

AFRL-IF-RS-TR-2005-343
Final Technical Report
September 2005



POWER AWARE REMOTE INFORMATION SYSTEM (PARIS)

Raytheon Company

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. P459

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

STINFO FINAL REPORT

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2005-343 has been reviewed and is approved for publication

APPROVED: /s/

WILMAR SIFRE
Project Engineer

FOR THE DIRECTOR: /s/

JAMES A. COLLINS, Deputy Chief
Advanced Computing Division
Information Directorate

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 074-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE SEPTEMBER 2005	3. REPORT TYPE AND DATES COVERED Final Jun 02 – Dec 04	
4. TITLE AND SUBTITLE POWER AWARE REMOTE INFORMATION SYSTEM (PARIS)			5. FUNDING NUMBERS C - F30602-02-C-0089 PE - 63739E PR - PARI TA - S0 WU - 01	
6. AUTHOR(S) Julius Boganowicz				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Raytheon Company 2000 East El Segundo Boulevard El Segundo California 90245			8. PERFORMING ORGANIZATION REPORT NUMBER N/A	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Defense Advanced Research Projects Agency AFRL/IFTC 3701 North Fairfax Drive 26 Electronic Parkway Arlington Virginia 22203-1714 Rome New York 13441-4514			10. SPONSORING / MONITORING AGENCY REPORT NUMBER AFRL-IF-RS-TR-2005-343	
11. SUPPLEMENTARY NOTES AFRL Project Engineer: Wilmar Sifre/IFTC/(315) 330-2075/ Wilmar.Sifre@rl.af.mil				
12a. DISTRIBUTION / AVAILABILITY STATEMENT APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.				12b. DISTRIBUTION CODE
13. ABSTRACT (Maximum 200 Words) A sensor network is characterized as a set of locally powered sensors linked together via wireless communication with the objective to accurately assess activity within the sensor network field of regard and forward that assessment to command and control elements for appropriate action. To minimize power utilization within that network, each power consuming element must be evaluated for effectiveness to the overall network objective. An assessment of the overall sensor network power optimization was performed in a realistic framework based upon the exploitation of orthogonal sensing elements, environmental situation characterization, and invocation of adaptive processing algorithms in a power aware computing and communicating structure. Issues such as the collective fixed and variable power costs associated with localized versus centralized processing, or the issue of variable costs associated with memory access and storage costs were explicitly evaluated in order to assess the system power efficiency tradeoffs. Robust processing algorithms were included in an adaptive processing suite that selects the optimum algorithm based on available power, target type, image quality, and track history. The architecture developed supports voltage and frequency scaling, sleep modes and power switching that allows the Reduced Instruction Set Computer (RISC) processor to be turned off when not needed.				
14. SUBJECT TERMS Power Aware, PASTA, Infrared (IR) Sensor/Camera, Probability Of Detection, Probability Of False Alarm, MILAN, Distributed Sensor Network, Human Detection, Tracking Algorithm, Wireless Communication, Perimeter Security, Homeland Security, Power Tradeoff, Frequency And Voltage Scaling, Adaptive Algorithms			15. NUMBER OF PAGES 108	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	

TABLE OF CONTENTS

1. SUMMARY.....	1
2. PROJECT DESCRIPTION.....	2
2.1 RESEARCH OBJECTIVES	2
2.1.1. <i>Problem Description</i>	2
2.1.2 <i>Research Goals</i>	2
2.1.3. <i>Expected Impact</i>	2
3. TECHNICAL APPROACH	3
3.1. DETAILED DESCRIPTION OF TECHNICAL APPROACH	3
3.2. PARIS DETECTION AND TRACKING ALGORITHM.....	4
4. PARIS NODE EVOLUTION	4
4.1. PARIS CONFIGURATION 4.....	4
4.2. THE PARIS PASTA NODE.....	5
APPENDIX A: POWER AS FUNCTION OF CONFIGURATION.....	7
APPENDIX B: ENHANCED PARIS PASTA NODE	9
APPENDIX C: PARIS REQUIREMENTS SPECIFICATION & TECHNOLOGY PROJECTIONS	11
APPENDIX D: PARIS EMULATION TESTBED DESIGN.....	41
APPENDIX E: HIGH LEVEL PERFORMANCE ESTIMATION AND VALIDATION.....	52
APPENDIX F: PARIS HARDWARE MODULE DESIGN.....	66
APPENDIX G: ALGORITHM DESIGN	91
APPENDIX H: ALGORITHM DESCRIPTION / MAPPING TO PARIS NODE.....	99

LIST OF FIGURES

FIGURE 1 - PARIS DEVELOPMENT APPROACH	3
FIGURE 2. SYSTEM CONFIGURATION 4 SETUP	5
FIGURE 3: THE PARIS PASTA NODE	5

1. Summary

The Power Aware Remote Sensing System (PARIS) program was able to significantly exceed the objectives of the program. The average final power number achieved was 0.8 Watts versus the goal of 1.25 Watts. The PARIS node, using a 5 second duty cycle, is able to operate for 15 hours and 40 minutes using 6 AA batteries. The performance of the algorithm was able to achieve a 90% increase in Probability of detection (Pd) while reducing the Probability of false alarm (Pfa) by 99%.

This report is organized as follows:

- a. Project Description / Objectives
- b. Technical Approach
- c. PARIS node evolution
- d. Appendix A: Power as a Function of Configuration
- e. Appendix B: Enhanced PARIS node
- f. Appendix C: PARIS Requirements Specification & Technology Projections
- g. Appendix D: PARIS Emulation Testbed Design
- h. Appendix E: High Level Performance Estimation & Validation using Model-based Integrated Simulation (MILAN) Framework
- i. Appendix F: PARIS Hardware Module Design
- j. Appendix G: Algorithm Design
- k. Appendix H: Algorithm Description / Mapping to PARIS node

2. PROJECT DESCRIPTION

2.1 Research Objectives

Based on realistic DoD mission needs and environmental conditions, our objective was to develop a robust hardware and software structure of a heterogeneous processing and communicating node for a distributed sensor network that is power aware and cost effective for wide spread DoD usage.

2.1.1. Problem Description

Military applications and platforms have an urgent and growing need for power efficient computing and communications. More power efficient systems would: a) empower platforms to perform new missions, b) enable dramatically extended mission timelines, c) enable new capabilities on existing missions, and d) reduce logistics costs by requiring fewer energy resources and less frequent replenishment cycles. The technology base exists to address power efficient computing. Researchers have been able to develop point solutions that exhibit up to three orders of magnitude power reduction (as measured by energy-delay product or performance/watt) over conventional computing/communication approaches. A comprehensive, systematic program is needed to take that technology base in power aware computing and create broad, general purpose, energy efficient strategies applicable to a wide range of military platforms. Power Aware Computing and Communication (PAC/C) will provide a novel integrated software/hardware technology suite incorporating innovative individual power reduction technologies. This will enable embedded computing systems to reduce power requirements up to 100X – 1000X as measured by energy-delay product or performance per watt metrics.

2.1.2 Research Goals

	Month 18	Month 24	Month 30
Algorithms Performance relative to Baseline	10% improvement in Probability of Detection & 25% reduction in False Alarms over baseline algorithms		10% improvement in Probability of Detection & 50% reduction in False Alarms for enhanced algorithms
Power Reduction	2x (composite result based on use of more computationally intensive algorithms & use of power aware hardware)	5x (enhanced result based on execution on PARIS node)	10x (composite result based on efficient mapping of algorithms to final PARIS node (more power efficient) & use of power aware processing framework)
Performance Comparison	Energy Savings Relative to Baseline Algorithms - 80x	Energy Savings Relative to Baseline Algorithms - 200x	Energy Savings Relative to Baseline Algorithms - 400x
Mission Duration	2x – avg 6.25 Watts	5x – avg 2.5 Watts	10x – avg 1.25 Watts

2.1.3. Expected Impact

Mission duration will be extended by 10 X while increasing system algorithm performance through the use of more robust human detection and tracking algorithms adapted to be power aware.

3. TECHNICAL APPROACH

The PARIS team leveraged selected results from PAC/C Phase 1 (MILAN, USC-ISI “Stack Architecture”) to: 1) define application / mission requirements and refine system concept, 2) develop model / performance enhancements to the MILAN framework, 3) develop a flexible emulation test bed to evaluate alternative architecture implementation approaches and to validate MILAN tool results, 4) design and fabricate an enhanced PARIS processing node based on analysis / experimentation results and 5) evaluate / demonstrate PARIS node in a distributed sensor environment.

3.1. Detailed Description of Technical Approach

The phasing of the effort as executed is shown in figure 1.

- Leverages Phase 1 Results, Tools, & USC-ISI Power Aware Sensing, Tracking and Analysis (PASTA) HW/SW, Raytheon Sensor Testbed
- Exploits Joint Data Collections & Final Field Demo with USC-ISI PASTA Effort
- Enhanced MILAN Tools will be used to Support Architecture Trades / Design

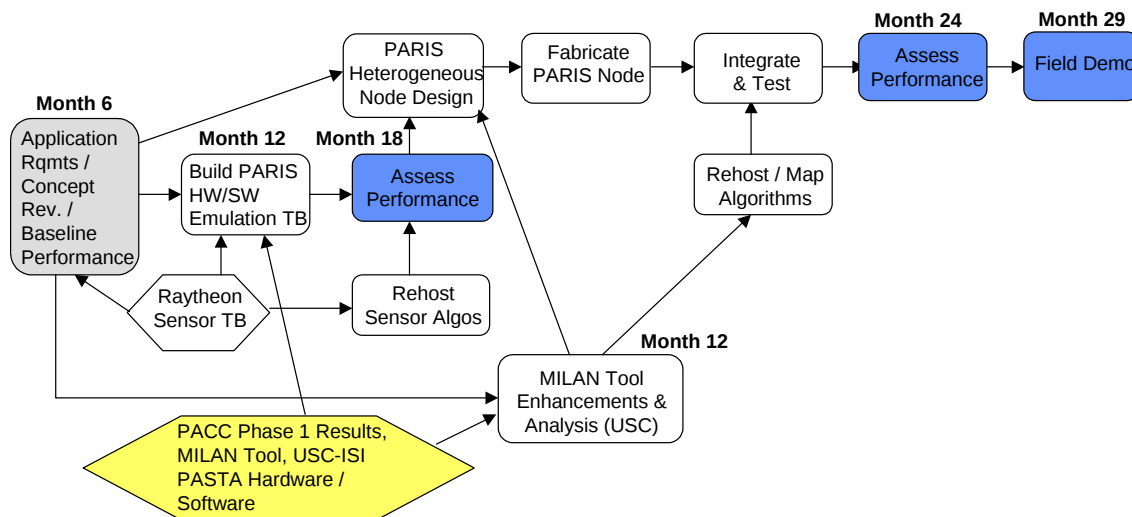


Figure 1 - PARIS Development Approach

The design of the PARIS node went through several iterations to minimize the power usage while maximizing algorithm performance. The current version of the PARIS node consists of a C8051 microcontroller with a TI C6713 floating point processor to do the algorithm number crunching. The PARIS node can operate either stand-alone or in conjunction with the PASTA stack. This document summarizes the effort to optimize the performance of the algorithm on the PARIS node while minimizing the power utilization on this platform configuration. The system duty cycle controls how images are acquired, processed to obtain a result and then to place major components within the PARIS node in a sleep or powered down state to minimize power utilization.

A high performance embedded processor was chosen for the system. The processor chosen consumes significant power but offers reduced execution time. The digital signal

processor (DSP) takes advantage of simplified scheduling opportunities available in the embedded environment. The embedded system is an interrupt driven design. This enables the PARIS image processor to sleep when there is no work to be done and to wake as processing is needed. To implement the design in this way Hardware Interrupts (HWI) are used to trigger event based activities including data collection and communication. Because HWIs are reserved for real time activities, they are designed to consume as little time as possible. These functions often trigger additional work to be performed from functions called Software Interrupts (SWI). SWIs are used when a lengthy amount of processing is to be performed, but does not require real time handling. In addition to the use of HWI and SWI the system utilizes Enhanced Direct Memory Access (EDMA). EDMA is used to handle data collection from the Multi-channel Buffered Serial Port (McBSP). The advantages of using the EDMA are two fold. It allows transmission at rates too high for a polling method of data collection to consistently capture all of the data. It also offloads processing that would have to be performed by the Central Processing Unit (CPU) to internal hardware. This allows the CPU to perform other processing in parallel with data collection. By allowing the CPU to perform image processing while data collection is occurring further increases in speedup can be achieved (approximately 1.28).

3.2. PARIS Detection and Tracking Algorithm

The Fast Adaptive Spatial Temporal algorithm was implemented on the PARIS node. A comparative test was performed with the baseline differencing algorithm and it was shown that the Pd was increased by 90% while reducing false alarms by 99%.

4. PARIS NODE EVOLUTION

In order to prove the energy saving concept of the PARIS project (Power Aware Remote Information Sensing) at Raytheon, two major versions of the hardware were developed. The PARIS Configuration 4 was an initial step to limit the system power consumption to 2.5 Watts, and the Compact PARIS PASTA node would further reduce system power to 1.25 Watts. Appendix A showed detail component power consumptions. Appendix B showed the current developing Enhanced PARIS PASTA with Motes radio and sensor board.

4.1. PARIS Configuration 4

The PARIS Configuration 4 included a Raytheon PARIS Controller board that controlled and supplied the 3.3v sources to a Raytheon IR Camera and an off-the-shelf Orsys DSP board. The IR Camera sent video stream to the Orsys DSP board a parallel bus interface. Figure 2 shows a picture of the setup. The setup consumed **2.5 Watts** to perform continuous target tracking 3 image scan per second.

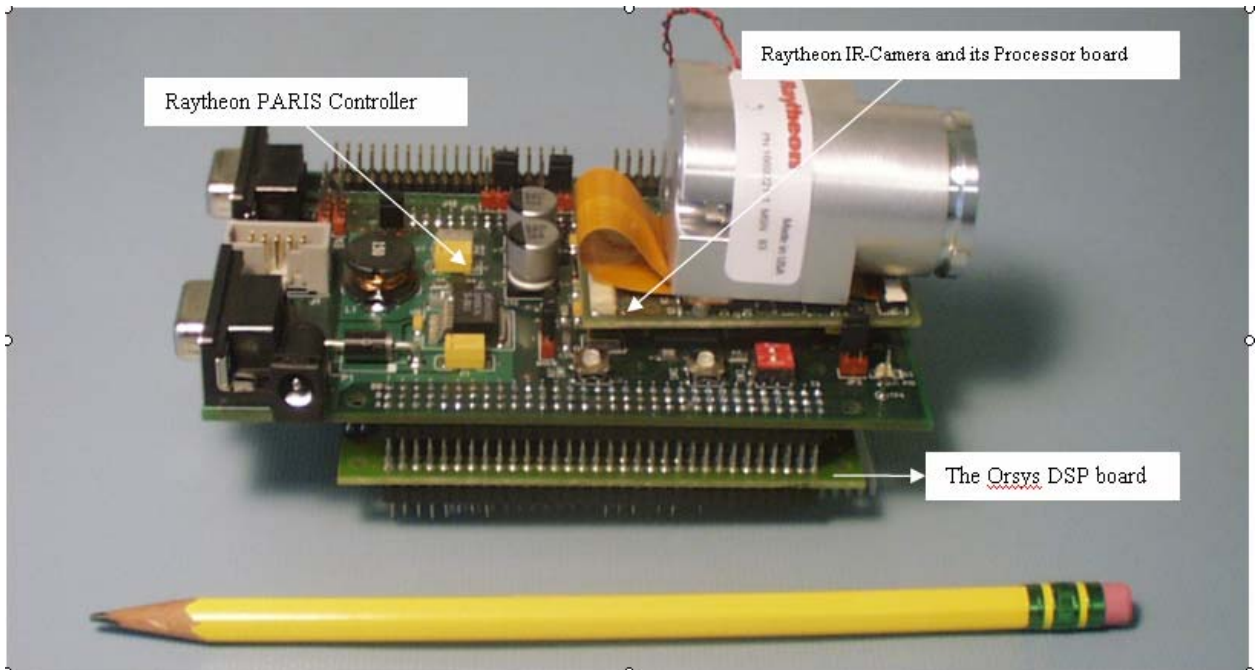


Figure 2. System Configuration 4 Setup

4.2. The PARIS PASTA Node

To further reduce power consumption of the PARIS Configuration 4 and to comply with the PASTA stack for added wireless sensing applications, the Compact PARIS board was designed to replace both the PARIS controller board and the Orsys DSP board. The Compact PARIS board was powered by the PASTA IO board. The Camera sent video stream to Compact PARIS board via the low Voltage Differential Signal (LVDS) Reformatter board. Figure 3 showed a picture of the PARIS PASTA node.

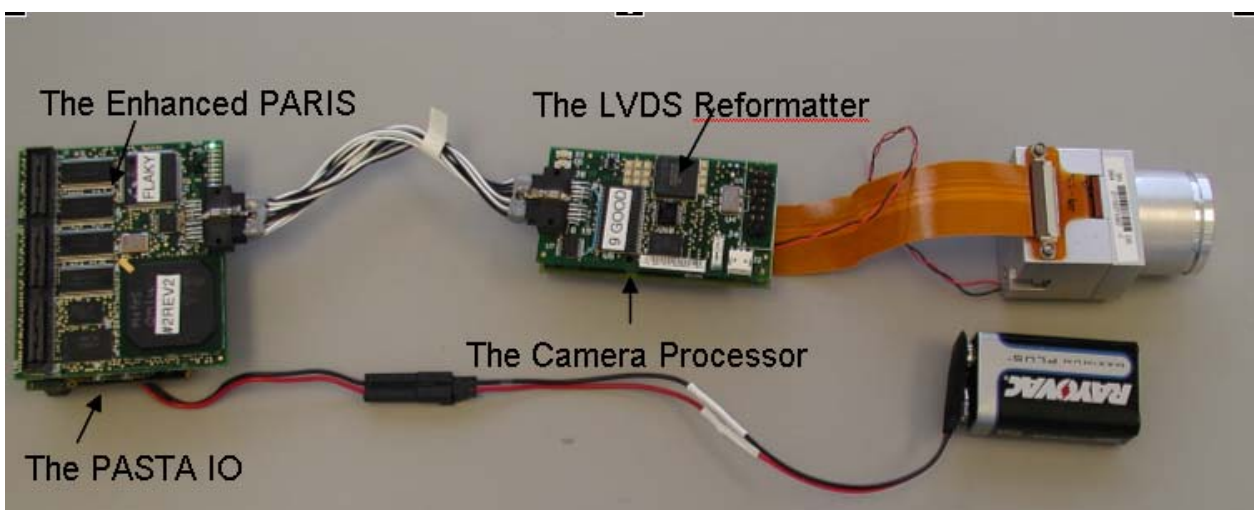
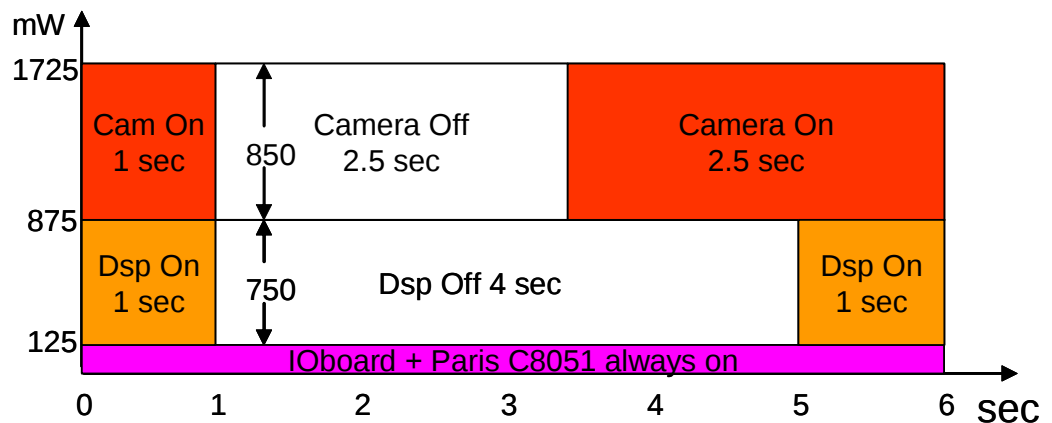


Figure 3: The PARIS PASTA Node

The PARIS PASTA node consumed **1.7 W** for continuous tracking in three image scan per second mode. In this mode, the PARIS PASTA node can perform continuous tracking for 70 minutes on a 9V alkaline battery and 9 hrs on six AA alkaline batteries. Please view Table 2 in appendix A for more details.

In some applications, the PARIS PASTA node can operate in a 5 second duty cycle mode. In this mode, it tracks for one second and goes into limbo for the next 5 seconds. This mode allows enough time for the system to turn on only the need resources to save power as shown in the following chart:



• **Power Saving**
$$= (750 \times 4 + 850 \times 2.5) / ((6 \times (125 + 750 + 850)))$$

$$= 5125 / 10350 = 0.495 \text{ or } 50\%$$

• **Average Power consumption** : $5125 \text{ mW} / 6 = 854 \text{ mW}$

As a result, 50% system power was saved. The PARIS PASTA node in a 5 sec duty cycle performed continuous tracking for 135 minutes on a 9V alkaline battery, and 15 hrs 40 minutes on six AA alkaline batteries.

Summary of Power Consumption

Setup	Power Consumption	9V Battery	6 AA Batteries
PARIS Config 4	2.5 W	NA	NA
PARIS PASTA node 1 sec duty cycle	1.7 W	70 minutes	9 hours
PARIS PASTA node 5sec duty cycle	0.8 W	135 minutes	15 hours 40 minutes

Appendix A: Power as Function of Configuration

Table 1: System Configuration 4 Power

Component	Voltage (VDC)	Current (mA)	Power (mW)	Comments
Switching Power Supply	12	4.4	52.8	Vout 3.3 V with a 15 mV ripple
Micro Controller, C8051F124, 24.5 MHz	3.3	31.2	103	@ 24.5 MHz, C8051 runs Power Aware OS and except commands from PC HyperTerminal to schedule power to Camera and Orsys boards
Micro Controller, C8051F124, under Reset	3.3	29	95.7	system clock run at 3 MHz
RS-232 Transceiver	3.3	5.4	17.8	Changed the power for the RS232 device is not significant whether in transmit or standby mode.
CPU FET	3.3	0.3	1	Notice the differences between two FET channels in the same package.
DSP FET	3.3	0.7	2.3	Notice the differences between two FET channels in the same package.
CAM LED	3.3	0.4	1.3	Can be turned off by jumper option.
DSP LED	3.3	0.4	1.3	Can be turned off by jumper option.
PWR LED	3.3	0.4	1.3	Can be turned off by jumper option.
Camera/Processor	3.3	353	1164.9	aSi Camera with the reformatter only needs 3.5 v source to operate and send video image to a monitor.
Camera/Processor/ shutter on	3.3	555	1831.5	aSi Camera consumed an addition of 200 mA or 700 mW when shutter on.
Orsys C6713	3.3	497	1640.1	The Orsys C6713 was turn on and blink its LED
Controller+Camera/ Processor+Orsys	3.3	768	2534.4	The Paris Controller board supplied power to both the Camera/ Reformatter and the Orsys boards

Table 2: Enhanced PARIS Power Analysis

Component	Active (mA)	Sleep (mW)	Full On (mW)	Sleep (%)	Active (%)	Dynamic (mW)	Comments
Switching Power Supply	4.4	0.33	14.52	0.5	0.5	7.26	3.3 V, 1.26 V for DSP core
Micro Controller, C8051F124, 100 MHz	31	0	102.3	0.1	0.9	92.24	@ 50 MHz, C8051 can quickly transfer video frame to wireless radio.
LVDS Receiver	10	0	33	0.5	0.5	16.5	power for the LVDS device is not significant whether in transmit or standby mode.
Power FET Switch	0.3	0.03	0.99	0.9	0.1	16.5	FET use for supplying power
Signal FET Switch	7	0	23.1	0.3	0.7	16.27	FET used for signal isolation.
Flash	7	0	23.1	0.9	0.1	0	512 K Bytes
SDRAM	100	0.03	100	0.5	0.5	165.02	32 Mega Bytes
LEDs	0.8	0	0	0	0	0	Power indicator and debug LEDs. Not enable for deliver module
aSi Sensor/LVDS Interface	325	231	1072.5	0.6	0.4	567.6	ASi Camera with the aSi LVDS Interface board needs 3.3 v source to operate
TMS320C6713 I/O	65	132	214.5	0.5	0.5	173.25	3.3 Volt
TMS320C6713 Core	690	113.4	869.4	0.5	0.5	491.4	1.26 Volt
PASTA IO board	37		125	0	1	125	The IO board is always on
Total	1240.5	476.8	2453.41	n/a	n/a	1671.03	Perform continuous tracking 3 scans/sec

Appendix B: Enhanced PARIS PASTA Node

Figure 1. The Enhance PARIS PASTA Node



Figure 2. The Enhance PARIS PASTA Node with Mote Radio and Sensor Board

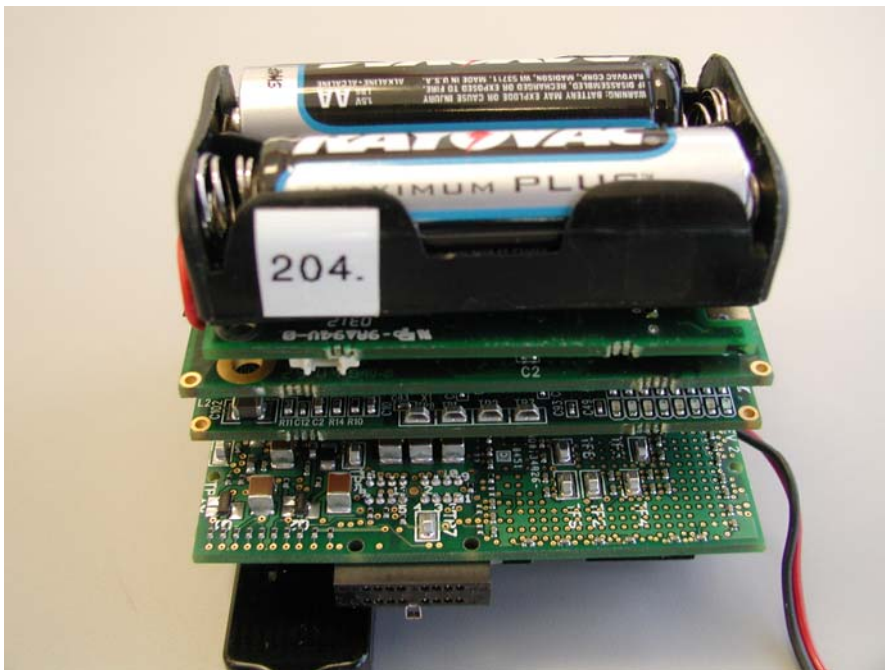


Table 1. Power as function of voltage as measured at PASTA IO board

Current measured at 3.3V supplied to the PARIS board (255MHz) and Camera – Full on

Voltage Source (Volt)	3.3
C8051 LED heart beat (mA)	20
+ DSP On (mA)	50
+ Camera On (mA)	240
Taking Picture (mA)	380
Tracking (mA)	400
Tracking Power (mW)	1320

Current measured at the PASTA IO board which supplied 3.3V to the PARIS board (255MHz) and camera

Voltage Source (Volt)	4.5	6	7.5	9	12
C8051 LED heart beat (mA)	22	17	15	13	11
+ DSP On (mA)	48	40	33	29	25
+ Camera On (mA)	215	170	144	126	106
Taking Picture (mA)	340	266	225	205	164
Tracking (mA)	360	273	243	206	177
Tracking Power (mW)	1620	1638	1822.5	1854	2124

Appendix C: PARIS Requirements Specification & Technology Projections

Table of Contents

1. OVERVIEW	14
1.1. SUMMARY	14
1.2. APPLICATION OVERVIEW	14
1.3 TRANSITION PLANS	14
2. PARIS NODE DESCRIPTION	15
2.1 ELEMENTS OF A PARIS NODE	15
2.1.1 <i>PARIS Node Structure</i>	16
2.2 PARIS HYPOTHESES TO BE TESTED	17
2.2.1 <i>Human Detection/Tracking Problem</i>	17
2.2.2 <i>Paris Node Specification/Requirements</i>	18
2.2.3 <i>Human Detection/Recognition and Tracking Challenges</i>	18
2.3 PARIS PERFORMANCE GOALS	19
2.4 DEVELOPMENT APPROACH.....	20
2.4.1 <i>Bounding – Path to “Trade Space”</i>	20
2.4.2 <i>Path Through “Trade Space” in Time</i>	22
3. TECHNOLOGY OVERVIEW / PROJECTIONS	23
3.1 UNCOOLED IR SENSOR PERFORMANCE TRENDS	24
3.2 CCD/IMAGE INTENSIFIER’S	24
3.3 <i>Non-Imaging Sensor Interface</i>	24
3.4 ADC RESOLUTION/SAMPLING FREQUENCY TRADE SPACE.....	25
3.5 INDUSTRY TREND OF HIGH POWER COMPUTING	26
3.6 INDUSTRY TREND IN LOW POWER COMPUTING	27
3.7 THE INCREASING IMPORTANCE OF STATIC POWER CONSUMPTION.....	28
3.8 INCREASING POWER EFFICIENCY OF PROCESSORS	30
3.9 CONFIGURABLE LOGIC	31
3.9.1 <i>FPGAs</i>	31
3.9.2 <i>CPLDs</i>	32
3.9.3 <i>CPLDs vs. FPGAs</i>	32
3.10 CONSERVING ENERGY	33
3.10.1 <i>Power vs. Latency</i>	34
3.10.2 <i>Energy Savings Contribution vs. Time</i>	35
3.11 MEMORY CONSIDERATIONS	35
3.12 ALGORITHM DEFINITION AND POWER AWARE CONCEPTS.....	36
3.12.1 <i>Sensor Engine Clock Diagram</i>	36
3.12.2 <i>Baseline ATD/MTI Function</i>	38
3.12.3. <i>Overview of the Detection Paradigm</i>	38
3.12.4 <i>Potential Algorithm Processing Techniques to Reduce Power</i>	39
4. REFERENCES	40

Table of Figures and Tables

FIGURE 1 - NODES COVERING BATTLE-SPACE	15
FIGURE 2 - PARIS NODE BLOCK DIAGRAM	16
FIGURE 3 - PARIS NODE STRUCTURE	17
FIGURE 4 - PARIS DEVELOPMENT APPROACH	21
FIGURE 5 - BOUNDING - PATH TO "TRADE SPACE"	22
FIGURE 6 - PATH THROUGH "TRADE SPACE"	22
FIGURE 7 - PARIS NODE - TECHNOLOGY PROJECTIONS	23
FIGURE 8 - A/D - POWER CONSUMPTION VS. SAMPLING RATE	25
FIGURE 9 - A/D POWER CONSUMPTION VS. RESOLUTION	25
FIGURE 10 - INDUSTRY TREND OF HIGH POWER COMPUTING	26
FIGURE 11 - INDUSTRY TREND IN LOWER POWER COMPUTING	27
FIGURE 12 - POWER CONSUMPTION, STATIC AND DYNAMIC	28
FIGURE 13 - POTENTIAL SEMICONDUCTOR SOLUTIONS	30
FIGURE 14 - mW/MIPS VS. TIME	31
FIGURE 15 - LATENCY VS. TECHNOLOGY IN TIME	34
FIGURE 16 - SA-1110 POWER VS. FREQUENCY, SA-1110 ENERGY VS. FREQUENCY	34
FIGURE 17 - ENERGY IMPROVEMENTS VS. TECHNOLOGY IN TIME	35
FIGURE 18 - SENSOR ENGINE BLOCK DIAGRAM	37
FIGURE 19 - BASELINE ATD/MTI ALGORITHM	37
FIGURE 20 - DETECTION ALGORITHM	38
FIGURE 21 - RECEIVER OPERATING CHARACTERISTIC CURVE	39
TABLE 1 - PARIS PERFORMANCE GOALS	19
TABLE 2 - FPGAs vs. CPLDs	32
TABLE 3 - CPLD vs. FPGA SPEED EXAMPLE	33
TABLE 4 - CPLD vs. FPGA EXAMPLE	33

1. Overview

APPENDIX C describes the PARIS node and application and makes projections on the technology that can be exploited in the design of the heterogeneous processing node.

1.1. Summary

Twenty years ago, Gordon Moore predicted a growth in chip complexity of approximately 2x every year. Industry has been able to match the pace of this law by increasing die size, shrinking feature size (scaling) and increasing circuit cleverness. These rapid advances in semiconductor technology which track Moore's law will encounter substantial development challenges in the 2006 time frame as feature sizes approach 0.06 microns. Leakage current will drive static power and growth until fundamental design changes are made to CMOS transistor designs. Unfortunately, many of these increasing performance features in semiconductors are at the expense of power. Therefore, it is projected that the largest gains in power aware system design will be the result of new system architectures and algorithm designs that exploit power aware computing features.

