A Modular Platform for Nanopower Computing

Pat Pannuto, UC Berkeley

https://patpannuto.com ppannuto@berkeley.edu

Secure Internet of Things Project (SITP)
June 18, 2019
“A computer class is... a new platform with a new programming environment, a new network, and new interface.”
Bell’s Law captures the evolution of computing platforms

By volume, the emerging computing classes are mostly energy storage

\textit{Volume is shrinking cubically}
Computational platforms will continue to scale

The next generation of computing will only be a cubic millimeter in size

“Smart Dust”
Millimeter-scale form factor is key to opening a wide array of new applications
Computational platforms will continue to scale

The next generation of computing will only be a cubic millimeter in size

Millimeter-scale batteries have capacities around 5 µAh

(would power an idle iPhone for 0.6 s)

“Smart Dust”
To support their target applications, Smart Dust systems must last longer on less energy
Energy constraints will play a central role in the evolution of computing platforms

How must traditional paradigms change, adapt, or re-invent for the new computing classes?
Outline

• Introducing the nanopower computing class

• The development of a modular, nanopower computing platform
  Hardware — Composing micro-scale systems
  Software — Returning modularity to microcontroller software
  Services — Deployment and management of “dust”

• The future of nanopower computing and its broader impact
A very brief history of “Smart Dust”

Monolithic: Every sensing system requires a completely new chip

Abstract
Large-scale networks of wireless sensors are becoming an active topic of research. Advances in hardware technology and engineering design have led to dramatic reductions in size, power consumption and cost for digital circuitry, wire-
Claim: Efficient modularity is a necessary condition for the emergence of a computing class

Modularity...

- “increases the range of ‘manageable’ complexity”
- “allows different parts of a large design to be worked on concurrently”
- “accommodates uncertainty”
- adds costs

Quotes from pages 90-91: What is Modularity?
We have modular components...

Temperature Sensor
~10 pW standby, < 1 μW active

CPU
~1 nW standby, ~5 μW active

Radio
~10 pW standby, ~10 μW active

Energy Harvesting & Storage
1~10 nW indoors
2~10 μAh capacity
We have modular components... but no suitable, general purpose, common interconnect

Temperature Sensor
~10 pW standby, < 1 µW active

CPU
~1 nW standby, ~5 µW active

Radio
~10 pW standby, ~10 µW active

Energy Harvesting & Storage
1~10 nW indoors
2~10 µAh capacity
What’s wrong with existing interchip interconnects?

Energy.

- Inter-Integrated Circuit (I²C) Bus
  - Fixed wire count
  - Multi-master
  - Compact addressing
  - Hardware acknowledgements
  - Clock stretching
  - Easy voltage level translation
  - ...

- Too energy-inefficient for Smart Dust
$I^2C$ has fixed I/O requirements and a decentralized architecture
$I^2C$ has fixed I/O requirements and a decentralized architecture
I²C has fixed I/O requirements and a decentralized architecture
I²C is built on a simple circuit that enables its properties.
Problem is the energy costs of running an open-collector

Send “1”: 0 W

Send “0”: ...

Power Source
Resistor
Switch
Ground
Achieving energy efficiency with I²C sacrifices modularity

$V_{DD} = 1.2 \, V$

$R = 1.8 \, k\Omega$ (standard)

$R = 23.6 \, k\Omega$ (“best”)

$R = 10.1 \, k\Omega$ (ratioed)

Logic Low $0.3*V_{DD}$

5 nodes @ 3pF

Send “0”

0.01% Duty Cycle

474 µW

75 µW

4 µW
MBus enables Amdahl-balanced modularity for the nanopower computing class

- Temperature Sensor
  ~10 pW standby, < 1 µW active

- CPU
  ~1 nW standby, ~5 µW active

- Radio
  ~10 pW standby, ~10 µW active

- Energy Harvesting & Storage
  1~10 nW indoors
  2~10 µAh capacity
What makes MBus hard is satisfying **all** of the design constraints for current and future millimeter-scale systems

- Low active power
- Fixed pin count (4)
- Minimal standby power
- Multi-master design
- Fully synthesizable
- Robust timing (I/O drive/load)
- Efficient & future-proof addressing
- Data independent behavior
- Hardware acknowledgements
- Power aware design
Requirement: Low Active Power

Idea: Eliminate the energy-hungry pullup

- Per clock cycle...

