Sunflower Tool Suite Tutorial

http://www.sflr.org

Phillip Stanley-Marbell
Technische Universiteit Eindhoven

Diana Marculescu
Carnegie Mellon University

Welcome!

- Kindly pick up a welcome packet
  - DVD
    1. Simulator and benchmarks source tree, simulator manual
    2. Simulator GUI source tree
    3. Hardware platform designs
    4. PDF of tutorial guide
  - T-shirt
  - Printed tutorial guide

- Tutorial guide
  - Contains an outline of tutorial
  - Some space for taking notes on particular topics

- Anonymous tutorial questionnaire
  - Page 11 (penultimate sheet) of tutorial guide
  - Kindly detach, fill it out and hand it in at the end of the tutorial

Outline

- Motivation

- The Sunflower Simulator

Break (10:10 - 10:40)

- The Sunflower Hardware Platforms

- Summary, Q & A

Motivation

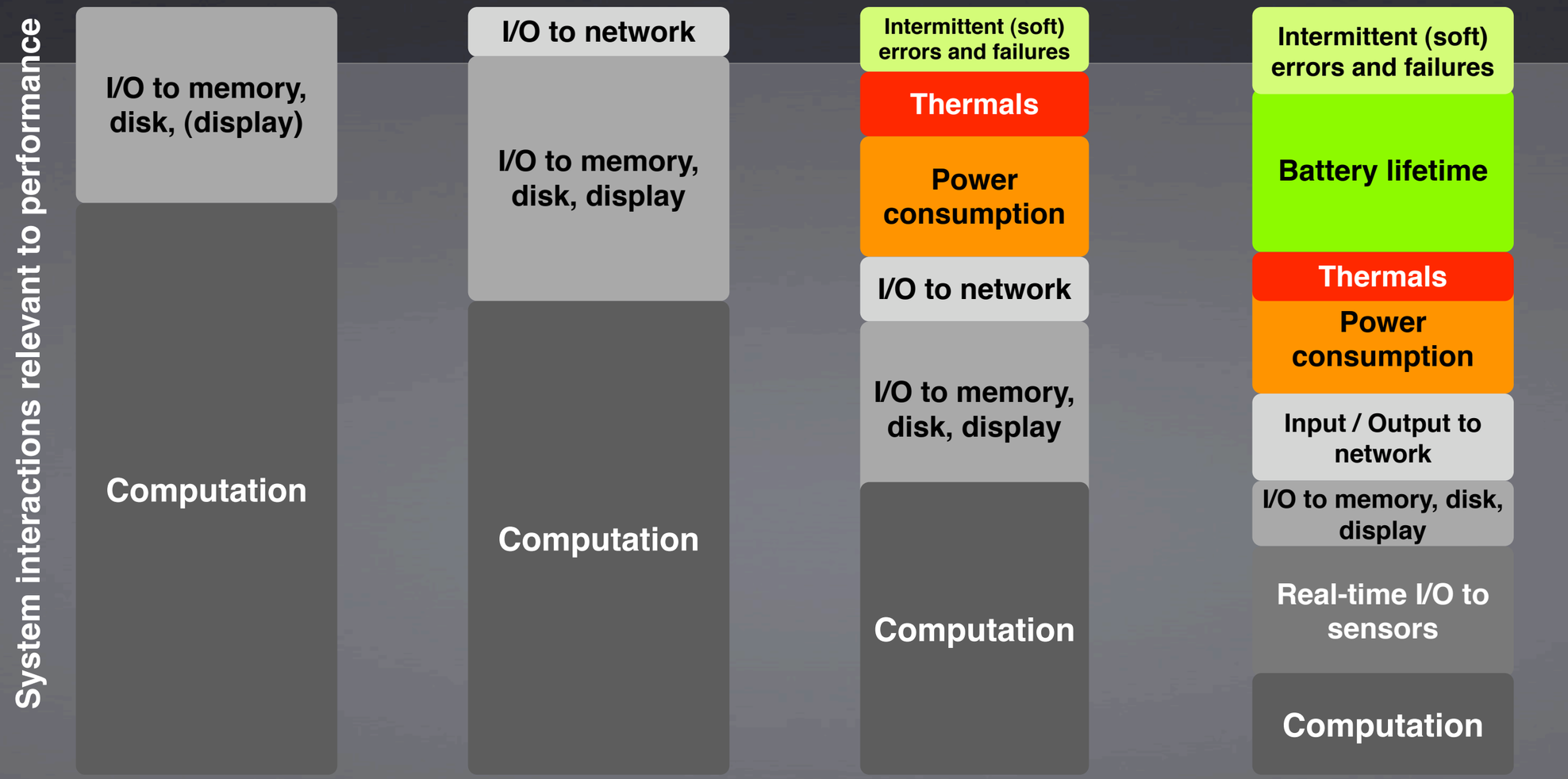
- Tools in computing systems research
  - Architectural simulators, analytic models, hardware prototypes, etc.
  - Higher abstraction layer models/tools often easier to deal with
  - Lower abstraction layer tools (closer to hardware) usually more accurate!

- High- and low-level models are complementary
  - High-level models often permit flexibility, OK for comparison studies
  - Hardware may be regarded as the ultimate truth-test, but it's often not flexible

- Can we strive to bridge this gap?
  - Calibrate high-level models (μ-arch simulators, etc.) with hardware measurements
  - Evolving the software simulators and hardware together

- *But, what do we want to use the tools to predict ?!*

What is relevant to be modeled?



Mainframes (e.g., IBM 704): Predominantly batch processing, batch I/O

Minicomputers (e.g., PDP-1): Timesharing, interactive I/O

Personal Computers, workstations, servers: Interactive and networked I/O, power and thermal concerns

Networked Embedded Computing Systems: Real-time interactive I/O with analog signals in environment, networked, often battery powered.

Figure is not to scale!

Target Application Sub-Domain

- Computing systems research is a broad field
  - Server architectures, workstations, mobile, embedded, ...

- The target domain here is "networked embedded systems"
  - For lack of a more accurate term...
  - Energy and computing resource constrained
  - Computation interacts with (drives, is driven by) analog signals in environment
  - Multiple systems communicating, or multiple processors per system
  - May be deployed in environments which make them susceptible to failures

- Some challenges in modeling (networked) embedded systems
  - Modeling computation across multiple (networked) processors
  - Modeling the interaction between computation and analog signals
  - Predicting absolute perf., power: e.g., can system process signals in real-time?

The Sunflower Tool Suite

- Goal
  - To develop a complementary set of hardware prototypes and simulation tools
- Target users
  - Micro-/system-architecture researchers
- Hardware (HW) ↔ software (SW) synergy
  - SW benefits by having a HW platform from which to obtain calibration data
  - Hardware benefits by having a SW tool for exploratory research
- Key points
  - Hardware platforms and simulator evolved as a single system; both open source
  - Regular HW revisions, and non-hardware people may request HW features
  - Characterization of hardware made available to calibrate simulation

What will you get out of this tutorial ?

- An overview of the Sunflower simulator
  - What it is, what it is useful for
  - How it works
  - How you can use it in your research
- An introduction to the Sunflower hardware platforms
  - Hardware capabilities
- An understanding of how you can probe further
  - Resources such as web pages/Wiki/bug tracking, mailing list

Outline

- Motivation

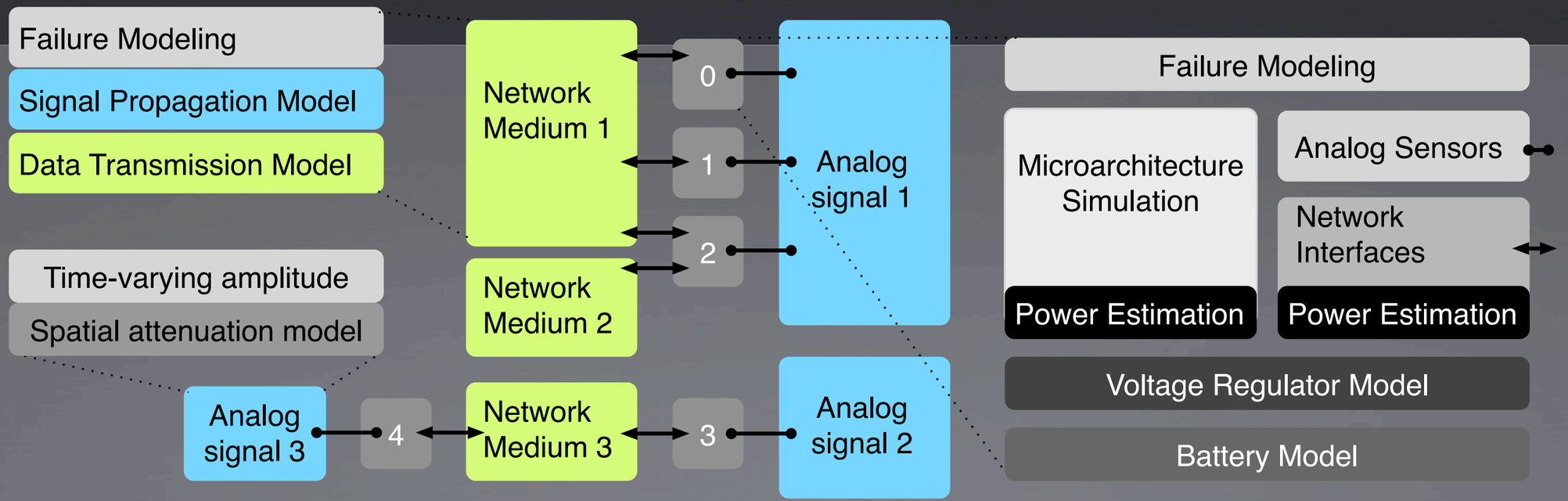
- The Sunflower Simulator

Break

- The Sunflower Hardware Platforms

- Summary, Q & A

Simulator Overview



- Sunflower is a “full-system” simulator for embedded systems
  - System/microarchitecture modeling of processors
  - Modeling networks of processing elements
  - Power estimation for computation and communication
  - Battery and voltage regulator modeling
  - Modeling analog signals in environment of computation
  - Modeling failures in network and computation

