing solutions and shortcomings are reviewed and a real-time system energy accounting method is introduced for enabling energy-efficient operations. Finally we present the

performance of one reference implementation of LEAP and its enabling software.

Chapter 4 describes an application of the LEAP design architecture in a UCLA CENS project named GeoNET, designed for rapid detection of earthquake aftershock events. We present the design challenges and requirements of the GeoNET application and provide solutions through a custom LEAP enabled design. Measurement data for the latest implementation of the GeoNET seismic detection system is also presented.

Chapter 5 describes several long term deployment exercises using the GeoNET system. The experiment objectives are discussed as well as review of the technical details for each exercise. This includes discussion of the specific algorithms, equipment, and communications methods used by RT155 sensing systems. Measurements from these experiments are used to provide energy-aware modifications while in the field, validating the LEAP design principles.

Chapter 6 discusses LEAP applications beyond the existing GeoNET program. This includes applications outside of traditional embedded sensing. Both commercial and research programs using LEAP technology are described.

Chapter 7 summarizes the work presented in this thesis and explores future research directions.

CHAPTER 2

Background and Motivation

2.1 Defining Energy Efficiency

We first address the energy problem by understanding the challenges involved in building an energy efficient computing system. As seen in Figure 2.1, we must consider energy as a system asset to be invested in platform resources such as sensors, processors, storage, and communications devices. Our desired outcome is to maximize the information benefit provided to an application for any given energy expenditure. To this end, an application may choose from a heterogeneous set of processing capabilities, storage media, communications devices, and sensor types. For a particular application, hardware platform, and operating environment, several candidate sets of resources may yield equivalent information return, yet may require different energy investments to achieve this. We consider the platform resource set with the lower energy investment and still providing the task's required information content to be more *efficient*. It is important to observe that the algorithms employed, implementation strategies, and sensor modalities may all be modified to achieve lower energy so long as the resulting application achieves the same objective.

The energy contributions of individual components within the system that are utilized for any given application are obviously dependent upon the application's operations and interactions with operating system services. An example is shown

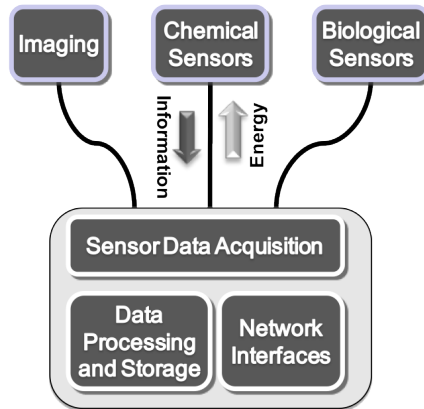


Figure 2.1: ENS system configured with example sensor array

in Figure 2.2 for several common PC applications running on a laptop computer. It is clear that the dominant power consumer is not always the laptop CPU, but rather depends on the specific application being executed.

2.2 Energy Tradeoffs

The optimal match of application to hardware resource is complex due to the wide range of requirements across the space of ENS application. Figure 2.3 notes several popular ENS applications and their typical communications and processing resources needs. Finding the best match requires understanding the spectrum of efficiency and operating power across different technologies.

In order to get a better understanding of the variations of peak power and energy efficiency across various system resource types we first survey several ubiquitous commercial technologies. The following sections present energy efficiency analysis of several common hardware component categories. We address the primary design challenges for constructing an energy efficient computing system including computation, networking, sensing, and data storage, focusing on hardware components used frequently in ENS platforms, though the results are also

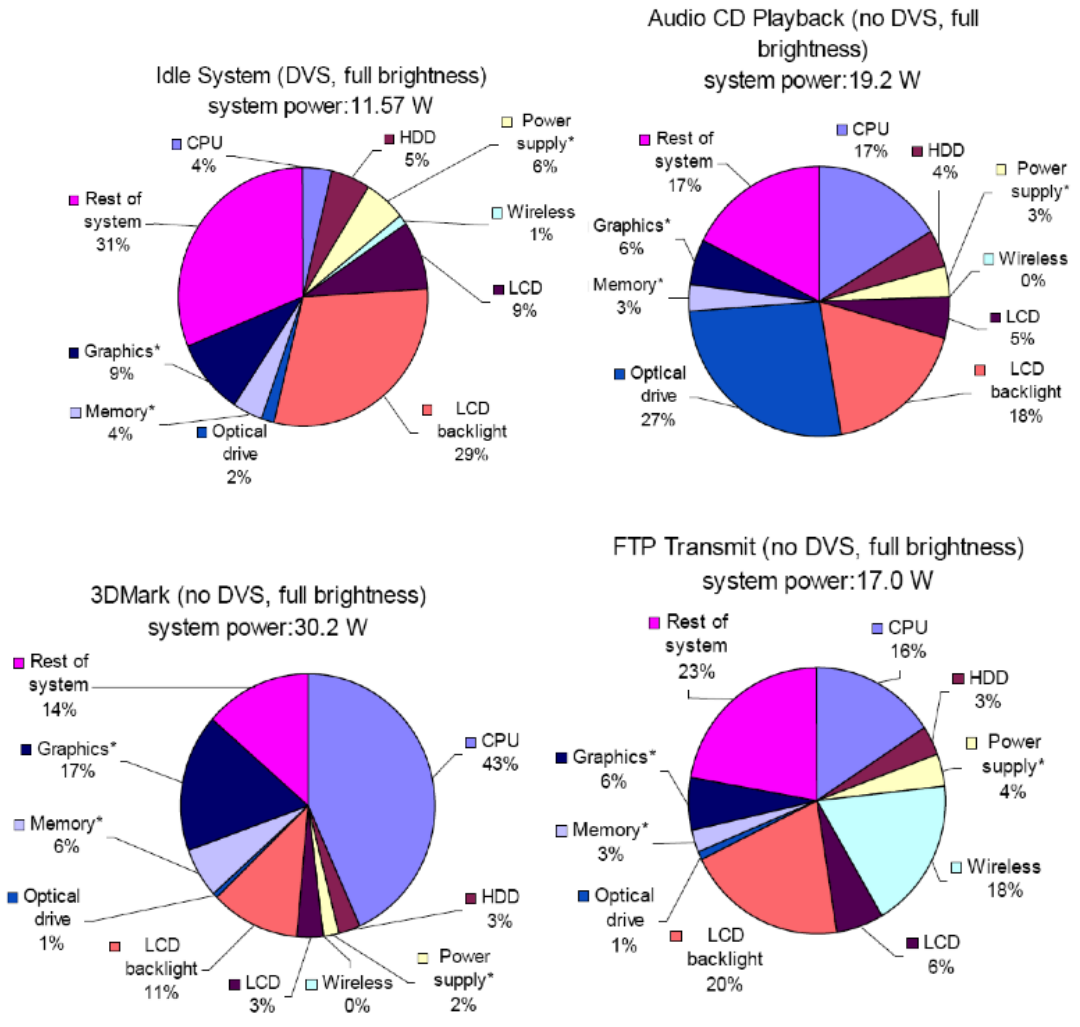


Figure 2.2: Graph of power contributions of various laptop subsystems [MV04a].

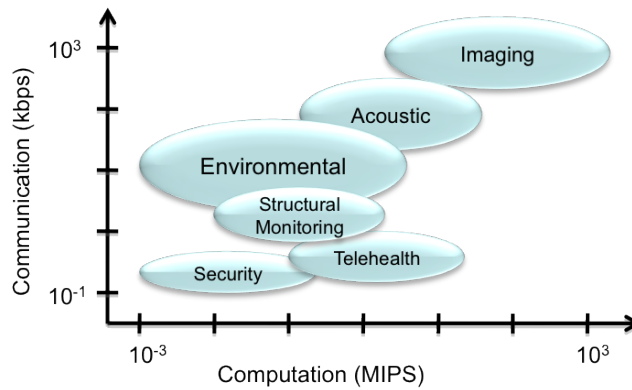


Figure 2.3: Notional computation and communications requirements by application

applicable to general purpose computing. The benchmark results influence proper hardware selection criteria, architecture choices, and system design.

2.2.1 Processing Resources

A critical decision early in the design cycle of any ENS platform is the choice of applications processor. This choice dictates to a large extent the cost, performance capability, and battery lifetime of the system. Designers are frequently incorrectly biased toward a processor choice that consumes the least amount of static or idle power and would therefore presumably have the longest battery lifetime. Unfortunately, a power metric alone is not sufficient for optimization of platform operations for each computing task since this does not consider task execution time and corresponding task energy costs. Alternatively choosing the processor option solely on the shortest execution time is also not sufficient since will neglect the peak power required. The proper choice requires both power and execution time for all components to be considered.

In order to assess the energy tradeoffs we selected several algorithms which are

typical in a typical ENS system workload. We ran these algorithms on a selection of processors previously applied to ENS platform design, to provide a quantitative value of processor efficiency relevant to embedded sensing applications. For each processor, we measured the benchmark algorithm's execution energy usage. Our first benchmark algorithm represents the ENS requirement for network applications relying on typically high error rate wireless communication channels. In this environment, data often requires verification of its validity. The cyclic redundancy check (CRC) algorithm is commonly required to establish reliable communications in these environments. The second and third benchmarks represent the requirement to process sensor data for many purposes including event detection and characterization using the widely applied finite impulse response (FIR) filter algorithm family and fast Fourier transform (FFT) [PK05].

The processors selected for benchmarking consisted of representatives from a wide range in peak power and performance ratings and have been used in numerous low power ENS platforms designs. These are listed in Table 2.1 The benchmark algorithms were run on each of the processors and the total energy consumed during the benchmark execution was measured. A high speed digital sampling oscilloscope recorded instantaneous current and voltage values during benchmark execution. These values were post processed to calculate energy usage for each processor.

The energy efficiency metric is calculated by integrating instantaneous power values taken by a bench top oscilloscope over a benchmark execution period. The energy-delay product [GH96] is useful when optimizing for energy consumption as well as execution latency. Some ENS applications may be latency tolerant; therefore trading off increased latency for reduced energy usage is desirable. In these cases, the energy efficiency metric alone is useful. However, in cases where

Processor	Data Path	Max. Frequency
ATmega128 (AVR)	8-bit	16 MHz
MSP430F1611	16-bit	8 MHz
LPC2106 (ARM7)	32-bit	60 MHz
PXA255 (Xscale)	32-bit	400 MHz
DM3730 (ARM-A8)	32-bit	1 GHz

Table 2.1: Processor Types for Energy Benchmarks

specific latency bounds need to be met, or when optimizing for both energy consumption and performance, the energy-delay product is an appropriate metric. The optimal operating point is one that minimizes both energy consumption and the energy-delay product.

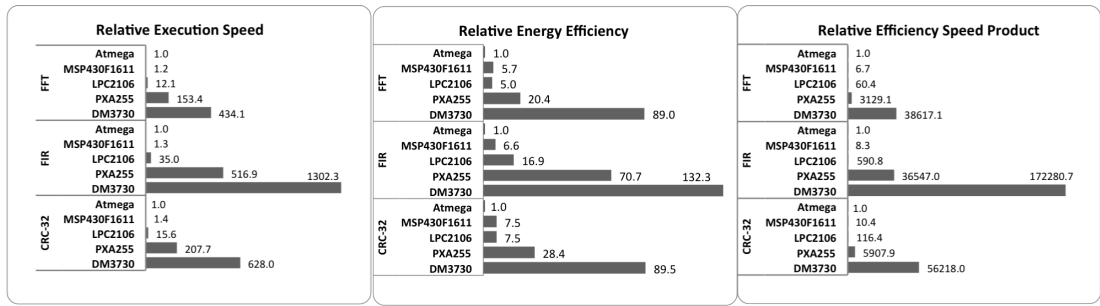


Figure 2.4: Relative energy efficiency, execution speed, and efficiency-speed product calculations for each of the three benchmark algorithms on each processor architecture

Examining the benchmark results shown in Figure 2.4 reveals several orders of magnitude energy efficiency variations between processor architectures. Also, these efficiency differences vary significantly depending on the type of operations used by each benchmark application. In general, one can see that higher performance architectures also tend to display a substantial execution energy advantage as well as energy-delay advantage. This may be expected since performance advances due to architectural features including pipelining, branch prediction, and others may result in dramatic improvement in energy efficiency due to workload

optimizations [GH96, ZBS04].

However, to effectively utilize the energy efficiency benefits of the higher performance architectures, it is important to also consider the typical high energy cost associated with a transition of the platform between operating states according to workload demand. For example the transition from a dormant standby condition to a high performance operating mode requires a substantial energy investment for restoration of system software and hardware state. These transition energy costs contribute to the determination of which processing resource is most efficient for a given task.

Many ENS experimental system deployments demonstrate that applications such as environmental monitoring or seismic sensing may be supported by systems that include lengthy dormant operational periods, during which ENS nodes are vigilant for events but may not be presented with large computing tasks. During these dormancy periods, in the absence of assigned computationally intensive tasks, the platform may be most efficiently supported by the lowest power processor selection. However, this selection is not applicable upon the arrival of events that present large computing demand. These events may require a high efficiency processing system to most effectively perform the required computing task. Here the associated transition energy costs are compensated by the energy efficiency savings.

To address this computing diversity, a heterogeneous multi-processor solution may be the proper design decision. By utilizing multiple processors, or multi-modal operation of a single processor, one is able to select the solution that best matches the present computing needs. The application must properly assign tasks to the different processors or modalities to effectively utilize the range of operations. For example the the Texas Instruments DM3730 with characteristics

described in Table 2.1 may be operated both as a high performance applications processor but may also be reconfigured to operate from the internal Davinci DSP while placing the ARM CPU into a low power suspend state.

2.2.2 Communications Resources

Just as energy efficiency has led to new design choices for processors, it should also guide selection of wired and wireless network interfaces. In addition to selection of interfaces, one may also potentially operate from multiple capability modes from a single technology option, for example selecting between the IEEE 802.11 a, b, g and n modes and transmissions rates. Indeed, the required link bandwidth and network capacity can be highly time variant and application dependent. The correct selection requires knowledge of transmission efficiency for a given communications technology.

We next survey a set of common radio chipsets in a fashion analogous to the previous processor review. The chipsets cover a broad spectrum of available bandwidth and networking standards in order to evaluate the spectrum of efficiencies available. For purposes of this comparison, data throughput was estimated to be 40 percent of the air interface rate for each system. This is consistent with simulation and empirical studies of IEEE 802.11 and IEEE 802.15.4 wireless interfaces [Gas02, KA00]. The link power values are calculated from the sum of transmitter and receiver power. Values are calculated from vendor data. An identical bit error rate (BER) and 90dB path loss is also applied for all interfaces.

While specific experimental conditions related to range, environment, presence of interference, and choice of networking protocol will influence choices, a clear contrast is noted between these interfaces. As shown in Figure 2.5, the IEEE 802.11 systems are approximately 10 times more energy efficient per bit

Chipset	Standard	Max. Throughput
TR1000	Proprietary	115.2 Kbps
CC2420	802.15.4	250 Kbps
CC1110	Proprietary	500 Kbps
Atheros 5004	802.11b	11 Mbps
Atheros 5004	802.11a	54 Mbps
Atheros 5004	802.11g	54 Mbps

Table 2.2: Radio chipsets used for energy efficiency calculations

transferred than the example IEEE 802.15.4 system and other proprietary radio alternatives. When applying the energy-delay metric, similar to the processor evaluation results, this deviation is more extreme. However, it is also seen that the 802.11g interface system requires a peak operating power much greater than that of the other choices. This result leads to an additional demand upon the ENS architecture to include multiple wireless interfaces each selected to provide both low power operation and high efficiency data transfer as variable application demands require.

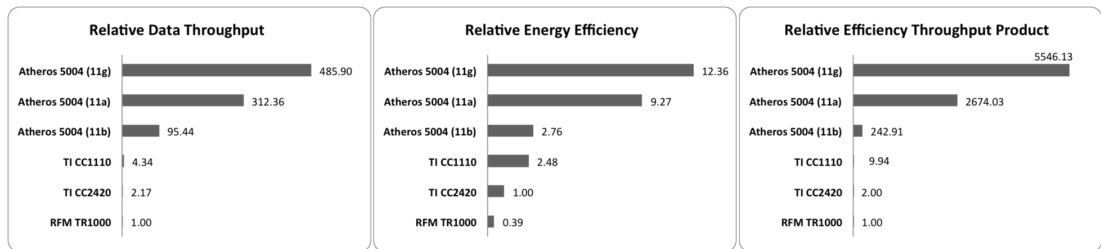


Figure 2.5: Comparison of radio chipsets with equal link margins at 10^{-6} BER showing relative energy efficiency per bit, throughput, and efficiency-throughput

2.2.3 Storage Resources

Platform data storage is an important design consideration for any embedded system. Increasing local storage resources may increase platform size, platform cost, and both static and dynamic power consumption. Conversely, increasing local

Part Number	Type	Density
28F256P30	NOR Flash	256Mb
CY62167DV18	SRAM	128Mb
MT29F8G08FABWP	NAND Flash (SLC)	8Gb
MT48LC16M32L2	SDRAM (SDR)	512Mb
MT42L128M32D2	SDRAM (DDR2)	2Gb

Table 2.3: Storage media types used for energy efficiency calculations

data storage may permit system implementations that rely on lengthy operation periods where only data acquisition and storage need occur. This reduces demand for frequent energy intensive use of processor and wireless interfaces. Selecting the proper storage type is equally critical. Each storage type has specific advantages and disadvantages when considering characteristics such as access latency, storage density, cost per byte, and power consumption. Traditional storage types used in embedded platforms consist of several varieties of SDRAM, low-power SRAM, memory-mapped NOR and NAND flash, and serial NOR flash.

The sequential read and write efficiencies for a variety of volatile and non-volatile memory types shown in Table 2.3 are calculated in Figure 2.6. The same trend toward high efficiency benefits gained by using high performance components is present in the storage media as was found in the previous sections. Again the power overhead required to maintain these high performance storage types prevents them from being used universally, but instead a determination must be made when to migrate from one type to another depending upon application specific data trends.

The optimal mixture of storage size and memory technology is highly application dependent. This dependency can be attributed to several aspects including data longevity, organization, and throughput [LC03]. Long lived data accessed sequentially, and acquired at a modest rate, e.g. chronological temperature data,

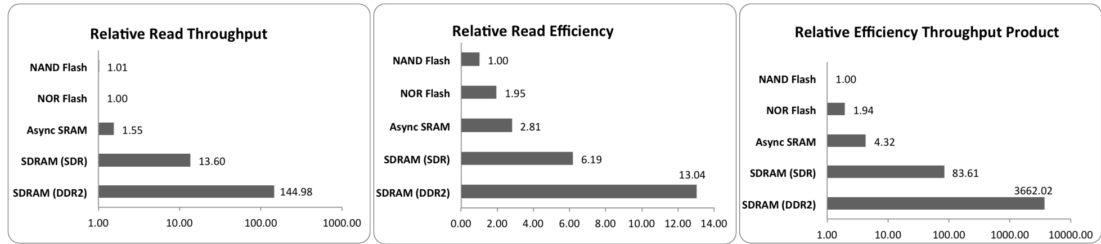


Figure 2.6: Comparison of storage media showing relative energy efficiency per read, write, throughput, and efficiency-throughput

will dictate a different storage medium type than data from short lived, randomly organized burst data such as temporary packet buffers in an active network. An energy efficient platform must be able to choose at real-time the most efficient storage type given the application’s desired storage characteristics.

2.3 Prior Work

We address related work in two sections. First we discuss power management solutions and commercial standards currently in widespread use. We then detail prior work in related research topic areas.

2.3.1 Established Power Management Solutions

Current power management solutions for general purpose platforms suffer from one or more of the following issues: a requirement for the host CPU to be active, a large memory footprint that is difficult to run on memory constrained systems, use of overly simplified models of device power requirements, or are single point solutions that are not adaptive to dynamic environments.

2.3.1.1 ACPI

The predominant power management standard in general purpose computing today is the Advanced Configuration and Power Interface (ACPI) [ACP05]. ACPI relies upon a dedicated register set, BIOS, and platform description table in order to provide device level power management status and control information. It is widely used in general purpose computing equipment as well as many other systems based on the Intel x86 architecture. This power management architecture has proven to be versatile and scalable as it is supported by the majority of hardware vendors and desktop operating systems today. While ACPI defines the hardware's power management capabilities, it relies upon a separate component within the operating system, the operating system directed power management (OSPM) layer, to the device power state most appropriate at a particular instant.

Traditionally, operating systems have used some form of software idle timers to make device level power management decisions [Lin07a, Mic06]. Software idle timers reduce energy consumption by powering down a component subsystem after a predefined period of inactivity. The time out settings can range in complexity from simple user defined constant values [Rus02] to non-stationary stochastic processes with online learning [BLR01, QQM00]. This approach is often effective in user based interactive systems, but may violate environmental real-time constraints for embedded systems. In addition, relying exclusively on idle timers neglects run time application profiling information which may be used to improve power management decisions or to modify application fidelity requirements [FS99a].

Another significant drawback of the software idle timer architecture is its dependence on the platform's host processor. The host CPU must continue to operate in order to maintain the idle timers, calculate new timeout values, and to

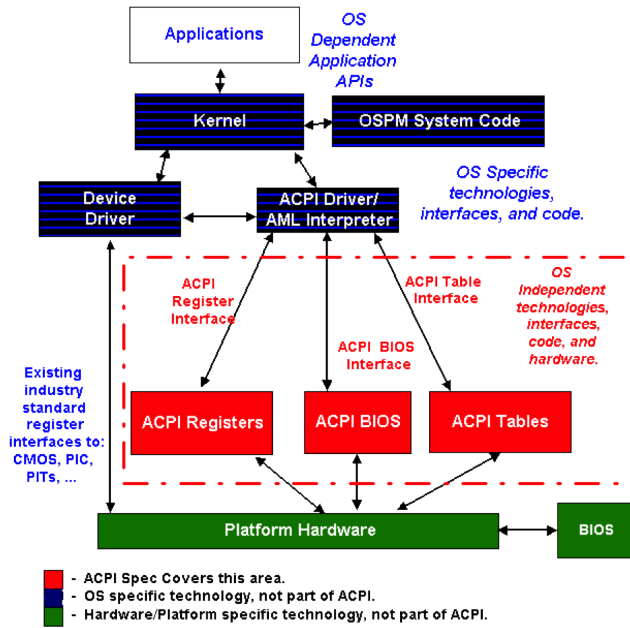


Figure 2.7: ACPI architecture diagram (ACPI Specification v3.0b)

transition devices into power management states. Often in embedded systems, the host processor may be a significant or even the predominant energy consumer [MV04b] making this choice less desirable. In environments that would allow extended periods of low vigilance, the energy overhead required to maintain CPU based idle timers may dominate. In contrast it would preferable to provide the platform with a schedulable means to transition device power states that does not require any host processor intervention whatsoever. This would prevent costly and unnecessary CPU wake up transitions.

In addition, ACPI is not well supported in compact, embedded platforms. This can be attributed to the need for a separate, complex BIOS (basic input/output system) software component as well as a memory intensive platform description table generally not included in compact embedded systems. Further, in these embedded systems power management is typically performed directly in the operating system with no intervening power management layer. Solutions are

frequently platform and processor specific and are often tuned for a particular application or environment. For example, the modern cellular phone platform has been optimized to prolong operating lifetime for a specific wireless protocol and user interface configuration. For this platform, networking power management settings are often fixed, while user interface settings may allow some manual adjustments such as the illumination timeout schedule and speaker volume. It is also important to note that the mobile phone platform does not adaptively change power management behavior as new applications are added or when supporting different users. These limitations severely restrict the platform's ability to support dynamic environments and adaptive behavior, since it would require frequent manual parameter adjustment.

2.3.1.2 OnNow

Microsoft's OnNow design initiative [Mic01] acknowledges the necessity to enable individual, device level power management changes in power-aware computing systems. OnNow seeks to build on top of ACPI enabled systems by extending power management capabilities into device driver and operating system software as well as recommending minimum hardware power management capabilities. It serves as a framework for ACPI power management information to be sent to power-aware applications and for power management directives to be propagated to platform hardware components. OnNow defines a limited set of global power management states, which are controlled directly by the operating system. The platform enters low power sleep states with any application required wake up capabilities enabled. This provides a means for individual applications to customize system sleep states depending on current requirements and provides application driven device wakeup capabilities.

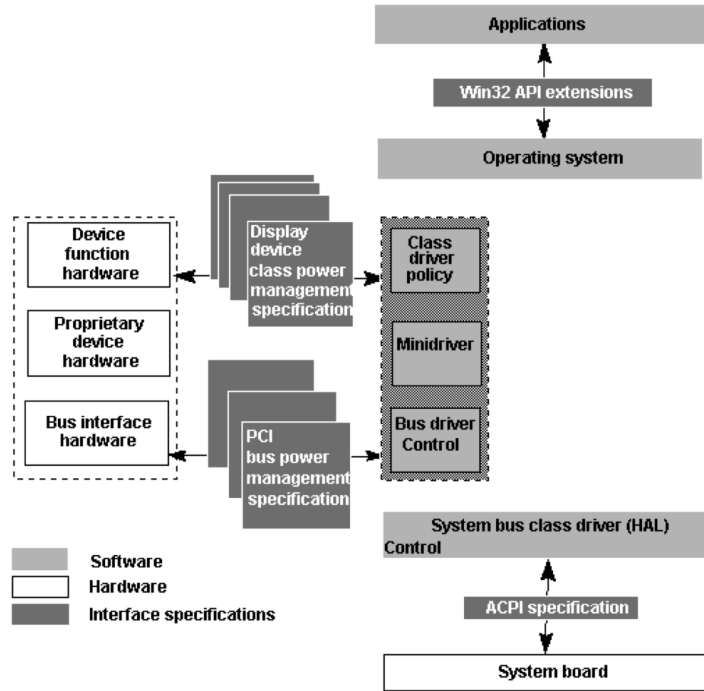


Figure 2.8: OnNow architecture diagram (Microsoft Corp.)

When the system is in its working state, devices are allowed to perform independent power management actions without coordination through the operating system. This provides a basic mechanism to do local power management based upon information available to the device driver (such as the number of open file handles). In addition the operating system dictates the CPU's operation based upon usage and power source information.

OnNow is limited strictly to ACPI enabled systems (largely laptops, desktops, and servers). As noted previously, ACPI enabled hardware is suboptimal for typical ENS platforms. In addition, device drivers must make power management optimizations based solely upon locally available information, whereas global system information is available at the operating system, no framework is provided to propagate this information.

OnNow also does not provide energy accounting features, which would enable

device level efficiency optimization. Rather, it is assumed that the individual device driver is sufficiently power-aware and will make proper power management decisions without operating system intervention. As was discussed previously, application level software can have significant impact on overall energy efficiency, thus we actually require a means to provide energy performance information to the operating system to make power management include both hardware and software decisions. This cannot be done efficiently within the device driver alone without system support for platform wide energy accounting information.

2.3.2 Related Research

The volume of prior related research is vast, encompassing general topics such as computer architecture, mobile computing, enterprise computing, as also specialty topics such as sensor networks and information theory. In general related research can be decomposed into two categories of solutions, software driven (top-down) and hardware driven (bottom-up). Following this review of related work, we analyze prior ENS platform design architectures.

2.3.2.1 Software Driven Solutions

The top-down approach to power management is typically a software defined solution requiring only limited augmentation to existing commercial hardware (for energy monitoring purposes) and utilizes one or both of the following features: device energy models for power prediction; external test equipment (such as digital multi-meters or oscilloscopes) to perform offline platform power profiling. The top-down approach can be organized into categories including: device level power management algorithms, power-aware memory management, energy-aware operating system design, application assisted power management, and system level

energy profiling.

Device Level Power Management Perhaps the most well studied topic in power-aware computing is the concept of dynamic power management (DPM). Not surprisingly, one of the most studied device subsystems in DPM is the traditional computer hard drive. It has one of the most dynamic power consumption profiles due to its numerous motors and relays. In addition, long spin-up and spin-down delays introduce significant energy penalties for incorrectly predicted power states. DPM algorithms for single device power optimization through batched I/O requests [PS03] and timeout schemes are well studied, including optimal non-adaptive algorithms [RIG02] as well as predictive [DKM94] and adaptive [HLS00] algorithms. Chung et al. used finite-state Markov chains to perform on-line adaptation, extending optimum stationary control policies through a sliding window approach to handle unknown and non-stationary workloads. Qui et al. [QQM00] used stochastic Petri nets to model concurrency and mutual exclusion primitives to DPM scheduling. Other solutions integrate real-time constraints [SCI01, UK03] for time critical applications. DPM implementations for IEEE 802.11 networking interfaces are numerous, typically augmenting or superseding the integrated power saving mode (PSM) capability. Anand et al. [ANF03] implemented an adaptive PSM feature for optimizing distributed file system energy. Additionally, optimization of processor voltage and frequency (digital voltage scaling), have been discussed at great detail [CB95]. In all reviewed DPM work however, power profiling information was either derived from manufacturer's data sheets or measured through external devices such as digital multimeters or oscilloscopes.

Power-aware Memory Management While DPM algorithms focus on the power states of individual platform devices, power-aware memory management software allows memory components to enter reduced power states through new allocation schemes incorporating memory technology characteristics. Zheng et al. [ZGS03] constructed a logical-disk subsystem allowing for the use of several different storage technologies, including hard disk, compact flash, and wireless LAN, as well as numerous file system providers. They assessed the complex interactions between these components and found strong energy dependencies on device power management mechanisms to file system usage patterns.

Others have constructed power-aware virtual memory systems which sleep RAM components dynamically by tracking active memory segments and utilizing allocation schemes to minimize the number of required active nodes. This allows inactive memory areas to be put into low power modes, reducing system energy. According to the Huang et al. [HPK03] significant energy reductions were possible (over 50 percent) using RAMBUS memory. However, this memory technology is no longer in widespread use.

Lee and Chang [LC03] proposed a memory allocation scheme incorporating the data retention energy requirements for several non-volatile memory technologies. Using trace-driven simulation techniques, the authors derived the location and number of required memory accesses for a particular task set. When mapped to the optimal storage technology they found a significant energy reduction (up to 26 percent) of memory components is possible.

Energy-aware Operating Systems Other projects have investigated energy management from the operating system level rather than due to individual components such as disk drives and CPU memory. Resource containers [BDM99] de-

couple resource tracking from the traditional operating system protection domain model. These containers allow object tracking of non-process based resources, such as network bandwidth, disk I/O, and battery energy. Waitz [Wai03] utilized resource containers to provide energy tracking of the CPU from within the Linux operating system. However, this system did not incorporate energy data from additional device subsystems such as CPU memory or network interfaces.

ECOSystem [HCA02] incorporates energy as a schedulable operating system resource, through the current abstraction. Per-process energy information is critical in ECOSystem as processes need to be charged appropriately, to enable energy-based scheduling. ECOSystem uses a combination of battery lifetime sampling (a system-wide energy measurement) and system modeling. It was one of the first to describe the dynamic power relationships between platform devices. However, in ECOSystem the CPU, and network interfaces are assumed to use fixed power values for simplicity, and the hard drive is modeled for a reduced number of operating states.

Application Assisted Power Management Application assisted power management operates by providing a direct communication capability between application software, the operating system, and specific hardware devices. This cross layer communication bridging capability allows tight coordination of power states through software triggering mechanisms. Microsoft's OnNow framework, described previously, is one such implementation. In addition, several research implementations exist with similar feature sets.

Anand et al. [ANF04] expose device power states to applications. These applications utilize 'ghost hints' to inform device drivers of missed I/O events due to incorrect power management states. Devices are then able to proactively

enable interfaces for applications prior to activation. One shortcoming of this implementation is the requirement for all applications using a managed device to support the ghosting hint capability to achieve significant energy reduction.

Heath et al. [HPH02, HPB02] investigated application transformations that increase device idle times and propagate idle time information to the operating system. Compiler optimizations transform application code to provide pre-activation and deactivation for laptop disk access. This was shown to significantly reduce disk energy consumption for the chosen task set using simulated disk hardware.

An alternative implementation [WBB02] allows applications to declare the priority of a disk I/O access as deferrable or abortable through the use of a operation time-out and cancel flag. When deployed on a hard disk using a modified Linux Ext2 file system, results were found to compare with oracle performance. Finally, Lu et al. [LBM00] describe an online task scheduling algorithm to optimize device I/O access requirement to optimally aggregate I/O requests across multiple devices. However, this algorithm assumes each device may operate independently from one another, which ignores hardware dependency relationships.

System Level Energy Profiling Energy profiling has been investigated using several methods ranging from purely model driven systems to online implementations. Tan et al. [TRJ05] characterize the operating system as a multiple entry/multiple exit program where the transition from each input to each output is first characterized with a probability distribution function and an energy dissipation value. Their intention is to allow high level designers to compare the energy costs using alternative operating systems as well as predicting the energy overhead when using different operating system primitives. The feasibility of

profiling an entire operating system for each version release is questionable.

Joule Watcher [Bel00] utilizes hardware performance monitoring units (PMUs) within the CPU to track instruction count and to calculate energy dissipation for the CPU, L2 cache, and main memory devices. Micro-benchmarks are used to find the correlation of events to energy values within the CPU. An external digital multimeter (DMM) provides the power profiling data during the micro-benchmark execution. The PMU is then used to track execution and energy of individual threads. XTREM [GMJ04] uses an identical profiling architecture for the Intel PXA255. The authors then ran energy benchmarks on several common multimedia applications to verify model accuracy to actual hardware.

PowerTOSSIM [SHC04] is a power-aware extension to the TOSSIM TinyOS simulator. PowerTOSSIM uses an empirically-generated model of hardware behavior to simulate power dissipation on mote-class devices. The model is obtained by instrumenting and profiling a single node during actual operations. An oscilloscope was used to measure power dissipation of the entire mote and a set of micro-benchmarks were used to exercise each subsystem independently.

Jung et al. [JTB07] introduce a model-based design tool that can identify the dominant energy-intensive hardware components over a set of operating patterns. The authors propose several operating states where the system components are operating in different power modes, where measured power values can be used to populate the model parameters. The proposed models assume that power dissipation of subsystems is constant among operations. However, this is not always the case, especially when a DVFS-enabled CPU is present.

PowerScope [FS99b] combines hardware instrumentation and kernel software support to provide per-process energy usage information. It uses an external multimeter and a second computer for data collection and the data processing

does not occur in real-time. Furthermore, the power-measuring system itself requires significant energy and physical resources to operate, thereby limiting its application in large-scale systems.

PowerTOP [Lin07b] uses platform ACPI information to show how effectively the system is using the various hardware power-saving features. It displays the list of processes causing CPU wakeups from low power states. It does not collect run-time energy information but instead helps the user find applications responsible for exiting low power idle and sleep states. This can be used to tune applications such that the CPU remains in sleep states for longer periods of time and thereby reducing system power.

```
PowerTOP version 1.0 (C) 2007 Intel Corporation
Cn      Avg residency (5s)  Long term residency avg
C0 (cpu running)      ( 3.8%)
C1          0.0ms ( 0.0%)          0.0ms
C2          4.4ms (57.3%)          4.4ms
C3         10.0ms (31.1%)         10.0ms
C4          2.3ms ( 7.7%)          2.3ms

Wakeup per second : 193.6
Power usage (ACPI estimate) : 13.0 W (6.5 hours left)

Top causes for wakeups:
35.2%    <interrupt> : 18042
28.4%    <interrupt> : yenta, i915@pci:0000:00:02.0
13.6%    <interrupt> : ipw2200, Intel 82801DB-ICH4
 4.6%    Xorg : do_setitimer (it_real_fn)
 3.7%    firefox-bin : schedule_timeout (process_timeout)
 3.5%    xchat : schedule_timeout (process_timeout)
 1.6%    firefox-bin : schedule_timeout (process_timeout)
 1.3%    gnome-terminal : schedule_timeout (process_timeout)
 1.1%    gnome-power-man : schedule_timeout (process_timeout)
 1.1%    emerald : schedule_timeout (process_timeout)

Suggestion: Enable the CONFIG_USB_SUSPEND kernel configuration option.
This option will automatically disable UHCI USB when not in use, and may
save approximately 1 Watt of power.
```

Figure 2.9: Wakeup information display from PowerTOP

With the emergence of the smart phone, a ubiquitous high performance mobile computing platform, consumers are able to run a wide variety of applications encompassing many of the elements of traditional sensor networks. Sensors such as geolocation, motion, light, compass direction, atmospheric pressure, and

temperature have become standard equipment on these systems [ifi12]. In addition, several wireless interfaces including short range (Bluetooth), medium range (*WiFi*), and long range (cellular, LTE) are available to accompany the gigahertz speed multicore applications processors needed to run the demanding application sets required on these systems. The hardware components used on these platforms have incorporated a number of power-saving features, allowing components to dynamically adjust their power consumptions based on required functionality and performance. In order to enable users to gain insight into the power expenditures of these platforms, several power profiling applications have appeared.

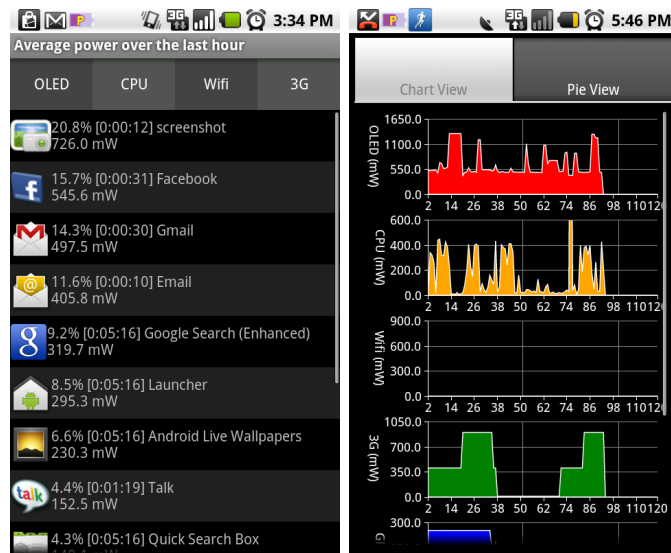


Figure 2.10: Power Tutor graphical display

In 2011 Zhang et. al. [ZTQ10] developed PowerTutor (Figure 2.10), an Android application that estimates energy usage of smart phone subsystems based upon prior statistical modeling of the hardware. Energy usage of several platform subsystems including CPU, wireless, camera, audio, and storage are included. These energy models are obtained using methods similar to Jung’s methods in that these models are derived by exercising micro benchmark applications that

attempt to isolate specific components under test (such as CPU, LCD, wireless, etc) while minimizing power variation due to other subsystems. As with prior energy benchmarking applications, the platform's lack of fine-grained power consumption data severely restricts the depth of insight one can gain for adaptive operation.

2.3.2.2 Hardware Driven Solutions

The bottom up approach to power management is driven by low-power hardware VLSI design and utilizes one or more of the following features: transistor or instruction level energy models; CPU instruction counting, battery modeling; compiler instruction set optimization.

Instruction Level Energy Optimization Numerous techniques exist for energy estimation at the instruction level. These typically involve cycle accurate CPU simulations utilizing energy information from transistor level or logic gate level information. Steinke et al. [SKW01] propose an instruction set energy model for an ARM processor without detailed knowledge of the internal processor structure. Their models are based upon a power profiling method derived by exercising the internal functional units and external data buses. In contrast, Rizo-Morente et al. [RCB06] proposed a method for estimating energy usage by modeling the CPU as a linear system excited by the current due to each instruction. The authors claim average correlation of 93 percent between estimated and measured values.

Using a different instruction level energy simulation model as well as a custom power profiling tool, Simunic et al. [SBM99, SBM00] discovered that several source code optimization techniques are able to dramatically reduce power con-

sumption by up to 90 percent on an ARM processor for an example MPEG decoding algorithm, while traditional performance based compilers were shown to be ineffective at finding low power instruction set implementations. Much of the inefficiencies were found at inter-procedural boundaries where traditional compilers did not optimize for unnecessary memory access.

SoftWatt [GSI02] models the CPU, memory hierarchy, and hard disk using a simulated operating system, SimOS. Each of these devices is modeled at the micro-level including cycle accurate emulation of the MIPS CPU, including energy calculations using analytical power models. The hard disk is modeled as a finite state machine composed of several management states, while the CPU cache energy is calculated from individual memory accesses. The authors do not present any data to verify accuracy of their models however.

Marculescu [Mar00] proposes a scheme to trade instruction level parallelism (ILP) for low power operation using software execution traces. His scheme searches for the point of diminishing returns whereby adding additional instructions into the dispatch unit will no longer efficiently execute due to data dependency hazards. Instead, by decreasing the number of simultaneously live instructions, data hazards are less likely to require pipeline stalls. CPU power values were calculated using a proprietary Intel power simulator.

Battery Modeling Other well-known power management techniques exist to extend system lifetime through battery relaxation. By controlling current draw characteristics, additional battery energy is available prior to complete discharge. Lahiri et al. [LDP02] presents an introduction to battery-aware scheduling including the underpinnings of the battery recovery effects. Benini et al. [BCM00] and Luo and Jha [LJ01] discuss DPM techniques that incorporate battery modeling

algorithms. By duty-cycling large current drains from the battery. Battery relaxation is able to recover lost charge due to electro-chemical phenomena. Martin and Siewiorek [MS96] describe the effects of battery relaxation when coupled to a DVS enabled CPU. They show the non-linear relationship due to battery recovery while duty-cycling (using DPM) versus the traditional approach used under DVS. This results in extended lifetime under DPM not typically considered.

Energy-aware Hardware SPOT [JDC07] is an add-on board that enabling energy accounting on the MICA2 mote platforms. Charge accumulation is performed via a voltage to frequency converter circuit, similar to a sigma-delta ADC architecture, achieving a high resolution output and large dynamic range. The SPOT module also includes a dedicated counter value to provide charge accumulation timing information. However, a low bandwidth read back channel hinders energy measurement at high sample rates. Additionally, SPOT is a single-channel design, monitoring only the mote platforms input current and cannot easily support per-subsystem power information.

In 2011, the Qualcomm Corporation released the MSM8660 mobile development platform (MDP) [Bsq12], a reference platform for development of software applications optimized for their flagship Snapdragon S3 applications processor. Integrated into the MDP architecture are numerous current sensors located on the voltage regulator supplies. These supplies power various functions on the MDP and several different subsystems within the MSM8660 itself, for example the SMP applications CPU's, Adreno graphics processor, sensor processor subsystem (SPS), and audio DSP. Simultaneously Qualcomm released TREPAN [Ben11], an Android application that acquires and displays energy profiling information from the MDP's current sensors. Using TREPAN software designers are encouraged to run power profiles of their applications in order to enable optimization

during development. Applications are able to send intents to the TREPAN profiler so that critical sections of software execution may be correlated with energy data.

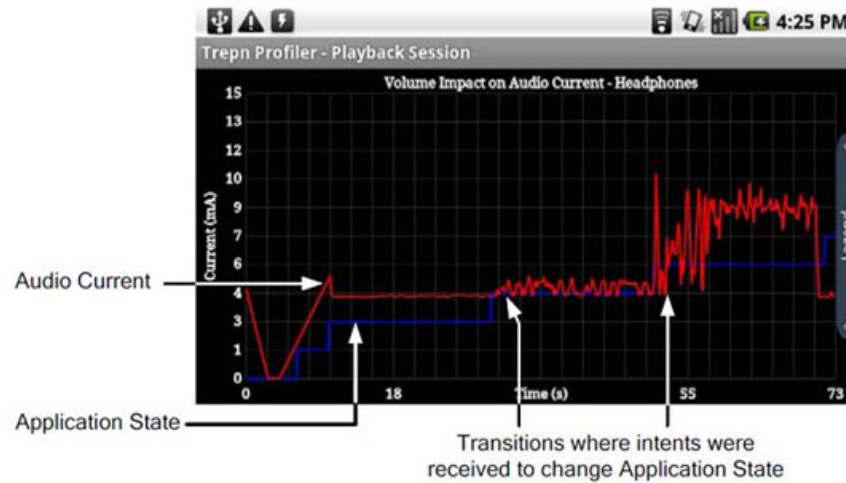


Figure 2.11: Qualcomm’s TREPAN power profiling

One significant limitation of the MDP’s power measurement features is its low bandwidth sampling. Power profiling data is limited to less than one hundred hertz per sensor due to the single I^2C bus that connects the all current sensors. As will be discussed later, operating the power profiling at less than the operating systems scheduler clock rate restricts the possibility of per process energy accounting. Additionally, since collection of the current sensor data requires software execution on the Scorpion applications processors, TREPAN does not provide a method for power profiling in low duty cycle or low power operations where many of the processing resources are disabled. Unfortunately, due to licensing concerns Qualcomm removed TREPAN from public access in late 2011 with no schedule for future release.

2.3.2.3 ENS Platform Architecture

In prior ENS research, a wide array of wireless sensor platforms with large variations in power, performance, and cost [Int02, SBC05, BBC96, ACP99, HSW00, PSC05, LRR05, Int02, LS05], have been constructed. These platforms can often be constrained by the performance limitations of low complexity components required for long battery lifetime, or by power limitations, caused by wasted energy during long device dormancy periods.

There are notable platform designs which have attempted overcome these constraints by operating over a wide dynamic range in power and capability. The PASTA [WBF04] platform attempted to solve both issues through dynamic scheduling of processing and communications resources. This platform was able to dynamically utilize various sensing, communications, and processing subsystems depending on anticipated workload. It also had an integrated macro level energy measurement capability. M-Platform [LPZ07] shares many similarities with PASTA, with emphasis on design for modularity, including a custom inter-module bus protocol designed for high efficiency data transfers. Each board has bus custom logic to perform TDMA data transfers on the platform's backplane bus.

Triage [BSC05] is a tiered hardware and software architecture for microservers utilizing a high-power platform with ample resources and a resource constrained low-power platform. The low-power platform schedules tasks to optimize the use of the high-power platform. Scheduling is based on hardware-assisted profiling of execution time and energy usage. The energy profiler uses information about the remaining battery energy as well as energy consumption during a particular operation. However, the energy profiler does not provide per-subsystem information or power tracing capabilities; instead, an external data acquisition board

must be used.

The first generation LEAP1 [MSR12] platform utilized a heterogeneous processor and radio architecture to provide both low power and high efficiency computation and networking operations. The EMAP module performed energy accounting, power domain scheduling, and sensor data acquisition. It was the first platform design to do highly granular, device level energy profiling and power management functions.

The iMote2 [Int07] platform is designed around a high performance PXA271 processor and has 802.15.4 connectivity through its CC2420 radio. It has numerous serial interfaces available for expansion boards on a dedicated connector. XSM [DGA05] is a microcontroller based platform utilizing micropower analog circuits to facilitate operating at very long intervals in a low vigilance mode of operation.

2.4 The Case for Energy Accounting

In our introduction we described a set of new ENS system requirements associated with many important applications and the resulting requirements for new platform architecture. In the previous sections we have described the wide range in energy efficiency between heterogeneous system components including computation, storage, communications, and sensing. We have also noted that heterogeneous sets of resources often permit the use of different methods to achieve the same system level objective. Finally we also note that a general characteristic of many ENS applications is demand for intermittent operation of many platform resources and therefore intermittent and infrequent dissipation of energy.

In order to make beneficial use of these characteristics and enable signifi-

cantly lower average power, we require a *new embedded system architecture*. This introduces a design where components are not selected solely for low average power dissipation, but are selected with both energy and performance criteria to achieve highest energy efficiency and lowest system energy dissipation for a specific measurement objective. This also includes energy management to enable scheduling of component operation. Thus, it is then possible to select the most energy efficient components to meet the demands of each sensing task. It is also important to note that this system design must address the requirements that arise from the need for dynamic task selection for applications that present events and phenomena that are unpredictable and known only at runtime. Specifically, an energy accounting capability is required for in-field adaptive optimization of energy and performance with methods that may only operate if provided with real-time knowledge of subsystem energy dissipation. In order to argue the need for a real-time energy accounting mechanism, we first address several misconceptions in power management and also present deficiencies of existing solutions.

2.4.1 Dynamic power profiles

When presenting new power management schemes in platforms lacking integrated support for device level, real-time energy profiling, a common misconception is to assume the accuracy of pre-generated, average power values when exercising power management states [DKM94, HLS00, LM99]. In these schemes, platform power models are empirically generated offline to simulate real-time power dissipation. The models are obtained by instrumenting and profiling a single node and running a predefined set of software benchmarks. Typically, an oscilloscope is used to measure power dissipation of the entire platform through a shunt current monitoring method. While this method proves applicable to limited complexity

devices such as microcontroller based systems, it does not accurately capture the dynamic nature of modern, high performance microprocessor based system-on-chip (SoC) devices. As can be seen in Figure 2.12, the power profile for a PXA270 processor enabled with dynamic voltage and frequency scaling (DVFS) may vary by over an order of magnitude while executing the benchmark application. This implies that assuming average power values leads to significant estimation errors when applied to power dynamic platforms. Additionally, average value estimations generated via under sampling methods can completely miss short term events altogether.

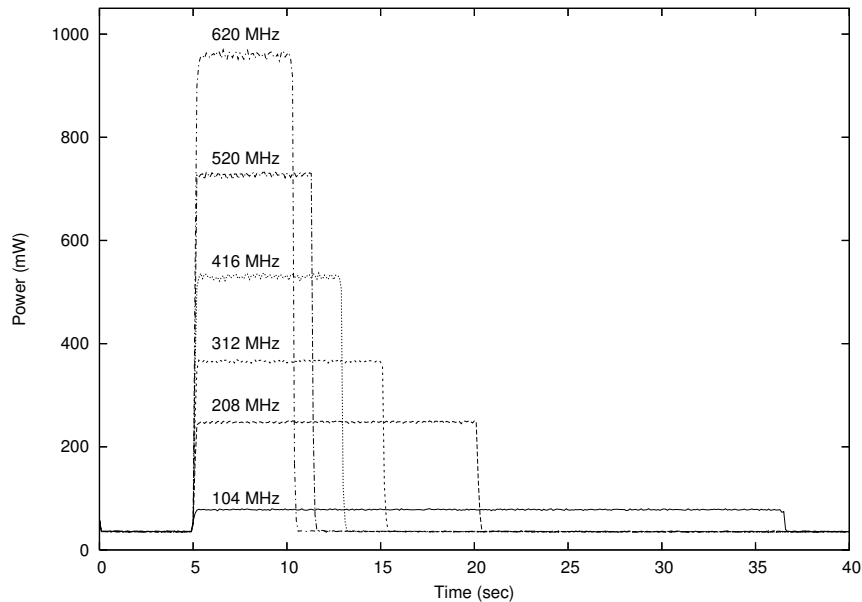


Figure 2.12: PXA270 power dissipation over time, when running the *dcache* test program, for six CPU speeds measured by the EMAP ASIC on LEAP2

2.4.2 Energy Correlation Effects

When operating without an integrated real-time energy accounting mechanism, power management schemes are unable to measure two critical aspects present

in complex systems: First, the correlation between software tasks and energy usage, and secondly, the correlation between individual hardware components themselves. We next show that energy independence of software tasks and of hardware components, as required by average power modeling, is invalid.

2.4.2.1 Software Task Sets

Figure 2.13-a shows the power profile implications for both the system CPU and flash storage media when operating a file system during a typical file copy operation. Since the flash device is a NOR based component, it supports an *erase-suspend* on write capability. This allows block erase operations (which may require tens of milliseconds to complete) to be suspended in order to perform other more immediate flash operations. The Linux file system software (Journaling Flash File System—JFFS2) utilizes this feature to perform interleaved write and erase operations. This behavior explains the pulse-like shape of Figure 2.13-b.

If instead, the NOR flash device is written directly through user-space erase and write operations (rather than through the JFFS2 file system), the power profile, is dramatically different. This is shown in Figure 2.14. It is apparent, from both the CPU and flash power profiles, that these software implementations, though providing equivalent functionality, have dramatically different impacts on overall energy usage. This is can be seen from the difference of the power profile integration between the two examples.

2.4.2.2 Hardware Components

The energy relationship between individual hardware components is also difficult to capture without specific hardware support for energy accounting. For example, when using the oscilloscope profiling method explained previously, a set of

software micro-benchmarks [GMJ04, Bel00, SHC04] are often used in order to exercise each hardware subsystem independently. This is required in order to derive individual hardware component energies. However, many hardware devices require hierarchical operational relationships with other components, thus preventing independent operation, for example the use of a communications bus between a CPU and device subsystem.

This is the case for components such as an 802.11 WLAN device attached to a PCI bus. Here, power independence of the WLAN device from its parent PCI bus, and as is shown in Figure 2.15 for a typical PC, from the parent bridging hardware (e.g. Northbridge and Southbridge) cannot be assumed, since bus activation is required prior to device operation. At the same time the parent PCI bus may also connect numerous other components not related to the targeted device. Therefore the operation of resources unrelated to the micro-benchmark, but connected to some parent resource can have significant impact on the execution and energy calculations related to the benchmarking application. In order to claim micro-benchmarking accuracy, device energy independence must be assumed, but due to connectivity of shared parent resources this is not correct.

Micro-benchmark use is also limited to known a priori hardware relationships since the benchmark models themselves are developed based upon exercising only limited subsets of the system and developing an estimation based upon weighted summations of these benchmarks. This may often neglect correlation effects that can have significant influence on total power usage. Figure 2.16 demonstrates one such example where the relationship between the CPU and cache is only discovered through our direct measurements. While it is generally accepted that there is an efficiency loss due to data cache miss cycles we also found CPU power is significantly increased during data cache hits due to fewer stalled instruction cy-

cles. This would have resulted in an 8% estimation error using micro-benchmark modeling techniques where the CPU's actual energy expenditure is dependent upon data set organization within the cache.

2.4.2.3 Traditional Energy Accounting Solutions

Traditional energy accounting techniques in ENS systems as well as in mobile computing rely on external device support, such as oscilloscope sampling or data acquisition systems [DKM94, BSC05, ANF03, ANF04, BCM00, RG00], on internal device support such as commercial "fuel gauge" circuits [LPZ07, SBC05], or peripheral circuit board modules [JDC07]. As shown in Figure 2.17, power dissipation is typically measured at the node power supply circuit input, thereby providing only system-wide energy consumption information. High sampling rate devices such as an oscilloscope or other data acquisition system provide high sampling rates and as such can acquire and display power dissipation data in real-time. However, they are external devices and may not probe all internal subsystem circuit paths. Likewise, these are impractical for use in deployed systems that need compact low-power real-time energy consumption information. Moreover, they are large, costly, and require substantial power to operate, and are thus impractical for use in a deployed system. A fuel gauge or an integrated peripheral solution is sufficiently low-power to be included in actual field applications; however, those devices cannot meet real-time constraints as they are limited either by their internal sampling rate or by the speed of the communications bus. We demonstrate this issue later in Section 3.1.5.2.