~1,200 pJ lost to pull-up

~150 pJ to pads/wires
Requirement: Fixed Pin Count
Requirement: Fixed Pin Count
Idea: Ring topology scales and prevents conflicts
Requirement: Low Active Power & Low Standby Power
Requirement: Low Active Power & Low Standby Power

Idea: Combinational frontend

- Clockless “shoot-through” ring
- All bus frontends share one bus clock
  - No local oscillators saves energy
  - Generated by mediator
  - Usually a CPU-like node
Requirement: Multi-Master Design
Requirement: Multi-Master Design

Idea: Topological priority can mitigate rare conflicts

- One node “mediates” arbitration
  - Does not forward during idle
  - Generate bus clock when DATA_IN falls
- Unambiguous winner
- Multiple arbitration rounds allow for non-topological priority schemes
MBus is a clean-slate design, built to satisfy interconnect requirements for this and the next generation of modular systems

- Low active power
- Fixed pin count (4)
- Minimal standby power
- Multi-master design
- Fully synthesizable
- Robust timing (I/O drive/load)
- Minimal protocol overhead
  - Safe & efficient arbitration
  - Efficient & future-proof addressing
  - Data independent behavior (end of message?)
  - Transaction acknowledgements

Power aware design
Energy-critical systems push from “dark silicon” to “pitch black” silicon

- **Clock-gating: “Dark Silicon”**
  - Stop driving the clock tree of regions of a chip
  - Eliminates switching power, but not static leakage
- **Power-gating: “Pitch Black Silicon”**
  - Switch off power to regions of a chip
  - Eliminates (almost) all chip leakage
- **Micro-scale systems aggressively power-gate to reach energy budget**
  - Power management is a looming systems synthesis problem
Distributed operation creates challenges for system power management

- Monolithic systems can bootstrap, monitor, and manage power state from a centralized local controller.
- Distributed operation can be complex and involve the bus interface between components.
Problem 1: Efficiently “talking through” unpowered chips

7 Gates
Problem 2: Can I talk to it? Is it on?
Problem 2: Can I talk to it? Is it on?
Problem 3: How do I turn something on?
Problem 3: How to turn something on?

- Cold-booting circuits is much easier if there is a stable clock available
  1. Turn it on
  2. Start your local clock
  3. Connect to the already powered parts
  4. Release power-on reset
MBus Insight: Problems 2 and 3 can be solved together

- Power Oblivious Communication

Awake, after “losing” arbitration
MBus introduces three, transparent, hierarchical power domains to maximize efficiency

- Minimal always powered frontend
  - 32 logic gates, 4 flops
- Small controller active during transmission
  - 27,300 µm² in 180 nm process

<\text{pW}: \text{Wire Controller} \\
\sim nW: \text{Bus Controller} \\
\sim \mu W+: \text{Rest of IC}
Extending hierarchical power domains one step further
MBus and the next, next, next generation

- MBus abstraction presented to chips:
  - Power control signals
  - Byte-oriented send/receive
  - “Always-on” interrupt
  - 304 nW analog motion detection
  - 20 μW digital image capture mode
MBus enables an ecosystem of millimeter-scale, nanopower platforms
Seamless and transparent interaction between power-aware and power-oblivious chips

- Facilitates integration with COTS chips

*No current COTS chip support MBus, these integrations leverage more traditional buses or bitbanging*
MBus is the next-generation system interconnect.

MBus is a chip-to-chip bus designed for ultra-constrained systems. MBus is a multi-master bus supporting an arbitrary number of nodes, priority arbitration, efficient acknowledgements, and extensible addressing, with only four wires and consuming power.

MBus is power-aware, enabling individual chips can fully power down regions of all the tricky details.

Overview

MBus was invented by the Michigan Micro Mote (MiMo) project. The goal of MBus, however, is to be a general purpose bus for ultra-constrained systems. MBus comprises four plus plus plus plus, support analog, digital, high-speed, power-aware inter-chip communication, and features a multi-symmetric bus with linear scalability and feature set. High-speed communication is targeted at a high-performance, low-energy chip, and primary mission is providing the MBus clock and initializing arbitration.

Contents

1. Requirements & Design Considerations
   1.1. Protocol Design
   1.2. Memory
   1.3. Power
   1.4. Design Consensus
2. Protocol & Memory Addressing
3. Bus Design
   2.1. Physical Design
   2.2. Memory
   2.3. Bus Configuration
   2.4. Interoperability

MBus © 2012 – 2014 The Regents of the University of Michigan
Check out the “World’s Smallest Computer” exhibit at Silicon Valley’s Computer History Museum!
Outline

• Introducing the nanopower computing class

• The development of a modular, nanopower computing platform
  Hardware  —  Composing micro-scale systems
  Software  —  Returning modularity to microcontroller software
  Services  —  Deployment and management of “dust”

• The future of nanopower computing and its broader impact
Tock is a new embedded operating system designed for safe, robust multitenancy on microcontrollers.

Programmable IoT starts at the edge

An embedded operating system designed for running multiple concurrent, mutually distrustful applications on low-memory and low-power microcontrollers.