1.2. Application Overview

A sensor network, as shown in Figure 1, is characterized as a set of locally powered sensors linked together via wireless communication with the objective to accurately assess activity within the sensor network field of regard and forward that assessment to command and control elements for appropriate action. Consequently, the sensor network is a combination of sensing, processing and communication. To minimize power utilization within that network, each power consuming element must be evaluated for effectiveness to the overall network objective. Such considerations in the past, for example, have led to sensor nodes that both sense and process locally since the communication function required to forward and distribute raw node data can take up to 40 times the power consumed by node computation. Also, each sensor node may be composed of multiple sensors. Our focus for this research effort is to assess the overall sensor network power optimization in a realistic unattended sensor system framework based upon the exploitation of orthogonal sensing elements, environmental situation characterization, and invocation of adaptive processing algorithms in a power aware computing and communicating structure.

1.3 Transition Plans

Reliable Power Aware Human Detection / Tracking is a critical need for:

- Homeland Security: Border / Facility Protection
- Army / Special Operations: Perimeter Security, Remote Location Monitoring, e.g. Caves
- Manportable Fire Control Systems, e.g. OICW, OCSW

Raytheon is active in each of these areas. Homeland Security is major company thrust. Raytheon produces fire control for OCSW and is a world leader in man-portable IR systems.

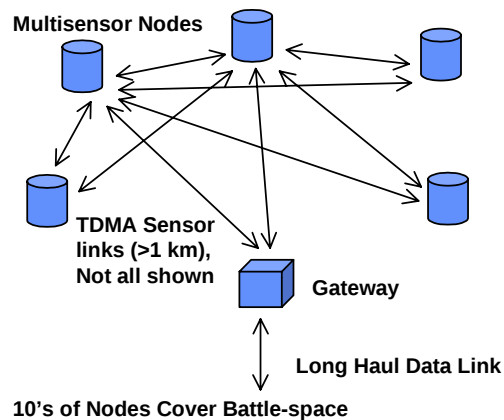


Figure 1 - Nodes Covering Battle-Space

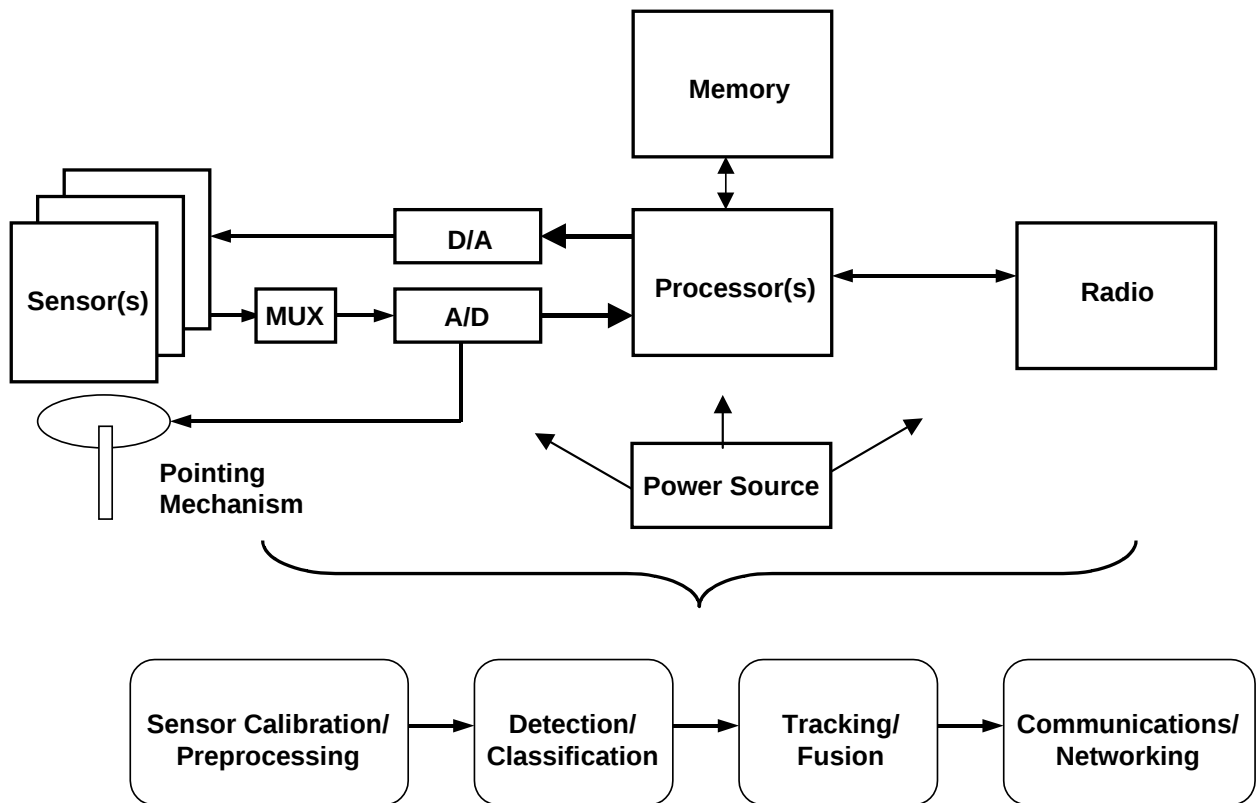
As capabilities are developed, Raytheon plans to demonstrate PARIS capabilities to interested parties both within / external to Raytheon.

2. PARIS Node Description

2.1 Elements of a PARIS Node

The HW elements of a PARIS Node include Processing capability, memory, a radio, sensors, sensor I/Fs and a power source. Processing SW elements include Sensor Calibration Preprocessing, Detection/Classification Processing, Tracking/Fusion Processing and Communications/Networking Processing. These are shown of Figure 1.

The key to PARIS Power Awareness is addressing all the components processing and algorithms, including both HW and SW, and how the system functions for particular missions in the field.

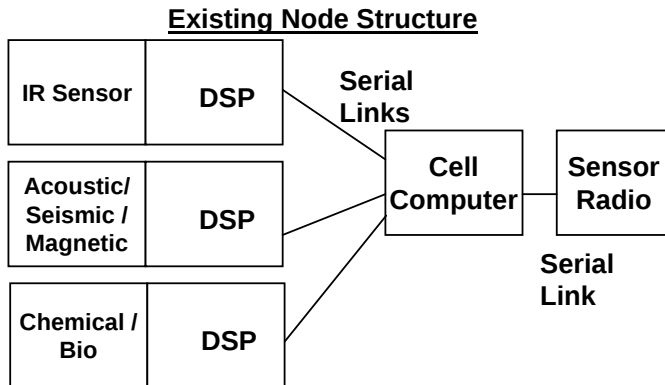


- **Power Awareness Design Must Address All Elements of the System: Sensors, Data Acquisition, Processing (Processors, Memory, Interconnections, Algorithms), Communications / Networking**

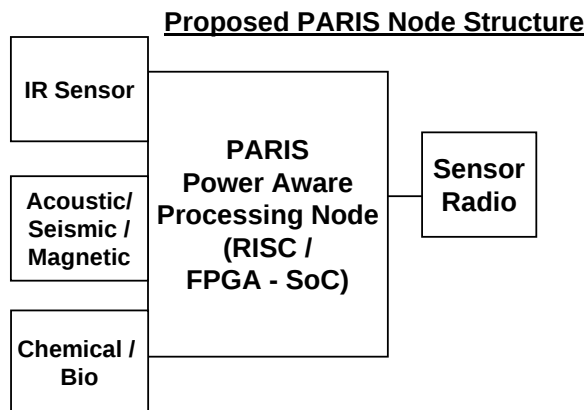
Figure 2 - Paris Node Block Diagram

2.1.1 PARIS Node Structure

The objective of the PARIS program will be to reduce the power dissipation by combining the signal and processing for both sensors and the control processing into a single heterogeneous processing unit composed of a power aware RISC processor with a tightly coupled adaptive computing element such as an FPGA shown in Figure 2. Figure 2 shows both the existing PARIS Node Structure and the proposed PARIS Node Structure, including the different sensor types.



- Sensor processing limited by DSP
- Cell computer performs Decentralized Data Fusion across all nodes & interface control between sensors & radios including TDMA processing



- Sensor processing is handled by scaleable, reconfigurable architecture with power aware features
- Hardware is time shared between sensors driven by mission tasking
- Node can be remote or self cued

Figure 3 - PARIS Node Structure

2.2 PARIS Hypotheses to be Tested

- Power Aware Computing / Communication's Design must Balance Energy Usage Across All Elements of System
- Robust Application / Mission Performance Requires Multiple Complementary Sensors
- Integrated Node vs. Distributed Nodes (Tripwire nodes Cue Detection / Tracking Nodes)
- Impact on System Performance & Energy Usage
- Use of a Shared Heterogeneous Processing Resource may be More Effective Approach to Meeting Mission Needs

2.2.1 Human Detection/Tracking Problem

- Problem: Detect, Classify, & Track Multiple Human Targets Through Combination of Un-attended Sensors.
- Improve System Performance While Reducing Energy Usage to Increase Mission Duration

2.2.2 Paris Node Specification/Requirements

- Specification Summary:
 - Number of Targets: Up to 3
 - Range: Up to 100m from Boundary
 - Motion Classes: Walking / Running / Start-Stop
 - Energy Usage: Average 1 Watt per node (10x Reduction Over Raytheon TB)
- Approach:
 - Integrated Node: Multi-sensors Used to Detect & Track Human Targets
 - Imaging Based Approach (Video / IR)
 - Non-Imaging Sensor Cued Imaging Node
 - Separate Trip-wire Like Used to Wake Up Imaging Node
 - Exploit Pico-node from PASTA effort
 - Evaluate Performance Over Range of Environmental Conditions

2.2.3 Human Detection/Recognition and Tracking Challenges

- System Level Trades / Performance Evaluation will be Explored on PARIS
- Mis-match of Non-Line of Sight (LOS) vs. LOS Sensors for Human Detection
- Non-LOS Sensors have short range performance
- LOS Sensors can have significantly greater range but have limited FOV
- Non-LOS Typical Performance:
 - Magnetic: 5 m typical, HE3 type: 25m
 - Seismic: 50 m
 - Electric Field Disturbance: 5 m
 - Acoustic: TBD (Not Clear if at all Effective)
 - RF: 50-100m dependant on Power Level
 - Chemical Point Detection: 2-5 m
 - Tripwires: 5 m
- LOS Sensors Range Determined by Local Terrain, Field of View (FOV), Sensor Resolution, Local Environment, e.g. Weather, Time of Day
- Video (Color / Grayscale)
- IR Sensor
- Image Intensifier
- Human Targets Don't Have Unique Signatures & Require Many Pixels on Target
- Self-Contained Multi-sensor Node vs. "Tripwire" like Cueing Sensor Node
- Combined Detection/Classification/Tracking Node

2.3 PARIS Performance Goals

Table 1 addresses the different system aspects in terms of how they will evolve throughout the PARIS program.

	Month 18	Month 24	Month 30
Platform	Emulation Testbed composed of hardware from USC-ISI & additional hardware simulating elements of PARIS node design	PARIS tightly coupled heterogeneous processing node	PARIS tightly coupled heterogeneous processing node
Algorithms	Enhanced Algorithms (40x more computationally intensive)	Initial mapping of algorithms to PARIS node	Power Aware processing framework; revised mapping of algorithms on PARIS node
Algo Performance relative to Baseline	10% improvement in Pd & 25% reduction in False Alarms over baseline algorithms	Same as Month 18	10% improvement in Pd & 50% reduction in False Alarms for enhanced algorithms
Power Reduction	2x (composite result based on use of more computationally intensive algorithms & use of power aware hardware)	5x (enhanced result based on execution on PARIS node)	10x (composite result based on efficient mapping of algorithms to final PARIS node (more power efficient) & use of power aware processing framework)
Performance Comparison	Energy Savings Relative to Baseline Algorithms - 80x	Energy Savings Relative to Baseline Algorithms - 200x	Energy Savings Relative to Baseline Algorithms - 400x
Mission Duration	2x	5x	10x

Table 1 - PARIS Performance Goals

2.4 Development Approach

- Leverages Phase 1 Results, Tools, & University of Southern California Information Sciences Institute Power Aware Sensing Tracking and Analysis (USC-ISI PASTA) program HW/SW, Raytheon Sensor Testbed
- Exploits Joint Data Collections & Final Field Demo with USC-ISI PASTA Effort
- Enhanced MILAN Tools will be used to Support Architecture Trades / Design

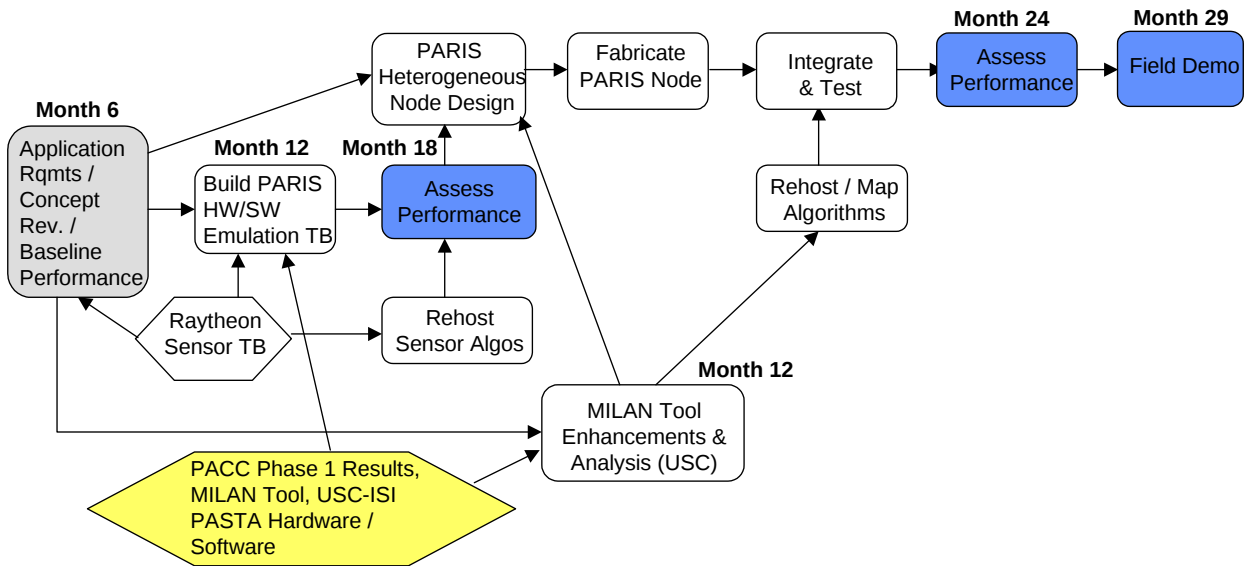


Figure 4 - PARIS Development Approach

2.4.1 Bounding – Path to “Trade Space”

Figure 5 shows how Trade Space is bounded. The path into “Trade Space” starts with defining the system requirements for the PARIS Node. This takes into consideration the system mission objectives, mission profiles and mission configurations. After these are identified, then other processing requirements and algorithm requirements can be identified. From these, the math intensive processing can then be estimated in terms of MIPS. Also, the availability of hardware from industry, and the predicted hardware availability from industry, can be used to select hardware components. These components can be characterized in terms of power and latency. Also, the sensors that are candidates for PARIS can be selected and characterized. Once entered into trade space, different parameters of each component can be traded against mission performance.

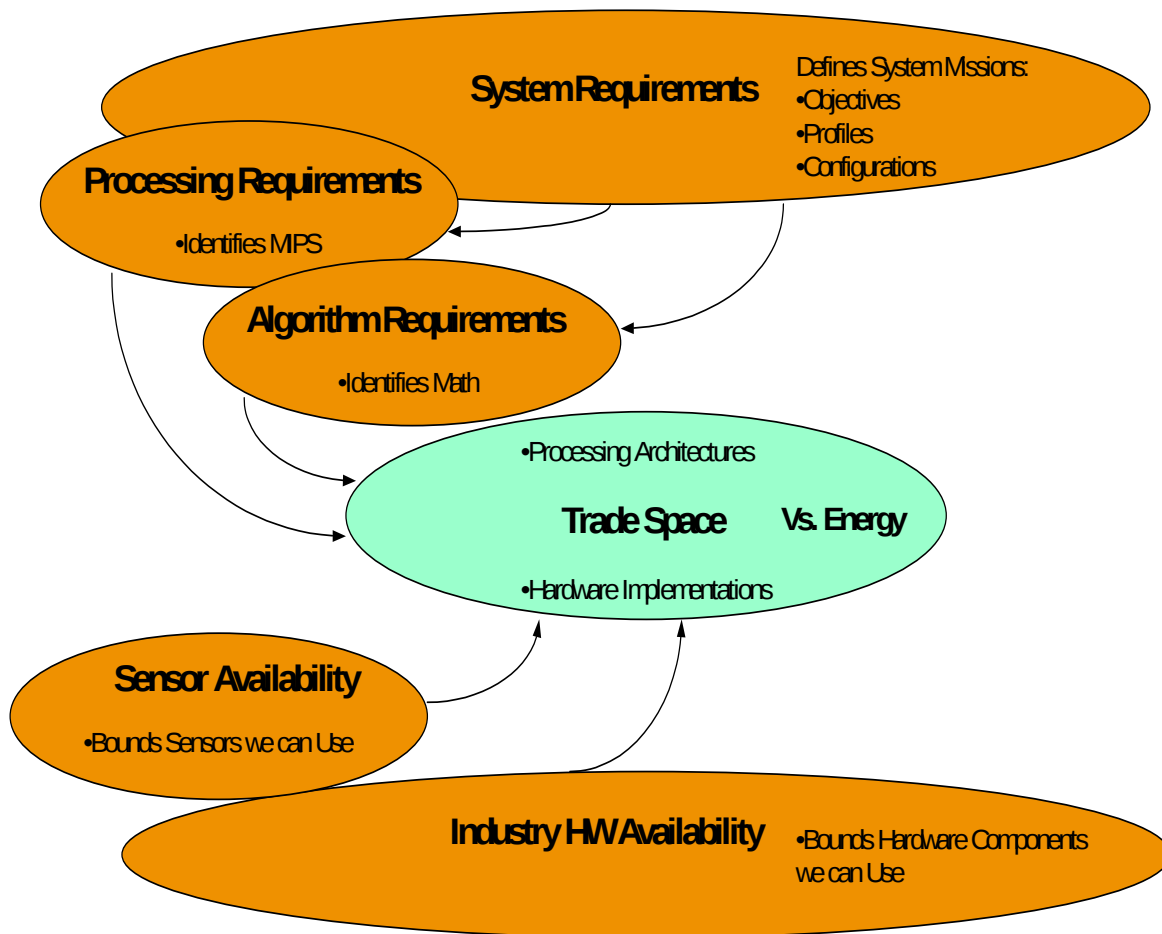


Figure 5 - Bounding - Path to "Trade Space"

2.4.2 Path Through "Trade Space" in Time

The key to the PARIS trade space, is monitoring the progression through trade space in time. At the outset, a Sensor Test Bed measurement for power will be taken. Then, a baseline Node concept design will be developed. Power estimation for the emulation testbed design will also be noted. As movement is made through trade space, continual monitoring of the system components and their energy contributions will be charted. This is shown in figure 5.

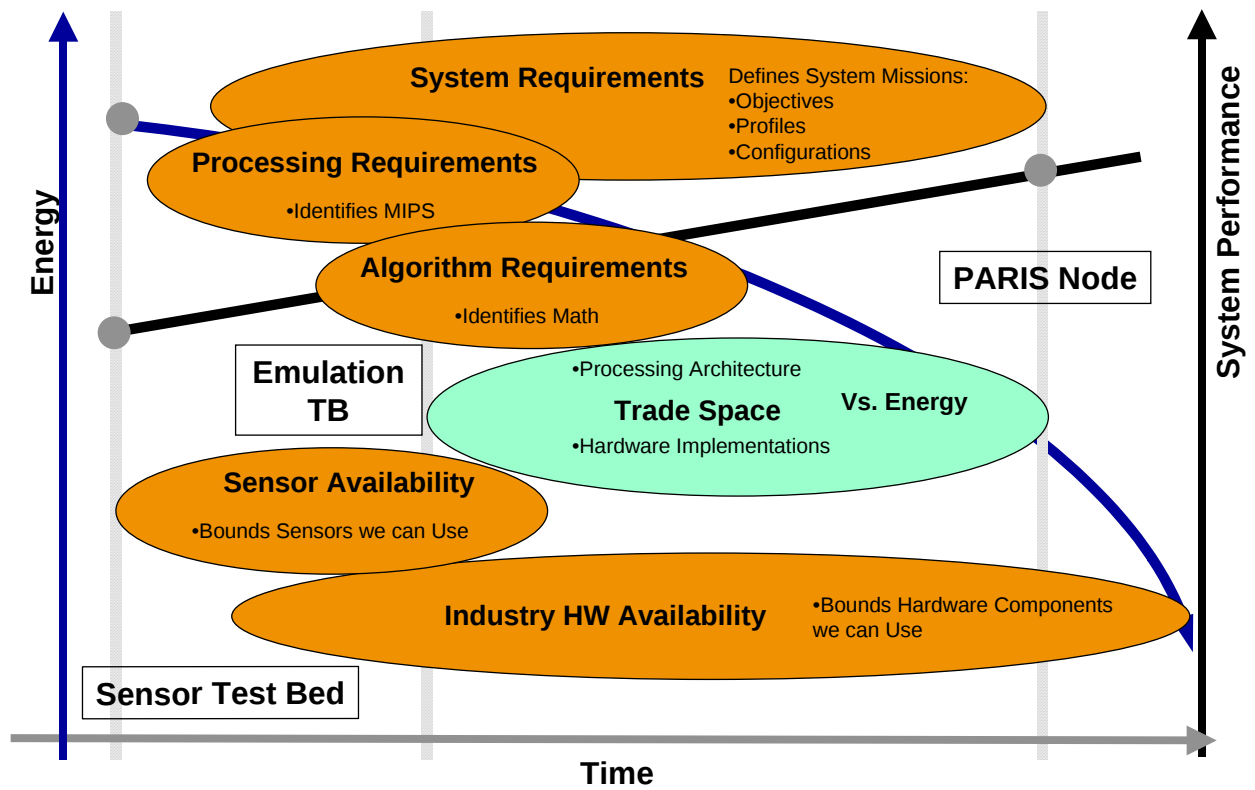


Figure 6 - Path Through "Trade Space"

3. Technology Overview / Projections

The following figure shows the PARIS Node Structure, with blue shaded areas, which indicate the major functions to be exploited in terms of energy savings under mission algorithm control.

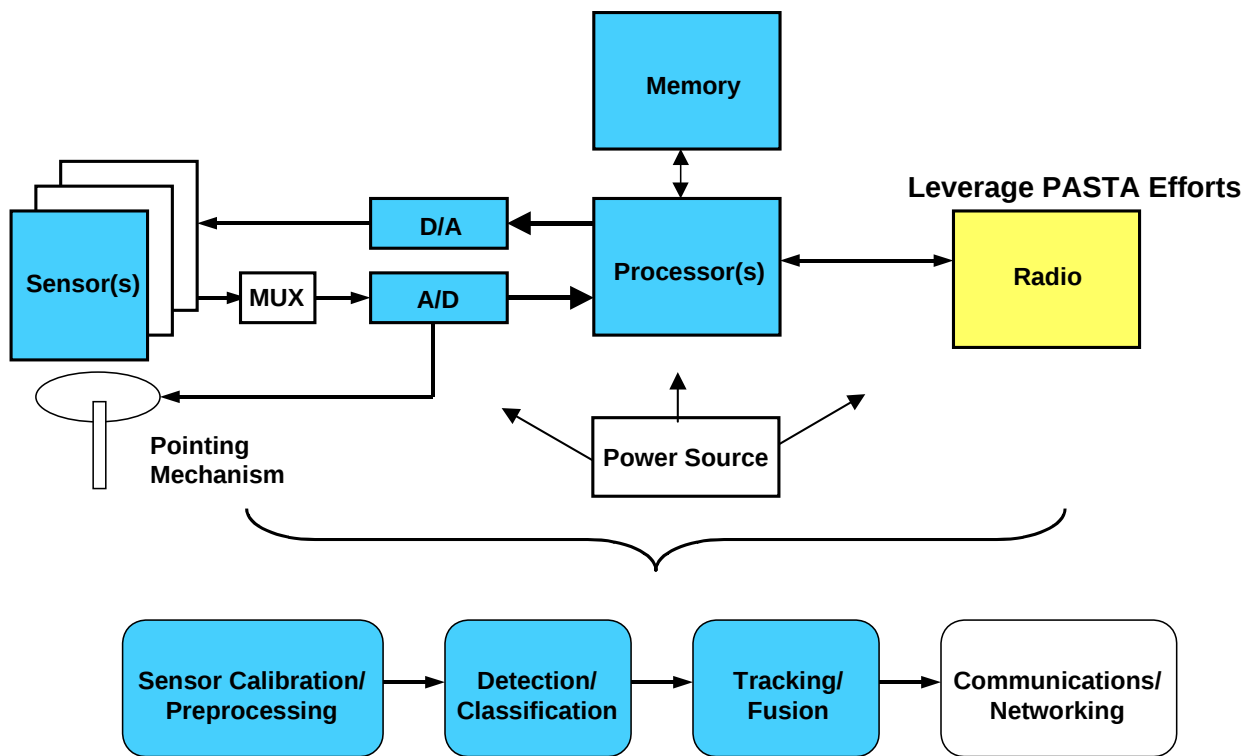


Figure 7 - PARIS Node - Technology Projections

3.1 Uncooled IR Sensor Performance Trends

Major power reduction approaches include: Reduction of FPA Capacitance & Integration Time and Minimizing / Eliminating Use of TEC. These approaches can impact sensitivity / image quality and requires more complex NUC which may require more operations and take multiple frames to converge.

Higher Resolution, Longer Range Performance:

Area Size: 160 x 120 -> 320 x 240 -> 640 x 480
Pixel Size: 50 micron -> 25 micron -> 20 micron

Higher Sensitivity in Degraded Conditions:

Turn-on Time: 10 seconds -> 1.5 seconds
Frame Rate: 20 Hz to 60 Hz
Sensitivity: 100 mk NEDt -> 25 mk

Reduced Power:

Thermal Stability: Thermal Electric Cooler -> Stabilized -> Athermal Operation

Additional Target Discriminants:

Multispectral: Long Wave -> Multi-Band

3.2 CCD/Image Intensifier's

- Rapid Advances in Size / Sensitivities of Charge Coupled Device (CCDs)/CMOS imagers have been driven by Camcorder & Digital Photography
- CCDs which have higher sensitivity require approximately 100 times more power than CMOS imagers.
- Major limitations is support for high frame rates for large arrays
- High Data Rates Needs Incompatible with Current Pixel Integration Times
- Small CMOS Arrays, e.g. 320 x 240, can be very power efficient, e.g. 25 mw active, 1 mw standby
- Coupling of CCDs with Image Intensifiers for Night Time Operations Requires a Minimum of 2-3 Watts of Power and Typically have a 50% loss in Resolution

3.3 Non-Imaging Sensor Interface

Typical non-imaging sensor interfaces:

- 1) Acoustic Sensor I/F, 1 kHz maximum data rate, 12 bits parallel
- 2) Magnetic Sensor I/F, 50 Hz maximum data rate, 20 bits parallel
- 3) Seismic Sensor I/F, 400 Hz maximum data rate, 12 bits parallel

3.4 ADC Resolution/Sampling Frequency Trade Space

One important trade space that PARIS can exploit is the trades between Analog to Digital (ADC) sensor sampling rate and power consumption.

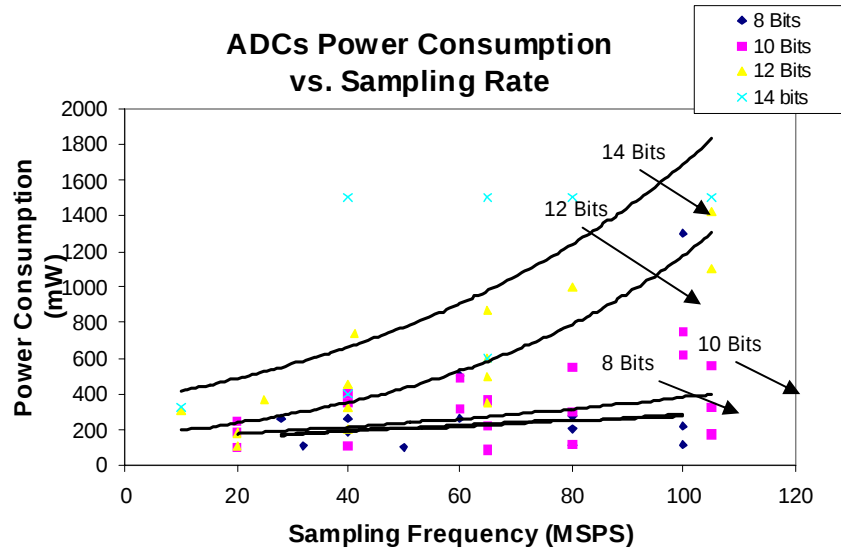


Figure 8 - A/D - Power Consumption vs. Sampling Rate

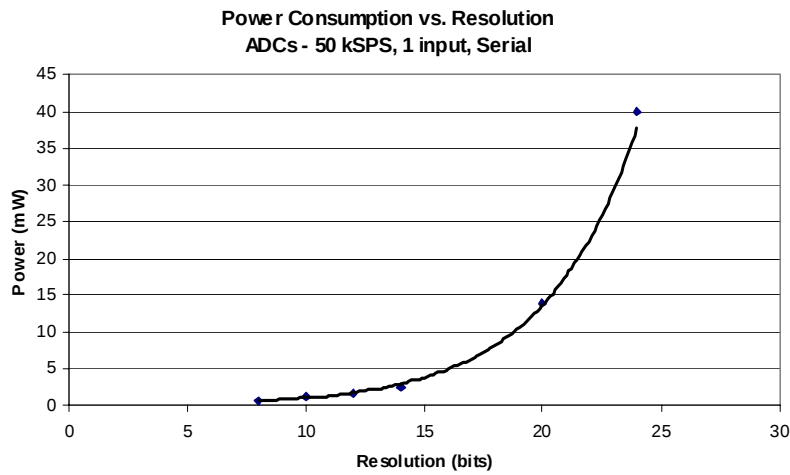


Figure 9 - A/D Power Consumption vs. Resolution

The front-end power consumed by the analog to digital converters (ADCs) vary exponentially with respect to resolution and sampling frequency.