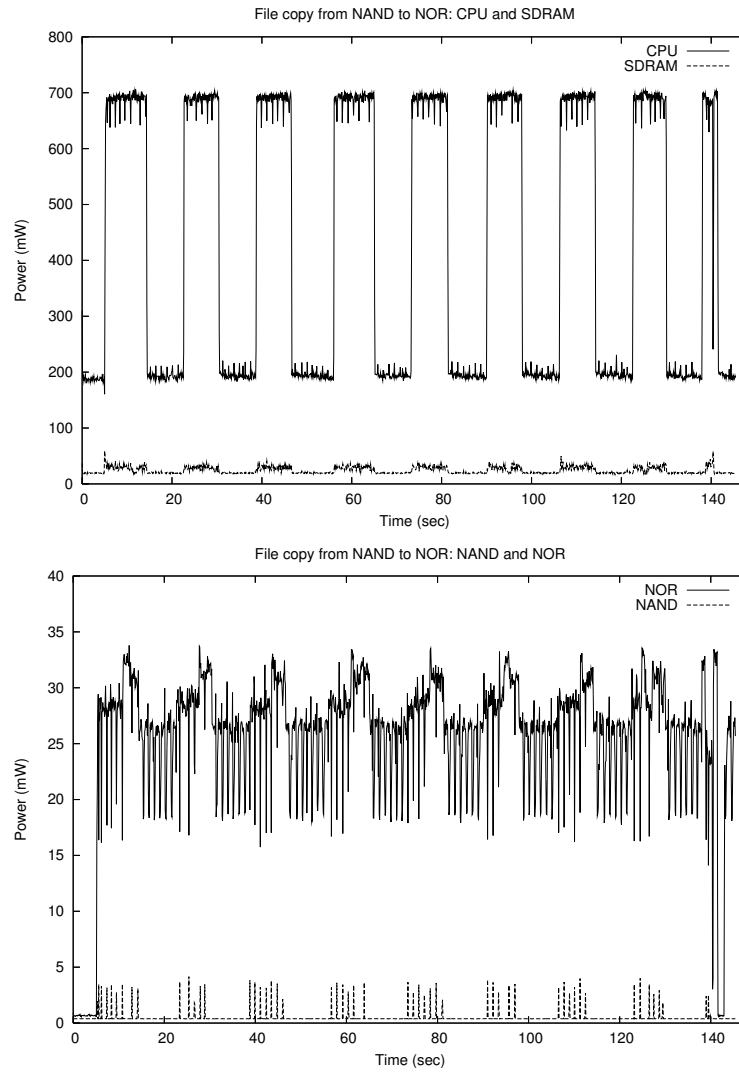


Figure 2.13: CPU, SDRAM, NAND and NOR power draw over time when copying a 7Mb file from NAND to NOR using JFFS2.

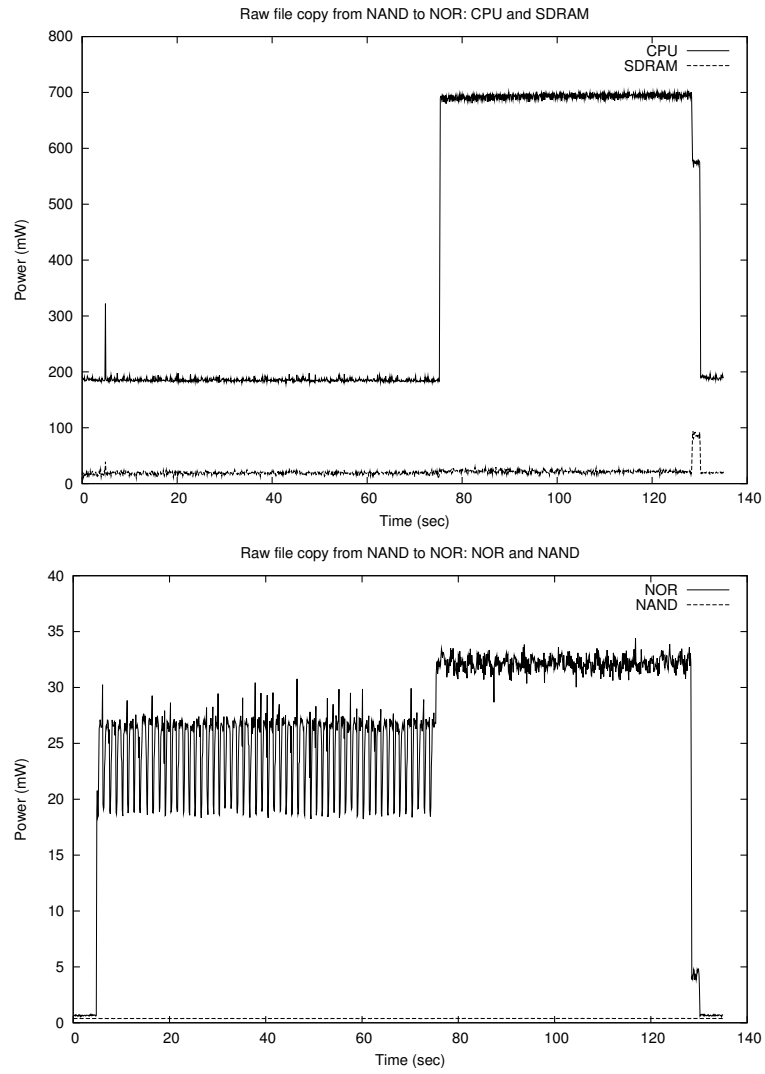


Figure 2.14: Power profile when writing directly from user space utilities.

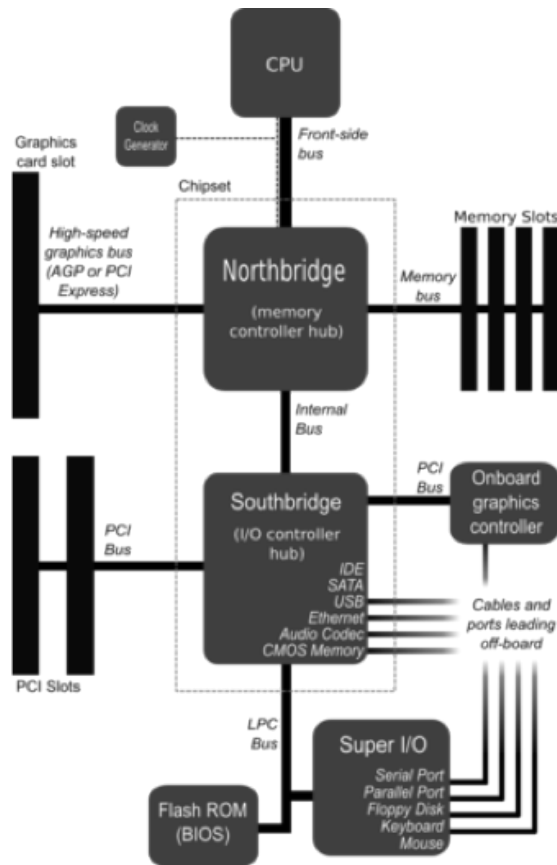


Figure 2.15: Typical PC hardware architecture [Wik07]

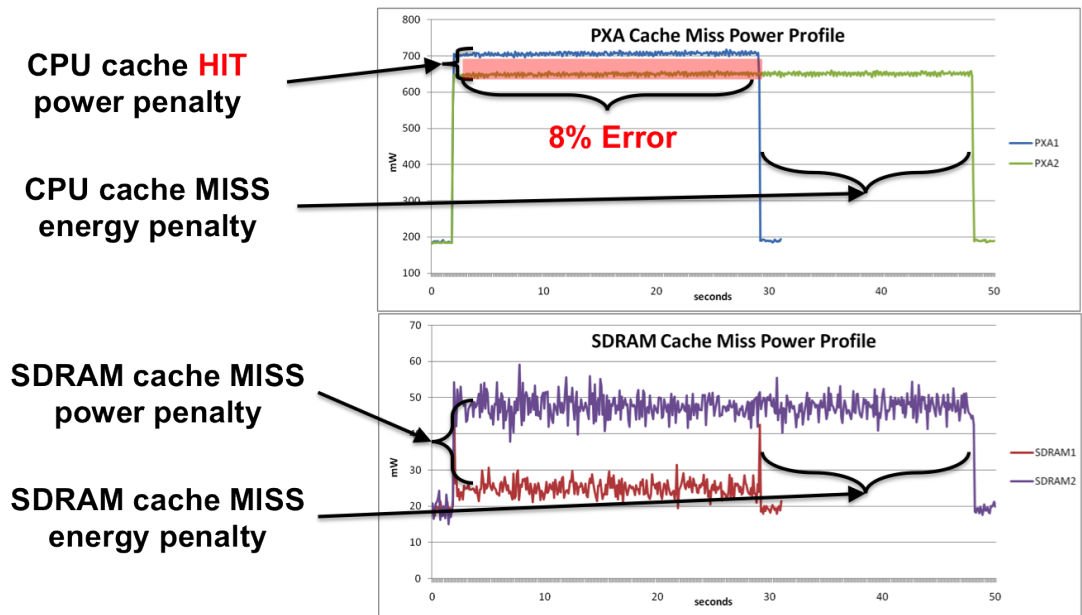


Figure 2.16: Energy correlation effects showing CPU power increase due to L1 cache hit

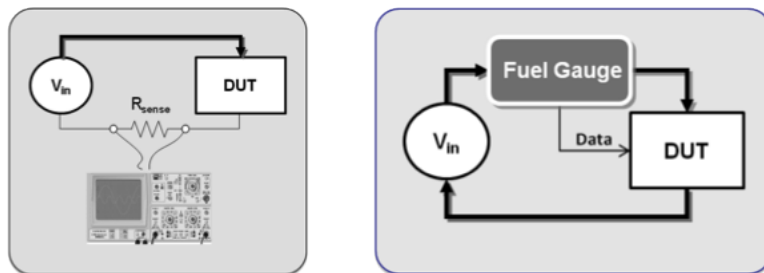


Figure 2.17: Traditional sampling methods: oscilloscope power profiling setup measuring the UUT (a) and fuel gauge circuit (b)

CHAPTER 3

A New Approach: LEAP

Having now described the needs of new ENS applications and detailing prior ENS design approaches to energy management we now discuss the specific design criteria of the LEAP architecture. We start by outlining our design goals for this new energy-aware architecture. Specifically we require the following architecture advancements in order to meet our energy-aware design criteria:

1. Creation of a new high resolution, in-situ power monitoring system.
2. Provide a constantly vigilant method yet require low overhead for operation.
3. Be consistent with platform's low power operation and not require significant additional energy resources to operate.
4. Provide fine grained, component or device level power measurement sufficient to attribute energy to specific functions within the system.
5. Utilize sampling rates to be consistent with accurately measuring all dynamic power features of the region under measurement. This may vary with the specific component being measured and its operating state.
6. Provide energy measurement across a large number of separate power domains. Each must be resolvable as an independent energy contribution to the system's whole.

7. Enable power collapse of idle resources while maintaining sufficient boundary isolation as to not prohibit normal operation of operational resources.

These architecture advancements can be used to enable a new *energy-aware application development* process as shown in Figure 3.1. The highlighted section in blue notes the enhanced design cycle including in-situ power measurements and visualization tools providing iterative development based upon energy dissipation data. We now detail how the noted design criteria flow into a design specification

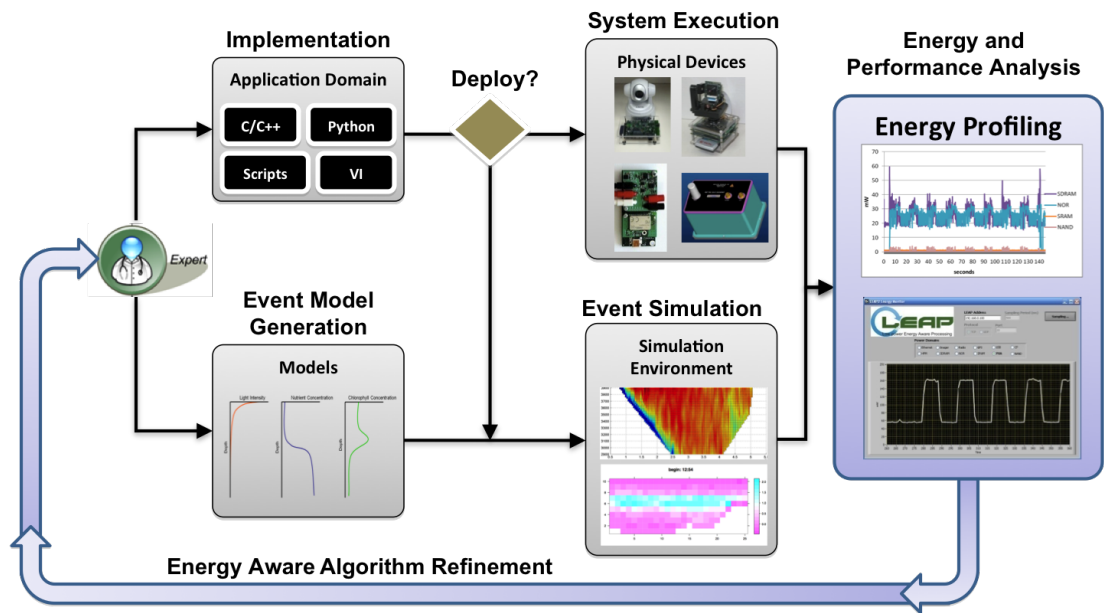


Figure 3.1: Applications development using a LEAP enabled design flow

for the LEAP architecture. The following sections provide specifics of both the software and hardware components for the LEAP architecture including design rationale and technical details. This is followed by a description a custom designed reference implementation of this architecture in the LEAP2 ENS platform. Our LEAP2 platform has served as both a test bench for validation of the LEAP principles as well as an enabler for development of energy-aware ENS applications including profiling utilities, networking protocols, and detection algorithms.

3.1 Hardware Enablement

In order to fulfill our stated objectives, we first assess the hardware requirements for in-situ energy measurement. A proper design must overcome a significant challenge to provide both a highly accurate measurement capability and a sophisticated accounting system all while maintaining very low power operation. Our measurement system is composed of several building blocks noted in Figure 3.2. Specifically, we require the following blocks: power sensors, power switches, analog-to-digital converters (ADC), and energy accounting processors.

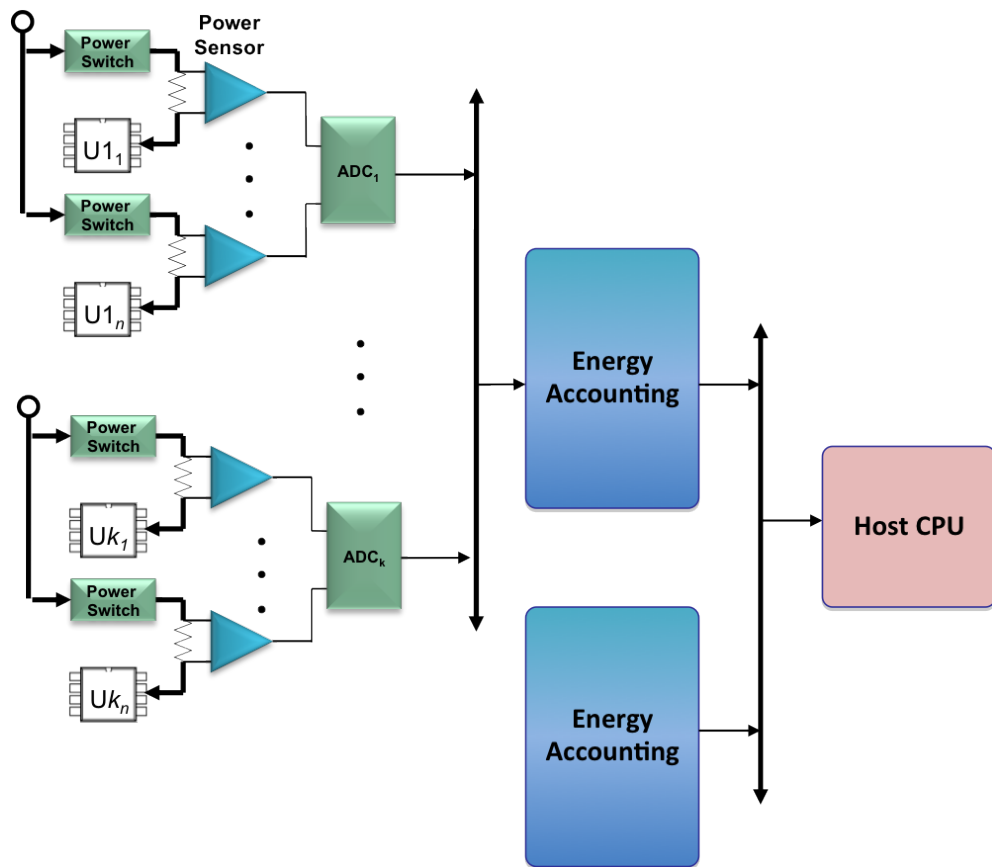


Figure 3.2: Energy-aware architecture components: power sensors, power switches, analog conversion, energy accounting

These building blocks form a naturally hierarchical design where each block may

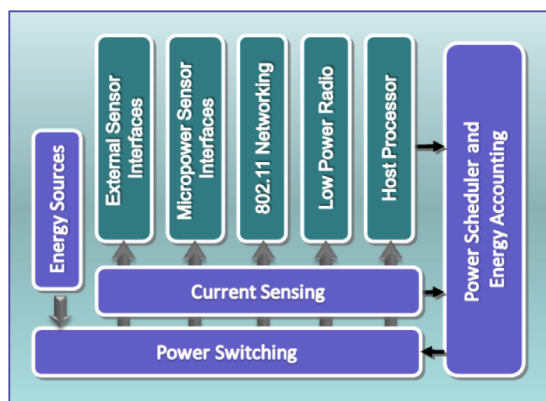


Figure 3.3: Typical set of ENS subsystems included the in-situ measurement in LEAP architecture

be replicated n times when connected into the next level block. For example, one or more power sensors may be aggregated into a single ADC as well as one or more ADC's may be connected to an energy accounting processor. The design for each building block may be homogenous across the entire system or custom building blocks may be used in a heterogeneous design where each building block may be optimized to the particular power domain. The system designer may choose from a variety of building blocks depending on the specific requirements. We now discuss the specific selection criteria for each of these building blocks.

3.1.1 Power Supply Characteristics

Before embarking on design of the individual building blocks for the in-situ measurement system, we first look at the characteristics of power usage in typical ENS electronics. A plot of the frequencies in our supply voltage and supply current can be used to determine the information bandwidth of the power supplies. A plot of the power spectral density (PSD) provides insight into the frequencies of interest. We first perform an experiment to measure the current on the CPU and SDRAM power supplies in a typical PC motherboard at high frequency—

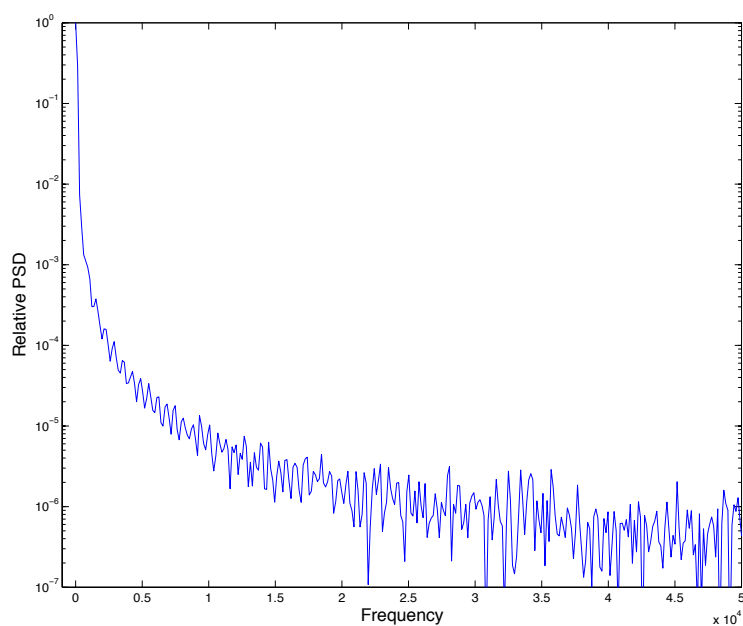


Figure 3.4: Power profile of Intel Core 2 Quad CPU Q6600 2.4GHz

5MSa/s. (It is assumed that the regulated supply voltages will maintain a small variation compared to its DC value, so that power is proportional to the supply current). The power profile of a high performance CPU, such as the Intel Core 2 Quad, operates with the highest frequency clock rates and contains complex internal clock gating functions. It thus likely contains the highest frequency components of any of the PC power supplies. The PSD of this CPU's power supply, measured in-situ while exercising dynamic test bench loading, is shown in Figure 3.4. As can be seen, 99% of the CPU supply's power is contained within the first 500Hz and 99.9% within the first 1kHz. All higher frequency components of the power signal have been attenuated by the bypass capacitors placed near each of the CPU power pins as well as the inductive properties of the PCB traces themselves as shown in Figure 3.5. Each of the building blocks constructed for our in-situ measurement system must comply with the maximum supply frequency

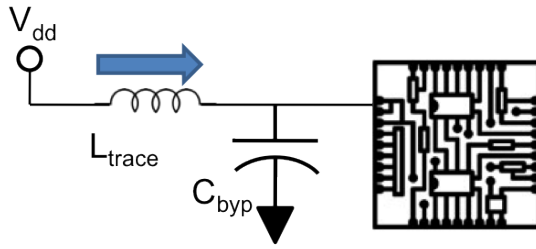


Figure 3.5: Power filter formed through bypass capacitors and trace inductance

expected for that power domain. Also per the Nyquist theorem, a minimum ADC sampling frequency of at least 2 times the maximum supply frequency can fully capture the signal and thus adequately meet the resolution requirement. Of course actual ADC sample frequency selection will depend upon any front-end filtering required for rejection of higher frequency aliasing components, but our maximum supply frequency provides the filter's cutoff point.

Having established the minimum signal bandwidth for our in-situ measurement system we next proceed with the design of each of the building blocks in that system.

3.1.2 Power Measurement Sensor

Our architecture first requires multiple sensors capable of accurately measuring instantaneous power dissipation across a wide dynamic range. This power measurement is most frequently performed by separately measuring instantaneous voltage $v(t)$ and current $i(t)$ to generate the product $P(t) = v(t)i(t)$. The voltage measurement component is commonly made by direct connection to the ADC input. (We address the measurement considerations for the ADC in the next section). The instantaneous current calculation is performed either through magnetic coupling or through a shut resistor placed in series with either the supply voltage (high-side) or the ground reference (low-side). Magnetic coupling

methods, while non-intrusive to the measurement circuit are restrictive due to the size, cost, and nonlinearity issues. While shunt resistors are ideal from cost and size perspectives but induce a power efficiency loss of the measured resource. The resistive method was chosen for our LEAP design though we later detail the power efficiency loss due to this technique. We also must consider whether to use

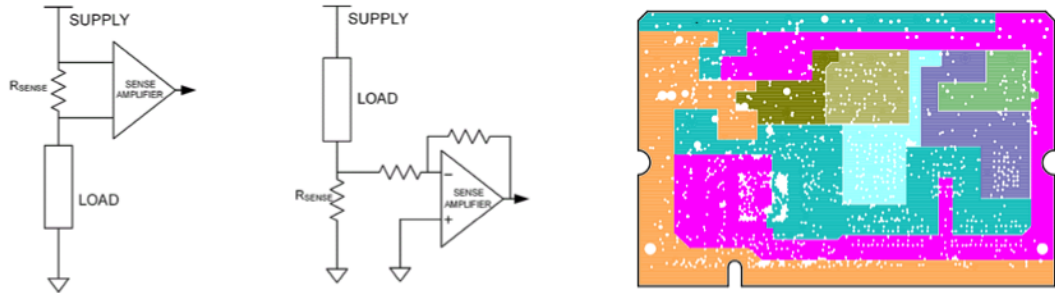


Figure 3.6: Resistive current sensor types a) high-side b) low-side and typical PCB power plane regions

the high-side or low-side resistive measurement method. A challenge when using low-side measurement is the insertion of the shunt resistor into the signal return path. This causes signal integrity issues for high frequency signals due to the disruption of the return path. Also, as part of our LEAP design requirements we require isolation of individual power domains, creating additional non-continuous supply planes. A low-side solution would then lack any low impedance, continuous return path and would cause signal integrity issues for data transmission. Figure 3.6 shows a typical LEAP enabled circuit board power plane with isolated supply voltages. We therefore select high-side measurement methods for LEAP in order to maintain a low impedance ground path and allow the supply planes to be isolated into regions dedicated to particular hardware resources.

In order to minimize power loss in the current sensing circuit, a small valued shunt resistor is used combined with a specialized *current-sense* amplifier circuit. In the high-side sensing configuration, the current-sense amplifier is low-power,

high gain circuit that is highly resistant to common mode noise. Additionally the current sensing amplifier must also provide accuracy across a wide dynamic range, since as part of our LEAP design philosophy we seek to provide a wide operating range for all system components. To maximize the sensor's dynamic range we choose a resistor value, R_s based upon the anticipated maximum current value, I_{max} , the maximum full scale voltage of the ADC V_{fs} , the gain value of the current amplifier, A_g , and its given input offset voltage, V_{off} . Using higher gain values, A_g , permits smaller sense resistors, R_s , at the expense of more error due to non-linearity of the input offset voltage, exacerbated by the high-side common mode signal. As the differential input voltage developed across the shunt resistor at the input of the amplifier decreases, the inherent input offset voltage becomes a larger percentage of the measured input signal, resulting in an increase in measurement error. The correct component choices are made by reviewing the tradeoffs among the various values for R_s and A_g based upon the given V_{fs} and V_{off} . We define the design limits to be set at a maximum permitted measurement error E_{max} , which occurs at the minimum expected measurement current I_{min} , and the maximum current sense output value defined by the V_{fs} of the ADC at I_{max} . The total measurement error E_{tot} is composed of several error components including using the root-sum-square (RSS) method.

$$E_{tot} = \sqrt{(E_{off}^2 + E_g^2)} \quad (3.1)$$

We now detail the selection of components using an example following the procedure used in development of the LEAP architecture. First components capable of operating of a wide dynamic range in power dissipation are selected. The dynamic range of many components spans several orders of magnitude. For our example implementation we assume a region of operation for a circuit that

operates in between P_{min} and P_{max} . We assume the region uses a typical supply voltage, which sets the maximum and minimum currents across the shunt resistor. We also select the same supply voltage for the ADC's V_{fs} to maximize the reference voltage range. While it would be possible to increase V_{fs} to reduce amplifier error, it is not a common feature of ADC components to allow V_{fs} greater than V_{dd} . Finally we select a maximum error E_{max} at the minimum input level as this gives comparable accuracy to existing off the shelf fuel gauge components. For our example we also select the Texas Instruments INA21x family of off the shelf current sense amplifier components based upon their low quiescent current, high common mode rejection, and low input offset voltage. The INA21x family data sheet states a quiescent current of 100uA, common mode rejection ratio of 120dB, input offset voltage of 5uA. The maximum error requirement sets the minimum sense voltage required at the input to the amplifier $V_{s,min}$ determined by P_{min} and R_s . Amplifier total error, E_{tot} , is set by the combination of V_{off} , common mode offset voltage V_{cm} , and gain error E_g as follows:

Selected Design Constants:

$$P_{min} = 1mW, P_{max} = 1W$$

$$V_{dd} = 3.3V$$

$$V_{fs} = V_{dd}$$

$$E_{max} = 10\%$$

INA21x amplifier component constants:

$$V_{off} = 5uV$$

$$E_g = 0.02\%$$

$$CMRR = 120dB$$

$$I_q = 100\mu A$$

Common mode voltage is then:

$$V_{cm} = \frac{1}{10^{\frac{CMRR}{20}}} \cdot V_{dd} \quad (3.2)$$

$$V_{err} = \sqrt{V_{io}^2 + V_{cm}^2} \quad (3.3)$$

$$E_{io} = \sqrt{E_{max}^2 - E_g^2} \quad (3.4)$$

$$V_{s,min} = \frac{V_{err} \cdot 100}{E_{io}} = I_{min} \cdot R_s = \frac{P_{min}}{V_{dd}} \cdot R_s \quad (3.5)$$

Then we get:

$$V_{cm} = 3.3\mu V$$

$$V_{err} = 5.99\mu V$$

$$V_{s,min} = 599\mu V$$

$$R_s = 0.2\text{ohms}$$

The maximum gain value is now defined using V_{fs} , R_s , and P_{max} . We desire the output of the amplifier to be at the ADC's full scale setting at the point the sensed load reaches its maximum current.

$$A_{g,max} = \frac{V_{fs}}{R_s \cdot \frac{P_{max}}{V_{dd}}} \quad (3.6)$$

Since we set V_{fs} to be the same as V_{dd} this becomes:

$$A_{g,max} = \frac{1}{R_s \cdot P_{max}} = 5 \quad (3.7)$$

The final step is to calculate the power overhead required to operate the circuit. This is found as the ratio of the power used by the load P_{load} to the power used by the measurement circuit composed of sum of the power used in the sense amplifier P_{sense} and the power used in the shunt resistor P_{shunt} . We find this for both the maximum and minimum loads and derive the circuit efficiency as follows:

$$Efficiency = 1 - \frac{P_{sense} + P_{shunt}}{1 + P_{load}} \quad (3.8)$$

Where the power at the load, P_{load} , is in the range $\{P_{min}, P_{max}\}$. The power dissipated in the sensing amplifier circuit is just:

$$P_{sense} = V_{dd} \cdot I_q \quad (3.9)$$

The power dissipated in the shunt resistor is found from the relationship

$$P_{shunt} = I_{shunt}^2 \cdot R_s = \left(\frac{\{P_{min}, P_{max}\}}{V_{dd}}\right)^2 \cdot R_s \quad (3.10)$$

For the current example we find the efficiency $Eff_{min} = 75\%$ and $Eff_{max} = 98\%$ at the operating points $\{P_{min}, P_{max}\}$ set to $1mW, 1000mW$. The above example can be repeated for find the best R_s and A_g for each power domain in the design using the estimated P_{min} and P_{max} for each of the platform's power domains. Table 3.1 shows the calculated values for several power domains measured on the LEAP2 reference platform.

Power Domain	Power Range	V_{dd}	R_s	A_g	Efficiency
HPM -Processor	25mW-1.5W	0.8-1.1V	0.1	24	0.99
HPM -SDRAM	10mW-500mW	1.8V	0.2	60	0.98-0.99
Ethernet	1mW-300mW	3.3V	0.2	180	0.75-0.99
GPS	1mW-200mW	3.3V	0.47	115	0.75-0.99
SDIO	.1mW-100mW	3.3V	0.47	230	0.23-0.99

Table 3.1: Power Domains with operating range, Sense Resistors, Amplifier Gain, and Measurement Efficiency

3.1.3 Data Conversion

Having described the requirements for our power sensor design we now address digitization of the power sensor's analog signals using an analog to digital converter (ADC). The continuous time power calculation $P(t) = v(t)i(t)$ is frequently measured as the discrete time equivalent using ADC sampling. The energy consumed over a given period measured by an ADC can be calculated using a summation of the products of ADC samples for the current and voltage signals. The sums of the products correspond to the incremental dissipated energy values and can be calculated using Equations 3.11.

$$\Delta E = \sum_0^t (K_i \cdot I(t) \cdot K_v \cdot V(t)) \quad (3.11)$$

$$K_i = \frac{V_{ADC,range}}{(2^B - 1) \cdot A_i \cdot R_{sense,i} \cdot f_{sample}} \quad (3.12)$$

$$K_v = \frac{V_{ADC,range}}{(2^B - 1) \cdot A_v \cdot f_{sample}} \quad (3.13)$$

where,

$V_{ADC,range}$ = input dynamic range of ADC (V)

B = number of bits available in ADC

A_v = voltage gain of the ADC amplifying stage (V/V)

R_{sense} = current-sense resistor (Ω)
 f_{sample} = sampling frequency of ADC (Hz)

This Since K_i and K_v are typically constants in the design, the energy measurement distills into a single multiply and accumulate (MAC) operation per cycle per channel. However, since most systems employ voltage regulators at the supply, the $V(t)$ measurement can be replaced with a simple constant V_{supply} for each power domain. The equation drops to a simple accumulation operation (charge measurement) with constant scaling factors applied at the end as shown in Equation 3.14. However with the use of digital voltage scaling techniques, we cannot make this assumption and must revert to methods consistent with Equation 3.11.

$$\Delta E = \Delta Q \cdot V_{supply} \quad (3.14)$$

The proper ADC selection involves considering multiple design properties simultaneously including maximum sampling rate, sample data word size/resolution, number of input channels, and data interface type. Each property has direct impact on the ADC's power dissipation and must be selected with careful consideration of its implications on both ADC and total system power use. For example increasing the ADC sample rate not only increases the power consumed for sample conversion, but also implies increased power dissipation for data transfer out of the ADC sample buffer as well as additional power require for data processing.

Most modern ADC's also implement power down capabilities so that the ADC can be put into low power sleep states between conversion cycles. Here the ADC power consumption is a combination of sleep mode power P_{sleep} and active mode

power P_{on} . Equation 3.15 shows the amount of time the ADC spends in active mode is a function of the ADC's sample clock rate. Normally we would seek to reduce the sample clock rate to match the minimum acceptable value in order to reduce the power dissipated by the clock generator. However, in the case of the ADC this may actually increase the total ADC power. Since the ADC requires a constant number of sample clock cycles to generate the conversion result, slowing the clock extends the time required to generate the output sample. Thus there will be additional power consumption due to the added time spent out of the low power sleep state due to the extended sample conversion. Equation 3.16 shows how average power relates to the particular ADC settings for sample rate f_s , sleep power P_{sleep} , ADC sample width in bits N_b , power up transition time t_{pu} , and sample clock f_{clk} .

$$t_{on} = (f_s \cdot (t_{pu} + f_{clk} \cdot N_b)) \quad (3.15)$$

$$P_{ave} = P_{on}(f_{clk}) \cdot t_{on} + P_{sleep}(1 - t_{on}) \quad (3.16)$$

As an example, Figure 3.7 shows the average power for a Texas Instruments AD7888 over a wide range of sample rates. As can be seen, significant power savings in the ADC are possible by accurately matching the sample rate to the power supply's information bandwidth.

In addition to proper sample rate selection, ADC resolution also has significant implications on power. Selecting a sample size beyond the fidelity of the power sensor inputs provides no additional information at the expense of additional conversion power, data word transfer power, and computation power. The correct choice defines the effective number of bits (ENOB) to match the expected error margin set during the specification of the power sensor noted in a previous section. Likewise, careful attention to the ADC data interface is necessary to 1) assure

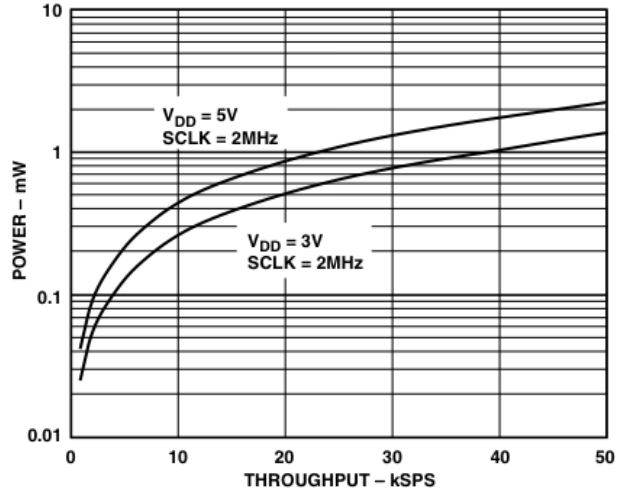


Figure 3.7: ADS7888 ADC power consumption over a range of sampling frequencies

sufficient bandwidth for the selected data rate, resolution and number of channels and 2) prevent the selected data interface from imposing significant overhead necessary for to data collection. We first verify that the ADC’s data interface can handle the required aggregate data rate by combining the parameters to generate the total required data throughput, BW_{tot} , including any data addressing or framing overhead:

$$BW_{tot} = f_s \cdot t_{on} N_b \cdot t_{on} N_c + BW_{overhead}(f_s, N_c) \quad (3.17)$$

It is important that the aggregate bandwidth should not oversubscribe the underlying data bus. As an example using common ADC values, $f_s=1\text{KHz}$, $N_b=12$, $N_c=8$ the aggregate data bandwidth is calculated as 96Kbps. A low speed I²C bus operates with a maximum data rate of 100kbps, which should provide sufficient throughput to support the design. However when considering the bus overhead needed for register addressing and 8-bit register alignment the $BW_{overhead}$ can be 1.5 bytes per sample, making the total bandwidth required BW_{tot} of 132Kpbs

Characteristics	Influences
I_q	Since this is always powered, this value must be significantly lower than the minimum expected load of the region being supplied
I_{max}	Must be able to supply maximum expected load current for the region being supplied
V_{min}, V_{max}	Must be able to operate over the expected voltage supply range
I_{dis}	Should be able to discharge regions in power down state to prevent floating pins and creating discharge paths from neighboring supplies

Table 3.2: Power Switching Considerations

exceed the maximum bus bandwidth. The overhead necessary to operate the ADC and transfer the sample significantly increases the necessary bandwidth and cannot be neglected during architecture selection.

3.1.4 Power Switching

As part of the LEAP design criteria we not only need a method to measure power across a wide variety of system components, but also provide a method to enable powering down unused or idle subsystems. While much advancement in integrated circuit design have focused on on-chip power management capabilities, including power down of large regions of the die, we still require a method to collapse the supplies on larger regions of the system including entire circuit boards. This necessitates a dedicated switching component to enable dynamic power control over these regions. The important factors to consider when selecting these components include the quiescent current, maximum load, input voltage range, and load discharge. These are described in Table 3.2.

In addition to power switch component selection, it is critical to perform detailed analysis of the design at all boundary nodes between different voltage

domains. Many circuits require dedicated isolation circuitry to prevent current flow through the signal pins via the protection diodes. These act as parasitic diodes and can cause significant power consumption or damage in the unpowered circuits. Isolation circuits such as cross-bar bus switches [Ins] should be used to prevent these parasitic diodes where necessary to prevent these paths in Figure 3.8.

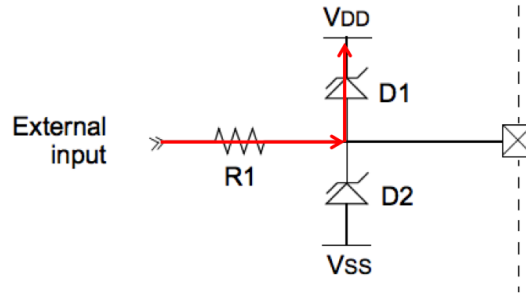


Figure 3.8: Parasitic diode affect when driving I/O from a powered domain to an unpowered domain ($V_{in} > V_{dd}$)

3.1.5 Energy Accounting

Having described the components necessary to generate proper data samples for accurate power measurement within each of the system’s voltage domains, we now address the design and implementation of the components required for data sample acquisition, processing, and host processor data transfer. The aggregate functionality of these components we describe as *energy accounting*. In order to provide high fidelity energy accounting information in an efficient manner we must address the proper architecture for the energy accounting subsystem. We base our experience selecting the proper energy accounting architecture upon lessons learned from prior generation LEAP enabled implementations [MSR12] where several design shortcomings were found. This early implementation was based upon a low power MSP430 microcontroller operated as the coordinator of the

energy management system, and connected to the main applications processor via an I^2C hardware bus as shown in Figure 3.9. During extensive experimentation this platform, including several classroom projects of the graduate course EE209, we discovered the the following issues with the original implementation:

1. The energy accounting function dominates the power required to implement LEAP and thus must be implemented most efficiently. This dictates the use of specialized hardware designed for this purpose.
2. The energy accounting function must not require intervention from external resources such as the applications processor to function. It must be able to measure and calculate power and accumulated energy values across multiple channels simultaneously and with high sample rates.
3. The energy accounting function must be able to communicate with the applications processor via a high speed bus. This is required to minimize the overhead needed to transfer energy values to the applications processor during operating system scheduling operations.
4. The energy accounting function must be able to perform automatic resets of energy values independently on each of its channels. This is necessary for the operating system to attribute different time intervals with specific software applications with minimum overhead.

Solutions to these issues led to a custom hardware design based on a custom designed ASIC solution, currently implemented in FPGA logic for cost reasons. This ASIC design was named the *energy management and accounting processor* (EMAP). We now detail this custom hardware design as well as the performance results obtained during evaluation.

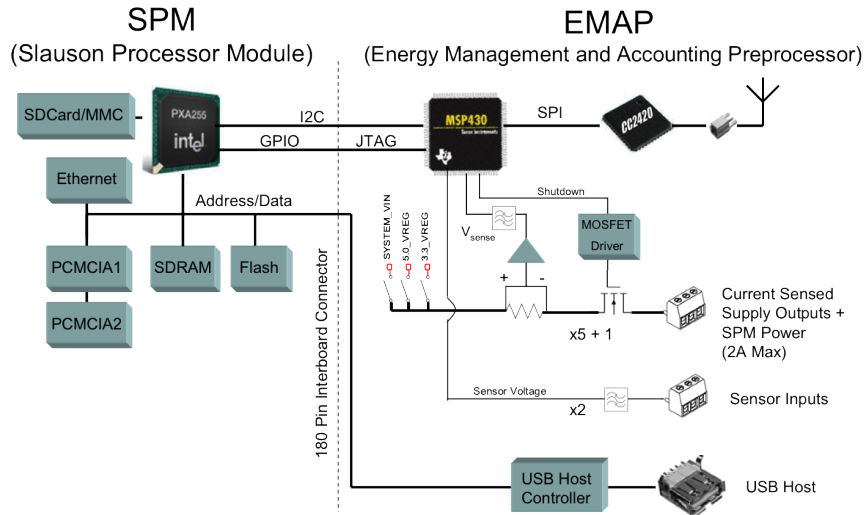


Figure 3.9: Original LEAP platform hardware architecture

3.1.5.1 EMAP ASIC

As shown in Figure 3.10, the EMAP ASIC consists of a number of top level subsystems which are all connected via a unified system on chip (SoC) backbone bus. This backbone is provided by the open source Wishbone interconnect [Ope12a] revision B4 implemented as a multiplexor bus logic interconnect. The Wishbone bus provides a low logic count yet high speed interconnect between each of the modules in the EMAP design. This includes support for all single cycle commands as well as block read from energy accounting module to enhance throughput during energy sampling events. The bus provides a 32-bit data path and a maximum burst throughput for sequential reads of 4 bytes per clock cycle. To improve access time for cycles initiated from the applications processor, each of the modules support partial address decoding at the Wishbone interconnect and full address decoding within each of the module blocks.

The EMAP's Wishbone interconnect includes two masters (the Host Interface Controller and Energy Accounting Manager), so a bus arbiter function is used

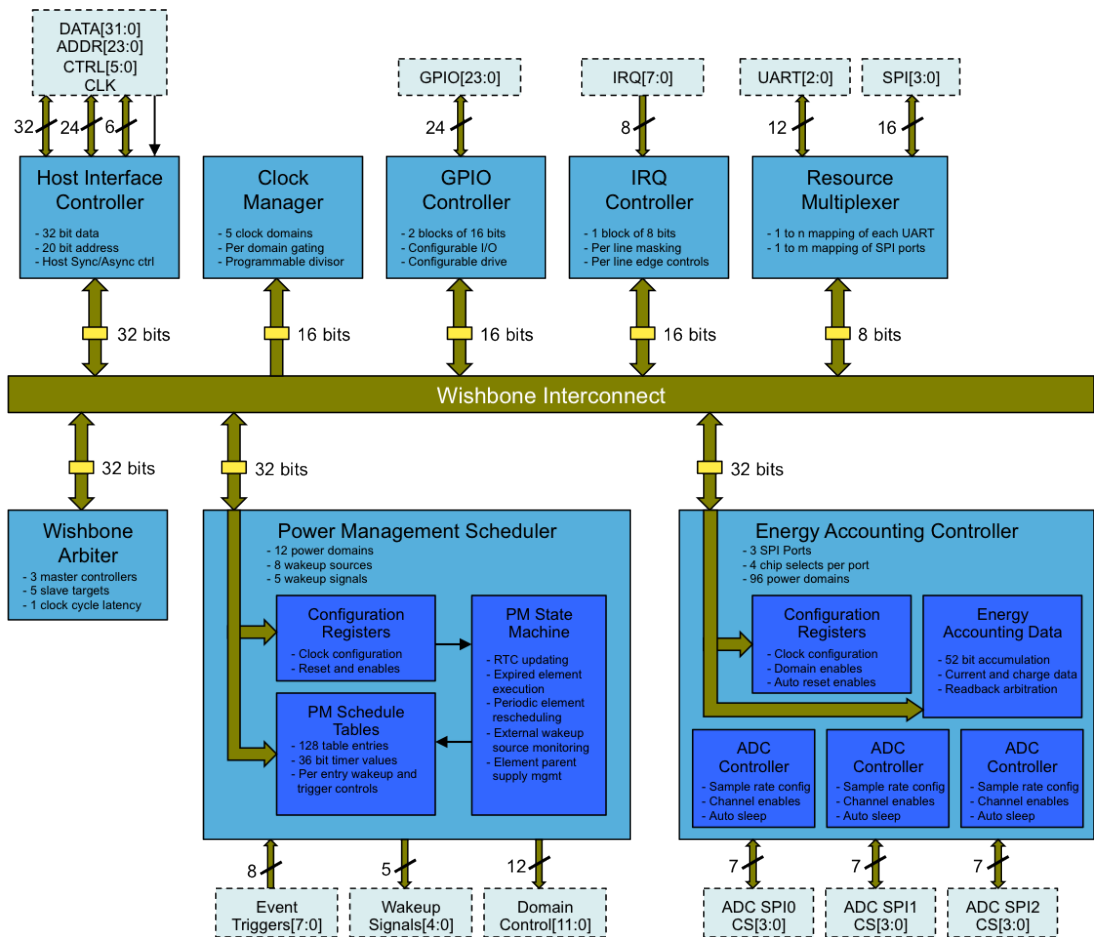


Figure 3.10: EMAP ASIC architecture block diagram with bus interconnect

Wishbone Address	Module
0x000000 - 0x000010	System Configuration
0x000010 - 0x000100	Clock Manager
0x000100 - 0x001000	Interrupt Controller
0x001000 - 0x001100	GPIO Controller
0x002000 - 0x008000	Energy Accounting Manager
0x008000 - 0x010000	Power Management Scheduler
0x010000 - 0x110000	Host Interface Controller

Table 3.3: EMAP Wishbone SoC bus addresses

to prevent bus deadlock. The Wishbone arbiter utilizes a fixed priority method with highest priority given to the Host Interface Controller for lowest latency register access for the applications processor. Additionally, the arbiter utilizes an arbitration parking scheme, where the current bus master retains bus ownership during idle periods in order to reduce latency for burst transactions from the same bus master.

The main functional blocks attached to the EMAP’s Wishbone interconnect include a Clock Manager, GPIO Controller, Programmable Interrupt Controller, Resource Multiplexer, Energy Accounting Manager, and Power Management Scheduler. Each of these EMAP subsystems, as shown in Figure 3.9, is accessed via the Wishbone bus. The Wishbone addresses for each block are noted in Table 3.3. A complete list of register addresses is found in Appendix A. The following sections describe the operation of each subsystem in detail.

Clock Manager The Clock Manager module is responsible for generation of the different clock domains operating within the EMAP ASIC as well as provide clock gating functions to reduce power consumption for modules that are placed into the sleep state. Each clock domain frequency may be derived from the primary clock input as either a basic clock divider to provide $f_i = f_{clk}/N_i$ or into

Clock Domain	Maximum Frequency
Wishbone Clock	34.6MHz
Host Interface Clock	48.5MHz
Energy Accounting (CAC) Clock	26.9MHz
ADC Sampling Clock	16.4MHz
Power Management Scheduler Clock	12.2MHz

Table 3.4: Clock Manager domain frequency bounds for an Actel IGLOO FPGA derived from place and route timing reports

a phase locked loop (PLL) macrocell for general clock multiplier capabilities given $f_{dom} = (f_{clk} \cdot M_i)/(N_i \cdot C)$. The clock domain's output frequency is configured individually via a control register which sets the division factor and optionally enables a PLL for clock multiplication. The maximum clock rate for each domain is defined during logic synthesis for the target FPGA architecture and speed grade. Example frequency bounds for the clock division and PLL multiplier are provided for an implementation in an Actel IGLOO FPGA architecture in Table 3.4. In order to mitigate metastable conditions between each clock domain a synchronizer delay circuit is used for signals passing between clock domain boundaries.

GPIO Controller A general purpose input/output (GPIO) controller core is provided to enable low complexity interconnect to a wide array of sensor types. Each GPIO may be individually configured as an input or output with the output drive strength setting controlled by the pin configuration during place and route. The output driver may be set to either a push-pull or open-collector architecture. To alleviate the need for locking when performing read-modify-write operations, all GPIO output values may be set using individual set and clear registers. Finally, GPIO inputs may be routed to the programmable interrupt controller for mapping to one of the interrupt pins to the applications processor.

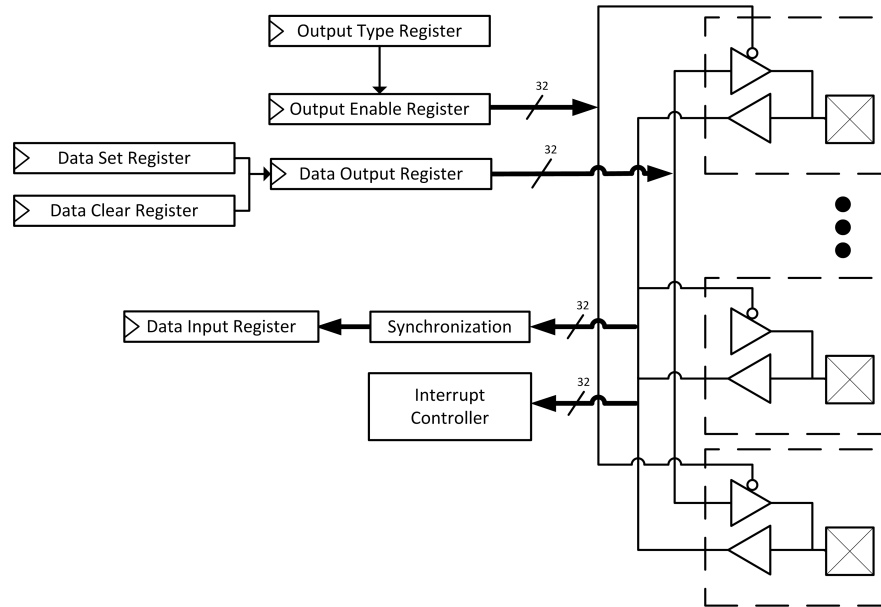


Figure 3.11: Schematic of GPIO pin logic

Figure 3.11 demonstrates the control logic implemented for each GPIO pin and module register memory map.

Programmable Interrupt Controller In order to provide a highly configurable interrupt request (IRQ) capability for the host applications processor, the EMAP includes a programmable interrupt controller (PIC) subsystem. The PIC is responsible for routing the large number of possible input interrupt sources to the reduced set of applications processor IRQ input lines. This routing is controlled through the PIC's IRQ configuration registers. Each IRQ includes input sources from other EMAP subsystems including the GPIO Controller, Power Management Scheduler, and Energy Accounting Manager. The PIC additionally provides global interrupt masking per individual IRQ line to simplify software locking needed for interrupt service routines. To further enhance compatibility with external interrupt sources, the input lines routed from the GPIO controller may be configured as either edge or level triggered. Additionally, edge triggered

interrupts may be configured as sensitive to rising edge, falling edge, or both edge transitions. Figure 3.12 demonstrates the interrupt controller logic.

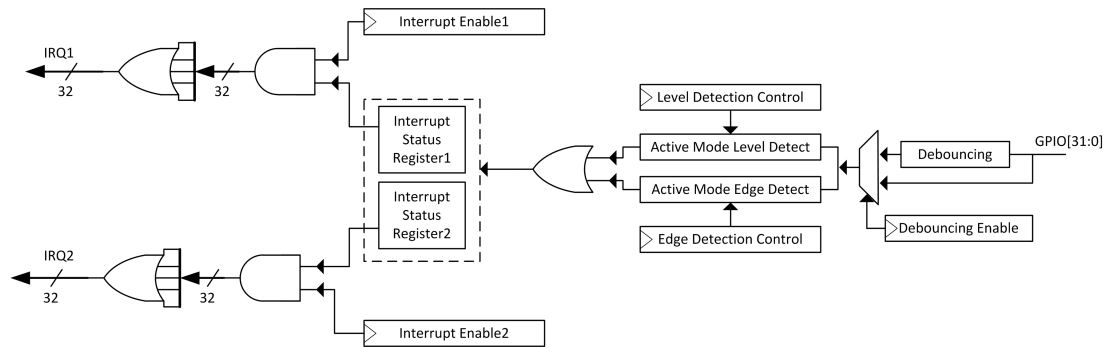


Figure 3.12: Programmable interrupt controller logic

Host Interface Controller Data accesses between the applications processor and the EMAP's internal Wishbone bus fabric are managed by the Host Interface Controller (HIC) bridge. The HIC bridge provides a host bus interface that may be configured to mate with a variety of applications processor architectures. Both synchronous and asynchronous bus types are supported via configurable data strobe lines and an optional bus clock line. The default configuration set by boot-strap pins that are latched during reset. Data access is byte masked with a maximum data width of 32-bits per cycle. Since the internal Wishbone bus is operated on a separate clock domain, the HIC bridge uses a host ready signal to delay bus transactions until the HIC is able win arbitration on the Wishbone and perform the requested transaction. This stalls the bus operation on the applications processors memory bus until the Wishbone can complete the bus cycle within the FPGA. Once completed the HCI releases the external memory bus to acknowledge completion. Example memory bus transactions to the Wishbone backbone are shown in Figure 3.13 demonstrating a read cycle when the Wishbone arbitration has been won and secondly when the bus is stalled due

to arbitration lost. The worst-case latencies are demonstrated here as well.