- **Extensible**
  Safely use drivers and kernel extensions from third parties

- **Reliable**
  Run processes reliably with minimal resource overhead

- **Low-power**
  Automatic low power operation

**Multiprogramming a 64kB Computer Safely and Efficiently**

*Proceedings of the 26th Symposium on Operating Systems Principles (SOSP'17)*
The case for multitenancy on a millimeter-scale computer

Modularity...

• "increases the range of ‘manageable’ complexity"
• "allows different parts of a large design to be worked on concurrently"
• "accommodates uncertainty"

"And generality always wins"

• *Universality* – any computer can take over the function of another given sufficient memory and interfaces
Networking often drives multitenancy, but comes with additional risk for unattended devices

- Multitenancy != multi-app
  - Black box vendor libraries (e.g. BLE stacks)
  - FCC compliance
- Modular computing means arbitrary compute
  - Phones, watches... fixed-function to platform
- Multiprogrammability amortizes costs
  - Siloed sensor infrastructure is expensive
  - Future-proof, short & long-run adaptive
Microcontrollers rather than microprocessors dominate the emerging compute class, and must learn to become a platform

- Why is the software so different? The hardware is different...
  - MPU (Memory Protection Unit) rather than MMU (Memory Management Unit)
  - Transient storage (SRAM) is limited, and not growing (because limited energy!)
  - From 2004 to 2017... 10 kB ➔ 64 kB

<table>
<thead>
<tr>
<th></th>
<th>Min CPU</th>
<th>Min RAM</th>
<th>Min Disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows 10 (32-bit)</td>
<td>1 GHz</td>
<td>1 GB</td>
<td>16 GB</td>
</tr>
</tbody>
</table>
An embedded OS must balance stability and flexibility with fewer resources on a less flexible hardware platform

- Challenge: How to be **robust** and **adaptive** at the same time?
  - Robust: Memory exhaustion is common, and no virtual memory out
    - Kernel must be statically allocated
    - (contrast to Linux’s “too small to fail” rule)
  - Adaptive: Support varying application sets with varying workloads
    - Kernel must support dynamic allocation of resources
  - These are at odds!

**The benefits and costs of writing a POSIX kernel in a high-level language**

Cody Cutler, M. Frans Kaashoek, and Robert T. Morris, *MIT CSAIL*

“The most challenging puzzle was handling the possibility of running out of kernel heap memory”
Traditional kernel heap allocation results in shared fate across all processes
Traditional kernel heap allocation results in shared fate across all processes
Traditional kernel heap allocation results in shared fate across all processes
Traditional kernel heap allocation results in shared fate across all processes
Traditional kernel heap allocation results in shared fate across all processes
Traditional kernel heap allocation results in shared fate across all processes
Traditional kernel heap allocation results in shared fate across all processes.
Traditional kernel heap allocation results in shared fate across all processes.
Grants safely and efficiently fragment the kernel heap inside the process triggering the dynamic allocation.
Grants: Kernel heap safely borrowed from processes

```rust
cfn enter\textasciitilde{a}, F)(\textasciitilde{a} self, pid: ProcId, f: F) \rightarrow \text{where } F: \text{for }\textasciitilde{b}\text{ FnOnce}(\textasciitilde{b} \text{ mut } T)

// Can’t operate on timer data here
timer\_grant\_enter(process\_id, |timer| {
// Can operate on timer data here
if timer\_expiration > cur\_time {
    timer.fired = true;
}
})

// timer data can’t escape here
```
There are more resources than memory… Correct-by-construction power management

• Challenge: How to ensure peripherals are in the correct power states?
  – On all execution paths?
Modularity mismatch: Isolated peripherals have complex state machines, software mixes execution paths
Ongoing work: Leveraging rich type systems to capture peripheral state machines

Case Study: USART

• 11,000 SLOC
• `clock_enable()`/`clock_disable()` from 20 calls to just 1
• Removed 35 unsafe blocks
• ~40 minutes of expert developer time
• 20x reduction in energy on test workload
Deeply embedded platforms today can and should provide the types of safety and correctness guarantees, afforded by modular abstractions, expected from traditional compute platforms.

Foreshadowing future work
We've done some of this, but there is more to go.
Tock today: In use by universities around the globe, major corporations, startups, and hobbyists. [www.tockos.org](http://www.tockos.org)
Outline

• Introducing the nanopower computing class

• The development of a modular, nanopower computing platform
  Hardware — Composing micro-scale systems
  Software — Returning modularity to microcontroller software
  Services — Deployment and management of “dust”

• The future of nanopower computing and its broader impact
Two management questions

1. How do you program something too small to attach wires to?

2. How do you keep track of 1,000s of millimeter-sized computers?
We can realize an always-on optical wakeup frontend for just 100’s of picoWatts
The solid state revolution: not just a substitute good

Global replacement of lighting infrastructure!

- European Commission. Commission adopts two regulations to progressively remove from the market non-efficient light bulbs.