Knowing the resolution / sampling frequency trade space allows the system to intelligently tradeoff power consumption and mission performance. Minimizing the sampling rate and number of bits per sample can significantly reduce power consumption

3.5 Industry Trend of High Power Computing

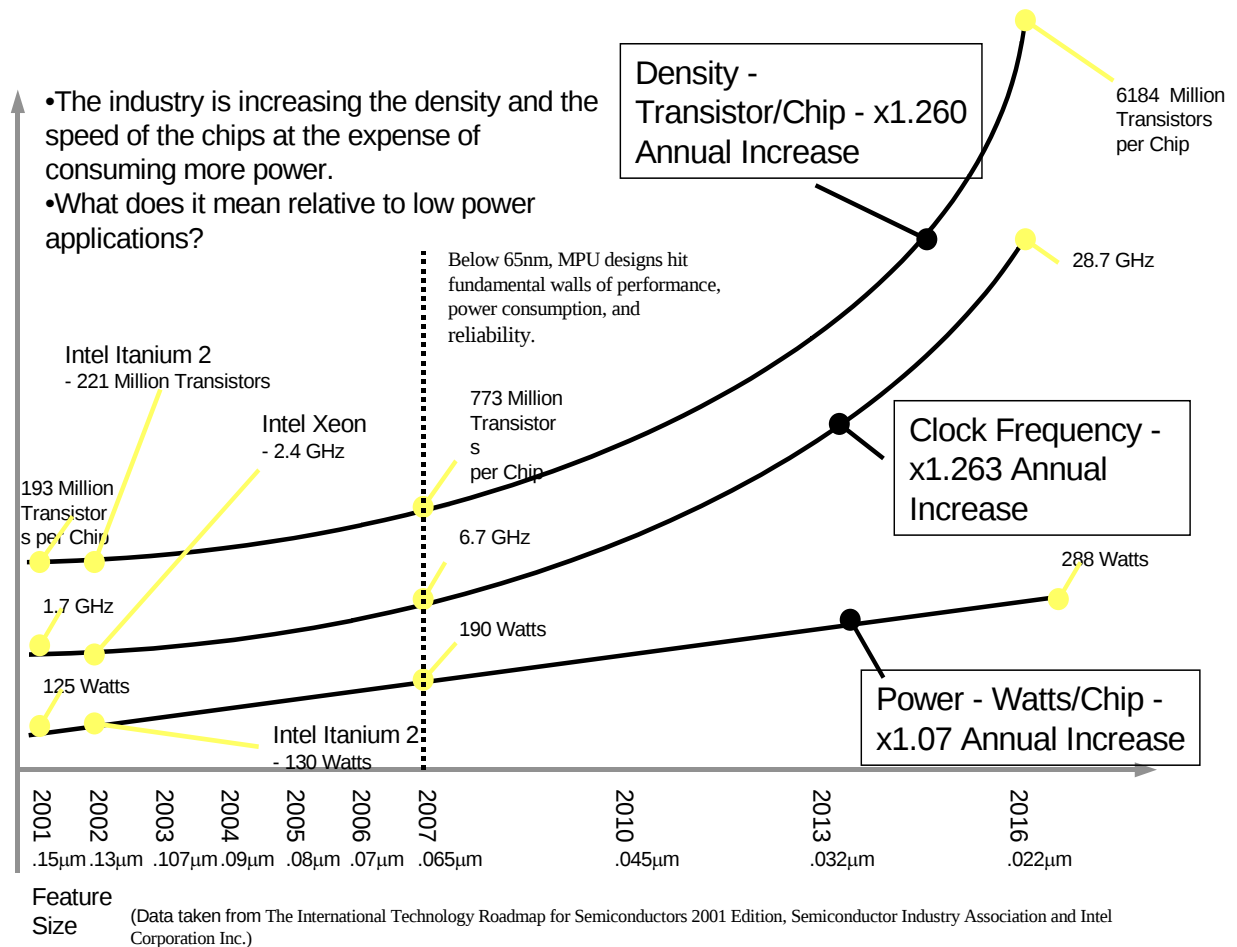


Figure 10 - Industry Trend of High Power Computing

Gordon Moore made a prediction that the number of components on a single chip would double every 18 months. This exponential growth has become known as Moore's law and has held relatively true over the past 40 years. The general trend in high power computing is to pack more transistors on a chip, run the chip faster, and to consume more power.

We expect to see an annual increase of 1.26x in terms of chip density and 1.263x annual increase in clock frequency. Along with density and speed, power will also increase every single year. It's projected that by the year 2016, high performance microprocessors will consume close to 300 Watts.

With a high degree of confidence, it is expected that these trends will hold until 2007. As transistor technology scales below 65 nm, semiconductor technology will hit fundamental walls of performance, power consumption, and reliability. For digital applications, these challenges include exponentially increasing leakage currents, short channel effects controlling threshold voltage, continuing to increase I_{on} and control of V_t over the die.

For analog / RF applications, the challenges additionally include sustaining linearity, low noise figure, power-added efficiency, and transistor matching. Though it is uncertain how the semiconductor industry will continue experiencing the exponential growth predicted by Moore's law, research and development will hopefully overcome these fundamental barriers.

3.6 Industry Trend in Low Power Computing

Year of Production	2001	2004	2007	2010	2013	2016
Process Technology (nm)	130	90	65	45	32	22
Supply Voltage (V)	1.2	1	0.8	0.6	0.5	0.4
Clock Frequency (MHz)	150	300	450	600	900	1200
Application (Maximum Required Performance)	- Still Image Processing - Web Browser - E-Mail - Scheduler	Real Time Video Codec (MPEG4 / CIF)		Real Time Interpretation		?
Application (Other)		TV Telephone Voice Recognition Authentication / Cryptography				
Processing Performance (GOPS)	0.3	2	15	103	720	5042
Required Average Power (mW)	100					
Required Standby Power (mW)	2.1					
Battery Capacity (Wh/Kg)	120	200		400		
Dynamic Power Management Gap (X)*	0.06	0.59	1.03	2.04	6.43	23.34
Static Power Management Gap (X)*	0.85	5.25	14.55	30.18	148.76	828.71

Figure 21 - Industry Trend in Lower Power Computing

*Power Management gap - the factor improvement in power management that must be achieved jointly at the levels of application, operating system, architecture and IC design.

PARIS nodes require microprocessors that are power efficient, not high performance processors that consume too much power for our application. This table reflects a study done by the Japan Semiconductor Technology Roadmap Working Group 1 and was originally introduced in the 2000 International Technology Roadmap for Semiconductors

(ITRS) update. It sets the requirements for various attributes of a low-power, consumer-driven, handheld wireless device with multimedia processing capabilities.

This study investigated the increase in processing speed and performance if we keep average power consumption constant. It's important to note that the required power reduction factors exceed 20x for dynamic power and 800x for standby power. Not only is power consumption a problem, but static power consumption is emerging as one of the top concerns for high performance circuit design.

3.7 The Increasing Importance of Static Power Consumption

Driven by the recent popularity of portable devices, the industry and research community have started to turn its attention to power consumption. Power consumption has become a particularly important issue for chips that need relatively high performance but that operate on batteries, requiring the minimum in both active and standby power consumption. Prime examples are TI's digital signal processors, used widely in what has become the largest power-aware consumer market: cell phone baseband processors.

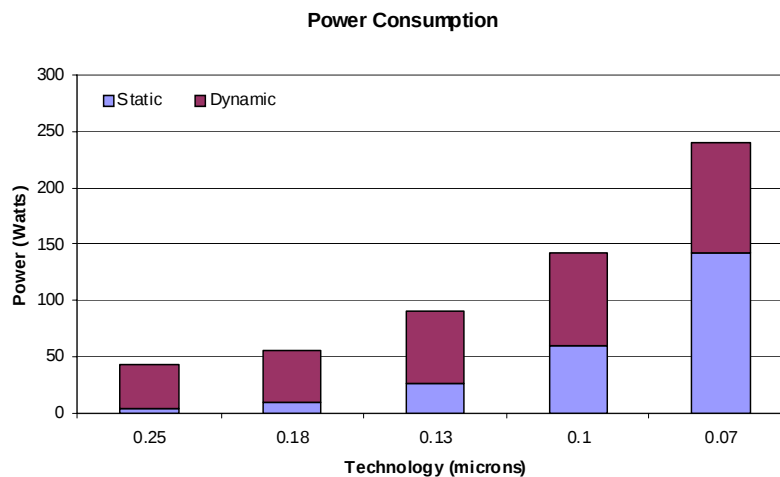


Figure 32 - Power Consumption, Static and Dynamic

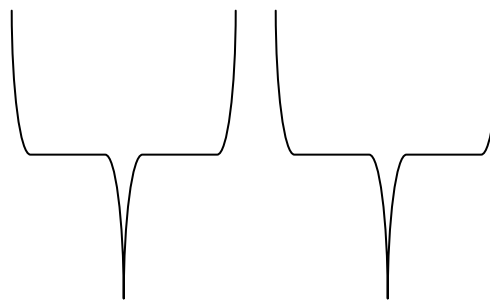
Source: Intel Corporation

From figure 12, we see that total power consumption is increasing for each process generation. It is also clear that static power consumption is accounting for a larger percentage of the total power consumption as the semiconductor technology shrinks.

As transistors have gotten faster, feature sizes have decreased and operating voltages have decreased resulting in an exponential increase in leakage current. Static power consumption is growing faster than dynamic power consumption. Unfortunately there is no quick fix on the horizon to quell the rise in both active power and dynamic power consumption. Leakage current, a phenomenon that causes transistors to consume power during their off state, has emerged as one of the top concerns for circuit design, particularly at the 0.13-mircron process node. Leakage current can rise by a factor of two to three for each process generation. Since there is no silver bullet when it comes to power consumption, the industry is attacking the problem with a series of 10 percent improvements. Intel, IBM, and other chip makers have come up with a flurry of

proposals suggesting several lines of attack on the power problem. Figure 13 highlights potential memory and logic solutions that are being implemented or are currently being researched to control short-channel effects and to limit power consumption.

Logic Power Consumption

$$\alpha C V_{dd}^2 f + I_{off} V_{dd}$$


Dynamic Power Consumption

Static Power Consumption

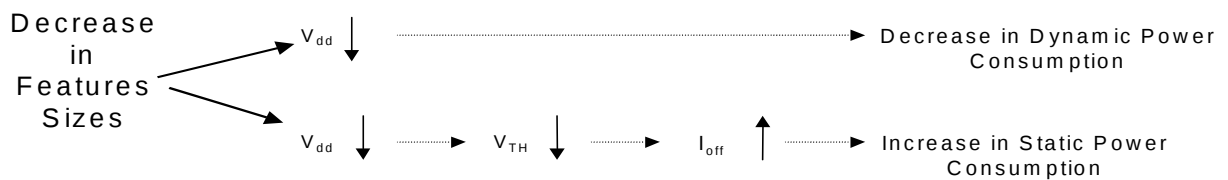
α - Technology Factor

C - Capacitance

V_{dd} - Supply / Operating Voltage

f - Clock Frequency

I_{off} - NMOSFET drain current at room temperature (NMOS sub-threshold, gate, and junction leakage current components).



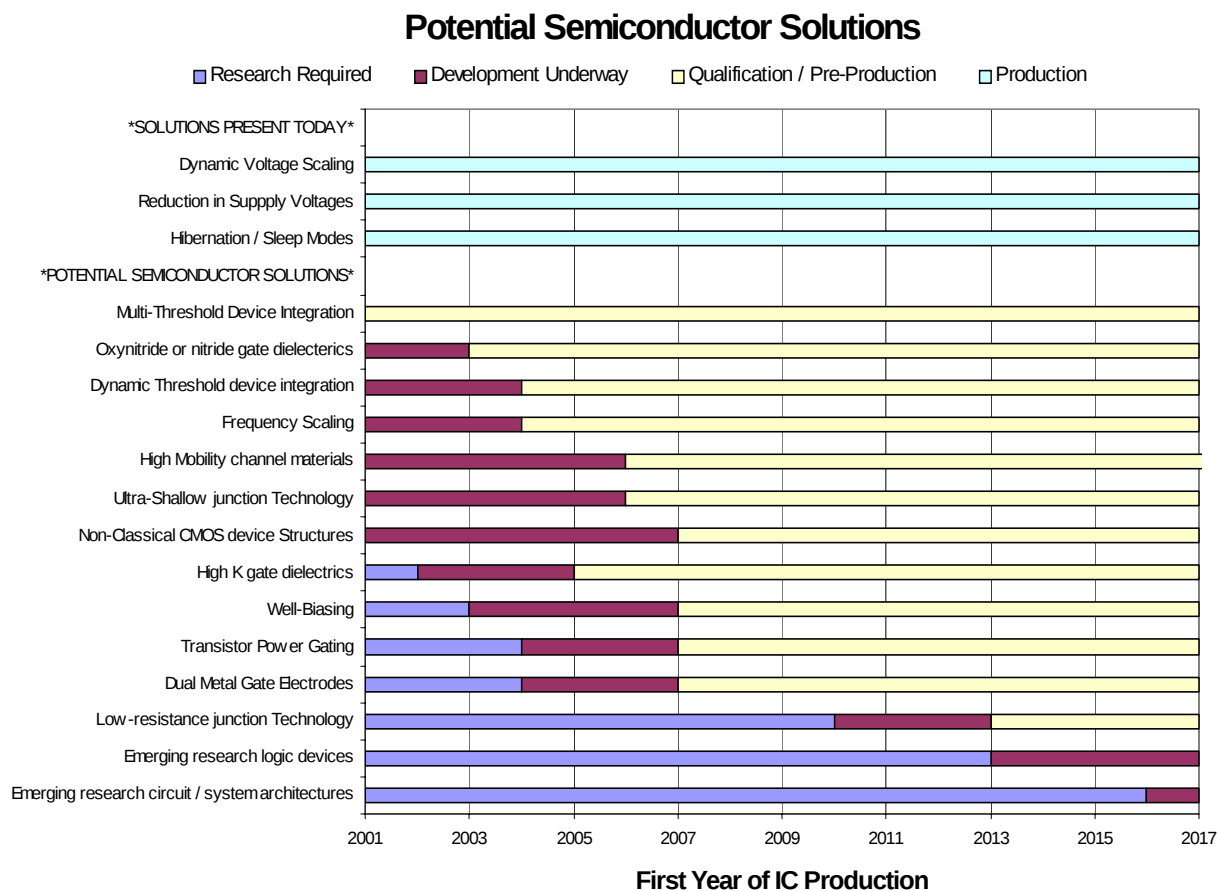


Figure 4 – Potential Semiconductor Solutions

Source: The International Technology Roadmap for Semiconductors 2001 Edition

3.8 Increasing Power Efficiency of processors

The industry is increasing the density and speed of chips at the expense of consuming more power. What does this mean for low power applications? A novel way of looking at the problem is to observe how power efficient processors are. Even though power consumption is increasing, it takes a lot less time to do the same amount of work.

Historically, the amount of power to perform one MIPS has decreased annually by about 0.658x. PARIS can exploit the increasing power efficiency of processors by keeping it on only long enough to get the job done.

Another observation that can be made from Figure 14 is static power consumption due to transistor leakage current constitutes an increasing fraction of the total power in modern semi-conductor technologies. As static power dissipation increases, power issues must be addressed at the semiconductor, system, and the algorithm level to stay on this curve. If we do not control the static power consumption, it is likely that we will not see the improvements in power efficiency anymore.

Note that we chose to characterize the power efficiency of low power embedded processors because they have a low operating power and their power efficiency is clearly defined (mW / MIPS) unlike FPGAs or CPLDs. Unlike DSPs most of the development effort is not tailored to a specific signal processing task, such as FFTs, image processing, Media Access Controller (MAC), etc.

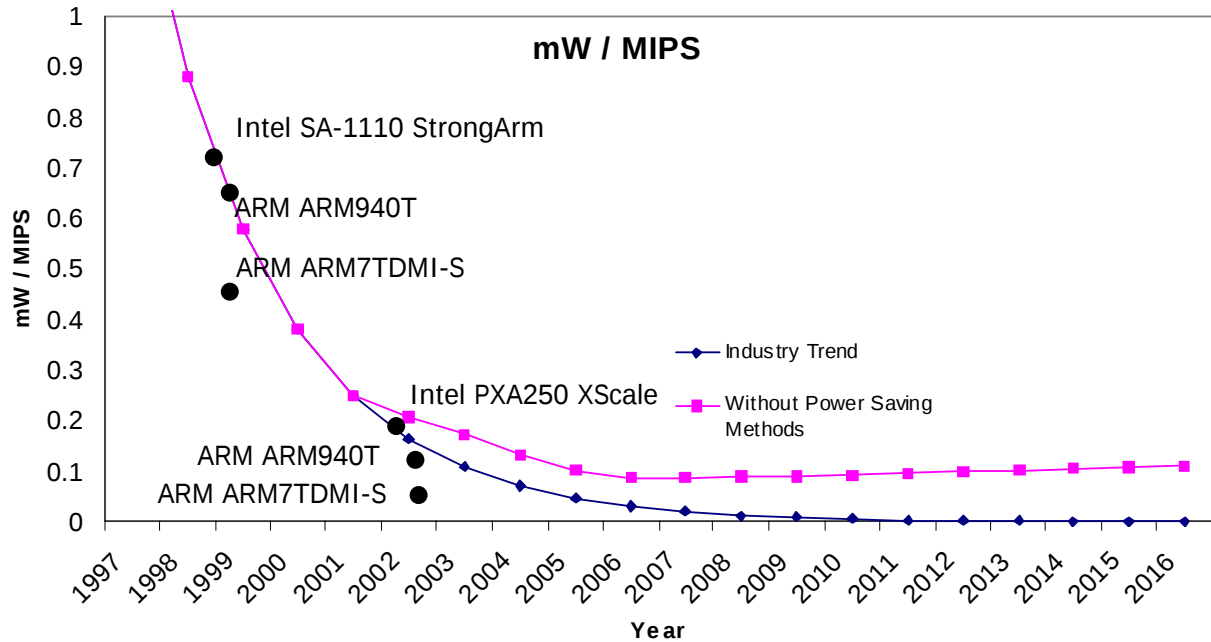


Figure 5 - mW/MIPS vs. Time
Source: ARM Ltd. and Intel Corporation

3.9 Configurable Logic

Configurable Logic devices such as Field Programmable Gate Arrays (FPGA) and Complex Programmable Logic Devices (CPLD) allow us to reconfigure our logic devices to complete different tasks in the most power efficient manner using dedicated hardware. Though there are many companies that manufacture programmable logic, we look at FPGAs and CPLDs from Xilinx Inc. Xilinx is the worldwide leader in FPGA technology, shipping the industry's largest and fastest FPGA. Though Xilinx is a relatively new player in the CPLD industry, their new CPLDs are the most technologically advanced, consuming less power than all competitors.

3.9.1 FPGAs

An FPGA consists of an array of logic blocks, surrounded by programmable I/O blocks, and connected with programmable interconnect. It offers the highest density of all programmable solutions and is static RAM or anti-fuse based.

Xilinx, the leader in FPGA technology, produces the Virtex-II Pro.

- 1.5 Volts operating voltage
- Up to 8 million system gates.
- Runs at 300 MHz.
- 0.13 μm / 9-level copper technology.
- 1 TeraMAC/s possible
- Up to 4 PowerPC cores each running at 300 MHz

3.9.2 CPLDs

CPLDs are composed of macrocells, which contain a sum-of-products combinatorial logic function and an optional flip-flop. They are usually EPROM, EEPROM, or Flash based.

Xilinx produces the most advanced CPLD, the CoolRunner-II series.

- Pure CMOS technology
- Flash Based
- Eliminated sense amplifiers found in other CPLDs
- 1.8 Volt operating voltage
- 100 μA standby current
- Up to 12,000 system gates

(Source: Xilinx Corporation)

3.9.3 CPLDs vs. FPGAs

Table 2 below shows a brief high level comparison of FPGAs and CPLDs

	FPGAs	CPLDs
Density	Around 8 Million gates.	Around 12,000 Gates Density is too low for our application.
Growth	4-5x per Generation Very fast growth	2x per Generation Very slow compared to FPGAs
Standby Power Dissipation	Much higher than CPLDs.	100 μA standby current
Sleep Mode	Does not have a sleep mode. Configuration data is lost in the SRAMs when device is shut off. Reprogramming requires about 300 – 500 mA and takes anywhere from 10 – 100ms. Very high cost to reprogram.	Does not have a sleep mode but offers much more flexibility than FPGAs. Unlike FPGAs, configuration data is not lost. No need to reprogram the CPLD after turning it back on.

Table 2 - FPGAs vs. CPLDs

The low power characteristic of CPLDs is ideal for our application. Unfortunately CPLD density does not grow as fast as FPGAs and the required density will not be reached until around 2008.

3.9.3.1 Simple Sequential Circuit using CPLDs and FPGAs

Traditionally, CPLDs have been chosen over FPGAs whenever high-performance logic is required. Because of its less flexible internal architecture, the delay through a CPLD is more predictable and usually shorter.

	CPLD		FPGA	
	5-bit	13-bit	5-bit	13-bit
Speed	100 MHz	100 MHz	57 MHz	40 MHz

Table 3 - CPLD vs. FPGA Speed Example

The CPLD implementation of the sequential circuit is much faster than the FPGA version. One surprising result is the difference between the 5-bit and 13-bit versions of the circuit. Both versions operated at about 100 MHz for the CPLD implementation, while the 13-bit version was much slower than its smaller counterpart on an FPGA. This is an example of how FPGAs are less suitable for implementing circuits that require “wide” logic gates, where CPLDs can easily implement such designs.

3.9.3.2 Speed Performance of a 32-BIT Counter

CPLDs are also much easier to design for. Table 4 shows an example of a 32-bit counter implementation on a CPLD and multiple implementations on FPGAs. The speed of CPLD based counters is independent of counter size. The original implementation was much faster on the CPLD compared to an FPGA.

	Device Type	# Cells (% of Device)	Maximum Frequency
Basic Design	CPLD	32 (33%)	125 MHz
Basic Design	FPGA	34 (5%)	65 MHz
Design A	FPGA	39 (5%)	103 MHz
Design B	FPGA	48 (7%)	164 MHz

Table 4 - CPLD vs. FPGA Example

Design A and B were a result of tweaking the design to exploit the FPGA architecture. Intimate knowledge of the FPGA architecture and a great deal of manual work was needed on the part of the designer.

3.10 Conserving Energy

Advances in semiconductor technology are making processors, DSPs, reconfigurable logic (FPGAs and CPLDs) more power efficient. Though power consumption is not decreasing, less energy is needed to complete the same task. We must implement “sleep” modes (duty cycling) or “turn off” the device when done processing to take full

advantage of the increase in processing power to save the amount of energy needed. PARIS will intelligently tradeoff different parameters in order to exploit the underlying semiconductor advances.

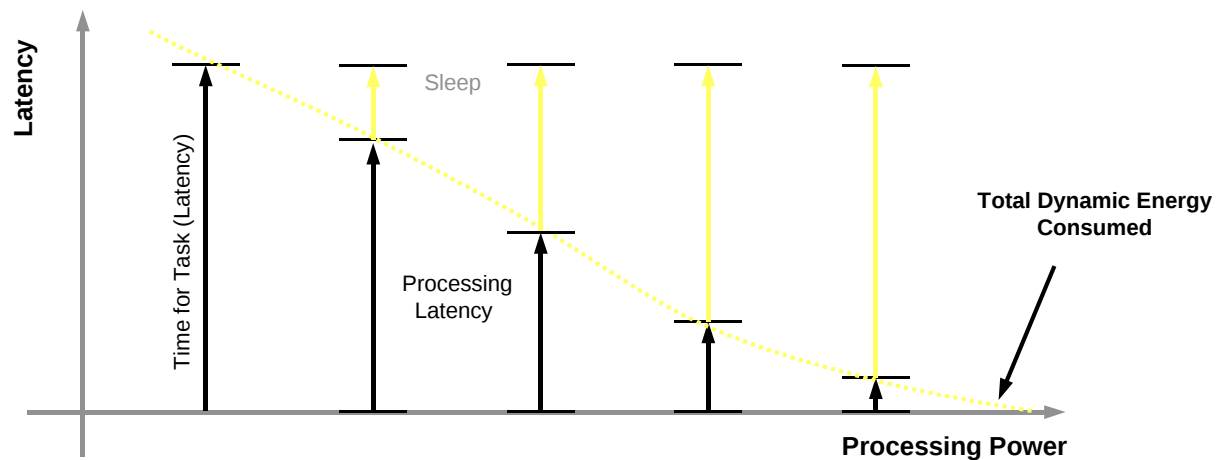


Figure 15 - Latency vs. Technology in Time

3.10.1 Power vs. Latency

Figure 15 shows a study done on the Intel StrongArm SA-1110 processor. Though the power consumption increases linearly with clock frequency, the amount of energy required decreases.

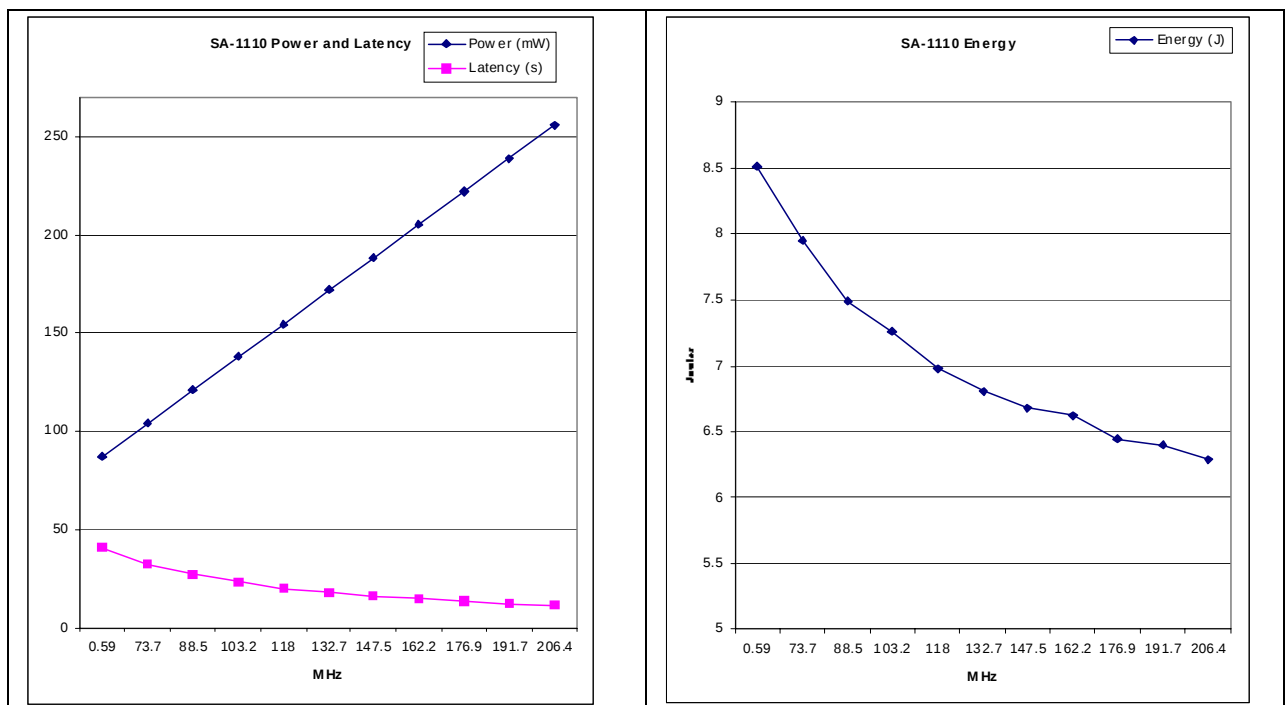


Figure 16 - SA-1110 Power vs. Frequency, SA-1110 Energy vs. Frequency.

3.10.2 Energy Savings Contribution vs. Time

Total energy contributions (or consumption's) come from three main areas: Industry Available Hardware Components contributions, PARIS Reconfigurations contributions and PARIS Mission contributions.

Although lower power and energy can be realized in all three areas in the future, the following figure illustrates that over time, the most energy reduction contributions will come from the power aware PARIS Mission algorithm.

This algorithm will take into account all environmental conditions, available power conditions, and mission demands (profile, configuration, etc.) and maximize the nodes power usage.

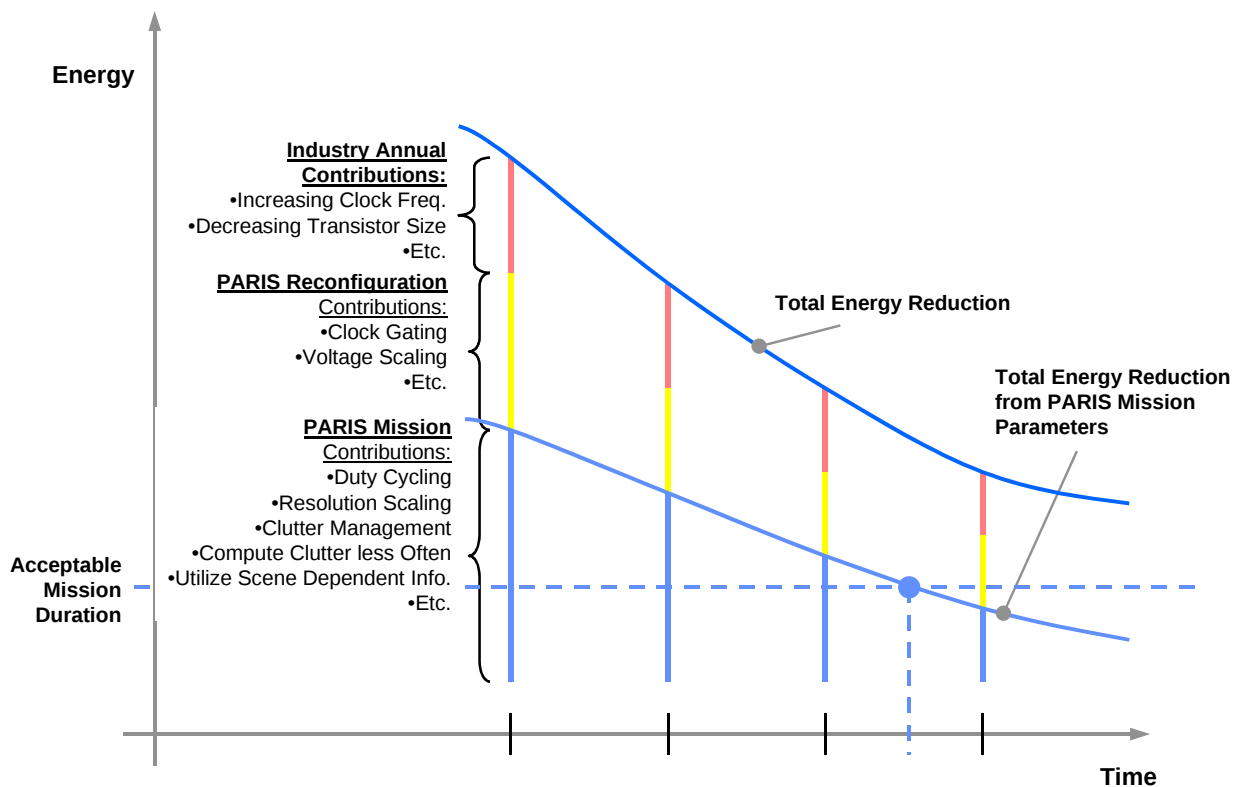


Figure 17 - Energy Improvements vs. Technology in Time

3.11 Memory Considerations

For data intensive applications (e.g., PARIS), memory energy consumption is a significant contribution.

Trends:

- Substantial Increase in number of Transistors per Die for Memory than for Logic
- Dramatic Increasing Percentage of Memory per Die in SoC Applications (up to 95% projected for 2016)

- Increasing Burden on OS and Compilers to take Advantage and Optimize Low Power Memory Features
- Decreasing Power per Memory Cell
- Up to 1/4 (Next Generation SDRAM is double in size, 1/2 Active Power)

PARIS will Exploit Emerging Industry Design Features:

- Architecture Features
- Pipelining
- Cache Implementations
- Paging
- Power Modes
- Active, w/ Access
- Active, no Access
- Power Down

3.12 Algorithm Definition and Power Aware Concepts

Our target application is personnel detection and tracking. We will address the introduction of power aware concepts through the introduction of an Intelligent Mission Driven Application Framework.

The mission need will control:

- Sensor Tasking / Scheduling, Parameter Selection, e.g. Data Rate, Sample Resolution
- Algorithm Selection / Parameter Selection Based on Environmental Data & Required Accuracy

We will exploit more computationally enhanced algorithms to improve Pd and Pfa while adjusting power utilization to achieve mission need.

3.12.1 Sensor Engine Clock Diagram

The figure below shows the Sensor Engine Block Diagram for a Raytheon aSi uncooled IR camera. The total power consumption is approximately 800 mw at a 20 Hz frame rate.