Hands-on Simple Example

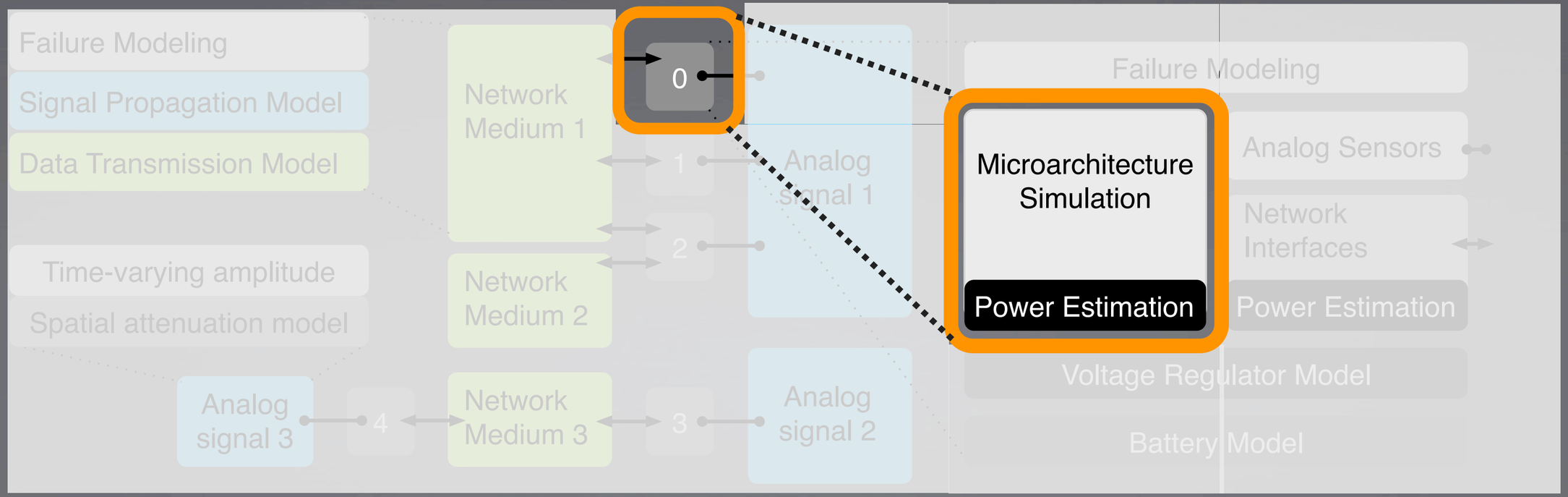
- First, lets briefly look at a trivial example, bubblesort
  - Compile with gcc for one of the target processors modeled by simulator
  - Load the compiled binary into memory
  - Run the application
  - See the output
  - Probe register contents

Hands-on Simple Example

- Observations
  - Applications compiled with a stock version of Gcc, linked against Newlib
  - Starting up the simulator, console version and GUI

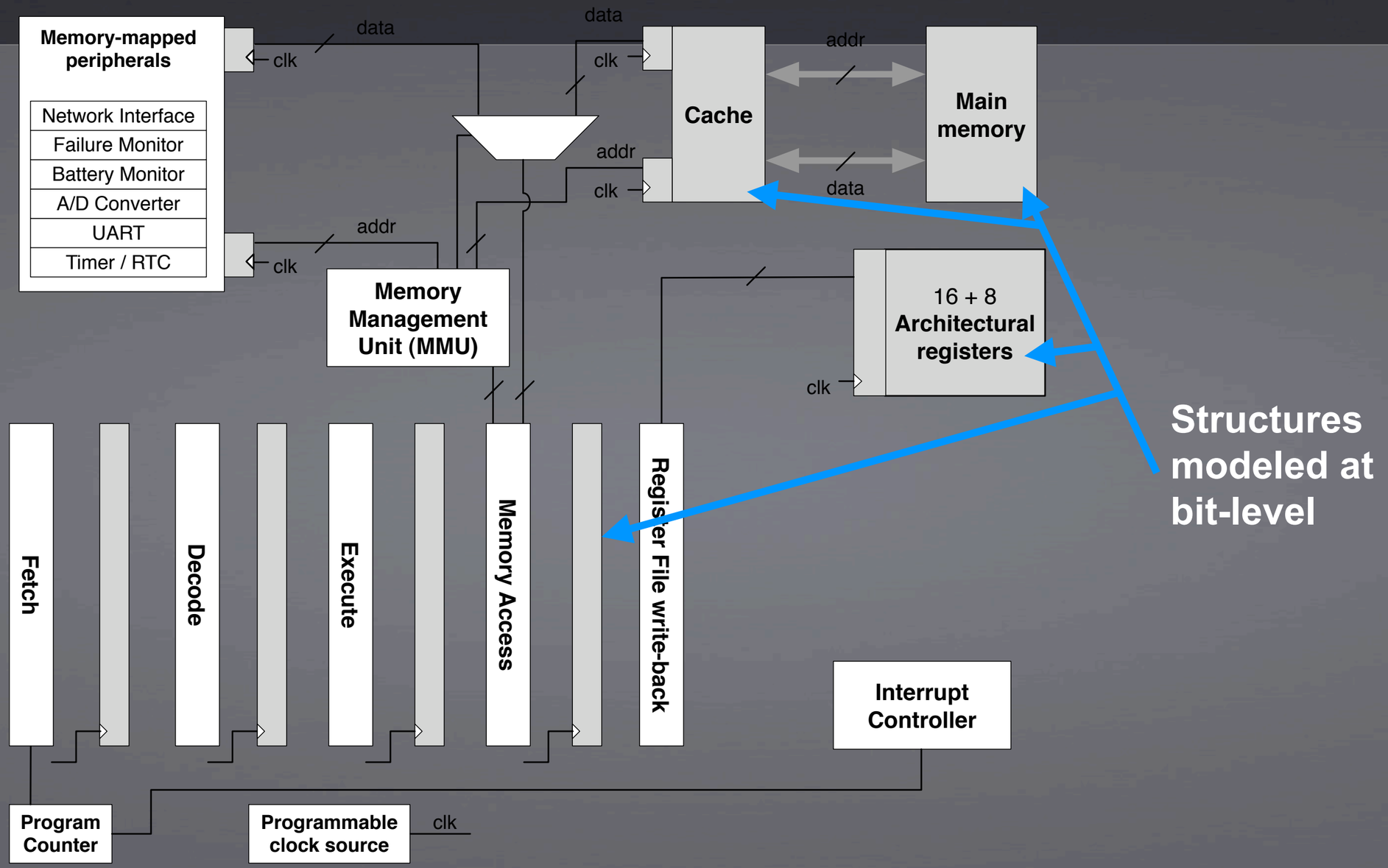
- Simulator is interactive
  - Issue commands at command prompt
  - Can probe (and modify!) simulated machine state while simulation is running
  - Can also run simulator in non-interactive mode

µ-architecture Simulation & Implementation

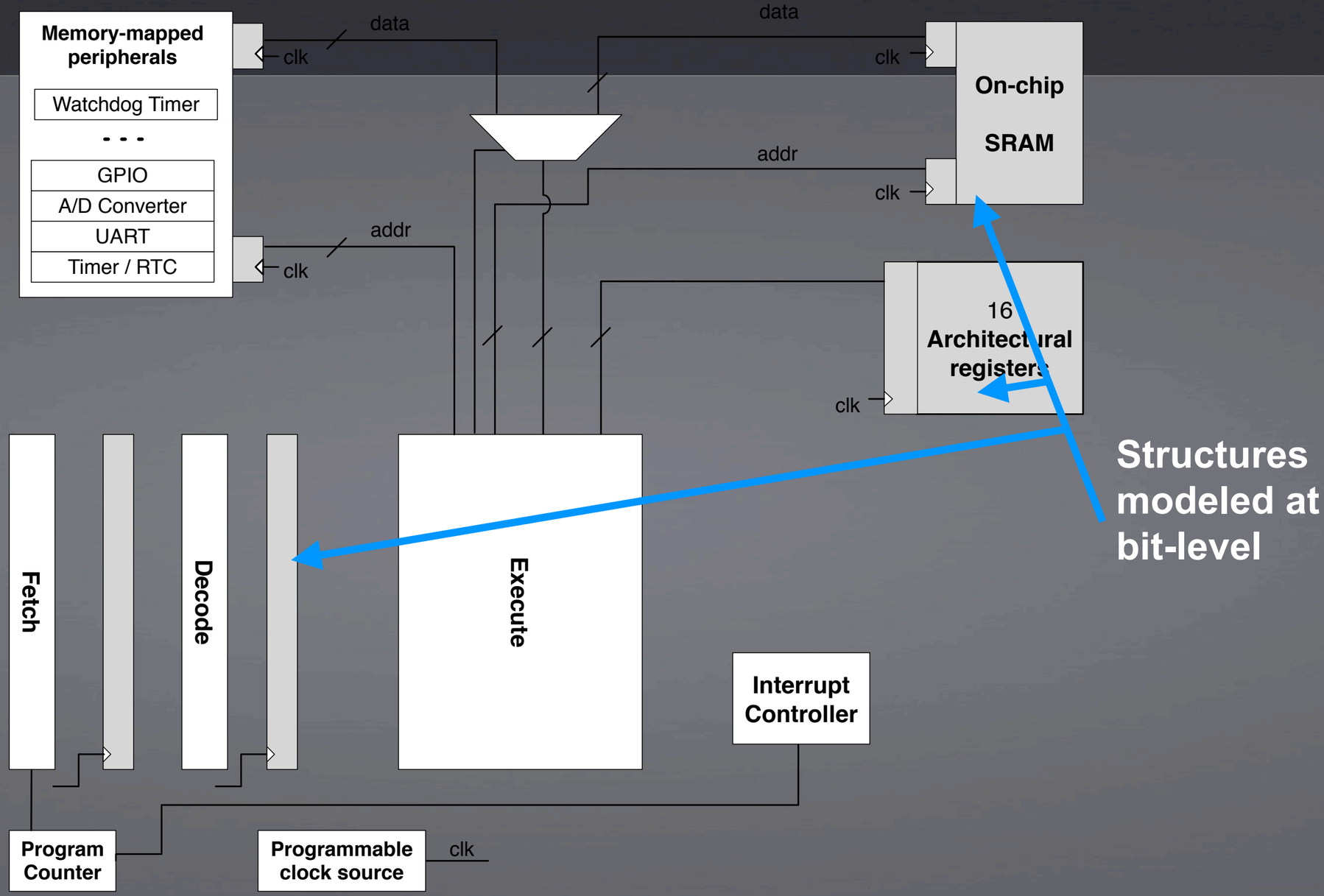


- Architecture simulation is one part of the entire system

µ-architecture Simulation: Hitachi SH



µ-architecture Simulation: TI MSP430



: Structures modeled at bit-level, enabling signal transition activity and logic upset modeling

µ-architecture Simulation: Data Structures

- Implementation maintains a data structure for each processor
  - `State` struct, defined in `sim/main.h`
  - All routines that deal with architecture modeling interact with a State structure
  - State data structure is passed as an argument to most arch-modeling routines

- Another important data structure is `Engine`
  - Holds all state relevant to a given collection of processors
  - `Engine` struct defined in `sim/main.h`

Recap

- So far we've briefly looked at modeling single processors
- Each processor has several peripherals, including
  - Analog-to-digital (A/D) converter peripherals for interacting with environment
  - Network interface peripherals for communicating with other processors
- Next
  - Simulator's modeling of environment and A/D peripherals (referred to as "sensors")
  - Modeling of interconnect networks