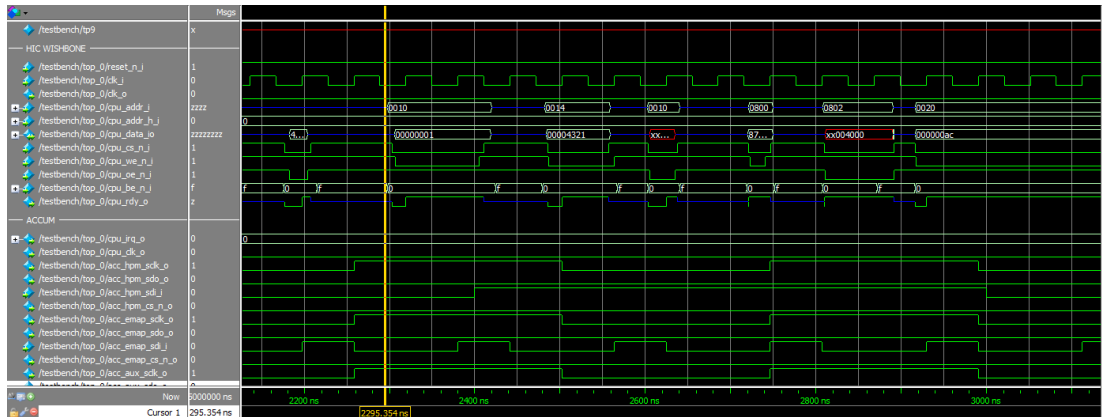


Figure 3.13: Wishbone to Host Controller read cycle waveforms

In order to minimize the number of CPU cycles required for data collection from the Energy Accounting Manager (EAM), the HCI also support a bus mastering direct memory access (DMA) cycle. In this operation, the applications processor configures the Energy Accounting Controller with the target memory address of the external memory bus attached to the HCI. The Energy Accounting Controller then arbitrates for ownership of the Wishbone bus. Once this is achieved, the HCI's Wishbone slave is enabled, which then arbitrates for ownership of the external memory bus via a dedicated request/grant (DMAREQ, DMAGNT) handshake signal pair with the applications processor. The Wishbone write cycle originated from the EAC stalls until the applications processor awards ownership of the external memory bus to the EMAP and the HCI target is able to complete its write cycle. If a block write cycle is indicated (by setting a DMA transfer size greater than one), the EMAP's Energy Accounting Controller will maintain ownership of the Wishbone and the HCI target address. This holds the HCI DMAREQ line asserted during following write cycles so long as the DMAGNT signal remains asserted. Should the applications processor interrupt the write progress, the HCI will again stall the Wishbone and retry the DMA

request cycle. This occurs until the entire data block transfer is complete.

Resource Multiplexer Due to the pin limitations imposed by the small IC packages common in modern applications processors, it is common to find a shortage of serial peripherals available from the application processor to connect large numbers of sensors, especially for non-multiplexed bus types (e.g., UART and SPI). This is exacerbated when the combination of required sensors and types are only known at run-time and therefore allocation of bus resources is problematic. In addition many of these sensors are only intermittently required. For example GPS, typically occupying a processor UART interface, is only necessary when acquiring position updates, but could be reused for other purposes during idle periods. To simplify this mapping of sensor serial buses to interfaces on the applications processor, the EMAP ASIC includes a Resource Multiplexor module. The Resource Multiplexor expands the number of available serial buses and connects these in a configurable crossbar switch configuration. This provides run-time management capability for the various peripheral buses by allowing 1-to-N connections between the host processor and external sensors. This ability to bypass and combine ports in many combinations simplifies design-time interconnect since over provisioning of the serial bus ports at the applications processor can be eliminated.

Energy Accounting Manager The design and implementation of the Energy Accounting Manager (EAM) is one of the major contributions of this thesis. The purpose of the EAM is to provide autonomous collection of high resolution energy information without necessitating intervention from higher power resources such as the applications processor. By enabling high-fidelity energy measurement across a large number of power domains, yet maintaining minimal external

management requirements and a low power operation, the EAM is a notable improvement from prior systems, as these are designed for power profiling only during development or debugging where inefficient measurement methods are not as critical. We describe his implementation and its enhancements to energy measurement here.

The EAM architecture consists of three main functional units, the configuration and control registers, the charge accumulation data RAM and controller, and the analog to digital sampling controllers). The configuration and control registers are Wishbone accessible read/write registers responsible for EAM operation. The memory map for the EAM is shown in Table 3.5. The applications processor typically configures EAM functions by writing to these registers during initial setup. Common operations include setting the master sample rate based upon the frequency of the sampling clock provided by the Clock Manager described above, enabling sampling for each of the active channels, setting whether the channel is a current or voltage measurement, and setting each channels sample rate divisor (which may be $1/N$ of the master sample period). For each channel that is configured as a current measurement type, it is possible to additionally assign a one of the configured voltage channels in order to perform sample by sample energy calculations. This is important since charge and energy are not proportional measurements in DVFS regulated domains as the voltage varies dynamically during operation. This is compounded by modern mobile processors who operate with automatic voltage scaling governors that are not set by software methods [CG11]. Accurate energy calculation requires instantaneous measurements of both voltage and current.

In addition to initial setup configuration, the applications processor may elect to enable a DMA method to push samples directly to its memory. To support this,

Wishbone Address	Module
0x002000	Version Register
0x002004	Configuration Register
0x002008	Charge Sample Clock Divider
0x00200C - 0x002018	Charge Enable Register [2:0]
0x00201C - 0x002024	Channel Configuration[2:0]
0x002028 - 0x002030	Channel Sample Clock [2:0]
0x002800 - 0x0028FF	Voltage/Current RAM
0x002900 - 0x0031FF	Reserved
0x003200 - 0x0033FF	Channel Count RAM
0x003400 - 0x0035FF	Channel Charge RAM
0x003600 - 0x0037FF	Channel Charge+RESET RAM

Table 3.5: EMAP Energy Accounting Controller Registers

the EAM's configuration registers include a DMA target address, transfer size, and transfer-enable register. Once configured, the applications processor may set the transfer enable register to start a data transfer from the EAM's block RAM directly to the applications processor as described in the HIC section. After configuring the proper information for each sampling channel, the applications processor sets global enable bit to start data collection. This begins the state machine within the charge accumulation controller (CAC).

The CAC first initializes its block RAM to zero to prepare it for storage of charge and voltage data. After initialization completes, the CAC waits for read access requests from the Wishbone bus or write requests for storing new samples from each of the ADC sampling controllers (ASC). An arbiter is used to prioritize simultaneous requests to the RAM with the Wishbone read given highest priority and each of the ASC's are given equal priority with round-robin sequencing. During Wishbone read cycles, an ASC with a pending write request will stall until the read cycle completes. If one or more ASC's indicate a new sample is ready, the CAC will grant write access to the highest priority ASC and store the available sample into the block RAM at the proper RAM address for the

sample data. The CAC additionally provides sample conversion according to the state diagram shown in Figure 3.14. For charge measurement channels, samples are first accumulated with the existing RAM contents and then written back to the RAM as charge measurements. For these channels a counter register is also incremented, providing the number of accumulations since the channel was last reset. Accumulation data is stored as 53 bit values and accumulation counts are 36 bit values providing a minimum of 2 years between rollovers when sampling at 1KHz with a 12-bit ADC (additional rollover periods are listed in Table 3.6). Additionally for energy measurement channels (those with an associated voltage channel), data samples are first multiplied with the latest voltage measurement, then summed with the current RAM contents, and then written back to as energy values. The voltage channels are not accumulated nor do they keep a summary count, and instead simply replace the prior value in RAM on each update.

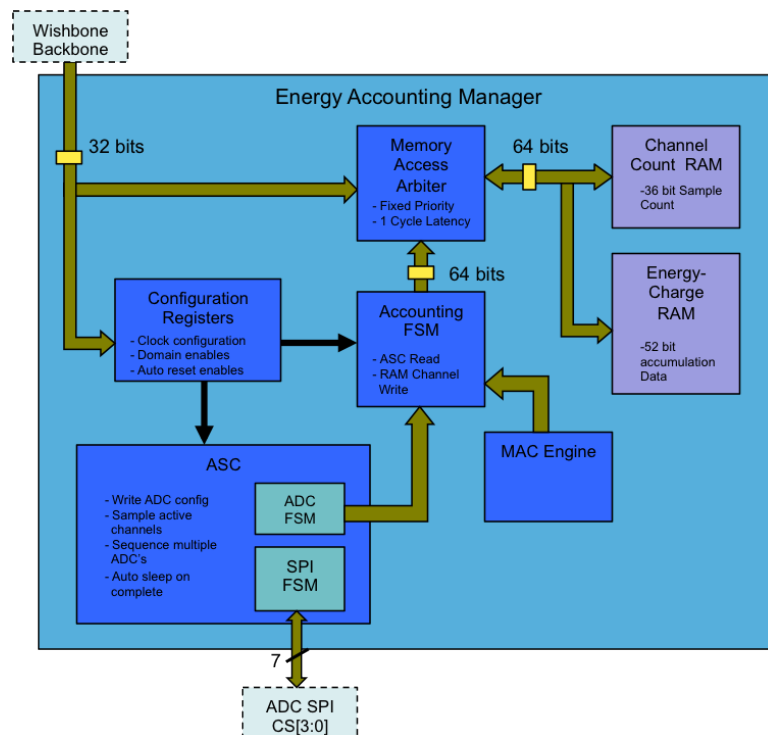


Figure 3.14: Data collection state machine showing energy measurement method

ADC resolution	Sample Rate	Rollover Period (data)	Rollover Period (count)
8-bit	100sps	1000+ years	21 years
12-bit	1Ksps	69 years	2.2 years
16-bit	10Ksps	159 days	79 days

Table 3.6: CAC rollover periods for data width and sample rates

To support autonomous operation during extended power states, each channel in the CAC RAM is sized to prevent rollover for long periods of operation. This is important since the applications processor must service the EAM to read back energy data within the rollover period to prevent accidental data loss. If the rollover period is set too short, additional energy is wasted due to extra servicing from the applications processor.

In order to do effective energy attribution to various software functions running on the applications processor, the CAC provides an efficient method to read incremental energy data. This is useful for example to provide the CPU energy dissipated between scheduler ticks. In a typical RAM based implementation, the applications processor would first read samples from the RAM block and then perform a separate write to each channel. The EMAP instead uses a unique 'read with auto reset' capability to simplify this operation. Each channel within the RAM block is aliased to multiple address locations as shown in Table 3.5. For normal read operations, RAM data is accessed from its base address. If the same channel is read from its auto-reset aliased address (the base address with alias offset), the RAM block location for this channel will automatically be reset to zero upon completion of the read cycle. This simplifies application software to perform a single read operation during energy attribution functions, which eliminates software race conditions inherent read-modify-write operations and reduces access time by half.

The final subsystem within the EAM is the ADC sampling controller. The

ASC's are responsible for initialization of the external ADC's, reading sample data from each ADC, and writing each sample to the CAC. The ASC design is modular to support different ADC components (e.g TLV2548 vs ADS7844). This feature is necessary to allow component variations in sample size, sample rate, and interface signaling. The ASC's state machine also manages any unique power management features of the external ADC, for example power down of voltage references during idle periods.

The Energy Accounting Manager typically has several instances of the ASC module instantiated, with each ASC module supporting one or more different external ADC components. The ASC is able to support several external ADC's on a single SPI bus via different chip select lines. The ASC automatically multiplexes operation of the different ADC's to support a larger number of logical channels to the CAC. The maximum sample rate, sample size, and SPI clock rate dictate the number of ADC components that can be connected to a single ASC. For example the ASC implementation for the TI TLV2548 supports 4 ADC's with 8 channels on each ADC, thus presenting 32 channels to the CAC. Additional channels are supported by replicating additional ASC modules within the EAM. The CAC arbitration logic supports cascading additional ASC modules as necessary to expand the channel count.

Power Management Scheduler A second significant contribution of this thesis is the design and implementation of the Power Management Scheduler (PMS). The role of the Power Management Scheduler is to enable control and scheduling of the platform's power domains while maintaining very low power operation. Most commonly the applications processor is responsible for central control of power management functions including scheduling of device sleep and wake operations. Low power operation is limited since the applications processor is on

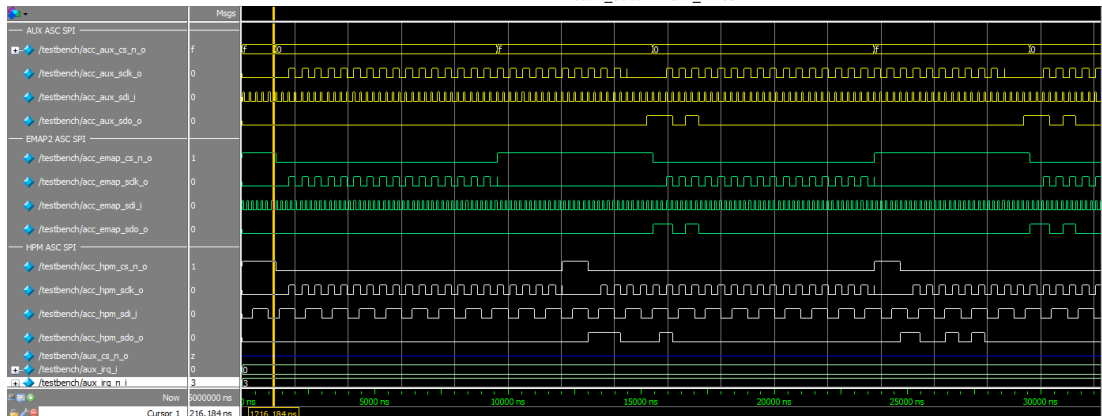
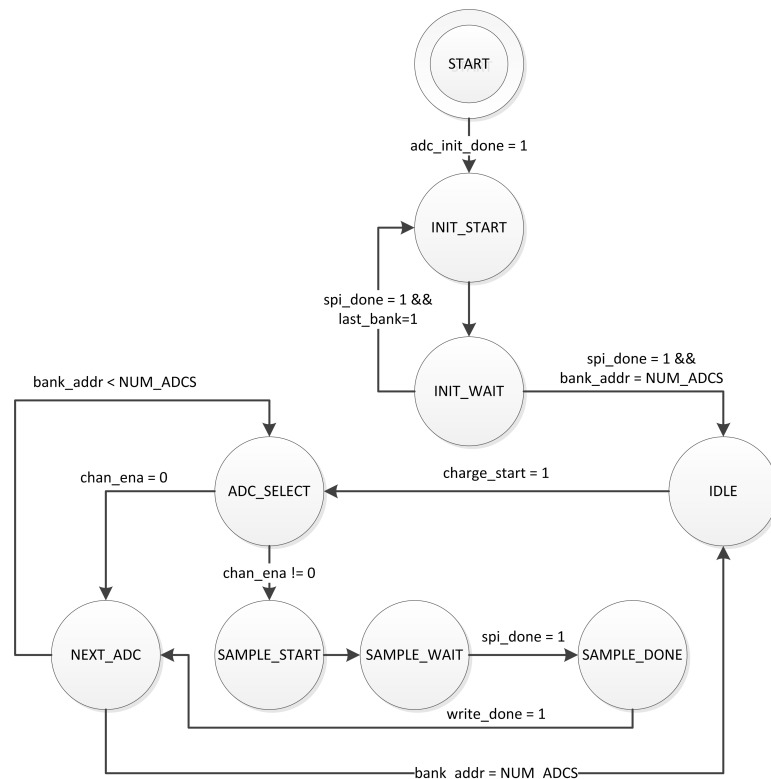


Figure 3.15: ASC state machine for the TLV2548 (a) and waveform example of multiple ASC modules sampling in parallel on EMAP2 (b)

during power management events. Instead a dedicated power manager function which operates independently from the applications CPU is required. The PMS architecture is shown in Figure 3.16. The PMS manages the power state of multiple power domains simultaneously. Each power domain may be in one of three logical states: on, off, or sleeping (although multi-level sleep states may be supported in the PSM if required). The on/off power state is presented to a pin that may be connected to either external power switches or regulator enable signals in order to turn off the selected power domain. The wake/sleep state is output to a separate pin which is used an interrupt to wake external devices from sleep modes. Each power domain's operating state is managed by the PMS. At reset, by default all power entries are set to the 'on' state.

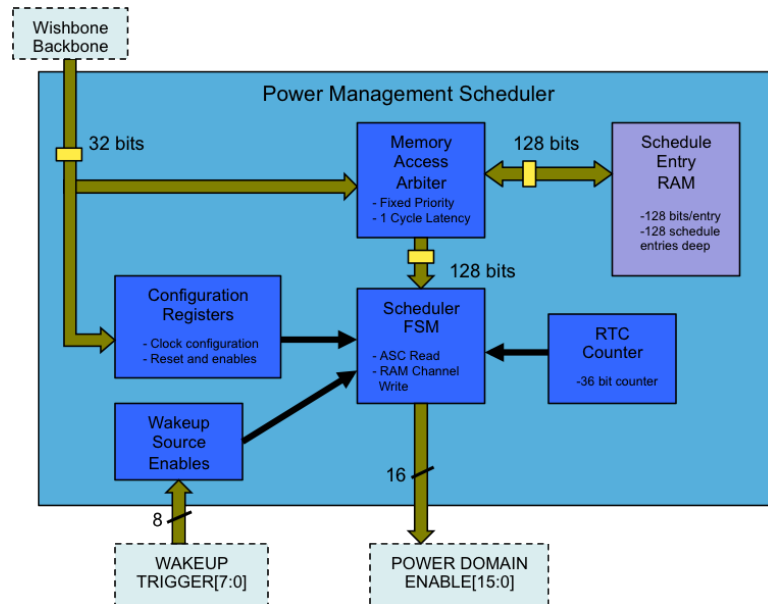


Figure 3.16: Power Management Scheduler architecture

The PMS is composed of as set of three subsystems: the configuration and control registers, power management processor, a RAM table of comprised of power management entries. The configuration and control registers provide basic reset and enable signals as well as clock rate controls which determine the

maximum rate at which power management operations may occur. The PMS main clock can be divided both by the Clock Manager and internally to generate frequencies from 1MHz to down to 100Hz. This clock signal is used to drive a 36-bit real-time counter (RTC), providing counter rollover periods from 19 hours at 1MHz to over 21 years at 100Hz. The RTC value provides the system timer for the PMS, enabling configurable scheduler resolution from 1us to 10ms based upon the configured input clock rate.

The PMS block RAM is responsible for storage of power management entries. Each entry consists of several fields as shown in Figure 3.17 and include: delay, period, actions, and identifier tag. These entries support scheduling different types of events such as single events occurring at specific time, periodic events with fixed repeat intervals, periodic events beginning at a specific future time and repeating with fixed intervals, and asynchronous events. Each scheduling entry contains: a 36 bit delay field, 16 bit power domain bitmask, 2 bit action tag, 32 bit period field, and 18 bit user identification (UID) field. Each entry's action tag and power domains bitmask set the operations for that entry. The action tag sets the entry to be one of: power domain on, power domain off, or set domain wakeup. Entries may also include a user identifier tag (UID) for the applications processor to use as a reference for each of the table entries. This is important since power management entries may need to be modified during operation. The identification tag provides a simple method for application software to locate entries within the table as well as control permissions in multi-user systems where different power domains may be managed by separate users.

The applications processor is responsible for writing power management events to the PMS RAM via Wishbone write cycles initiated through the HIC. The applications processor uses read/write access to the RAM blocks to load, modify, and

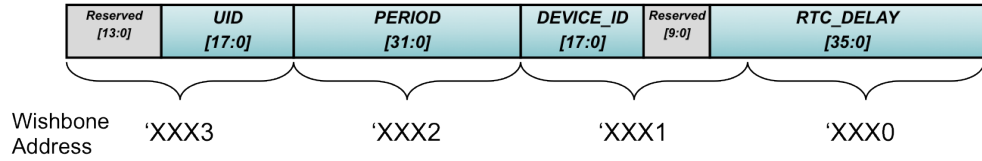


Figure 3.17: Power Management Scheduler table entry

delete power management scheduling entries. The power management processor within the PMS is responsible for executing all power management activities based upon these entries in the RAM table. The control logic for entry processing is shown in Figure 3.18. At each RTC update, the power management processor searches the table for expired entries. It then sequentially parses each slot with the power management table to locate valid entries. An entry is active if its enable bit is set and the delay field is less than or equal to the RTC counter. When an active entry is found, the element's action bitmask field is read to determine the required power management action. If the entry's period field is non-zero, this period value is added to the element's delay field and the entry is re-enabled, otherwise the entry is disabled via a write to the action field. Once the scheduler completes parsing all entries in the RAM block, it is disabled until the next update of the RTC. This gives the worst case response latency for any scheduled power management action of 128us when driven by a 1MHz external clock. If an entry is found that is both valid and expired, its action tag is executed on the selected power domains. Any periodic entries are then rescheduled to reset the new activation time stamp and reenabled while non-periodic entries are invalidated. If asynchronous event has occurred such as an interrupt, the configured output events are activated. We show a summary of the pseudo code logic in Algorithm-1 for reference.

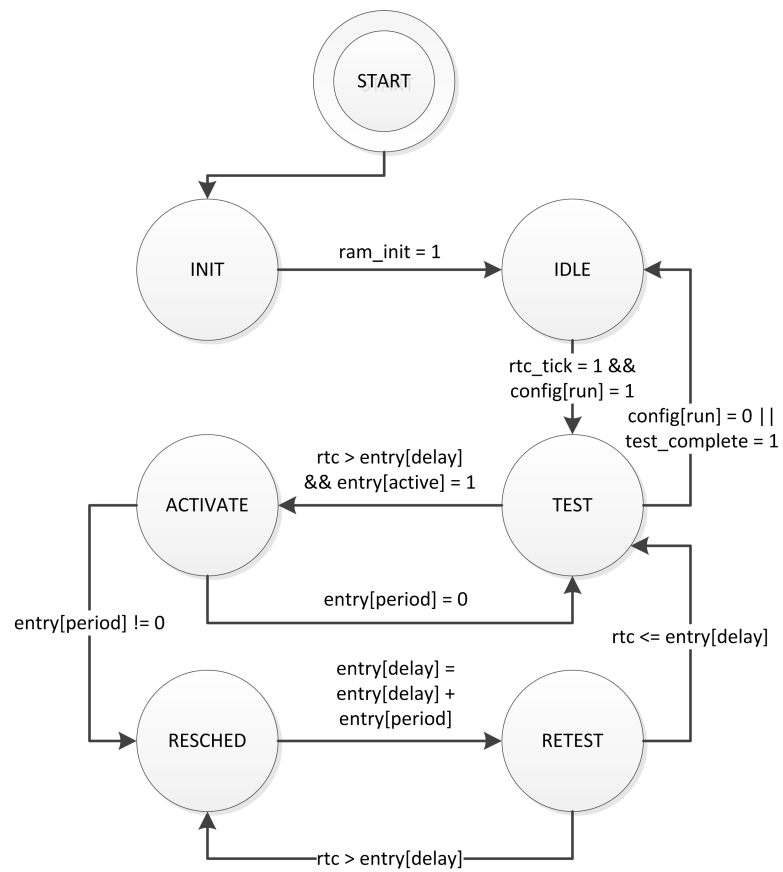


Figure 3.18: Power Management Scheduler state logic

Algorithm 1 PSM pseudocode logic

```
1: INIT:  $i = 0$ ,  $RTC = 0$ , power_state = all on
2:
3: repeat
4:    $RAM[i] = 0$  { Initialize all entries in RAM }
5:    $i = i + 1$ 
6: until  $i = 128$ 
7: while  $CONFIG[RUN] = 1$  do
8:   if RTC has changed then
9:      $i = 0$ 
10:    repeat
11:      while memory arbiter not ready do
12:        wait one cycle
13:      end while
14:      entry =  $RAM[i]$ 
15:      if entry[delay] < RTC and entry[enabled] = 1 then
16:        power_state = entry[action]
17:        if entry[period]  $\neq 0$  then
18:          entry[delay] = entry[delay] + entry[period]
19:        else
20:          entry[action] = invalid
21:        end if
22:      end if
23:       $i = i + 1$ 
24:    until  $i = 128$ 
25:  end if
26: end while
```

Performance Analysis Having now described the LEAP architecture and its primary hardware components, we now present an analysis of LEAP implementations, including its performance through direct measurements. We first review the operation of the EMAP function, a critical part of LEAP operation, through instantiations of the EMAP logic into two different FPGA’s architectures and present the details of these implementations.

	QL8250	AGL600
Quiescent Power	4.3 mW	0.057 mW
Maximum Clock Rate	28.9 MHz	34.6 MHz
Resources (% of total)	96%	44%
Clock Manager	7%	3%
Interrupt Controller	5%	2%
GPIO Controller	9%	4%
Energy Accounting Manager		
Block RAM	50%	25%
Logic Tiles	23%	12%
Power Management Scheduler		
Block RAM	50%	25%
Logic Tiles	19%	8%
Host Interface Controller	33%	15%

Table 3.7: Quicklogic QL8250 and Actel IGLOO AGL600V2 implementation details

3.1.5.2 EMAP Implementations

During EMAP ASIC development we generated two implementations of the design. Our initial implementation utilized a Quicklogic QL8250 [Qui] based upon an anti-fuse design principle. These FPGA's are one-time programmable parts which offer very low quiescent current draw do to the read-only anti-fuse architecture. While this has distinct power advantages over SRAM based FPGA's the one-time programming feature was a burden during rapid development cycles. Following the QL8250 we selected an Actel IGLOO AGL600 [Mic12], an FPGA featuring low quiescent current due to its flash based gate architecture. Unlike the QL8250, the AGL600 is reprogrammable via its JTAG port. As will be described in the later applications section, the JTAG reprogramming feature may also be performed via the applications processor while deployed in the field.

Table 3.7 shows the FPGA resources, maximum clock rate, and power consumed by the Quicklogic QL8250 and Actel IGLOO AGL600 FPGA's. As can

be seen in the table, the AGL600 has significant power and speed advantages over the older QL8250 architecture. This is due to its lower core logic voltage operation (1.2V vs 1.8V) and smaller gate feature size. In addition to its lower quiescent power, the AGL600 also performed at lower operating power than the QL8250 implementation at various sampling frequencies. Additionally to validate the EMAP solution against other alternative designs, we performed experiments using the prior LEAP1 platform design and also a design constructed from off-the-shelf (OTS) components providing similar, though not identical functions. This power and performance comparison is shown on Figure 3.19. An important differentiator of the EMAP ASIC based solutions is they maintain a nearly constant power profile (for a fixed sample rate) that does not depend on the number of channels being measured. This is due to the EMAP's ability to parallelize data sampling by adding additional ADC sampling controller's to the EAM without increasing the core clock rate.

The LEAP1 solution, which operates with low average power, suffers from low communications bandwidth between the MSP430 microcontroller and the applications processor. This increases data collection overhead due to wasted power in the applications power. The MSP430 also suffers from a limitation to the number of sampling channels. The OTS solution, which uses discrete Texas Instruments BQ27000 fuel gauge IC's for each power domain, is superior when used at low sample rates and low channel counts. This is intuitive since the additional hardware of the EMAP solution is not fully utilized in low channel count applications. However as channel count increases, the redundant charge accumulation hardware (replicated in every BQ27000) quickly increases the total power of the OTS solution beyond the custom EMAP implementations.

As shown in the performance graphs, the custom EMAP solution based upon

	EMAP: QL8250+	EMAP: AGL600+
	TLV2548	ADS7844
Sample Rate(max 1 ch)	38462	76923
Charge Accuracy(min uVh)	8.79	8.79
Maximum Channels	64(CAC RAM limited)	128(CAC RAM limited)
Host Charge Read Time(us)	0.096	0.070
VDD range(V)	1.2-5.5	1.2-5.5
Auto Reset Capable	Yes	Yes
	MSP430F1611	BQ27000
Sample Rate(max 1 ch)	95400(CPU limited)	256
Charge Accuracy(min uVh)	16.11	3.57
Maximum Channels	8(ADC port limited)	10(I2C address limited)
Host Charge Read Time(us)	12	320
VDD range(V)	1.8-3.6	2.6-4.2
Auto Reset Capable	No	No

Table 3.8: Implementation details of energy measurement solutions

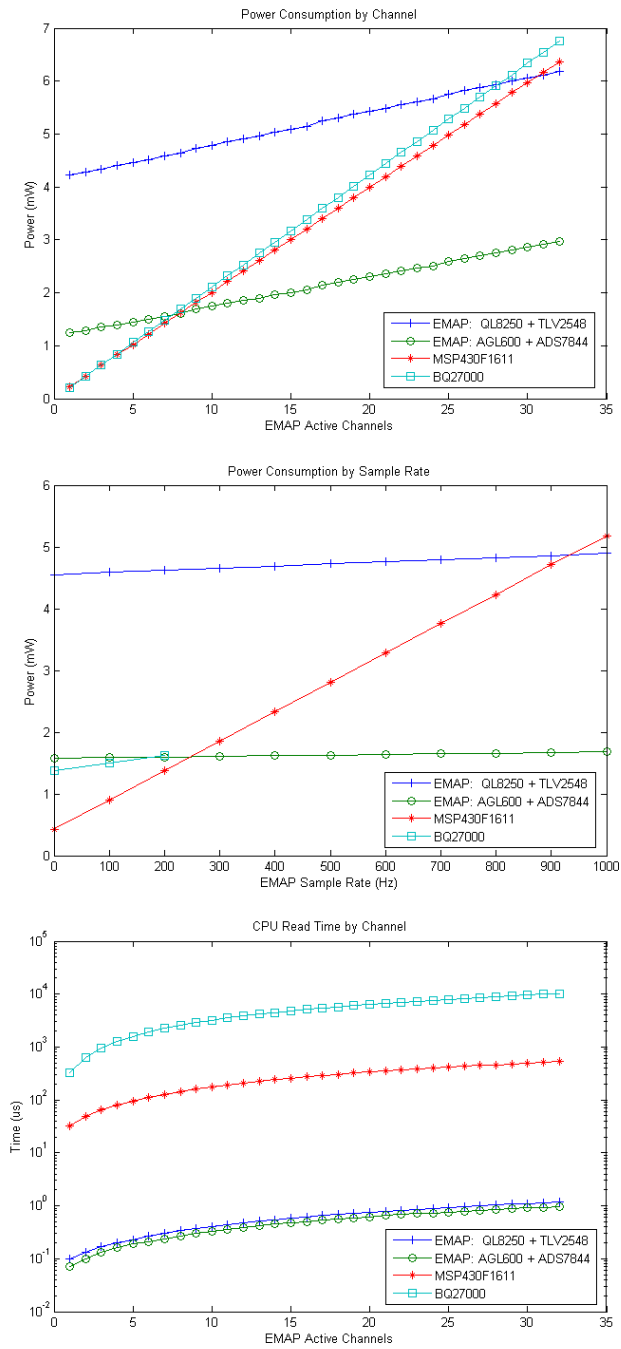


Figure 3.19: Power comparison of four energy measurement solutions: (a) Total sampling power by number of channels (b) Total sampling power by number by sample rate and (c) applications processor sample read back time

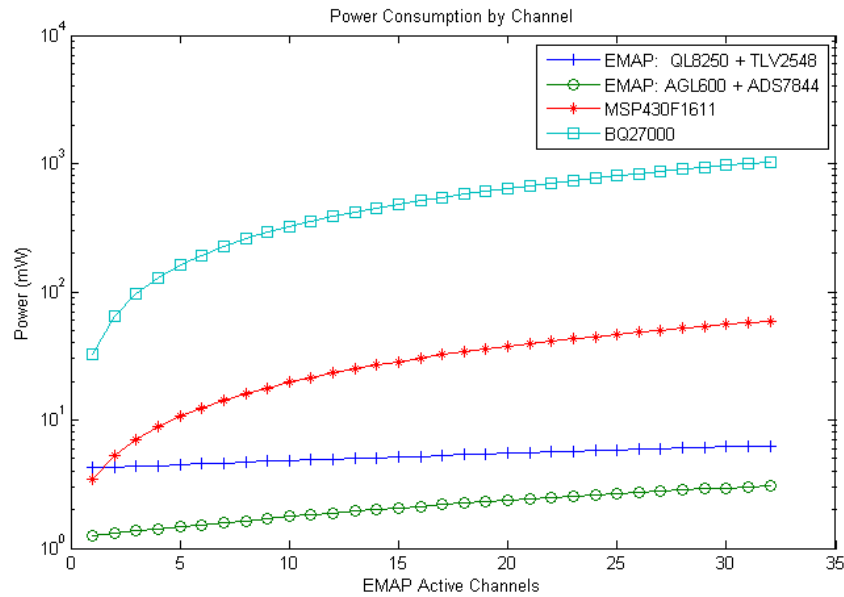


Figure 3.20: Comparison of four energy measurement solutions when including applications processor power, assuming 1W operation with 100 sample reads/sec

the AGL600 provides lower power performance than both the MSP430 and BQ27000 solutions once approximately 8 channels are sampled. Once the power consumed by the applications processor is also considered the advantage of the dedicated solution is enhanced. Figure 3.20 shows the additional power overhead incurred with traditional low bandwidth interfaces becomes significant as channel count increases. As shown, the OTS fuel gauge solution reaches 100% of the available bandwidth at its peak, which requires the applications processor to remain on and consume significant idle power.

LEAP2 Platform The previous section describes the advantages of the dedicated EMAP energy accounting method versus traditional solutions. In order to demonstrate and utility of EMAP capabilities and the value of the LEAP architecture in generating energy-aware ENS systems, a modular stackable platform was designed and constructed from custom printed circuit boards. This platform,

named LEAP2, contains all the advancements of the LEAP architecture previously described and has been used as test bench for development and validation of several energy-aware software tools.

The LEAP2 platform is a second generation, low power, LEAP enabled ENS system allowing high accuracy, low overhead energy measurement of platform computing, storage, sensing, and communications devices at granularity levels previously unachievable. LEAP2 incorporates greatly enhanced energy accounting and scheduling capabilities over the previous LEAP1 design including the EMAP ASIC, and expanded peripheral and sensor interface selection, and a larger set of storage and communications devices to enable run-time energy optimization. LEAP2 is a highly modular design. Energy measurement domains may be added to or removed from the platform through an expandable energy monitoring bus integrated into the LEAP2 stacking connectors. This enables future add-on modules to benefit from the LEAP2 energy management and accounting features.

Each PCB module was constructed with high resolution energy measurement capabilities at fine grained levels previously unattained to our knowledge. The LEAP2 platform is shown in Figure 3.21 along with a block diagram highlighting the various PCB modules. We first describe these platform modules and their designs. We then provide details on the many software capabilities built upon the energy sensing capabilities of LEAP2.

3.1.5.3 EMAP2 Communications Module

The EMAP2 module (Figure 3.22) provides the primary LEAP functions including the EMAP ASIC's energy measurement capabilities, sensor serial bus multiplexing, expandable power management bus, a modular stacking interface, and the communications functions. The module's EMAP ASIC performs continuous,

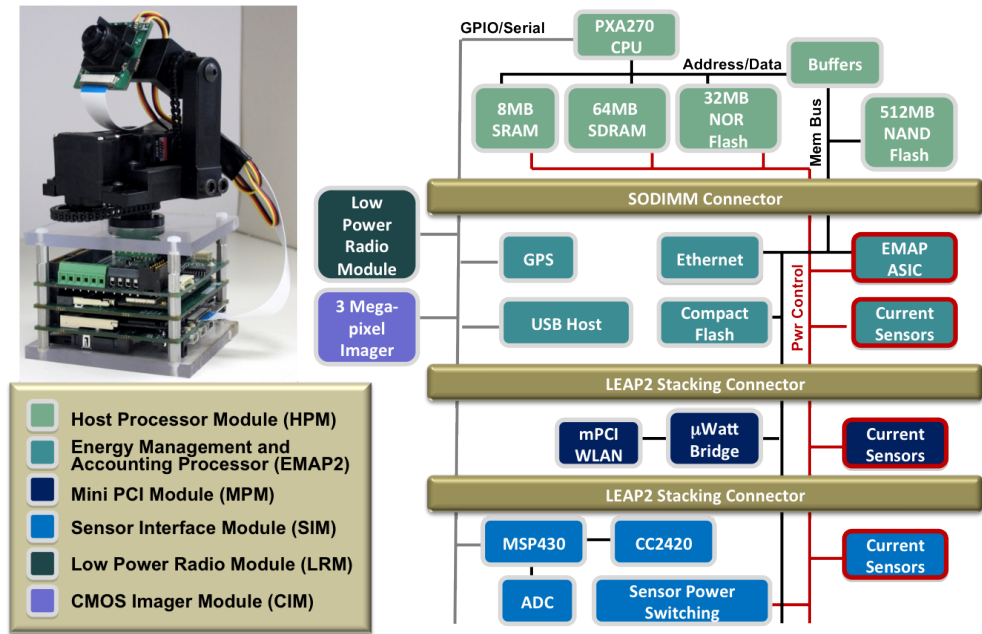


Figure 3.21: Photo and block diagram of the LEAP2 platform

real-time energy monitoring at a fine-grained device level across the entire platform including all add-on expansion modules through its modular, expandable power management serial bus. The module, as shown in Table 3.9, has eight current sensing channels allocated to the EMAP2 module's peripherals such as Ethernet, USB, quick capture camera, and Compact Flash. Eight channels are allocated for the host processor module's use through the host interface connector, and 32 additional sensors may be connected through the power management bus on separate modules.

Each of the EMAP2 module's power domains is electrically isolated, requiring all voltage supplies to pass through precision current sensing resistors. The voltage across the sense resistor is amplified through Texas Instruments INA216, a high common mode rejection, differential amplifier. The signal is then filtered through a single pole, passive low pass filter, and then sampled by an 8 channel,

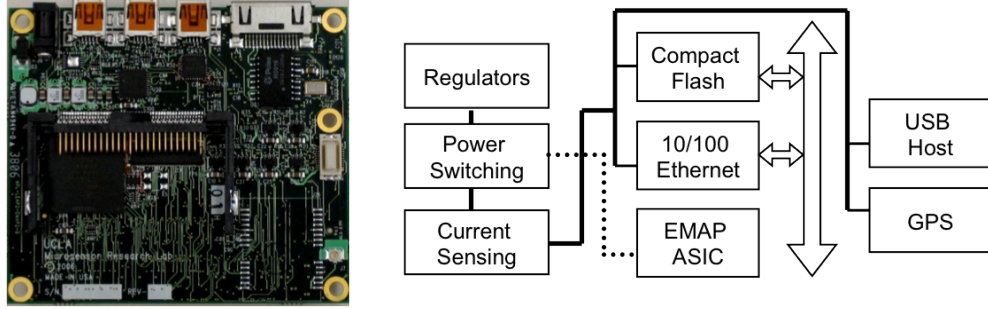


Figure 3.22: Photo and block diagram of the energy management and accounting (EMAP2) module

EMAP2 Operating State	Power(mW)	
	Typical	Max
RTC Only (hibernate)	2.3	2.6
EMAP Sampling	4.64	6.5
Ethernet	198	429
USB	125	2500
Camera	286	356

Table 3.9: EMAP2 operating states and power consumption

12 bit analog to digital converter (ADC), a Texas Instruments TLV2548. The sense resistor and amplifier gain values are scaled per domain to prevent saturation of the ADC signal and to maximize the measurable dynamic range. The maximum anticipated current draw is used to set the amplifier output voltage to the ADC full scale value, set by a 3.0V precision voltage reference.

We verified EMAP2 power measurement accuracy, which includes both current sensor and ADC measurement, by comparing applied current loads against their sampled values for several of the EMAP2 power domains. This testing was performed on a sample of units over the range of specified operating power points in Table 3.9. The results of several calibration measurements are shown in Figure 3.23. Our measurements verified that the expected and actual measurements were within 10% across the operation spectrum with a 95% confidence and

average errors were typically less than 2%.

We also verified the energy cost for the applications process on the LEAP2 platform to read from the EMAP ASIC. The CPU's execution time and energy during sampling of the energy data was recorded for several configurations of channel number and sample rates. As shown in Figure 3.24 the high bandwidth interface of the EMAP ASIC requires minimal additional energy even at large channel counts and high sample rates. This validates our EMAP design has achieved our LEAP design goals.

The EMAP2 module provides many of the communications interfaces present in modern ENS platforms. Ethernet is valuable for providing gateway functions for network edge nodes while USB and Quick Camera interface are used for adding imaging capabilities. The power consumed by each of these communications interfaces is individually measured. In addition, the module utilizes a stacking architecture to enable adding new platform enhancements. Located on this stacking connector are numerous serial and parallel buses, all routed through the EMAP's resource multiplexor to facilitate run-time resource management from the applications processor. Several power resources on the stacking connector may be controlled via the EMAP ASIC's Power Management Scheduler.

In addition the EMAP2 provides an ultra-low power hibernation mode to enable extended operation from battery in long deployment applications. In this mode, the EMAP ASIC (via the Power Management Scheduler) initiates a platform shutdown whereby all power domains (including their voltage regulators) are shut off to reduce power. The EMAP ASIC then enters a lower power standby mode, which disables the high frequency internal clock domains. An external low frequency clock signal (provided by a micro-power silicon oscillator or watch crystal) is enabled to provide updates to the EMAP's real time clock register.

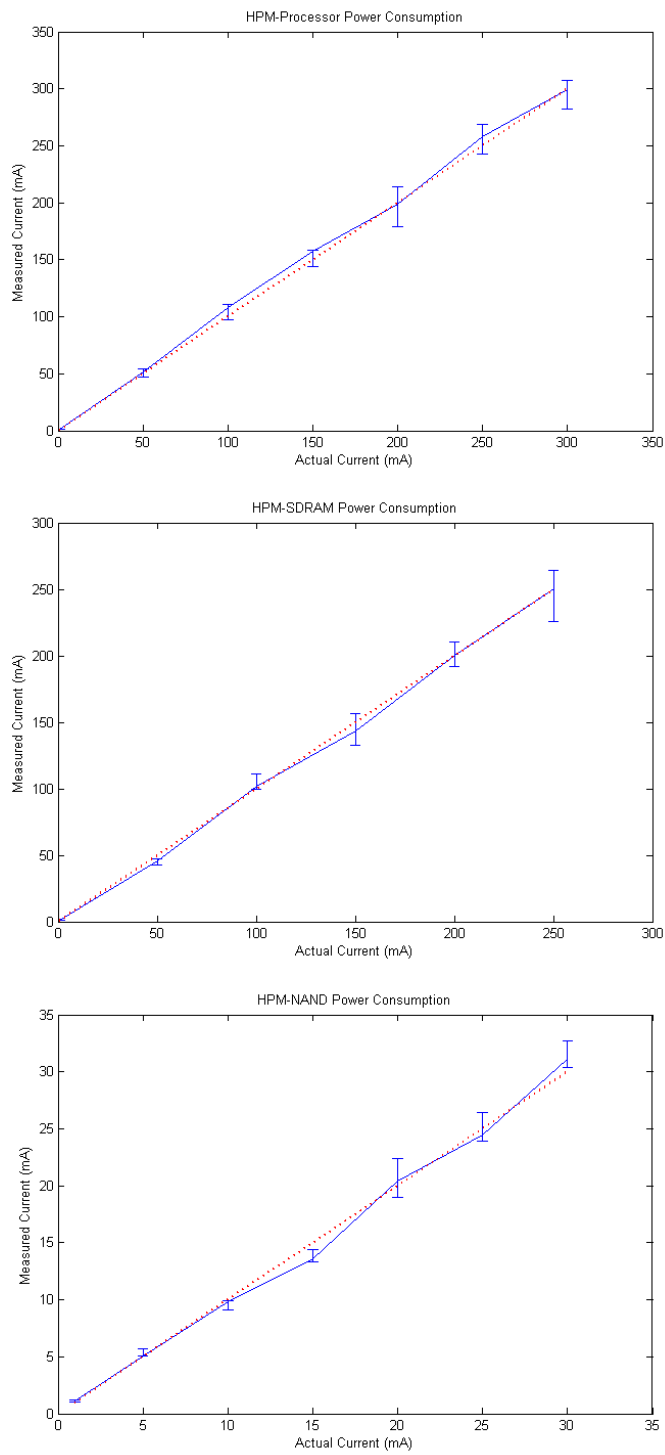


Figure 3.23: Measurement accuracy of the EMAP2 current sensors and ADC for the following channels: (a) HPM CPU (b) HPM SDRAM (c) HPM NAND

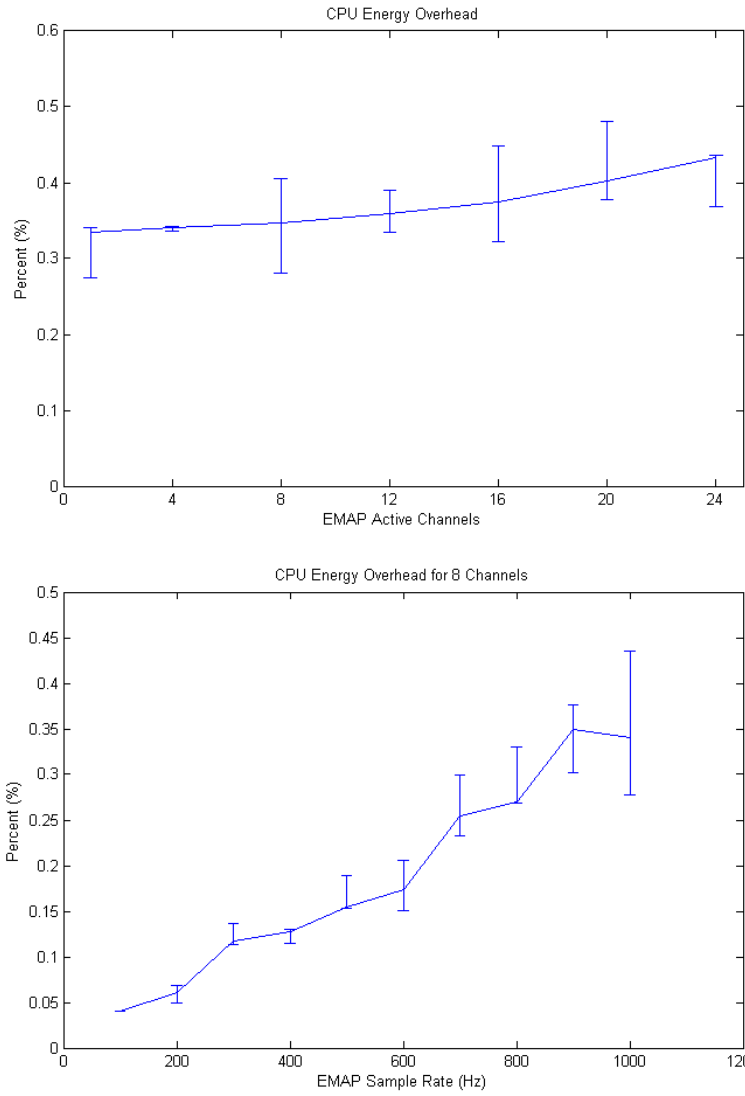


Figure 3.24: CPU energy overhead when reading from the EMAP ASIC on EMAP2: (a) CPU sampling energy by number of sampled channels (b) by sample rate

Likewise, the Power Management Scheduler retains its table entries and continues to operate, albeit at a reduced rate. Wakeup from the micro power sleep state are limited to asynchronous trigger inputs or scheduled power management events. For example, the platform could be configured to restart at a fixed future date and time (perhaps for network synchronization purposes), or by low energy triggers such as PIR sensors or tripwires.

3.1.5.4 Host Processor Module

The Host Processor Module (HPM) provides the main applications processor as well as several volatile and non-volatile storage media for local processing of sensor and communications data. The HPM is attached to the EMAP2 via a standard SODIMM200 connector for high bandwidth and high point count connectivity. The HPM's interface to the EMAP2 is designed such that the processor module may be redesigned to incorporate different CPU architectures (including future updates) while maintaining a common interface to the EMAP2. One architecture, based on the Marvell PXA270 applications processor, was implemented and fabricated for proof of concept of the LEAP design principles. This design includes several heterogeneous storage types including SDRAM, SRAM, NAND flash and NOR flash. As was noted early in section 2.2.3, each of these storage types provides different energy advantages depending on its usage pattern. In this design, each processing and storage resource is isolated into its own power domain and connected via the power management bus to the EMAP ASIC. This allows unprecedented access to high resolution energy dissipation information of the CPU and each of its memories. With this architecture it is now possible to do run-time analysis of storage efficiency for a specific applications needs.

In addition to a traditional 32-bit memory bus mapped to each of the storage

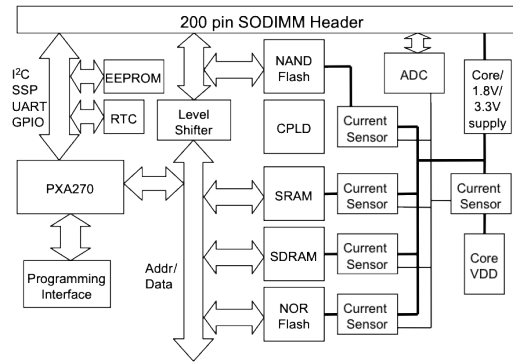
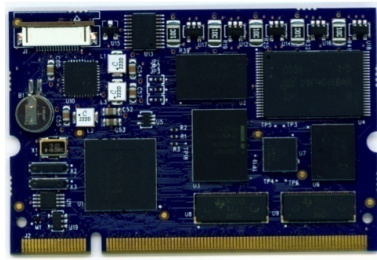


Figure 3.25: Photo and block diagram of the host processor module (HPM)

peripherals, a programmable logic device (CPLD) on the HPM module enables direct memory-to-memory DMA transfers for both the SDRAM and SRAM peripherals. This works with the EMAP’s Host Interface Controller bus mastering capability described in section 3.1.5.1 to perform direct writes from the Energy Accounting Module into the HPM memory areas. This capability simplifies high frequency sampling of energy data during time critical events on the host CPU. This, for example, is needed in energy attribution to operating system threads, which may be swapped at frequencies as high as 1kHz. This DMA function is also critical to enable low power sensing operations, where data processing is performed not by the applications processor, but specialized detection circuitry. In these applications data logging to the applications memories is possible while keeping the applications processor itself in low power standby or powered off states.

The PXA270’s memory map is shown in Figure 3.26. In addition to the the onboard heterogeneous memory resources, several chip selects are routed to the EMAP2 and stacking interface connector allowing additional modules to be directly mapped to CPU memory without the need for an intervening bridging function. The PXA270 CPU is capable of operating over a wide frequency range using standard DVFS techniques. This DVFS function provides both low power

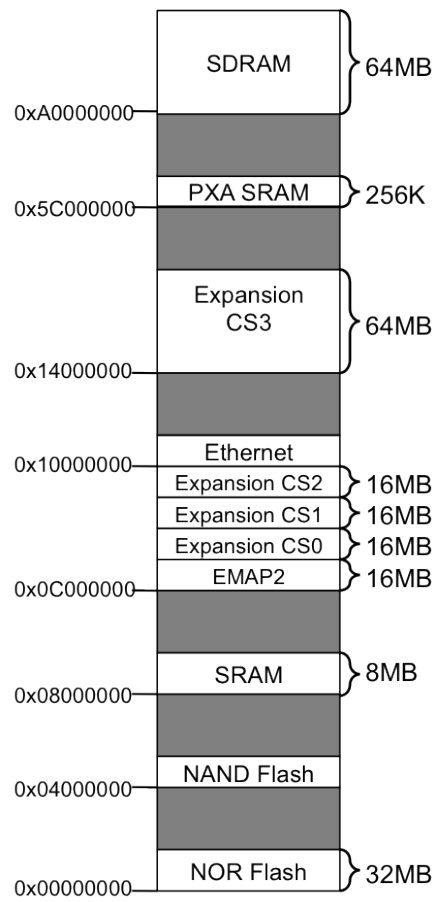


Figure 3.26: Memory map of the HPM resources from the PXA270 CPU

PXA270 CPU Frequency (MHz)	Current(mA)@5V	
	Idle	Max
624	55	214
520	47	170
416	40	131
312	34	101
208	30	77
104	16	44
13 (in SRAM)	5	5

Table 3.10: PXA270 operating frequencies and operating power

operation during processor idle times and well as high capacity processing during peak operating periods. The operational states for the PXA270 CPU are listed in Table 3.10 as well as the measured current consumption of the HPM in this state. This included a custom designed, ultra-low power 'microcontroller' mode where the PLL circuits were disabled and the CPU operated directly from the crystal oscillator. In this mode the CPU operated out of the static RAM and was able to perform basic platform vigilance functions. The CPU was then able to switch into higher performance modes dynamically to enable enhanced capabilities such as sophisticated operating system functions.

3.1.5.5 Sensor Interface Module

The Sensor Interface Module (SIM) provides low power sensing operations including data collection and micro power processing, as well as low bandwidth radio compunction and a bank of switched power supply outputs to external sensors. Each of these operations is supported by energy monitoring through independent power domains connected through the power management bus to the EMAP ASIC. A block diagram of the SIM is shown in Figure 3.27 noting these general capabilities. The SIM is capable of independent operation from the

SIM Operating State	Power(mW)	
	Typical	Max
MSP430 PM2	0.14	0.23
MSP430 ADC Sampling	4.4	5.9
External Switches On	5.7	10.2
CC2420 Tx	65	72
CC2420 Rx	62	66

Table 3.11: SIM operating states and power consumption

previously described modules in the LEAP2 stack. The entire platform may be powered down leaving only the energy measurement functions of the EMAP2 and the SIM operating enabling basic operations.

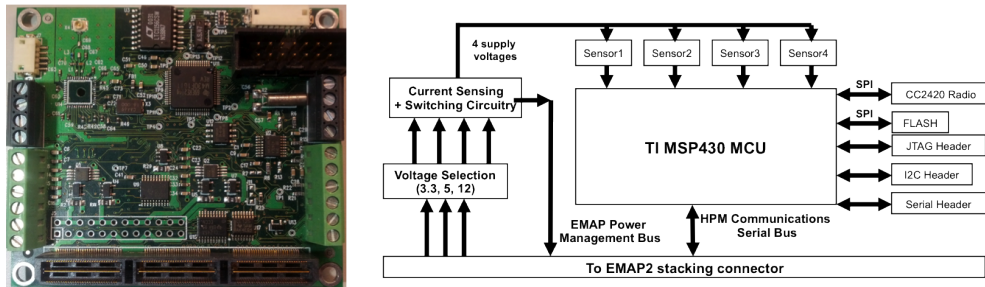


Figure 3.27: Photo and block diagram of the Sensor Interface Module (SIM)

To facilitate coordinated sensing activities, we added a low power CC2420 radio was added to the MSP430’s interfaces providing short range communications and networking capabilities. The SIM’s MSP430 and CC2420 architecture along with energy sensing circuitry was the basis for a derivative platform named uLEAP, as shown in Table 3.11. We note several operating states of the SIM and their power consumption in Figure 3.22. In addition to the on board sensing and communications options, the SIM supports powered external sensor equipment. Operation of each of these external sensors may be scheduled through the EMAP Power Management Scheduler.

3.1.5.6 MiniPCI Module

The MiniPCI Module (MPM) provides the functions necessary to enable high bandwidth, long-range wireless communications. These functions include: a high current power supply, a local memory bus to PCI bridging component, a commercial standard interface connector, and integrated energy management capabilities. The MPM includes a miniPCI connector, which complies with the commercial standard used widely in PC laptop technology. The miniPCI form factor is a plug-in card standard for many off-the-shelf *WiFi* and cellular wireless technologies. The MPM's high current power supplies support even the specialized 'high-power' *WiFi* card options. This option may be necessary when deploying into challenging environments such as in dense foliage and/or large inter-device spacing.