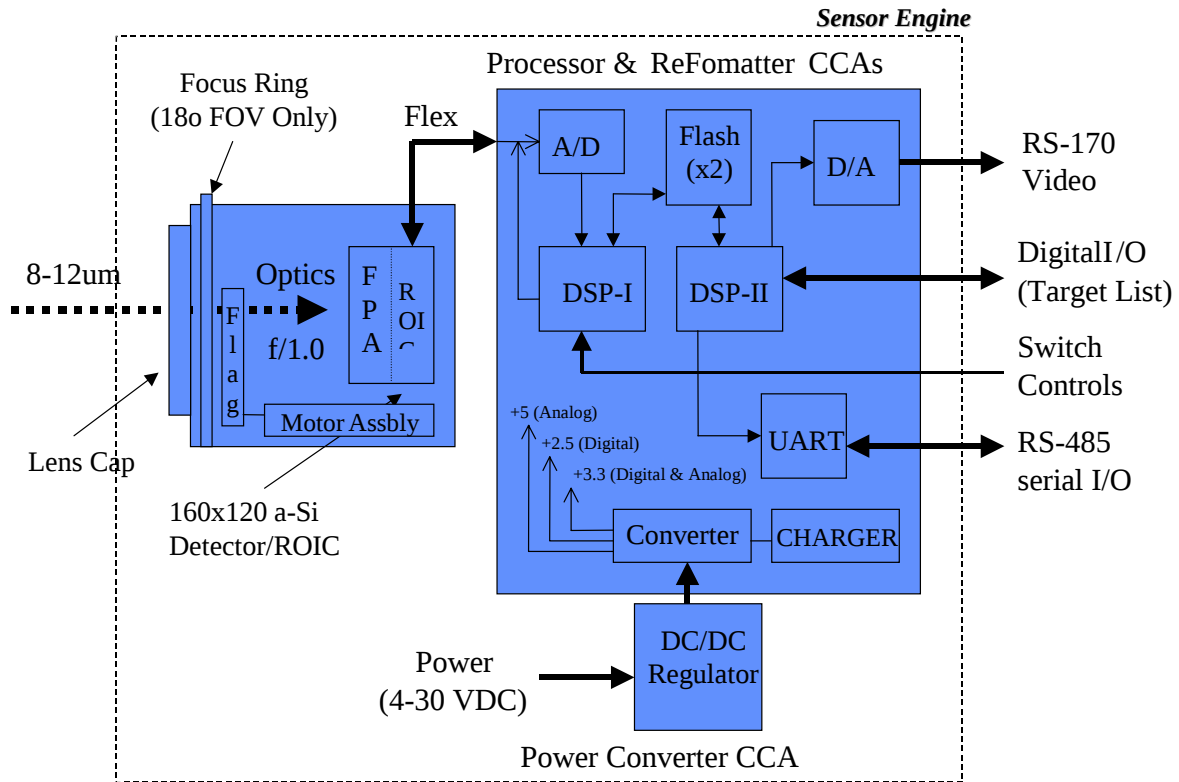
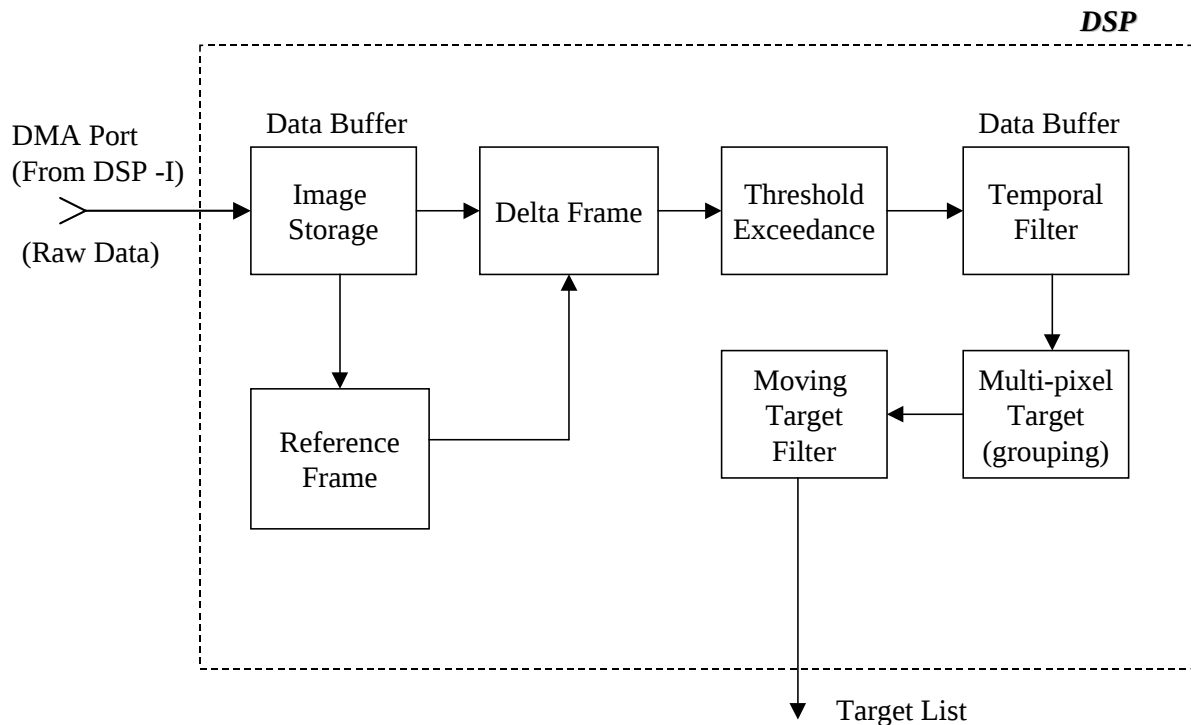


Figure 18 – aSi Uncooled IR Sensor Engine Block Diagram



Algorithm Works Well in Benign Environments, e.g. Indoors

Figure 19 - Baseline ATD/MTI Algorithm

3.12.2 Baseline ATD/MTI Function

A key element of the PARIS design will be a software framework to adaptively activate processing routines based on the sensor events, software selections, and mission goals. The baseline sensor processing algorithm was developed on the previous IR Personnel Detection (IRDP) program and was integrated into the aSi sensor. This algorithm shown in figure 18 was optimized for simplicity / low power at the expense of detection/false alarm rate goodness. A new class of detection and tracking algorithm originally developed for cruise missile applications apply more complex mathematics and fusion techniques to the human detection and tracking problem. This algorithm provides opportunities to trade mission effectiveness and power.

3.12.3. Overview of the Detection Paradigm

Figure 20 shows the Adaptive Moving Target Indicator Algorithm. This algorithm uses a frame specific feature extraction and tracker. One innovative feature is in the detection filter's "track before detect" feature, wherein sensor inputs are analyzed in alternative parallel fashion, reducing noise and augmenting true signal prior to declaring a track. The result is a spatial-temporal algorithm able to detect and track targets with signal (clutter + noise) ratios [S(C+N)R] as low as 3 with no false tracks.

- Designed for the Detection of Small Dim Targets in Heavy Clutter
- Local DeMean: Removes Low Frequency Clutter
 - Clutter Sample Covariance
 - Matched Filter
- V-Filter Bank: Bank of Velocity Filters
- Local CFAR: Clutter Adaptive Constant False Alarm Rate Detector
- MHT/IMM: Multiple Hypothesis Tracker with Interacting Multiple Models
- Track Features
- Allow the Use of Lower Detection Thresholds
- Faster Track Confirmation Times

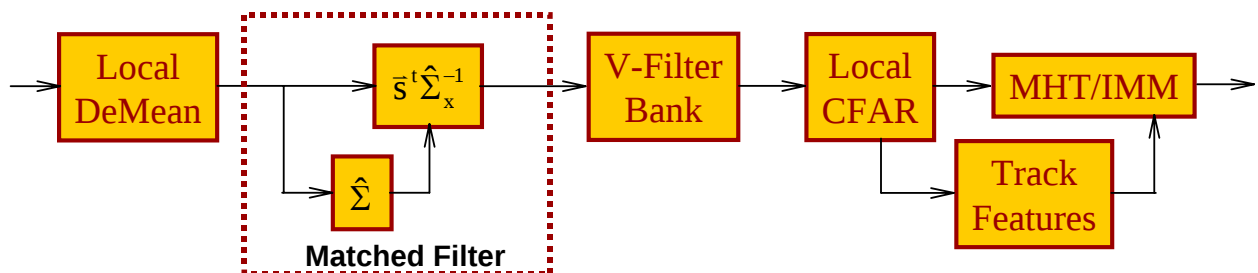


Figure 20 - Detection Algorithm

3.12.4 Potential Algorithm Processing Techniques to Reduce Power

3.12.4.1 Control of Processing Power

- Collect the Next Frame Only When the Tracker Needs an Update
- Use the Tracker to Selectively Process ROI's within the Collected Frame
- Update All Tracks within the Collected Frame
- Trade Detection Range for Reduced Power Consumption
- Reduce the Size n of Processing Windows - A Performance Loss
- Let N be the Total Number of Pixels
- Let n be the Number of Pixels per Window
- Number of Additions:

$$N \left(\frac{3n-1}{2} \right)$$

- Number of Multiplications:

$$N \left(\frac{3n+1}{2} \right)$$

- Consider Scene Dynamics when Determining the Need to Update the Clutter Covariance

3.12.4.2 Reduce Power Consumption by Reducing Word Size

Figure 21 illustrates trades that can be made for different amounts of resolution for a given sensor. In this example, a resolution of 2 was examined. The steps of quantization loss appear to be significant. Performance Loss Due to Quantization Depends Upon: Number of Quantization Levels and Operating Point, e.g. false alarm rate.

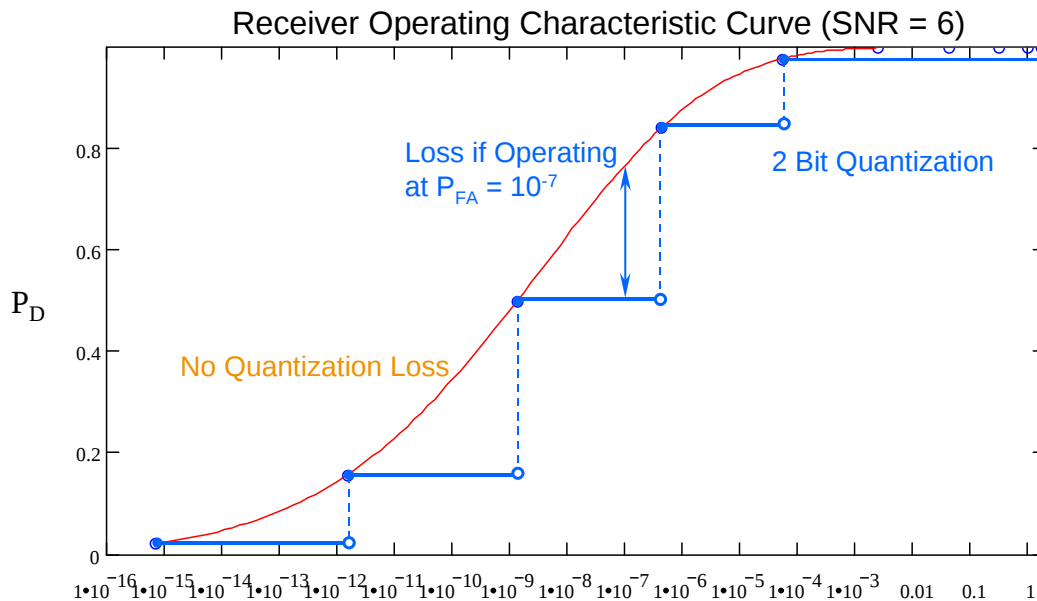


Figure 21 - Receiver Operating Characteristic Curve

4. REFERENCES

- The International Technology Roadmap for Semiconductors 2001 Edition*, Semiconductor Industry Association, San Jose, California., 2002; <http://public.itrs.net/>.
- S. Borkar. Design Challenges of Technology Scaling. *IEEE Micro*, 1999, 1999.
- J. Butts, G. Sohi. A Static Power Model for Architects. *33rd International Symposium on Microarchitecture*, 2000.
- A. Cataldo, D. Lammers. Cracking IC's Toughest Nuts – No easy fix for power issues. *Electronic Engineering Times*, June 17, 2002.
- W. D. Farwell. Future growth of Semiconductor Technology. *Unpublished work for Raytheon Company*, 2001.
- J. Flinn, K. I. Farkas, J. Anderson. Power and Energy Characterization of the Itsy Pocket Computer (Version 1.5), Available on-line from <http://research.compaq.com/wrl/projects/itsy>.
- J. Pouwelse, K. Langendoen and H. Sips. Dynamic Voltage Scaling on a Low-Power Microprocessor. *Mobile Computing Conference (MOBICOM)*, 2001.
- S. Thompson, P. Packan, and M. Bohr. MOS Scaling: Transistor Challenges for the 21st Century. *Intel Technology Journal*, Q3 1998.

Appendix D: PARIS Emulation Testbed Design

TABLE OF CONTENTS

1. SCOPE.....	44
1.1 MISSION	44
2. APPLICABLE DOCUMENTS	44
3. EMULATION TESTBED REQUIREMENTS	44
3.1 INTERFACE REQUIREMENTS.....	44
3.2 DIGITAL SENSOR INTERFACE.....	44
3.3 ANALOG SENSOR INTERFACE	45
3.4 RADIO INTERFACE	45
3.4.1 <i>Radio Data Interface</i>	45
3.4.2 <i>Radio Controls Interface</i>	45
3.5 POWER.....	45
3.6 DEVELOPMENT SYSTEM REQUIREMENTS	45
3.6.1 <i>Host Development System Requirements</i>	45
3.6.2 <i>Target System Requirements</i>	45
3.6.3 <i>PASTA Development System Requirements</i>	46
3.7 EMULATION/EVALUATION REQUIREMENTS.....	46
3.7.1 <i>Clock Scaling</i>	46
3.7.2 <i>Power Measurement</i>	46
3.8 EMULATION/EVALUATION RESOURCE REQUIREMENTS	46
4. EMULATION TESTBED CONCEPT	46
4.1 EMULATION TESTBED CONCEPT OVERVIEW	46
4.2 EMULATION TESTBED HARDWARE	47
4.3 EMULATION TESTBED SOFTWARE	48
4.4 EMULATION TESTBED EVALUATION BOARD/TOOLS	48
4.5 EMULATION TESTBED INTERFACES	49
4.5.1 <i>Emulation Testbed aSi Camera Host Port Interface</i>	49
4.5.2 <i>Emulation Testbed PASTA Interfaces</i>	49
4.5.3 <i>Emulation Testbed PASTA Control</i>	51

List of Figures

FIGURE 1 - EMULATION TESTBED CONCEPT	47
FIGURE 2 - EMULATION TESTBED HARDWARE	48
FIGURE 3 - EVALUATION BOARD DEVELOPMENT/TOOLS	49
FIGURE 4 - ASI CAMERA HOST PORT INTERFACE	50
FIGURE 5 - PASTA INTERFACES WITH PARIS MODULE	50
FIGURE 6 – TOP LEVEL CONTROL STRUCTURE FOR A PARIS MODULE WITHIN A PASTA STACK.....	51

1. Scope

This document contains the requirements and design concept for the PARIS Node Emulation Testbed. It is a living document and will be updated as PASTA stack hardware details are refined and their impact on the PARIS emulation system elements is assessed.

1.1 Mission

The PARIS Emulation Testbed shall provide hardware, software, software tools, and interfaces for emulating and evaluating system architecture concepts for the design of the final PARIS Node. This system will allow developers to “turn Power knobs” by modifying existing algorithms and structures in order to emulate and evaluate (measure) performance differences.

The PARIS Emulation testbed will use elements of the hardware stack and software being developed under the PASTA effort.

2. Applicable Documents

None

3. Emulation Testbed Requirements

The subsequent paragraphs list the requirements for the PARIS Emulation Testbed.

3.1 Interface Requirements

The PARIS Emulation Testbed shall utilize a subset of the interfaces of the PASTA Stack (PASTA uAMPS II System Architecture). These interfaces shall include the common digital address and data I²C interface, and a power control interface.

The PARIS Emulation Testbed shall be able to operate with or without the PASTA Processor Module in the stack.

The PARIS Emulation Testbed shall support a modular interface to external sensors such as an IR sensor, acoustic sensor, magnetic sensor, etc.

3.2 Digital Sensor Interface

The PARIS Emulation Testbed shall provide digital data interfaces for the aSi uncooled IR sensor.

The aSi digital data interface consists of an TI DSP 8-Bit Host Port Interface (HPI-8) Interface (see Appendix A).

3.3 Analog Sensor Interface

The PARIS Emulation Testbed shall provide analog interfaces for an acoustic sensor.

3.4 Radio Interface

3.4.1 Radio Data Interface

The PARIS Emulation Testbed shall provide a PASTA Data I²C Interface. Format, protocol and data content definition are TBD.

3.4.2 Radio Controls Interface

The PARIS Emulation Testbed shall provide a PASTA Power Control Interface. Data and bit definitions are TBD.

3.5 Power

The PARIS Emulation Testbed shall be powered by either battery or power supply.

The Hardware shall allow sub-elements to be powered off, placed in a sleep mode or removed if not used to minimize power consumption.

3.6 Development System Requirements

3.6.1 Host Development System Requirements

The PARIS Emulation Testbed Development System shall provide interfaces for downloading of developed code to commercial host evaluation boards.

The PARIS Emulation Testbed Development System shall provide a platform for developing (editing, compiling, synthesizing) target hardware code (VHDL and C).

3.6.2 Target System Requirements

The PARIS Emulation Testbed hardware shall support software re-programming / re-configuration either onboard or via an external connector.

The PARIS Emulation Testbed hardware startup time shall be minimized and not exceed 100 ms.

The PARIS Emulation Testbed hardware shall be expandable through either bus extenders, plug-in slots, or daughter cards.

The PARIS Emulation Testbed shall provide PC interfaces for downloading of VHDL developed code to the target reconfigurable hardware (FPGAs, CPLDs, ASICs, etc.).

The PARIS Emulation Testbed shall provide PC interfaces for downloading of C developed code to the target processing hardware.

3.6.3 PASTA Development System Requirements

The PARIS Emulation Testbed shall maintain provisions for the PASTA stack development system interfaces.

3.7 Emulation/Evaluation Requirements

3.7.1 Clock Scaling

The PARIS Emulation Testbed hardware shall support variable clock frequencies (evaluate clock scaling).

3.7.2 Power Measurement

The PARIS Emulation Testbed shall provide power measurement “hooks” to allow for power measurements for on different hardware functions and operating modes. The hardware shall be capable of allowing power consumption to be monitored.

3.8 Emulation/Evaluation Resource Requirements

The PARIS Emulation Testbed shall utilize commercial evaluation/test boards from common suppliers (Xilinx, Altera, Actel, Chameleon, IBM, etc.).

The PARIS Emulation Testbed shall utilize the PASTA stack provided by USC ISI East.

The PARIS Emulation Testbed shall utilize sensors provided by Raytheon.

4. Emulation Testbed Concept

4.1 Emulation Testbed Concept Overview

The purpose of the PARIS Emulation Testbed is to provide a “hands-on” development environment system and evaluation station for the PARIS Node, in a cost effective manner. It must also satisfy the requirements as stated above. In order to accomplish this, commercial hardware evaluation boards and standard software development tools will be exploited.

The Emulation Testbed consists of 4 main sections (see Figure 1).

First are the “Sensors”. The sensors consist of inferred imaging cameras and acoustic sensors. They provide sensor data which is fed to the Target Commercial Evaluation Boards. These sensors are also controlled by these boards. These sensors are provided by Raytheon.

Second, is the Target Commercial Evaluation Boards. These boards consist of commercial hardware evaluation boards. These boards have the capability of interfacing with both the Sensors and PASTA Stack. The boards may be modified by Raytheon for inserting power measurement instrumentation, clock control, etc.

Third, is the Host Code Development System. This section consists of a PC, which serves as the development workstation for C code and VHDL code development. It will

also host any software or hardware development tool that is required. In addition, this system will host the code for the Imaging and Target Algorithms.

Fourth, and last, is the PASTA Stack. The PASTA stack will be provided by USC-ISI East.

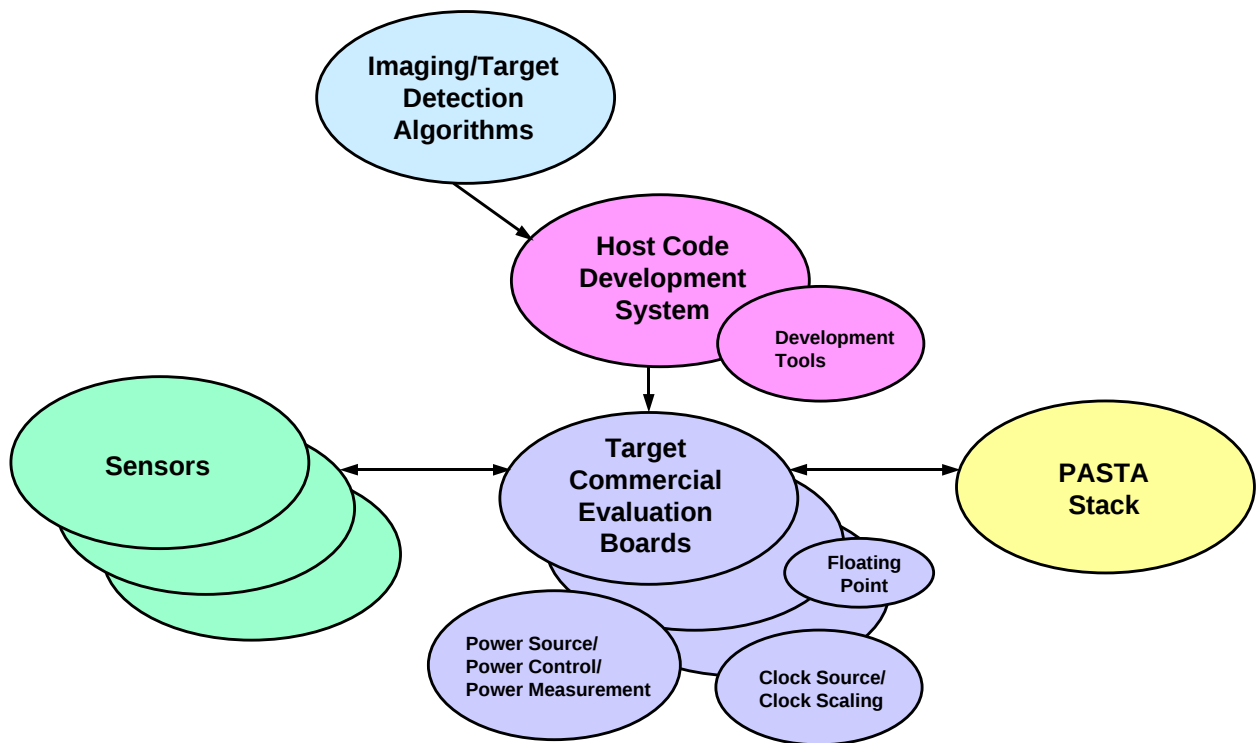


Figure 1 - Emulation Testbed Concept

4.2 Emulation Testbed Hardware

The PARIS Testbed Hardware consists of sensors, Target Commercial Evaluation Boards, a PC, a PASTA Stack, and a Power Source (see Figure 2).

The Target Commercial Evaluation Boards consist of commercial, “off-the-shelf” boards that can emulate and evaluate reconfigurable architecture devices (Actel FPGAs, etc.), emulate and evaluate reconfigurable architecture processing (Chameleon Systems, etc.), and emulate and evaluate the PARIS Node micro-controller 8051. Also included, is a Prototype Board that contains power switching and signal switching devices.

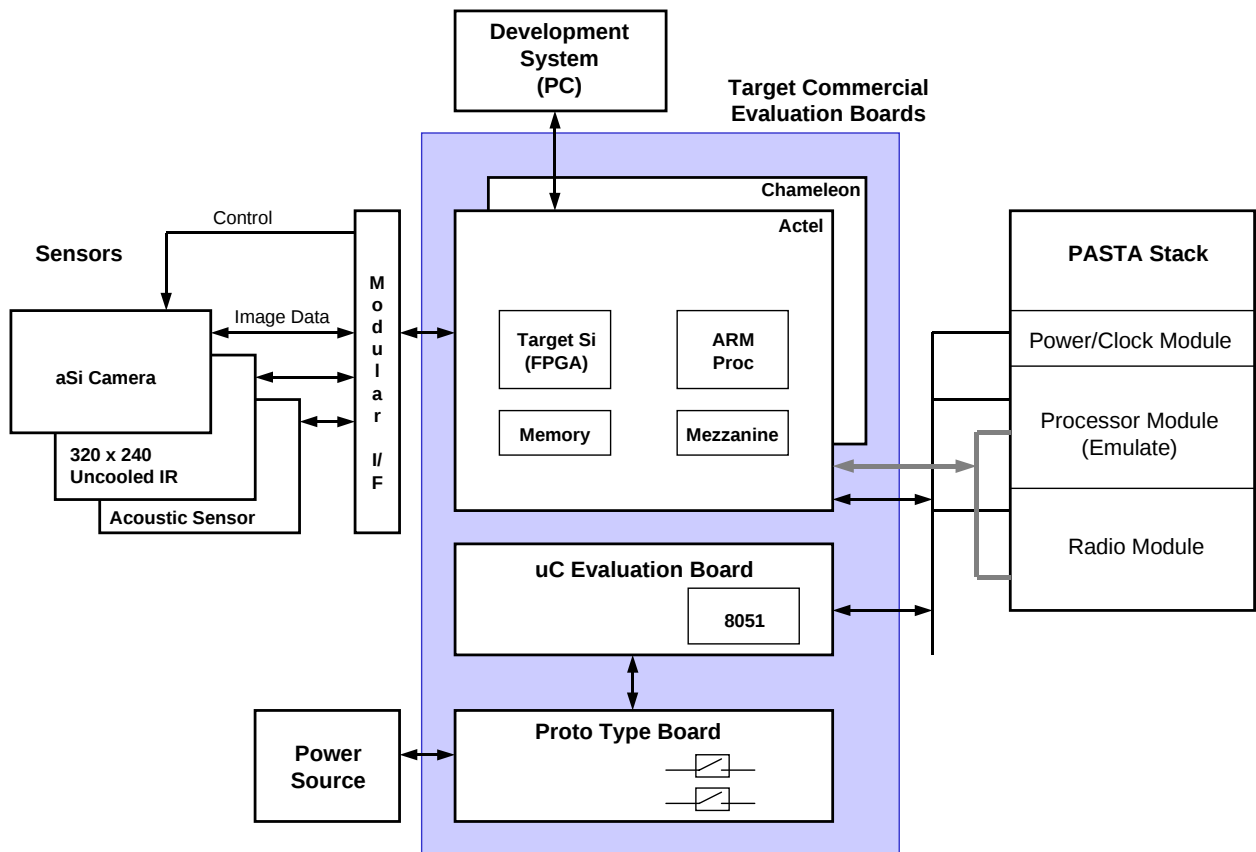


Figure 2 - Emulation Testbed Hardware

4.3 Emulation Testbed Software

The Emulation Software consists of standard commercial software development tools. These tools include editors and compilers for both C code (processor development) and VHDL (reconfigurable architecture development). Also, the software will contain programs to download the code to the evaluation boards, and program the target devices. The system level control software will leverage the software being developed under the PASTA effort. Figure 3 shows a preliminary control structure of a PARIS module within the structure of a PASTA stack.

4.4 Emulation Testbed Evaluation Board/Tools

The Emulation Testbed contains at least one commercial FPGA Evaluation Board (See Figure 3). An example of this type of board is the Actel ProASIC Plus System Design Board SBD-750/1000 from Inicore. This board contains both an Actel ProASIC FPGA (either 75K or 1M System Gates) and an ARM Processor. In order to support FPGA development, compilation, simulation, place and route, synthesis and power estimation tools will be provided. Similarly, for the ARM Processor, a compiler, debugger and downloader will be provided. These tools are available from a variety of sources (GNU, Green Hills, Jenni, etc.). This board and support tools will allow development of VHDL coding, simulation and synthesis, prior to downloading to the target evaluation board.

The ARM processor has the capability of configuring the FPGA via the JTAG interface. The ARM processor may also be developed to assist the FPGA in PARIS image processing and detection/target tracking algorithm tasks.

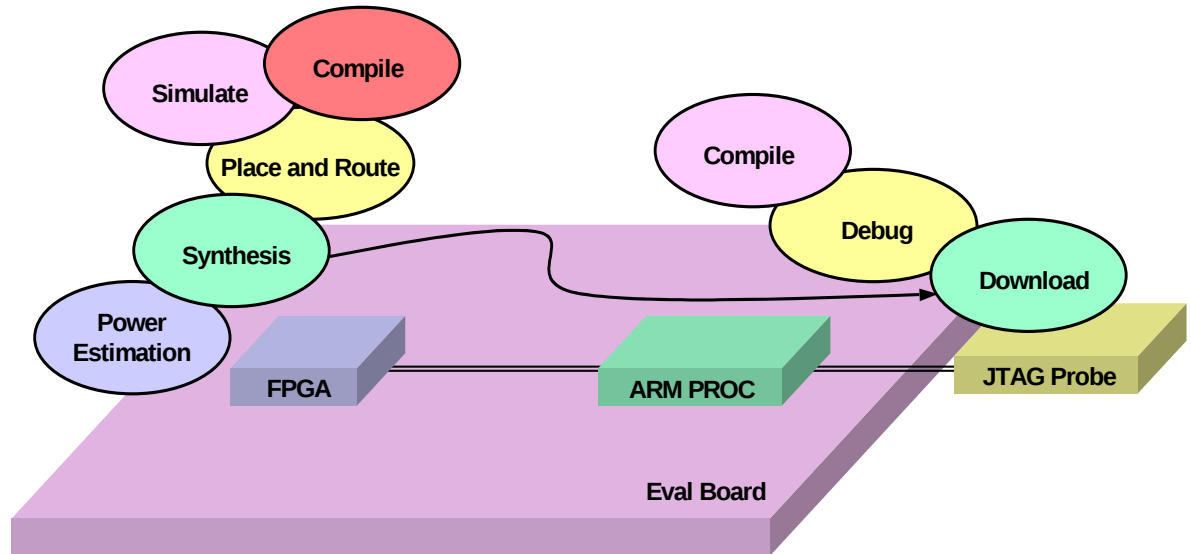


Figure 3 - Evaluation Board Development/Tools

4.5 Emulation Testbed Interfaces

4.5.1 Emulation Testbed aSi Camera Host Port Interface

The Inferred aSi Camera digital interface is implemented by the TI TMS320VC5410 DSP Host Port Interface. The digital interface consists of an 8-bit bi-directional data bus and 10 discrete handshake/control/status signals (see Figure 4).

4.5.2 Emulation Testbed PASTA Interfaces

The Emulation Testbed PASTA Interfaces consist of a 3.3V DC Regulated Power Bus and two I²C interfaces (within the Bank Signaling Bus and Control Bus). The Power Source (shown as a battery in Figure 5) is fed to the Power/Clock Module in the PASTA Stack. The Power/Clock Module switches the power, and feeds it to the PARIS Module, which is emulated by the hardware in the PARIS Emulation Testbed. The Controller on the PARIS Module communicates with the Power/Clock Module via a I²C interface in the Control Bus. This function will be implemented in the PARIS Emulation Testbed by the micro-controller Evaluation Board. The PARIS Module Hardware communicates with the Radio Module via the I²C interface in the Bank Signaling Bus. This will be implemented in the PARIS Emulation Testbed by the Target Commercial Evaluation Boards.