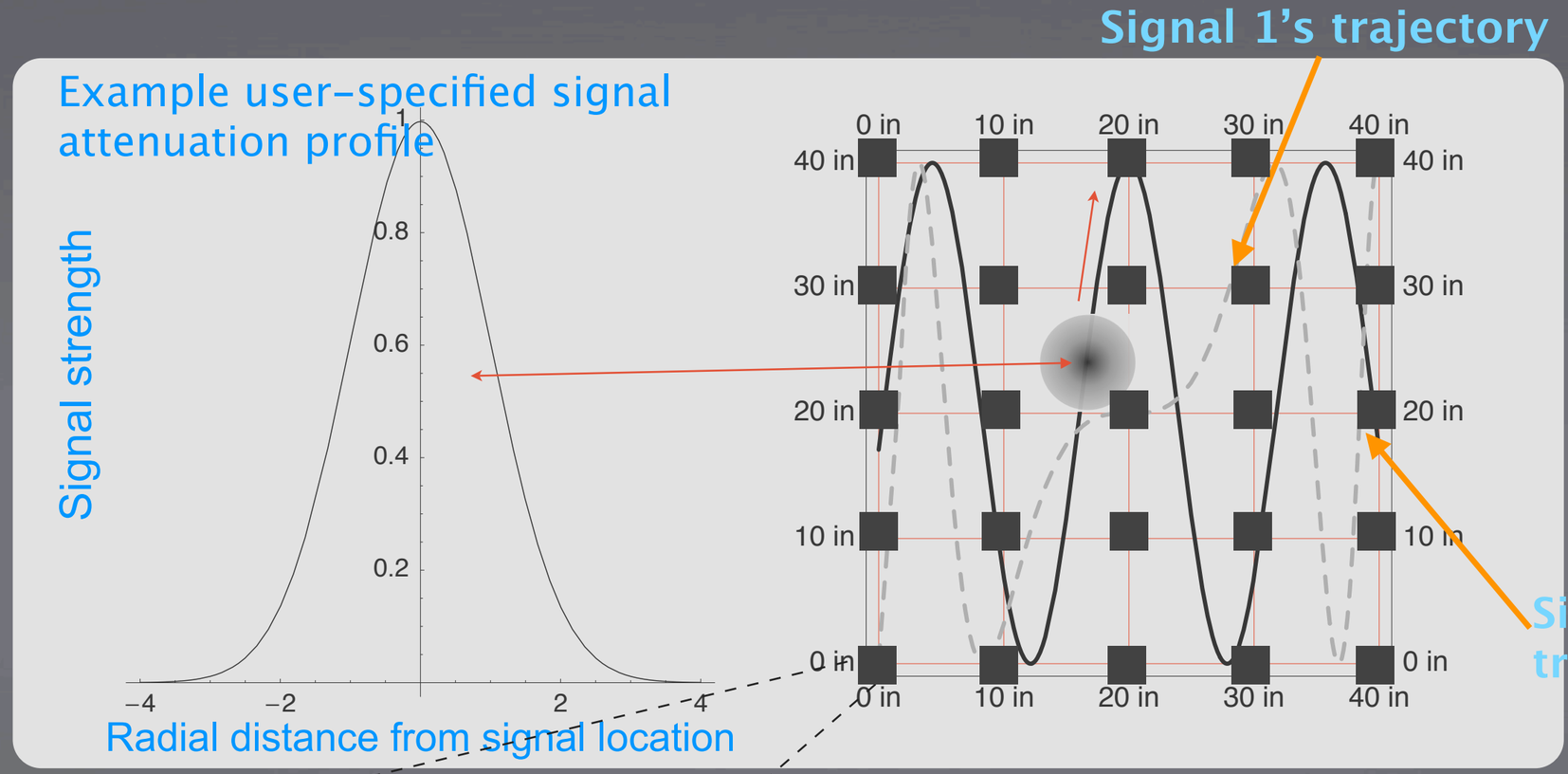
Modeling Analog Signals and Environments

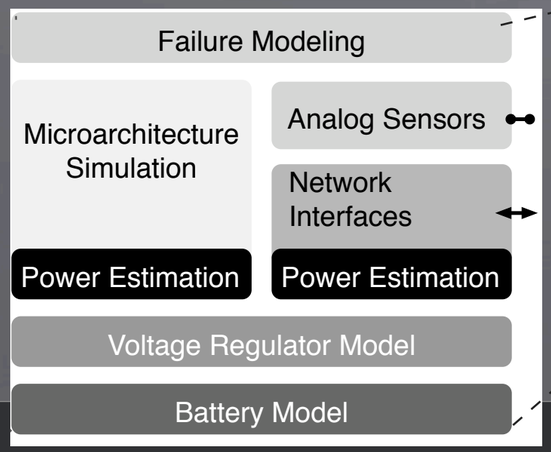
- Simulator maintains a notion of a deployment environment
  - Every simulated processor has an assigned location in 3-D space
  - Processors may have associated trajectories in space
  - Processor sensor peripherals can be “attached” to an analog signal in environment
  - Values read from sensor peripheral are based on analog signal’s location, etc.
  - Computation may also drive values onto an analog signal

- Analog signals defined by specifying two components
  - Intensity (positive or negative) and radial attenuation function
  - Location or trajectory

- Interaction between signals
  - When a sensor is “attached” to multiple analog signals, they are summed
  - Arbitrary signal intensity maps can be created in this manner

Example





Example: two signals with trajectories

Modeling Communicating Networks

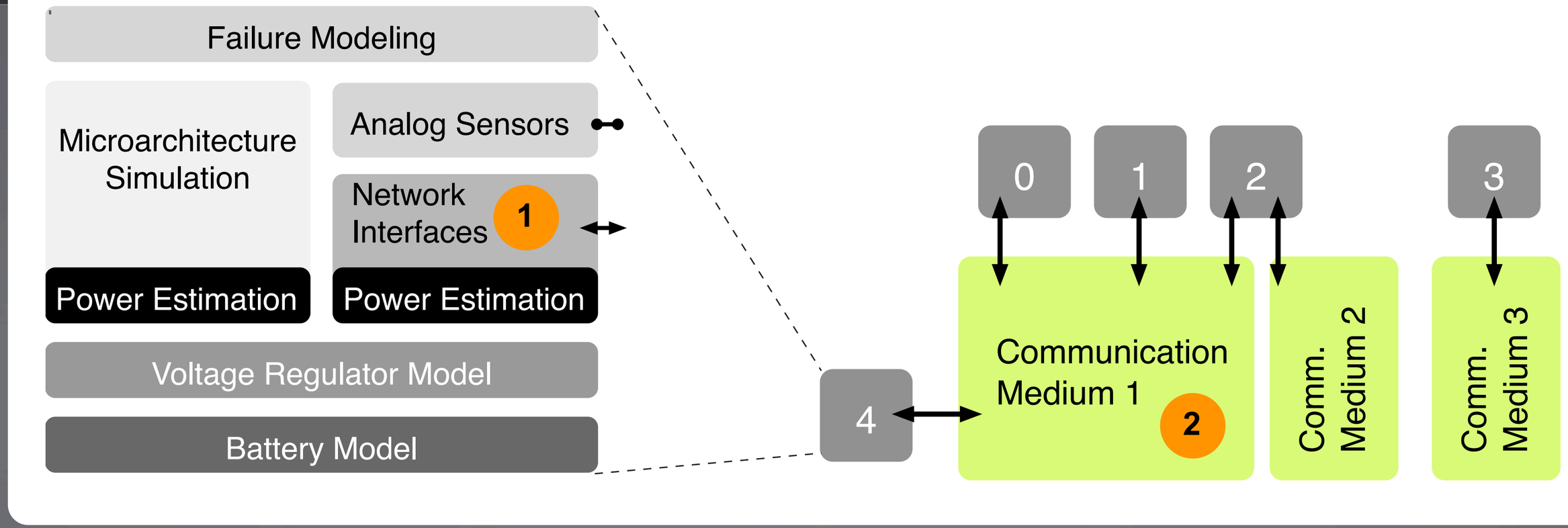
- In principle
  - Interconnect networks can be modeled with the analog signal modeling — as wires
- In practice
  - Hardware platforms often have some HW communication assist support, e.g., UARTs
- “Hardware-assisted communication” in Sunflower
  - Processors have a generic network interface peripheral (one or more interfaces)
  - This is used, in conjunction with network medium modeling

Modeling Communicating Networks

- Transmitting
  - Applications write to a set of memory-mapped registers to achieve frame transmit
  - Interface is similar to an 85C30 UART
  - A “device driver” for this peripheral is included in benchmarks distribution

- Receiving
  - Incoming communications to a processor cause the generation of interrupts
  - Applications receiving communications must have interrupt handlers installed

Computation-Driven Network Modeling

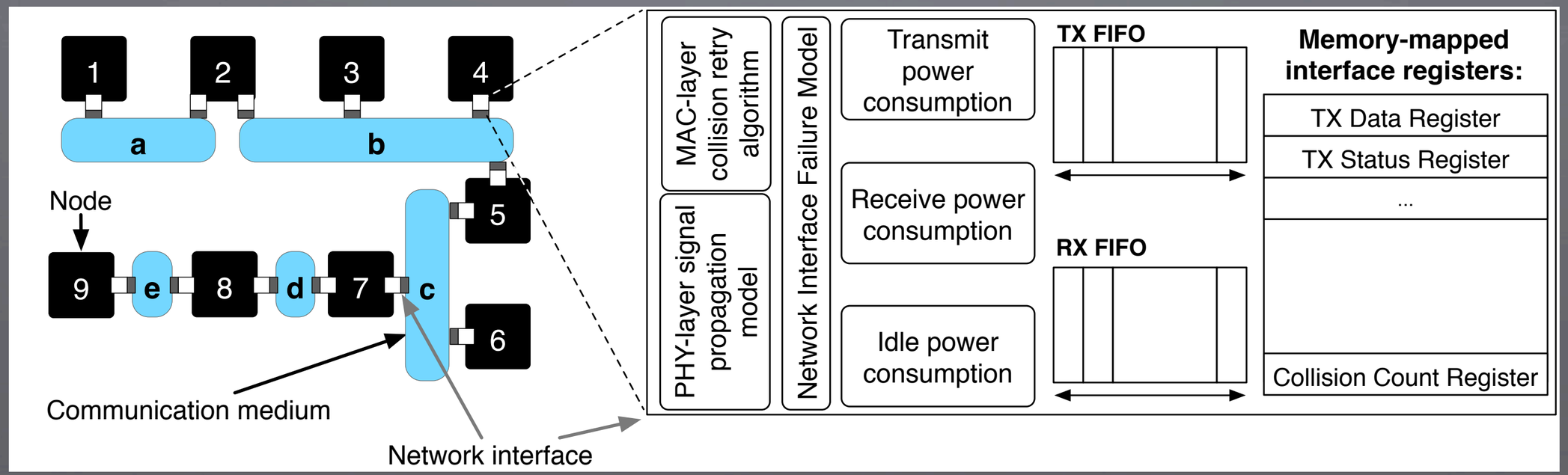


- Communication media
  - Define properties of a communication link, such as bit rate, propagation delay