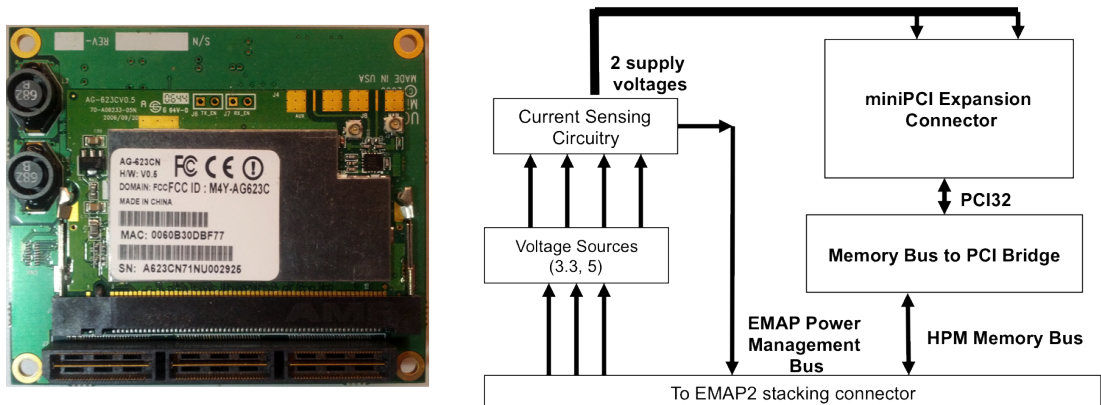


Figure 3.28: Photo and block diagram of the MiniPCI Module (MPM)

The miniPCI bus interconnect provides high networking throughput by connecting directly to the PXA270's SDRAM through bus mastering DMA from the cards PCI memory, through the MPM's bridge, and directly onto the HPM. Data words (32 bits wide) are transferred at a full $33MHz$ with single cycle latency. The aggregate bandwidth including bus mastering setup times exceeds

100MB/sec, which accommodates high throughput bursts to and from the *WiFi* cards local memory with very low overhead. In addition the DMA operations do not require applications processor activity, so the CPU may idle during data transfer operations to reduce power consumption.

3.2 LEAP Enabled Software

We have described the key hardware components required to enable LEAP technology as well as provided technical details of one example, the LEAP2 hardware platform, which serves as a reference design to showcase the LEAP design principles. In order to realize the capabilities provided by LEAP enabled hardware, we must now turn to development of the necessary software components. We begin with a discussion of the fundamental operating system enhancements and low-level device driver modifications made to accompany the LEAP2 reference design. This includes integrating EMAP measurements into key functions within the operating systems to enable attribution of hardware energy dissipation to abstract operating system objects (such as applications and users) that share access to the underlying hardware. We then describe several energy profiling applications which have been build atop the energy-aware operating system software.

3.2.1 Operating System Integration

The EMAP ASIC provides unprecedented access to energy dissipation data of platform hardware components. It does not provide any direction on how this energy dissipation information is best used by software components to enhance low power operation and extend battery lifetime. In contrast the operating system is a set of software services and programming abstractions allow coexistence

of many individual applications by coordinating accesses the many platform resources and serving as a central control mechanism for platform services. The operating system must therefore be augmented to integrate this new energy data to its fundamental components as to enhance these software abstractions with ownership of energy utilization data. As part of LEAP development, we have enhanced one existing open source operating system, Linux, with these additional energy-aware capabilities to provide a fine grained energy attribution capability to a variety of operating system functions.

The Linux source code already integrates several software tracing projects whose objectives are to provide low level monitoring functions in order to gather information about a running Linux system. This capability is frequently utilized for debugging, to locate and correct hard to generate errors, or for performance tuning, to increase throughput or efficiency. These tracing methods include both static and dynamic trace point insertion methods. Linux *Tracepoints* [Bar] are static markers within the kernel source which, when enabled, can be used to hook custom software routines into the running kernel at specific points where these markers are located. Linux *Tracepoints* can be used by a number of tools for kernel debugging and performance problem diagnosis. One of the advantages of *Tracepoints* are they allow developers to monitor many aspects of system behavior without needing to know much about the kernel operation itself. In contrast, Linux *Kprobes* are a dynamic probing capability. With *Kprobes*, the developer is able to insert tracing methods (taps) into arbitrary locations within the running kernel without precompiling these tap locations a priori. However, the *Kprobes* developer needs intimate knowledge of the kernel operation in order to effectively insert probes into the proper location within the kernel binary.

During LEAP software development we found that both methods provided

value and have utilized each of these methods to enhance the Linux kernel with a new energy-aware functionality. This is provided by a custom, centralized *energy registry* within the kernel. Due to the extremely low overhead energy sampling of the EMAP ASIC, it is now possible to add energy sampling methods at even the high frequency operations of the kernel. Using our *energy registry* software developers may enable energy sampling from any of the existing *Tracepoints* or *Kprobes* tap locations. In order to use the energy registry, software components first register their intent to measure specific power domains and whether those measurements are should be 'self-clearing' operations (i.e. incremental energy usage measurements). Once this intent has been registered, the tracing software may call the energy registry's sample request methods. The registry then accesses the EMAP ASIC's energy sample data (for the previously registered power domains), and returns energy sample data to the caller. For a complete listing of *energy registry* source code the reader is referred to Appendix B.

In addition to accounting functions, the *energy registry* is responsible for managing requests for power domain scheduling. The registry provides a simplified abstraction layer for software components by managing the low level event table stored within the EMAP. Operating system software may then request basic power domain operations such as power on, power off, or wakeup triggers each domain while the energy registry manages request conflicts. The avoid scheduling request conflicts, the energy registry utilizes the user ID tag field found in each entry of the PMS RAM table as a handle to each registry user. The handle enables retrieval of all unexpired schedule entries for each user and domain. Via this handle, the energy registry is able to verify that different software components do not operate on the same power domain.

To provide basic command line scripted operation of EMAP functions, a cus-

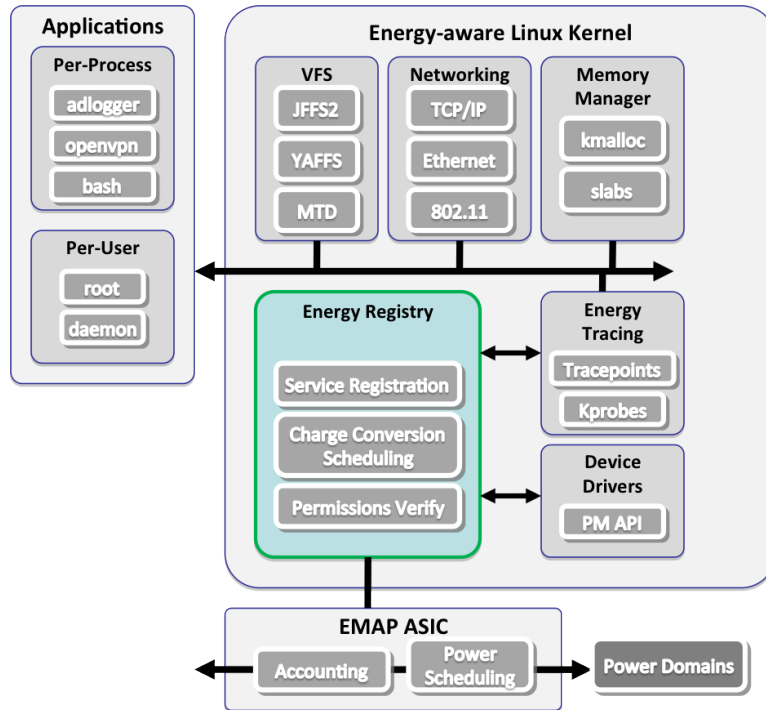


Figure 3.29: Block diagram of the developed Linux energy registry

tom set of */proc* interface files are available. These export the EMAP’s energy information from the kernel to user space applications. A list of these information files located in */proc/leap* files is below:

1. *energy*—Provides latest energy dissipation information for each power domain and the voltage settings for each voltage channel.
2. *energy-reset*—Provides latest energy dissipation information as above but each energy counter is reset after reading.
3. *schedule*—Provides a command line interface to access the PMS scheduler table data.
4. *status*—Provides current power domain status information.

5. *trigger*—Provides a list of the currently enabled wakeup triggers in the EMAP PMS.
6. *rtc*—Provides latest real-time counter value from EMAP PMS.

Listing 3.1: Example data from `/proc/leap/energy` file showing power domains and energy dissipation

```

External Voltage: 15.788
External Current: 0.203
Battery Current: 0.003
Battery Voltage: 14.628
Energy Status:
Name:           Current   Charge    Energy    Time
SDRAM:         0.067A   16.671C   30.041J   2707.871S
NOR Flash:     0.000A   0.202C    0.364J    2707.871S
SRAM:          0.000A   0.198C    0.356J    2707.871S
PXA Core:      0.054A   72.870C   112.292J   2707.871S
NAND Flash:    0.000A   0.399C    1.316J    2707.871S
Ethernet:      0.001A   3.011C    9.936J    2707.871S
System:        0.413A   782.835C  3815.537J  2707.871S
Radio:         0.000A   1.615C    5.329J    2707.871S
GPS:           0.000A   8.128C    26.822J   2707.871S
USB:           0.155A   454.975C  1501.417J  2707.871S
SD:            0.001A   4.790C    15.807J   2707.871S
HPM:          0.253A   309.202C  1507.050J  2707.871S

```

3.2.2 Energy Profiling Tools

We now provide three examples of user applications showing the value of this energy registry. Each application is a tool providing energy profiling information to the developer from different energy usage perspectives. The first, *etop* collects and displays aggregated energy dissipation of platform applications and power domains and is valuable for comparing energy dissipation *between* applications. The second example, *endoscope* provides application energy tracing at higher time resolution, to locate high energy dissipation at critical sections *within* an application. The third example improves the aforementioned *endoscope* by enhancing the Linux Trace Toolkit [Ope12b] to provide energy dissipation information in graphical form in addition to existing kernel and application execution trace information.

3.2.2.1 *etop*

Etop is a user-space tool that enables rapid observation of energy consumption, when running an arbitrary set of processes. Using *etop*, an application developer can easily visualize, in real-time, the power draw of subsystems and energy consumption of running processes. Due to its rapid real-time information display, it can also be used as a runtime debugging tool, for quickly ascertaining system and process energy performance, as well as for understanding the energy behavior of processes in a dynamic environment. *Etop* is based on the well-known UNIX program *top*. *Etop* adds two additional capabilities: per-subsystem current, power and energy information and per-process energy accounting. Figure 3.30 presents a screenshot of *etop*, with the top section displaying per-subsystem information and the bottom section displaying per-process energy consumption.

Etop's per-subsystem information is directly linked to the output of the EMAP2

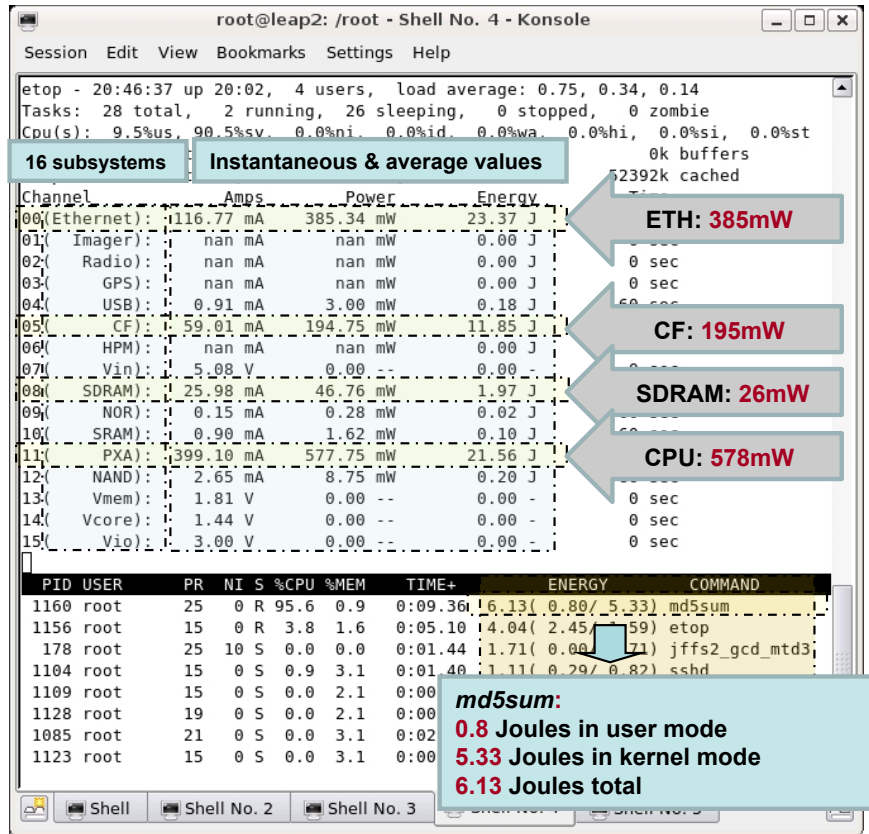


Figure 3.30: Etop screenshot displaying per-subsystem power and energy information as well as per-process energy consumption.

through the the *energy registry*. On every refresh cycle (a user-controlled parameter that *top* provides, with a default of 3 sec), *etop* presents current, power and energy consumption information of individual subsystems. It also provides voltage information for the four voltage-only channels. Current and power values are by default averaged over the refresh period; maximum values can also be displayed.

To provide *etop* with per-process energy accounting, we introduced a modification to the Linux OS scheduler so as to record energy consumption information, in a manner similar to ECOsystem [HCA02]. Specifically, we utilized *Tracepoint* markers around the Linux scheduler and augmented the *task_struct* structure

(that contains information about a specific process) with two energy containers: one for user-level information and one for kernel-level information. On every scheduler tick, Linux determines whether the current process time slice executed in user or kernel mode and charges an appropriate container accordingly. This information is used by the kernel (and *top*) to determine the fraction of CPU time spent in kernel (system) mode and in user mode. We then requested the *energy registry* to read from all channels on every scheduler tick. We charged this to the corresponding user- or kernel-level process energy container. As a result, we can disambiguate between energy consumption that occurred when a process was performing work in support of a user mode task and when the system was performing work during the process time slice in support of kernel mode tasks. Both these values are exported using the */proc* interface, in a manner similar to */proc/<pid>/stat*. *Etop* then reads those values and displays them in standard *top*-like column format.

3.2.2.2 endoscope

Since *etop* displays system energy performance in real-time it is not optimized for in-depth detailed measurements. *Etop* is a user-space process with non-trivial memory and CPU usage (especially at refresh rates higher than 1 Hz) and as such can interfere with the measurements themselves. To provide high-rate sampling in software with a very low energy overhead, we implemented the *endoscope* kernel module. *Endoscope* uses the *energy registry* to access the EMAP ASIC's energy data at frequencies as high as the default Linux scheduler tick resolution (1KHz in our system) and with very low CPU overhead, as no expensive user-kernel boundary crossings are required. The results are then stored in a circular buffer in kernel memory, a very fast and low-overhead operation. We utilize a

	Etop	Endoscope
Execution space	User	Kernel
Kernel modifications	Scheduler for per-process data	Kernel Module
Data display interface	<i>top</i> -like	/proc
CPU usage	1 – 25%	Negligible
Sample storage	No	Yes
Real-time data collection	Yes	Yes
Real-time data display	Yes	No
Per-subsystem resolution	Yes	Yes
Per-process resolution	Yes	No

Table 3.12: Qualitative comparison of *Etop* and *Endoscope*

standard */proc* interface for *user-space* data display and control purposes. To avoid frequent periodic polling of the */proc* interface from user-space, the circular buffer has sufficient memory to store several minutes worth of *real-time* data. To avoid buffer overrun, a user-space application only needs to read from the */proc* interface at an interval that is slightly smaller than the buffer’s capacity at a specific sampling rate. *Endoscope*’s negligible operating overhead, combined with the minimal power requirements of the EMAP2 ASIC, have enabled post-deployment monitoring and analysis of the power behavior of applications, both on a single node and on the entire network. As a result, *endoscope* has been used extensively to collect experimental datasets. *Endoscope*, unlike *etop*, measures energy consumption of entire subsystems rather than those associated with individual processes. As a result, when conducting our experiments, care is taken to ensure that *endoscope* measures energy solely attributed to a single application. For this development of *endoscope*, this procedure is effective for ENS systems that typically support one dominant application, or a set of applications that can be considered dominant. Table 3.12 summarizes the capabilities and differences of these software tools.

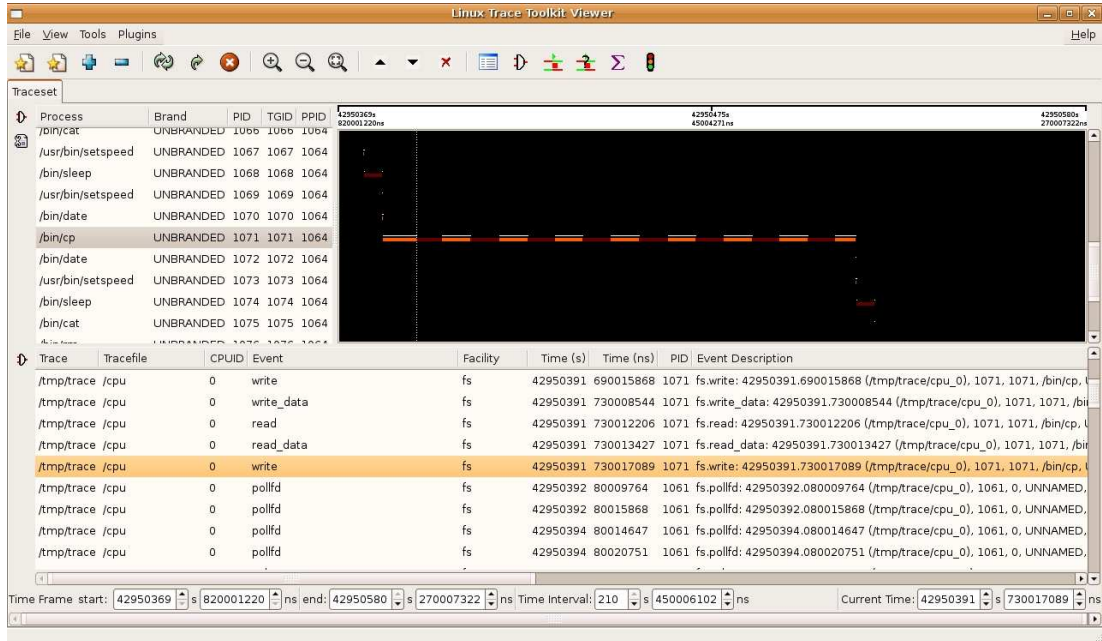


Figure 3.31: LTT screenshot displaying NOR flash write information as well as per-process energy consumption.

3.2.2.3 Energy-aware Linux Trace Toolkit

While *endoscope* provides real-time energy tracing data, it was soon discovered that the data generated during applications profiling can become a significant and may interfere with the testing application. We discovered a open source tracing and visualization application which solved many of the fundamental limitations of *endoscope* while providing a modular software architecture to enable a broad energy-aware development capability to both kernel and user-space applications. The Linux Trace Toolkit (LTT) is a sampling framework build upon the Linux *Tracepoints* and *Kprobes* capabilities to record software execution traces into a kernel buffer that is optimized for low overhead data transfer to the user-space recording applications. *LTT* supports multiple standard visualization tools including *Babeltrace* [Eff12] and *Eclipse* [Fou] for trace display.

We augmented *LTT* kernel trace generator code to include energy recording

data as part of its trace buffer records. *LTT* supports dynamically selectable context information to augment event payload such as PID, PPID, process name, and nice values. We modified added an additional context type to support energy accounting information in the event records. Now *LTT* users may choose to collect EMAP energy data at any of the enabled trace markers. This enhancement to the *LTT*'s sample collection routine has provided valuable insight into platform energy dissipation. Figure 3.31 displays a trace showing the CPU activity and power dissipation during a NOR flash filesystem copy operation. This data corresponds to the previously described NOR versus NAND energy storage comparison in Section 2.4.2.1. As was shown in Section 3.2.2.2, the selection of not only file system, but data copy method can have significant impacts on energy dissipated by the CPU and memory subsystems. *LTT* enabled with energy tracing features when coupled with kernel system call trace information was critical in finding these energy differences.

3.2.3 Power Scheduling Tools

While the previously developed tools are valuable for energy accounting functions, we still require a software method to activate the EMAP's power management scheduler feature. The current Linux power management framework allows suspend and resume operations of the entire platform, but it is not well suited to dynamic power management at the subsystem or device level. While the `/sys` interface exposes device power states, they cannot be operated from user-space operations. Likewise, the power management API cannot be operated at the *per bus* or *per class* level from within the kernel. These are set only during platform level suspend and resume functions. In addition power management events may only be handled by the applications CPU, which requires a complete resume of

the platform in order to process new wakeup events. To support a finer grained control as well as scheduled operation of components, we created a new power management utility, *emap-client*. The *emap-client* utility provides a script-based user input to generate complex power management features including:

1. Wall time based sleep and resume functions for each EMAP managed power domain.
2. Periodic or interval based sleep and resume for cyclic sensing and processing.
3. Asynchronous wakeup signal for event driven operation.
4. Cancellation of existing power management schedules.
5. User permission verification to prevent unauthorized schedule modification.

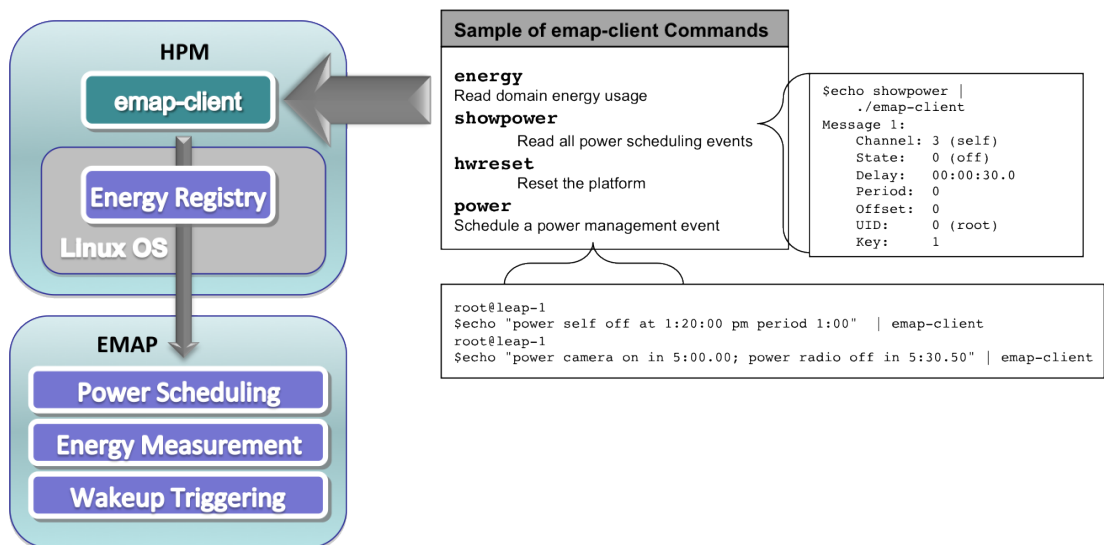


Figure 3.32: Example commands from *emap-client* demonstrating scheduling capability

Emap-client control is managed by a simple language based parser that allows users to provide complex scheduling actions. For instance script time references

are based upon Linux system time (and are compatible with standard *date* values) rather than referencing the EMAP's RTC value. The *emap-client* software automatically performs time conversion during schedule entry as well as conversion of RTC value to local time during display operations. In addition, power domains are referenced by configurable name handles (read from the configuration file) rather than using EMAP PMS register values. Finally, *emap-client* allows users to add custom reference handles for each entry. These reference handles may be used in later scheduling commands to modify or delete the existing entry, simplifying schedule maintenance. Figure 3.32 demonstrates some simple command activities using *emap-client* from the command line.

CHAPTER 4

LEAP Application: GeoNET

The previous chapter described the LEAP architecture including supporting hardware circuit schematics and performance measurements, energy profiling and scheduling software tools, and a complete reference platform design. We now provide a case study demonstrating a LEAP-enabled application in a commercial product offering. We begin with a detailed description of this platform and its capabilities. This is followed by a presentation of experimental data collected during numerous field trials lasting over one year in duration. The data includes a validation of LEAP capabilities enabling *in field* refinement of the energy usage while operating in *continuous deployment*, including platform software modifications providing subsequent increases to battery lifetime. We believe the ability to perform this in field optimization for platform power dissipation to be a unique contribution of the LEAP architecture.

Autonomous sensing systems have become increasingly important in environmental applications as a method to collect and process data for detection of specific phenomena. One important environmental application is the detection of seismic episodes following a major catastrophic event. An urgent need has arisen, in the wake of the 2011 Japan earthquake measuring 9.0, for a rapidly deployable, battery operated, aftershock detection capability. This new detection system would be deployed following large earthquakes, when much of the existing infrastructure may be damaged or destroyed. Its purpose is to alert

first responders of approaching aftershocks and allow their evacuation from unstable or dangerous locations prior to the arrival of new aftershocks. The early warning capability was lacking during search and rescue operations performed in several of Japan's major metropolitan areas. In the months that followed the initial earthquake an additional 1000 aftershocks were felt including a 7.7 and 7.9 magnitude on March 11, 2011 [Wik11], as shown in Figure 4.1. Each of these had the potential to create deadly repercussions for the thousands of first responders searching for survivors.

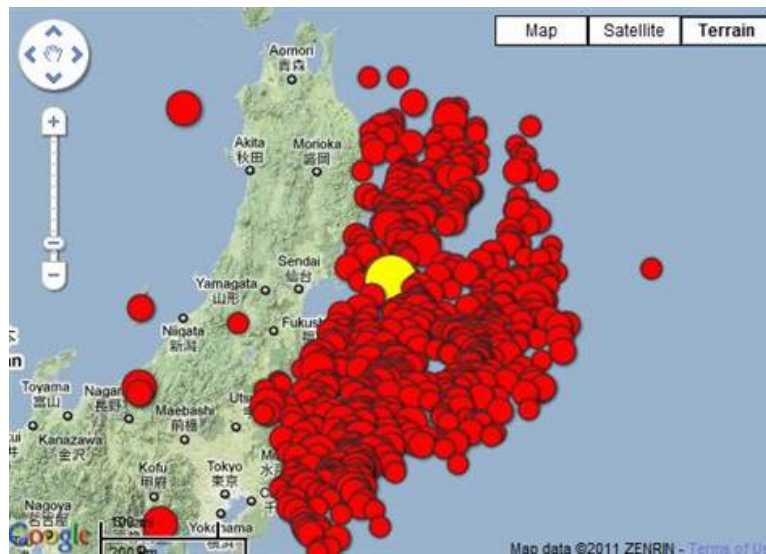


Figure 4.1: Map of aftershock activity near Japan as of March 31, 2011

The specific challenges of this pre-aftershock warning system impose a unique and difficult set of design requirements. First external power sources and other infrastructure will be unreliable in the aftermath of the primary earthquake event. Therefore these systems must be capable of operating for several weeks to a month (during search and rescue operations) on battery power alone with only a small solar panel for secondary charging possible in some situations. However, in order to provide a rapid deployment capability, strict limitations on size and weight are needed, greatly limiting the size of the battery allowed.

Conversely seismic applications require extremely high precision amplifiers and high fidelity sampling components. These are necessary for effective event detection [TEK] in the presence of noise from natural phenomena such as rainfall, wind, and lightning. Deployments in urban environments during aftershocks pose additional challenges due to manmade vibrations from pedestrians, vehicles, and other heavy equipment. This challenge is additionally compounded in an aftershock detection situation due to the need for rapid deployment. This prohibits time consuming placement of the geophone sensors thus degrading further the signal to noise ratio of the sensor.

The previous constraints imposes a critical energy budget challenge, as little power remains for critical tasks such as processing and communication. However reliable event detection in the presence of a wide variety of natural and man-made noise sources requires sophisticated algorithms to reduce false alarms. In addition, an ad-hoc wireless communications network is necessary to coordinate event detection as well as distribution of event alerts. Due to the propagation speed of the aftershock event this limits the maximum warning time to only a few seconds imposing severe limitations on network formation and alert propagation through the network. Transitioning the system from lower power detection operations to fully vigilant during alert distribution must be rapid with a minimum time necessary for route learning and path establishment.

To address these design challenges, UCLA's Center for Embedded Networked Sensing (CENS) has been working with Refraction Technologies Inc., a leading seismic equipment supplier, to develop a novel aftershock detection and alert capability specifically designed for this urgent application. This *GeoNET* project incorporates LEAP capabilities to provide a low average energy sensing capability while enabling robust sampling algorithm development and heterogeneous

communications capabilities. As part of defining the GeoNET objectives, we defined a specific set of platform requirements based upon contributions from field experts from both the Geology and Civil Engineering departments. Table 4.1 lists these platform criteria.

4.1 GeoNET Design

The GeoNET platform, a custom configuration of a Refraction Technologies RT155, is a ruggedized, all-weather sensing system designed for rapid deployment and long term autonomous operation. The RT155 may be equipped with a variety of seismic sensors including both passive and active types. The standard GeoNET configuration of the RT155 includes a 3-axis L-4 passive geophone sensor. The standard GeoNET configuration of the RT155 includes a 3-axis L-4 passive geophone sensor. The GeoNET enclosure is shown in Figure 4.2-a.



Figure 4.2: GeoNET (RefTek RT155) product photo (a) and internal bays (b)

4.1.1 Hardware Architecture

The RT155 internally houses several modular bays supporting different power, sensing, and processing modules to suite a wide range of applications. The inter-

Requirement	Specification	Details
No. Channels	3 (6ch option)	Support passive and active sensors
Sample Rate	200sps	
Resolution (Dynamic Range)	24-bit (135.5dB)	
Power Input	200mW ave. 1W	For 3 passive channels For 3 12V channels
Power Output	$\pm 12V$	Geophones (passive), Episensor 12V, LVDT 6-30V, Non-Contact Dis- placement 10-18V
Radio	>100mW Output	Able to communicate over 0.5-1km
Timestamp	GPS and RBS ready	Option to obtain GPS coordinates
Timing Error	<0.1ms	
Event Detect	>60s pre-event >120s post-event	Trigger store-and-forward
Buffer Size	8MB	3ch x 200sps x 24bit 6.5MB/hr Buffering to capture data for 15-20min Concurrent buffering and data transfer without inter- ruption of data collection
On-site storage external storage	1GB	Enough for 1 week without power \rightarrow 168hr 1.1GB
Overall Dimensions	25x25x15cm	Rugged IP67 box Includes CPU housing and battery

Table 4.1: GeoNET design requirements list

Slot ID	Module	Components	Function
1	CPU	RT619+HPM	CPU and communications module
2	PWR	RT618+RT620	Power module and battery charging
3	AD	RT618+RT617	Three channel analog data sampling
4	AD	RT618+RT617	Optional module to extend 3 to 6 channels

Table 4.2: GeoNET configuration of the RT155’s modular bays.

nal view of the RT155 is shown in Figure 4.2-b. The three modules in the figure can be found along the bottom of the enclosure. The GeoNET configuration loads the RT155’s bays with the hardware listed in Table 4.2.

The RT619 module, located in slot 1 of the GeoNET configuration, is a custom LEAP-based design, derived from the EMAP2 circuit board described in Section 3.1.5.3 and accompanying HPM connector. This design was provided to RefTek who fabricated and assembled the RT619 module as part of GeoNET product development. A photo of the RT619 circuit board with an installed HPM is shown in Figure 4.3 along with a circuit block diagram. The remaining slots in the



Figure 4.3: RT619 module photo

GeoNET configuration of the RT155 are installed with a power supply module (PWR module), and a high fidelity 3 channel analog to digital sampling module (AD module). The power supply module supports battery charging from external DC supplies or solar panel arrays. The AD module includes a state-of-the-art Texas Instruments ADS1282 sampling engine. We provide measurement plots of

the AD module’s noise floor and spurious free dynamic range (SFDR) in Figure 4.4 for reference. It is important to note the dynamic range of the ADS1282, over *110dB* SFDR including harmonics and *150dB* excluding harmonics. Additionally AD module noise floor is near $-120\text{dB } V^2/\text{Hz}$, necessitating 24-bit or larger sample sizes. As was demonstrated in section 2.2.1, we reiterate the larger data types are most efficiently handled by higher performance computation resources so long as the idle power can be handled effectively.

The RT155’s modules are connected through a high density ribbon cable with shared power, ground, and data signals. Analog sample data is passed through the ribbon cable in a daisy-chain using a round-robin token passing method to manage arbitration for the source module. This allows multiple AD modules to be connected in series and seamlessly extends the number of channels through adding additional AD modules into the RT155 bays. At each sample interval, the AD modules generate time stamp information and sample data. These are then transferred via the serial bus in a fixed priority sequence. Each 32-bit word in the data stream contains a unique identifier tag. The details of the token passing and serial frame types are shown in Figure 4.5.

4.1.1.1 RT619 Design

We now highlight the important distinctions of the RT619 from the prior LEAP2 design described in Section 3.1.5.2. For detailed design information on the RT619, please refer to Appendix A. While the RT619 design was leveraged from the EMAP2, several enhancements were added to support the GeoNET application. We provide a block diagram of the RT619 with new GeoNET functions highlighted for reference (Figure 4.6). Specifically, we added a high precision time reference via a TCXO with voltage control pin. The 16.384MHz TCXO output

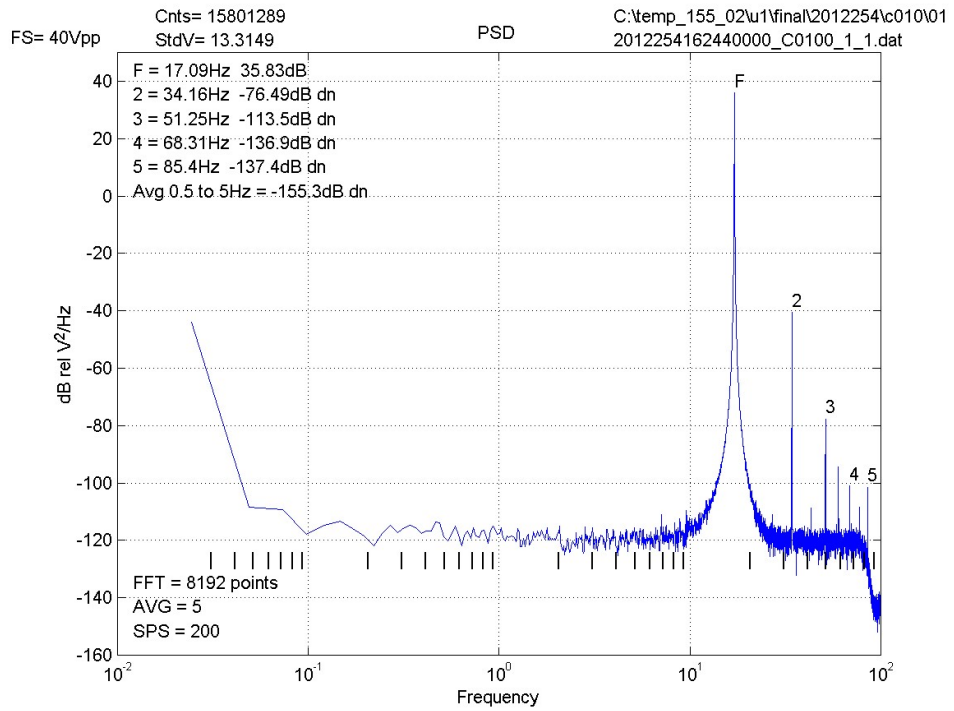
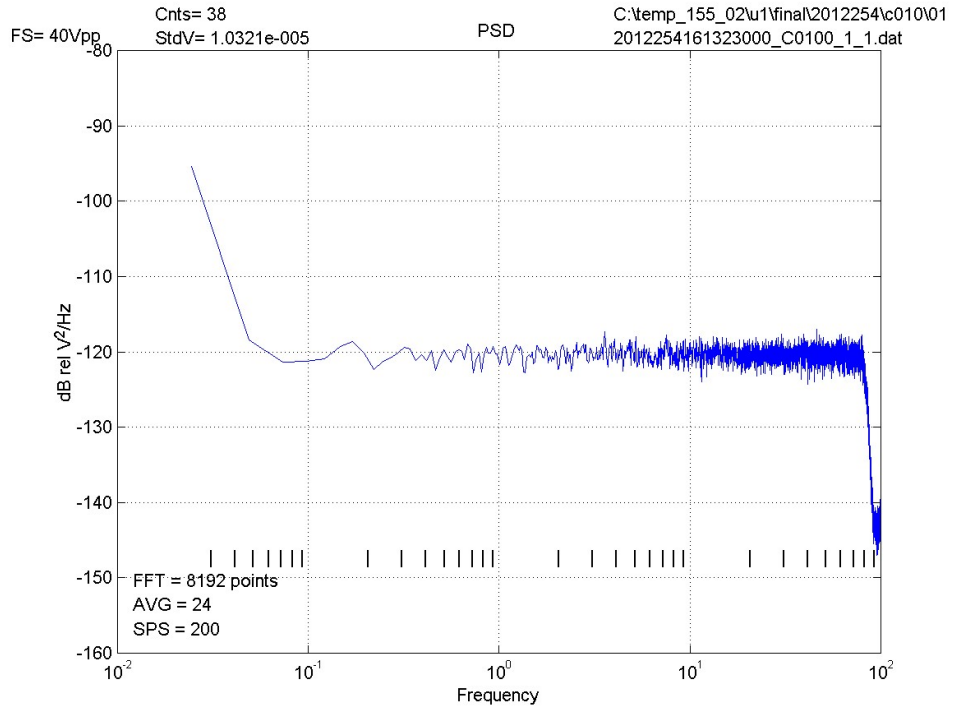


Figure 4.4: GeoNET RT614 Analog module performance measurements: (a) PSD of the noise floor from DC to 100Hz and (b) dynamic range of a 17.08Hz tone

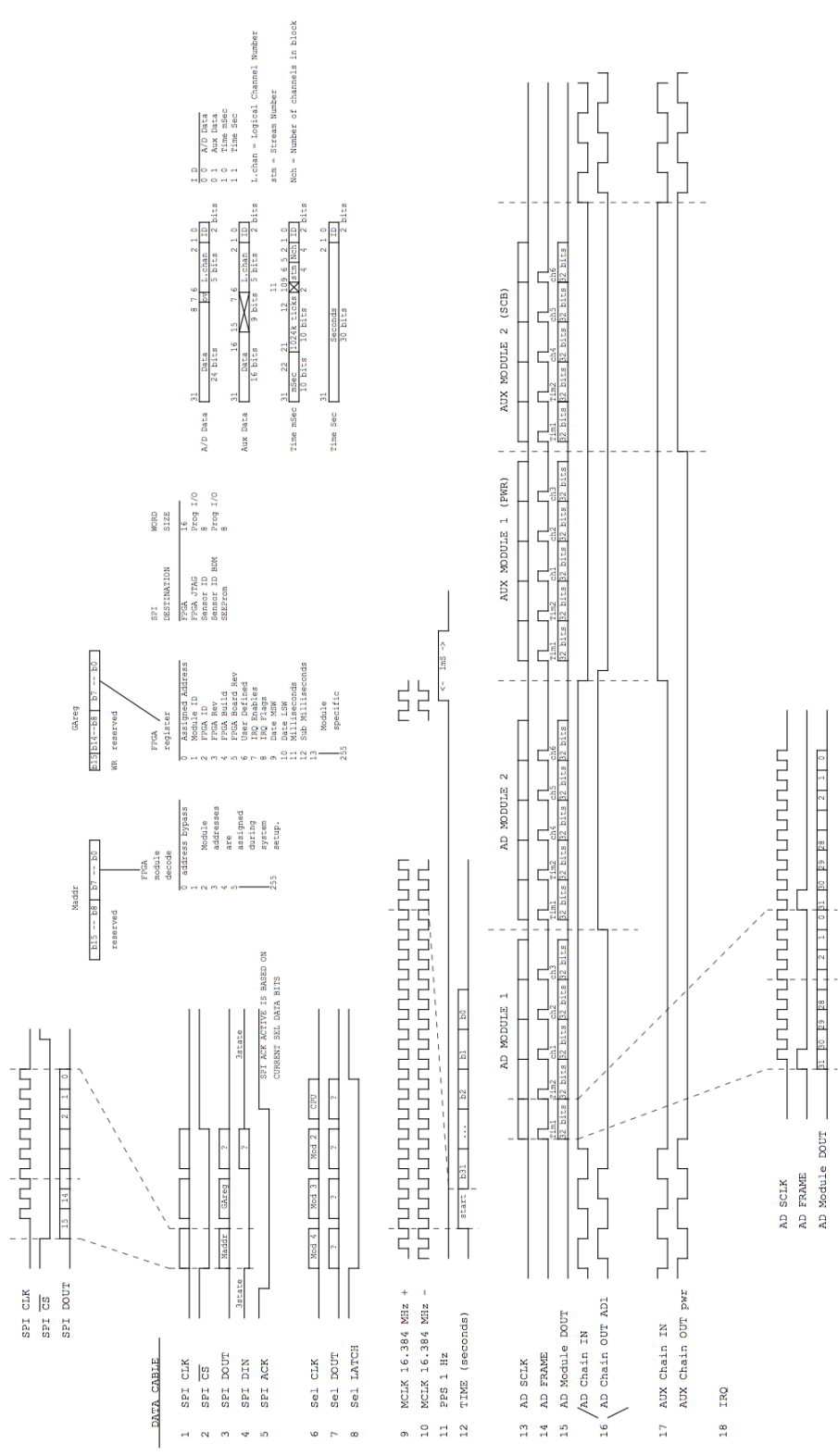


Figure 4.5: RT155 Backbone Serial Data Protocol

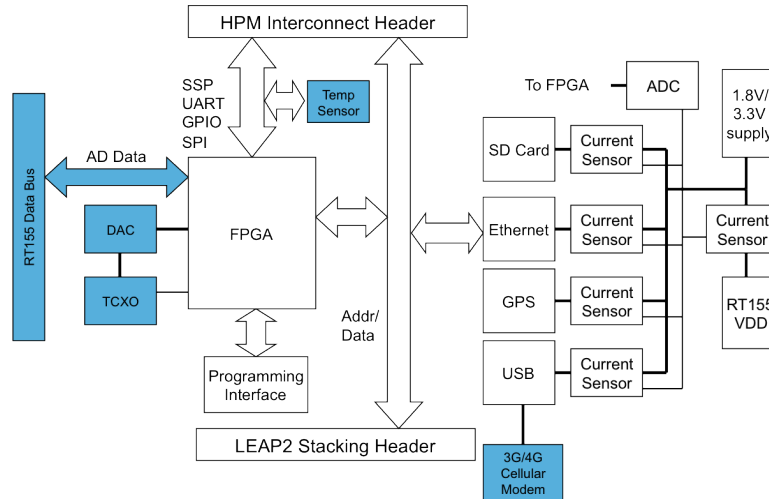


Figure 4.6: RT619 architecture highlighting additional GeoNET capabilities

may vary up to $\pm 10ppm$ over the specified input voltage range. The TCXO voltage control pin is driven by a 16-bit DAC reference that is controlled by the RT619 FPGA and the TXCO clock output fed back into the FGPA. This feed-back circuit is used to provide local time synchronization such that local time may be precisely aligned to a GPS pulse-per-second signal or other precision reference. We provide a more detailed description of the software controls and time synchronization circuits in Section 4.1.2.2. The RT619 additionally supports the new RT155 backbone data bus via a common ribbon cable connection between the different modules. The RT155's data bus connection on the RT619 is routed directly to the FPGA for local data buffering and processing. A diagram noting new GeoNET FPGA enhancements is shown in Figure 4.7 and now described.

4.1.1.2 Data Sampling, Buffering, and Event Detection

The RT619's FPGA provides two data paths for sample collection through its Data Acquisition Unit (DAU). This is managed through Wishbone accessible configuration registers in the DAU. First the DAU provides a direct synchronous

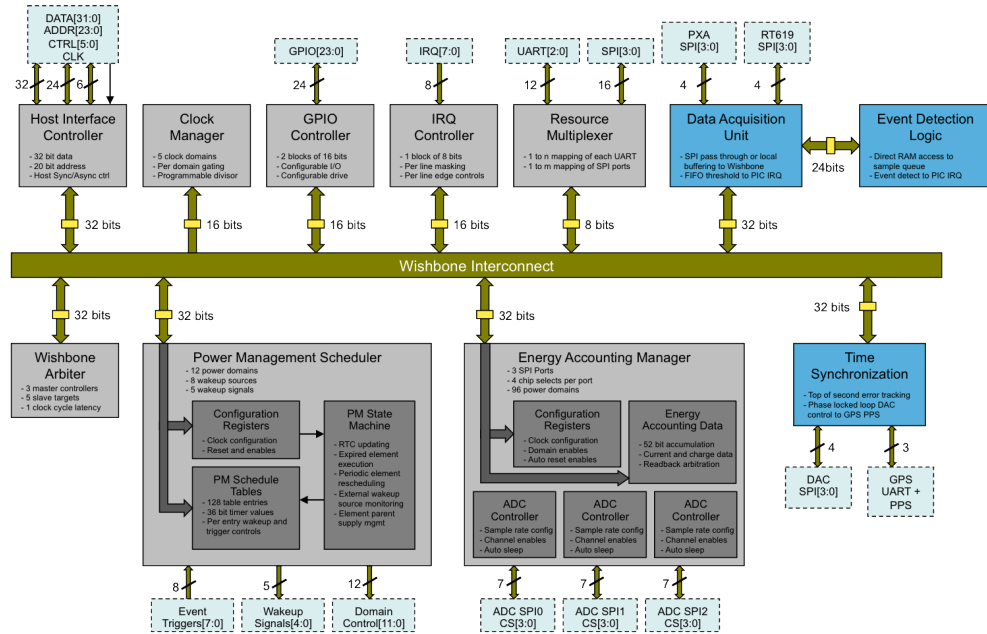


Figure 4.7: GeoNET FPGA architecture highlighting additional sensing capabilities

serial bus connection between the RefTek AD data backbone and the applications processor's on-chip peripherals, enabling a low latency data collection. This is beneficial when buffering delays prevent accurate inter-unit timing. Alternatively samples may be buffered within the FPGA's internal RAM blocks via a serial-to-parallel conversion and subsequent FIFO write operation. A FIFO configuration register sets the threshold value, above which IRQ events are sent to the applications processor. A diagram of the DAU is shown in Figure 4.8 for reference. The threshold IRQ is also routed to the FPGA's power management unit to provide wakeup events for registered power domains. In addition to the FPGA's data sampling and buffering function, the DAU provides an interface to optional event detection blocks. The event detection interface is clocked by both the high speed TCXO as well as a top-of-second indicator facilitating coordinated inter-node event timing. This module tightly coupled to the data buffer RAM, including single cycle access latency to any RAM address. This permits

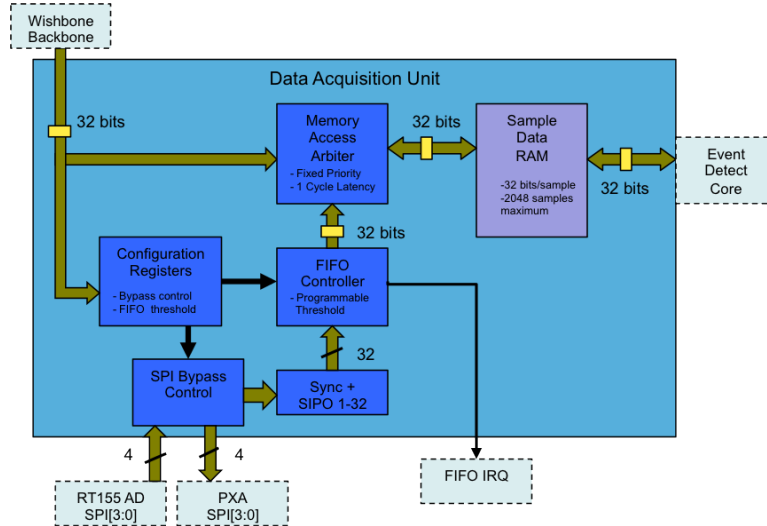


Figure 4.8: Diagram of the Geonet FPGA's Data Acquisition Unit

algorithms requiring multiple passes over RAM data in addition to those needing sequential access. Several event detection implementations are described in the later experimentation section.

4.1.1.3 Time Synchronization

The RT619's FPGA also manages the RT155 time reference signal through the previously mentioned TXCO and DAC voltage control. This reference signal is routed through the RT155 backbone to provide a synchronization signal across each of the AD sampling modules. We implemented a software managed frequency control loop using the FPGA to provide precision error estimation of the TXCO frequency using the GPS's top-of-second time synchronization signal. We have qualified the measurement accuracy as follows: The Ublox LEA-4T GPS module derives its synchronization timing signal from the module's internal 24.103 MHz oscillator. We have also measured an average rising edge pulse delay of 6ns on the RT619 circuit boards, dictating a module timing uncertainty

of $\pm 47ns$. When combined with the local TXCO's frequency of 16.384 MHz, our phase error measurement is limited to $\pm 109ns$ resolution. We assume the cable delay to be identical between different units as to be neglected in the error calculation.

The FPGA's time synchronization circuit provides two distinct modes of operation: frequency tracking and phase tracking. The frequency tracking circuit is used during initialization and allows rapid convergence of the TXCO frequency to the top-of-second signal. In this mode, the rising edge of the top-of-second signal generates frequency error estimation in TCXO clock cycles, and resets the count value to realign the local counter with the synchronization signal. Since frequency errors may be large during this mode, the phase offset is not considered. Once frequency offset has converged below a minimum threshold, phase tracking is enabled. In phase tracking mode the top-of-second signal no longer resets the TXCO count value. Instead the phase error continues to accumulate in the offset register. A separate top-of-second event counter is then enabled to provide frequency error calculation as the phase error accumulates. A software phase lock loop, described below, is used to modify the DAC voltage, setting TXCO frequency, based upon periodic phase error measurements.

4.1.2 Software Architecture

The RT619's addition to the RefTek RT155 platform enables highly sophisticated sensor data collection, processing and communications methods. These new methods are provided through our development of new software applications and platform services. These new services include:

1. Provide platform health monitoring such as battery charge state and wireless interface connection status, including emergency preservation modes.

2. Provide specialized time synchronization software including custom device drivers enabling the FPGA frequency control circuit and software phase locked loop.
3. Provide data collection services to acquire geophone samples from the RT155's AD modules and format this data for later post-processing.
4. Perform detection of seismic events by selecting from a library of available methods and algorithms (both on the CPU and FPGA).
5. Enable rapid reporting of events through established infrastructure methods such as email and SMS.
6. Perform synthesis of event data into formats consistent with rapid user response.
7. Enable resource power scheduling.
8. Provide platform software management with in-field updates.

As part of the GeoNET software development, each of these services is provided through a combination of custom and open source software. Figure 4.9 provides a simplified diagram of the primary RT619 software applications. Each is described in the following sections.

4.1.2.1 System Health Monitoring

A fundamental operation in any remotely deployed system is persistent vigilance of platform health. As GeoNET devices are expected to be left unattended for many weeks, health monitoring is a critical concern. We developed a specialized health monitoring daemon *sysmon*, which provides continuous verification of battery health and wireless communications link state. Failures of internal software

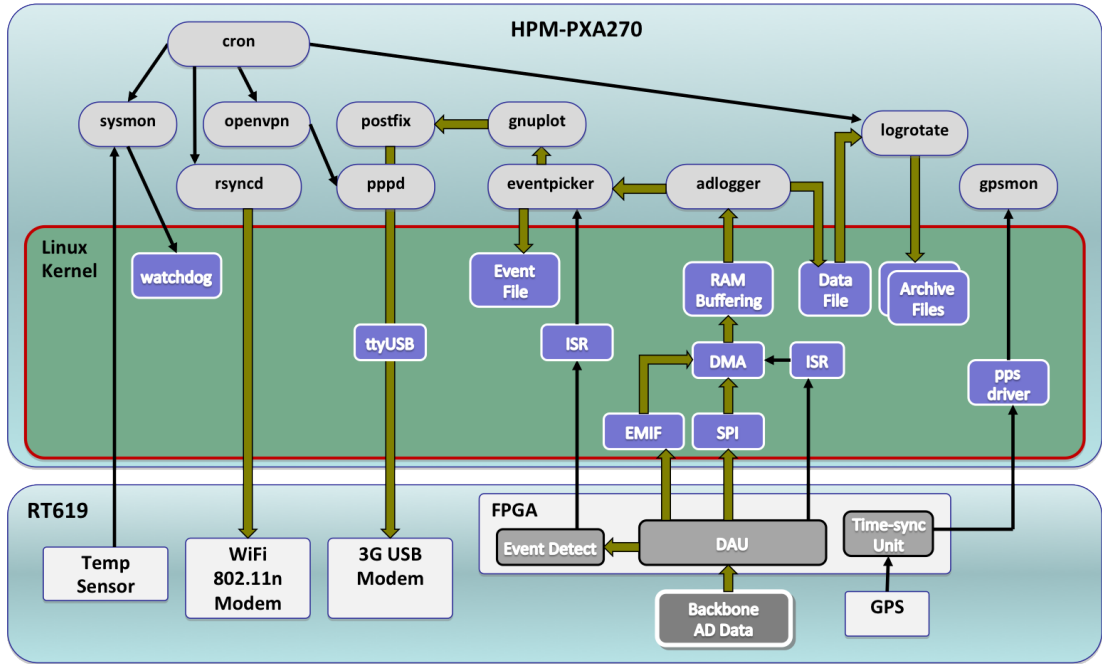


Figure 4.9: Diagram of RT619 software applications

services or hardware triggers automatic watchdog rebooting of the platform to avoid deadlock situations. As part of battery health monitoring, should *sysmon* measure the battery voltage at critical levels while battery charging is disabled, *sysmon* forces entry into emergency shutdown state. In this state all software services are terminated to reduce immediate power consumption. Following this, as imminent malfunction or failure is likely, *sysmon* schedules a future wakeup time to restart the unit into the FPGA’s PMS table. *Sysmon* then shuts down power to all domains other than the FPGA. At the prescribed daily wakeup time, scheduled to corresponding with the maximum solar charge, the PMS restarts the platform. Once booted, *sysmon* performs a retest of battery state and reports current health information to the remote server. This continues until sufficient charge in the battery disables the emergency shutdown state.

4.1.2.2 Time Synchronization

As noted in our requirements (Table 4.1), inter-node timing error may not exceed 0.1ms, yet GPS enabled time synchronization on the RT619 provides significantly better timing accuracy. However, generating the GPS module's timing signal requires significant energy. If we instead rely upon the RT619's on-board TCXO, sufficiently conditioned by the FPGA to match the GPS's synchronization signal via a periodic realignment, we may achieve a net power savings. Since the TCXO clock error as referenced to GPS, may be measured in the FPGA we utilize standard phase lock algorithms for local TCXO frequency adjustments. During periods when local clock error is less than our defined maximum, the GPS module is shut down to reduce power. Our time synchronization is managed through three separate components: the FPGA's time synchronization module, a time synchronization Linux kernel driver, and a time synchronization daemon *gpsmon*. The FPGA performs error estimation via clock difference measurements between GPS and the local TCXO as previously described. The time synchronization kernel module provides FPGA status information to the *gpsmon* daemon including periodic calculations of drift rate and current error in parts-per-billion. This kernel information is periodically refreshed via a timer thread. An example of the kernel's provided information is shown below displaying FPGA's time synchronization status information.