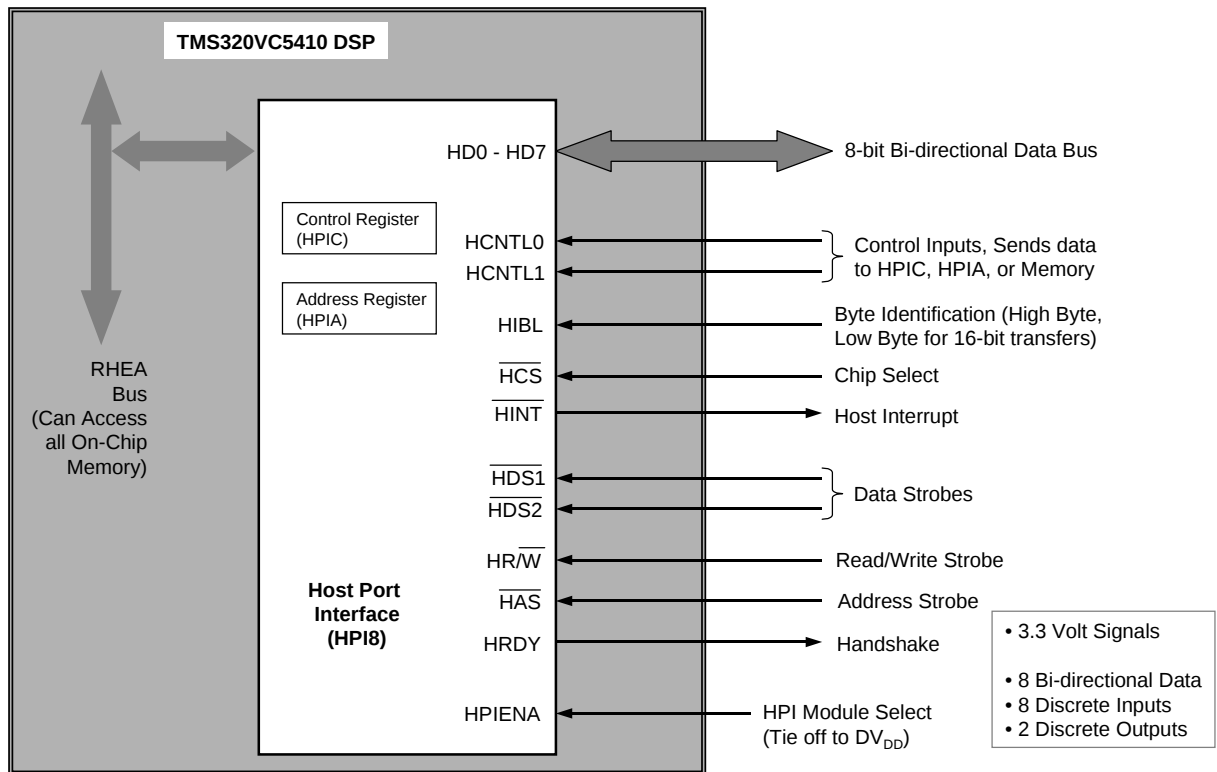


Figure 4 - aSi Camera Host Port Interface

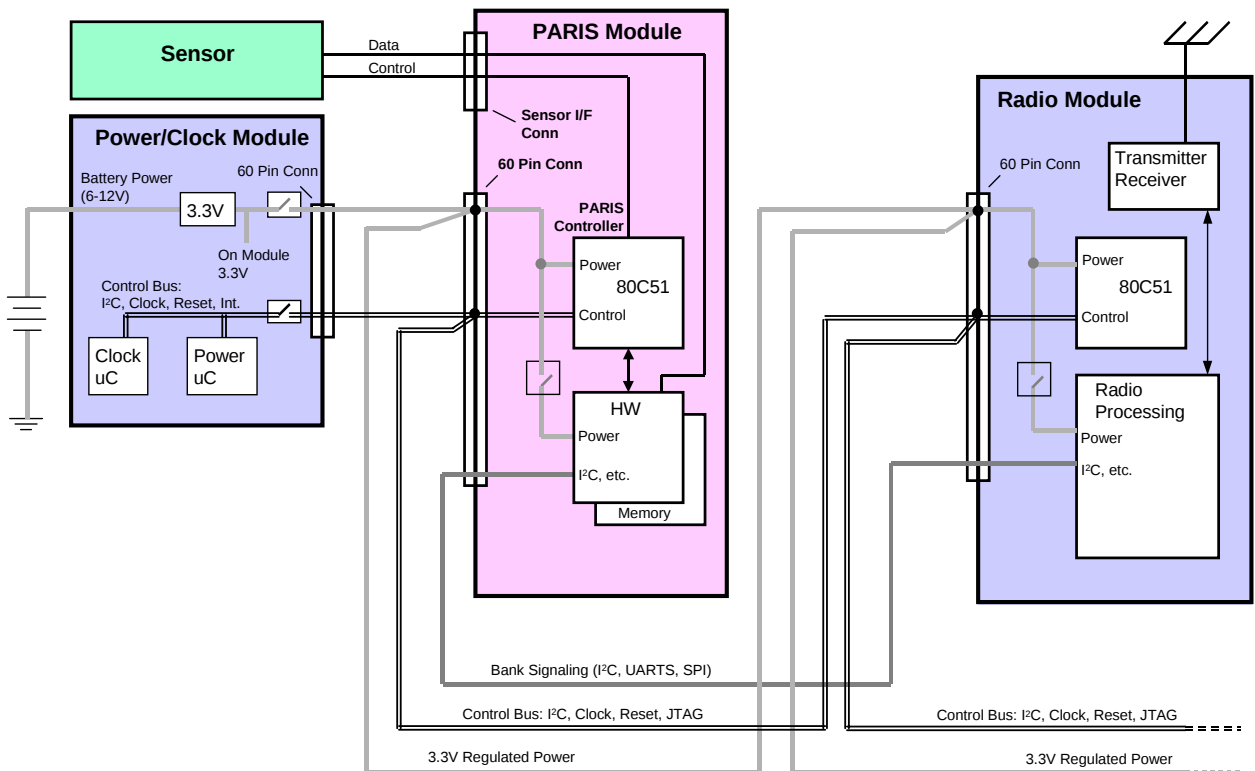


Figure 5 - PASTA Interfaces with PARIS Module

4.5.3 Emulation Testbed PASTA Control

The PASTA Stack, with the PARIS Module inserted, will allow for wake-up and data communication via control commands and messages (see Figure 6). The Power Controller in the PASTA Stack will send control messages to the PARIS Module and Radio module to setup interconnects and to turn on select modules. The PARIS module has its own controller to issue commands to the other PARIS Module Hardware such as to initialize, sleep, shutdown, change parameters to send status.

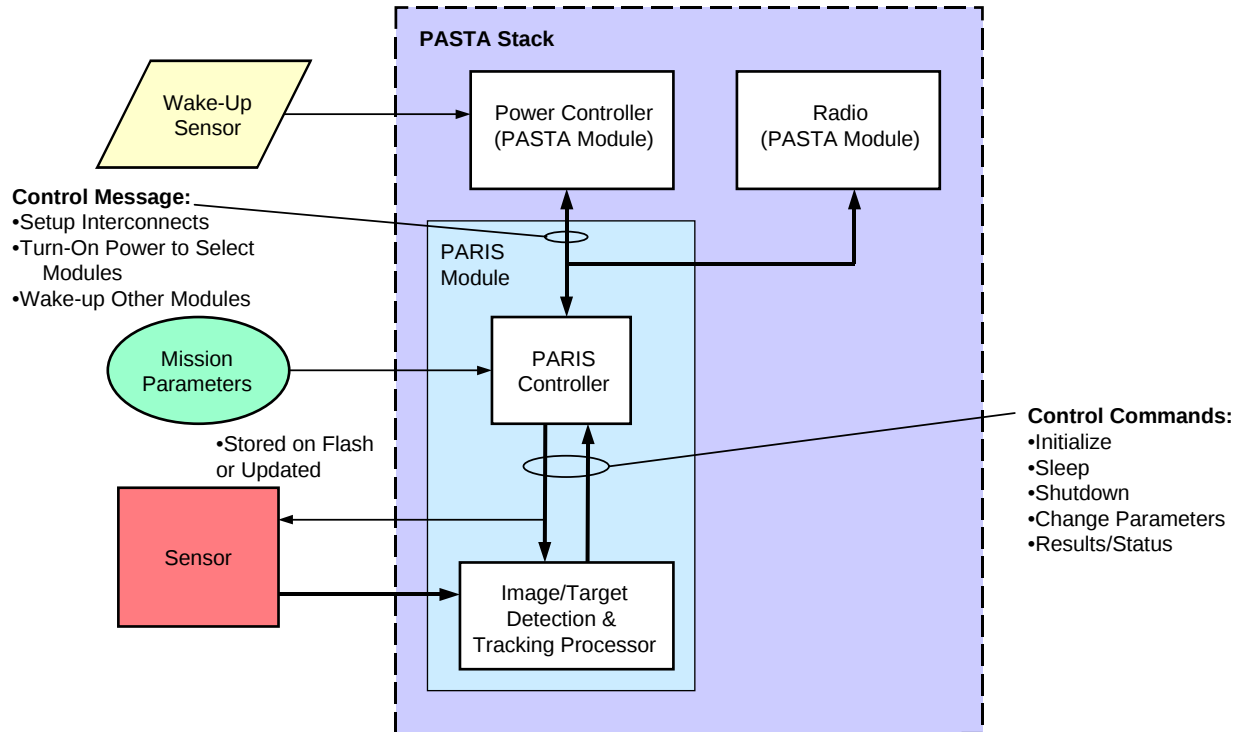


Figure 6 – Top level control structure for a PARIS module within a PASTA stack.

Appendix E: High Level Performance Estimation and Validation

Table of Contents

1. INTRODUCTION	54
2. HIGH-LEVEL PERFORMANCE ESTIMATION FOR KERNELS	56
2.1 HIGH-LEVEL PERFORMANCE ESTIMATION FOR KERNEL IMPLEMENTATIONS ON EMBEDDED PROCESSORS AND DSPS	56
2.2 HIGH-LEVEL PERFORMANCE ESTIMATION FOR KERNEL IMPLEMENTATIONS ON FPGAs	57
3. VALIDATION	59
3.1 VALIDATING DOMAIN-SPECIFIC MODELS	60
3.2 VALIDATION OF DEVICE COMPARISONS	60
4. CASE STUDY: DEMEAN FOR THE PARIS APPLICATION	61
4.1 PROGRAMMABLE PROCESSOR IMPLEMENTATIONS	61
4.2 FPGA IMPLEMENTATION	62
5. SYSTEM-WIDE HIGH-LEVEL PERFORMANCE ESTIMATION.....	63
6. CONCLUSION	65

1. Introduction

There are many dimensions in the design space for the PARIS application (see figure 1). This abundance of variables leads to a design space containing approximately 35,000 designs. Clearly, it is not possible for a design team to evaluate all such designs. As a result, we employ the MILAN modeling framework to model the application and resources and to explore the design space.

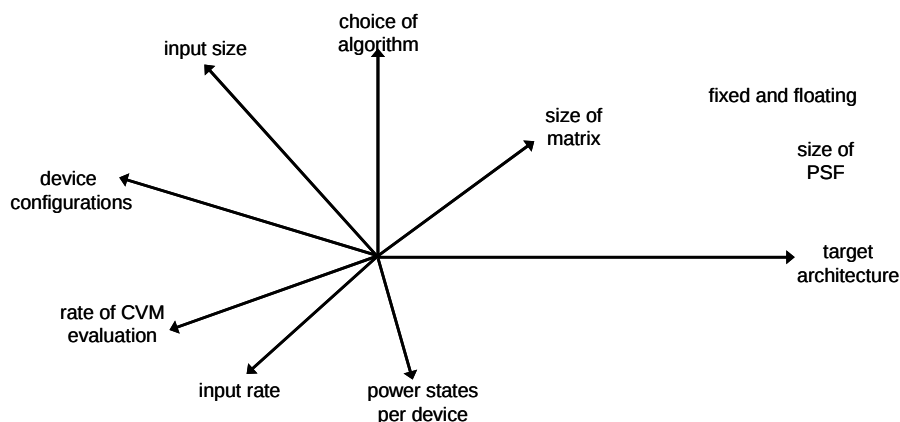


Figure 1: Dimensions in the PARIS design space

The flow for modeling an application in MILAN is shown in figure 2. First, the tasks in the application are modeled in the *application model* and the possible target hardware devices are modeled in the *resource model*. The *mapping model* captures performance data for each task when mapped onto each resource. For example, once the latency, energy, and area for demeaning on an Actel ProASIC FPGA are known, this information is captured in the mapping of the demean task in the application model to the Actel ProASIC FPGA in the resource model. With these performance numbers and user-provided constraints on the design, it is possible to use DESERT and HiPerE to explore the design space. The quality of this system-level design space exploration is largely dependent upon the quality of the performance numbers. It is, therefore, of paramount importance to have accurate performance numbers in the mapping model.

There are several methods by which performance numbers can be obtained and entered into the mapping model. One is to implement the code on the actual hardware and make measurements to find the results. Two main drawbacks to this method are that it requires that the designers have already purchased all the target hardware devices and that it requires that the designers have already coded and verified a working design. Because of these drawbacks, this method of populating the mapping model is infeasible.

Another method is to simulate the designs. In this method, actual hardware is not required. However, there is still a need for the developers to have access to simulators for every target hardware device and the developers must have implementable code for each design. Further, simulating each design is very time-consuming. For an application

with as large a design space as that of the PARIS application, it is impractical to simulate every possible design.

Because of the infeasibility of the aforementioned methods for populating the mapping model, we have developed methods for rapidly estimating the performance of implementations on different types of hardware. We term these methods *high-level estimation* to distinguish them from the time-consuming, low-level simulations.

There are two parts to the high-level estimation of the PARIS application. The first is to estimate the performance of each of the kernels in the application. The second is to estimate the performance of the complete application, that is, of the kernels working together in sequence. We describe each of these in this report.

2. High-Level Performance Estimation for Kernels

FPGAs, embedded processors, and digital signal processors are the types of hardware that are studied for use as part of the PARIS node. We have one method of estimation for kernel implementations for embedded processors and digital signal processors and one method for kernel implementations on FPGAs.

2.1 High-level Performance Estimation for Kernel Implementations on Embedded Processors and DSPs

To estimate the performance of kernel implementations for DSPs and embedded processors, we use a combination of analysis of the algorithms used in the implementations and the datasheet information about the processor.

Analysis of the algorithm is used to determine the number of operations that will be executed. This information is part of determining the latency for this implementation of the kernel. If operation counts can be determined by running some version of the algorithm on sample input, these operation counts can be substituted for analysis of the algorithm. For instance, Raytheon has provided the USC team operation counts for the kernels in the FAST algorithm. The USC team uses these operation counts in their determination of the latency and energy for each kernel (see section 4 for an example). Note that using the operation counts in the latency determinations provides a lower bound on the latency; control overhead is not included.

With the operation counts, there are several ways to proceed in order to find the latency. If some measure of operations per second is available, this can be used to find the latency. For example, the Analog Devices TigerSharc DSP datasheet provides the peak MFLOPS of that device. Knowing the operations per second and the number of operations, it is easy to determine the latency, in seconds. If no such operations per second information is given, the designers use information about the pipelining of the device and any special instructions available. For example, the Intel PXA250 has a dual-MAC instruction with its own pipeline. Thus, it can be assumed in the latency calculations that MACs can be done two per cycle with the PXA250. The number of

cycles required to execute the operations in the algorithm and the frequency of the device are then used to estimate the latency.

The power dissipation of the devices is obtained from the datasheets. Oftentimes, the power is given for different activity levels. Based on the algorithm for the kernel, the appropriate activity level is selected. To determine the energy dissipation for a device executing an algorithm for a kernel, the power dissipation is multiplied by the latency.

2.2 High-level Performance Estimation for Kernel Implementations on FPGAs

FPGAs do not have a high-level structure comparable to that of DSPs and embedded processors. This lack of high-level structures in FPGAs makes modeling their performance more difficult. As a result, we have developed *domain-specific modeling* to estimate performance of kernels mapped onto FPGAs. Domain-specific modeling, described in detail below, is a top-down + bottom-up approach to performance estimation in that it involves both the analysis of algorithms (top-down) and the low-level simulation of components in the architecture¹ (bottom-up), which is a much faster process than simulating an entire kernel design.

A *domain* is an algorithm-architecture pair such as matrix multiplication on a linear array of processing elements (PEs). Figure 3 is an illustration of the first step in domain-specific modeling: the selection of a domain. When choosing the domain, the designer must consider the characteristics of the target FPGA. For example, in Xilinx FPGAs, long wires dissipate much more power than local connections. Thus, when designing for energy efficiency, designers should try to utilize domains where the architecture employs as few long wires as possible. Such domains include those with single processing element or linear array of PEs architectures.

As figure 3 shows, after domain selection, the domain(s) are modeled. In this process, the architecture is abstracted into *components*. Components are of two types: *Relocatable Modules* (RModules) and *Interconnect*. RModules are computation and storage units that are assumed to dissipate the same amount of power regardless of their location on the chip. Examples of RModules include registers, on-chip memory, and multipliers. Interconnect is the connection between the RModules.

Once the components have been identified, the next step is to determine their power dissipations. This is accomplished through low-level simulation of the components on the target device (the bottom-up portion of domain-specific modeling). Figure 4 illustrates the flow of this procedure when the target FPGA is a Xilinx Virtex-II. The flow for other FPGA devices is similar, though the tools that are used vary. A component is coded in VHDL. It is then synthesized and placed-and-routed. The output from place-and-route is converted to back-annotated VHDL and simulated. The testbench

¹ Throughout this report, architecture refers to the virtual architecture (like a linear array of processing elements) that is mapped onto an FPGA. It does not refer to an instruction set architecture or a type of device.

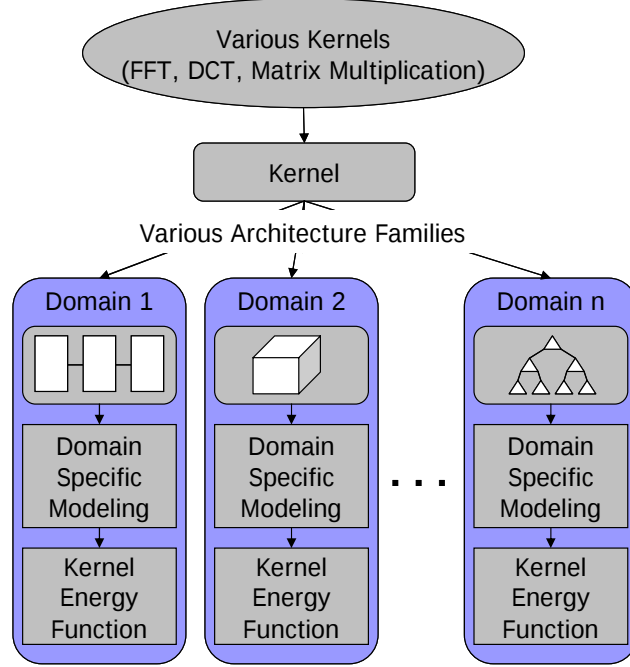


Figure 3: Domain selection

waveforms for simulation consist of randomly generated data. We have employed 1000 samples of data from a uniform distribution which leads to an input switching activity of 50%. The switching activity influences the power dissipation of the component. If the characteristics of the input are known, the testbench waveforms can be adjusted accordingly. Or, if sample input is available, it can be used in the waveforms. In simulation, a vcd file is created. This file catalogs the activity on each of the signals in the design. The vcd file and the output from place-and-route can then be used to determine the average power dissipation of the component. If the components can run in different power states during the course of the algorithm, this procedure should be followed for each power state. It is important to note that this procedure need only be done once per component and then the data can be reused. That is, once the power dissipation of a component has been determined for one implementation of one kernel in one domain, that power dissipation value can be used in the estimations for that component in other domains for other kernels. This data reuse limits the amount of low-level simulation that must be done.

Another step in domain-specific modeling is the creation of the component power state (CPS) matrices for each type of component (see figure 5). This matrix describes for each component, for each cycle of the algorithm, which power state the component is in. Making the CPS matrix requires analysis of the algorithm (the top-down portion of domain-specific modeling). Analysis of the algorithm yields both the total number of cycles and which power state each component is in at each cycle.

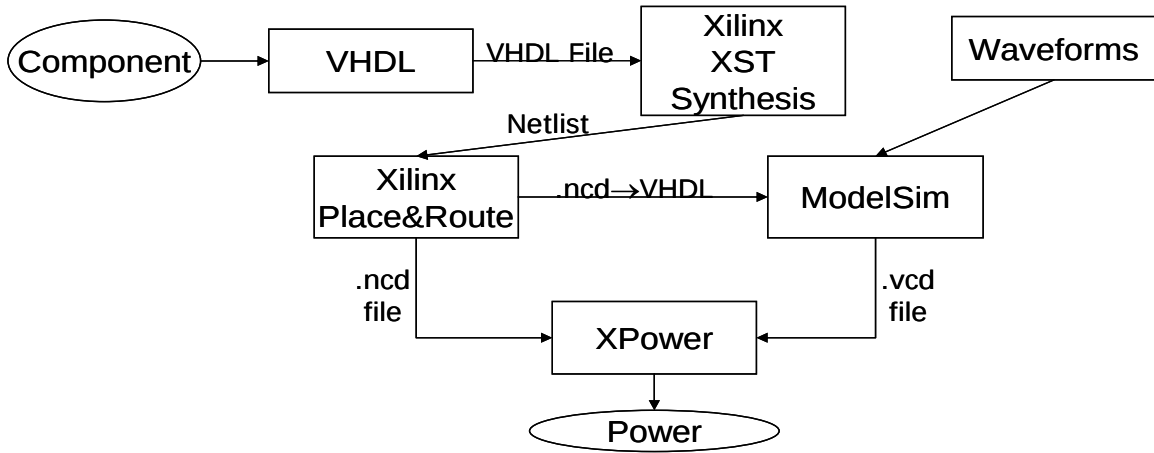


Figure 4: Component power dissipation characterization

From the CPS matrix and power dissipations of the components, it is then possible to create a function whose result is the energy dissipation of the kernel. The constants and coefficients in this function are the power dissipations. The arguments to the function are those parameters in the design that will affect its energy and/or power dissipation. For example, with a linear array architecture, the number of PEs influences both power dissipation and latency.

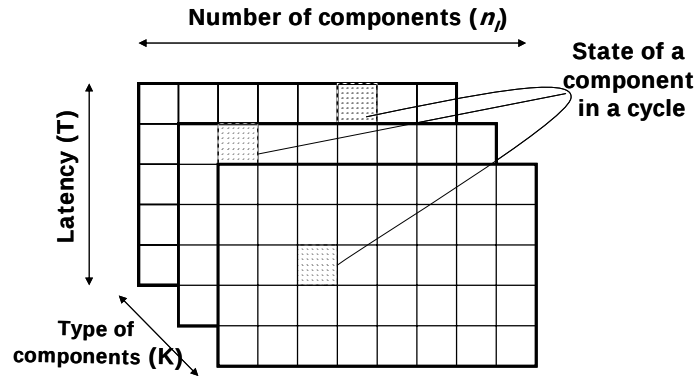


Figure 5: The CPS matrix

3. Validation

When using high-level estimation, it is important that the estimates are accurate. If high-level estimates are not accurate, design decisions based on these estimates may in fact miss designs that are actually efficient. It is also important that the estimates for each type of device are accurate so that valid conclusions can be reached from their comparison.

3.1 Validating Domain-Specific Models

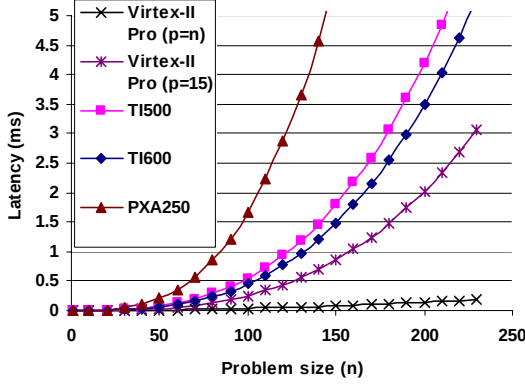
The estimates from a domain-specific model are validated through low-level simulation. The procedure for validation is almost the same as that for low-level simulation of components described above and pictured in figure 4. The only difference is that here, instead of simulation of components, the entire design is simulated. The energy dissipation of the algorithm and architecture is computed by multiplying the power dissipation found through low-level simulation by the latency. If the values differ significantly, there is a problem in the model. Possibly, the model is too abstract. In this case, it needs to be refined before it can be used to investigate design tradeoffs. In our experience, the error in a domain-specific model is usually about 10%. See table 1 for the error between the high-level estimates and the low-level simulation results for computing the FFT (a widely used signal processing kernel) with a Xilinx Virtex-II.

Table 1 Error in domain-specific modeling for FFT implemented on the Xilinx Virtex-II

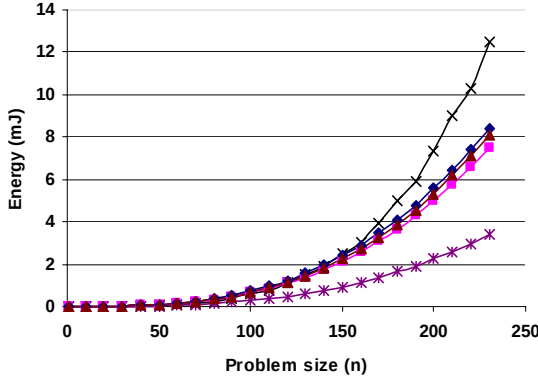
Problem Size	Estimated Energy Dissipation (nJ)	Measure Energy Dissipation (nJ)	Error
16	65.4	77.0	15%
64	403.2	400.4	1%
256	2203.2	1971.3	12%
1024	14963.5	13739.4	9%

3.2 Validation of Device Comparisons

In this subsection, we investigate the validation of results of comparing FPGA, DSP, and embedded processor implementations. We choose matrix multiplication as an example because it is a well-understood problem. We perform high-level estimation for matrix multiplication implementations on the Texas Instruments TMS3206415 DSP, Intel PXA250 embedded processor, and Xilinx Virtex-II Pro FPGA. The latency and energy results for this comparison are shown in figure 6. For the TI DSP, we have used the equation provided by TI for its optimized matrix multiplication library function to determine the latency. For the Virtex-II, we have used the domain-specific modeling procedure described above. For the PXA250, we use the facts that there are n^3 MAC operations in $n \times n$ matrix multiplication and that there is a dual-MAC instruction in the PXA250 to estimate that a lower bound for the latency in computing the matrix multiplication is $n^3/2$. Notice in figure 6 (a) that the latency estimate for the PXA250 implementation is higher than those for the other devices. Because this curve is only a lower bound, it is valid to conclude that the actual implementation on a PXA250 will have a longer latency than the implementations on the other devices. Thus, if latency is the main concern, the PXA250 should not be chosen for implementing the kernel.



(a)



(b)

Figure 6: Latency and energy estimates for matrix multiplication as a function of problem size

4. Case Study: Demean for the PARIS application

Demean is the first kernel in the PARIS application. It computes the mean over a window of pixels in the raw image, removes that mean from every pixel in the window, and stores the result. We now show the high-level performance estimation for the demean algorithm provided by Raytheon. First, we show the estimation for programmable processors (one embedded processor, one DSP) and then we show estimation for one FPGA. In each case, estimates are for demeaning $160 \text{ pixel} \times 120 \text{ pixel}$ images with a demean window size of $5 \text{ pixels} \times 5 \text{ pixels}$

4.1 Programmable Processor Implementations

To estimate the latency, operation counts provided by Raytheon are used.² These are shown in table 2. As mentioned above, the PXA250 has a dual-MAC instruction. However, none of the instructions in demean are MACs, so this feature is not utilized. Thus, we estimate that the PXA250 will be able to compute one operation per cycle. Operating at a frequency of 398.2 MHz, we assume, then, that it is able perform 398.2

² These operation counts are for the baseline, unoptimized version of the demean algorithm

million operations per second. Dividing the total number of operations by 398.2 million gives a latency of 526 μ s.

Table 2: operation counts for demeaning a 160×120 image

Operation	Count
Add	87210
Subtract	104652
Divide	17442
Total	209304

According to the PXA250 datasheet provided by Intel, the PXA250 dissipates 950 mW of power at 398.2 MHz. Multiplying 950 mW with 526 μ s gives the estimate that the PXA250 will dissipate 499 μ J of energy when computing the demean. As mentioned above, this is a lower bound because it considers only mathematical operations and assumes that all of them can be done in one clock cycle. Also, note that this estimate is for fixed point calculations; the PXA250 does not include a floating point unit.

For the Analog Devices TigerSharc, the operation counts provided by Raytheon are again employed in the latency calculations. However, in this case there is data that describes the number of operations that the device can execute per second. The vendor provides the peak MFLOPS for the device: 1800. Unfortunately, Analog Devices does not provide the sustained MFLOPS, so it is estimated to be 80% of the peak MFLOPS which turns out to be 1440 MFLOPS. Dividing the total number of operation by the MFLOPS gives the estimated latency in seconds for computing the demean using Raytheon’s algorithm implemented on the TigerSharc DSP. For a 160×120 image this value is 145 μ s.

The TigerSharc datasheet categorizes the DSP’s power dissipation into several activity levels. The “typical” (second-highest) level is chosen for the estimates because we assume the device is doing a high amount of computation, but not the maximum. This power dissipation is 1.8 W. Multiplying 1.8 W by 145 μ s gives an energy dissipation of 264 μ J. Note that these are estimates for floating point operations and that these estimates are, like those for the PXA250, lower bounds.

4.2 FPGA Implementation

In this case study, estimating the performance of the demean algorithm implemented on a Xilinx Virtex-II FPGA is examined. We again focus on a 160×120 image. The architecture uses 16-bit fixed point computation and the FPGA’s clock frequency is an achievable 75 MHz. The domain consists of the algorithm for demean that Raytheon has provided and a serial, single processing element architecture. The architecture is shown in figure 7. For the purpose of illustration and brevity, we only consider the datapath and ignore the control logic. Additionally, we will only consider the RModules but not the Interconnect as all connections are local and past experience dictates that their power dissipations will be negligible compared that of the logic.

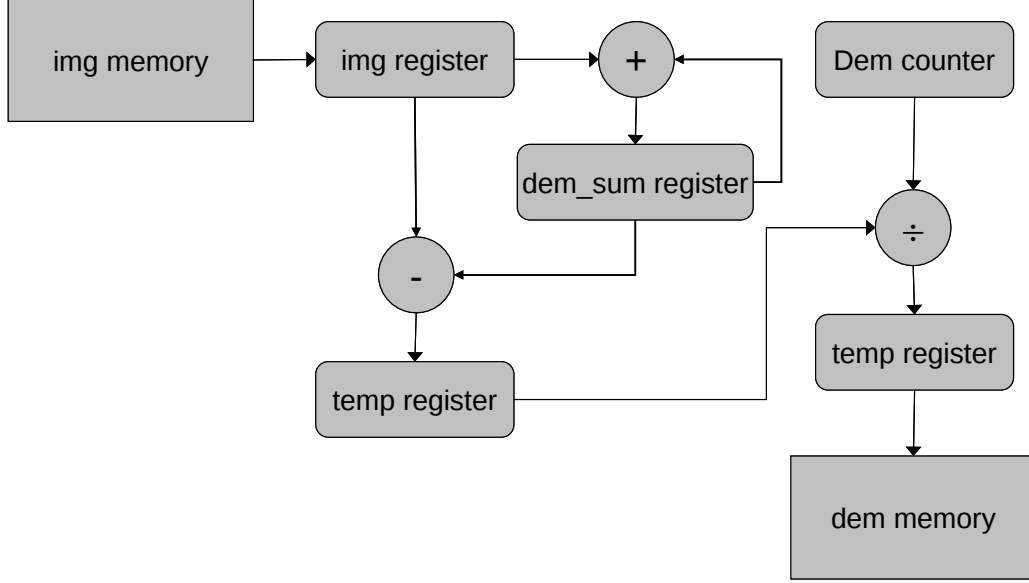


Figure 7: Demean datapath

In this design, the RModules are on-chip memory (img memory, dem memory), registers, adders (the counter is modeled as an adder and a register), a subtracter, and a divider. Because the target FPGA does not have any low-power features like dynamic voltage or frequency scaling, there are only two power states for each component: on and off, corresponding to the activity of the component. When the component is not active, it is assumed to not dissipate any power. The power dissipation for each component when it is on is given in table 3.

Based on the operation counts provided by Raytheon, the CPS matrix is created. From that matrix, we find that each type of component is active for the number of cycles given in table 4. Converting these values to seconds and using the power dissipations, the equation for energy dissipation can be derived. In this example, we keep all parameters such as precision and number of processing elements constant so that the energy dissipation is the sum over each component type c of the product of the power dissipation of c and the amount of time that c is active. Thus, the energy dissipation is estimated to be 50.6 μJ .

5. System-Wide High-Level Performance Estimation

Once performance has been estimated for each of the kernels and the mapping model populated, it is necessary to estimate the performance of the complete system. In MILAN, this is done with the High-level Performance Estimator (HiPerE).

The PARIS application has several parameters whose effect on energy performance needs to be evaluated. Among these are image size, image rate, covariance matrix size, point spread function size, and rate of covariance matrix computation. HiPerE can be run on designs with different combinations of values for these parameters. For each combination, HiPerE outputs an activity report. This report details such values as total latency and energy, energies from different kernels, and device activity. Part of an

activity report for the PARIS application is shown in figure 8. In this case, the application is implemented on a Virtex-II FPGA.

Table 3: Power dissipation of the components in the demean architecture

Component	Power Dissipation (mW)
On-chip memory	12.15
Register	2.12
Adder	2.77
Subtractor	2.77
Divider	30.11

Table 4: Number of cycles for which each type of component is active

Component	Number of active cycles
On-chip memory	226746
Register	104652
Adder	87210
Subtractor	104652
Divider	17442

The screenshot shows the MILAN+PARIS Demo software interface. The top pane displays a table of candidate designs with columns for component names, latency, energy, and area. The bottom pane shows an activity report for the highlighted design, including task details, state transitions, and execution times.