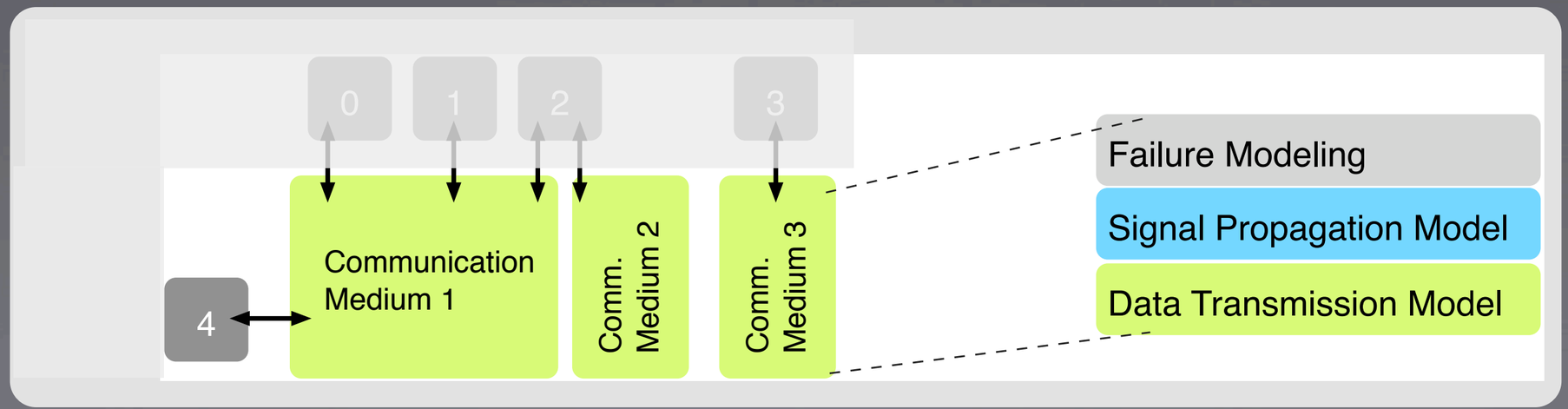
- Each processor can have multiple network interface peripherals
  - These can be attached to communication media to form arbitrary topologies

Network Interfaces

- Network interface
  - Transmit, receive, idle and listen power consumption
  - Number of TX and RX FIFO entries
  - MAC-layer collision retry algorithm choice
  - PHY layer signal propagation (we will see a concrete example later)
- Example:



Communication Media



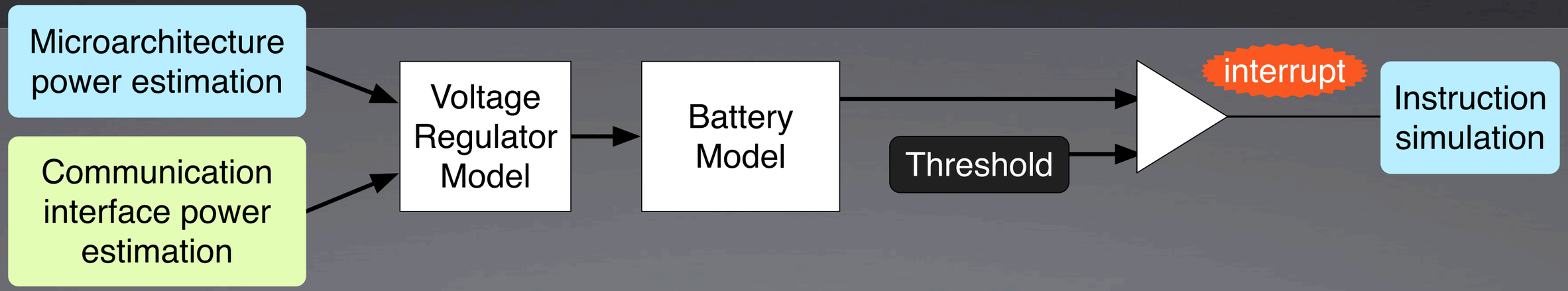
- Communication media
  - Bit rate, frame size, # of simultaneous transmissions that can be accommodated, rate and distribution of intermittent failures, signal propagation model

- Communication modeling cycle-accurate w.r.t. computation
  - Example: for a 8Mb/s network, processor at 100MHz, destination will receive 1000 byte frame after 1ms or ~100,000 instructions (depending on what is happening in μ-arch)

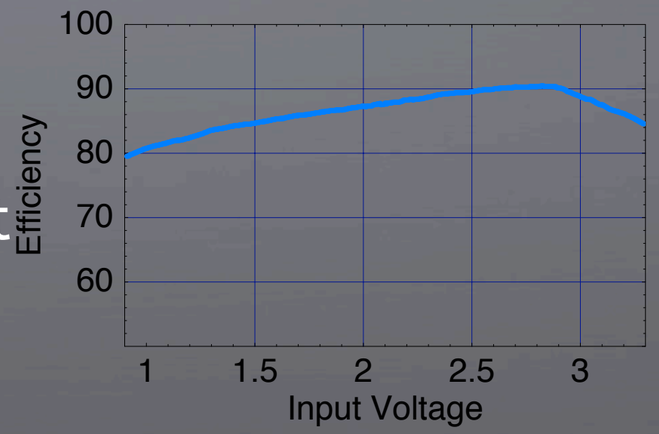
Power Estimation, Power Supply, Batteries

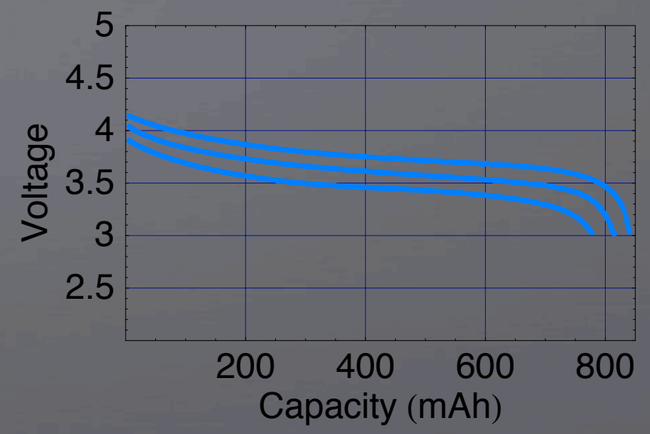
- Power estimation in Sunflower
  - Processor power during architectural modeling
  - Communication power
- Processor power estimation: three forms
  - Instruction-level model based on hardware measurements
  - Circuit activity estimation based on microarchitecture modeling
  - Coarse-grained mode-based models based on processor data sheets
- Communication Interface
  - Transmit, receive, idle and listen state power
- Total load power is fed to a battery subsystem model...

Power Estimation, Power Supply, Batteries



- Why model batteries and voltage regulators ?
  - Their nonlinear properties can have a large effect on system lifetime
- Voltage regulators
  - Efficiency is a function of load current



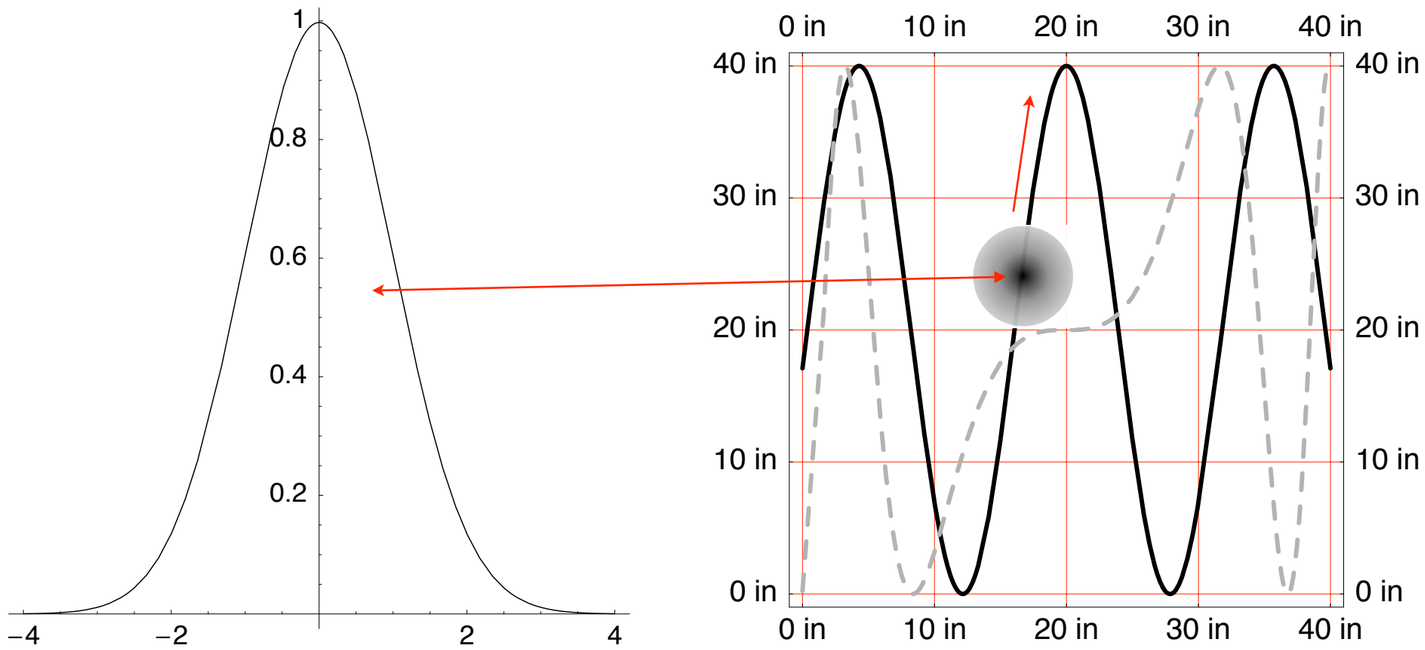


- Batteries
  - Effective lifetime depends on, temporal load current profile, temperature, etc.