Listing 4.1: Example data from */proc/leap/energy* file showing power domains and energy dissipation

```
AD Time: 1318565717.664
Local Time: 1318565718
Time difference: -.336
PPS Error: no
```

```
Time Sync:  enabled
Lock Mode:  phase
PPS Drift:  +0.206ppm
Time Error:  61ns
DAC Value:  39040
```

Gpsmon reads time synchronization state from the kernel driver through the `/proc/leap/timesync` interface and adjusts the DAC feedback voltage to the TCXO. Once phase error is reduced below a specified threshold *gpsmon* switches off the GPS unit for a prescribed period of time using various algorithms based upon the last frequency and phase error estimations. We do not devote considerable time to list these algorithms here, but provide the measured energy efficiency for these in the field experiments in the next section.

4.1.2.3 Data Collection

GeoNET's data collection service provides continuous buffering, time stamping, file compression, and archiving operations for samples from the AD modules. The process uses several software applications to fulfill the service. Figure 4.9 shows the architecture of sampling process. The RT155's AD data arrives on the backbone bus at the FPGA where it is delivered to the HPM in one of two methods. If configured for serial pass-through, the FPGA forwards the AD data directly to the HPM's serial data bus. The application's processor DMA is used to transfer data from the serial bus into a CPU local memory buffer. If configured to buffer AD data within the FPGA, the applications processor is idle until the FPGA sets the IRQ indicating the FIFO fill level is reached. This wakes the CPU out of idle to setup a DMA transfer using the host memory bus rather than the

serial data bus. Data is copied via the DMA to a CPU local memory buffer.

After DMA transfer completes, the kernel driver receives an IRQ from the DMA. This causes the kernel driver to wake the reader application *adlogger*, which is blocked waiting for new data to arrive. *Adlogger* daemon reads the buffer data via a device interface file, */dev/ad*. The daemon converts the raw RT155 formatted binary data to one of several output formats including binary integers, floating point, comma separated text, or SQLite database format. After converting sample data to the appropriate format, *adlogger* writes to a storage file whose location is determined by the daemon's configuration file.

Since the GeoNET application requires data to be sampled continuously for long periods, we provide a service to deal with filling available storage resources. We use a two stage data archiving mechanism to prevent data loss. The first manages data archiving using an open source log file storing utility *logrotate*. We configured *logrotate* to provide an hourly file archiving service, where at the start of each hour, the current data file is terminated and a new empty file is opened for *adlogger* to continue storing data. Once the previous file is terminated, *logrotate* renames the file with epoch time stamp information and then compresses the file using *gzip* or *bzip2* before moving this to the long term storage location, typically on an SD Card or USB flash drive if installed. The second stage of the archiving application provides graceful retirement of old data files. Once the filesystem reaches a preset capacity, the oldest files are removed to provide space for new files.

4.1.2.4 Event Detection

In addition to data collection and archiving, GeoNET provides an event detection service. Event detection is handled via a specialized daemon *eventpicker*.

The *eventpicker* daemon is an algorithm agnostic, providing hooks to support both software based event detection methods as well as FPGA based detection. *Eventpicker* receives a continuous stream of data from the *adlogger* daemon by reading directly from the current data file by opening it as read-only. Upon each new data read, *eventpicker* informs one or more detection algorithms of the data arrival by passing this through a shared memory object. If either an interrupt signal from the FPGA hardware or a software signal from an event the event detection software indicate an event arrival, *eventpicker* initiates the event reporting mechanism by first extracting samples from *adlogger's* data file corresponding to a time window around the current time. This is set via the *eventpicker* configuration file. Once the event file has been extracted, *eventpicker* initiates the event reporting mechanism described next.

4.1.2.5 Event Reporting

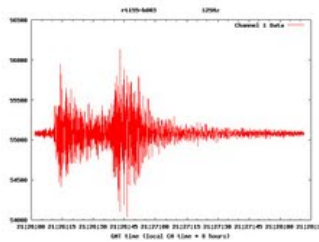
The event reporting service is composed of a series of activity scripts executed by *eventpicker* using the Debian *run-parts* utility. These activity scripts provide a simple method to add and remove actions initiated by local event detection. An example of standard GeoNET event reporting activities is demonstrated in Algorithm 2.

The alert sequence shown in Algorithm 2 may be augmented with additional activity scripts, such as supporting SMS messaging via the Linux *smsclient* application. When enabled, the *smsclient* activity script sends a copy of the message body from the email alert activity, using an SMS text message, to the list of phone numbers contained in the *sms-list* file. Figure 4.10 demonstrates the email reporting format for a distributed event detection using two a two unit collaboration.

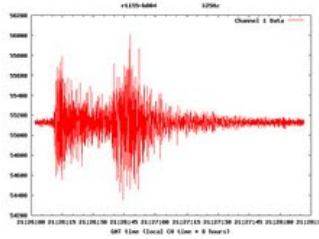
rt155-b003 leap.geonet@gmail.com 11/22/11
to dustin.mcintire, stubailo

An event was detected by rt155-b003 on Mon Oct 17 21:26:13 UTC 2011. An image and binary data file is attached. Location: [http://maps.google.com/maps?f=q&source=s_q&hl=en&geocode=&q=34.09486766,-118.65631550+\(rt155-b003\)&aq=&sl=34.09486766,-118.65631550&sspn=0.021458,0.045447&ie=UTF8&t=h&z=14](http://maps.google.com/maps?f=q&source=s_q&hl=en&geocode=&q=34.09486766,-118.65631550+(rt155-b003)&aq=&sl=34.09486766,-118.65631550&sspn=0.021458,0.045447&ie=UTF8&t=h&z=14)
An event was detected by rt155-b004 on Mon Oct 17 21:26:13 UTC 2011. An image and binary data file is attached. Location: [http://maps.google.com/maps?f=q&source=s_q&hl=en&geocode=&q=34.09399950,-118.65735766+\(rt155-b004\)&aq=&sl=34.09399950,-118.65735766&sspn=0.021458,0.045447&ie=UTF8&t=h&z=14](http://maps.google.com/maps?f=q&source=s_q&hl=en&geocode=&q=34.09399950,-118.65735766+(rt155-b004)&aq=&sl=34.09399950,-118.65735766&sspn=0.021458,0.045447&ie=UTF8&t=h&z=14)

5 attachments — [Download all attachments](#) [View all images](#)



plot-rt155-b003.jpg
52K [View](#) [Download](#)



plot-rt155-b004.jpg
59K [View](#) [Download](#)




-  **event-data-rt155-b003-1318886773.bin**
191K [Download](#)
-  **event-data-rt155-b004-1318886773.bin**
191K [Download](#)
-  **plot-bin-data.gnu**
2K [View](#) [Download](#)

Figure 4.10: Example of email report generated during El Centro deployment demonstrating display of distributed detection from two units

4.1.2.6 Configuration and Management

In order to remotely monitor unit operational status and energy efficiency performance, a remote configuration and management service is installed on GeoNET. This service is comprised of several distinct operations including:

1. Generation of application state, power profile, battery status, event detection log, and time synchronization log files.
2. Perform log file compression and archiving.
3. Transfer log archives via *WiFi* to gateway device.
4. Synchronize current event files with remote server.
5. Synchronize sample data archive with remote server.
6. Receive new commands from server for execution.

To minimize the significant energy needed to maintain the wireless communications interfaces, these devices are scheduled for only minimal operation; the objective being to sufficiently inform a remote observer of network status and operational performance. We use a combination of several open source applications such as *cron*, *logrotate*, and *ppp* in coordination with the FPGA power management scheduling capability, to enable the wireless interfaces only periodically to fulfill the tasks outlined above.

Since each GeoNET unit is time synchronized to a fine-grained level, we coordinate operation of the *WiFi* devices to rapidly establish an ad-hoc network. Once the network has formed, each remote unit synchronizes log file, event file, and optionally sample data archives with the gateway unit. After remote units have synchronized with the gateway unit (or a timeout action occurs), the *WiFi*

interface is disabled. Gateway units then initiate a cellular connection to a remote server using *openvpn* to create a tunnel through *ppp*. The VPN tunnel provides a secure and reliable method for remote operators to access the gateway units. This is necessary since many cellular providers do not allow inbound connections through their network firewall. The *openvpn* tunnel enables direct *SSH* access to the gateway unit from the server providing interactive command and control operation.

After successfully creating a VPN connection, the gateway unit synchronizes its log files, event files, and (optionally) sample data archives with the server into a unique *outbox* directory. This synchronization includes any data from remote units that was synchronized to the gateway over the *WiFi* link in the preceding time window. Upon completion of the data upload, the gateway unit also retrieves new configuration data, schedules, and commands from its *inbox*. The inbox may include an *inbox/run* path to enable automatic executable or script operation.

4.1.2.7 Software Update

To emphasize dynamic in-field optimization we provide an over-the-air (OTA) method for continuous update of platform software and FPGA images. Application software is bundled using the OpenEmbedded *opkg* tool, which provides implicit versioning, merging, and installation capabilities. New software packages are deployed to the package repository, which may be hosted on any internet accessible server. During the previously mentioned synchronization of *inbox*, new software packages may be installed via *opkg* commands. This causes the new software packages to be downloaded over HTTP using *wget* into local flash memory. The package then installs new software according to the package's configuration data.

In addition to its application software, the RT619's FPGA is OTA reprogrammable. The FPGA's JTAG port is wired to the HPM's applications processor via GPIO lines, enabling in-field reprogramming of the FPGA's image. We created a custom Linux application *fpga-prog*, based upon Actel's *directc* source code, which reads the FPGA image from a local file and then writes this image via the JTAG port to the FPGA's flash memory. Upon completion of the erase, write, and subsequent verify stage, the FPGA restarts using the newly updated image. As will be discussed in the next chapter, FPGA reprogramming is essential to match the most energy efficient event detection method with a specific environment.

Algorithm 2 Event reporting pseudocode

```
1: Startup WiFi mesh network
2: Distribute local event file to all neighbors using rsync
3: Receive events from any neighbors with detections via rsyncd
4: if Device is cellular gateway then
5:   if Distributed event detection enabled then
6:     repeat
7:       Run event detection algorithm on neighbori data
8:       if Event time on neighbori  $\leq \pm\Delta t$  local event time then
9:         Increase likelihood
10:      else
11:        Decrease likelihood
12:      end if
13:    until All neighbor events processed
14:    if Likelihood < threshold then
15:      Cancel alert reporting
16:    end if
17:  end if
18:  Start ppp dialup of the 3G modem.
19:  Generate email report including event time, date, and geographic location
20:  Run gnuplot on local event file
21:  Attach local plot graphic to email report
22:  repeat
23:    Run gnuplot on neighbori event file data
24:    Attach neighbori plot graphic to email report
25:  until All neighbor events plotted
26:  Add email cc recipients from cc-list file
27:  Run postfix mailer to send email to leap.geonet@gmail.com
28:  Shutdown pop connection and disable 3G modem
29: end if
30: Shutdown WiFi mesh network.
```

CHAPTER 5

Field Experiments

In the preceding chapter we documented the unique capabilities of the Refraction Technologies RT155, including our custom RT619 module design, and its use in the GeoNET application. We provided a basic overview of the software and hardware components comprising the RT619 and its host processor module so that we may now address the value of energy-aware sensing. This chapter focuses on experimental data collected with our GeoNET devices across two long term deployments. These deployments, together accumulating over 9 months of in-field operation, were vital in validating an energy-aware design approach through iterative software development and deployment while operating in continuous deployment. Through the FPGA's novel EMAP operation, providing high-fidelity, fine-grained energy dissipation of each component operating in the GeoNET system, we were able to make substantive, iterative improvements to energy efficiency while maintaining high performance. This performance is shown through accurate, rapid event detection as well as sophisticated event reporting operation through feature rich applications.

Our field experiments were conducted at two sites. The first site was located outside El Centro, California, south of the Salton Sea and near the Mexican border. The second site was located at a Stunt Ranch, part of the Santa Monica Mountains Reserve near Los Angeles. Both sites were only remotely accessible through cellular connections and no infrastructure power was provided. In both

experiments the GeoNet units were equipped with 20W solar panel arrays and expected to be energy self-reliant. These panels were wired directly to the battery charger circuit accessed through the power connector on the RT155 lid. WiFi access between the units was established during site survey using high gain yagi antennas in order to provide sufficient range in rugged terrain.

5.1 El Centro

Our first experiment took place at the El Centro site during March 2011 through June 2011. The GeoNET units were installed along a remote section of Highway 78 approximately 50m from the side of the road approximately 350m apart. The unit map locations and photos of the installation sites are found in Figure 5.1 and Figure 5.2.

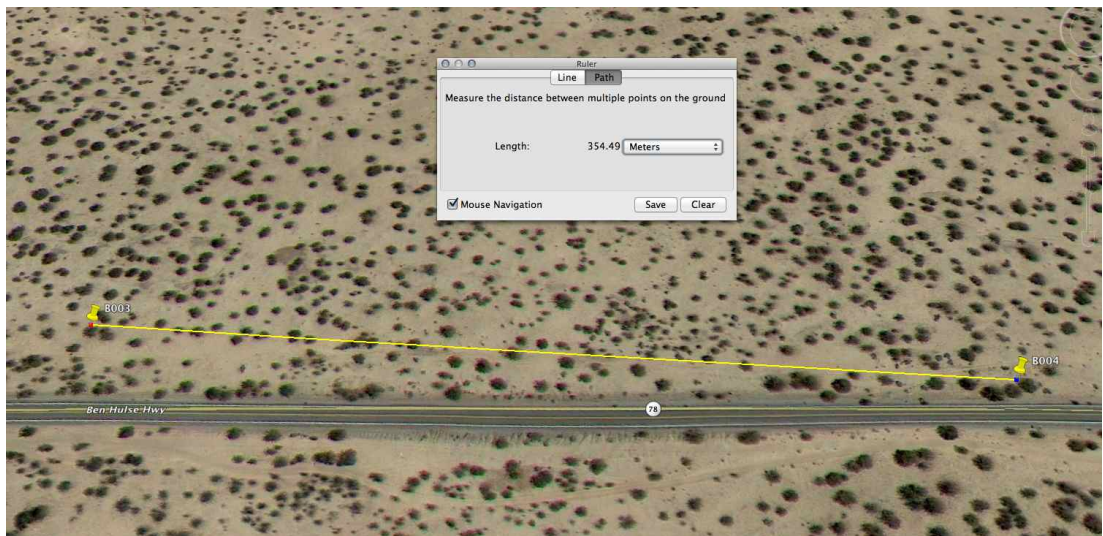


Figure 5.1: Locations of the B003 and B004 units installed in El Centro, CA



Figure 5.2: Photo of B003 and B004 unit installations at El Centro test site

5.1.1 Data Collection Efficiency

During the three month deployment, GeoNET units were equipped to continuously monitor energy expenditure of the GeoNET software applications using the measurement tools described in section 3.2. We periodically harvested energy usage information reported to the server in the intent of locating methods to increase platform efficiency. Our first experiment focused on investigating alternatives for the data collection process using the various output formats provided by *adlogger* and underlying flash storage media and file system times. During the experiment we modified the *adlogger* configuration script to output data samples in binary, CSV, and SQLite formats. Additionally we varied the storage file location across both a RAM backed tmpfs partition (*/tmp*), and a ubifs volume on the raw NAND device (*/mnt/nand*). We note that experiments to provide a direct write to the SD Card and USB flash drive were unsuccessful due to occasional extended write times on these devices due to background flash erase operations. The results of our experiment are found in Figure 5.3. The tmpfs storage provides lower energy write operations than the ubifs option across all three output

formats by approximately 10%. We also find that there is a significant energy cost to convert from RT155 binary format to either CSV text or SQLite database formats.

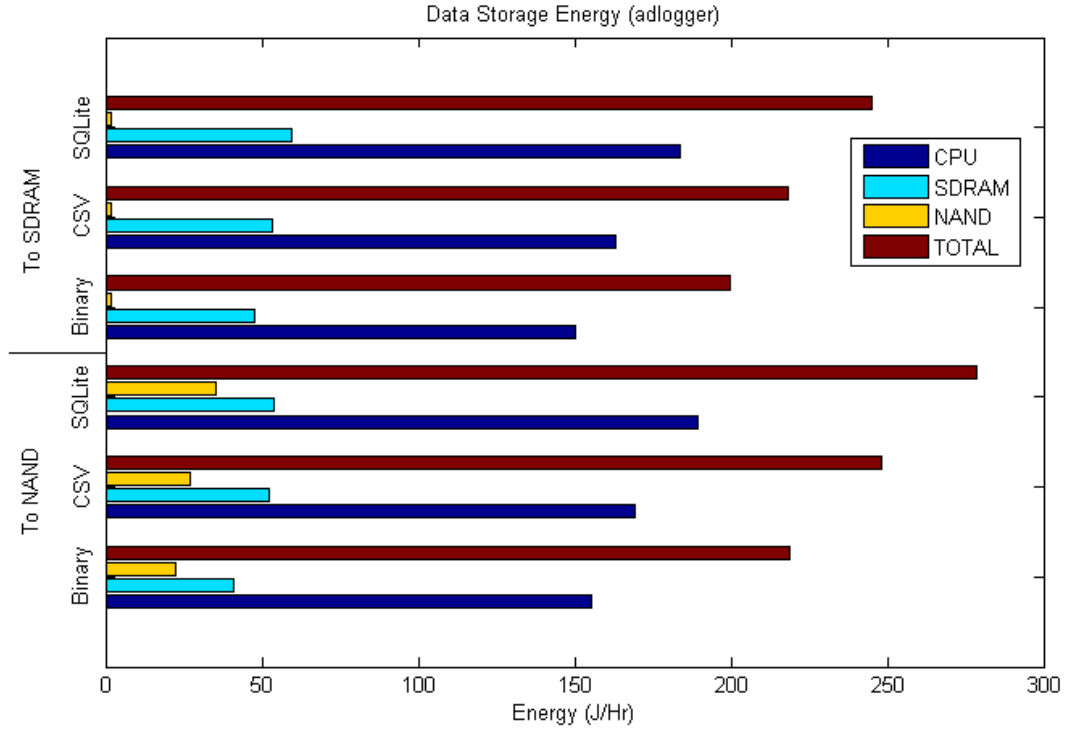


Figure 5.3: Energy consumed by adlogger to capture and store different format types and file systems

5.1.2 Data Archiving Efficiency

After *adlogger* stores converted data to its sample file, an archiving application, *logrotate*, will periodically read, compress and store sample data to a its archiving directory. We next test different options for the archive process through experimentation with different source file locations. For each we assume the destination location remains constant, a vfat formatted SD Card (*/mnt/sd*). The source data is located one of the following: a second file on */mnt/sd*, a ubifs volume residing

in the raw NAND device (*/mnt/nand*), or our tmpfs partition in */tmp*. As part of the experiment, both *gzip* and *bzip2* compression schemes were measured.

The results of this experiment are shown in Figure 5.4. The energy supplied to each of the hardware components that is directly attributable to the *adlogger* and *logrotate* processes as well as the file system read and write operations has been accumulated into the graph. For simplicity we attributed all of the energy consumed by the storage media to the *adlogger* and *logrotate* applications. This was a fair assumption since no file access is performed during normal operation beyond periodic log file generation and rotation. As expected the *bzip2* algorithm performs superior compression, approximately 40% file size reduction on average, but at the expense of significantly longer cpu execution times. This results in 2-3 times more energy expended using *bzip2* versus *gzip*. The expected return for higher data compression is a reduced transmission during file synchronization across the network and reduced energy for data transfer. This is measured as part of our distributed detection experiment.

5.1.3 Detection Efficiency

One unanticipated challenge at the El Centro site was due to the high volume of tractor-trailer traffic traveling Highway 78. This route is frequently traveled by trucks transporting agricultural cargo. The net result was a significant false alarm rate due to vehicle ground vibration rather than geologic phenomena. In the initial deployment, our GeoNET units were configured to do independent event detection and reporting without event collaboration. While sophisticated detections algorithms currently exist to discern vehicles noise from geologic phenomena [ZTX11] [BRV11], this is outside of our expertise. Instead we developed a distributed detection method to discriminate traffic noise based upon each unit's

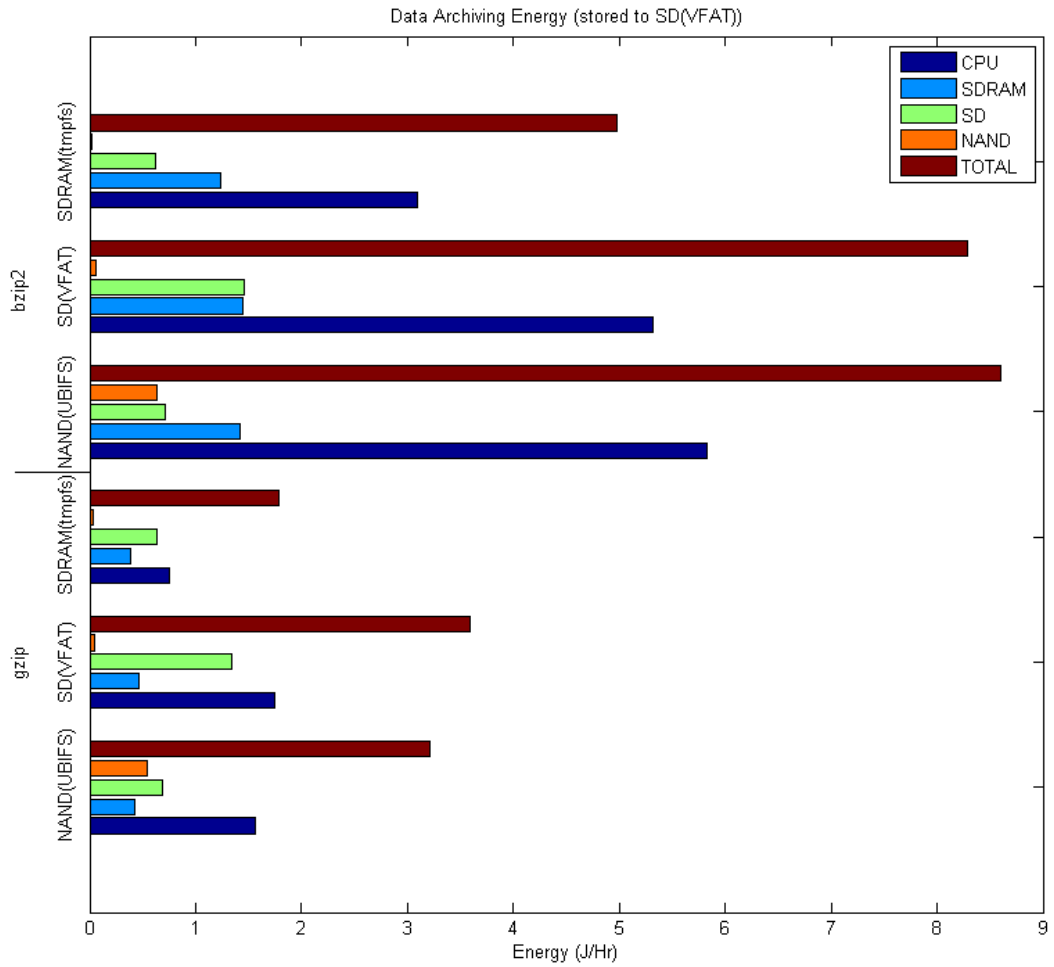


Figure 5.4: Energy consumed using different file systems and compression for data archiving

spacial diversity and correlated local timing references. This rests upon the observation that P-wave geological events travel at speeds of 5 to 8 km/s and will exhibit a time of arrival difference under 100ms at our node spacing rather than multiple seconds at vehicular speeds. Using this observation, a distributed detection algorithm was deployed in April with the object of decreasing false alarm rate. (Once deployed we then able to measure the speed of passing vehicles using doppler shift and the node separation distance). Our false alarm rate, shown in

Table 5.1, was greatly reduced however, the the distributed detection algorithm requires additional energy to manage event transfers from neighboring units. We next determine the effective energy *efficiency* of both the local detection and distributed methods based upon direct measurements of each approach.

Our efficiency metric is calculated as the net energy cost to perform acquisition and storage, detection, and reporting per each valid reported event. We neglect outside factors such as cellular or SMS usage costs in this metric. Our metric dictates we first measure the average energy costs for each of the services noted. The energy cost for *adlogger* measured in Figure 5.3 provides the first component. We next provide energy use data for several alternative detection algorithms. Each was implemented and deployed on GeoNET during our experiments. These algorithms correspond to generally accepted methods for geological events [MGP82] [WAY98] [HWL84] including a short-term to long-term variance ratio, finite impulse filtering, and Fourier transform. In order to further explore energy tradeoffs, we implemented each of the candidate algorithms both as software on the applications processor’s CPU and as an HDL component in the RT619 FPGA. The software implementations were incorporated into the *eventpicker* detection service while the HDL implementations operated from the FPGA’s event detection block. The performance of each algorithm is shown in Figure 5.5. As anticipated, the hardware based solutions outperform the software driven solutions by significant margins. However, the development time for each of the HDL based implementations was significantly higher than its software equivalent. Additionally, the most sophisticated algorithms, such as the Fourier transform based operation was not achievable within the available FPGA gates.

We now determine the energy cost for our final component, event reporting. Event reporting is the most challenging service to measure as it encompasses

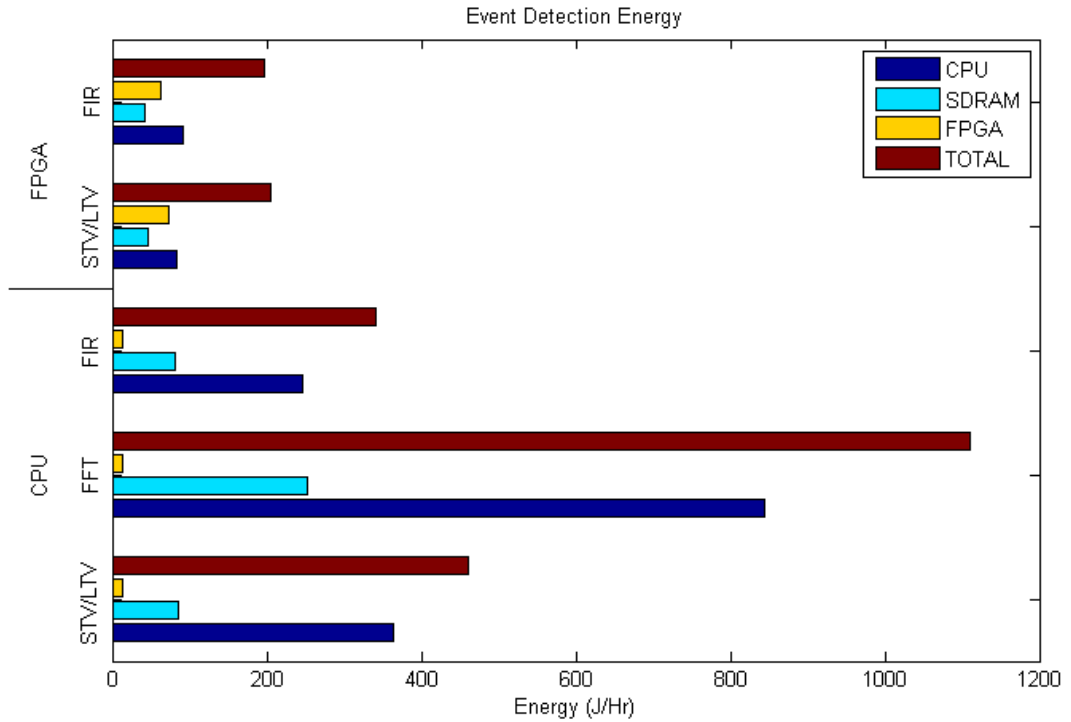


Figure 5.5: Energy consumed for several event detection methods

multiple software applications, data transformations, file operations, and data communications. The data flow through these services is complex and accurate energy measurement challenging. This tested our LEAP profiling software’s ability to track several simultaneously running applications and to differentiate energy expenditure across multiple platform hardware resources. Using *etop* as our measurement tool for applications software and *tracepoints* for energy dissipation due to file I/O operations and communications, we established the total energy cost to report a single event. The event energy total is the sum of the individual contributing applications and their resulting kernel operations. A summary of these operations is shown in Figure 5.6.

The energy efficiency provided by our local detection versus distributed detection methods is shown in Table 5.1. We provide the contribution of each service

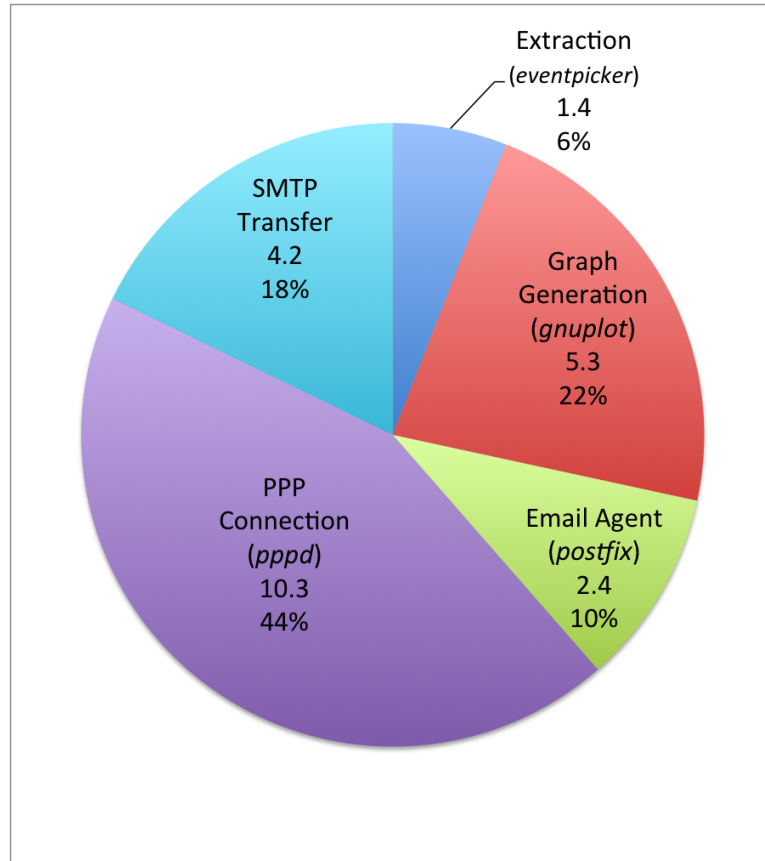


Figure 5.6: Energy contribution in joules from the component software applications used for event detection reporting

(data storage, event detection, and event reporting) to the efficiency calculation. The false alarm rate using distributed detection versus our most efficient local detection algorithm (STV/LTV) is given. While our local detection method provides lower energy event detection, it's higher false alarm rate causes significant wasted energy through erroneous event reports. The distributed event detection method performs with 23% less energy at the El Centro site. Interestingly this observation runs counter to common heuristics inferring local processing methods provide more efficient applications and would only be known through direct measurement as provided by LEAP.

El Centro Reporting	Detection Type	
	Local	Distributed
Detection Energy	461.5	627.4
Event Reporting	23.6	23.6
False Alarm Rate	94%	7%
Effective Detection Energy	855	653

Table 5.1: GeoNET Energy Efficiency during El Centro experiment assuming one valid event per day

5.2 Stunt Ranch

Our second experiment took place at the Stunt Ranch Reserve property from July 2011 through December 2011. The Stunt Ranch Reserve is located along a remote region of the Santa Monica mountains in Calabasas, CA. The property is rugged and undeveloped with no large public roadways. Figure 5.7 notes the GeoNET installation locations this site. Figure 5.8 provides a photograph of our installation including solar panel placement and antenna placement.

There are several key differentiators between the El Centro and Stunt Ranch sites. First the public roadway at Stunt Ranch is set back from the installation site by several hundred meters and is not traveled by large vehicles. There is no human development within several miles. Accordingly, we anticipated the false alarm rate to decrease as compared to the El Centro site. Second, the El Centro installation was in a flat barren desert area with little vegetation or other solar obstructions during the day. The weather was consistent, with mostly direct sunlight throughout the day. In contrast, our installation site at Stunt Ranch was located on the north facing slope of the mountain chain and exposure to direct sunlight was reduced significantly during winter months. The mountain chain typically experiences fog during the morning hours due to its proximity to the Pacific Ocean. We next describe how the GeoNET system was able to adapt to

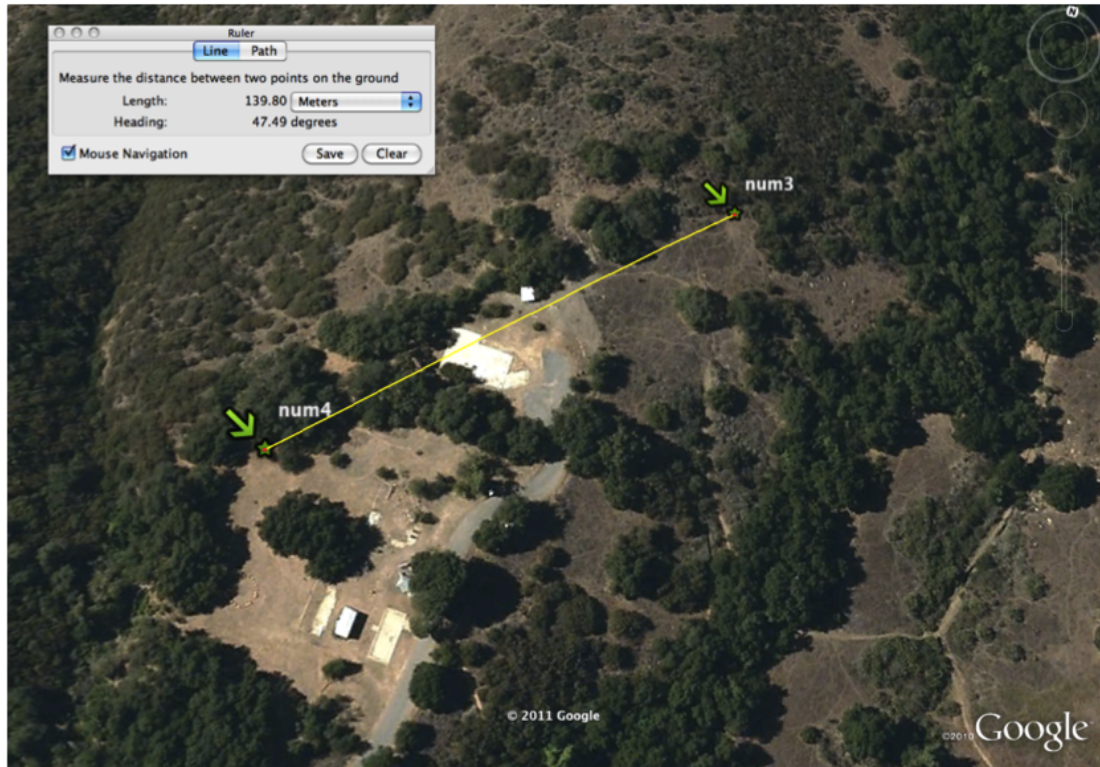


Figure 5.7: Locations of the two units installed at Stunt Ranch

these site differentiators during the course of the experiment through our in-situ energy profiling.

5.2.1 Detection Efficiency

We repeated an identical set of energy efficiency experiments as in El Centro in order to determine if our previous selection was best suited to the Stunt Ranch site. The results of these tests are provided in Table 5.2 and demonstrates that local event detection algorithm provides higher efficiency at Stunt Ranch. This was as result of two major factors. First, the lower noise level reduced the false alarm rate as expected (the small number of remaining false alarms were attributed to animal activity). The second factor is due to an improved cellular connection at



Figure 5.8: Photo of B003 unit installation at Stunt Ranch facility

Stunt Ranch Reporting	Detection Type	
	Local	Distributed
Detection Energy	461.5	627.4
Event Reporting	18.4	18.4
False Alarm Rate	53%	1%
Effective Detection Energy	501	646

Table 5.2: GeoNET Energy Efficiency during Stunt Ranch experiment assuming one valid event per day

Stunt Ranch. Its higher bandwidth and reduced power consumption resulted in a lower event reporting energy cost. (We discovered later the our site is near a Verizon 4G cellular tower.) These factors created sufficient influence to modify our detection solution.

Adaptive Operation The LEAP capabilities of the GeoNET system enable it to measure long term trends and modify local operation to adapt to changing environmental conditions. The Stunt Ranch installation occurred in mid-summer months when coastal fog is infrequent and the sun’s northern path provides maximum battery charging power. During the course of the experiment, the sun’s position was increasingly eclipsed by the mountain peaks as well as winter rain

storms causing greater than expected reductions in daily battery charge replenishment. As part of GeoNET's *sysmon* health monitoring service, battery and charging state are continuously logged. These logs provide a history of daily and seasonal changes. We provide an abbreviated history containing one week from July and another from November in Figure 5.9. The figure provides solar panel voltage and charging current as well as battery voltage and current draw.

The daily charging cycles are clearly evident in the increased panel voltage and charging currents. The July and November records, though similar, contain important distinctions. The November solar charging current pulses are noticeably diminished in size versus July. A plot of the net battery charge, Figure 5.10, clarifies this difference. In July the battery's net charge reaches an equilibrium, where nightly battery use is balanced by daily charging. The November record shows a disparity between nightly battery use and daily charging indicating a critical energy imbalance. A corrective change to platform was necessary to prevent battery depletion. Using LEAP profiling tools, we discovered an issue with the CPU's frequency governor, the software responsible for its DVFS settings, was not allowing the CPU to decrease its core frequency below a prescribed threshold. This was due to our use of the CPU's SPI port to collect samples directly from the RT155 AD data bus. The CPU throttling restriction caused unnecessary idle power dissipation not previously attributed to any application. The FPGA sampling architecture was modified to allow AD data to buffer within the FPGA while the CPU was able to govern itself to the minimum clock frequency. Once the FPGA sampled buffer filled, it wakes the CPU from the low power state to acquire the buffered data. This change to the FPGA resulting in a net energy savings as measured in Figure 5.11.

In addition to the energy reduction attributed directly to the sampling process

and shown in the Figure, an additional power savings in the CPU during idle time provided a platform reduction to reach energy equilibrium during the winter months of operation. This in-field adaptive behavior was made possible through a LEAP enabled platform and accompanying software tools.

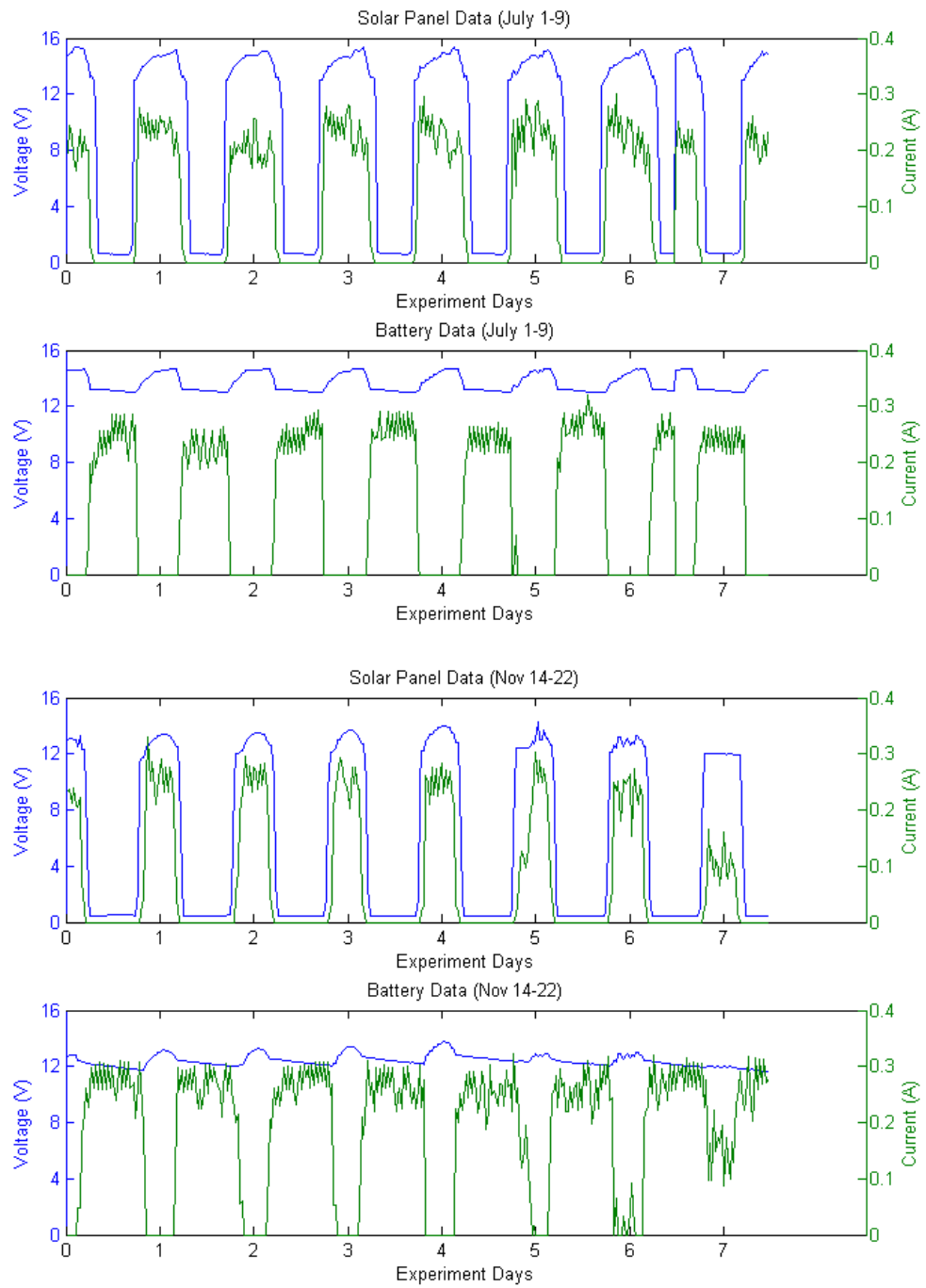


Figure 5.9: Solar charging and discharging over a one week time period

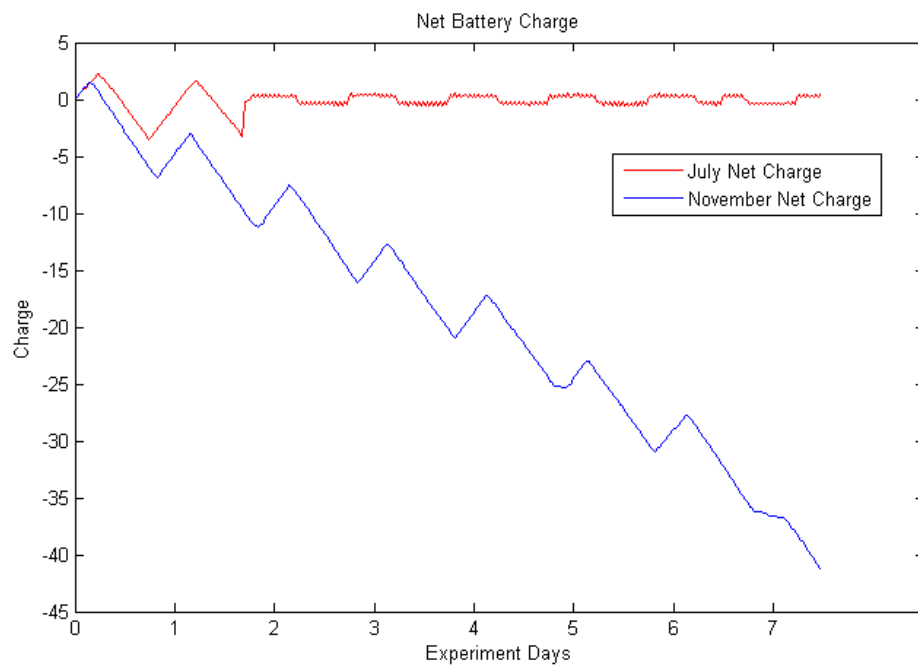


Figure 5.10: Battery state during July and November periods

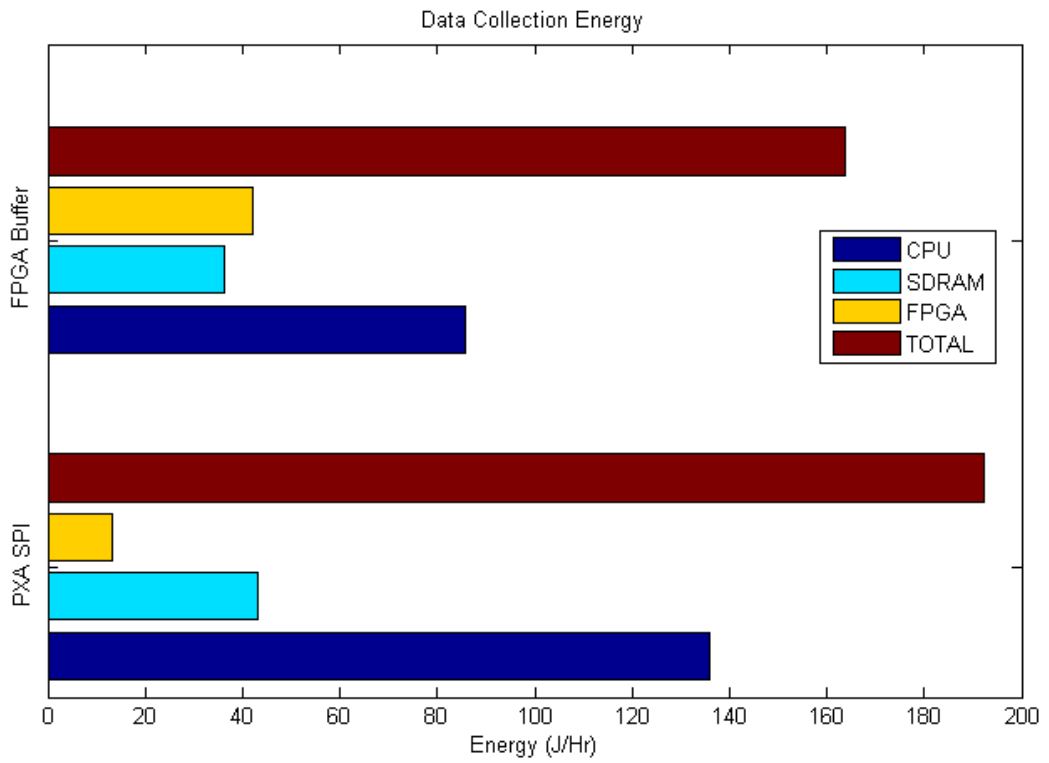


Figure 5.11: Energy consumed using CPU and FPGA data collection methods

CHAPTER 6

LEAP Applications and Partnerships

While this thesis has directed the LEAP architecture into a very specific network sensing application space, this was done solely to provide a concise demonstration of its utility rather than due to a fundamental limitation of its applicability. Rather, the LEAP principles have found an audience in a great number of other industries and markets. These markets range widely in customer (from consumer to industrial and medical) and size (from small medical wearables to large enterprise servers). This chapter discusses some applicable markets as well as highlights several of the research and product development programs that have been inspired by or developed directly out of LEAP research.

6.1 LEAP Applications and Markets

Smart Portables The latest smart phones and tablets now contain high performance multi-core applications processors operating at gigahertz speeds. They typically utilize a wide range of radio technologies and several highly optimized heterogeneous computation resources to operate cellular and local area wireless basebands, provide graphics rendering, and collect phone sensor data. Power management has become a critical factor in maintaining product battery life as the drive to become thinner and lighter while not compromising customer expectations for talk-time and media applications. As was noted in Section 2.3.2.2,

chipset manufactures have realized the need to monitor the dozens of different supplies needed by recent applications processors and enable software developers to perform energy benchmarking during development in order to optimize battery performance. This may lead to one day integrating LEAP capabilities into commercial phone hardware in order to adapt operation to the dynamic requirements of consumer application software.

Cloud Data Centers Recent attention to global climate changing factors, such as greenhouse gas emissions due to power generation, has generated new interest in reducing worldwide energy demand. A significant component of the increasing demand is due to electronic and computing equipment. The creation of large data centers to handle the explosive growth of internet services has begun taxing local power grids and has created new challenges for building temperature control and HVAC systems. In response to this energy demand, new government and industry sponsored programs have been created to address problems such as power supply conversion efficiency by recommending computing hardware standards [Mar06]. These small changes are expected to reduce current computing energy demands by as much as 20 percent. However, these new initiatives do not address some fundamental energy efficiency issues. Instead, new approaches are needed to address efficiency at a basic computing level. The LEAP architecture is well suited to this application and has broad application in the traditional server and enterprise equipment markets.

6.2 LEAP Inspired Products, Research, and Partnerships

MicroLEAP - Wireless Health In collaboration with Laurence Au's research in wireless health, LEAP has led to improvements in wearable medical devices.

Dr. Au architected the MicroLEAP (uLEAP) wireless sensor, which provides a new architectural solution for Body Sensor Network (BSN) applications in wireless health by introducing real-time system energy accounting. His MicroLEAP design provides the required sensing resolution and wireless interface for compatibility with mobile devices during data acquisition and processing. In addition, MicroLEAP provides on-board real-time system energy accounting and management via software in methods compatible with the LEAP architecture. Consequently, MicroLEAP can keep track its own energy consumption; signal acquisition and data transmission can be done on demand. This hardware-software system approach encourages the usage and scheduled operation of sensors and wireless interface, and it is a key consideration in MicroLEAP's design. According to Dr. Au, the distributed nature of BSNs also allows energy-efficient wearable sensors to collectively infer activity context through sensor fusion algorithms [WBB08].

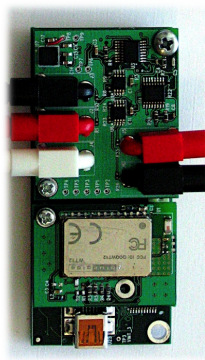


Figure 6.1: Photo of MicroLEAP platform

Runtime Direct Energy Measurement System (RTDEMS) In collaboration with Sebastian Ryffel from the Swiss Federal Institute of Technology we assisted in development of the RTDEMS enhancement to LEAP. Mr. Ryffel presents an energy attribution and accounting architecture for multi-core server systems, based upon LEAP, which can provide accurate, per-process energy in-

formation of individual hardware components. He introduces a hardware-assisted direct energy measurement system that integrates seamlessly with the host platform and provides detailed energy information of multiple hardware elements at millisecond scale time resolution. This is built upon a performance counter based behavioral model that provides indirect information on the proportional energy consumption of concurrently executing processes in the system. The direct and indirect measurement information is fused into a low-overhead kernel-based energy apportionment and accounting software system that provides visibility of per-process CPU and RAM energy consumption information on multi-core systems.

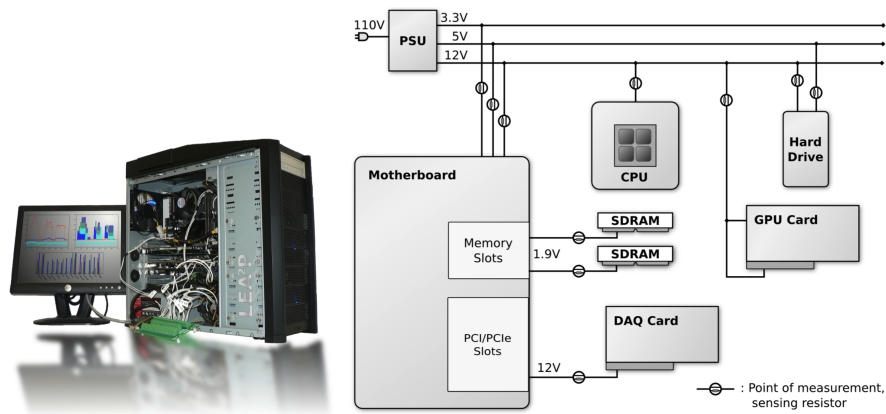


Figure 6.2: Photo of the LEAP server project and RTDEMS block diagram

National Instruments LabVIEW LEAP has also been part of programs involving industry affiliations. National Instruments is a leading supplier of graphical development tools and display software. Their LabVIEW software is used broadly in ENS applications and elsewhere. As part of a research program in collaboration with National Instruments, LabVIEW Microprocessor was ported to the LEAP2 platform. This was led by Timothy Chow, a UCLA graduate student. Mr. Chow constructed the first ever LabVIEW Microprocessor port for ARM Linux hosts, including methods for remote software deployment. In ad-

dition, the LabVIEW Microprocessor port to LEAP2 included enhancements to the standard LabVIEW software to enable energy monitoring and power scheduling controls. These new energy-aware virtual instrument (VI) capabilities enable non-traditional embedded systems developers to get real-time performance of individual VI components. This greatly reduces the complexity and technical barrier to LEAP enabled system development. Figure 6.3 highlights a couple of the LEAP VI components developed during this research program.

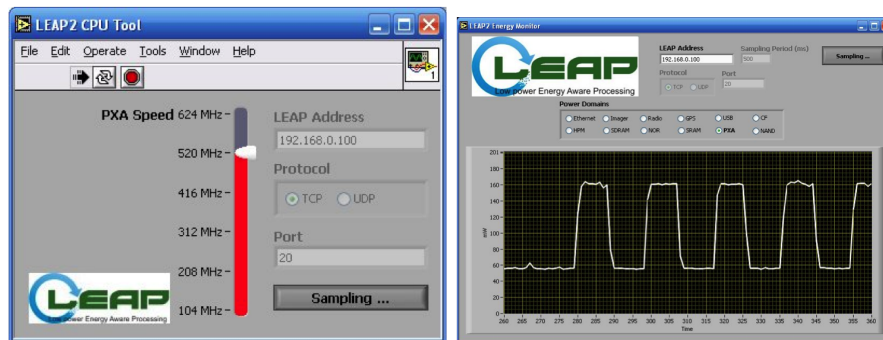


Figure 6.3: Photo of LEAP enabled LabVIEW components

Reftek RT155 As discussed in chapter 4, the GeoNET system is based upon a commercial product offering, the RT155, from Refraction Technologies Inc. The UCLA Geology department has recently purchased additional RT155 units including a revised RT619 design. The updated RT155 platform is a highly capable data acquisition system, capable of now providing 12 channels of 24-bit data with sample rates up to 4 KHz. The new RT619 includes numerous improvements including a new DM3730 with combined CPU and DSP, higher precision time synchronization circuit, and higher fidelity energy accounting capability. This enhancement is performed while maintaining average power dissipation less than 200mW. The platform is able to operate continuously in low power vigilant states while performing event detection. If events require, the platform can

quickly transition to a fully vigilant mode where networking and additional platform resources can be enabled through the custom EMAP logic. The platform is able to continuously record data through these power mode transitions. The LEAP-enabled RT619 CPU module is a crucial part of the RT155 design providing in-situ energy usage information allowing remote optimization of software operation and power management scheduling.



Figure 6.4: DARPA ADAPT module attached to expansion carrier board.