MILAN+PARIS Demo -- PARIS-FAST_FASTRoot120x160DSE1_back.xml					
File Display Execute Help					
	psf	whiten	Latency	Energy	Area
ON_SHARC, velFilter_IN_CFloat	psf_ON_SHARC, psf_IN_CFloat	whiten_ON_SHARC, whiten_IN_CFloat	159612.0	74452.0	0
ON_PXA, velFilter_IN_F398	psf_ON_ProASIC, psf_IN_CFixed	whiten_ON_PXA, whiten_IN_F398	194757.0	99518.0	0
ON_PXA, velFilter_IN_F398	psf_ON_PXA, psf_IN_F398	whiten_ON_PXA, whiten_IN_F398	194757.0	99518.0	0
ON_PXA, velFilter_IN_F398	psf_ON_ProASIC, psf_IN_CFixed	whiten_ON_PXA, whiten_IN_F398	195976.0	99468.0	0
ON_PXA, velFilter_IN_F398	psf_ON_PXA, psf_IN_F398	whiten_ON_PXA, whiten_IN_F398	195976.0	99468.0	0
ON_ProASIC, velFilter_IN_CFixed	psf_ON_ProASIC, psf_IN_CFixed	whiten_ON_PXA, whiten_IN_F398	198996.0	100717.0	0
ON_ProASIC, velFilter_IN_CFixed	psf_ON_PXA, psf_IN_F398	whiten_ON_PXA, whiten_IN_F398	198996.0	100717.0	0
ON_ProASIC, velFilter_IN_CFixed	psf_ON_ProASIC, psf_IN_CFixed	whiten_ON_PXA, whiten_IN_F398	200215.0	100667.0	0
ON_ProASIC, velFilter_IN_CFixed	psf_ON_PXA, psf_IN_F398	whiten_ON_PXA, whiten_IN_F398	200215.0	100667.0	0
ON_PXA, velFilter_IN_F398	psf_ON_ProASIC, psf_IN_CFixed	whiten_ON_ProASIC, whiten_IN_CFixed	259743.0	92482.0	0
ON_PXA, velFilter_IN_F398	psf_ON_PXA, psf_IN_F398	whiten_ON_ProASIC, whiten_IN_CFixed	259743.0	92482.0	0
ON_Virtex, velFilter_IN_CFixed	psf_ON_Virtex, psf_IN_CFixed	whiten_ON_Virtex, whiten_IN_CFixed	315434.0	64267.0	0

Device: Virtex											
Task Details		Total		State Transition		Idle (before)		Execution		Start	End
Task Name	State	Time	Energy	Time	Energy	Time	Energy	Time	Energy	Time	Time
psf_IN_CFixed	Config1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
dmean_IN_CFixed	Config1	465.0	51.0	0.0	0.0	150000.0	0.0	465.0	51.0	150000.0	150465.0
comp_cvm_IN_CFixed	Config1	69994.0	2703.0	0.0	0.0	0.0	0.0	69994.0	2703.0	150465.0	220459.0
inverse_IN_CFixed	Config1	69994.0	2703.0	0.0	0.0	0.0	0.0	69994.0	2703.0	220459.0	290453.0
whiten_IN_CFixed	Config1	22226.0	1063.0	0.0	0.0	0.0	0.0	22226.0	1063.0	290453.0	312679.0
velFilter_IN_CFixed	Config1	2755.0	747.0	0.0	0.0	0.0	0.0	2755.0	747.0	312679.0	315434.0

Performance related to execution

Figure 8: The top pane shows several candidate designs. The latency and energy values were derived using HiPerE. The activity report for the highlighted design is shown in the bottom pane.

6. Conclusion

Because it is infeasible to perform low-level simulations on all possible designs in a design space as large as that for the PARIS application, it is important to be able to estimate the performance of various designs at a high-level. This allows the designer to focus on a few promising designs. To facilitate high-level estimation, we have developed techniques both for kernel and complete system designs. For kernels, we are able to estimate performance for implementations on DSPs, embedded processors, and FPGAs.

Appendix F: PARIS Hardware Module Design

TABLE OF CONTENTS

1 Introduction to the PARIS Module.	68
1.1 Key Features	68
1.2 Functional Overviews	69
1.3 Memory Map	69
1.4 Configuration Settings	71
1.5 Power Supply	71
1.6 Basic PARIS Stackup.	71
1.7 Other Typical Stackup.	72
2 Board Component Specification	73
2.1 The Cygnal C805	73
2.2 TMS320C6713BGDP225	74
2.3 McBSP0	74
2.4 McBSP1	74
2.5 SDRAM Interface	75
2.6 Flash ROM Interface	75
2.7 LED's and Configuration Resistors	75
3 Physical Specifications	75
3.1 Board Layout	75
3.2 Connectors	77
3.3 Phantom Connectors	80
A Schematics	81
B Mechanical Information	85
C PARIS Power Modes	87
D PARIS Power States	88
E Breakout Board Rework	89

- DSP McBSP1 emulated I2C, SPI, UART (mutually exclusive).
- JTAG emulation for both CPU and DSP
- Access high speed video frames via LVDS connector
- Integration with other Pasta Modules via 180 pin connectors.
- Store video frames to Pasta Compact Flash, or sent out via wireless
- Single voltage power supply (+3.3V)

1.2 Functional Overview of the PARIS Module.

The PARIS Module includes a smart power controller C8051 that gets +3.3V power from the PASTA IOBoard. The C8051 uses the Power Aware Operating Systems (PALOS) to control two QFET'S that supply +3.3V to the DSP C6713B and the LVDS aSi Camera. It also controls passive FET switches to isolate or connect board's active signals from the PASTA stack.

The C8051 can download DSP boot code and upload DSP tracking information via HPI interface and then sending tracking information wirelessly through the PASTA Mote radio. The C8051 can also communicate with the DSP via IIC bus.

The DSP TMS320C6713B interfaces to on-board SDRAM and FLASH though a 32- bit external memory interface (EMIF). The DSP McBSP0 interfaces to the aSi Camera through a LVDS cable. DSP McBSP1 is used to emulate DSP_I2C, or DSP_SPI, or DSP_UART for future development.

The PARIS Module has 8 LED's and 5 configuration resistors that can be used to provide the user with simple interactive feedback.

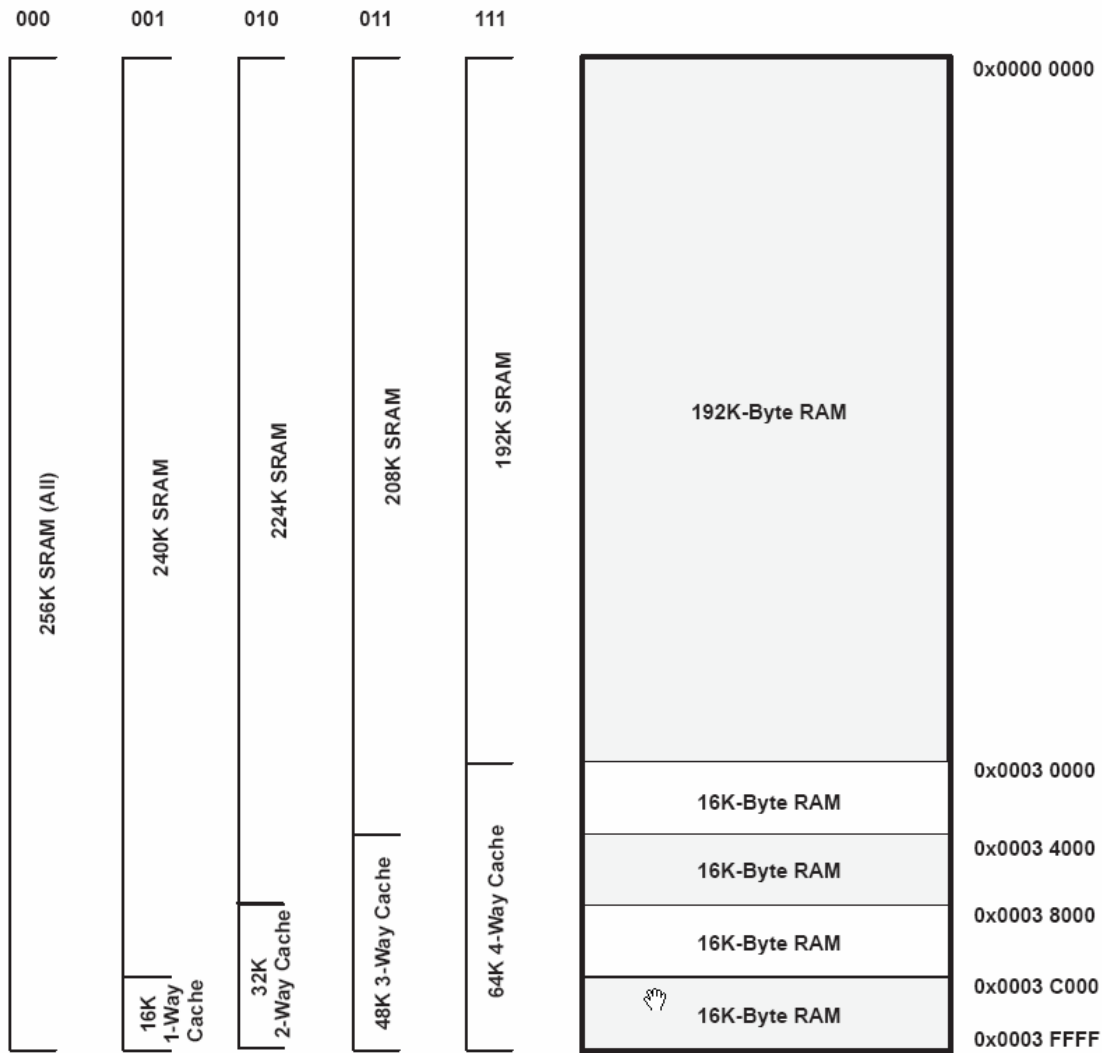
On-board switching voltage regulators provide the +1.26V DSP core voltage. The DSP is held in reset by the C8051 until the power supplies are within operating specifications will be shut down if the current exceed a certain limit.

Ti Code Composer and Cygnal Code Developer communicate with the DSP and the CPU through the external JTAG connectors on a special built breakout board.

1.3 Memory Maps

The CPU C8051 only uses 8K internal data RAM and 128 Kbytes built in FLASH. The DSP TMS320C6713B itself provides up to 256 Kbytes internal memory, usable program and/or data memory. 64 Kbytes of this memory area can be configured as a 4-way second level cache to increase throughput if large data structures or program code reside in external memory.

32 Mbytes on-board Mobile SDRAM is used for large data buffers as commonly required by image processing applications.



512 Kbytes Flash Memory is used for nonvolatile data and program storage. The EMIF (external memory interface) has 4 separate addressable regions called chip enable spaces (CE0-CE3). The SDRAM occupies CE0 while the Flash occupies CE1. CE2 and CE3 are reserved.

Address	C6713 Memory Type	C6713 PARIS
0x00000000-0x0002FFFF	Internal Memory	192K Internal Memory
0x00030000-0x0003FFFF	64K 4-Way Cache	64K 4-Way Cache
0x80000000-0x81FFFFFF	EMIF CE0	32 M SDRAM
0x90000000-0x90080000	EMIF CE1	512K FLASH
0xA0000000	EMIF CE2	Reserved
0xB0000000	EMIF CE3	Reserved

1.4 Configuration Resistor Settings

The PARIS Module has 5 configuration resistors. Under default operation, configuration resistors are not needed and are not installed. Configuration mode is determined when DSP is released from reset.

- R30 control DSP JTAG modes
 - o Normal operation mode (not installed, default)
 - o Boundary scan mode (installed)
- R32 controls DSP endianness
 - o Little endian (not installed, default)
 - o Big endian (installed)
- R34, R33 controls the boot mode that will be used when the DSP starts executing.

R34	R33	Boot Mode
Not installed	Not installed	8 bit external Flash (default)
Installed	Installed	16 bit external Flash
Installed	Not installed	32 bit external Flash
Not installed	Installed	32 bit HPI boot

1.5 Power Supply

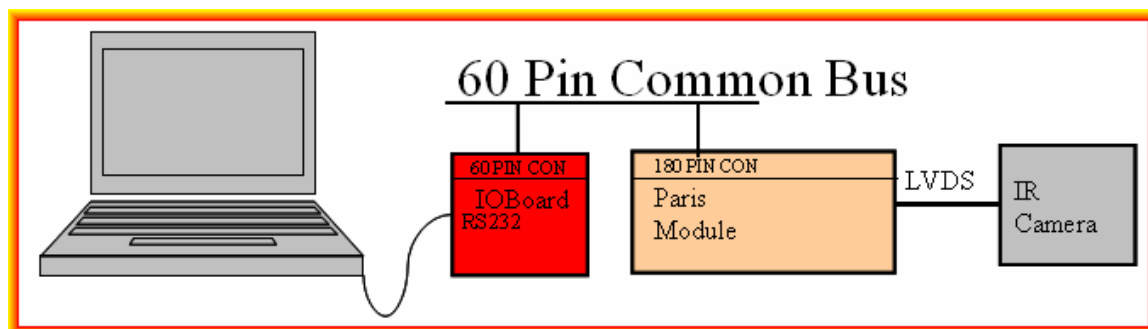
The PARIS Module requires a +3.3V single voltage power supply only. Secondary 1.26V for DSP core are generated on board using regulators and charge pumps. The Cygnal 8051 microprocessor supplied and monitor the power to the DSP. It also controlled the power sequencing and holds the DSP in reset if the supply voltage is below limit or cut off the power if the supply voltage exceeded a certain limit. There are four power test points on the PARIS Module: TP5: V3 (3.3 V), TP6: V3_Local (3.3 V), TP7: V3_CAM (3.3 V), TP8: V1_26 (1.26 V).

R8 indicates board's power consumption. R10 indicates the DSP core power consumption, and R35 indicates Camera power consumption.

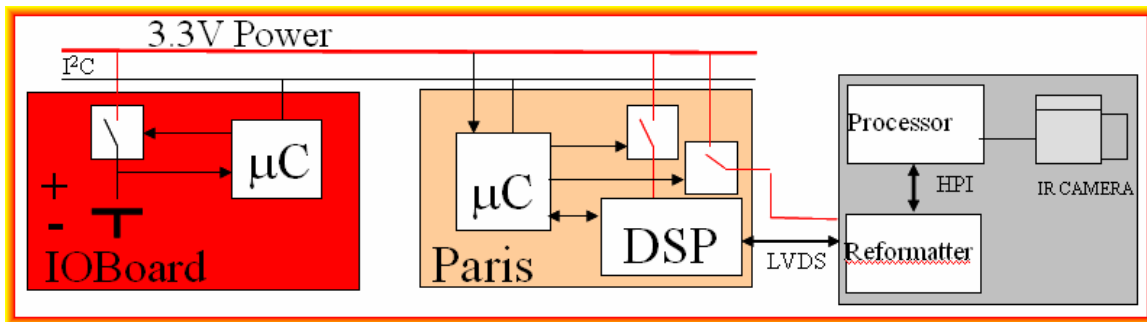
1.6 Basic Paris Module Stack up

The PARIS Module follows the PASTA Module Architecture. It connects to other PASTA Modules by 180 pin connectors in which 60 pins are common to all (pass through)

For development purpose, a PC can also communicate with the PARIS Module through an RS232 connector on the PASTA IOBoard.

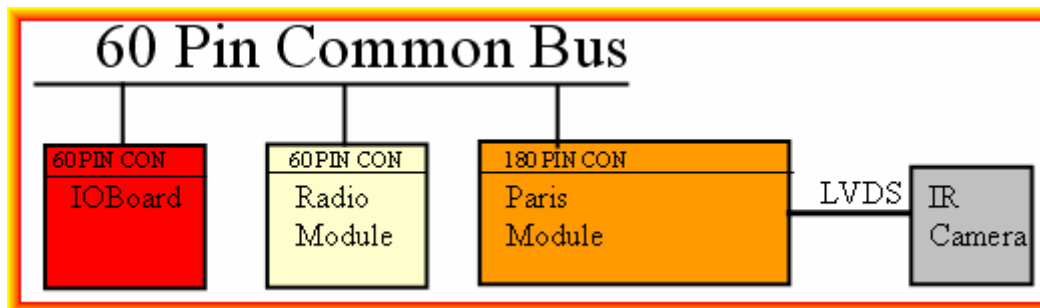


The following picture shows the setup above in more detail:

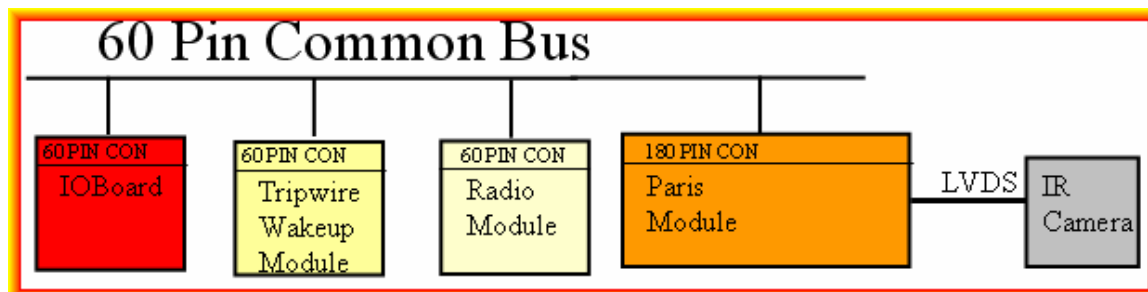


1.7 Other Typical PARIS PASTA Unattended Ground Module Setups:

A simple PARIS Unattended Ground Module includes an IOBoard, a PARIS Module, an IR camera and a Mote radio module. A central (PC) unit with a Mote radio can send commands wirelessly to wake up PARIS Module to capture video, perform image processing and then send tracking information wirelessly back to the central Mote radio. It then sends command to put the setup to sleep to save battery power.



To save power even further, a Tripwire module can be added to the PARIS Unattended Ground Module. The Tripwire module is always on but it only consumes micro watts. When the tripwire detects a certain sound, it will wake up the PARIS Module to send tracking information to the central Mote radio.



The PARIS Unattended Ground Module can be much more sophisticated and powerful when working with the PXA255 processor board. Multiple units can be deployed 100 yards apart across a vast land to form an adhoc monitor network.

2. BOARD COMPONENTS

This section describes the operation of the major components on the PARIS Module.

2.1 *The Cygnal C8051*

In the PARIS Module, the C8051 gets it +3.3V from the IOBoard and supplies the power to the DSP. It periodically communicates with the IOBoard (Power Module) and the DSP via I2C bus.

The C8051 normally operates at 32 KHz; however, it can crank up to 50 MHz to get tracking information from the DSP via the I2C0 bus and send the information out of its UART0 to a PC or PASTA Mote radio.

Because of its limited real estates, the PARIS module must use connectors from other PASTA boards. For example, it borrows the COM connector from the IOBoard and JTAG connectors from the Breakout board.

Breakout board was originally designed to use with the PXA255 processor board, its JTAG connectors are not pin compatible with our Cygnal C8051 and Digital Spectrum JTAG ports and its UART driver could cause contention with the driver on the IOBoard. Two special built JTAG adapters has been requested and it would take some setup time to bring the JTAG interfaces. Since I was tight up with the PARIS Module layout, any help on this task would be highly appreciated.

After the Cygnal C8051 JTAG successfully downloads code to the C8051, the C8051 would request the IOBoard to route its UART0 on BANK2 to connector J9 of the IOBoard so that it could receive commands from a PC. The request includes the following tasks:

1. Set BNK_SW_N2 (P2.2) to low to enable BANK2 connection
2. Set CPU_UART0_SWAP (P1.4) to low to line up UART0 TX and RX (Optional, only perform if needed)
3. Send I2C command to request IOBoard to route its Connector J9 to BANK2

When receive a PC command, i.e. upload tracking information, the C8051 would perform the following tasks:

1. Set V3_LOCAL_EN_N (P3.0) to low (logic 0) to enable +3.3V power to the DSP
2. Wait for the supplied +3.3V to stable before releasing DSP from reset. (using current sense pin18 AIN0)
3. Set V3_CAM_EN_N (P3.1) to low (logic 0) to enable +3.3V power to the Camera.
4. Send I2C commands to the DSP request tracking information (similar to I2C operation in LPRAFF)
5. Upload tracking information and sent it out through UART0 to the PC

The PARIS Module has more control signals to save system power; however, if not use, most could be left alone in its default states. Additional details can be found at the appendices.

The Cygnal C8051 is always on and gets its +3.3V power from the IOBoard. It normally operates at 32 KHz to save power. When needed, it can scale up to 50 MHz to perform demanding data transfer tasks. It constantly communicates with other micro power sensor on other board on the PASTA stack and sometimes with the on board DSP via the common I2C0 bus.

To perform an image processing task, the C8051 would turn on QFET Q1 by pulling down (logic 0) the V3_LOCAL_EN (P2.1) signal to provide power the DSP. Meantime it would wait for the supply voltages to stable before release DSP from reset. If the supply voltage is beyond a certain limit it would put the DSP back to reset or cutoff the power supply.

The stack goal is to reuse the IIC protocol between CPU and DSP in LPRAFF; however, because of power saving purpose, there on 8k on chip RAM is available. It is ok because the Mote radio is slow at 19kbit typical.

2.2 TMS320C6713BGDP225

The processing power of the PARIS module is the advanced VLIW architecture floating point digital signal processor TMS320C6713 from Texas Instruments. This RISC architecture DSP provides 8 instruction units which operate in parallel, yielding a maximum performance of 2400 MIPS, 1800 MFLOPS. Up to 256 Kbytes internal memory and a two level cache architecture (64 Kbytes L2 cache, 4 Kbytes L1 program cache and 4 Kbytes L1 data cache) guarantee the memory bandwidth required to sustain high data throughput. Furthermore, two multi-channel buffered synchronous serial ports, an I2C bus interface, two timers, an enhanced DMA controller and a 16 bit wide Host Interface are built into this processor. The multi-channel buffered serial port McBSP0 provides a direct connection to Raytheon aSi LVDS Camera and the McBSP1 provides emulate I2C, SPI, or UART interfaces. The built-in DMA controller provides enhanced features for 1D and 2D transfers and auto initialization, which allows maintaining circular and ping-pong buffers without any CPU intervention.

The DSP interfaces to on-board peripherals through a 32-bit EMIF (External Memory Interface). The SDRAM, Flash are all connected to the bus. EMIF signals are also connected to the P1 connectors for debugging purpose.

2.3 LVDS Connection to the aSi Camera Using McBSP0

McBSP0 is used to receive video frames from the Lvds aSi camera at 19 MHz.

2.4 DSP'S I2C, SPI, or UART Emulation Using McBSP1

McBSP1 is used to emulate I2C, SPI, or UART for the Ti DSP

2.5 Synchronous DRAM

The DSP uses a 32 Mbytes SDRAM (MT48LCM32LF-B5-8) on the 32-bit EMIF. The SDRAM is mapped at the beginning of CE0 (address 0x80000000). The integrated SDRAM controller is part of the EMIF and must be configured in software for proper operation. The EMIF clock is derived from the PLL settings and should be configured in software at 90MHz. This number is based on an internal PLL clock of 450MHz required to achieve 225 MHz operation with a divisor of two and a 90MHz EMIF clock with a divisor of five.

When using SDRAM, the controller must be set up to refresh one row of the memory array every 15.6 microseconds to maintain data integrity. With a 90MHz EMIF clock, this period is 1400 bus cycles.

2.6 Flash Memory

The DSP uses a 512Kbyte external Flash (AM29LV800BB-70WBI) as a boot option. It is visible at the beginning of CE1 (address 0x90000000). The Flash is wired as a 256K by 16 bit device to support the PARIS Module's 16-bit boot option. However, users can use the Flash as a 8-bit device (ignoring the top 8 bits) to match the 6713's default 8-bit boot mode. In this configuration, only 256Kbytes are readily usable. The Flash Memory is divided in multiple sectors of 64 Kbytes. Each sector can be erased individually and (re-)programmed on a 16 bit word basis. The DSP has direct access to the Flash Memory. Identification, Sector-Erase and Programming is handled by TI Code Composer and Digital Spectrum JTAG emulator.

2.7 LED's and Configuration Resistors

The PARIS Module includes 8 LED's as a simple way to provide the user with interactive feedback. Six are accessed by the CPU and two are accessed by the DSP.

3. PHYSICAL DESCRIPTION

This section describes the physical layout of the PARIS Module and its connectors

3.1 Board Layout

The PARIS Module is a 10 layer board which is power by a single 3.3V source. Figure 3.1 and 3.1.1 shows the layout of the board.

Figure 3.1: TOP VIEW

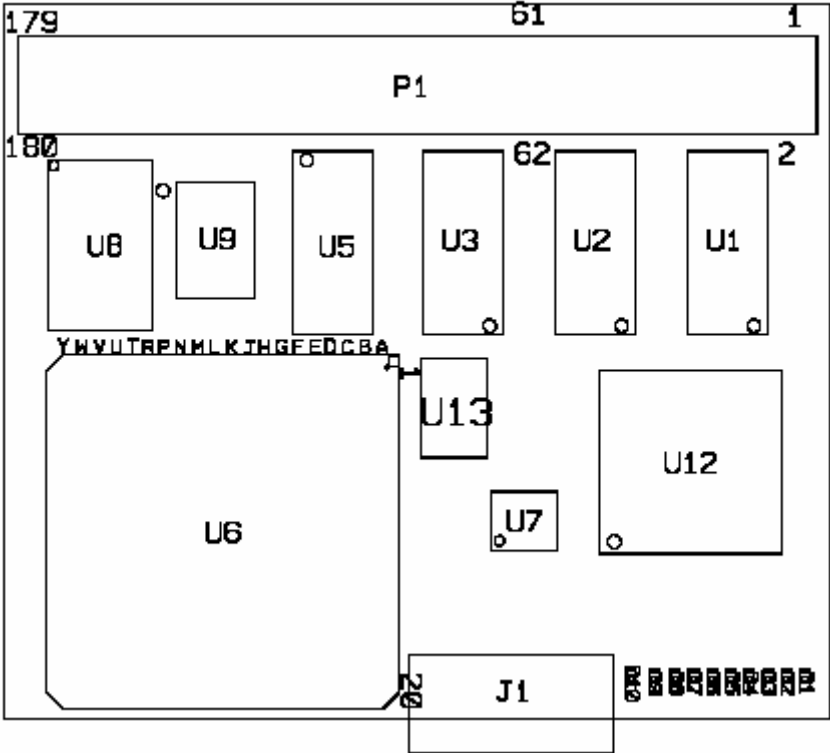
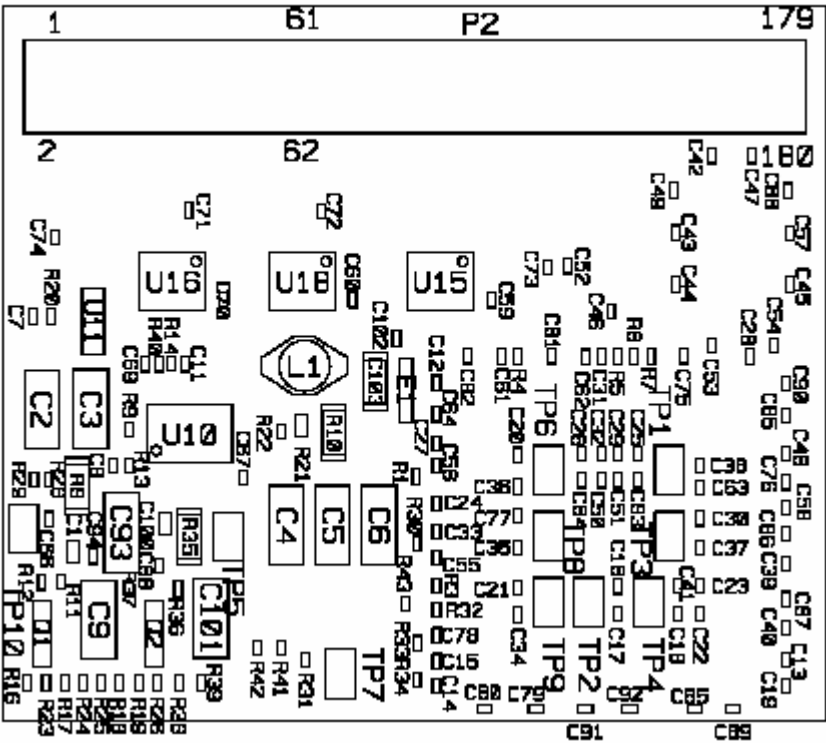


Figure 3.1.1: BOTTOM VIEW



3.2 Connectors

The PARIS Module connects to the Camera by LVDS connector J1 and integrates to the PASTA stack through P1 and P2 connectors.

Connector	Pins	Function
P1	180 Socket	60 pin pass thru, 120 pin PARIS debug
P2	180 Header	60 pin pass thru, 120 pin PXA interface
J1	16 Right Angle	LVDS Connector

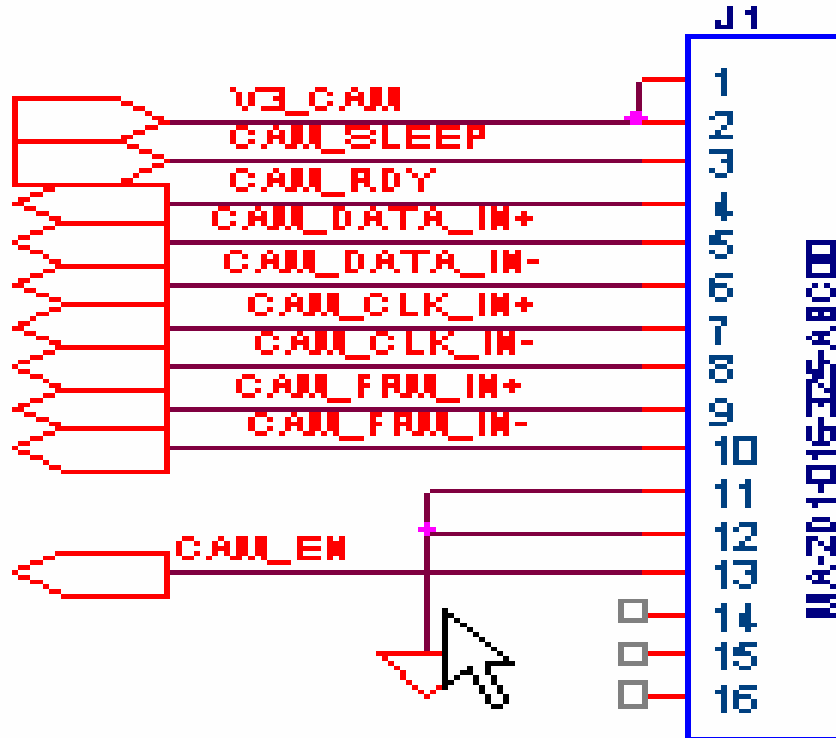


Figure 3.2.1. aSi LVDS Camera Connector (MA-2D1-016-325-ABC00)

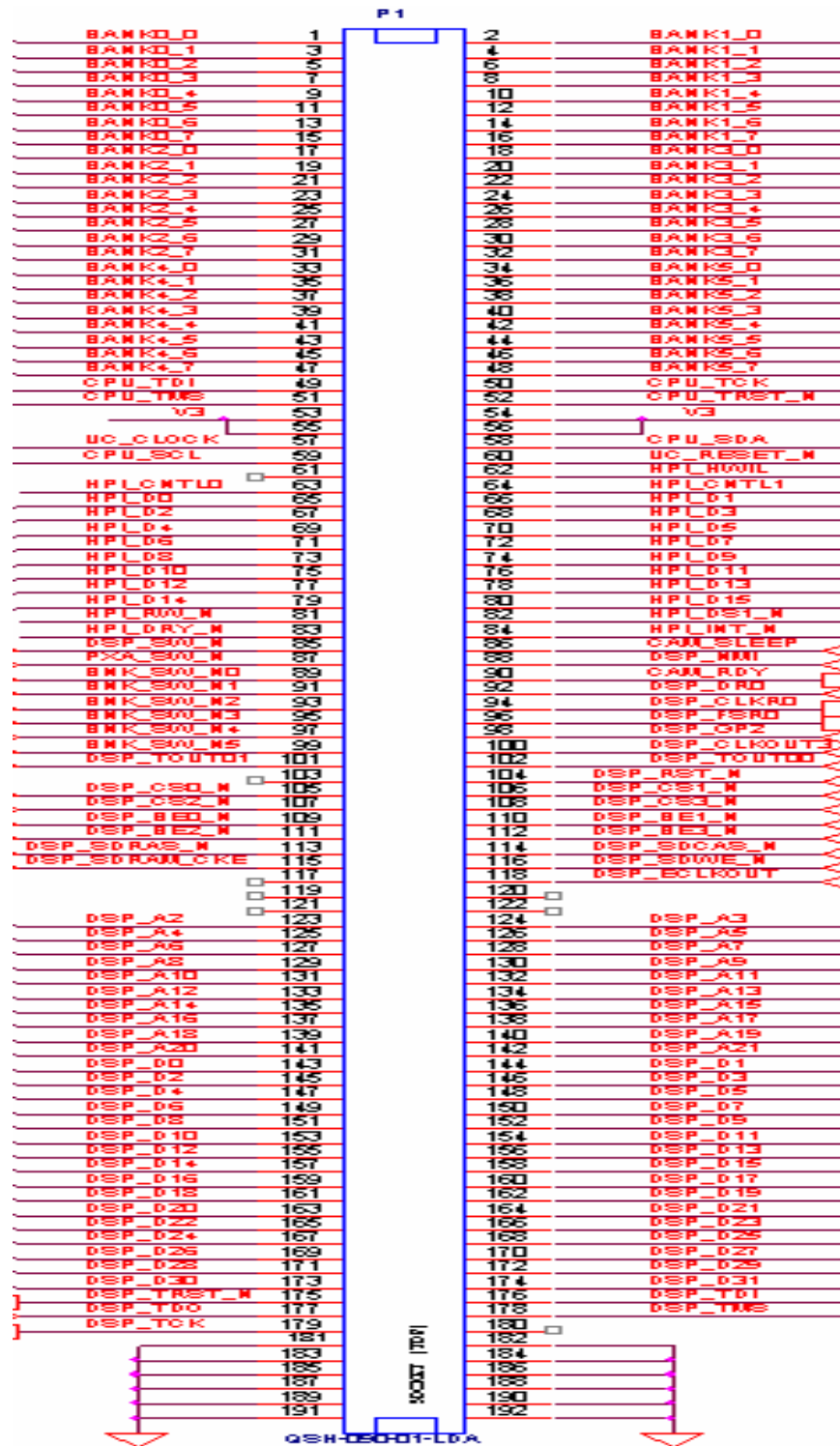


Figure 3.2.2. 180 Pin Socket (SAMTEC_QSH-090-01-DA)