Recap

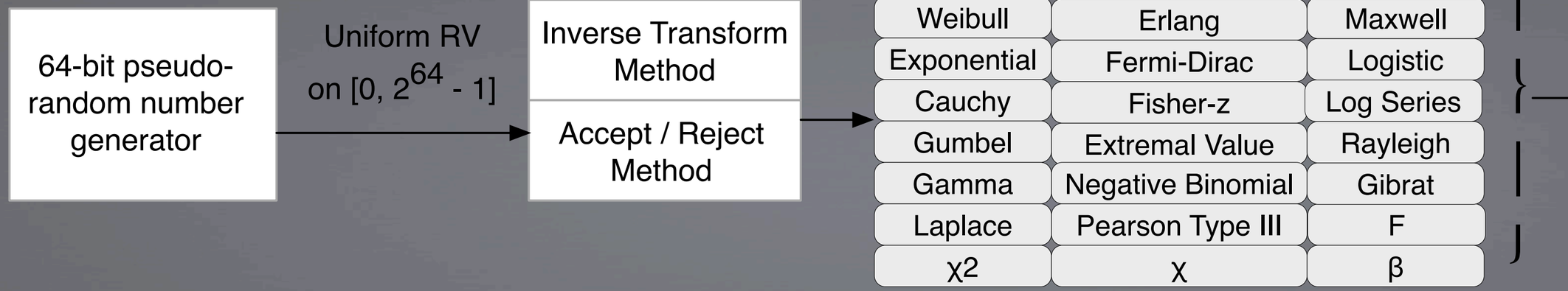
- We've seen
  - Single-processor modeling in Sunflower
  - Environment/analog signal modeling
  - Communication network modeling
  - Power estimation
- Putting it all together
  - An example application that executes over a network of embedded systems
  - Communication network is configured to model a wireless network
  - The application executing on nodes implements a clustering algorithm
  - Clusters formed based on intensity of sensed signals, which move in space
  - Application executes over an implementation of a software-based MAC layer

Large example: benchmarks/sources/DAM+S-MAC/



- System configuration
  - Three analog signals: two moving, one background noise; Gauss attenuation profile
  - 25 embedded systems
  - Communication handled by a software medium access control (MAC) protocol
  - MAC-layer frames passed to clustering algorithm
  - Communication medium configured as shared, wireless properties

Stochastic Distributions



Random Variables from some common distributions

Modifying the Simulator

- Simple example: adding a new command/primitive
  - Files: `lex.c`, `sf.y`
  - Auto-generated built-in help
  - Autogenerated LaTeX for manual

Benchmark Suites

- ALPBench
  - `benchmarks/source/ALPBench`
- MiBench
  - `benchmarks/source/MiBench`
- A sensor network benchmark suite
  - `benchmarks/source/sbench`

Recap

- Computing systems research and system modeling
  - Analytic models, behavioral models/arch. simulation, and hardware prototyping
  - One would like to validate and calibrate models against HW implementations

- The Sunflower Tool Suite
  - A suite of hardware and software tools for modeling resource-constrained systems

- Tutorial thus far:
  - Simulator

- Next:
  - A short break!
  - Hardware platforms

Break

Outline

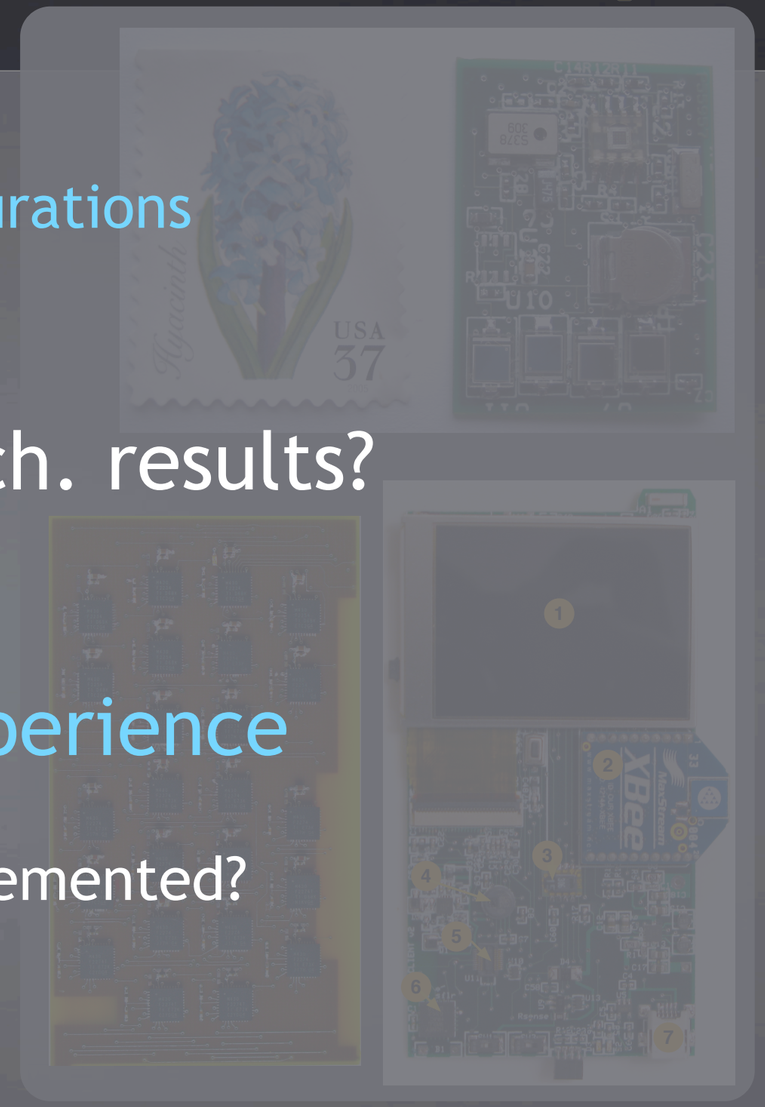
- Motivation
- The Sunflower Simulator
- *Break*
- The Sunflower Hardware Platforms
- Summary, Q & A

Hardware Platforms Overview

- Thus far we've talked about simulation
  - Simulation provides flexibility to try many system configurations
  - Not all system configurations are valid in the real world!

- How can we gain confidence in simulated arch. results?
  - One answer is by calibrating simulation to real hardware

- Not all researchers have hardware design experience
  - Use an existing commercial or academic platform
  - But what if you want a hardware feature that is not implemented?

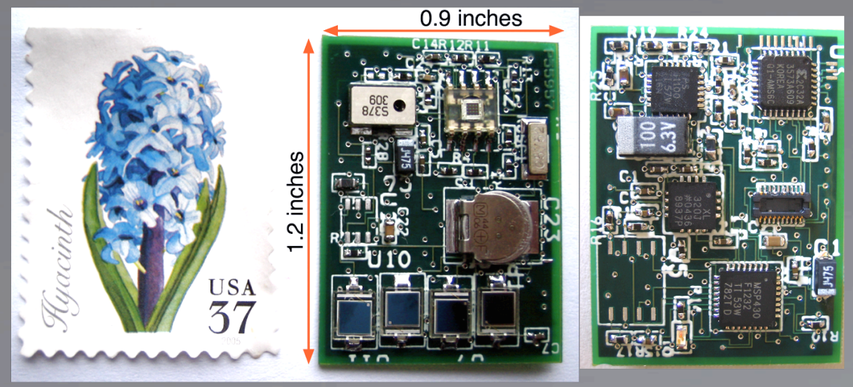


Sunflower Hardware Platforms

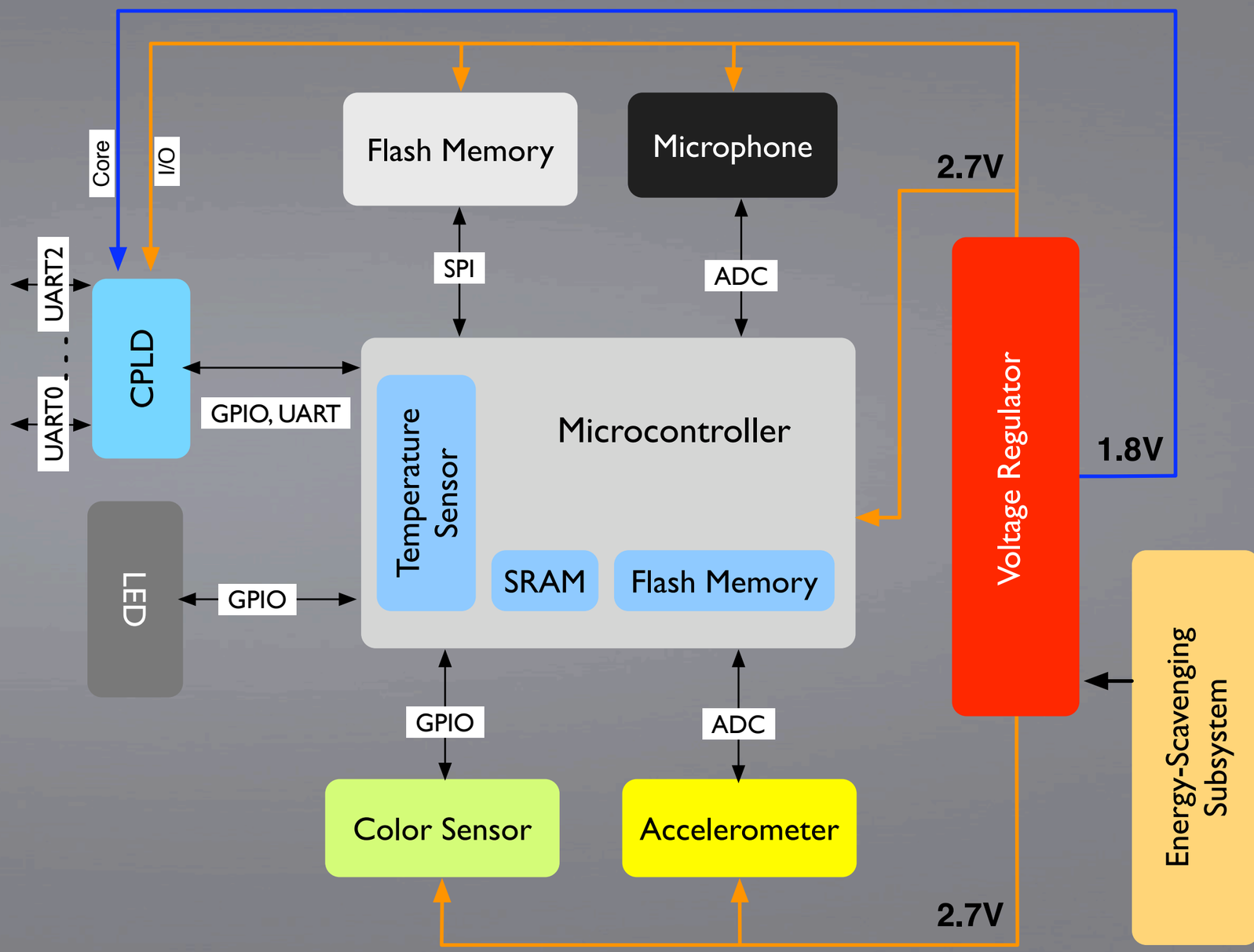
- Goal
  - Provide a suite of hardware platforms relevant to various embedded systems areas
  - To enable these platforms to be evolved in much the same way as SW
- Approach
  - A set of three hardware platforms relevant to 3 important embedded systems areas
  - Hardware designs are made freely available
  - Updates to hardware made available on a regular schedule
  - Researchers can request/suggest/provide features to be included in future revisions
  - Tools setup to make it easy for non-experts to obtain instances of HW revisions