System Architecture Directions for a Software-Defined Lighting Infrastructure
Ye-Sheng Kuo, Pat Pannuto, and Prabal Dutta
Workshop on Visible Light Communication Systems (VLCS '14)
The solid state revolution: not just a substitute good

LED Luminaire

Smart Phone

01100101000

Captured using a rolling shutter

Image processing extracts beacon locations and frequencies

AoA geometry returns position and orientation

Illumination

Entertainment

Communications

Device Configuration

Time Synchronization

Indoor Positioning

Luxapose: Indoor Positioning with Mobile Phones and Visible Light
Ye-Sheng Kuo, Pat Pannuto, Ko-Jen Hsiao, and Prabal Dutta
The 20th Annual International Conference on Mobile Computing and Networking (MobiCom ’14)
The solid state revolution: not just a substitute good

The multitenant lightbulb is an exciting platform for modular, composable services
Two management questions

1. How do you program something too small to attach wires to?
   - A modular visible light platform supports programming, configuration, and synchronization of devices

2. How do you keep track of 1,000s of millimeter-sized computers?
Slocalization: Ultra wideband backscatter localization (Not quite millimeter-sized yet)
Why RF, why ultra wideband, why backscatter for ubiquitous localization?

Why RF, why ultra wideband, why backscatter for ubiquitous localization?
Reflections make time-of-flight estimation difficult and inaccurate
Ultra wideband can better disambiguate multipath and identify signal arrival time
Why RF, why ultra wideband, **why backscatter** for ubiquitous localization?
There is a new tradeoff to introduce to enable wide-area ultra-low power, high-quality localization

- Covers areas 30m+
  - “through walls”
- Decimeter accurate
- <1 μW tag
  - (COTS, can do order of magnitude or more better with VLSI)
- (Nearly) unlimited number of concurrent tags
- 1-15+ minutes per location fix
  - A latency/energy tradeoff for localization

Slocalization: Sub-μW Ultra Wideband Backscatter Localization

Pat Pannuto, Kempke, Benjamin, and Prabal Dutta
UWB Backscatter is passive reflection of a lot less energy than traditional communications.
UWB Backscatter is passive reflection of a lot less energy

Typical receiver sensitivity ranges from -94 to -106

<table>
<thead>
<tr>
<th>Packet Error Rate</th>
<th>Data Rate</th>
<th>Typical Receiver Sensitivity</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>1%</td>
<td>110 kbps</td>
<td>-106</td>
<td>dBm/500 MHz</td>
</tr>
<tr>
<td></td>
<td>850 kbps</td>
<td>-101</td>
<td>dBm/500 MHz</td>
</tr>
<tr>
<td></td>
<td>6.8 Mbps</td>
<td>-93 (*-97)</td>
<td>dBm/500 MHz</td>
</tr>
<tr>
<td>10%</td>
<td>110 kbps</td>
<td>-102</td>
<td>dBm/500 MHz</td>
</tr>
<tr>
<td></td>
<td>850 kbps</td>
<td>-106</td>
<td>dBm/500 MHz</td>
</tr>
<tr>
<td></td>
<td>6.8 Mbps</td>
<td>-94 (*-98)</td>
<td>dBm/500 MHz</td>
</tr>
</tbody>
</table>
How do we recover a signal that is way below the noise floor?

- Exploit tag stationarity and environmental stability
How do we recover a signal that is way below the noise floor?

- Exploit tag stationarity and environmental stability
Ideally, the only change in the channel impulse response is the tag reflection

- Subtracting the environment finds the tag
The goal is to estimate the time difference of arrival (TDoA) and laterate

- First peak is anchor—anchor path, then anchor—tag—anchor
Extracting the tag signal in the real world has a few additional challenges

- The environment is not actually static
  - But noise is largely white & Gaussian
  - And we can filter out the rest (sets floor for tag frequency, active power)

- Finding tag phase offset currently requires brute force search
Directly generating and recovering UWB signals is challenging (especially circa 2014-2018... changing fast!)

RTLS Systems are black box

802.15.4a has protocol expectations & overhead

Research receivers: expensive, noisy, or niche

Ubisense Research Package (now Investcorp Technology Partners)

Time Domain P440 (now Humatics)

Pixie

Ciholas DWUSB

Pozyx

DecaWave DW1000


Directly generating and recovering UWB signals is challenging (especially circa 2014-2018... changing fast!)

- Developed bandstitched UWB transceiver architecture
  - Generic narrowband SDR (USRP)
  - Measure Channel Frequency Response in 20~25 MHz chunks

![Diagram of UWB transceiver architecture](image-url)
Directly generating and recovering UWB signals is challenging (especially circa 2014-2018... changing fast!)