DARPA ADAPT The DARPA ADAPT program (BAA-12-11) focuses on rapid development cycles and high integration through the use of commercial technology. The ADAPT module hardware integrates the state of the art Qualcomm S4 (MSM8960) processor coupled to a low power Actel FPGA; a similar architecture to the LEAP2 platform. A photo of the ADAPT module from Phase I is shown in Figure 6.4. As part of the ADAPT platform development objectives, long battery lifetimes are required, much longer than traditional cell phone operation, necessitating deep power cycling and fine grained power management of the many processing, communications, and storage resources. As a program awardee,

we are integrating many LEAP capabilities, including EMAP operations into the FPGA, and energy-aware design concepts needed to meet this program goal. Future iterations of the ADAPT module may be enhanced new LEAP capabilities emphasizing the heterogeneous multi-core processing.

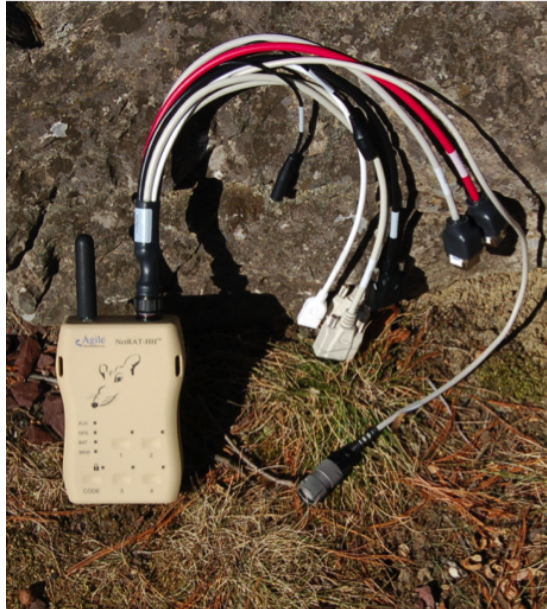


Figure 6.5: Photo of the Agile Communications NetRAT handheld computing device

Agile Communications NetRAT In addition to research programs, LEAP technology has been integrated into commercial and government related products. One example is the NetRAT handheld portable data analysis tool from Agile Communications, shown in Figure 6.5. The NetRAT-HH is intended for networked system test and evaluation applications and also provides users with a rugged, general purpose handheld device that can be easily extended to support a wide range of end user applications. Embedded into the core of NetRAT hardware is an FPGA design based upon LEAP2 including its EMAP functionality. The NetRAT includes numerous power sensing features across many of the

communications. Developers are able to use the LEAP energy profiling software applications previously described to monitor and manage platform energy and to assess energy impacts of various IP routing protocols, QoS policies, and network bandwidth settings.

CHAPTER 7

Conclusion

This thesis has described a new ENS platform design approach, and also applicable to a wide spectrum of other applications, in which the system is able to adaptively select the most energy efficient hardware components matching an application's needs. The LEAP2 and GeoNET platforms support highly dynamic requirements in sensing fidelity, computational load, storage media, and network bandwidth. Their design approaches focus on episodic operation of each component and considers the energy dissipation for each platform task. In addition to the platforms' unique hardware capabilities, their software architectures have been tailored to provide an easy to use power management interface, a robust, fault tolerant operating environment, and to enable remote upgrade of all software components.

The hardware has been chosen to exploit energy efficiency advantages of high performance resources while also providing low power capabilities for use in dynamic environments. As we have shown through direct experimental results, the ability to measure *in-situ* energy dissipation at *high resolution* provides a new insight into platform performance and enables new efficiency metrics. When coupled with a remotely managed software update capability, we enable a new energy-aware development cycle. We are then able to actively *tune* and *optimize* software execution and power management cycles to best match the environment.

While the LEAP architecture described in this thesis emphasizes the use of

multiple heterogenous hardware components in order to exploit high energy efficiency across a wide dynamic range of capability, future LEAP-enabled systems may include multi-modal hardware assets utilizing heterogeneous operational characteristics rather than multiple discrete hardware resources. Current computer architecture and networking projects [JSR04, BCE07] may one day allow run-time changes to low level hardware operation (e.g CPU microcode) in order to adaptively configure a resource's capability and power profile to match current system requirements. This may be coupled with on-chip energy measurement features, such as the EMAP ASIC, will enable a highly integrated energy-aware solution.

APPENDIX A

Hardware Design Files

The complete set of design files for the RefTek RT619 design are located in the LEAP subversion and git repositories.

Listing A.1: LEAP File Repositories

```
git://ascent.ee.ucla.edu/git/leap-linux.git  
svn://ascent.ee.ucla.edu/svn/leap2/
```

A.1 RefTek RT619

For reference the schematic for this module follows:

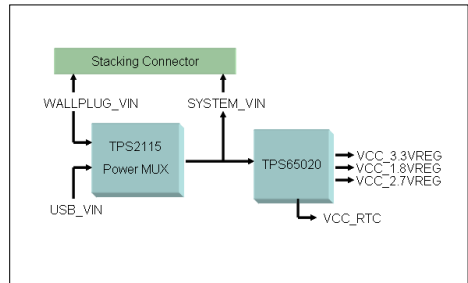
A.1.0.1 Schematics

LEAP - RT619

* Author: Dustin McIntire
 * Copyright (c) 2006 The Regents of the University of California.
 * All rights reserved.
 *
 * Permission to use, copy, modify, and distribute this software and its
 * documentation for any purpose, without fee, and without written agreement is
 * hereby granted, provided that the above copyright notice, the following
 * two paragraphs and the author appear in all copies of this software.
 *
 * IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR
 * DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT
 * OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF
 * CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES,
 * INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
 * AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS
 * ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO
 * PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS."

- Schematic Pages**
1. Title Sheet
 2. USB UART
 3. Current Sensing
 4. USB Host and OTG
 5. Ethernet
 6. SD Card
 7. Radio IF
 8. GPS
 9. Reftek Subsystem
 10. FPGA
 11. Stacking Connector
 12. Power Supplies

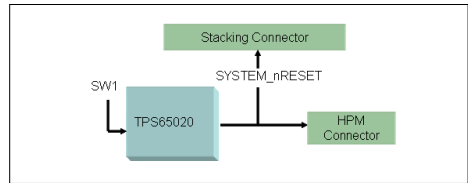
Power Supply Hierarchy



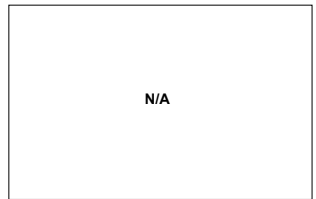
LAYOUT NOTES:

1. No vias inside SMT caps and resistors.
2. No vias inside SMT pads.

Reset Hierarchy



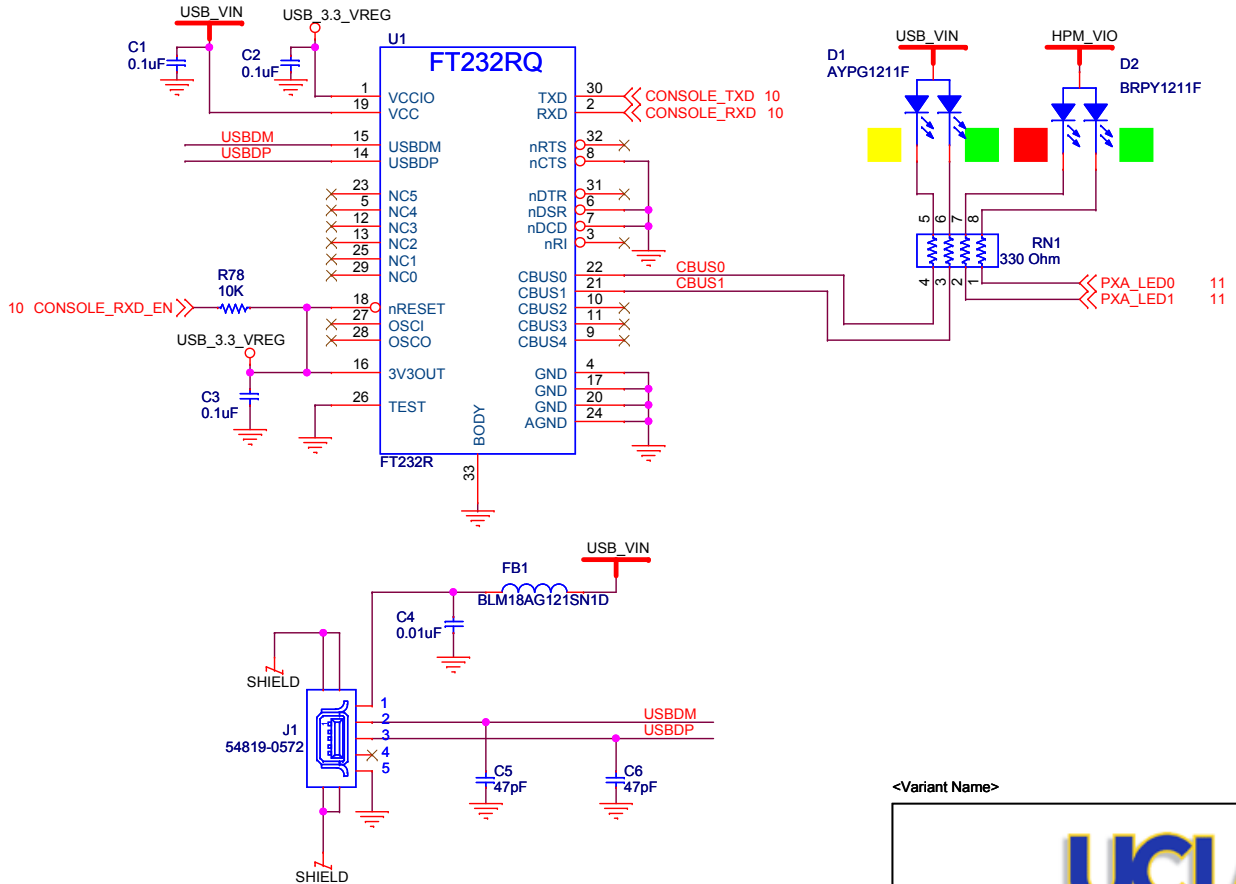
JTAG Chain



MTG149217
 MTG HOLE
 X
 MTG149217
 MTG HOLE
 X
 MTG149217
 MTG HOLE
 X
 MTG149217
 MTG HOLE
 X
 CHASSIS_GND

<Variant Name>
UCLA
 File: EMAP2 - Title Page
 Size: Document Number Rev: C
 Date: Wednesday, December 11, 2013 Sheet: 1 of 16

USB Console UART

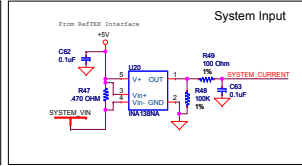
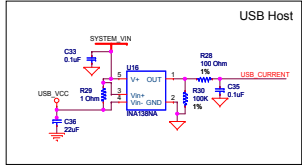
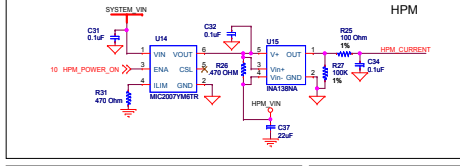
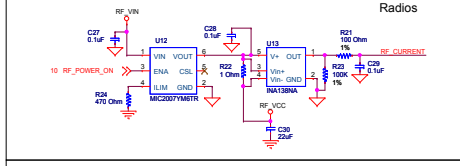
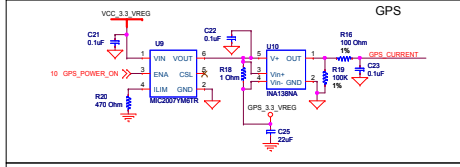
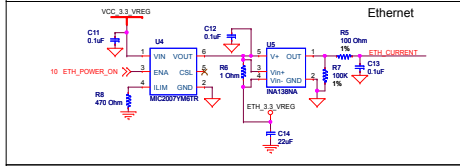
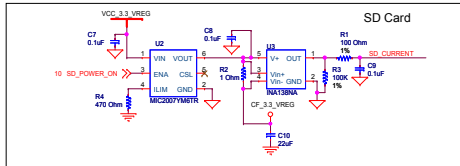


<Variant Name>

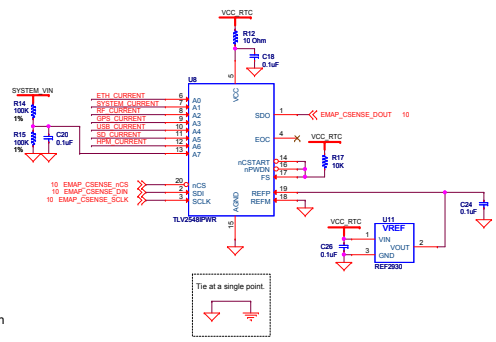


Title		EMAP2 - USB UART	
Size	Document Number	Rev	
A		C	
Date:	Friday, April 02, 2010	Sheet	2 of 16

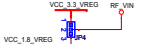
CURRENT SENSING



Current Sense ADC

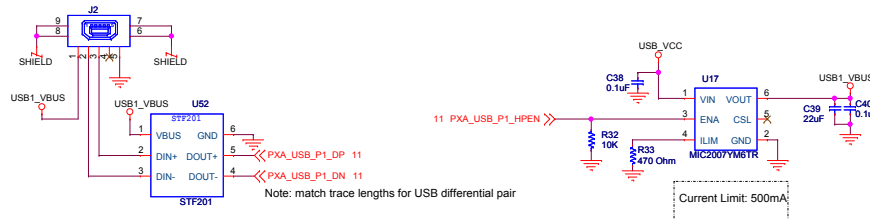


Radio Voltage Selection

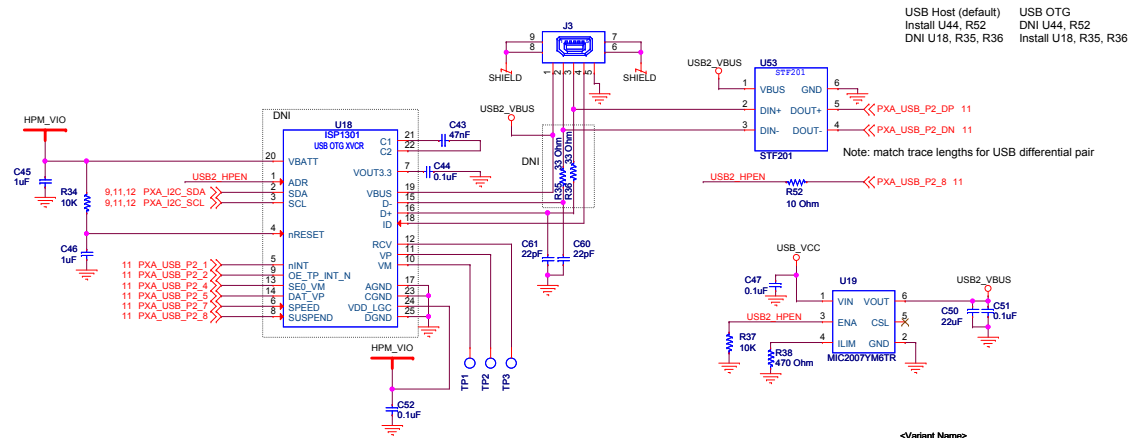


Tie at a single point

USB Host Port 1



USB Host Port 2 or USB OTG Using External Transceiver

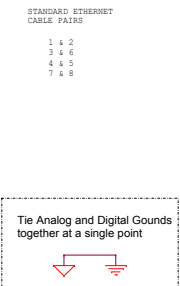
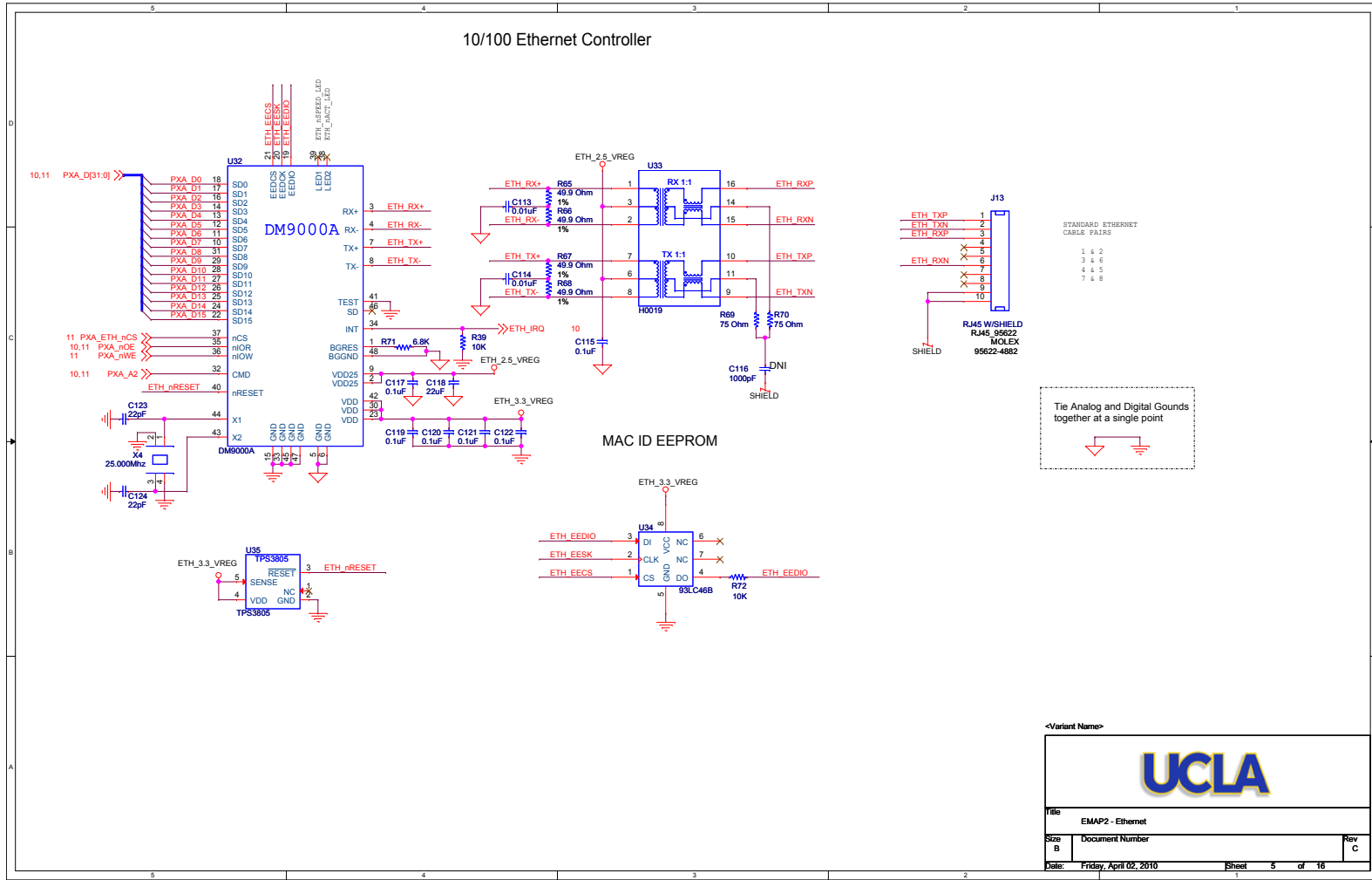


<Variant Name>



File		EMAP2 - USB OTG	
Size	B	Document Number	Rev
Date:	Friday, April 02, 2010	Sheet	4 of 16

10/100 Ethernet Controller

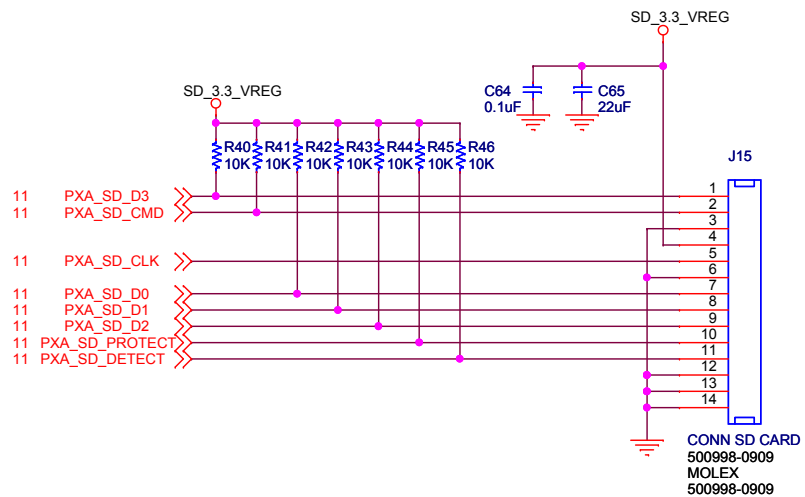


<Variant Name>

File: EMAP2 - Ethernet

Size: B	Document Number:	Rev: C
Date: Friday, April 02, 2010	Sheet: 5 of 16	

SD Card Interface



<Variant Name>

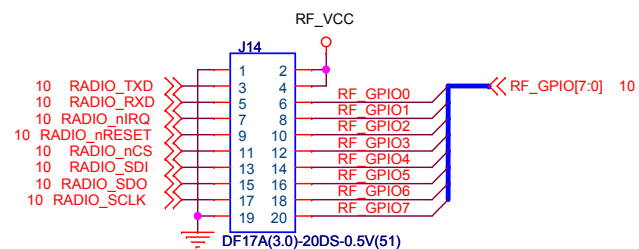


Title
EMAP2 - SD Card Interface Schematic

Size A	Document Number ASCENT-LEAP2-EMAP2	Rev C
-----------	---------------------------------------	----------

Date: Friday, April 02, 2010 Sheet 6 of 16

Radio Interface



MTG HOLE
MTG142/217

<Variant Name>

UCLA

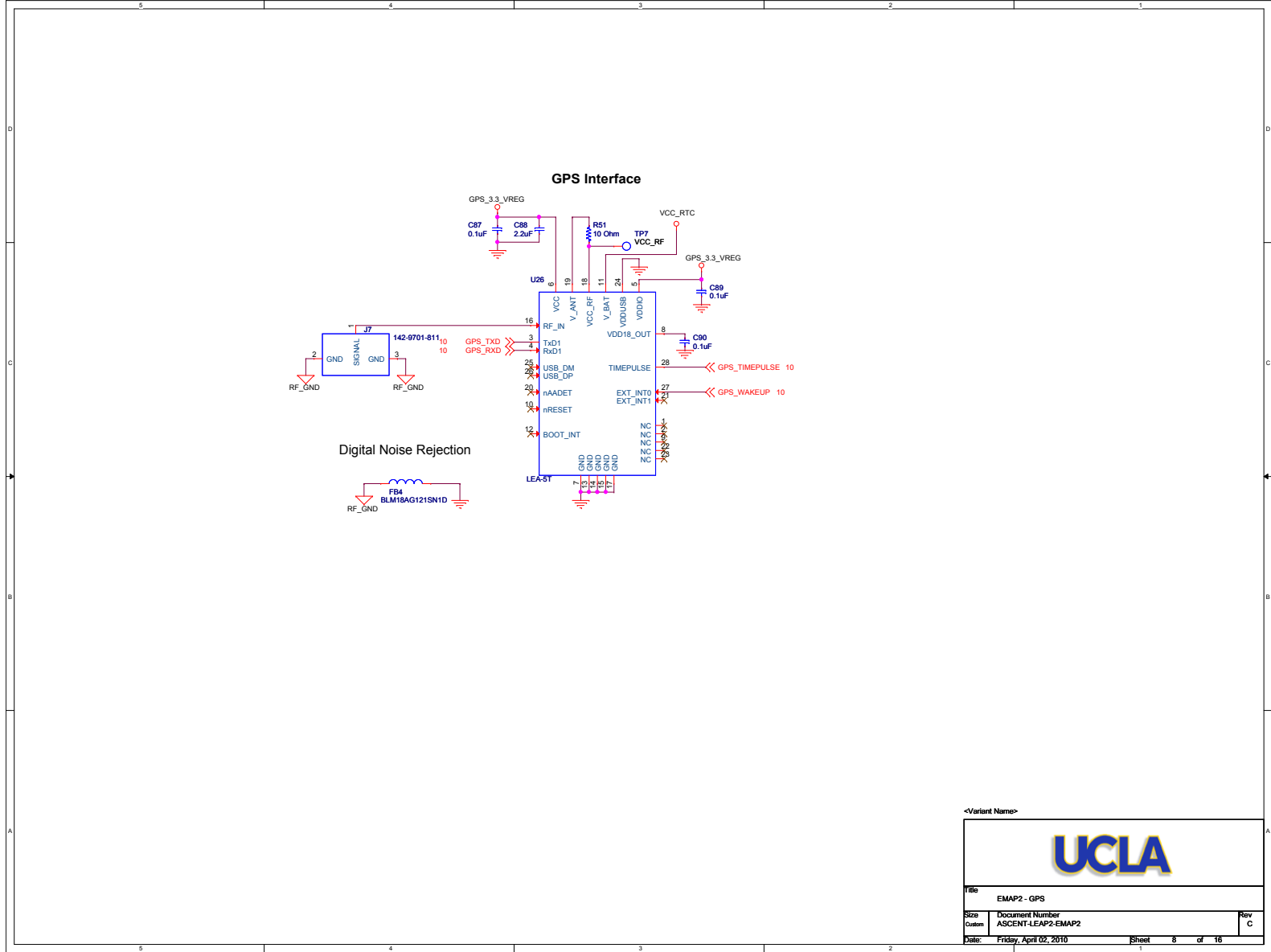
Title
EMAP2 - Radio Interface

Size
A Document Number

Rev
C

Date: Friday, April 02, 2010

Sheet 7 of 16

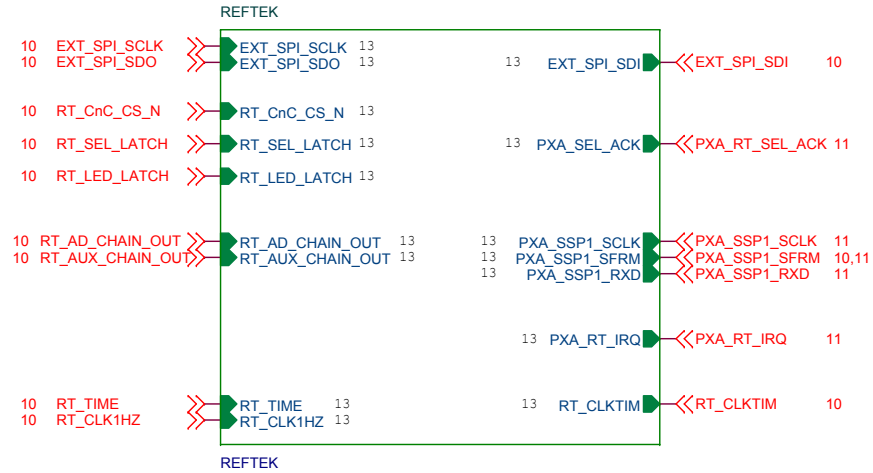


<Variant Name>



Title		EMAP2 - GPS	
Size	Document Number	Rev	
Custom	ASCENT-LEAP2-EMAP2	C	
Date:	Friday, April 02, 2010	Sheet	8 of 16

RefTEK Sub System



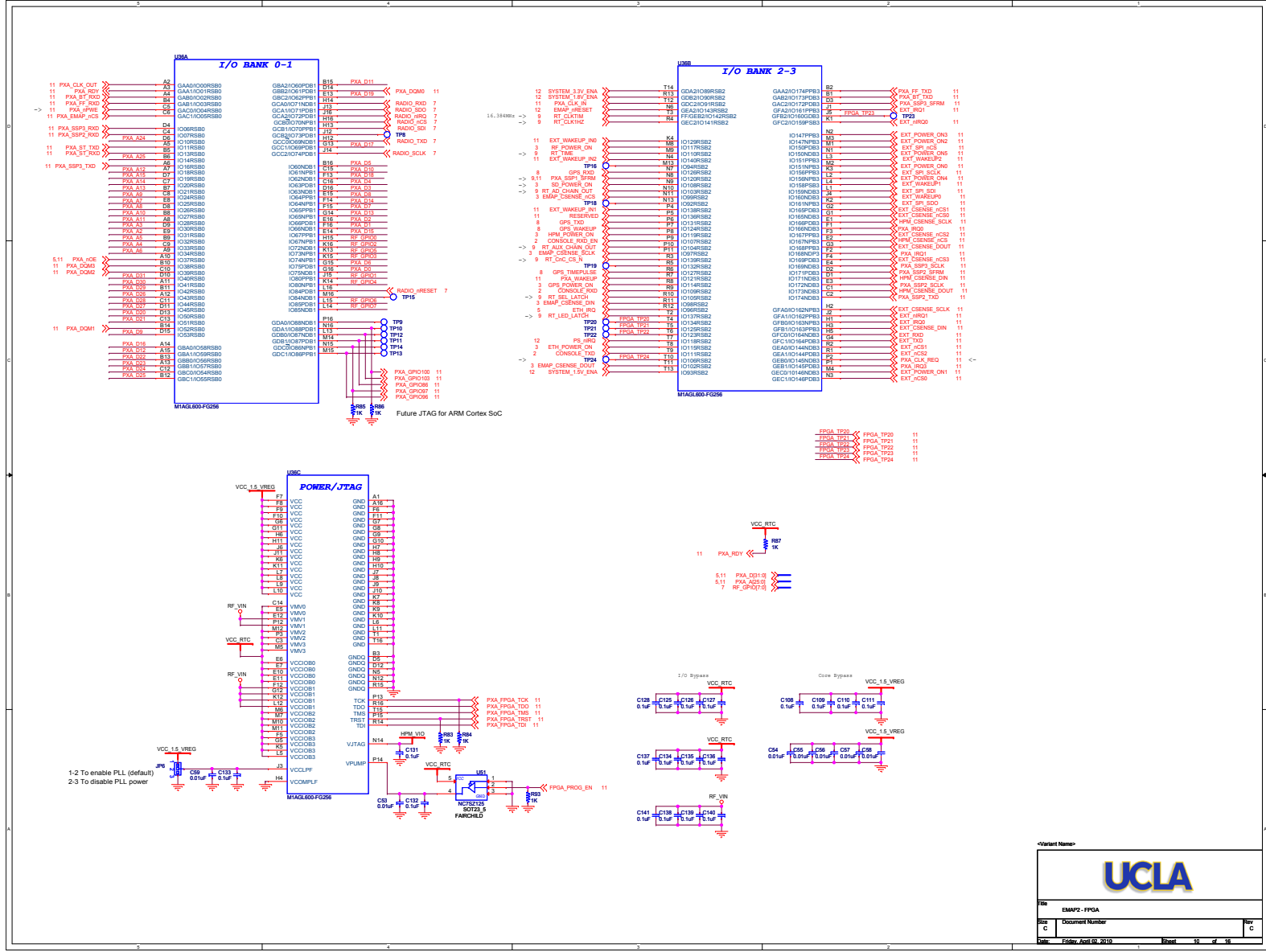
<Variant Name>



Title
EMAP2 - RefTek Sub System Interconnect

Size A	Document Number	Rev C
-----------	-----------------	----------

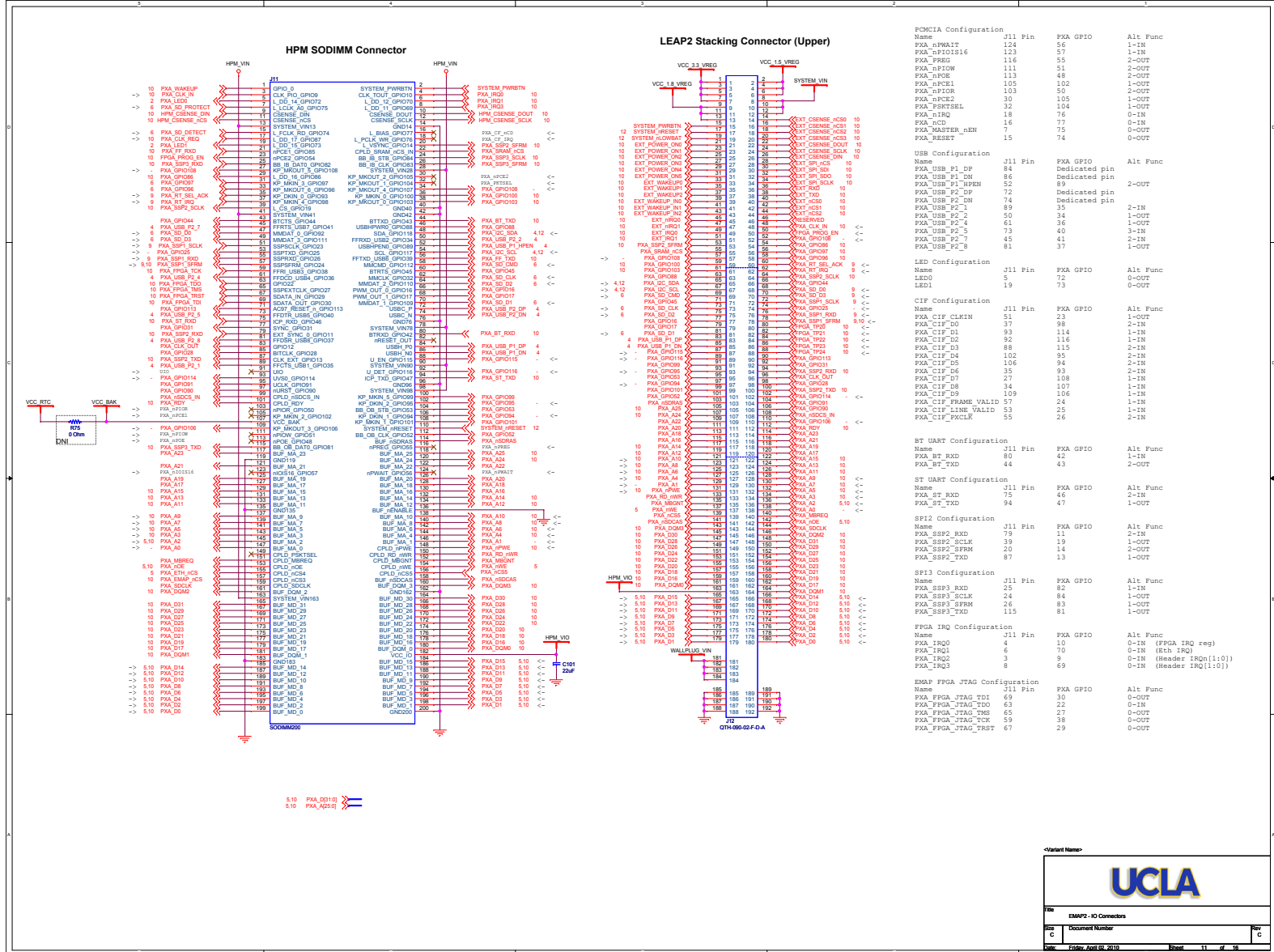
Date: Friday, April 02, 2010 Sheet 9 of 16



<Variant Name>

UCLA

File	EMAP2 - FPGA		
Size	Document Number	Rev	C
Date	Friday, April 02, 2010	Sheet	10 of 16



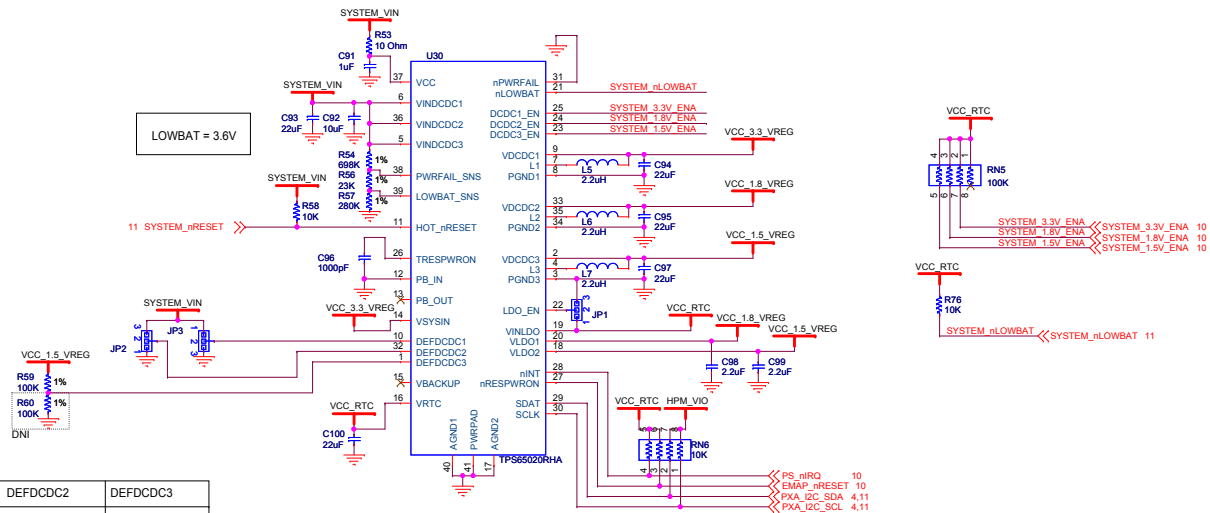
UCLA

File: EMAP2 - IO Connectors

Doc C Document Number

Date: Friday, April 02, 2010 8:04am 11 of 16

POWER SUPPLIES



LOWBAT = 3.6V

DEFDCDC1	DEFDCDC2	DEFDCDC3
JP3 1-2 VCCIO=3.3V 2-3 VCCIO=3.0V	JP2 1-2 VCCMEM=1.8V 2-3 VCCMEM=2.5V	Vout= 0.6*(R1+R2)/R2

<Variant Name>

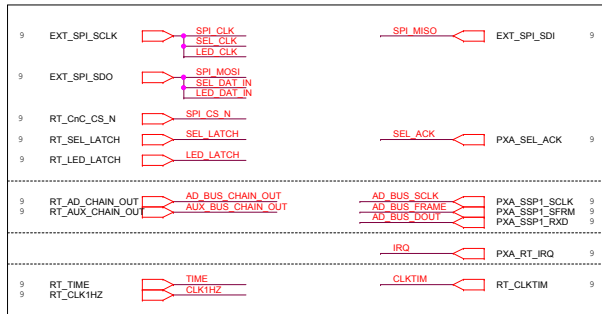
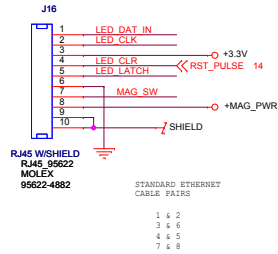


File		EMAP2 - Power Supply
Size	B	Document Number
Date:	Friday, April 02, 2010	Sheet 12 of 18

Rev C

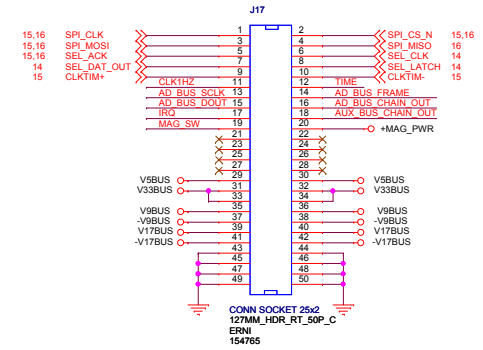
RefTek Module Connections

INDICATORS TO DAS LED

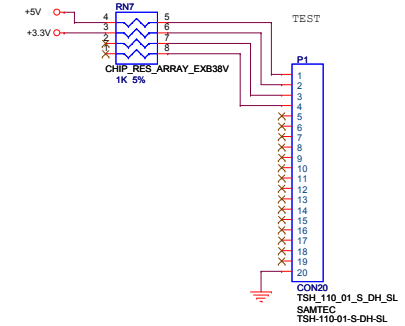
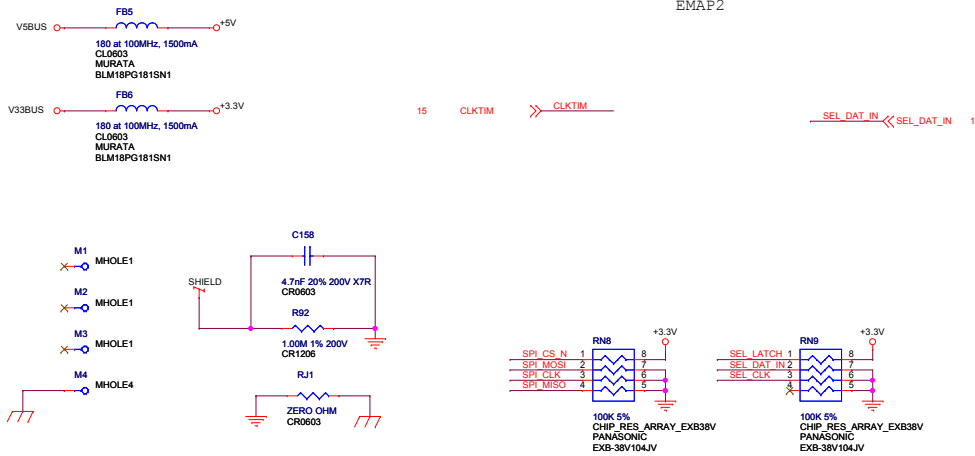


COMMAND & CONTROL AND POWER

RIBBON CABLE TO ADDITIONAL SHIELDED MODULES

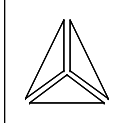
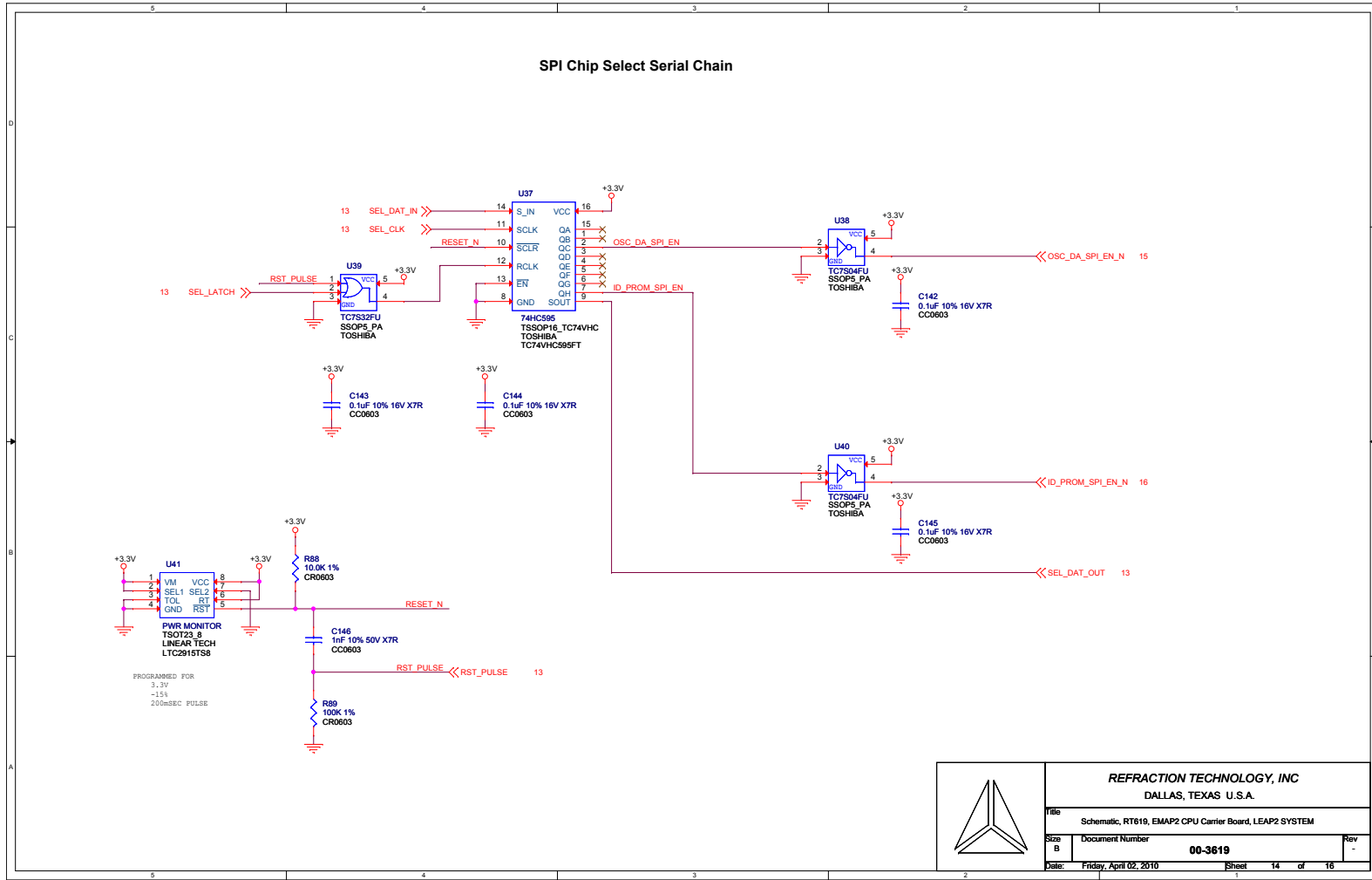


EMAP2



REFRACTION TECHNOLOGY, INC DALLAS, TEXAS U.S.A.			
File	Schematic, RT619, EMAP2 CPU Carrier Board, LEAP2 SYSTEM		
Size	Document Number	00-3619	Rev
B			-
Date:	Friday, April 02, 2010	Sheet	13 of 16

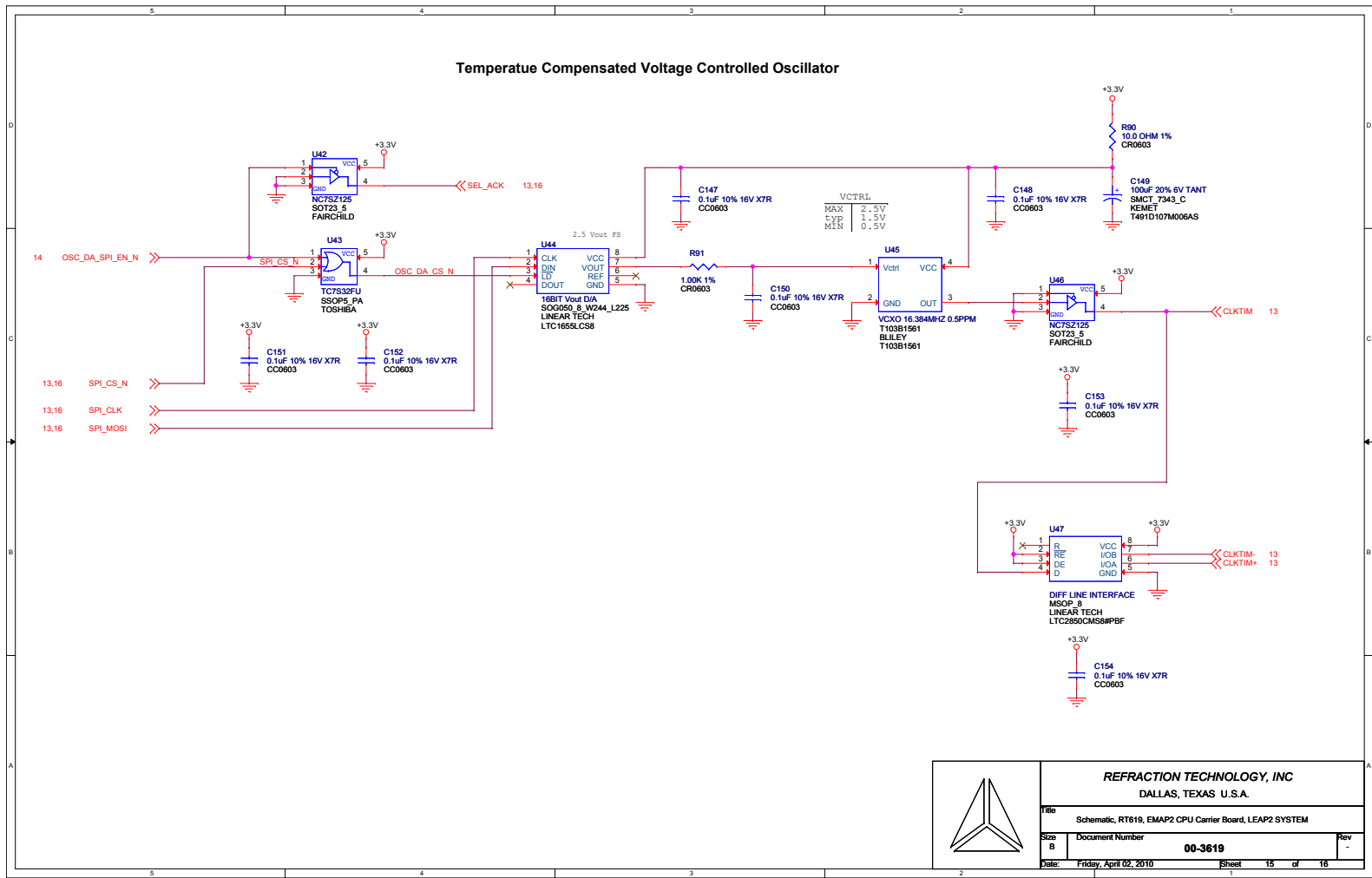
SPI Chip Select Serial Chain



REFRACTION TECHNOLOGY, INC
DALLAS, TEXAS U.S.A.

File		Schematic, RT619, EMAP2 CPU Carrier Board, LEAP2 SYSTEM	
Size	B	Document Number	00-3619
Date:	Friday, April 02, 2010	Sheet	14 of 16

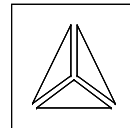
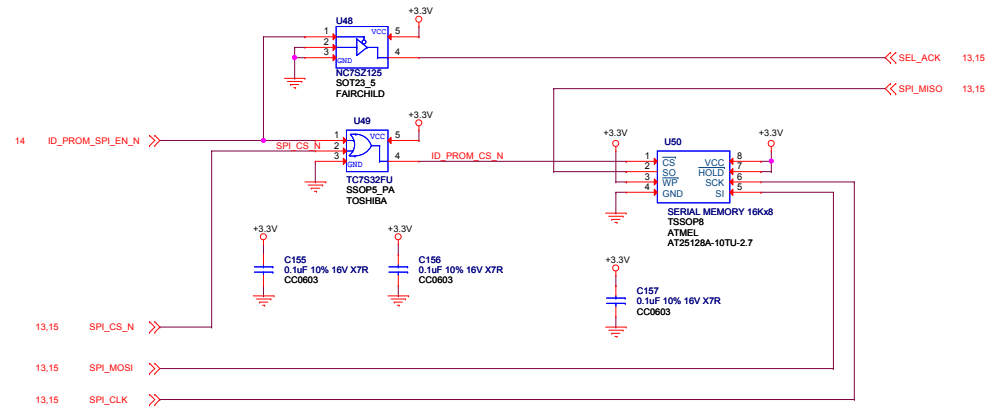
Temperature Compensated Voltage Controlled Oscillator



REFRACTION TECHNOLOGY, INC
DALLAS, TEXAS U.S.A.

File		Schematic, RT619, EMAP2 CPU Carrier Board, LEAP2 SYSTEM	
Size	Document Number	00-3619	
Date:	Friday, April 02, 2010	Sheet	15 of 16

Board Identification Serial EEPROM



REFRACTION TECHNOLOGY, INC
DALLAS, TEXAS U.S.A.