Sunflower Sensor Node Platform

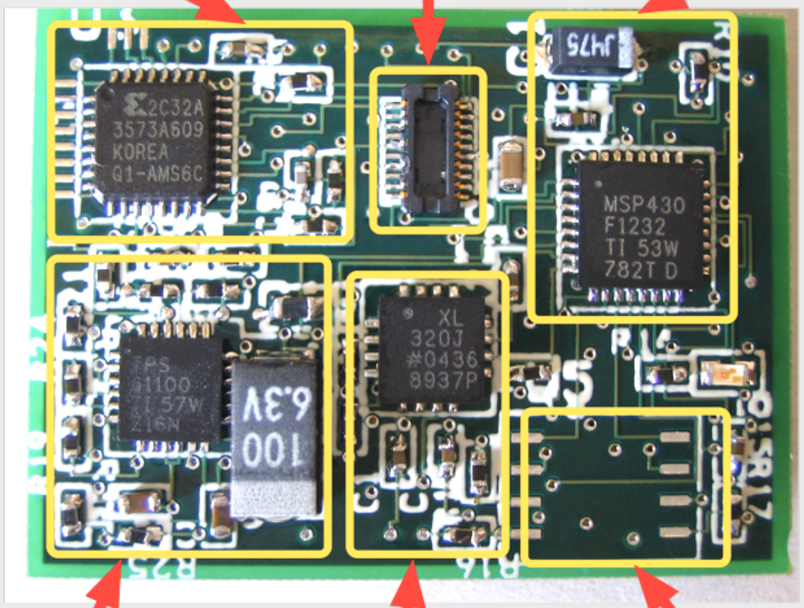
Size Comparison (0.9”x1.2”)



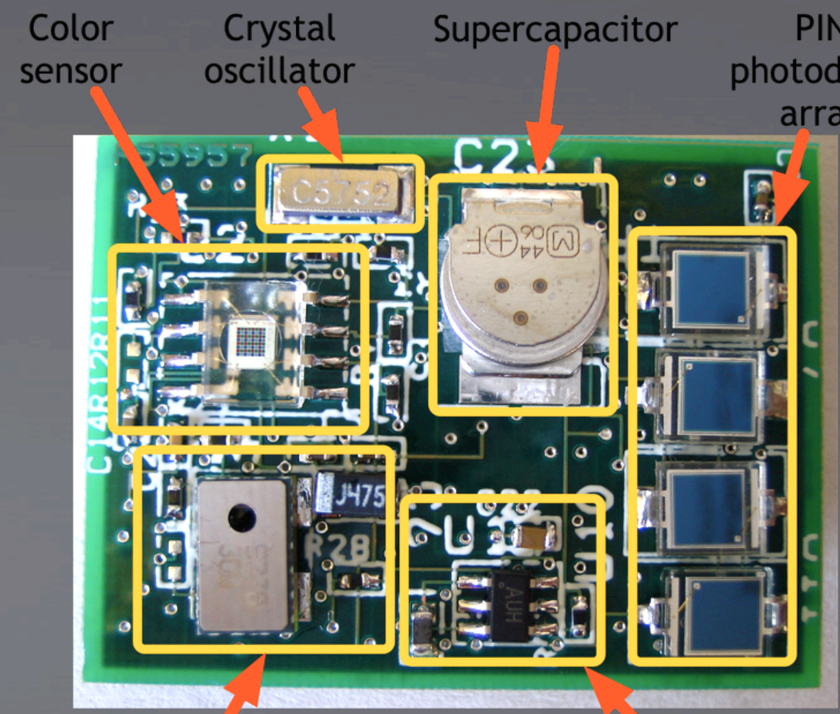
“An 0.9x1.2 inch, Low Power, Energy Harvesting System with Custom Multi-Channel Communication Interface”, P. Stanley-Marbell and D. Marculescu. DATE 2007



Sunflower Sensor Node Platform

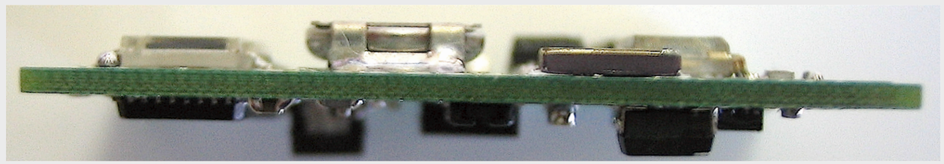


Bottom Surface



Top Surface

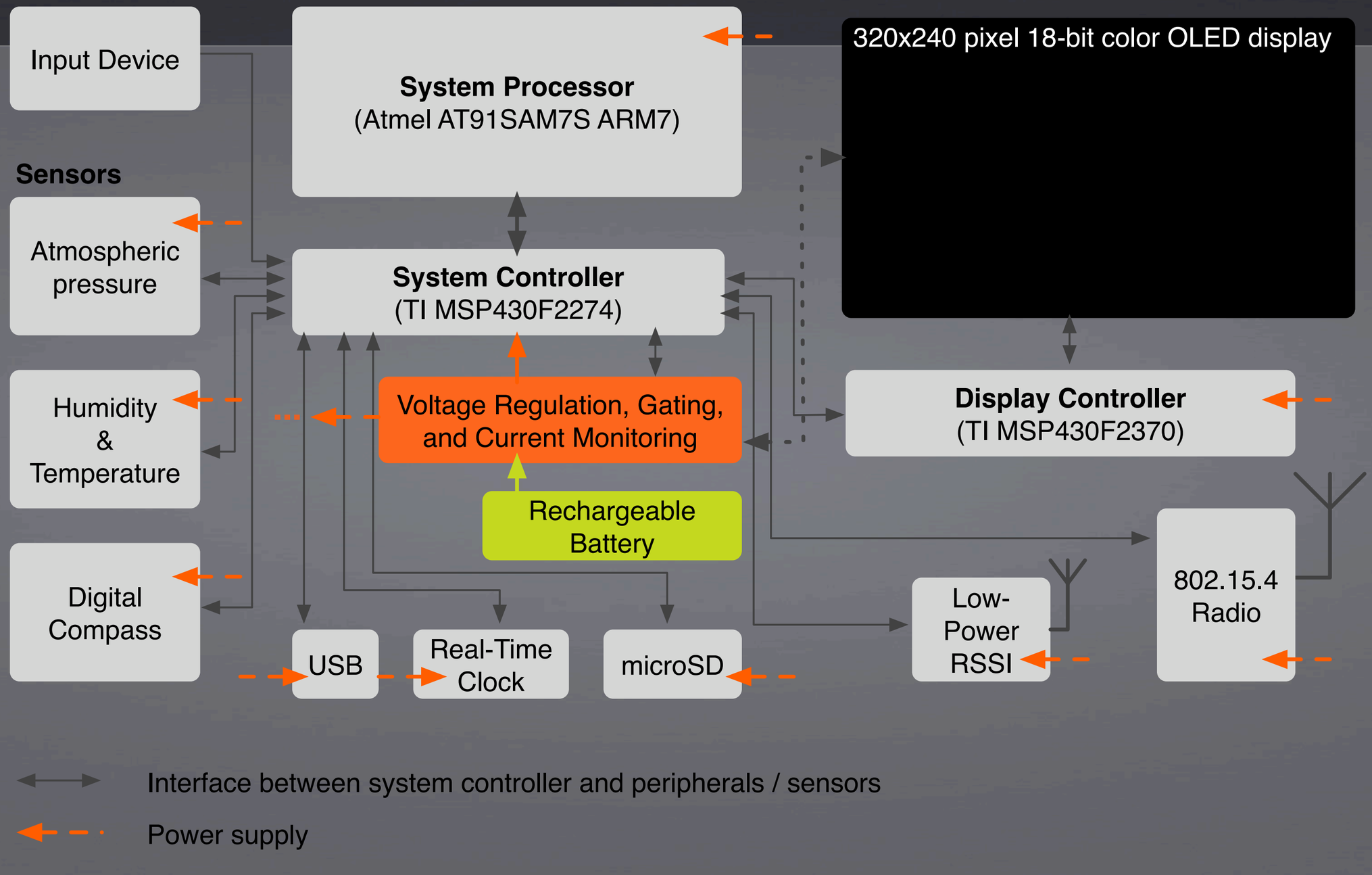
Side View:
(complete system, including energy source)



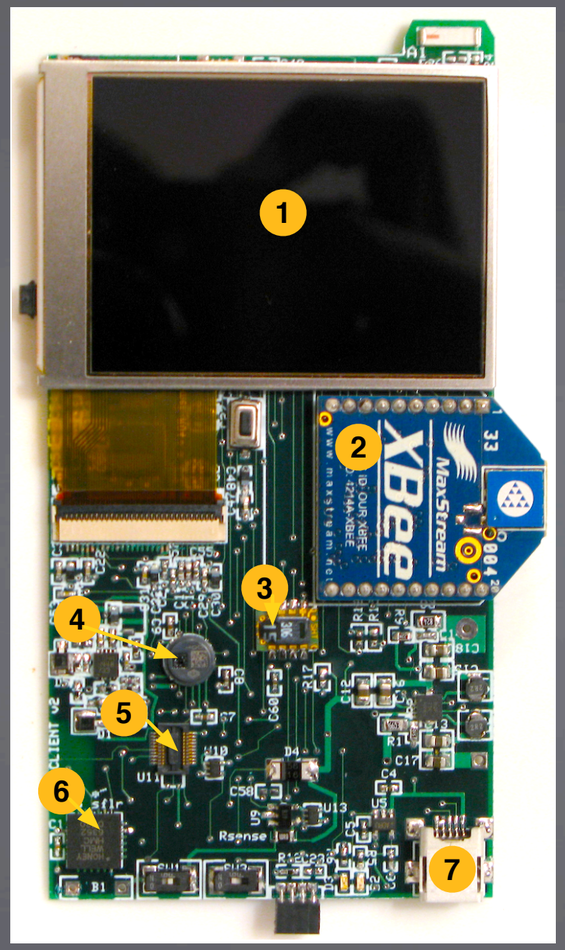
Sunflower Sensor Node Platform

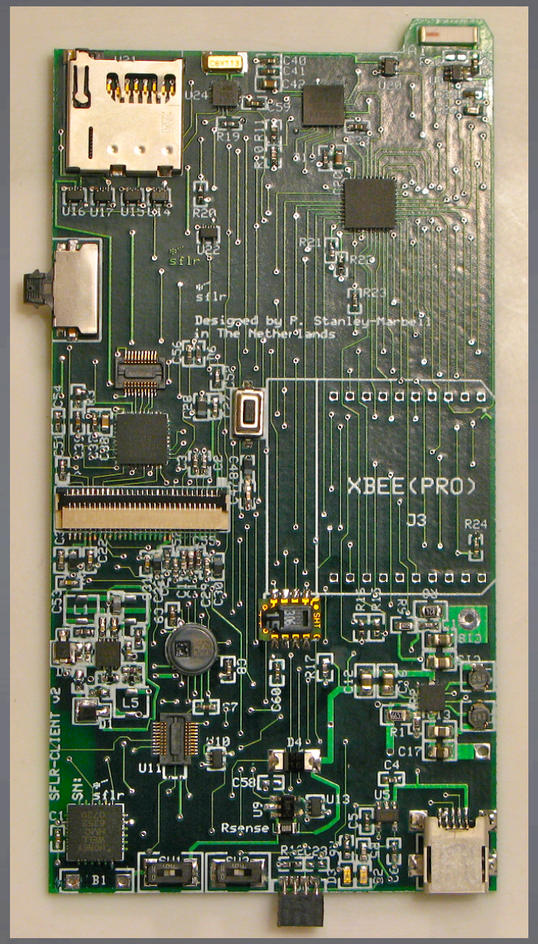
- Research facilities
  - Integration of energy-scavenging facilities onto circuit board
  - System microcontroller can monitor energy store level to implement adaptive algs.
- Platform evolution directions
  - Integration of more efficient energy-scavenging subcircuits
  - More energy-efficient system components
- Simulator calibration
  - Develop simulator configuration files and calibration data to match platform
  - Verify simulated instances against hardware in period revisions