- Developed bandstitched UWB transceiver architecture
  - Generic narrowband SDR (USRP)
  - Measure Channel Frequency Response in 20~25 MHz chunks

![Figure 3: RX Interferer Immunity on Channel 2](DW1000 Datasheet)
Does it really work?

- 15 minutes can cover 30 meters
- 7 cm error (3D Euclidean distance)
The same infrastructure can track moving devices on-demand, enabling adaptivity

- 14cm median
  31cm 90%ile

- COTS
  - Tag weighs 3g and costs ~$4.50
  - Draws 75mW transmitting, 3.9mJ / fix

- VLSI
  - 0.2 mm² IC
  - Draws 1mW transmitting, 60µJ / fix

Harmonia: Wideband Spreading for Accurate Indoor RF Localization (HotWireless’14)
Harmonium: Asymmetric, Bandstitched UWB for Fast, Accurate, and Robust Indoor Localization (IPSN’16)
Harmonium: Ultra Wideband Pulse Generation with Bandstitched Recovery for Fast, Accurate, and Robust Indoor Localization (TOSN’18)
Part of a broader collection of localization technologies

**Human interaction tracking**
- Opo. 93h battery, 5cm, 0.5 Hz

**Robust Ranging**
- SurePoint. 53cm 99th %ile

---

**Opo: A Wearable Sensor for Capturing High-Fidelity Face-to-Face Interactions (SenSys’14)**

**SurePoint: Exploiting Ultra Wideband Flooding and Diversity to Provide Robust, Scalable, High-Fidelity Indoor Localization (SenSys’16)**
Two management questions

1. How do you program something too small to attach wires to?
   – A modular visible light platform supports programming, configuration, and synchronization of devices

2. How do you keep track of 1,000s of millimeter-sized computers?
   – Perhaps with ultra wideband backscatter
Outline

• Introducing the nanopower computing class

• The development of a modular, nanopower computing platform
  Hardware — Composing micro-scale systems
  Software — Returning modularity to microcontroller software
  Services — Deployment and management of “dust”

• The future of nanopower computing and its broader impact
Smart Dust Delivered.
Delivering the millimeter-scale computing class

• Communication is always the hardest part
  – Backscatter is promising but carries infrastructure demands
  – First systems used VLC! Can we bootstrap communication with localization?

• Advanced computation at the edge
  – Executing ML in resource-constrained environments
  – Training ML from physically challenging deployments (federated learning?)

• Security, Privacy, Ownership, and Enforcement
  – Discovery and interaction paradigms for owners, visitors, or opportunists
  – Long-running tasks across an evolving physical compute fabric
  – Detecting illicit devices
The first two theses?

• Revitalizing microarchitecture for microcontrollers
  – MMUs aren’t actually expensive if you only have 64k of memory...
  – Where are the 64-bit MCUs? What about an IOMPU?
  – RISC-V is a clean slate opportunity!

• How to guarantee a battery-powered system will last for 10 years?
  – Correct-by-construction power management
  – Hardware and OS runtime monitoring
  – Energy-adaptive applications; programming environments?
“Digitizing the Physical World,” aka what can’t your computing access that you really want to?

- Application verticals are a joy of embedded research, currently:
  - City/country-scale power grid health monitoring with Berkeley economists
“Digitizing the Physical World,” aka what can’t your computing access that you really want to?

• Application verticals are a joy of embedded research, currently:
  – City/country-scale power grid health monitoring with Berkeley economists
  – Longitudinal parent/child interaction tracking with Vanderbilt psychologists
“Digitizing the Physical World,” aka what can’t your computing access that you really want to?

• Application verticals are a joy of embedded research, currently:
  – City/country-scale power grid health monitoring with Berkeley economists
  – Longitudinal parent/child interaction tracking with Vanderbilt psychologists

• Democratizing embedded systems & untethering the maker movement
  – Shrinking the chasm between Arduino tutorials and engineered product
  – Freshman engineers should be exposed to hardware!
A Modular Platform for Nanopower Computing

Pat Pannuto, UC Berkeley

https://patpannuto.com
ppannuto@berkeley.edu
Enhancing access to computer science more broadly

• Course for early-career EECS students (600+ students over 2 semesters)
  • This course is invaluable for students who want to be in EECS but don’t have the background that independent programmers come in with. Classes like this are what enable students who feel behind to enter classes at the same level as others with more experience. I really appreciate everyone who set up this course – Student 2
  • Simply put, it’s hard to be a newcomer to CS. I think the lecturers did a really good job of erasing some of the barriers of learning new skills – Student 5

• What I am thinking about next
  – Empower people to interact with their physical world
Backup: Computing for Computer Scientists

• Pedagogical goals for CS versus CS evaluation metrics
  – Computer science principles vs software engineering?