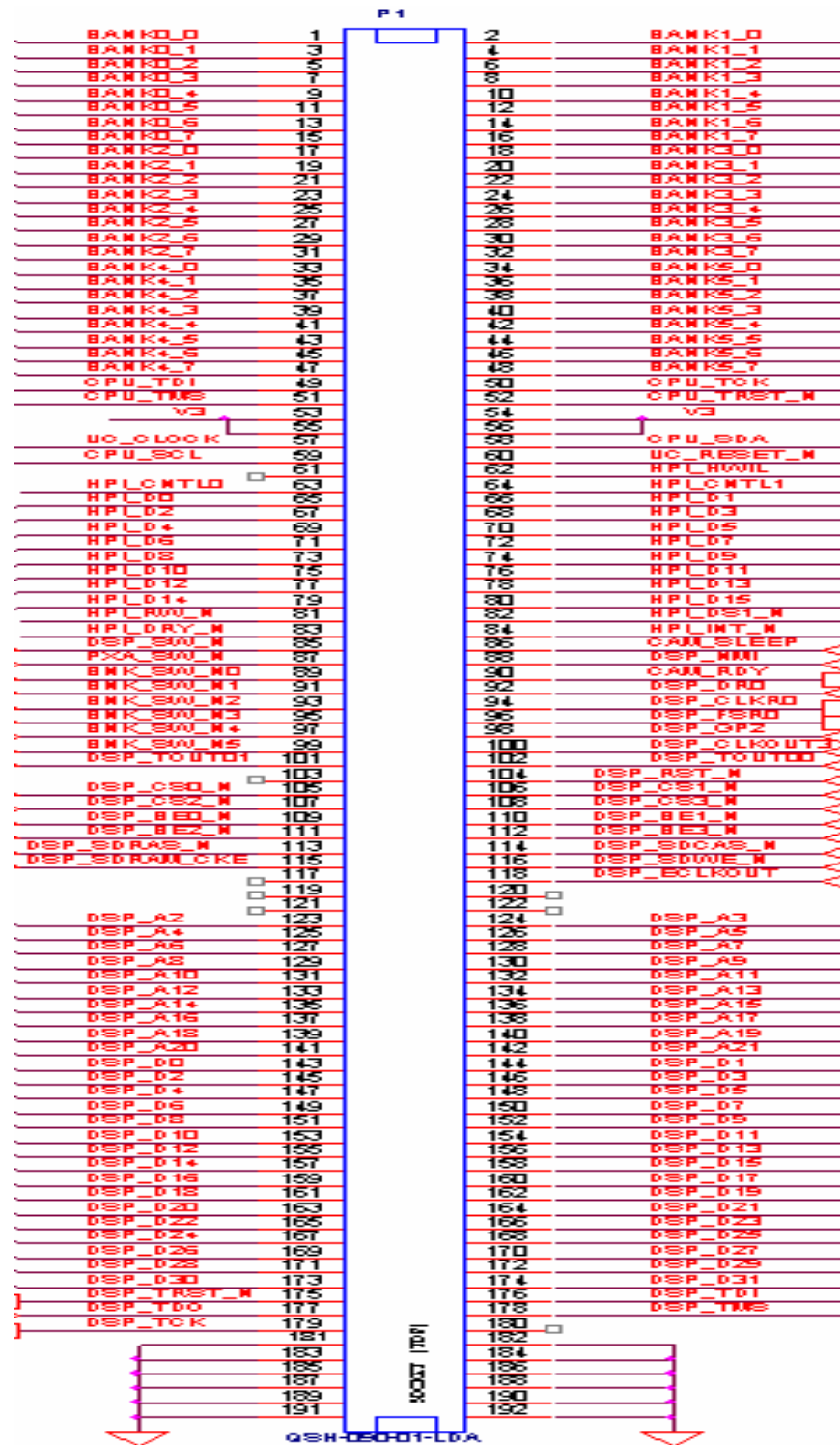


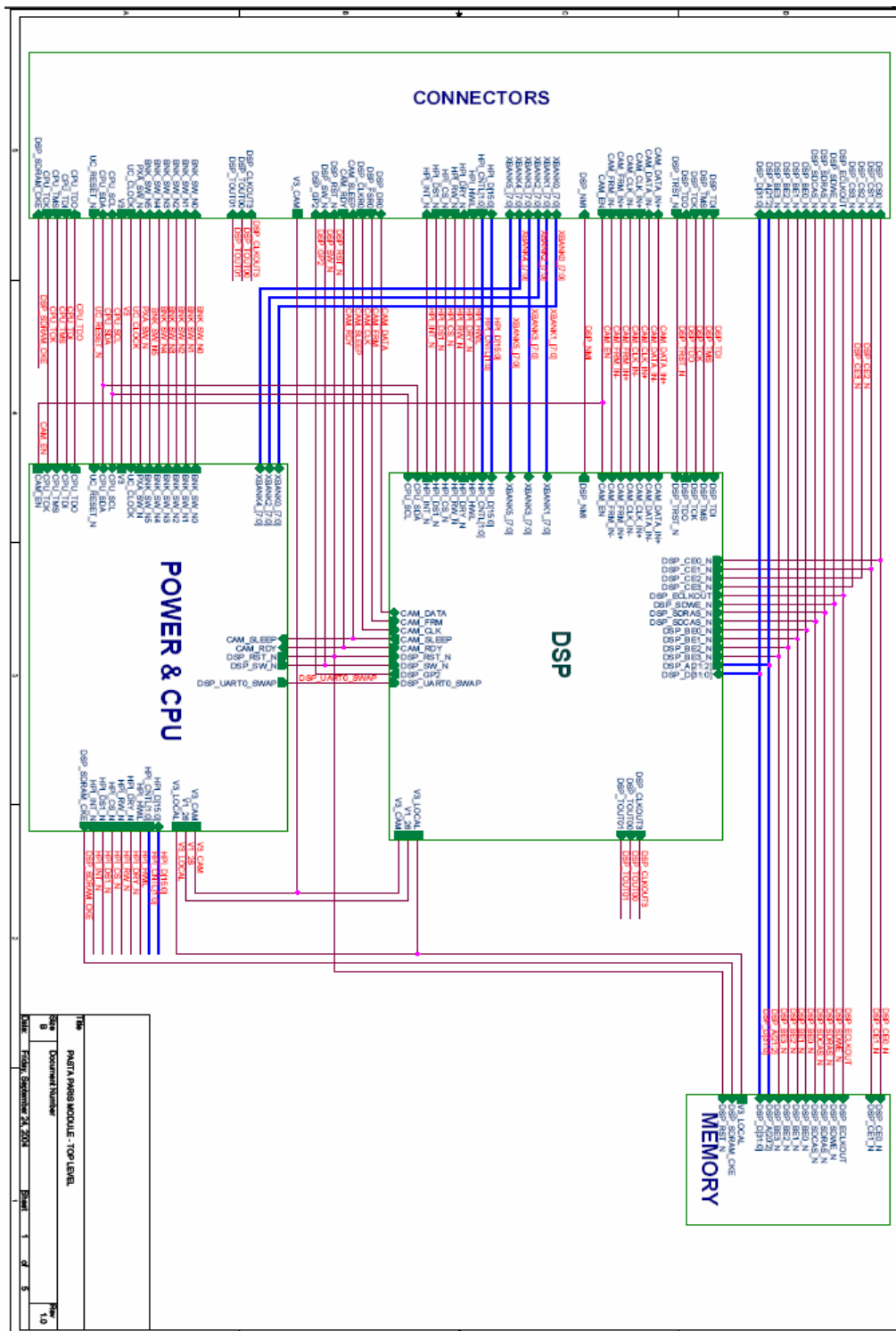
Figure 3.2.3. 180 Pin Header (SAMTEC_QTH-090-01-LDA)

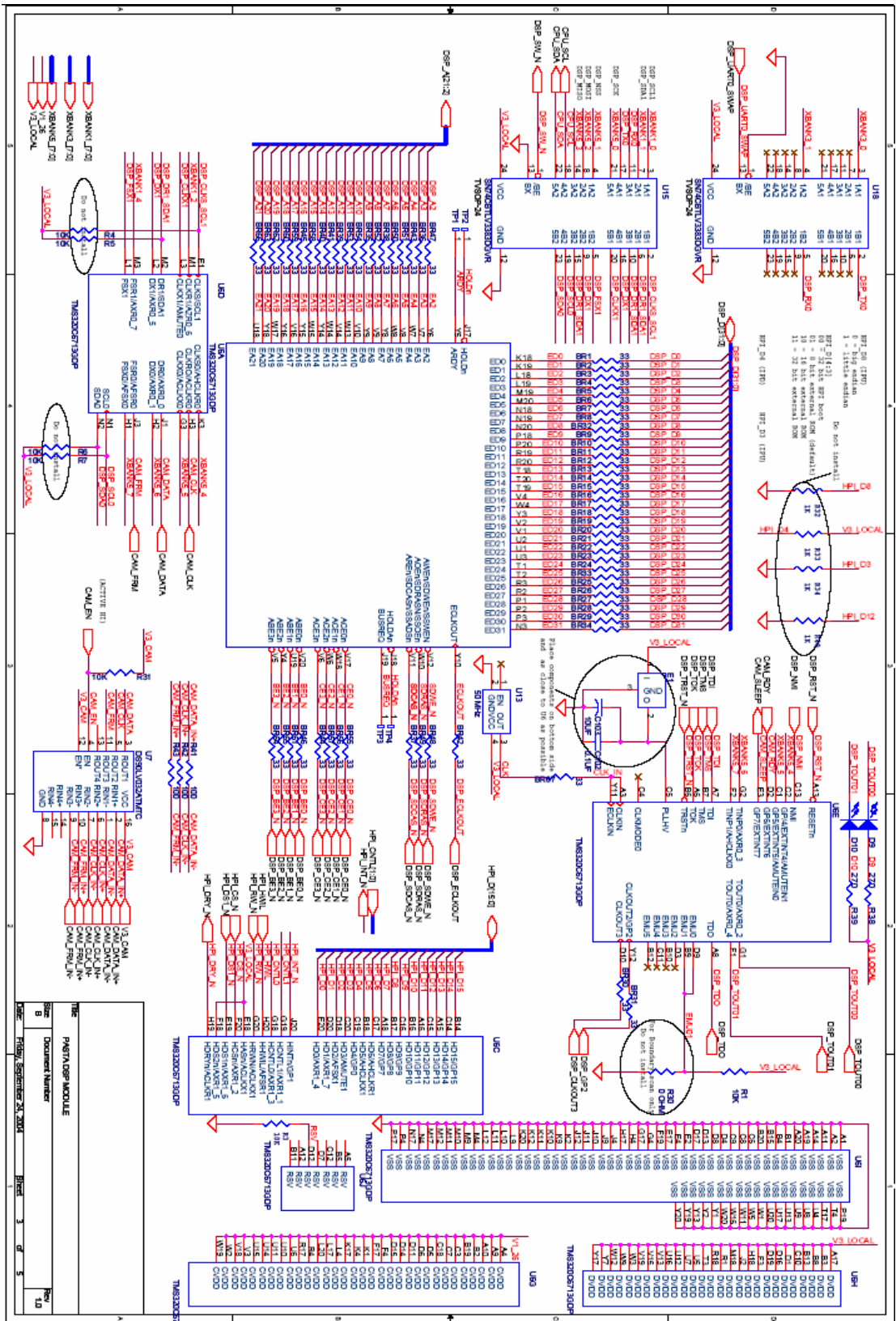
3.3 Phantom Connectors

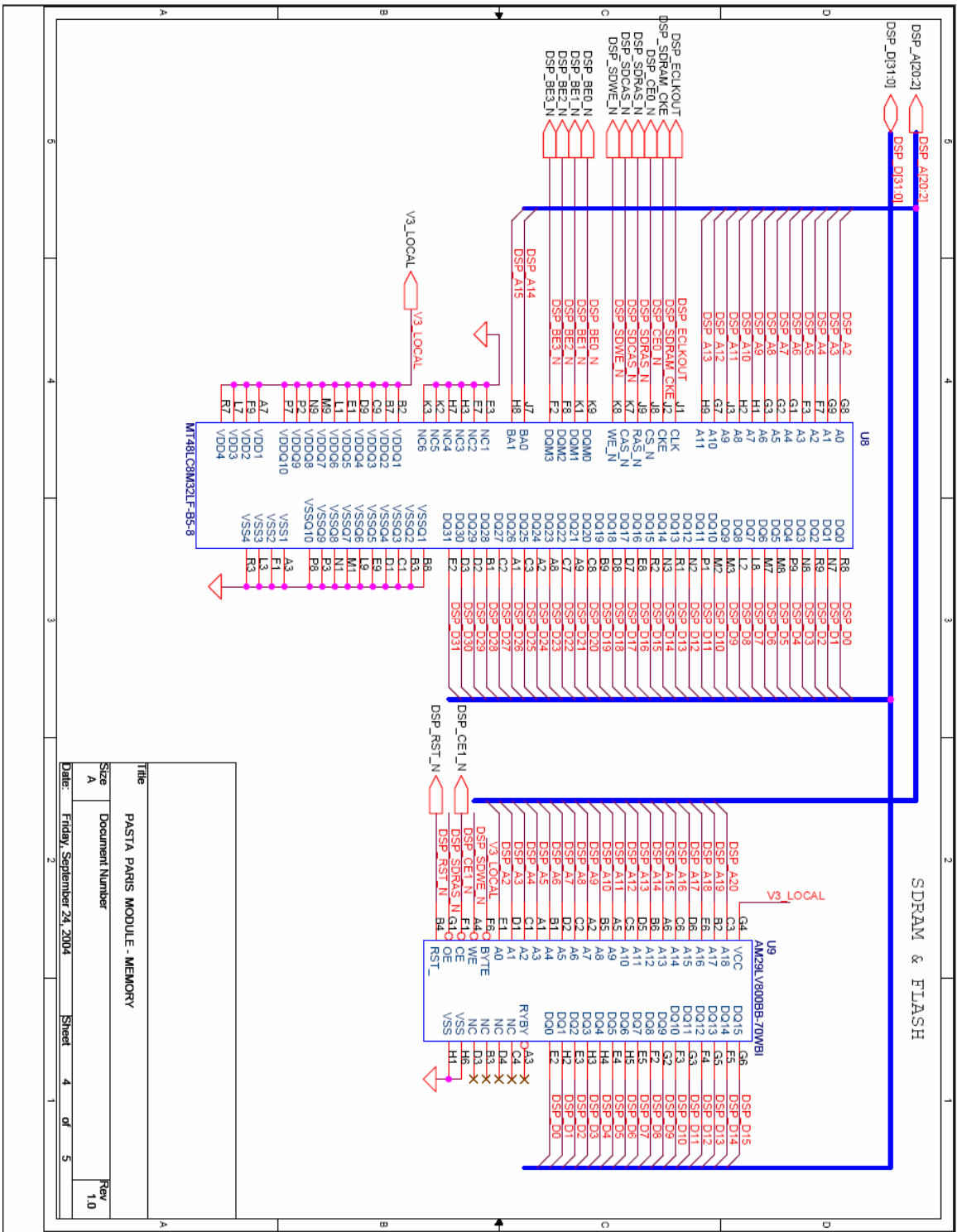
Because of its size, PARIS Module borrows connectors from the PASTA IOBoard for System Reset, CPU and DSP UART, SPI, IIC interfaces, and uses connectors from the PASTA Breakout Board for CPU and DSP JTAG interfaces.

Connector	PASTA Boards	Function on PARIS Board
SW1	IOBoard	Hardware Reset
J9	IOBoard	C8051 UART 0
J13	IOBoard	System Power Supply
J19	Break Out Board	DSP JTAG
J16	Break Out Board	C8051 JTAG
P1	Break Out Board	C8051 UART 0

This appendix contains the schematics for the PARIS Module







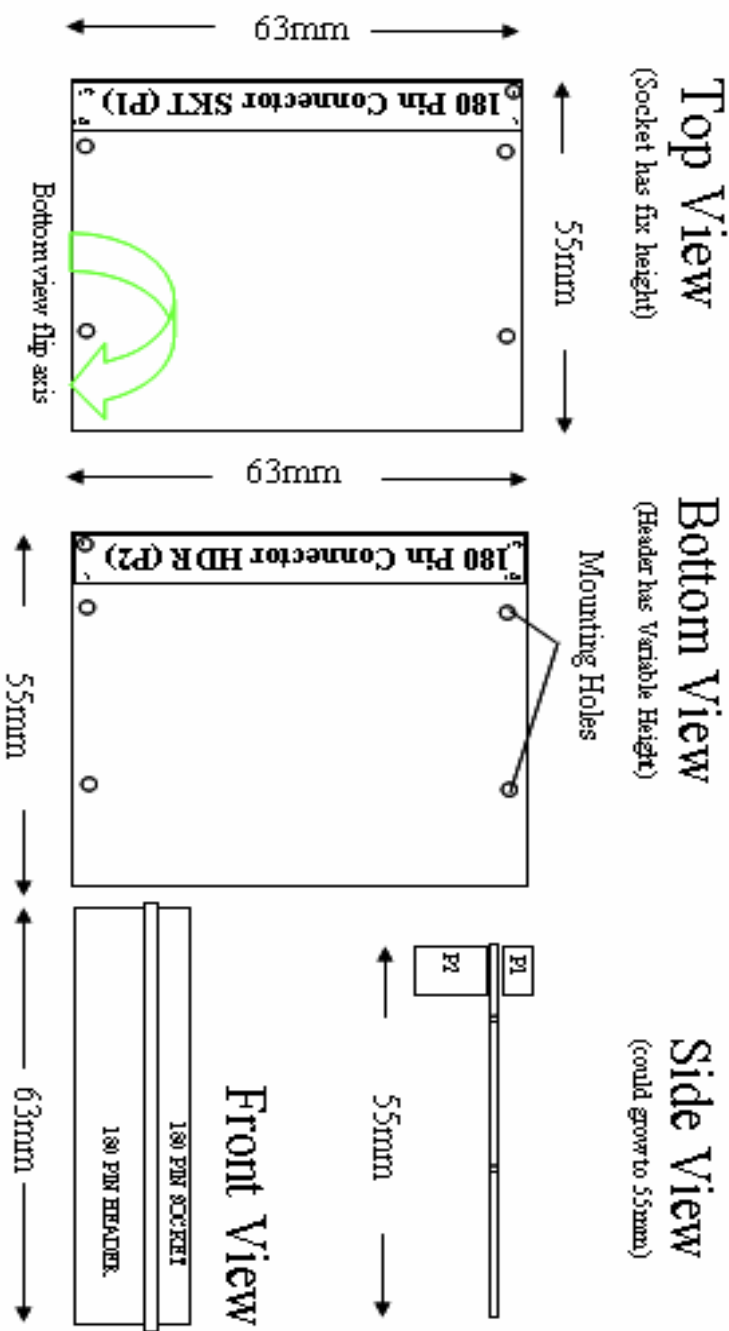


Paris Module Physical layout

Raytheon

Blank PCB (Scale 1:1 on 8.5x11)

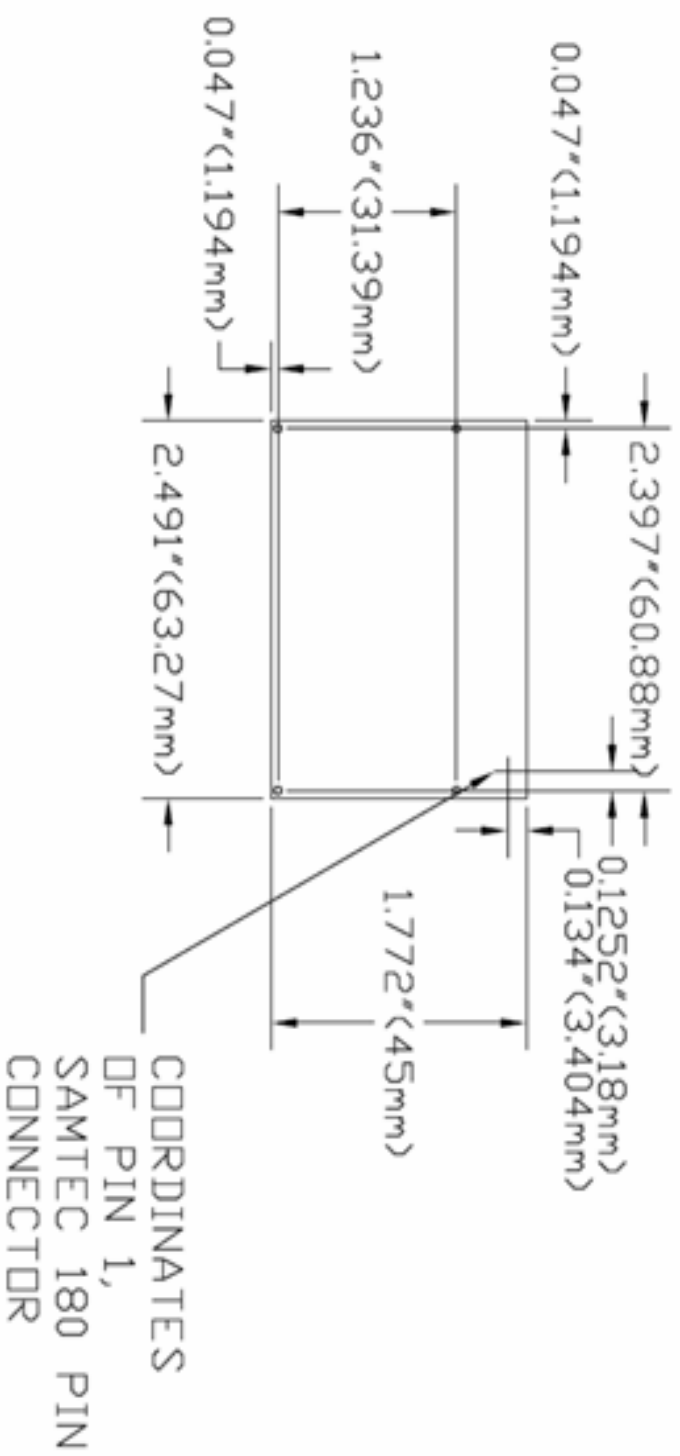
- The Paris Module will plug into an existing Pasta stack via header P1 and socket P2. It have the following dimensions:



APPENDIX B: MECHANICAL INFORMATION

This appendix contains the mechanical information about the PARIS Module

Four Mounting Holes of Paris Module

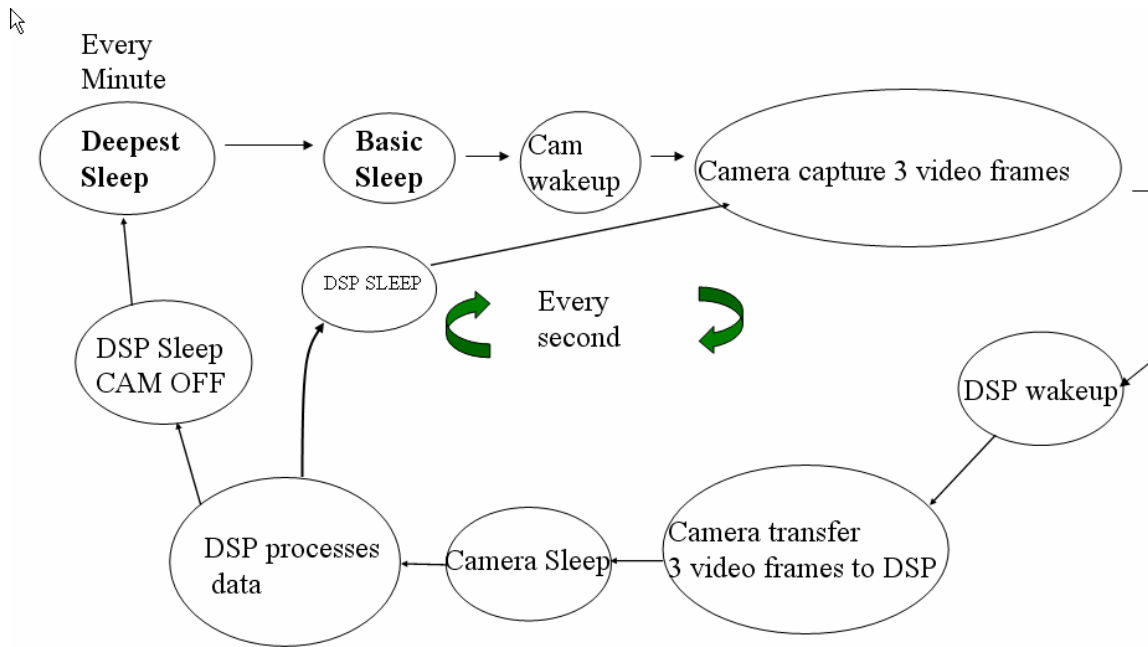


APPENDIX C: PARIS MODULE POWER MODES

This appendix contains the mechanical information about the PARIS Module Power Modes.

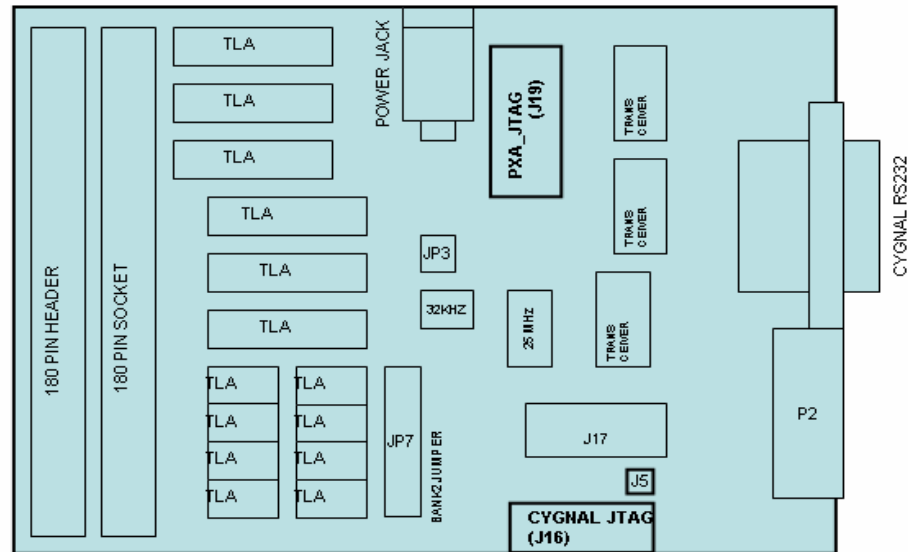
Power Modes	Module States			Description
	Power Module	Paris Module	IR Camera	
Deepest Sleep	Sleep	* * * * Disconnected * * * *		Lowest power
Basic Sleep	Sleep	Off	Off	Turn off every .5 minute
Video Scan	Sleep	Off	On	Camera scan 3 frame for x sec
Data Transfer	Sleep	On	On	Cam transfer data to DSP via LVDS
Process data	Sleep	On	Off	DSP process data
Send result	On	Off	Off	Send result to host

APPENDIX D: PARIS MODULE POWER STATES



APPENDIX E: BREAKOUT BOARD REWORKS

PASTA BREAK OUT MODULE REWORK

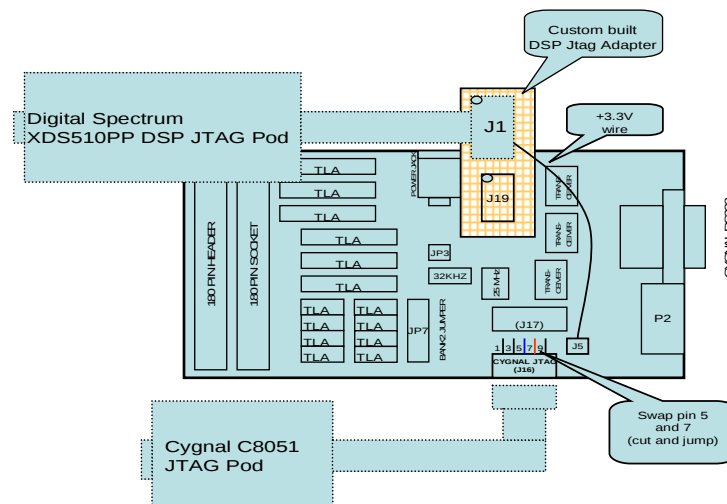


THE BREAK OUT BOARD

The Breakout Board was designed to debug the PASTA PXA255 board. It is re-used to debug the PARIS Module; however, the Jtag connectors are not pin compatible to The Cygnal C8051 and the DSP Digital Spectrum Jtag pods. Some reworks are needed:

- Swap pin 5 and 7 on connector J16 (cut and jump the connector pins)
- Build the DSP Jtag Adapter as instructed below

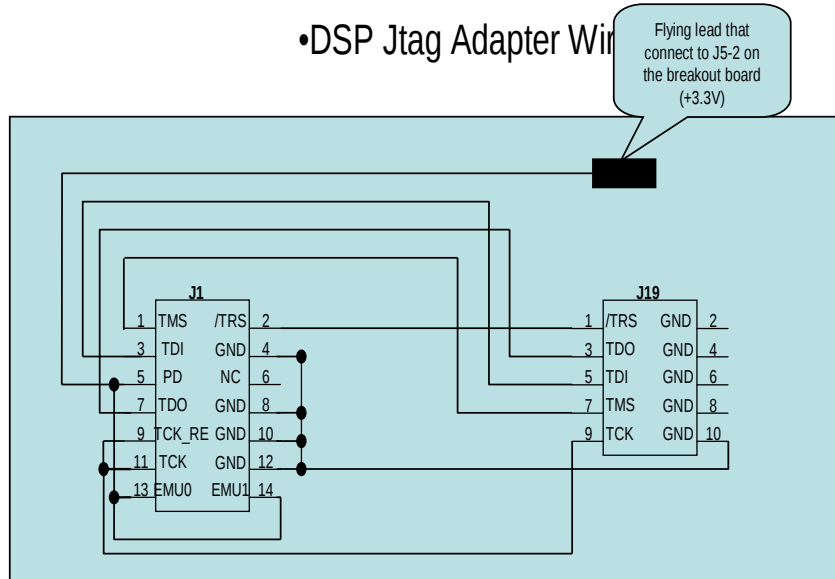
Instruction for rework and building the DSP Jtag adapter



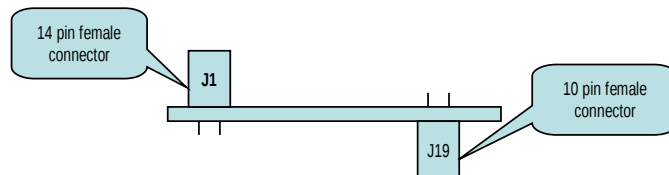
DAVID BUI
310-607-7310

Instruction for building the DSP Jtag adapter

•DSP Jtag Adapter Wiring



•Side View of the DSP Jtag Adapter



Appendix G: Algorithm Design

TABLE OF CONTENTS

1. GENERAL IMAGE PROCESSING STRATEGY AND TIMING	93
2. GENERAL THEORY FOR DETECTION ALGORITHM	93
3. SPECIFICS OF THE FAST IMPLEMENTATION ON PARIS	94
3.1 Data Collection	94
3.2 Demeans	94
3.3 Covariance & Invert	95
3.4 Whiten	96
3.5 Velocity Filter	96
3.6 CFAR Detection	97
4. MULTI OBJECT TRACKER	97
5. SUMMATION	97
6. ACRONYMS	98
7. REFERENCES	98

1. OVERALL IMAGE PROCESSING STRATEGY AND TIMING

The method of visually finding targets in the PARIS system can be described by the block diagram shown in Figure 1. The first step involved in finding targets is to perform data collection. The data collection step collects 3 frames at 20 Hz (from the IR camera mentioned above). The reception of each frame takes 16.7 ms. This results in the entire data collection process taking 116 ms. After all data has been collected the local demean routine removes any net average from the image. Local demean occurs after data collection has completed and requires 70 ms to complete. Once the image has been demeaned the matched filter suppresses noise and emphasizes features that *match* a template. The matched filter step takes 90 ms to complete. The velocity filter looks for one pixel motion of features in any direction. Depending on the amount of motion in the scene execution time is variable. Typically this step can be completed in 35ms for a scene with about 30 detections. This time period can scale up to 50 ms for 200 targets. The clutter adaptive CFAR detector manages a list of how many consecutive times a pixel has a detection. If the pixel has a detection three consecutive time or greater, then that detection is invalidated. This process takes less than 1 ms to execute. The multi-object tracker looks for detections that are moving in a consistent direction across the FOV. When it finds objects that are moving across the field of view consistently it updates its history and provides the tracks as output for the system. The execution time for the multi-object tracker ranges with the number of detection from a low of 3 ms up to 5 ms.