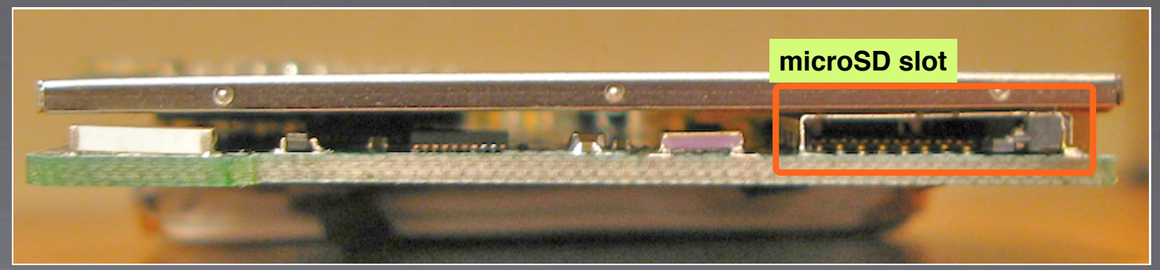
Sunflower Handheld Platform

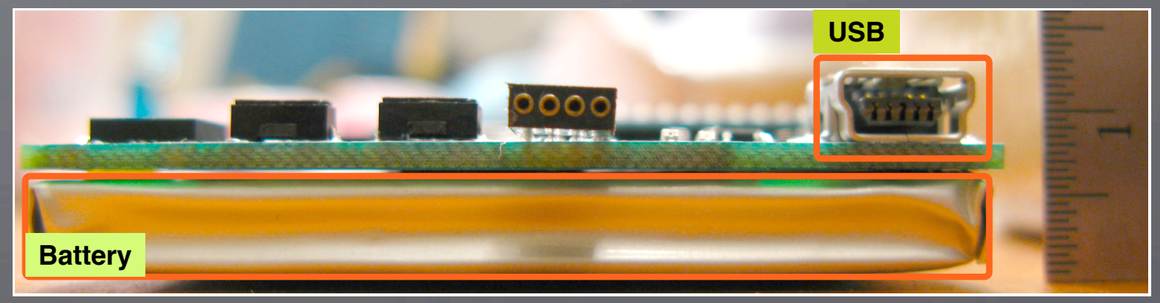


Sunflower Handheld Platform





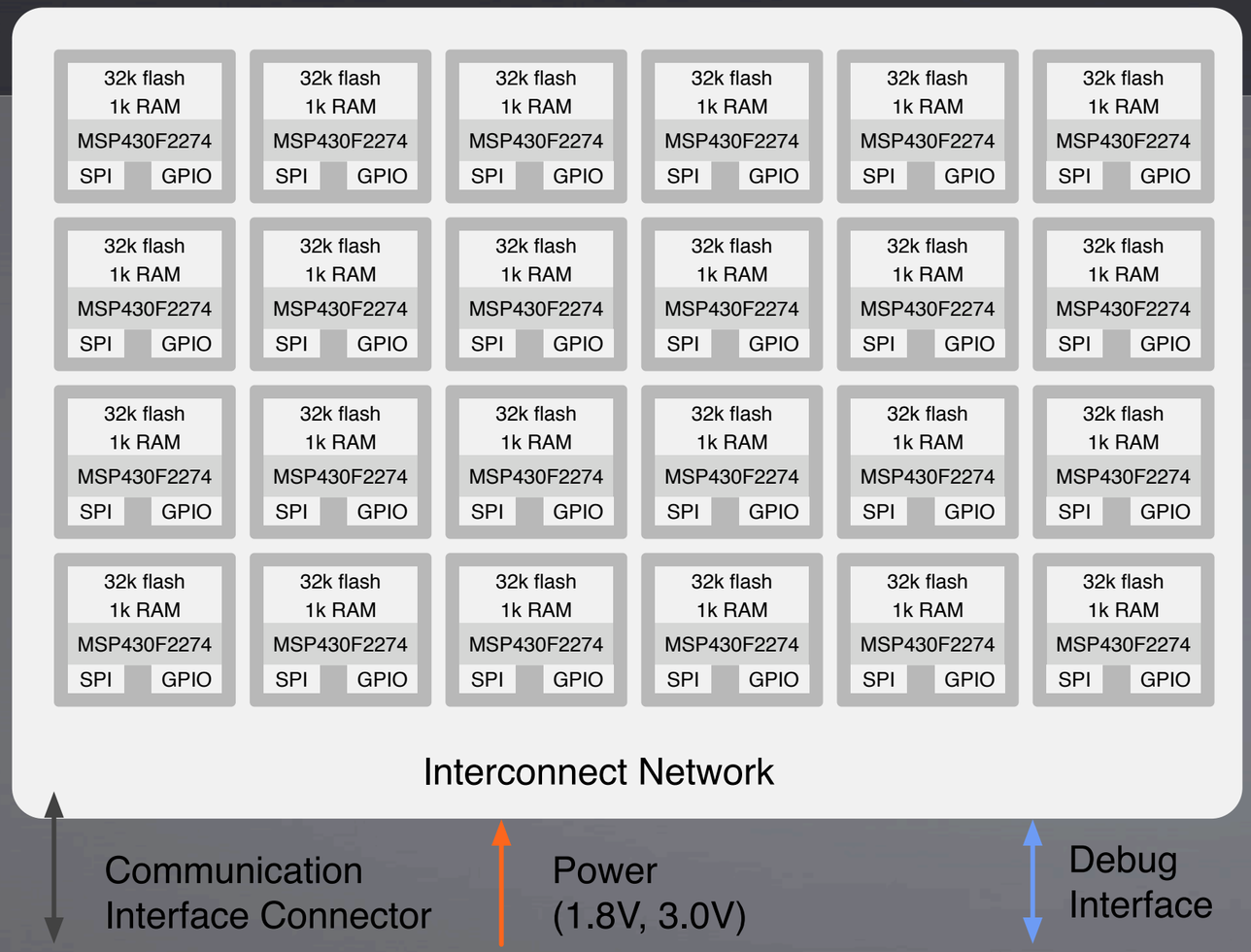


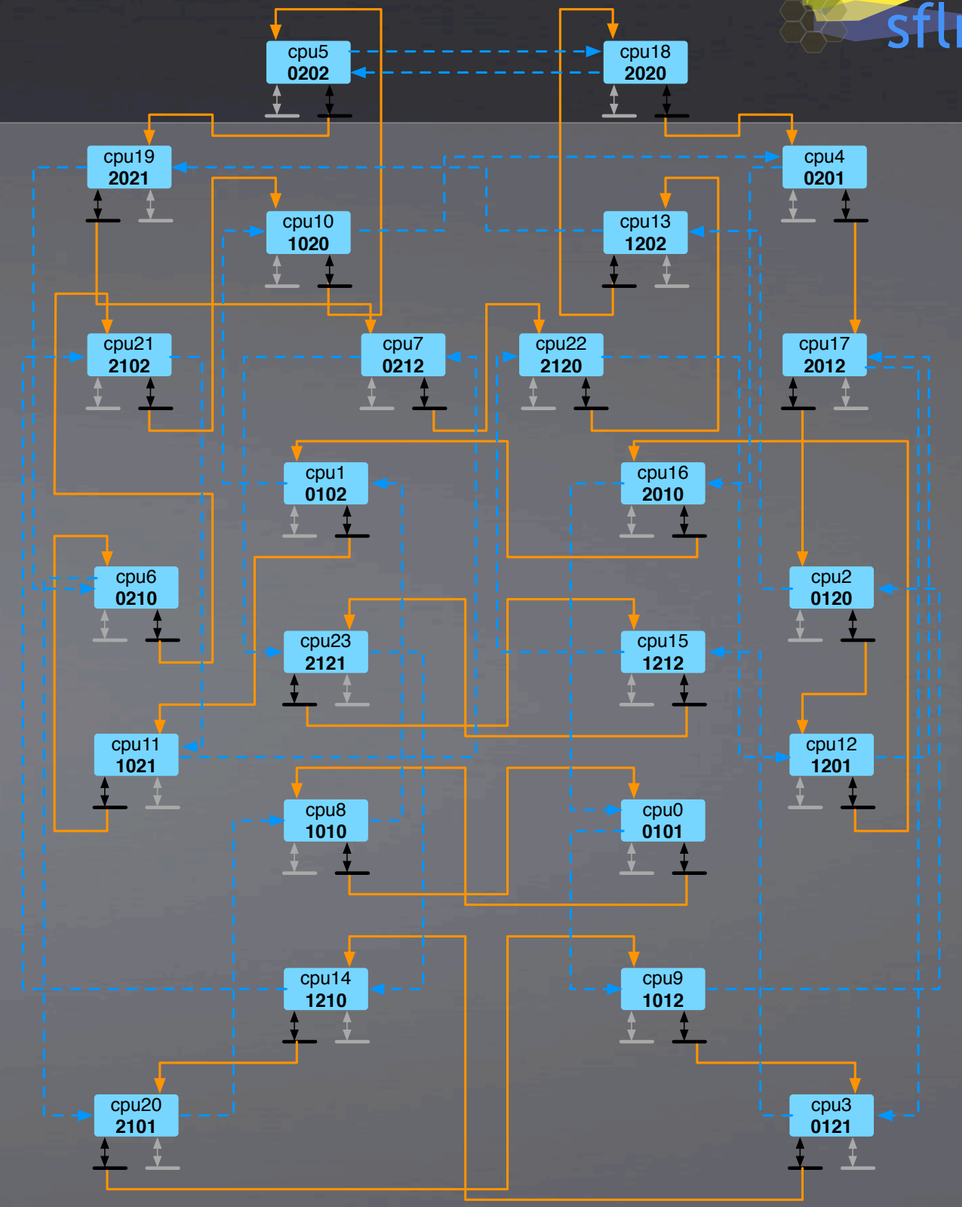


Sunflower Handheld Platform

- Research facilities
  - Software-controlled power-supply gating for selected peripherals
  - System facilities implemented across four processors
  - Low-power radio channel monitoring hardware (RSSI), in addition to comm. radio
  - Low-power OLED display: no backlight, power $\propto$ to image characteristics
- Platform evolution directions
  - Automatic hardware-controlled power supply gating
  - More efficient radio power / channel monitoring circuit
- Simulator calibration
  - Develop simulator configuration files and calibration data to match platform
  - Verify simulated instances against hardware in period revisions