• High variance in student background
  – Achievement gap, belonging gap

• Created course for first-year EECS students
  – 1,500+ students and counting (I’ve taught around 600)
  – Now permanent course at Michigan (EECS 201)
    • This course is invaluable for students who want to be in EECS but don’t have the background that independent programmers come in with. Classes like this are what enable students who feel behind to enter classes at the same level as others with more experience. I really appreciate everyone who set up this course – Student 2
    • Simply put, it’s hard to be a newcomer to CS. I think the lecturers did a really good job of erasing some of the barriers of learning new skills – Student 5
Backup: What’s your home community?

• SenSys + IPSN (3+3)
  – MobiCom (2)
• Highly collaborative
  – 40 co-authors
  – 9 institutions
  – 4 continents
What I’d like to hear from you

• What do you (or your algorithms) want to know about the world?
  – What data do you want that you don’t have?
  – And why can’t you get it?
The Signpost platform: Infrastructure-free infrastructure for city-scale sensing applications

- This means multiple, independent, untrusted applications must share
Backup: How does machine learning fit into all this?

- Bringing EdgeML all the way to the edge
- What does battery-powered ML look like?
- What does battery-free ML look like?
Backup: Pushing ML into sensors
Backup: What about embedded OSes?

- Extant embedded OSes are really more like libraries

<table>
<thead>
<tr>
<th>System</th>
<th>Concurrency</th>
<th>Memory Efficiency</th>
<th>Dependability</th>
<th>Fault Isolation</th>
<th>Loadable Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arduino [6]</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RIOT OS [5]</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Contiki [14]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>FreeRTOS [8]</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>TinyOS [33]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>TOSThreads [28]</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>SOS [23]</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Tock</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
Backup: Teaching

• Undergrad
  – Core embedded [CSE145, 237A-D]
  – Broad background
    • Digital Logic [140/ECE25], Computer Architecture [30/141], Operating Systems [120/121], Networking [123/124], Ubicomp [118/218]
      – less so: Compilers [131], Synthesis/Optimization [143], Signals & Systems [ECE45]
  – Computing for Computer Scientists (C4CS)

• Graduate
  – “Resource Constrained Computing”
  – Systems (Embedded, Operating), Networking, Wireless
Backup:
Pedagogical goals of an undergraduate embedded course

• Understand the mechanics of how software interfaces with hardware
  – MMIO, peripheral buses, etc
• Understand how hardware interacts with the physical world
  – ADCs/DACs, quantization, uncertainty
• Understand how communication works
  – On-device (UART/I2C/SPI)
  – Off-device (Wireless – *maybe* some of the why/how of 15.4/LoRa w.r.t. energy)
• Understand how system design affects energy
Backup: Extracting the tag signal in the real world has a few additional challenges

• The environment is not actually static
  – But noise is largely white & Gaussian
  – And we can filter out the rest

• When to add and when to subtract?
  – Problem: Need to know when the tag is reflecting or absorbing
  – Solution: Guess and brute force search
    • Tag stability limits Slocalization range
Backup: Recovery of modulated signal improves with greater integration time

- $R_1 = 6 \text{ m}, R_2 = 24 \text{ m}$
Backup: Recovery of modulated signal improves with greater integration time

- \( R_1 = 6 \text{ m}, R_2 = 24 \text{ m} \)
Backup: Recovery of modulated signal improves with greater integration time

- $R_1 = 6 \text{ m}, R_2 = 24 \text{ m}$

Raw Channel Impulse Response

Correlated Channel Impulse Response

250 seconds
Backup: Recovery of modulated signal improves with greater integration time

- $R_1 = 6 \text{ m}$, $R_2 = 24 \text{ m}$

![Raw Channel Impulse Response](image)

![Correlated Channel Impulse Response](image)

500 seconds (8.3 minutes)
Backup: Numerous diversity sources allows Slocalization to scale to very many tags

- Frequency division scales linearly in frequencies
  - Caveat: 256.00 Hz low-power RTCs exist, less so 256.20 Hz, etc
  - Caveat: Power draw scales linearly with switching frequency
- PN codes scale linearly in tag length
- Temporal code rotation scales factorially, but is very slow
  - Idea: Exploit tag stationarity further, rotating PN codes over time

$$f_{\text{STEPS}} \times \text{PNbits} \times \text{Codes} = 1,280 \times 63 \times 4! = 1,935,360$$ concurrent tags in roughly a few hours
Backup: Can we localize every physical thing? Even dust?

- Super-resolution technique from MobiCom’17
  - Use frequency information to refine localization

- Key idea:
  - If traditional localization can get close (7cm), refine estimate based on the estimate from each frequency

- What about mm-scale antennas?
  - Today, in frequencies of interest, fall from 0 to -15 dBi
In the next few years...
Energy is a deployment concern and a first-order resource

Q: What should OS policy be for energy as a resource?

Q: What is the abstraction for cross-platform energy performance?