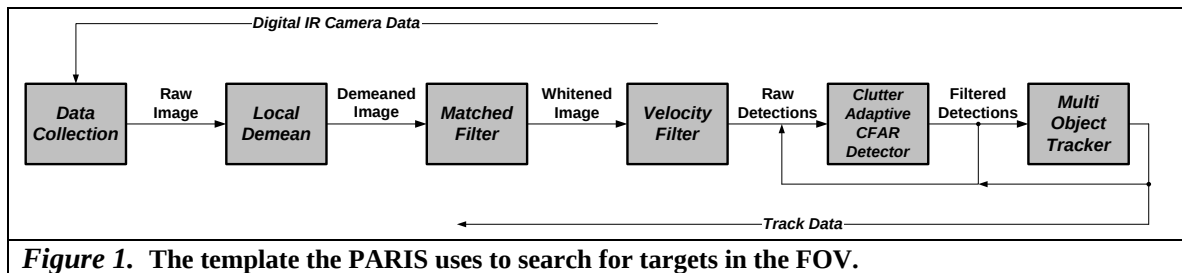


Figure 1. The template the PARIS uses to search for targets in the FOV.

2. GENERAL THEORY FOR DETECTION ALGORITHM

For the detection algorithm a matched filter is used. A matched filter is the optimal linear filter for the detection of a known signal in the presence of noise. The key component that makes a filter a ‘matched filter’ is convolving the whitened input with a known signal. The known signal used to search through the images is a 3×3 matrix and is represented graphically in Figure 2.

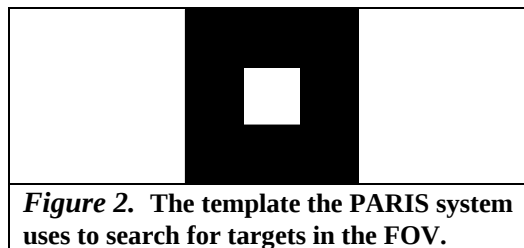


Figure 2. The template the PARIS system uses to search for targets in the FOV.

The name of the matched filter used in the PARIS system is the *Fully Adaptive Spatial Temporal* detection algorithm or FAST. To be fully adaptive, the estimate of the noise used in the filter design must be derived from the data (images) to be filtered. When using a fully adaptive filter it is important to understand the phenomenon of signal capture loss. If the signal is present, then it will be included in the estimate of the noise statistics used in the matched filter. Since these statistics are used by the matched filter to suppress noise, the signal will act to suppress itself (this is referred to as signal capture loss). This can become an issue if the number of total pixels is small and the number of targets becomes large. This is not expected to be an issue in the PARIS system because the number and size of targets is relatively small.

Covariance of the data can be described as the variation which exists in the image data. This variation is measured and used to whiten or even out the frequency spectrum of the data. After whitening has occurred the result is then convolved with the template above.

The process can be explained mathematically as follows:

If we assume that the signal ($\vec{\mathbf{S}}$) is known, but of unknown amplitude (a) with input noise of $\vec{\mathbf{X}}_0$. Then the input to the filter is:

$$a\vec{\mathbf{S}} + \vec{\mathbf{X}}_0$$

If this is the input to the system then the noise characteristics Σ_N can be calculated as follows:

$$\Sigma_N = \frac{1}{N} \sum_{k=1}^N \vec{\mathbf{x}}_k \cdot \vec{\mathbf{x}}_k^t$$

where N is the number of samples taken to compute the covariance. The method for producing the sample vector ($\vec{\mathbf{x}}_k$) is explained in the section *Covariance and Invert* on page 113. Once the covariance has been determined, its inverse is taken (Σ_N^{-1}) and along with the expected template, is multiplied by the input to produce:

$$\begin{aligned} \text{signal} + \text{noise} &= (a \cdot \vec{\mathbf{S}}^t \cdot \Sigma_N^{-1} \cdot \vec{\mathbf{S}}) + (\vec{\mathbf{S}}^t \cdot \Sigma_N^{-1} \cdot \vec{\mathbf{X}}_0) \\ &= (a \cdot \vec{\mathbf{S}}^t \cdot \Sigma_N^{-1} \cdot \vec{\mathbf{S}}) + (\vec{\mathbf{S}}^t \cdot \Sigma_N^{-1} \cdot \Sigma_x \cdot \Sigma_N^{-1} \cdot \vec{\mathbf{S}}) \end{aligned}$$

If the variance of the noise is defined as σ_o^2 , then it is shown in [1] how:

$$\sigma_o^2 \cong \vec{\mathbf{S}}^t \cdot \Sigma_N^{-1} \cdot \vec{\mathbf{S}}$$

This results in a signal to noise ratio of:

$$SNR \cong \frac{(a \cdot \sigma_o^2)^2}{\sigma_o^2} = a^2 \cdot \sigma_o^2$$

3. SPECIFICS OF THE FAST IMPLEMENTATION ON PARIS

In the PARIS system FAST is broken down into five distinct steps. The steps are **demean**, **covariance**, **invert**, **whiten** and **velocity filter** (and are performed serially in this order). As it relates to the matched filter the steps can be broken down as follows:

- | | |
|--------------------------------|----------------------------------|
| Calculate the noise statistics | – covariance & invert |
| Whiten (noise suppression) | – first part of whiten |
| Convolve template | – second part of whiten |

3.1 DATA Collection

Data collection is performed by capturing 3 consecutive 160×120 digital video frames. Since this frame data is digital and passed over a digital link, the capture process is lossless. As mentioned previously, this collection is performed by the EDMA. Because of this fact, image processing for a data set can occur while the next data set is being received.

3.2 Demeans

The demean step (figure 3) takes the average of a 5×5 array of pixels and removes it from the center pixel. The input to the FAST algorithm is a set of three images, therefore demean is performed on all three input images. To speedup execution time, changes were made to the demean process. It was observed that the total number of additions required to calculate demean for a each image are:

$$(5 \times 5) \times (156 \times 114) = 444,600 \text{ integer operations per image}$$

An optimization was found that greatly reduces execution time. It takes advantage of the fact that after the first calculation has been performed it is possible to simply add in the next column and subtract off the first column. This results in a reduction to:

$$(5 \times 5 + (5 + 5) \times 155) \times 114 = 179,550 \text{ integer operations per image}$$

This results in total overall savings of:

$$(444,600 - 179,550) \times 3 = 795,150 \text{ integer operations per demean.}$$

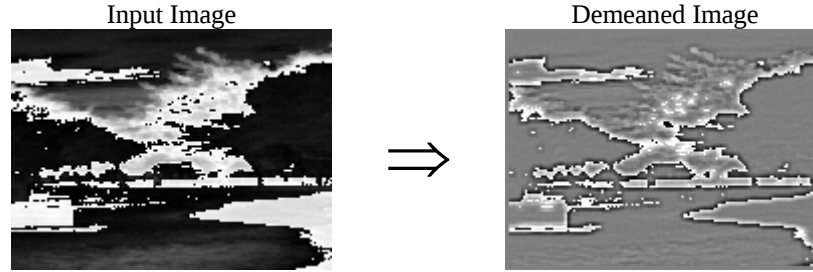


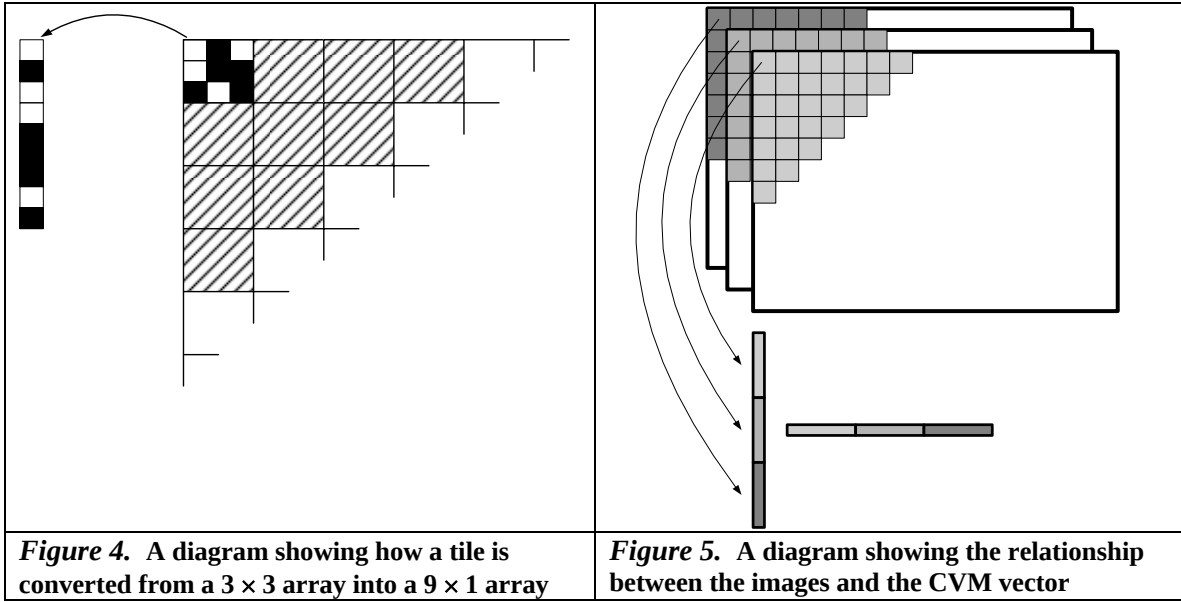
Figure 3. A before and after image showing the results of the demeaning process. This image was created using the $160 \times 120 \times 100$ camera from RCI.

Because of the manner in which the demeaned image is generated, the two outer columns and rows of pixels cannot be demeaned. This results in a demeaned image that is 156×116 pixels.

3.3 Covariance & Invert

The task performed by the covariance step is to generate the covariance matrix from the demeaned image. The covariance step breaks the demeaned image into non-overlapping 3×3 tiles. Because the number of rows left over from demean (116) is not a multiple of three, the number of rows available for use by the covariance step are limited to 114. This results in 1976 tiles per image.

The covariance matrix is generated by taking one 3×3 tile from each image. Each tile is converted into a 9×1 array (Figure 4) and then the three 9×1 arrays are stacked to form a 27×1 array (Figure 5). Once the 27×1 array is created it is multiplied by its transpose to generate a 27×27 array. This operation is performed once for each of the 1976 distinct tile positions and the results are accumulated. To speedup execution time for the calculation of the covariance matrix only certain values are computed. Since the matrix is symmetric, values do not need to be computed for both sides of the diagonal. An optimization that takes advantage of this fact only calculates the upper portion of the array and copies the values above the diagonal to below the diagonal. After all of the tiles have been processed the results are summed and then divided by the total number of distinct tile positions to produce an average. After averaging all of the variances matrices the covariance matrix (or CVM) is completed. The invert step uses the Shipley-Coleman method of matrix inversion to invert the completed CVM.



3.4 Whiten

The covariance and invert steps served to produce the second order noise statistics needed by the whiten step to suppress the noise. To accomplish this, the demeaned image is multiplied by the inverted covariance matrix to produce a whitened image (Figure 6). After creating the three whitened images, each is convolved with the template to create the final images to be used by the velocity filter (figure 7).

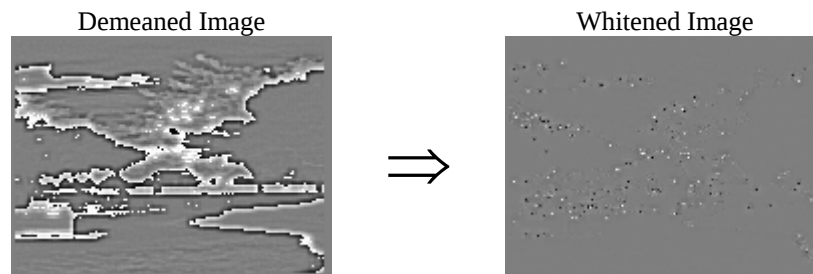


Figure 6. The result of noise suppression on the demeaned image. Generated by multiplying the CVM against 3×3 tiles of the demeaned images. This is done for all three images, only one is shown here.

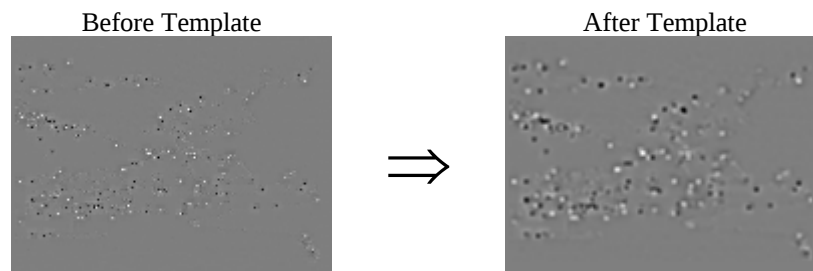


Figure 7. The result of convolving the template in Figure 1 with the whitened image. This is done for all three images, only one is shown here.

3.5 Velocity Filter

The velocity filter step looks for motion in the three images. Speed is restricted to movement of a single pixel in any direction. The velocity filter independently tests for movement in 9 directions. The directions are left, right, up, down, diagonal and no movement. This is accomplished by holding the center image constant and shifting the first and last image in opposite directions (as shown in Figure 8). When the images are shifted the corresponding pixels in each image are summed to create a new image. This new image represents movement in a particular direction. Once the 9 images are created each is scanned for local maximums. When a maximum is found that is greater than the required threshold, it is added to a list of possible detections and its SNR is calculated. This is completed for all 9 directions before the list is reduced again by searching more thoroughly for local maximums.

The points are reduced by comparing the distance between each point with every other point. If the distance is below a certain threshold the point with the lowest SNR is invalidated.

To speedup the velocity filter, changes were made to the routine to replace a mandatory divide with an optional divide that occurs less frequently. The original divide occurred once for each pixel in each summed image. This is equal to:

$$(156 \times 114) \times 9 = 160,056 \text{ divides}$$

The new divide occurs in a different part of the algorithm and at a much lower rate. In the new algorithm the divide occurs 200 times or less. The total overall effect of this optimization was to reduce execution time of the velocity filter step from ~275 ms to ~35 ms.

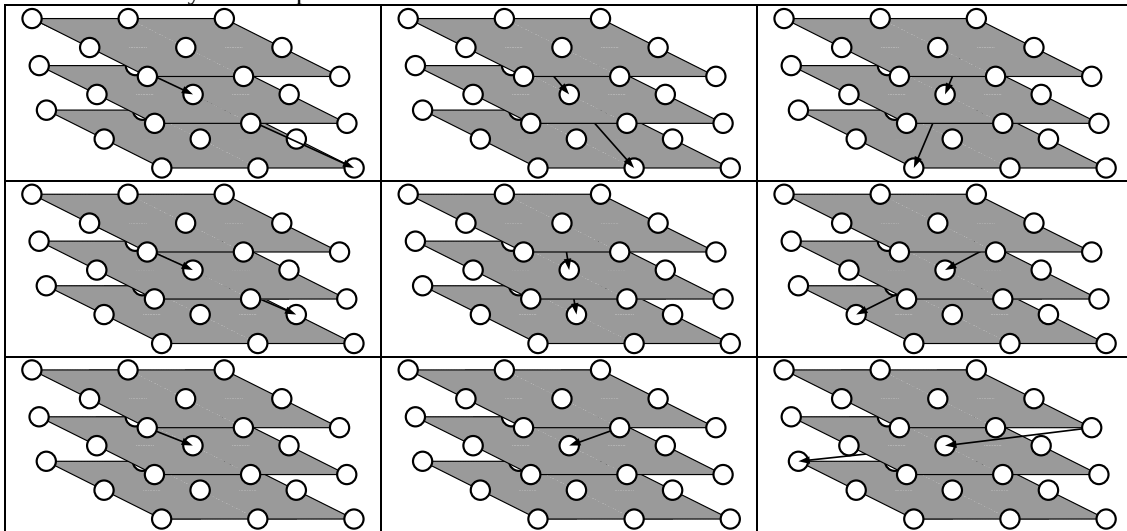


Figure 8. Diagram representing the shift that occurs during the velocity filter step. The white circles represent pixels to be summed. Nine different sets of summations occur during the velocity filter. The images are stacked top to bottom, image one to three. In the upper left hand corner the pattern of summation is shown where a pixel from the center image is added to the value of the pixel shifted up and to the left in image one. This sum is added to a pixel below and to the right in image three. This process is repeated for every pixel in image two. After completing this for each pixel this process is repeated for each of the remaining eight directions shown above.

3.6 CFAR Detector

Every time a detection is made it is added to a list. If a detection appears in the same place over three data collections it is considered a dead pixel. As long as the detection continues to appear it will be removed from the valid detection list (starting with the third time it is detected). If the detection does not occur for one cycle it will then be removed from the dead pixel list, as if it had never been on the list.

4. MULTI OBJECT TRACKER

The first step in the tracking algorithm is to extrapolate the expected position of any existing tracks. Once this has occurred then detections are assigned to these expected positions. This is done by computing the distance between the expected detection location and observed detections. The best detection is added to the track. Once a detection has been added to a track the velocity and SNR is updated. If no detections are within a threshold distance of the expected detection location then no detection will be matched to the track. If this occurs then the track will be allowed to 'coast' or update its current position with the expected value. If the track coasts for more than three data collection periods the track is deleted.

All detections that cannot be assigned to a track are potentially new tracks (this is how tracks are initiated). These detections are received from FAST with associated velocities and SNR which are used to initialize the tracker.

5. SUMMATION

The PARIS system uses advanced image processing techniques that have been tailored to maximize the capabilities of the X100 IR camera while minimizing the execution time and power consumption of the DSP.

6. ACRONYMS

CFAR	Constant False Alarm Rate
CVM	Covariance Matrix
EDMA	Enhanced Direct Memory Access
FAST	Fully Adaptive Spatial Temporal
FOV	Field of View
IR	Infrared
Mb	Megabit
NUC	Non-uniformity correction
PARIS	Power Aware Remote IR Sensor
RCI	Raytheon Commercial Infrared
SNR	Signal to Noise Ratio

7. REFERENCES

1. Singer, P., Sasaki, D. Analysis of Signal Capture Loss for Fully Adaptive Matched Filters, Raytheon SAS, El Segundo CA.

Appendix H: Algorithm Description / Mapping to PARIS Node

TABLE OF CONTENTS

1. OVERVIEW	101
2. ALGORITHM MAPPING	101
3. SUMMARY	103
4. ACRONYMS	104

1. OVERVIEW

The design of the PARIS node has gone through several iterations to minimize the power usage while maximizing algorithm performance. The current version of the PARIS node consists of a C8051 microcontroller with a TI C6713 floating point processor to do the algorithm number crunching. The PARIS node can operate either stand-alone or in conjunction with the PASTA stack. This document summarizes the effort to optimize the performance of the algorithm on the PARIS node while minimizing the power utilization on this platform configuration. The system duty cycle controls how images are acquired, processed to obtain a result and then to place major components within the PARIS node in a sleep or powered down state to minimize power utilization.

2. ALGORITHM MAPPING

There were many challenges to overcome in the integration of image processing algorithms on the Texas Instruments DSP (TMS320C6713). This effort included re-hosting existing software, planning robust testing and analysis as well as taking full advantage of the enhancements possible with new technology. Many options for improvement of the system had to be considered and weighed against their impact. These options included alternative ports for data collection and utilizing external high performance processing resources. To re-host PC software on an embedded system many changes had to be addressed including memory considerations and different operating systems. Verification and validation was performed using external resources such as oscilloscopes and C compilers in concert with custom IDE plugins. To improve performance of the embedded design, the features available to the DSP were closely examined and utilized to maximize efficiency (power vs. performance). To create the best possible system these challenges had to be recognized and addressed.

In order to improve performance new technologies and methods were examined to assist in image processing. One of the costly steps in image processing is the **local demean** step. This task consumes approximately one-third of the total processing time (about 70 ms). Reduction of this processing time is possible if dedicated external hardware performs all or part of this operation as data is received. This function could be implemented in an FPGA and many demean operations could be performed in parallel. This method was closely examined for its benefits and disadvantages. The benefits of improved processing time were overshadowed by the costs of added complexity, added risk, and increased power consumption (increased number of devices – reduced processing time = net increase in power consumption for same amount of processing). Because of this analysis it was determined that an external hardware implementation was less advantageous than a native software implementation.

On a properly optimized system data collection can take over 50% of the processing time. Three common ports of the sensor and image processor were identified as potential data collection options. These ports are the UART, Host Port Interface (HPI) and McBSP. All three of these ports were implemented and have their own advantages and disadvantages. The HPI port allows one processor to write directly to the memory of another processor. Advantages of using the HPI port are speed and reliability. Unfortunately speed degrades with cable length and cables require upwards of 20 conductors. To obtain acceptable data transmission speed it is necessary to mount the sensor directly to the image processor. Alternatively using a UART to transmit the data is not quite as fast, however it allows for reasonable cable length and a minimum of conductors. Unfortunately UART transmission is not reliable enough to depend on zero errors in about 460kB of transmitted data (the amount of data collected in one detection and tracking execution). A checksum and retransmission scheme was introduced to improve the quality of the data however reliability was so low that the time required to collect all of the required data was vastly increased. A third alternative, the McBSP port was found to be highly reliable and requires only three conductors. The three conductors were converted to LVDS allowing transmission over desired distances at desired speed. Issues with the compiling and filling of data in the McBSP port arose due to the high speed of data creation and number of execution cycles required to transmit each byte. In addition on the receiving end the Image processor has to receive and place each byte in the proper memory location (something not required by HPI). To optimize transmission of the data from the camera a DMA was implemented to move data from memory to the McBSP channel at the highest possible rate. On the receiving end, data was being received so quickly that not all of the information could be stored before it was overwritten by new data (buffer

overflow). To answer this issue an EDMA was implemented to automatically receive camera data and store it in the proper location. The relationship between data collection and image processing can be seen in Figure 1-A. Data is transmitted in 16-bit packets over a 19 MHz channel.

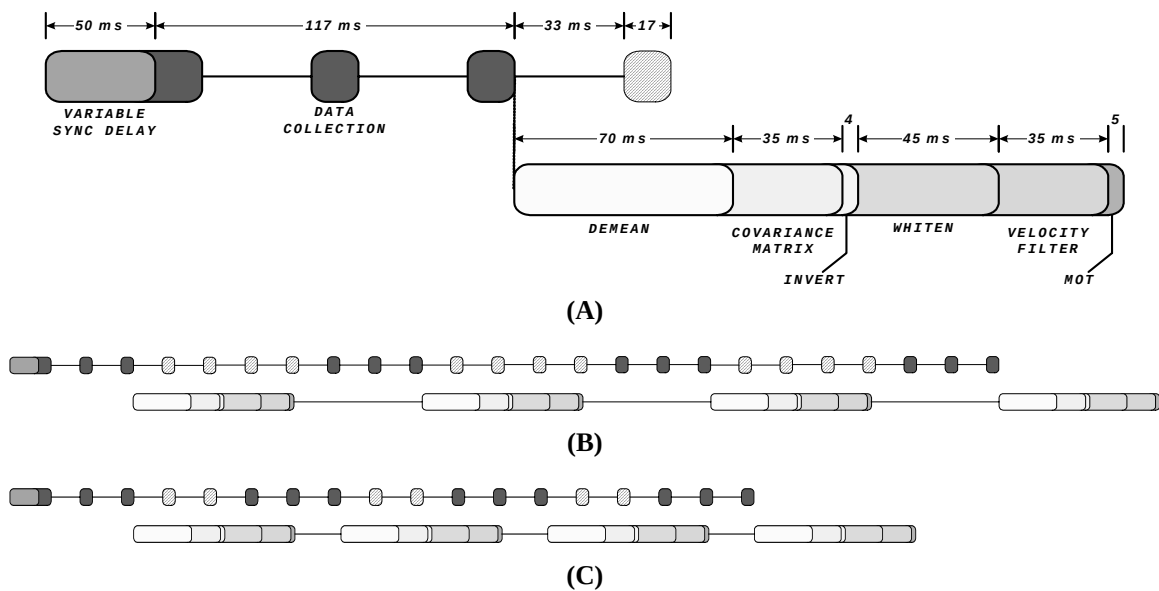


Figure 1. Timing of data collection and image processing. The figure at top (A) shows the breakdown of an individual scan collection and processing. The figures at bottom (B & C) show how data collection interacts with continuous processing.

This results in 19200 8-bit pixels being transmitted in about 17ms. The frame rate of the camera is 20 Hz which means a new frame is ready to be sent every 50 ms. Once three frames have been received (three frames are called a scan) processing can begin. Processing takes about 194 ms at its current level of optimization. The different processing steps are broken out in the diagram but will not be discussed here in detail. The lower part of diagram shows how data collection can be performed sequentially (B) with image processing or in parallel (C). The different options for data collection were all potentially viable. However the selection of the McBSP port provided a high speed reliable connection that can be transmitted over significant lengths.

Re-hosting of the image processing algorithms onto the embedded DSP required a thorough reexamination of the original software. The most obvious change that took place was the change in operating systems. The DSP uses the TI BIOS which has been specially developed for the PARIS DSP. This required PC system calls to be replaced by TI BIOS analogs. Another impact of migration of the code to an embedded platform is the elimination of PC style file system. All file I/O and calls to standard output had to be replaced with either memory structures in the case of data storage and header files in the case of initializations. In a PC the cache is configured automatically by the operating system. In the embedded design the cache had to be custom configured by the designer. The designer then had to work within the restrains setup by the cacheable region of the memory. Many optimizations were made that improved performance such as configuring cache, using the proper variable precision and making algorithm optimizations. As improvements in performance were made, the speedup increased. By increasing speedup, execution time fell as did the power required to perform the same amount of processing. This produced a dramatic effect on processing power that can be seen in Figure 2. Re-hosting the image processing software required certain modifications but also presented new opportunities for performance improvements. Changes made to get the algorithm software ported to the embedded system were non-trivial.

Verification and validation were proven on the system. Verification was accomplished by matching input and output through different phases of the system. Initially data collection was verified by sending a stream of incremented numbers through the McBSP and examining the memory of the image processor.

Once robust data collection was proven sample images were processed on the PC and in the embedded system.

Intermediate results were verified throughout the detection and tracking algorithm using custom IDE plugins to extract the data from the DSP and standard out and file I/O to extract the data from the PC version of software. Verification was performed to ensure that the algorithms created and tested on the PC were generating almost identical results on the embedded system.

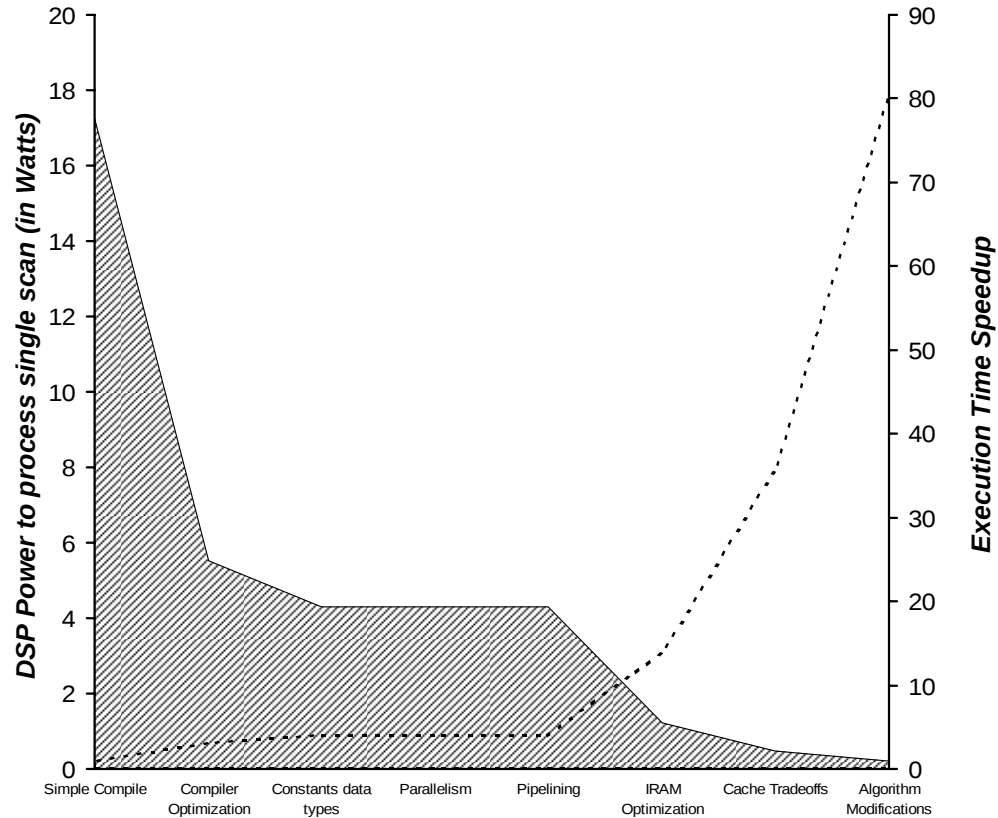


Figure 2. Power versus Execution time. The dashed line shows the speedup achieved as improvements (shown at bottom) were made. This can be compared to the area plot of power that is 240 mW at its minimum.

These tools were later used to validate operation of the system in real time. Tools were created that allowed extraction of images, movies and results from a system that was operating. These images and movies were inspected for personnel and compared against the results. Special commands were created that manipulated GPIO pins to indicate what was occurring in the system as time passed. These signals were observed using an oscilloscope and gave an extremely reliable indicator for the duration of specific functions and operations. Timing was observed and found to be below required thresholds (less than 500 ms). Verification and validation were performed by means independent of the system and produced results with high confidence.

3. SUMMARY

A high performance embedded processor was chosen for the system. The processor chosen consumes significant power but offers reduced execution time. The DSP takes advantage of simplified scheduling opportunities available in the embedded environment. The embedded system is an interrupt driven design. This enables the PARIS image processor to sleep when there is no work to be done and to wake as

processing is needed. To implement the design in this way Hardware Interrupts (HWI) are used to trigger event based activities including data collection and communication. Because HWIs are reserved for real time activities, they are designed to consume as little time as possible. These functions often trigger additional work to be performed from functions called Software Interrupts (SWI). SWIs are used when a lengthy amount of processing is to be performed, but does not require real time handling. In addition to the use of HWI and SWI the system utilizes Enhanced Direct Memory Access (EDMA). EDMA is used to handle data collection from the McBSP port. The advantages of using the EDMA are two fold. It allows transmission at rates too high for a polling method of data collection to consistently capture all of the data. It also offloads processing that would have to be performed by the CPU to internal hardware. This allows the CPU to perform other processing in parallel with data collection. By allowing the CPU to perform image processing while data collection is occurring further increases in speedup can be achieved (approximately 1.28). Notice the execution time difference between Figure 1-B and 1-C. Figure 1-B is collecting data, then processing it. Figure 1C is collecting data and processing data simultaneously. The communication channel is interrupt driven allowing the system to wake up. This prevents the CPU from waiting for each message to arrive and using additional power. In this model the CPU sleeps until a message is received and then wakes to process it. Once processing is completed the CPU then returns to sleep waiting for the next message to arrive. Significant effort was made to ensure the processor is in a low power state unless processing has been requested.

4. ACRONYMS

BIOS	Built in Operating System (or Basic Input Output System)
CPU	Central Processing Unit
DMA	Direct Memory Access
DSP	Digital Signal Processor
EDMA	Enhanced Direct Memory Access
FPGA	Field Programmable Gate Array
HPI	Host Port Interface
HWI	Hardware Interrupt
I/O	Input Output
IDE	Integrated Development Environment
IR	Infrared
LVDS	Low Voltage Differential Signal
McBSP	Multi Channel Buffered Serial Port
ms	milli-second
PARIS	Power Aware Remote Information Sensing
PC	Personal Computer
SWI	Software Interrupt
TI	Texas Instruments
UART	Universal Asynchronous Receiver Transmitter