File		Schematic, RT619, EMAP2 CPU Carrier Board, LEAP2 SYSTEM	
Size	Document Number	Rev	-
B	00-3619		
Date:	Friday, April 02, 2010	Sheet	16 of 16

Part Number: RT619-0001

Revision: A1

Unit Qty	Reference	Part Name	Manufacturer	Description	Value	Part Type	Notes
1	BT1	ML621-TZ1	FDK America	BATT, 3V, LITH COIN RECHARGEABLE	3V	Battery	
3	FB1-3	BLM18AG121SN1D	Murata	FLTR, EMI, 120 OHM, 200mA, 0603	120 OHM	Bead	
2	FB5-6	BLM18PG181SN1	Murata	FERRITE CHIP 180OHM 1500MA 0603	180 Ohm	Bead	
1	FB4	BLM18AG121SN1D	Murata	FLTR, EMI, 120 OHM, 200mA, 0603	120 OHM	Bead	
8	C4 C53-59	ECJ-0EB1E103K	Panasonic	CAP, 0.01UF, X7R, 25V, 10%, 0402	0.01uF	Ceramic	
1	C205	GRM155R71C223KA01D	Murata	CAP, 0.022UF, X5R, 10V, 10%, 0402	0.022uF	Ceramic	
18	C44-47 C142-145 C147-148 C150-157 C1-3 C7-9 C11-13 C15-24 C26-29 C31-35 C40-41 C48 C51-52 C60-64 C66-70 C73-74 C76 C78-103	C0603C104K4RAC	Kemet	CAP CER 0.1UF 16V 10% X7R 0603	0.1uF	Ceramic	
72		ECJ-0EB1A104K	Panasonic	CAP CER 0.1UF 10V 10% X5R 0402	0.1uF	Ceramic	
1	C146	C0603C102K5RAC	Kemet	CAP CER 1000PF 50V 10% X7R 0603	1000pF	Ceramic	
11	C10 C14 C25 C30 C36-39 C43 C50 C65	ECJ-2FB0J226M	Panasonic	CAP, 22UF, X5R, 6.3V, 20%, 0805	22uF	Ceramic	
2	C72 C207	GRM1555C1H300JZ01D	Murata	CAP, 30PF, COG, 50V, 10%, 0402	30pF	Ceramic	
1	C158	C0603C472M2RAC	Kemet	CAP CER 4700PF 200V 20% X7R 0603	4.7nF 20% 200V X7R	Ceramic	
5	C42 C49 C71 C75 C77	C0603C475K8PACTU	Kemet	CAP, 4.7UF, Y5V, 10V, 10%, 0603	4.7uF	Ceramic	
2	C5-6	ECJ-0EC1H470J	Panasonic	CAP, 47PF, NPO, 50V, 5%, 0402	47pF	Ceramic	
2	R19 R26	ERJ-3RQFR47V	Panasonic	RES, 0.470 OHM, 1/10W, TKF, 1%, 0603	0.47	Chip Resistor	
6	R2 R6-7 R13-14 R18	RC0603FR-071RL	Yageo	RES, 1.00 OHM, 1/10W, TKF, 1%, 0603	1	Chip Resistor	
1	R148	ERJ-2RKF10R0X	Panasonic	RES, 10 OHM, 1/16W, TKF, 1%, 0402	10	Chip Resistor	
1	R90	CRCW060310R0F	Vishay-Dale	RES 10.0 OHM 1/10W 1% 0603 SMD	10	Chip Resistor	
2	R12 R40	ERJ-2GEJ100X	Panasonic	RES, 10 OHMS, 1/16W, TKF, 5%, 0402	10	Chip Resistor	
4	R20 R22-23 R71	ERJ-2RKF49R9X	Panasonic	RES, 49.9, 1/16W, TKF, 1%, 0402	49.9	Chip Resistor	
8	R1 R5 R16 R21 R24 R27 R30 R54	ERJ-2RKF1000X	Panasonic	RES, 100 OHM, 1/16W, TKF, 1%, 0402	100	Chip Resistor	
4	R48-51	ERJ-2GEJ331X	Panasonic	RES 330 OHM 1/10W 5% 0402 SMD	330	Chip Resistor	
1	JP6	ERJ-2GE0R00X	Panasonic	RES, 0, 1/16W, TKF, 5%, 0402	0 Ohm	Chip Resistor	Position: 2-3
1	R91	CRCW06031001F	Vishay-Dale	RES 1.00K OHM 1/10W 1% 0603 SMD	1.00K	Chip Resistor	
1	R92	CRCW12061004F	Vishay-Dale	RES 1.00M OHM 1/4W 1% 1206 SMD	1.00M	Chip Resistor	
1	R88	CRCW06031002F	Vishay-Dale	RES 10.0K OHM 1/10W 1% 0603 SMD	10.0K	Chip Resistor	
1	R89	CRCW06031003F	Vishay-Dale	RES 100K OHM 1/10W 1% 0603 SMD	100K	Chip Resistor	
4	R3 R4 R10 R32	ERJ-2RKF1003X	Panasonic	RES, 100K, 1/16W, TKF, 1%, 0402	100K	Chip Resistor	
8	R8 R11 R36-39 R43 R47	ERJ-2GEJ103X	Panasonic	RES, 10K, 1/16W, TKF, 5%, 0402	10K	Chip Resistor	
10	R17 R25 R28-29 R31 R33-35 R72 R78	ERJ-2GEJ103X	Panasonic	RES, 10K, 1/16W, TKF, 5%, 0402	10K	Chip Resistor	
1	R156	ERJ-2RKF1242X	Panasonic	RES, 12.4K, 1/16W, TKF, 1%, 0402	12.4K	Chip Resistor	
1	R9	ERJ-2GEJ123X	Panasonic	RES, 12K, 1/16W, TKF, 5%, 0402	12K	Chip Resistor	
9	R15 R41-42 R44-46 R83 R85-86	ERJ-2GEJ102X	Panasonic	RES, 1K, 1/16W, TKF, 5%, 0402	1K	Chip Resistor	
1	R155	ERJ-2GEJ105X	Panasonic	RES, 1M, 1/16W, TKF, 5%, 0402	1M	Chip Resistor	
4	R48-51	ERJ-2GEJ331X	Panasonic	RES 330 OHM 1/10W 5% 0402 SMD	330	Chip Resistor	
2	R52-53	ERJ-2GEJ472X	Panasonic	RES 4.7K OHM 1/10W 5% 0402 SMD	4.7K	Chip Resistor	
1	RJ1	CRCW06030000Z0EA	Vishay-Dale	RES 0.0 OHM 1/10W 0603 SMD	ZERO OHM	Chip Resistor	
2	J4 J6	DF40C-100DS-0.4V(51)	Hirose	CONN RCPT 100POS 0.4MM SMD GOLD	100 pins	Connector	
1	J7	142-9701-811	Johnson Components	CONN, JACK, PC END MOUNT, RCPTL	142-9701-811	Connector	
1	J1	54819-0572	Waldom	CONN, MINI-USB, MINI-B TYPE, SMT	54819-0572	Connector	
2	J2-3	56579-0576	Waldom	CONN, MINI-USB, MINI-AB TYPE, SMT	56579-0576	Connector	
1	P1	TSH-110-01-S-DH-SL	Sametec	CONN UNSHROUDED HDR 20POS .05"	CON20	Connector	
1	J15	500998-0909	Molex	CONN MEMORY SECURE DIGIT REV SMD	CONN SD CARD	Connector	
1	J17	154765	ERNI	CONN SOCKET 50POS	CONN SOCKET 25x2	Connector	
2	J13 J16	95622-3981	Molex	CONN RJ45 RA TH JACK	RJ45 W/SHIELD	Connector	
1	Y1	ABM3B-25.000MHZ-B2-T	Abracon Corp	CRYSTAL 25.000MHZ 18PF SMD	25.000Mhz	Crystal	
1	U20	RCLAMP3304N.TCT	Semtech	IC, TVS ARRAY 4-LINE 3.3V, 10-SLP	RCLAMP3304N	Diode Array	
1	D16	1N4148WT-7	Fairchild	DIODE SCHOTTKY 2A 80V SOD532	1N4148WT-7	Diode	
1	J5	HRP10H-ND	Assman	HEADER, .100, 2X5, VERT	HDR2X5	Header	
1	U37	TC74VHC595FT	Toshiba	IC REGISTER SHIFT 8-BIT 16-TSSOP	74HC595	IC	

1	U24	TC74LCX74FT	Toshiba	IC FLIP-FLOP DUAL D-TYPE 14TSSOP	74LCX74	IC	
1	U34	93LC46B	Microchip	IC, EEPROM 93LC46B, TSSOP8	93LC46B	IC	
6	U2 U4 U6-7 U9 U25	AP2280-1WG-7	Diodes Inc	IC, LOAD SW CTRL 1CH, SOT-25	AP2280	IC	
1	U18	FIN1001M5X	Fairchild Semiconductor	IC DRIVER 3.3V LVDS HS SOT-23	FIN1001	IC	
1	U1	FT232RQ	FTDI	IC, FT232R, USB SERIAL, QFN32	FT232R	IC	
8	U3 U5 U10 U12-15 U30	INA216A2YFFR	Texas Instruments	IC CURRENT SHUNT MONITOR 4DSBGA	INA216A	IC	
1	U16	LAN9512	SMSC	LAN9512-JZX	LAN9512	IC	
1	U26	LEA-5T	UBLOX	MODULE, GPS, LEA-5T, SMT	LEA-5T	IC	
1	U28	LM73CIMK	National Semiconductor	IC, TEMP SENSR DGTL 2.7V, SOT23-6	LM73CIMK	IC	
1	U44	LTC1655LCS8	Linear Technology	IC DAC 16BIT R-R MICROPWR 8SOIC	LTC1655	IC	
1	U41	LTC2915TS8	Linear Technology	IC VOLT SUPERVISOR TSOT23-8	LTC2915	IC	
1	U36	AGL600V2-FGG256	Actel	IC, IGLOO FPGA 600K GATE, FG256	AGL600V2-FGG256	IC	
1	U17	MCP1603T	Microchip	IC SYNC BUCK REG 500MA TSOT-23-5	MCP1603T	IC	
1	U33	MIC2026-1BM	Micrel	IC SW DISTRIBUTION 2CHAN 8SOIC	MIC2026-1	IC	
4	U42 U46 U48 U51	NC7SZ125M5	Fairchild Semiconductor	IC BUFF TRI-ST UHS N-INV SOT235	NC7SZ125	IC	
1	U29	PCA9306DCUR	Texas Instruments	IC VOLT LEVEL TRANSLATOR 8-VSSOP	PCA9306	IC	
1	U11	REF2930AIDBZT	Texas Instruments	IC, REF2930, 3.0V REFERENCE, SOT23,	REF2930	IC	
1	U50	AT25128A-10TU-2.7	Atmel	IC EEPROM 128KBIT 20MHZ 8TSSOP	SERIAL MEMORY 16Kx8	IC	
4	U22-23 U38 U40	TC7S04FU	Toshiba	IC INVERTER HS C2MOS SSOP5	TC7S04FU	IC	
1	U21	TC7S08FU	Toshiba	IC GATE AND 2INP 74HC08 5-SSOP	TC7S08FU	IC	
1	U19	TC7S14FU	Toshiba	IC INVERTER SCHMITT 5-SSOP	TC7S14FU	IC	
2	U43 U49	TC7S32FU	Toshiba	IC GATE OR 2INPUT HS C2MOS SSOP5	TC7S32FU	IC	
1	U8	TLV2548IPWR	Texas Instruments	IC, 12 BIT ADC, TSSOP20	TLV2548IPWR	IC	
1	L1	CBC2518T4R7M	Taiyo-Yuden	INDUCTOR 4.7UH 20% 1007 SMD	4.7uH	Inductor	
1	D1	AYPG1211F	Stanley Electric	LED, YELLOW/GREEN BICOLOR, 2.1V, RT ANG, SMD	AYPG1211F	LED	
1	D2	BRPY1211F	Stanley Electric	LED, RED/YELLOW-GREEN BICOLOR, 2.1V, RT ANG, SMD	BRPY1211F	LED	
1	U27	450-0037	LS Research	MODULE TIWI-R2 U.FL	450-0037	Module	
2	RN8-9	EXB-38V104JV	Panasonic	RES ARRAY 100K OHM 4 RES 1206	100K	Resistor Array	
1	RN7	EXB-38V102JV	Panasonic	RES ARRAY 1K OHM 4 RES 1206	1K	Resistor Array	
1	C149	T491D107M006AS	Kemet	CAP TANT 100UF 6.3V 7343	100uF	Tantalum	
1	U45	T103B1561	Bliley	TVCXO 16.384MHZ 0.5PPM SMD	T103B1561	TCXO	
1	T1	H0019	Pulse	TRANSFORMER, 10/100 MAGNETICS, 0.500x0.350	H0019	Transformer	
3	Q1-2 Q10	BC847A-TP	Micro Semi	BJT, NPN, 100MA, 45V, SOT23	BC847A-TP	Transistor	
1	FB7	BLM18AG121SN1D	Murata	FLTR, EMI, 120 OHM, 200mA, 0603	120 OHM	Bead	NOT INSTALLED

A.1.0.2 Layout

APPENDIX B

LEAP2 Design Documents

B.1 Hardware Schematics and PCB Layout

This section provides all hardware design files.

B.1.1 EMAP2

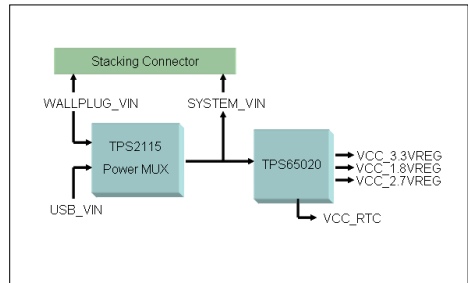
B.1.1.1 Schematics

LEAP2 - EMAP2

* Author: Dustin McIntire
 * Copyright (c) 2006 The Regents of the University of California.
 * All rights reserved.
 *
 * Permission to use, copy, modify, and distribute this software and its
 * documentation for any purpose, without fee, and without written agreement is
 * hereby granted, provided that the above copyright notice, the following
 * two paragraphs and the author appear in all copies of this software.
 *
 * IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR
 * DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT
 * OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF
 * CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 *
 * THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES,
 * INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
 * AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS
 * ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO
 * PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS."

- Schematic Pages**
1. Title Sheet
 2. USB UART
 3. Current Sensing
 4. USB Host and OTG
 5. Ethernet
 6. Compact Flash
 7. Radio IF
 8. GPS
 9. CMOS Imager IF
 10. FPGA
 11. Stacking Connector
 12. Power Supplies

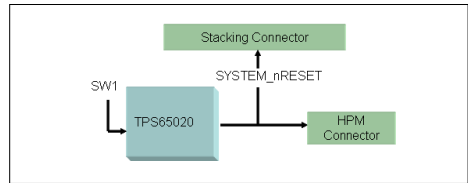
Power Supply Hierarchy



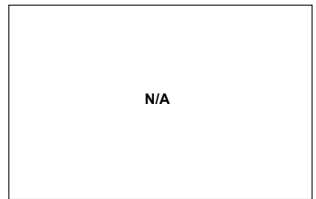
LAYOUT NOTES:

1. No vias inside SMT caps and resistors.
2. No vias inside SMT pads.

Reset Hierarchy



JTAG Chain

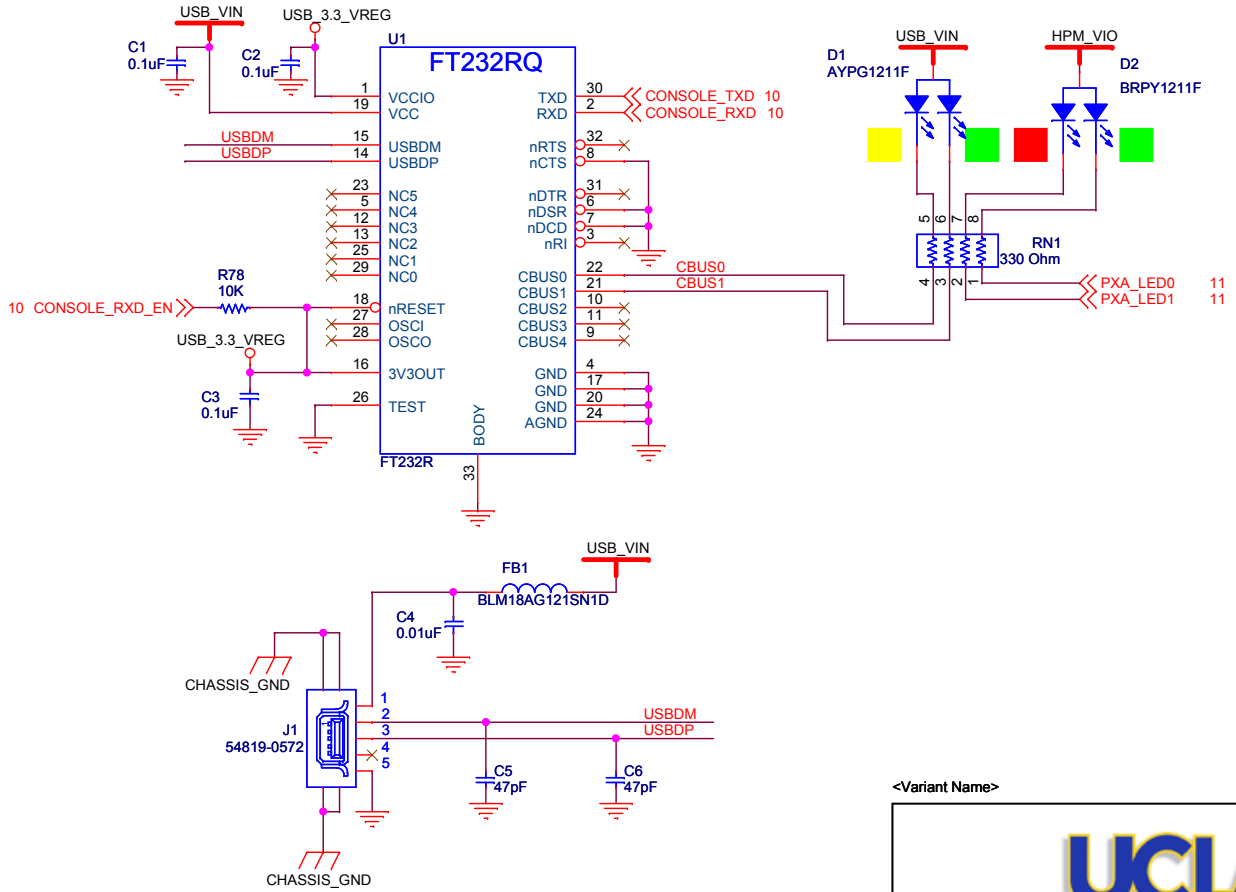


MTG49217
 MTG HOLE
 CHASSIS_GND
 MTG49217
 MTG HOLE
 CHASSIS_GND
 MTG49217
 MTG HOLE
 CHASSIS_GND
 MTG49217
 MTG HOLE
 CHASSIS_GND

UCLA

File: EMAP2 - Title Page		
Size: C	Document Number:	Rev: C
Date: September, November 01, 2012 Sheet: 1 of 12		

USB Console UART

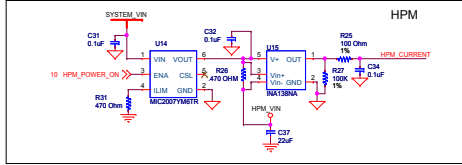
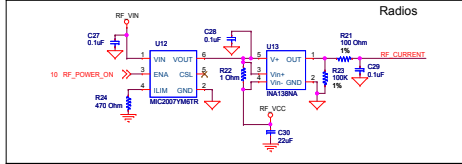
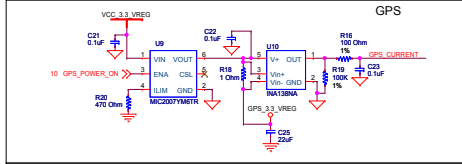
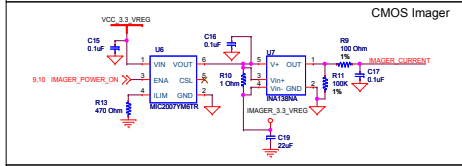
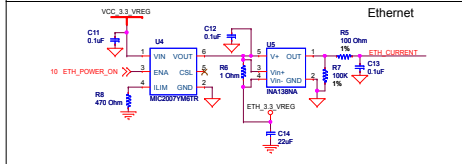
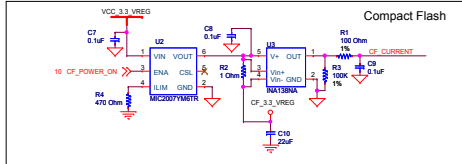


<Variant Name>

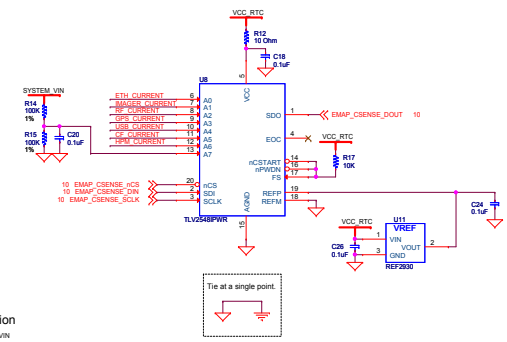


Title		EMAP2 - USB UART	
Size	Document Number	Rev	
A		C	
Date:	Saturday, November 03, 2012	Sheet	2 of 12

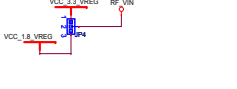
CURRENT SENSING



Current Sense ADC



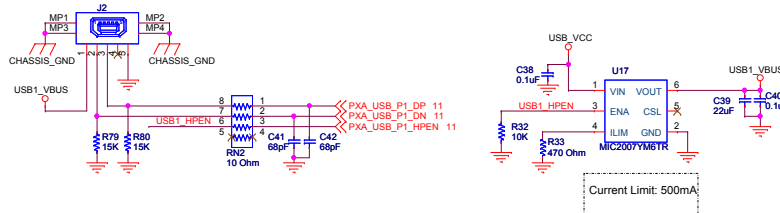
Radio Voltage Selection



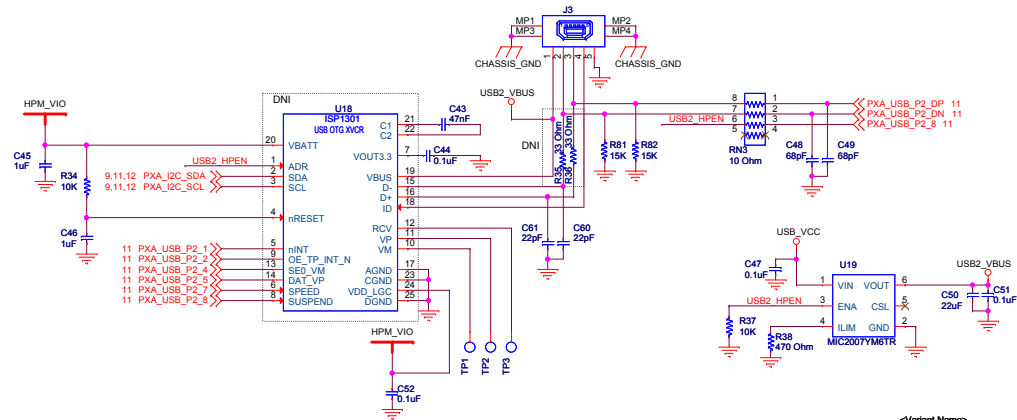
<Variant Name>

File: EMAP2 - Current Sensing
 Size: C Document Number
 Date: September, November 01, 2012 Sheet: 3 of 12

USB Host Port 1



USB Host Port 2 or USB OTG Using External Transceiver

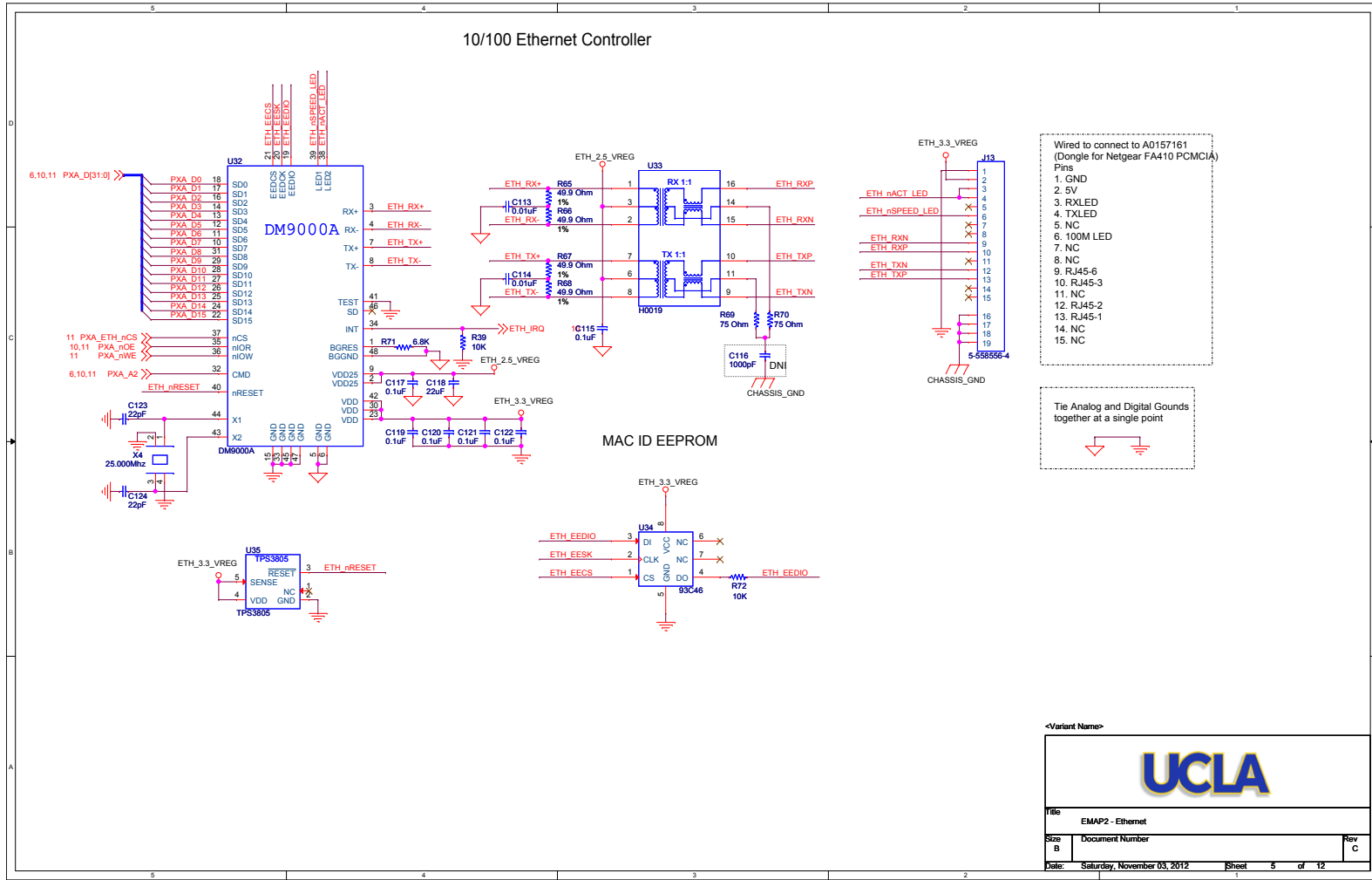


<Variant Name>



File		EMAP2 - USB OTG	
Size	B	Document Number	Rev C
Date:	Saturday, November 03, 2012	Sheet	4 of 12

10/100 Ethernet Controller

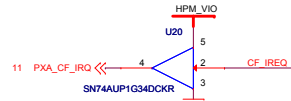
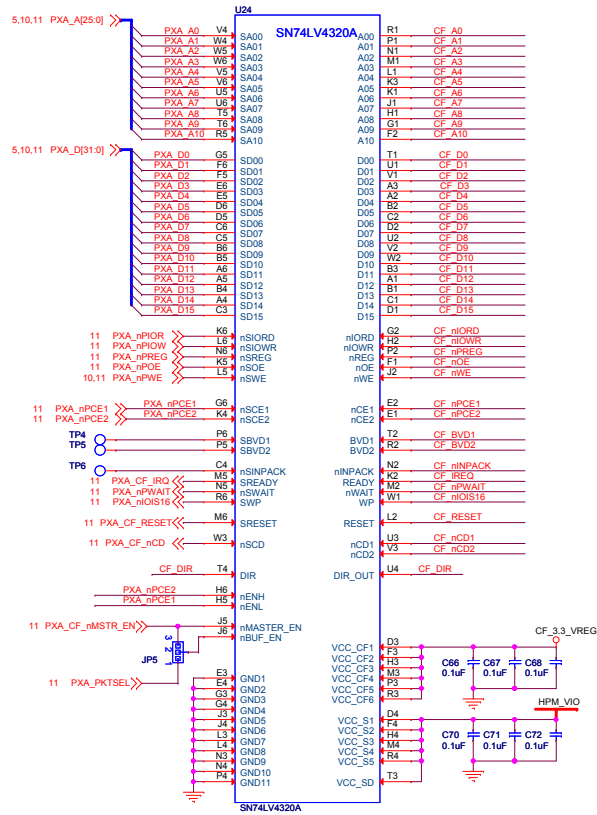


<Variant Name>

File: EMAP2 - Ethernet

Size: B	Document Number	Rev: C
Date: Saturday, November 03, 2012	Sheet: 5 of 12	

Compact Flash Interface Buffers



Compact Flash Connector

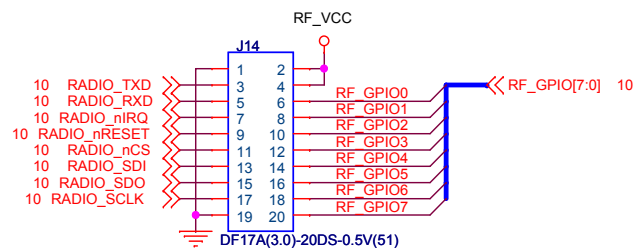


<Variant Name>



File		
EMAP2 - Compact Flash Interface Schematic		
Size	Document Number	Rev
B	ASCENT-LEAP2-EMAP2	C
Date:	Saturday, November 03, 2012	Sheet 8 of 12

Radio Interface

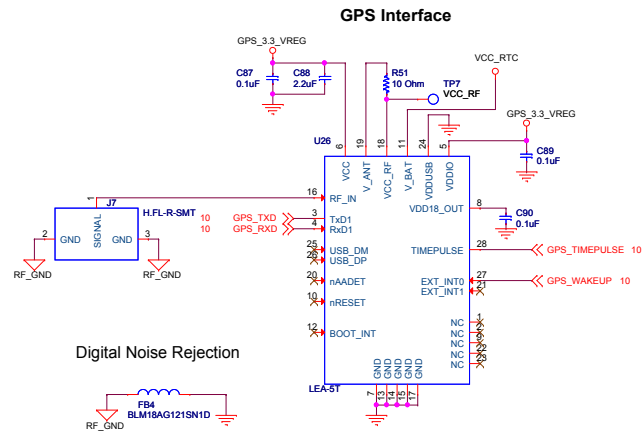


MTG HOLE
MTG142/217

<Variant Name>



Title EMAP2 - Radio Interface		
Size A	Document Number	Rev C
Date: Saturday, November 03, 2012	Sheet 7 of 12	

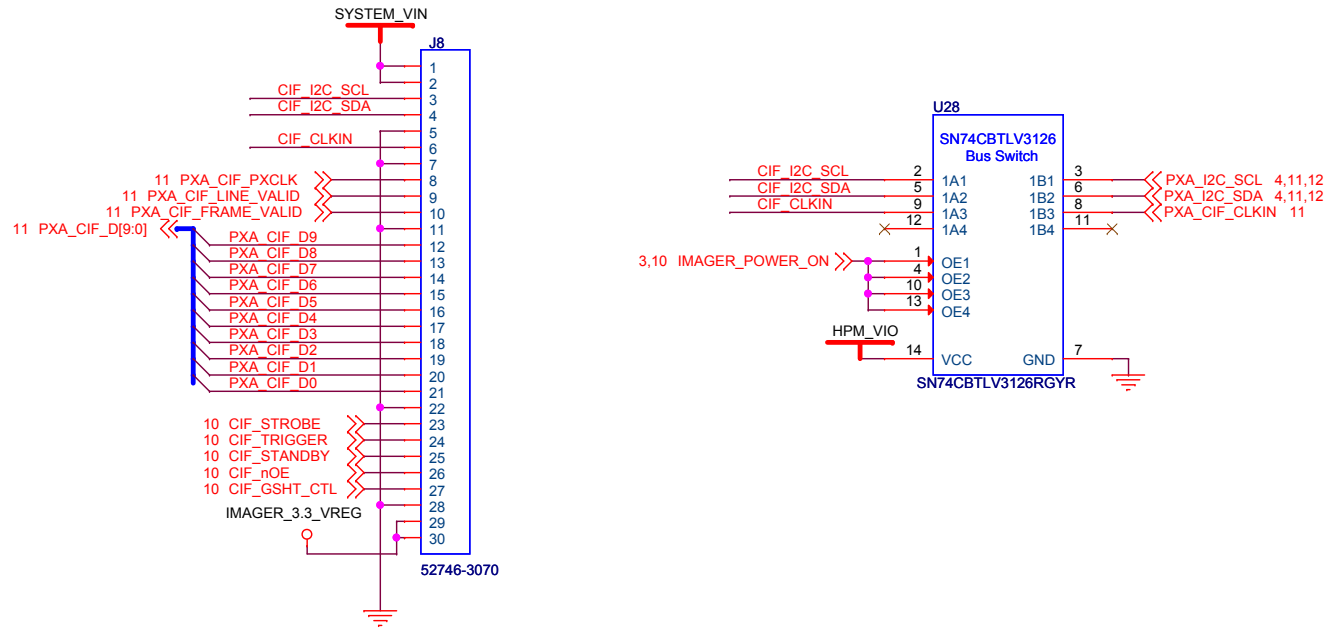


<Variant Name>



Title		EMAP2 - GPS	
Size	Document Number		
Custom	ASCENT-LEAP2-EMAP2	Rev	C
Date:	Saturday, November 03, 2012	Sheet	8 of 12

CMOS Imager Interface



<Variant Name>

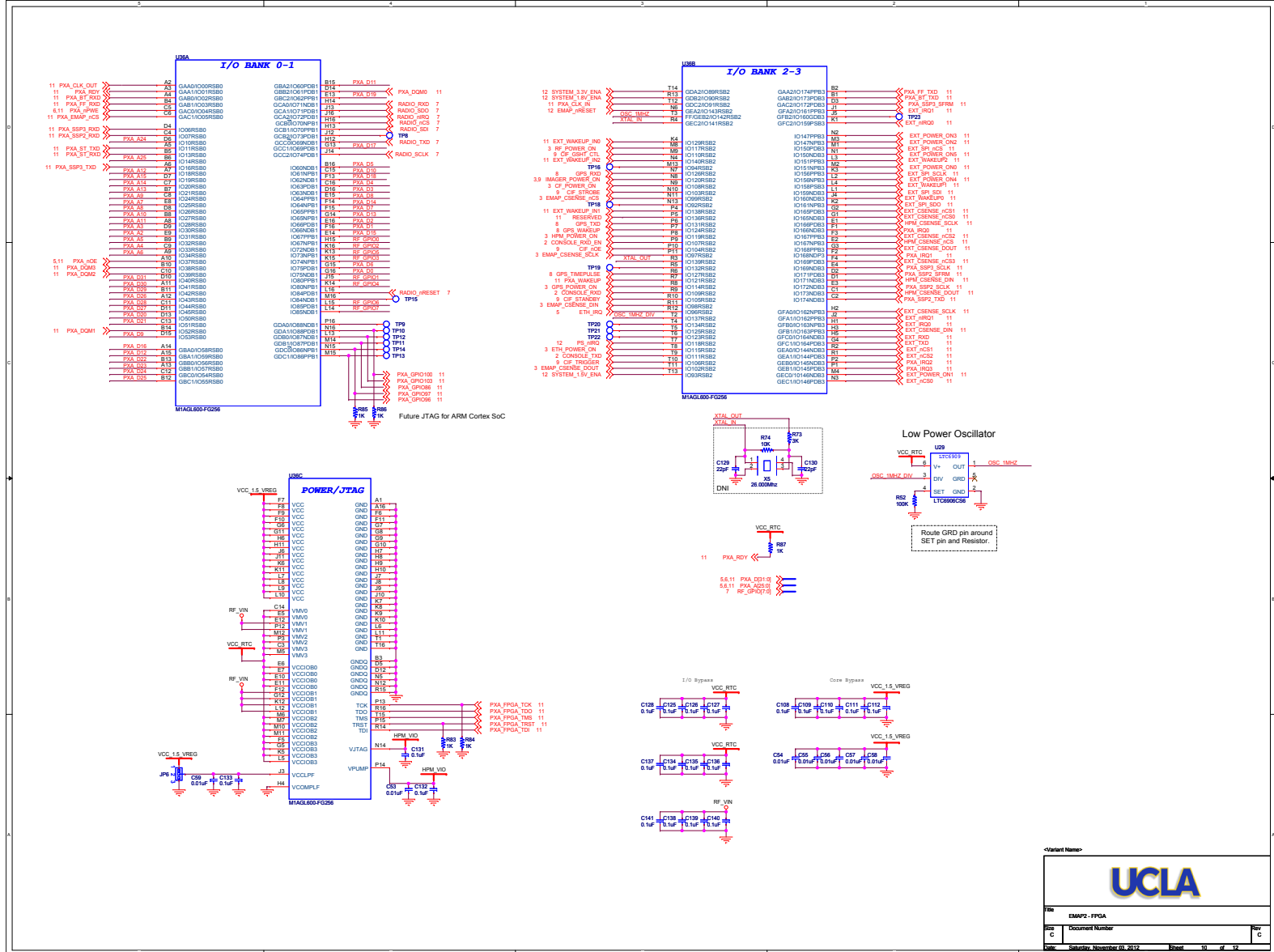


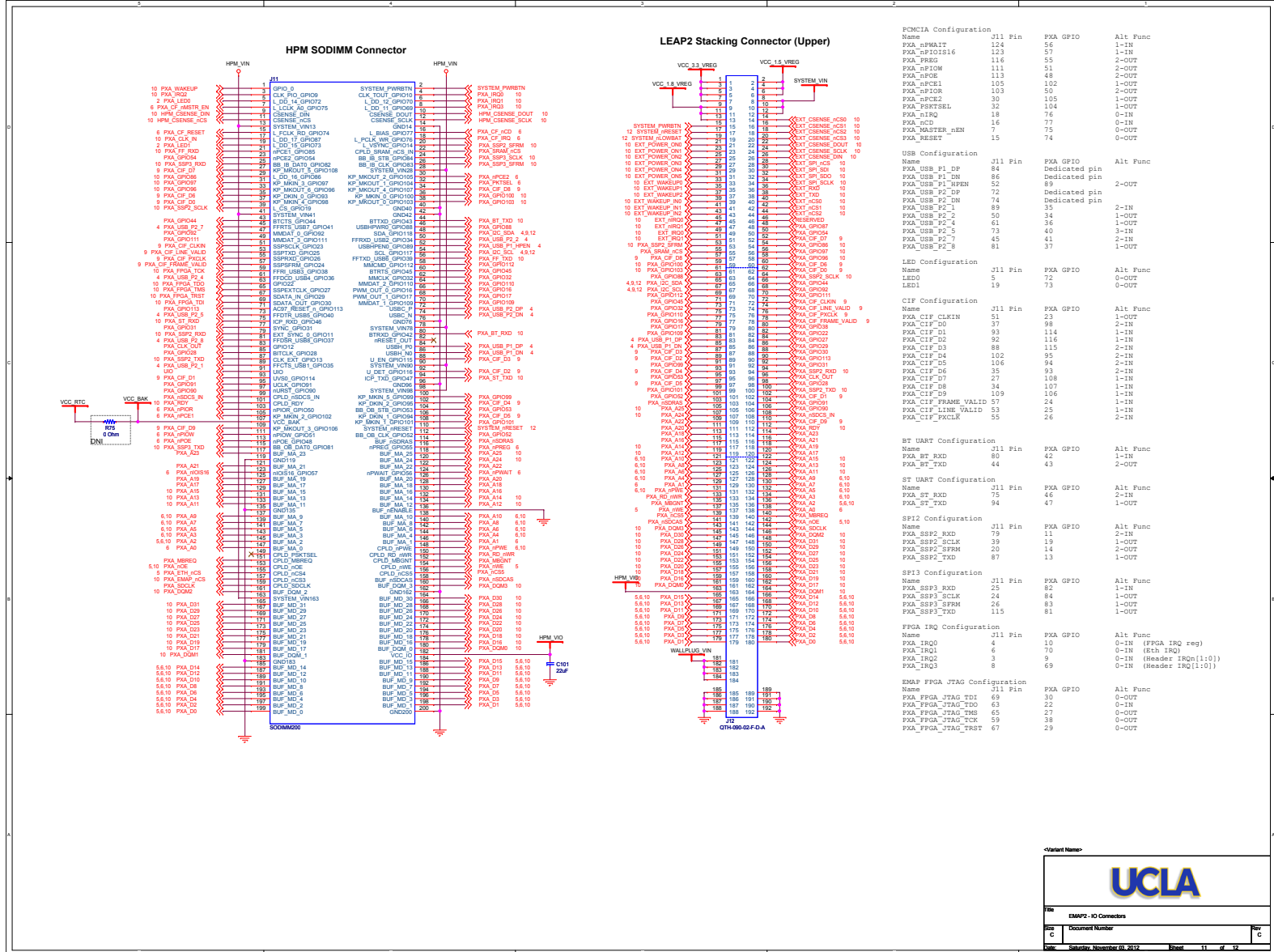
Title
EMAP2 - CMOS Imager

Size
A Document Number

Rev
C

Date: Saturday, November 03, 2012 Sheet 9 of 12





UCLA

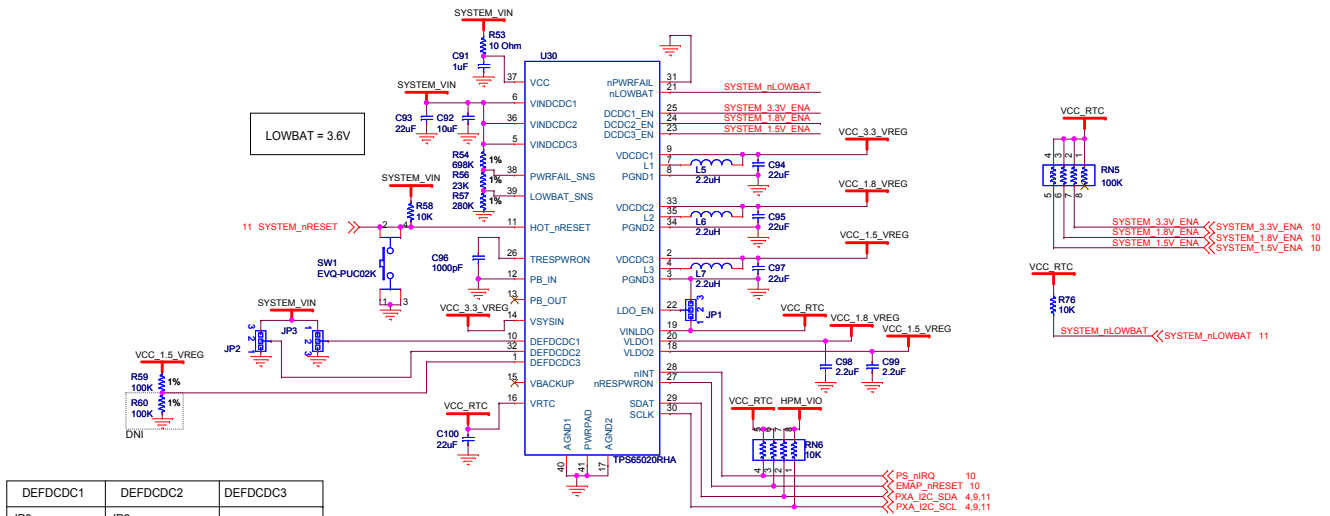
File: EMAP2 - IO Connectors

Doc: Schematic, November 01, 2012

Rev: C

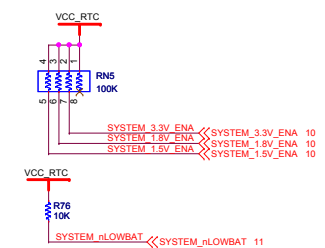
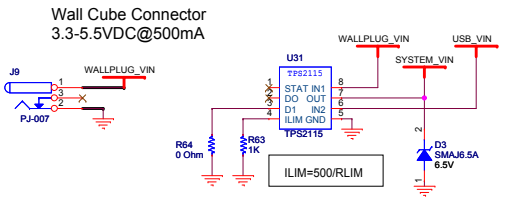
Sheet: 11 of 12

POWER SUPPLIES




LOWBAT = 3.6V

DEFDCDC1	DEFDCDC2	DEFDCDC3
JP3 1-2 VCCIO=3.3V 2-3 VCCIO=3.0V	JP2 1-2 VCCMEM=1.8V 2-3 VCCMEM=2.5V	Vout= 0.6*(R1+R2)/R2



- <> PS_nIRQ 10
- <> EMAP_nRESET 10
- <> FXA_I2C_SDA 4,9,11
- <> FXA_I2C_SCL 4,9,11

<Variant Name>



File: EMAP2 - Power Supply

Size: B Document Number: Rev: C

Date: Saturday, November 03, 2012 Sheet: 12 of 12

B.1.1.2 Layout

B.1.2 HPM

B.1.2.1 Schematics

LEAP2 - Host Processor Module

* Author: Dustin McIntire
* Copyright (c) 2006 The Regents of the University of California.
* All rights reserved.
* Permission to use, copy, modify, and distribute this software and its documentation for any purpose, without fee, and without written agreement is hereby granted, provided that the above copyright notice, the following two paragraphs and the author appear in all copies of this software.
* IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
* THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.*

Schematic Pages

1. Title Sheet
2. PXA270-1
3. PXA270-2
4. SDRAM
5. Flash and SRAM Tranceivers
- Power Supply
8. Current Sensing
9. I/O

LAYOUT NOTES:

1. No vias inside SMT caps and resistors.
2. No vias inside SMT pads.

Power Supply Hierarchy

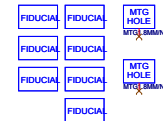
TBD

Reset Hierarchy

TBD

JTAG Chain

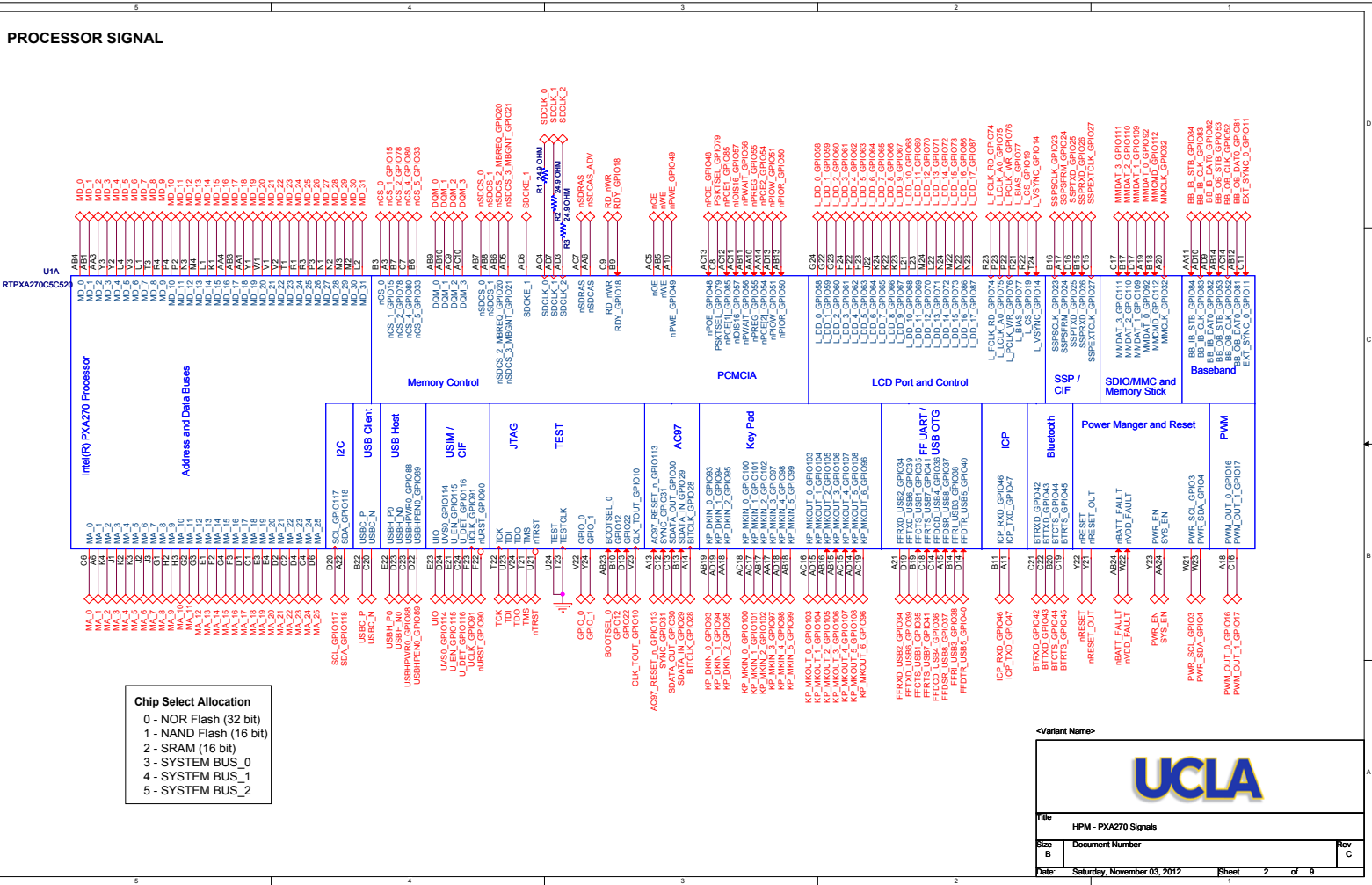
TBD

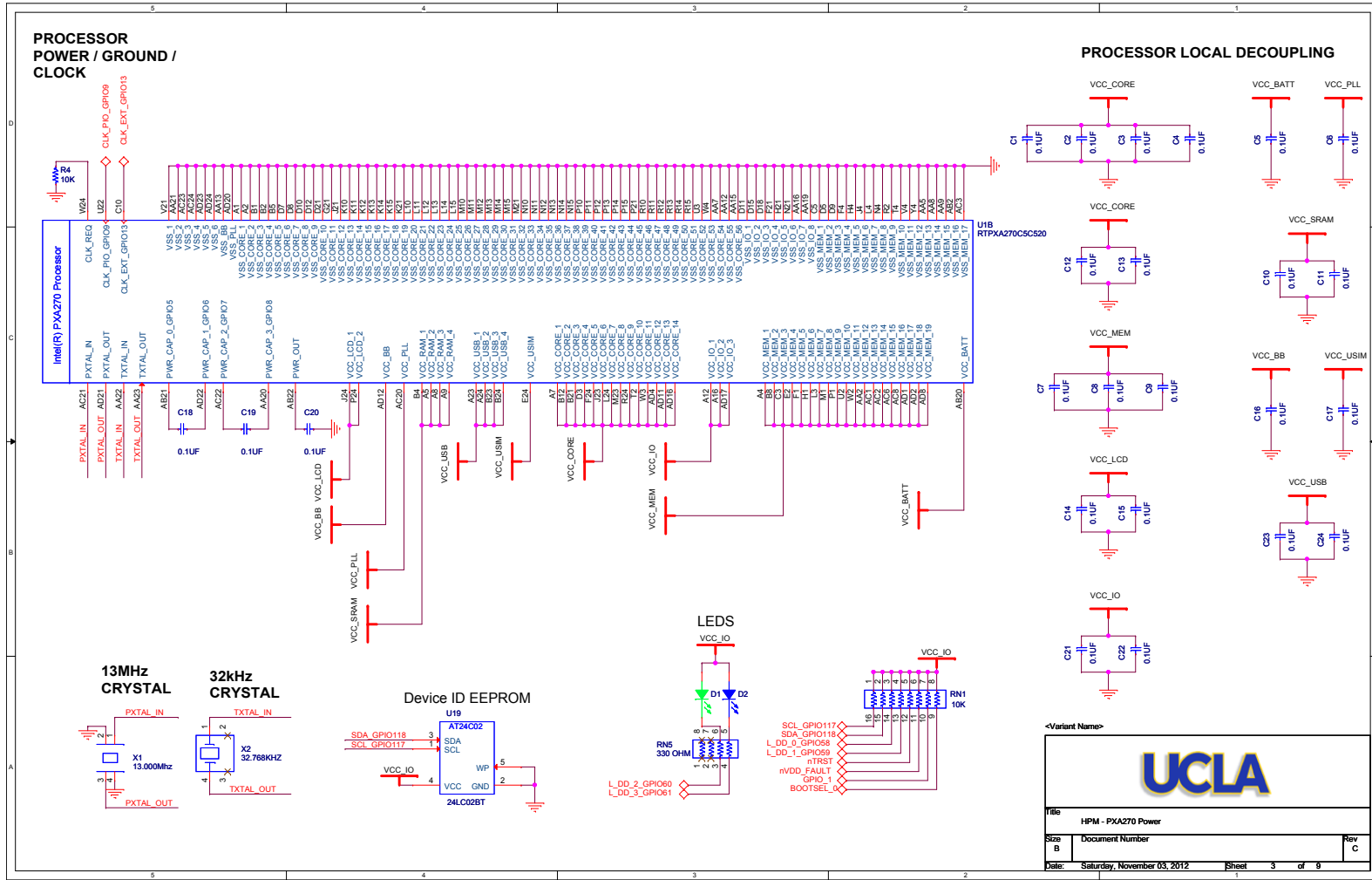


<Variant Name>

UCLA

File: HPM - Title Page
Size: C Document Number: Rev: C
Date: Saturday, November 01, 2014 Sheet: 1 of 9





PROCESSOR POWER / GROUND / CLOCK

PROCESSOR LOCAL DECOUPLING

13MHz CRYSTAL

32kHz CRYSTAL

Device ID EEPROM

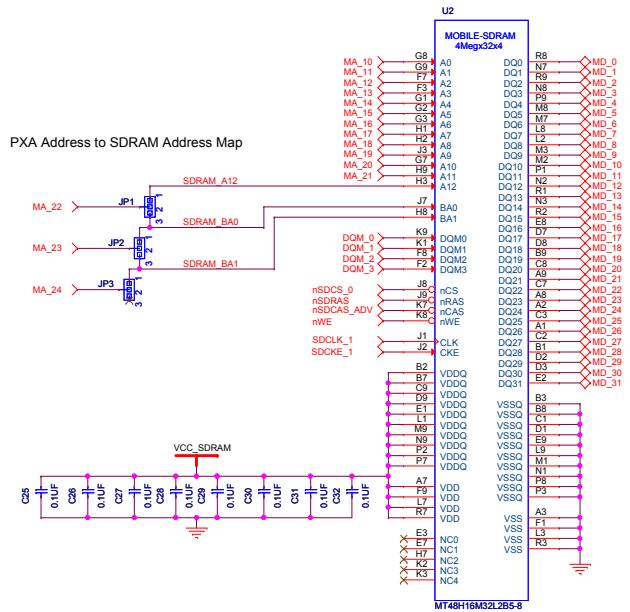
LEDS



<Variant Name>

File	HPM - PXA270 Power	
Size	Document Number	Rev
B		C
Date:	Saturday, November 03, 2012	Sheet 3 of 9

SDRAM



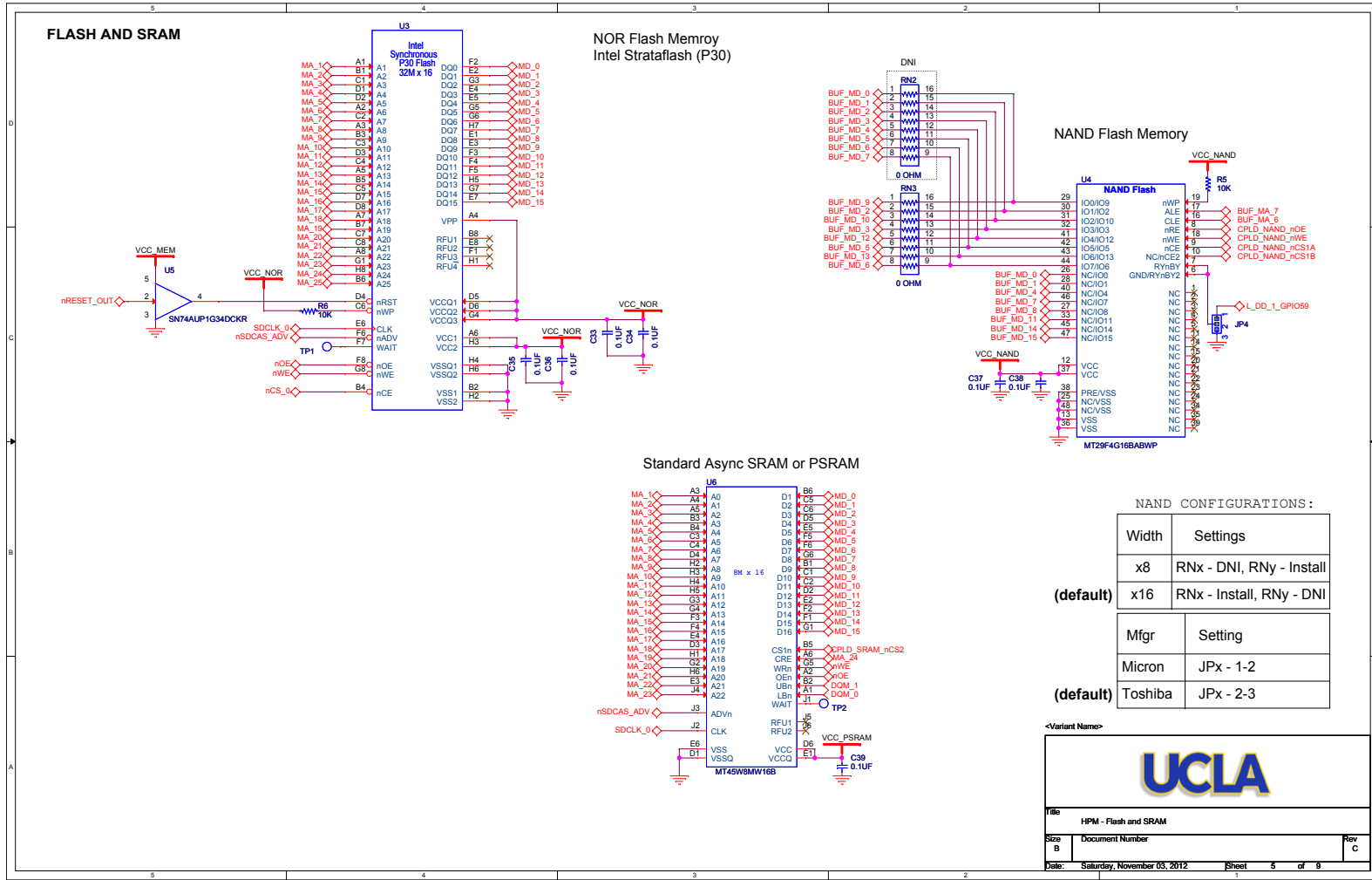
SDRAM CONFIGURATIONS:

	Bank Size (MByte)	Jumper Settings	SDRAM Parts
(default)	64MByte (512Mbit)	JP10,JP8,JP9 - 1-2	MT48H16M32L2B5
	32MByte (256Mbit)	JP10,JP8,JP9 - 2-3	MT48V8M32LFB5

<Variant Name>



File		
HPM - SDRAM		
Size	Document Number	Rev
B		C
Date: Saturday, November 03, 2012		
Sheet		4 of 9



FLASH AND SRAM

**NOR Flash Memory
Intel Stratataflash (P30)**

NAND Flash Memory

Standard Async SRAM or PSRAM

NAND CONFIGURATIONS :

Width	Settings
x8	RNx - DNI, RNy - Install
(default) x16	RNx - Install, RNy - DNI

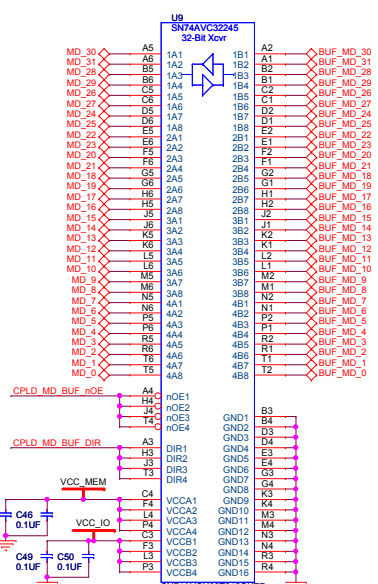
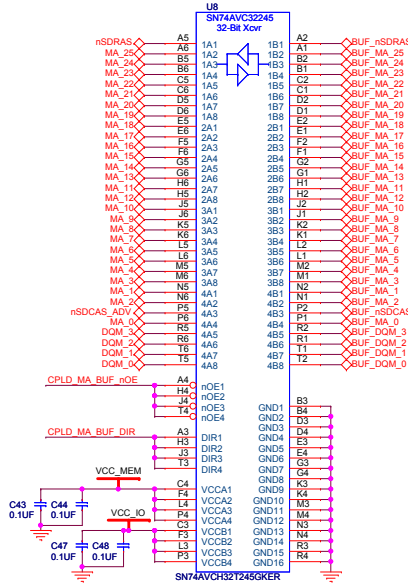
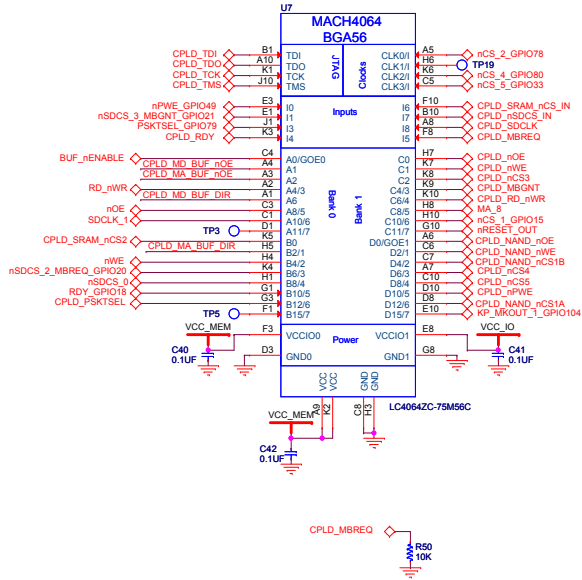
Mfgr	Setting
Micron	JPx - 1-2
(default) Toshiba	JPx - 2-3



<Variant Name>

File		HPM - Flash and SRAM	
Size	B	Document Number	Rev C
Date:	Saturday, November 03, 2012	Sheet	5 of 9

TRANCEIVERS/CPLD

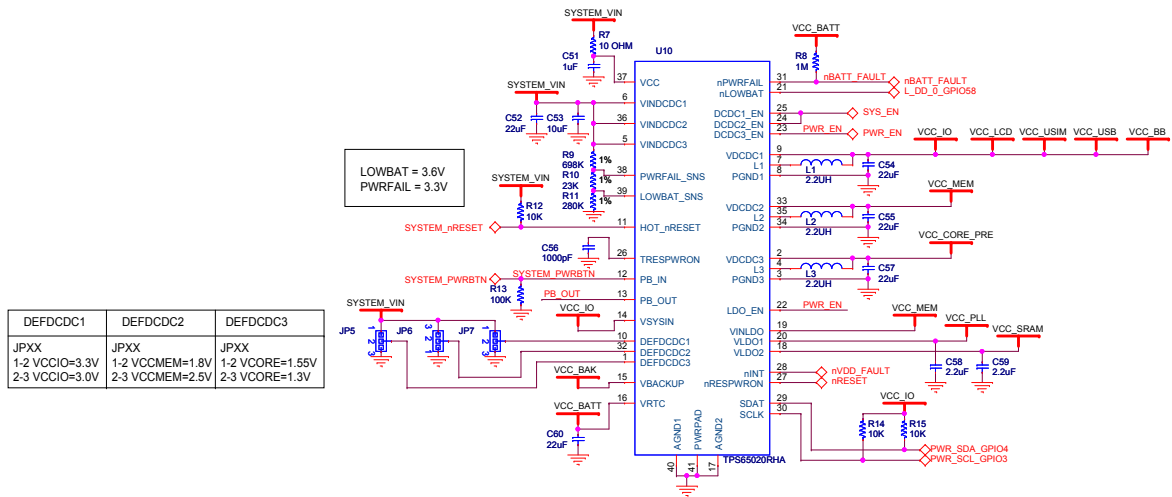


<Variant Name>



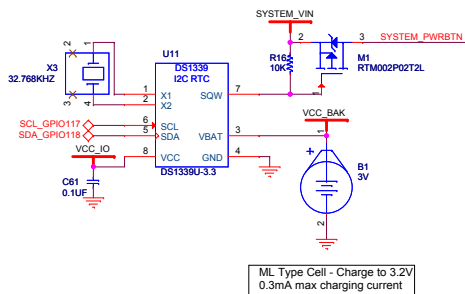
File		HPM - Tranceivers	
Size	B	Document Number	Rev C
Date:	Saturday, November 03, 2012	Sheet	6 of 9

POWER SUPPLIES

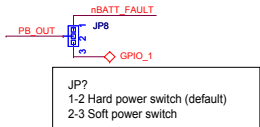


DEFDCDC1	DEFDCDC2	DEFDCDC3
JPXX	JPXX	JPXX
1-2 VCCIO=3.3V	1-2 VCCMEM=1.8V	1-2 VCCORE=1.55V
2-3 VCCIO=3.0V	2-3 VCCMEM=2.5V	2-3 VCCORE=1.3V


RTC and Trickle Charger



ML Type Cell - Charge to 3.2V
0.3mA max charging current



<Variant Name>

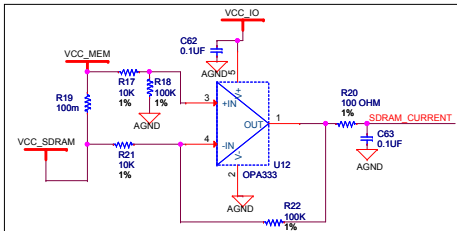


File: HPM - Power Supply

Size: B Document Number: Rev: C

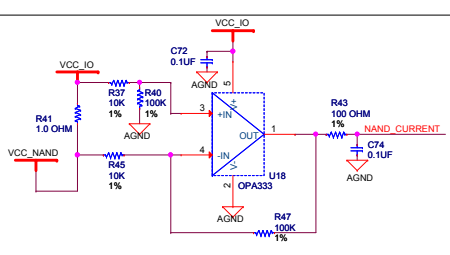
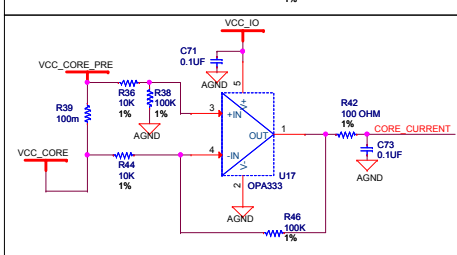
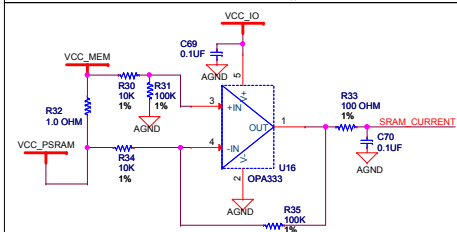
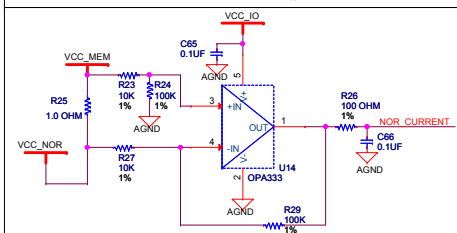
Date: Saturday, November 03, 2012 Sheet: 7 of 9

CURRENT SENSING

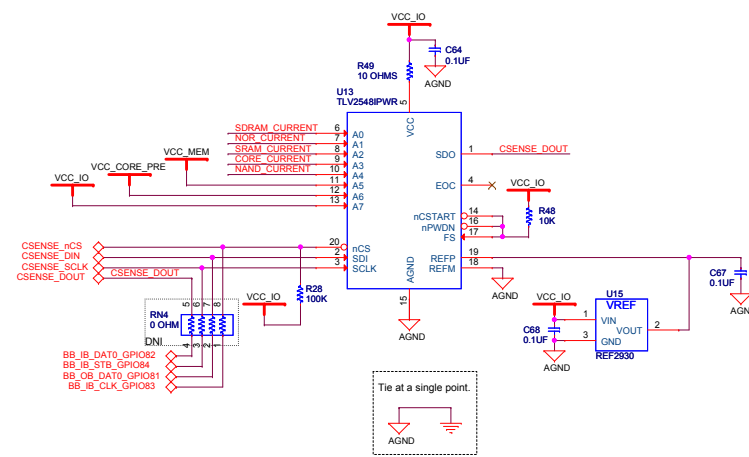


$$V_{out} = (100K/10K)(V_2 - V_1)$$

$$F_c = 1.6KHz$$



Current Sense ADC



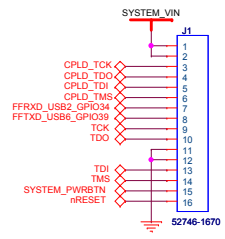
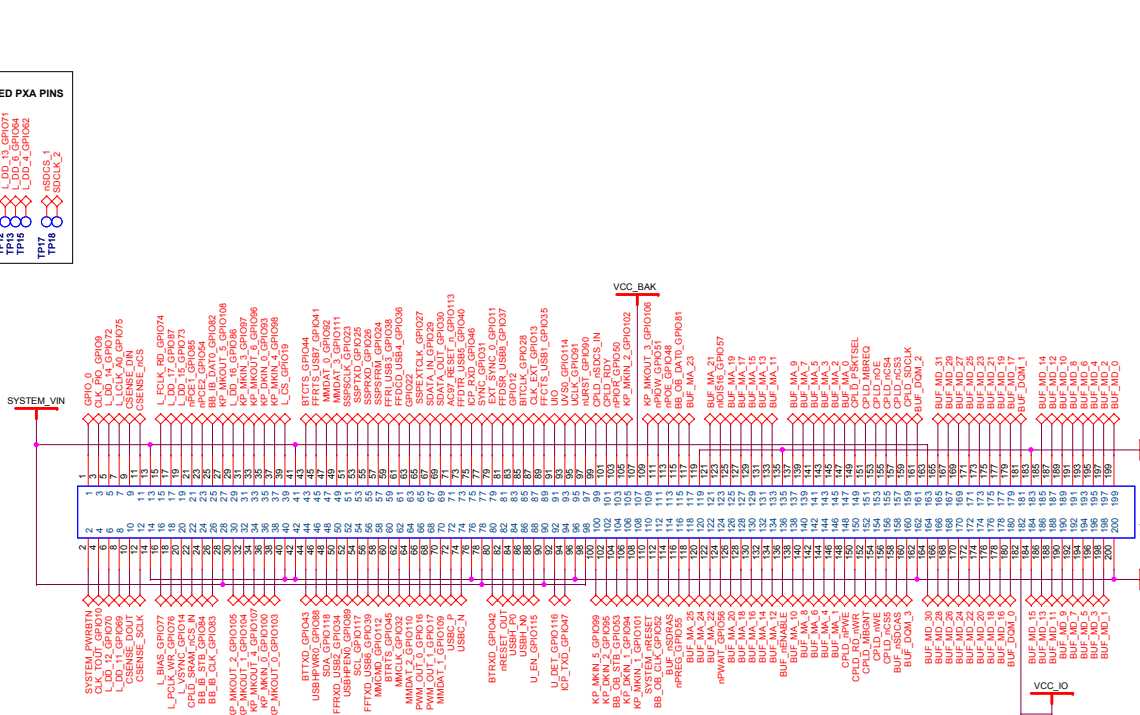
Tie at a single point.
AGND

<Variant Name>



File: HPM - Current Sensing		
Size: B	Document Number:	Rev: C
Date: Saturday, November 03, 2012	Sheet: 8	of 9

I/O CONNECTORS



<Variant Name>

File: HPM - I/O Connectors

Size: B Document Number Rev: C

Date: Saturday, November 03, 2012 Sheet: 9 of 9

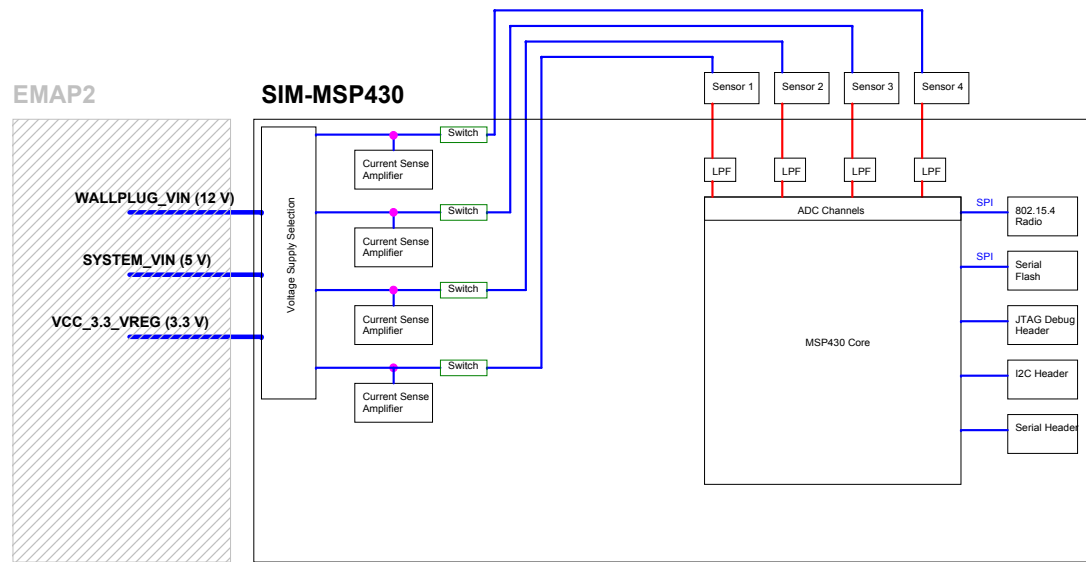
B.1.2.2 Layout

B.1.3 SIM

B.1.3.1 Schematics

LEAP2 - Sensor Interface Module (SIM-MSP430)

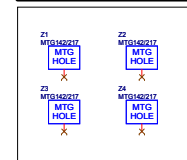
Block Diagram



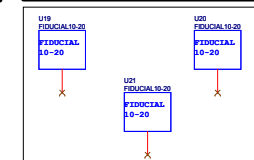
Schematics Pages

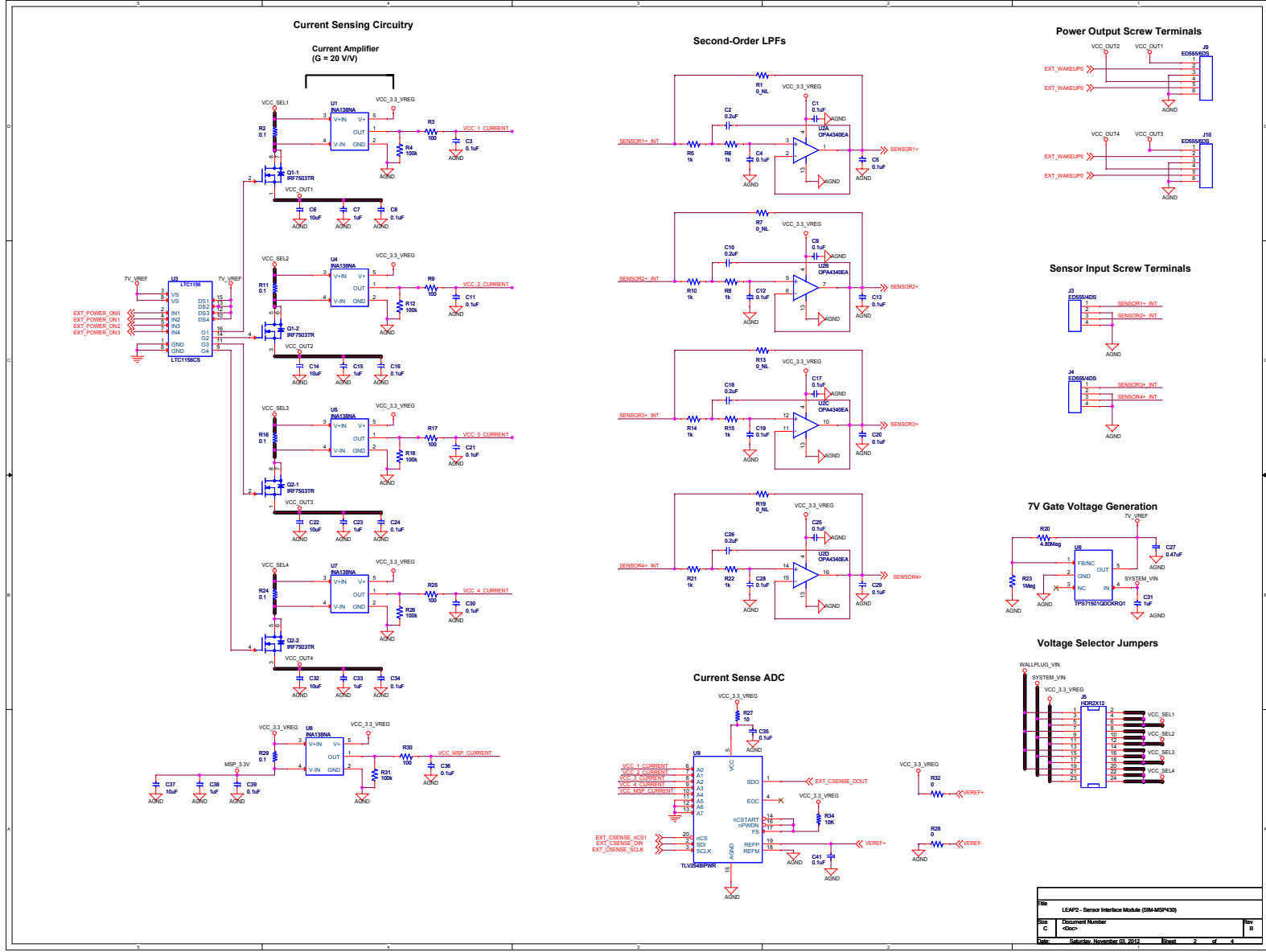
- (01) Title Page
- (02) Current Sensing
- (03) MSP430 Core & Peripherals
- (04) I/O Connectors

Mounting Holes

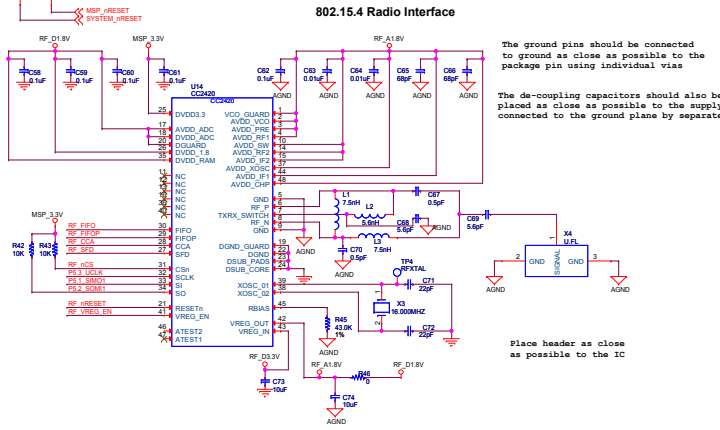
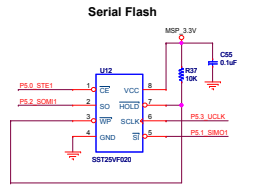
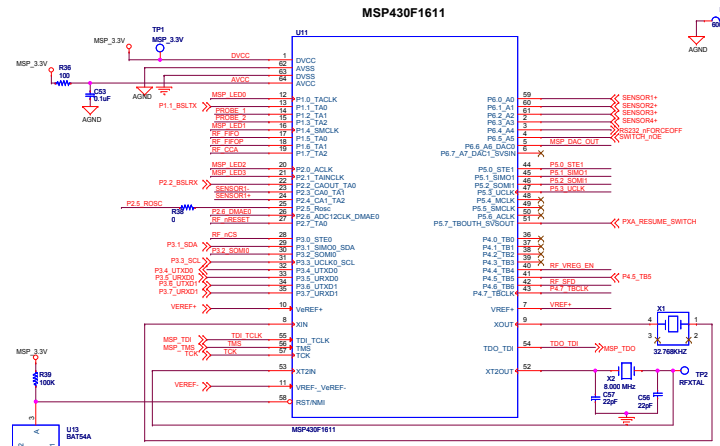
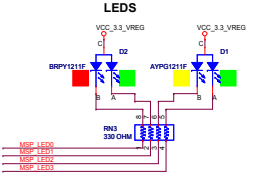
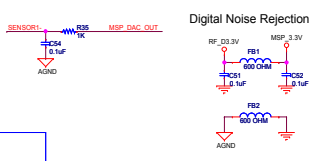
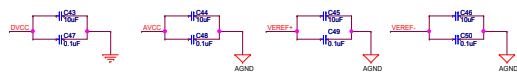


Fiducials





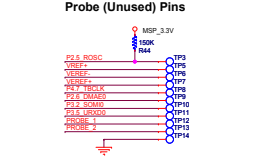
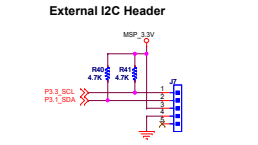
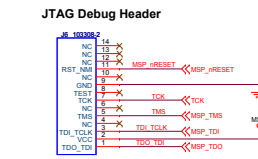
File	LEAP2 - Sensor Interface Module (GRAM5P43)	
Doc C	Document Number <Doc>	Rev B
Date	Saturday, November 03, 2012	Sheet 2 of 4



The ground pins should be connected to ground as close as possible to the package pin using individual vias

The de-coupling capacitors should also be placed as close as possible to the supply pins and connected to the ground plane by separate vias.

Place header as close as possible to the IC

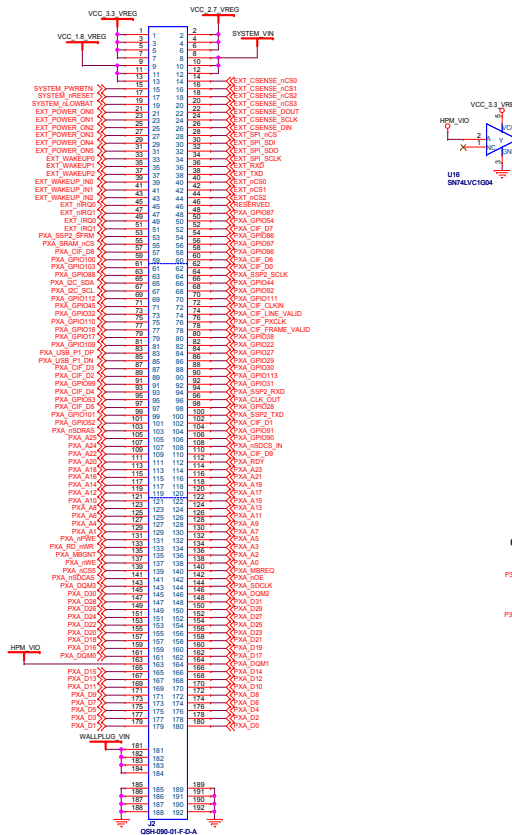


File	LEAP2 - Sensor Interface Module (GRM MSP430)	Rev	B
Size	Document Number		
C	<Doc>		
Date	Saturday, November 03, 2012	Sheet	3 of 4

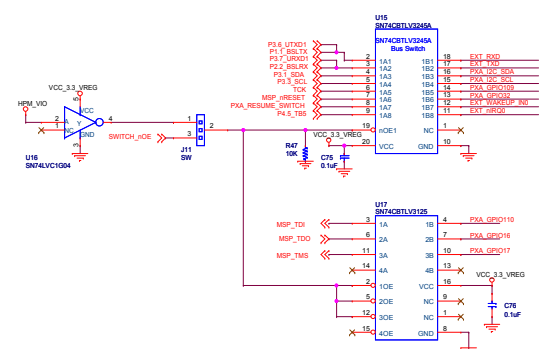
LEAP2 Stacking Connector (Upper)



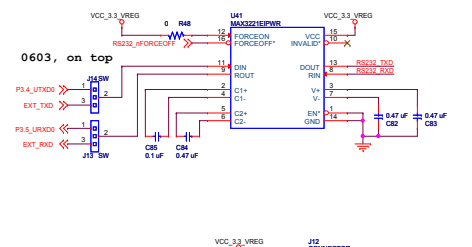
LEAP2 Stacking Connector (Lower)



SIM <-> EMAP2 (PXA) Bus Switch



RS-232 UART



File	LEAP2 - Sensor Interface Module (GRAM543X)	Rev	B
Doc	Document Number	<Doc>	
Date	Saturday, November 03, 2012	Sheet	4 of 4

B.1.3.2 Layout

B.1.4 MPM

B.1.4.1 Schematics

LEAP2 - Mini PCI Module

Schematic Pages

1. Title Sheet
2. Mini PCI and FPGA
3. Power Supplies
4. I/O Connectors

* Author: Dustin McIntire
* "Copyright (c) 2006 The Regents of the University of California.
* All rights reserved.
*
* Permission to use, copy, modify, and distribute this software and its
* documentation for any purpose, without fee, and without written agreement is
* hereby granted, provided that the above copyright notice, the following
* two paragraphs and the author appear in all copies of this software.
*
* IN NO EVENT SHALL THE UNIVERSITY OF CALIFORNIA BE LIABLE TO ANY PARTY FOR
* DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT
* OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE UNIVERSITY OF
* CALIFORNIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
*
* THE UNIVERSITY OF CALIFORNIA SPECIFICALLY DISCLAIMS ANY WARRANTIES,
* INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
* AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS
* ON AN "AS IS" BASIS, AND THE UNIVERSITY OF CALIFORNIA HAS NO OBLIGATION TO
* PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS."

LAYOUT NOTES:
1. No vias inside SMT caps and resistors.
2. No vias inside SMT pads.

Power Supply Hierarchy

TBD

Reset Hierarchy

TBD

JTAG Chain

TBD

MTG142217
MTG
HOLE
X

MTG142217
MTG
HOLE
X

MTG142217
MTG
HOLE
X

MTG142217
MTG
HOLE
X

<Variant Name>

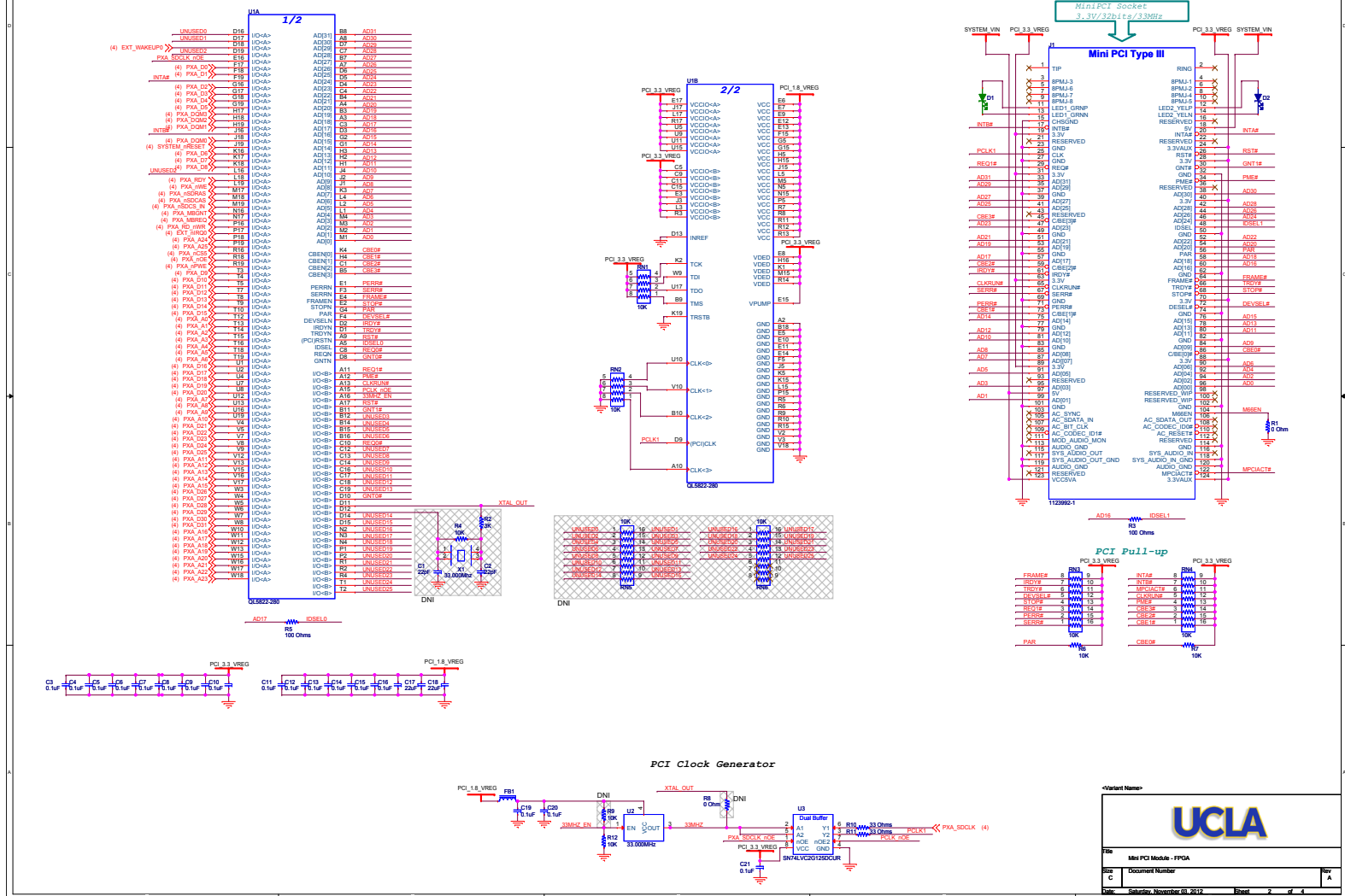
UCLA

File Mini PCI Module - Title Page

Size C Document Number Rev A

Date: Saturday, November 01, 2014 Sheet 1 of 4

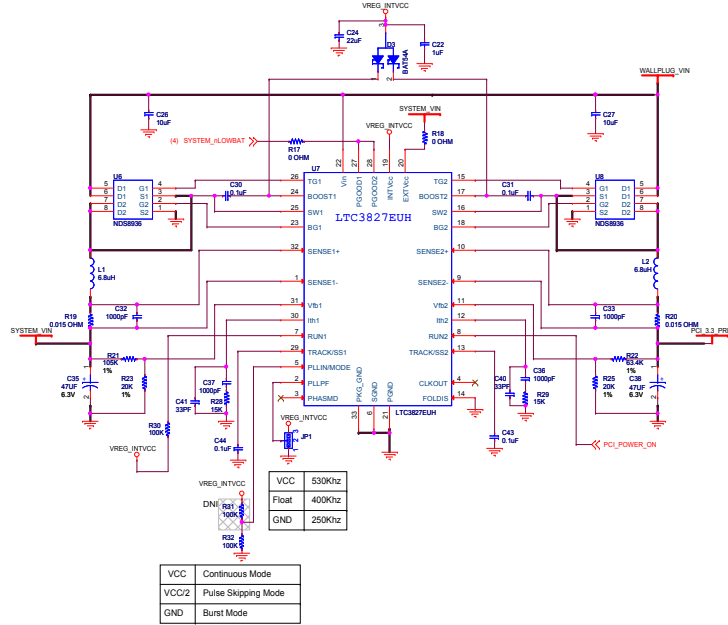
PCI Bridge FPGA



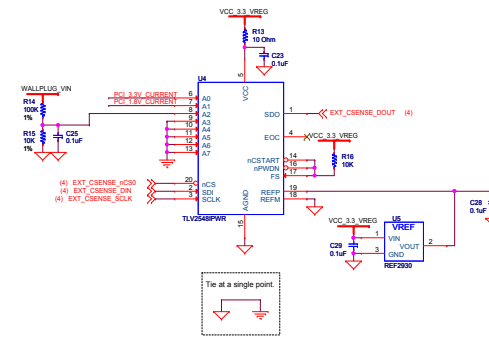
UCLA

File: Mini PCI Module - FPGA
 Size: C Document Number
 Date: September, November 01, 2012 Sheet: 2 of 4

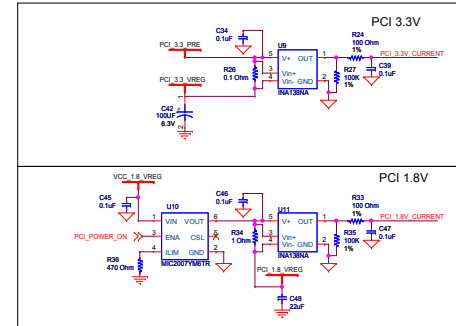
5.0V@2A and 3.3V@2A Power Supply



Current Sense ADC



CURRENT SENSING



<Variant Name>

UCLA

File Sensor Interface Module - ADC

Size C Document Number

Date: September, November 01, 2012 Sheet 3 of 4

Rev A

B.1.4.2 Layout

REFERENCES

- [ABF04] “Sensors for environmental observatories. Report of the NSF Sponsored Workshop.” <http://www.wtec.org/seo/final/2004>.
- [ACP99] J. Agre, L. Clare, G. Pottie, and N. Romanov. “Development platform for self-organizing wireless sensor networks.” *Aerosense, International Society of Optical Engineering*, pp. 257–268, 1999.
- [ACP05] ACPI. “Advanced Configuration and Power Interface Specification.” Technical report, December 30, 2005 2005.
- [ADL98] G. Asada, M. Dong, T. S. Lin, F. Newberg, G. Pottie, W. J. Kaiser, and H. O. Marcy. “Wireless Integrated Network Sensors: Low Power Systems on a Chip.” In *In Proceedings of the European Solid State Circuits Conference*, 1998.
- [ANF03] Manish Anand, Edmund B. Nightingale, and Jason Flinn. “Self-tuning wireless network power management.”, 2003.
- [ANF04] Manish Anand, Edmund B. Nightingale, and Jason Flinn. “Ghosts in the machine: interfaces for better power management.”, 2004.
- [ASS02] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. “Wireless Sensor Networks: A Survey.” *Computer Networks*, **38**:393–422, 2002.
- [AWB07] L.K. Au, W.H. Wu, M.A. Batalin, D.H. McIntire, and W.J. Kaiser. “MicroLEAP: Energy-aware Wireless Sensor Platform for Biomedical Sensing Applications.” pp. 158–62. IEEE Biomedical Circuits and Systems Conference (BIOCAS), November 2007.
- [Bar] “The Linux Kernel Tracepoint API.” <http://kernel.org/doc/htmldocs/tracepoint.html>.
- [BBC96] K. Bult, A. Burstein, D. Chang, M. Dong, M. Fielding, E. Kruglick, J. Ho, F. Lin, T. H. Lin, and W. J. Kaiser. “Low Power Systems for Wireless Microsensors.” In *IEEE International Symposium on Low Power Electronics and Design*, pp. 17–21, 1996.
- [BCE07] “GNU Radio - The GNU Software Radio.” <http://gnuradio.org>.
- [BCM00] Luca Benini, Giuliano Castelli, Alberto Macii, Enrico Macii, and Riccardo Scarsi. “Battery-driven dynamic power management of portable systems.”, 2000.

- [BDM99] Gaurav Banga, Peter Druschel, and Jeffrey C. Mogul. “Resource Containers: A New Facility for Resource Management in Server Systems.” *Operating Systems Design and Implementation*, pp. 45–58, 1999.
- [Bel00] Frank Bellosa. “The benefits of event: driven energy accounting in power-sensitive systems.”, 2000.
- [Ben11] “Developer Tool Spotlight - Using Trepn Profiler for Power-Efficient Apps.” <https://developer.qualcomm.com/blog/developer-tool-spotlight-using-trepn-profiler-power-efficient-apps>.
- [BLR01] Leonard Bacaud, Claude Lemarchal, Arnaud Renaud, and Claudia Sagastizbal. “Bundle Methods in Stochastic Optimal Power Management: A Disaggregated Approach Using Preconditioners.” *Comput. Optim. Appl.*, **20**(3):227–244, 2001.
- [Bri08] “Gartner Says More than 1 Billion PCs In Use Worldwide and Headed to 2 Billion Units by 2014.” <http://www.gartner.com/it/page.jsp?id=703807>.
- [BRV11] R. Bajwa, R. Rajagopal, P. Varaiya, and R. Kavalier. “In-pavement wireless sensor network for vehicle classification.” In *International Conference on Information Processing in Sensor Networks (IPSN)*, 2011.
- [BSC05] N. Banerjee, J. Sorber, M. Corner, S. Rollins, and D. Ganesan. “Triage: A power-aware software architecture for tiered microservers.” Technical report, University of Massachusetts-Amherst, April 2005.
- [Bsq12] “Snapdragon-based Mobile Development Platforms.” <http://www.bsquare.com/snapdragon-mobile-development-platforms.aspx>.
- [BSY04] M. Batalin, G. S. Sukhatme, Y. Yu, M. H. Rahimi, M. Hansen, G. Pottie, W. Kaiser, and D. Estrin. “Call and Response: Experiments in Sampling the Environment.” In *Proceedings of ACM SenSys, Baltimore, Maryland*, 2004.
- [CB95] A. P. Chandrakasan and R. W. Brodersen. *Low Power Digital CMOS Design*. Kluwer Academic Publishers, 1995.
- [CG11] “SmartReflex Power and Performance Management Technologies: reduced power consumption, optimized performance, white paper.” http://focus.ti.com/pdfs/wtbu/smartreflex_whitepaper.pdf.

- [Col03] D. Cole. *Energy Consumption and Personal Computers*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2003.
- [DGA05] Prabal Dutta, Mike Grimmer, Anish Arora, Steven Bibyk, and David Culler. “Design of a wireless sensor network platform for detecting rare, random, and ephemeral events.”, 2005.
- [DKM94] Fred Douglass, P. Krishnan, and Brian Marsh. “Thwarting the power-hungry disk.”, 1994.
- [DPN02] L. Dexin, H. C. Pai, and B. Nader. “Mode Selection and Mode-Dependency Modeling for Power-Aware Embedded Systems.” In *In Proceedings of the 2002 conference on Asia South Pacific design automation/VLSI Design*, 2002.
- [Eff12] “Babeltrace.” <http://www.efficios.com/babeltrace>.
- [EPS01] D. Estrin, G. Pottie, and M. Srivastava. “Instrumenting the world with wireless sensor networks.” In *Proceedings of ICASSP Conference*, 2001.
- [Fou] “Eclipse.” <http://www.eclipse.org/>.
- [FS99a] Jason Flinn and M. Satyanarayanan. “Energy-aware adaptation for mobile applications.”, 1999.
- [FS99b] Jason Flinn and M. Satyanarayanan. “PowerScope: A Tool for Profiling the Energy Usage of Mobile Applications.”, 1999.
- [Gas02] M. S. Gast. *802.11 Wireless Networks: The Definitive Guide*. O’Reilly & Associates, Inc., 2002.
- [GH96] R. Gonzalez and M. Horowitz. “Energy dissipation in general purpose microprocessors.” *IEEE Journal of Solid-State Circuits*, **31**(9):1277–1284, 1996.
- [GMJ04] Contreras Gilberto, Martonosi Margaret, Peng Jinzhan, Ju Roy, and Lueh Guei-Yuan. “XTREM: a power simulator for the Intel XScale core.”, 2004.
- [GSI02] Sudhanva Gurusurthi, Anand Sivasubramaniam, Mary Jane Irwin, N. Vijaykrishnan, and Mahmut Kandemir. “Using complete machine simulation for software power estimation: the SoftWatt approach.” *Eighth International Symposium on High-Performance Computer Architecture*, pp. 141–150, 2002.

- [HCA02] Zeng Heng, S. Ellis Carla, R. Lebeck Alvin, and Vahdat Amin. “ECOSystem: managing energy as a first class operating system resource.”, 2002.
- [HLS00] David P. Helmbold, Darrell D. E. Long, Tracey L. Sconyers, and Bruce Sherrod. “Adaptive disk spin-down for mobile computers.” *Mobile Networks and Applications*, 5(4):285–297, 2000.
- [HPB02] T. Heath, E. Pinheiro, and R. Bianchini. “Application-supported Device Management for Energy and Performance.”, February 2002.
- [HPH02] Taliver Heath, Eduardo Pinheiro, Jerry Hom, Ulrich Kremer, and Ricardo Bianchini. “Application Transformations for Energy and Performance-Aware Device Management.”, 2002.
- [HPK03] Hai Huang, Padmanabhan Pillai, and G Shin Kang. “Design and implementation of power-aware virtual memory.”, 2003.
- [HRA07] M. P. Hamilton, P. W. Rundel, M. A. Allen, W. Kaiser, D. E. Estrin, and E. Graham. “New Approaches in Embedded Networked Sensing for Terrestrial Ecological Observatories.” *Environmental Engineering Science*, pp. 192–204, 2007.
- [HSW00] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David E. Culler, and Kristofer S. J. Pister. “System Architecture Directions for Networked Sensors.” *Architectural Support for Programming Languages and Operating Systems*, pp. 93–104, 2000.
- [HWL84] D.J. Houlston, G. Waugh, and J. Laughlin. “Automatic real-time event detection for seismic networks.” In *Comput. Geosci.*, volume 10, pp. 431–436, 1984.
- [ifi12] “Samsung Galaxy Nexus Teardown.” <http://www.ifixit.com/Teardown/Samsung+Galaxy+Nexus+Teardown/7182/1>.
- [Ins] “Digital Bus Switch Selection Guide.” <http://www.ti.com/general/docs/lit/getliterature.tsp?baseLiteratureNumber=scdb006>.
- [Int02] “Stargate Platform.” <http://platformx.sourceforge.net>.
- [Int07] “Imote2.” <http://www.xbow.com>.

- [JDC07] Xiaofan Jiang, Prabal Dutta, David Culler, and Ion Stoica. “Micro power meter for energy monitoring of wireless sensor networks at scale.”, 2007.
- [JSR04] P. Jones, Padmanabhan Shobana, D. Rymarz, J. Maschmeyer, D. V. Schuehler, J. W. Lockwood, and R. K. Cytron. “Liquid architecture.” In *Parallel and Distributed Processing Symposium, 2004. Proceedings. 18th International*, p. 202, 2004.
- [JTB07] D. Jung, A Teixeira, A. Barton-Sweeney, and A. Savvides. “Model-based Design Exporation of Wireless Sensor Networks at Scale.”, 2007.
- [KA00] A. Kamerman and G. Aben. “Net throughput with IEEE 802.11 wireless LANs.”, 2000.
- [LBM00] Yung-Hsiang Lu, Luca Benini, and Giovanni De Micheli. “Low-power task scheduling for multiple devices.”, 2000.
- [LC03] Hyung Gyu Lee and Naehyuck Chang. “Energy-aware memory allocation in heterogeneous non-volatile memory systems.”, 2003.
- [LDP02] Kanishka Lahiri, Sujit Dey, Debashis Panigrahi, and Anand Raghunathan. “Battery-Driven System Design: A New Frontier in Low Power Design.”, 2002.
- [Lin07a] Linux. “The Linux Operating System.”, 2007.
- [Lin07b] “PowerTOP.” <http://www.awaretechs.com>.
- [LJ01] Jiong Luo and Niraj K. Jha. “Battery-aware static scheduling for distributed real-time embedded systems.”, 2001.
- [LM99] Y. Lu and G. De Micheli. “Adaptive Hard Disk Power Management on Personal Computers.” *IEEE Great Lakes Symposium on VLSI*, pp. 50–53, 1999.
- [LPZ07] Dimitrios Lymberopoulos, Nissanka B. Priyantha, and Feng Zhao. “mPlatform: a reconfigurable architecture and efficient data sharing mechanism for modular sensor nodes.”, 2007.
- [LRR05] Nachman Lama, Kling Ralph, Adler Robert, Huang Jonathan, and Hummel Vincent. “The Intel Mote platform: a Bluetooth-based sensor network for industrial monitoring.”, 2005.

- [LS05] Dimitrios Lymberopoulos and Andreas Savvides. “XYZ: a motion-enabled, power aware sensor node platform for distributed sensor network applications.”, 2005.
- [Mar00] Diana Marculescu. “Profile-driven code execution for low power dissipation.”, 2000.
- [Mar06] John Markoff. “Google to Push for More Electrical Efficiency in PCs.”, September 26 2006.
- [Mar11] “Data Centers Power Use Less Than Was Expected.” <http://www.nytimes.com/2011/08/01/technology/data-centers-using-less-power-than-forecast-report-says.html>.
- [MGP82] Andrew J Michael, Stephen P Gildea, and Jay J. Pulli. “A real-time digital seismic event detection and recording system for network applications.” In *Bulletin of the Seismological Society of America* 72, pp. 2339–2348, December 1982.
- [Mic01] Microsoft. “OnNow Device Power Management.”, 2001.
- [Mic06] Microsoft. “Power Policy Configuration and Deployment in Windows Vista.” Technical report, Microsoft Corporation, October 30 2006.
- [Mic12] “The ultra-low-power programmable solution.” <http://www.actel.com/products/IGLOO/>.
- [MS96] T. Martin and D. Siewiorek. “A power metric for mobile systems.”, 1996.
- [MSR12] Dustin McIntire, Thanos Stathopoulos, Sasank Reddy, Thomas Schmidt, and William J. Kaiser. “Energy-Efcient Sensing with the Low Power, Energy Aware Processing (LEAP) Architecture.”, 2012.
- [MV04a] Aqeel Mahesri and Vibhore Vardhan. “Power consumption breakdown on a modern laptop.” In *Proceedings of the 4th international conference on Power-Aware Computer Systems*, pp. 165–180, 2004.
- [MV04b] Aqeel Mahesri and Vibhore Vardhan. “Power Consumption Breakdown on a Modern Laptop.” *PACS*, **3471**:165–180, 2004.
- [OAB03] C. M. P. Ozanne, D. Anhuf, S. L. Boulter, M. Keller, R. L. Kitching, C. Korner, F. C. Meinzer, A. W. Mitchell, T. Nakashizuka, P. L. S. Dias, N. E. Stork, S. J. Wright, and M. Yoshimura. “Biodiversity meets the atmosphere: A global view of forest canopies.” *Science*, **301**:183–6, 2003.

- [Ope12a] “SoC Interconnection: Wishbone.” <http://opencores.org/opencores,wishbone>.
- [Ope12b] “Linux Trace Toolkit.” <http://ltnng.org/>.
- [PK05] Greg Pottie and William Kaiser. *Principles of Embedded Networked Systems Design*. Cambridge University Press, 2005.
- [PS03] Athanasios E. Papathanasiou and Michael L. Scott. “Energy efficiency through burstiness.” *Mobile Computing Systems and Applications, 2003. Proceedings. Fifth IEEE Workshop on*, pp. 44–53, 2003.
- [PSC05] Joseph Polastre, Robert Szewczyk, and David Culler. “Telos: enabling ultra-low power wireless research.”, 2005.
- [QQM00] Qiu Qinru, Wu Qing, and Pedram Massoud. “Dynamic power management of complex systems using generalized stochastic Petri nets.”, 2000.
- [Qua11] “Snapdragon S4 Processors: System on Chip Solutions for a New Mobile Age, White Paper.” <http://www.qualcomm.com/media/documents/snapdragon-s4-processors-system-chip-solutions-new-mobile-age>.
- [Qui] “Eclipse II.” <http://www.quicklogic.com/support/mature-products/eclipse-ii/overview/>.
- [RCB06] Jose Rizo-Morente, Miguel Casas-Sanchez, and C. J. Bleakley. “Dynamic current modeling at the instruction level.”, 2006.
- [RG00] Dinesh Ramanathan and Rajesh Gupta. “System level online power management algorithms.”, 2000.
- [RGK02] K. W. Roth, F. Goldstein, and J. Kleinman. “Energy Consumption by Office and Telecommunications Equipment in Commercial Buildings, Volume I: Energy Consumption Baseline.” Technical report, Arthur D. Little Inc, Cambridge, MA, January 2002.
- [RIG02] D. Ramanathan, S. Irani, and R. Gupta. “An Analysis of System Level Power Management Algorithms and their Effects on Latency.”, 2002.
- [Rus02] Charlie Russel. “Power Management in Windows XP.”, March 25 2002.

- [SBC05] B. Schott, M. Bajura, J. Czarnaski, J. Flidr, T. Tho, and L. Wang. “A modular power-aware microsensor with $\geq 1000X$ dynamic power range.” In *Information Processing in Sensor Networks*, pp. 469–474, Los Angeles, 2005.
- [SBM99] Tajana Simunic, Luca Benini, and Giovanni De Micheli. “Energy-efficient design of battery-powered embedded systems.”, 1999.
- [SBM00] Tajana Simunic, Luca Benini, Giovanni De Micheli, and Mat Hans. “Source code optimization and profiling of energy consumption in embedded systems.”, 2000.
- [SCI01] Vishnu Swaminathan, Krishnendu Chakrabarty, and S. S. Iyengar. “Dynamic I/O power management for hard real-time systems.”, 2001.
- [SHC04] Victor Shnayder, Mark Hempstead, Bor-rong Chen, Geoff Werner Allen, and Matt Welsh. “Simulating the power consumption of large-scale sensor network applications.” *Proceedings of the 2nd international conference on Embedded Networked Sensor Systems (SenSys04)*, pp. 188–200, 2004.
- [SKW01] S. Steinke, M. Knauer, L. Wehmeyer, and P. Marwedel. “An Accurate and Fine Grain Instruction-Level Energy Model supporting Software Optimizations.”, September 2001.
- [TEK] “High Resolution Seismic Recorders.” <http://www.reftek.com/products/high-resolution-seismic-recorders.htm>.
- [TRJ05] T. K. Tan, A. Raghunathan, and N. K. Jha. “Energy macromodeling of embedded operating systems.” *Trans. on Embedded Computing Sys.*, 4(1):231–254, 2005.
- [UK03] Osman S. Unsal and Israel Koren. “System-level power-aware design techniques in real-time systems.” *Proceedings of the IEEE*, 91(7):1055–1069, 2003.
- [Wai03] Martin Waitz. *Accounting and control of power consumption in energy-aware operating systems*. Diplomarbeit, Friedrich Alexander Universität Erlangen-Nürnberg, 2003.
- [WAY98] Mitchell Withers, Richard Aster, Christopher Young, Judy Beiriger, and Mark Harris. “A comparison of select trigger algorithms for automated global seismic phase and event detection.” In *Bulletin of the Seismological Society of America* 88, pp. 95–106, February 1998.

- [WBB02] Andreas Weissel, Bjoern Beutel, and Frank Bellosa. “Cooperative I/O-A Novel I/O Semantics for Energy-Aware Applications.” In *5th USENIX Symposium on Operating Systems Design and Implementation (OSDI’02)*,, 2002.
- [WBB08] Winston H. Wu, Alex A. T. Bui, Maxim A. Batalin, Lawrence K. Au, Jonathan D. Binney, and William J. Kaiser. “Medic: Medical embedded device for individualized care.” *Artificial Intelligence for Medicine*, **42**(2):137–52, 2008.
- [WBF04] C. Worth, M. Bajura, J. Flidr, and B. Schott. “On-demand Linux for Power-aware Embedded Sensors.” In *Ottawa Linux Symposium*, Ottawa, Ontario Canada, 2004.
- [Wik07] “A Typical North/Southbridge Layout.” <http://www.wikipedia.org>.
- [Wik11] “Map of aftershock activity near Japan as of March 31, 2011.” <http://www.wikipedia.org>.
- [ZBS04] V Zyuban, D Brooks, Viji Srinivasan, M Gschwind, Pradip Bose, P N Strenski, and P G Emma. “Integrated analysis of power and performance for pipelined microprocessors.” *IEEE Transactions on Computers*, **53**(8):1004–1016, 2004.
- [ZGS03] F. Zheng, N. Garg, S. Sobti, C. Zhang, R. Joseph, A. Krishnamurty, and R. Wang. “Considering the energy consumption of mobile storage alternatives.” In *IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, 2003.
- [ZTQ10] Lide Zhang, Birjodh Tiwana, Zhiyun Qian, Zhaoguang Wang, Robert P. Dick, Zhuoqing Morley Mao, and Lei Yang. “Accurate online power estimation and automatic battery behavior based power model generation for smartphones.” In *Proceedings of the eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, New York, NY, 2010.
- [ZTX11] Zhou Zhu-yu, Deng Tian-min, and Lv Xian-yang. “Study for vehicle recognition and classification based on Gabor wavelets transform & HMM.” In *International Conference on Consumer Electronics, Communications and Networks (CECNet)*, 2011.