Q: What does graceful degradation look like, how do we support it?

Q: What are the other salient resources: time, bandwidth, ?

Q: What is the role of hardware support (e.g. PRET machines)?

Q: How do we capture all of these constraints for app developers?
In the next few years...
From EdgeML to Peripheral ML

• Embedded == weird computers
  - Megasample / second sensors
  - 10-100’s MHz processors
  - 10-100’s kB of RAM
  - 1-100’s kbit/s communications

• With weird execution environments

Neal Jackson, Joshua Adkins, Prabal Dutta
To Appear (IPSN’19)
In the next few years...
From EdgeML to Peripheral ML

- Embedded are weird computers
  - Megasample / second sensors
  - 10-100's MHz processors
  - 10-100's kB of RAM
  - 1-100's kbit /s communications

- With weird execution environments

EdgeML is still plugged in

Q: What does ML look like on peripheral compute?
In the longer term...
How does the world change with billions of devices?

Q: How will we manage systems when devices outnumber people 100 or 1000:1?

Q: How do we push towards unattended computing? How do we make this technology “disappear?”

Q: How do we interact with dust?
Teaching at ETH

• Undergrad Qualifications
  – Core embedded
  – Broad background
    • Circuits, Signals & Systems, Digital Logic, Computer Architecture, Operating Systems, Networking; less so Compilers, Synthesis/Optimization
    • D-ITET Courses
      – Networks and Circuits, Digital Circuits, Signal and System Theory
      – Communication Systems, Communication Networks, Embedded Systems

• Graduate Seminars
  – Embedded, Systems, Networking, Wireless
Hard Problem 1: What’s the best way to terminate messages?

- An MBus message is 0…N bytes of data
- **Embed length in message**
  - Imposes large overhead for short messages
  - Forces fragmentation of long messages
- **“End-of-message” sentinel byte(s)**
  - Imposes large overhead for short messages
  - Requires escaping if sentinel is in transmitted
  - Data-dependent behavior, hard to reason about
    - Worst case 2x overhead!
MBus “interjections” provide an in-band end-of-message with minimal overhead

- During normal operation, Data toggles slower than Clock
MBus “interjections” provide an in-band end-of-message with minimal overhead

- During normal operation, Data toggles slower than Clock

![Diagram](image-url)
MBus “interjections” provide an in-band end-of-message with minimal overhead

- During normal operation, Data toggles slower than Clock

Independent circuit provides reliability
Transaction-level ACKs minimize common-case overhead while interjections preserve flow control

- I²C acknowledges every byte
  - How often do NAKs happen?
    - To a random byte?
    - 12.5% overhead
- MBus ACKs transactions
  - Receiver can interject too

Arbitration
  + Address
  + Interjection
Location context is fundamental to a bevy of ubiquitous computing applications

– **Little is more basic to human perception than physical juxtaposition, and so ubiquitous computers must know where they are**
  
  • *Mark Weiser, The Computer for the 21st Century*

• But we do not have uniform means of expressing location
• Nor do most computational elements posses it
• Suggestion: This has inhibited effective creation of computational systems for “smart spaces”
  
  – How to make a space smart with no sense of space?
Ultra Wideband affords extremely high-fidelity localization

UWB continually dominates the Top 10 in the IPSN Indoor Localization Competition
Ultra Wideband Radios (or impulse generators) are energy-hungry

<table>
<thead>
<tr>
<th>System</th>
<th>Technology</th>
<th>Precision</th>
<th>Accuracy</th>
<th>Update Rate</th>
<th>Multiple Tags?</th>
<th>Top Tag Speed</th>
<th>Tag Power</th>
<th>Tag Volume</th>
<th>Max Tag/Anchor Dist</th>
</tr>
</thead>
<tbody>
<tr>
<td>WASP [35]</td>
<td>NB (5.8 GHz) TDoA</td>
<td>16.3 cm</td>
<td>50 cm (82%)</td>
<td>10 Hz</td>
<td>Yes</td>
<td>Several m/s</td>
<td>2-2.5 W</td>
<td>Not Published</td>
<td>Not Published</td>
</tr>
<tr>
<td>UbiSense [38]</td>
<td>UWB TDoA+AoA</td>
<td>99% w/in 30 cm</td>
<td>15 cm</td>
<td>33.75 Hz</td>
<td>Yes</td>
<td>Not Published</td>
<td>Not Published</td>
<td>24.5 cm³</td>
<td>160 m</td>
</tr>
<tr>
<td>TimeDomain [2]</td>
<td>UWB TW-ToF</td>
<td>2.3 cm</td>
<td>2.1 cm</td>
<td>150 Hz</td>
<td>Yes</td>
<td>Not Published</td>
<td>4.2 W</td>
<td>97 cm³</td>
<td>“hundreds of m”</td>
</tr>
<tr>
<td>Laziz et. al [27]</td>
<td>Ultrasonic TDoA</td>
<td>Not Published</td>
<td>3 cm (med)</td>
<td>12 cm (90%)</td>
<td>0.9 Hz</td>
<td>Yes</td>
<td>Not Published</td>
<td>1.1 W³</td>
<td>88 cm³</td>
</tr>
<tr>
<td>Harmonia [19]</td>
<td>UWB TDoA</td>
<td>Not Published</td>
<td>39 cm (med)</td>
<td>82 cm (90%)</td>
<td>56 Hz</td>
<td>No</td>
<td>Not Published</td>
<td>120 mW**</td>
<td>Not Published</td>
</tr>
<tr>
<td>Tagoram [42]</td>
<td>NB (UHF) SAR</td>
<td>Not Published</td>
<td>2.3 cm (med)</td>
<td>150 m/s</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>8 cm³</td>
<td>10 m</td>
</tr>
<tr>
<td>WiTrack [3]</td>
<td>UWB ToF</td>
<td>31 cm</td>
<td>39 cm (90%)</td>
<td>16 Hz</td>
<td>No</td>
<td>Not Published</td>
<td>120 mW</td>
<td>9 cm³</td>
<td>50 m</td>
</tr>
<tr>
<td>RF-IDraw [40]</td>
<td>NB (UHF) Interferometry</td>
<td>3.6 cm (med)</td>
<td>19 cm (med)</td>
<td>At most 53 Hz</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>8 cm³</td>
<td>9 m</td>
</tr>
<tr>
<td>PolyPoint [20]</td>
<td>UWB ToF</td>
<td>9 cm (med)</td>
<td>39 cm (med)</td>
<td>140 cm (90%)</td>
<td>16 Hz</td>
<td>No</td>
<td>Not Published</td>
<td>75 mW</td>
<td></td>
</tr>
<tr>
<td>Harmonium [21]</td>
<td>UWB TDoA</td>
<td>9 cm (med)</td>
<td>31 cm (90%)</td>
<td>19 Hz</td>
<td>No</td>
<td>2.4 m/s</td>
<td>75 mW</td>
<td>1.5 cm³</td>
<td>78 m</td>
</tr>
<tr>
<td>Chronos [39]</td>
<td>Bandstitched UWB ToF</td>
<td>Not Published</td>
<td>65 cm (med)</td>
<td>170 cm (90%)</td>
<td>12 Hz</td>
<td>No</td>
<td>Not Published</td>
<td>2.7 cm³</td>
<td>Not Published</td>
</tr>
<tr>
<td>SurePoint</td>
<td>UWB ToF</td>
<td>12 cm (med)</td>
<td>29 cm (med)</td>
<td>1-12 Hz</td>
<td>Yes</td>
<td>at least 2.4 m/s</td>
<td>280 mW</td>
<td>3 cm³</td>
<td>50 m</td>
</tr>
</tbody>
</table>

Table from: SurePoint: Exploiting Ultra Wideband Flooding and Diversity to Provide Robust, Scalable, High-Fidelity Indoor Localization. Benjamin Kempke, Pat Pannuto, Bradford Campbell, and Prabal Dutta. SenSys'16.

Lowest power UWB is 75 mW
Backscatter renaissance is redefining low-power for wireless

- **Wireless communication from W—mW to μW—nW**
  - Zhang, Pengyu, Jeremy Gummeson, and Deepak Ganesan. "Blink: A high throughput link layer for backscatter communication." *MobiSys’12*
  - Kellogg, Bryce, Aaron Parks, Shyamnath Gollakota, Joshua R. Smith, and David Wetherall. "Wi-Fi backscatter: Internet connectivity for RF-powered devices." *SIGCOMM’14*
  - Ma, Yunfei, Nicholas Selby, and Fadel Adib. "Minding the billions: Ultra-wideband localization for deployed RFID tags." *MobiCom’17*
  - Varshney, Ambuj, Oliver Harms, Carlos-Perez Penichet, Christian Rohner, Frederik Hermans, and Thiemo Voigt. "LoRea: A Backscatter architecture that achieves a long communication range." *SenSys’17*
  - Carlos Pérez Penichet, Claro Noda, Ambuj Varshney, Thiemo Voigt. “Battery-Free 802.15.4 Receiver” *IPSN’18*
UWB can transmit 54 million times less power than traditional narrowband devices

- 3-10 GHz UWB $\rightarrow$ -41.3 dBm
- 900 MHz ISM $\rightarrow$ 36 dBm
- 900 MHz unlicensed
  - Control $\rightarrow$ -13.3 dBm
  - Periodic $\rightarrow$ -21.2 dBm

Let’s think bigger about localization:
Can we locate every physical thing?

Can we make location a piece of first-class context, available to every device?