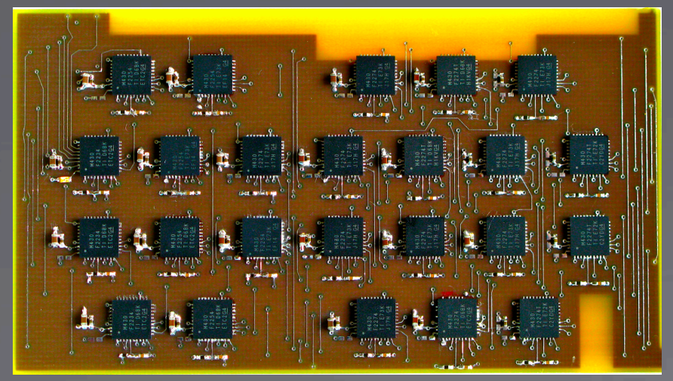
Sunflower Embedded Multi-Processor Module

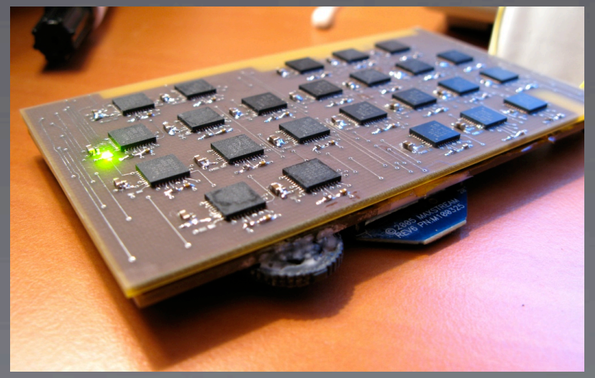


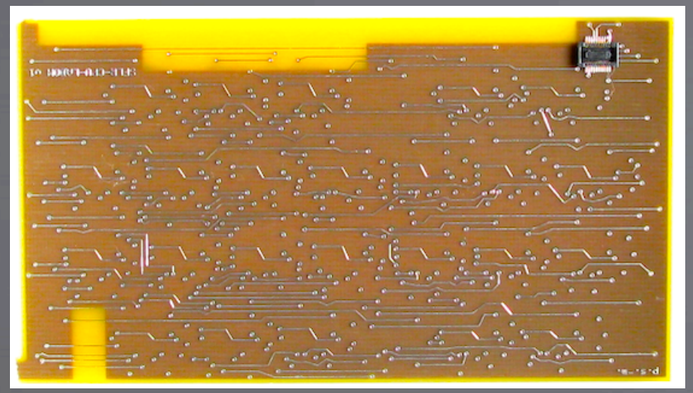


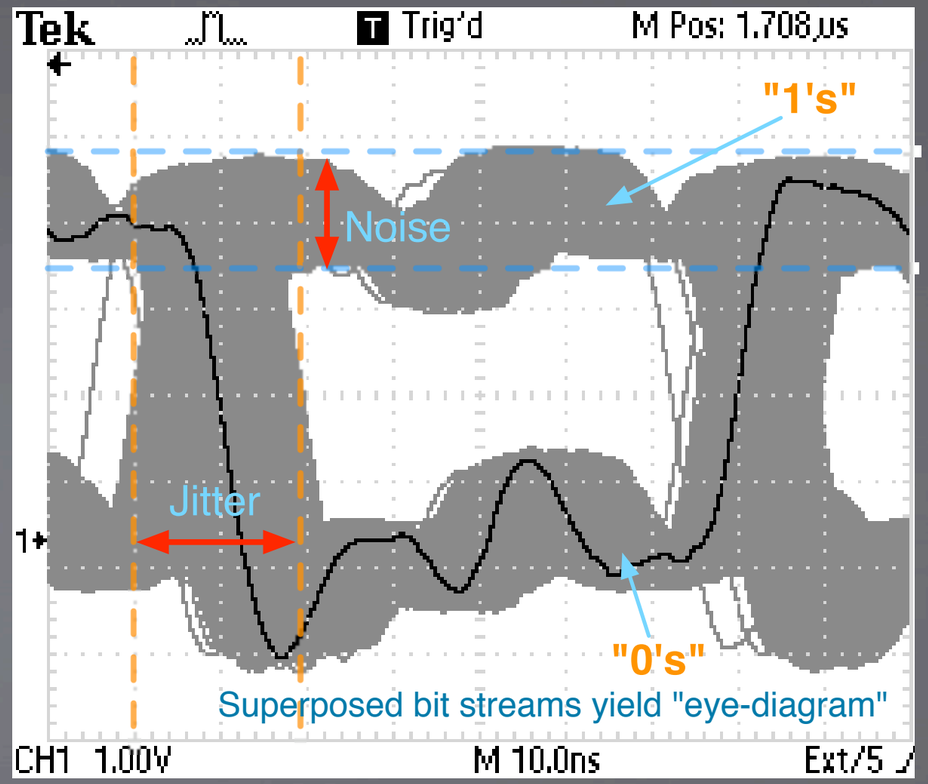
- System contains 24 low-power uCs
  - Connected in a low-diameter Kautz topology
  - Low idle power — < 3uW
  - (Theoretical) peak perf. of 384 MIPS

Sunflower Embedded Multi-Processor Module









Hardware-measured “Eye Diagram” at 8 Mb/s

- Plugs into rear of Sunflower handheld platform
- Can also be used as independent processor module

Sunflower Embedded Multi-Processor Module

- Research facilities
  - Large number of processors
- Platform evolution directions
  - Better impedance matching on interconnect
  - Smaller implementation size for same number of processors
- Simulator calibration
  - Simulator configuration files for topology
  - Simulated instances to be verified against hardware in period revisions

Hardware Tools

- Sunflower client and multiprocessor module programmed with EZ430-USB tool from TI

- Sunflower sensor platform also programmed with USB JTAG (FET430UIF) tools from TI

Compilation / Software Tools

- Hardware currently tested using IAR Kickstart for MSP430
- Free MSP430 port of GCC (msp430-gcc) also available

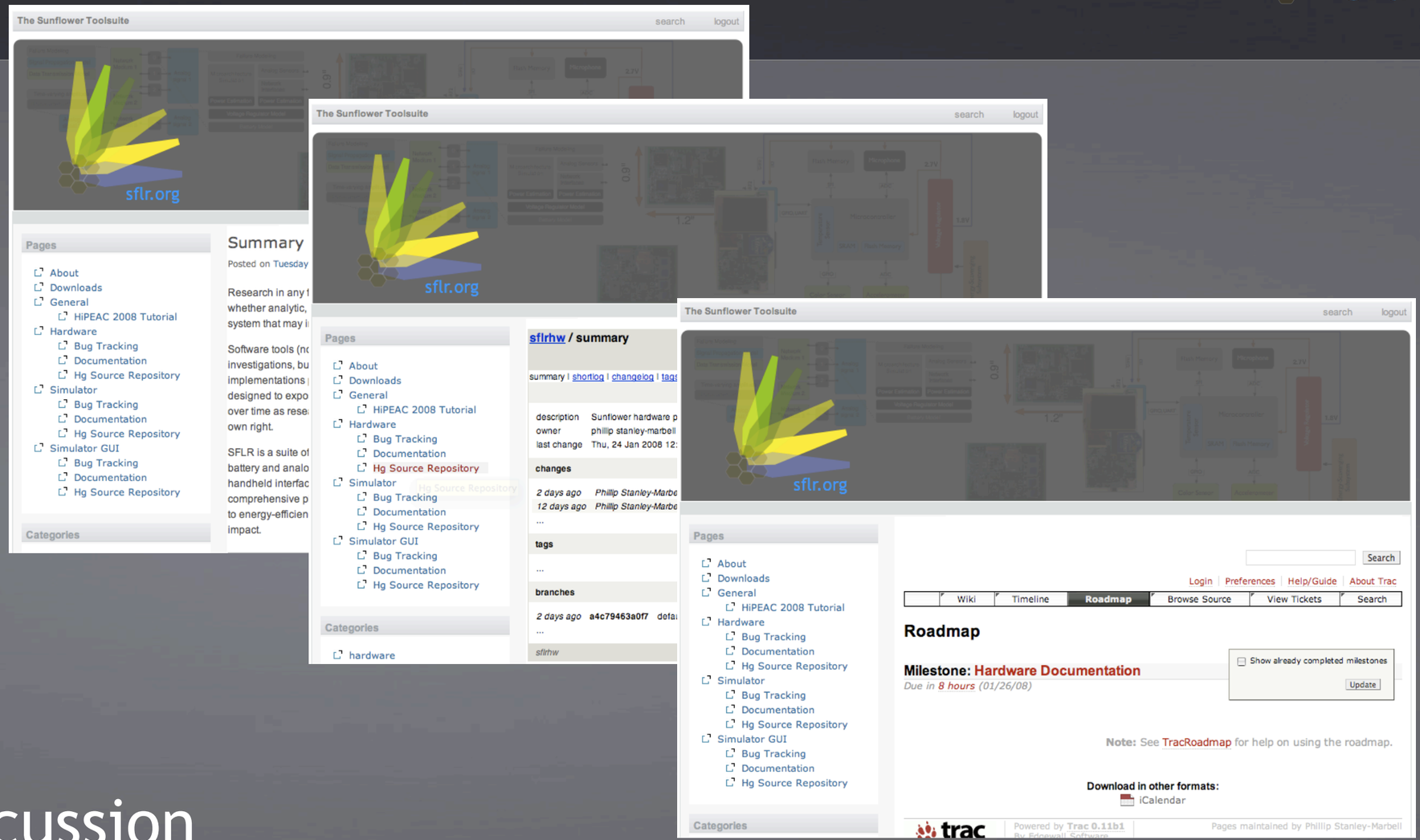
Getting Involved

- Anyone can:
  - Obtain all the source code and hardware design files are FREE on web
  - Submit hardware modification requests
  - Submit modified hardware design files
  - (You can obtain a username from me if you would like to be able to post bug reports)
  - Obtain Submitting hardware modification requests

- Web resources

Online Resources

- Web page
  - http://sflr.org
  - Version control
  - Wiki
  - Bug tracking system



- Google groups discussion
  - http://groups.google.com/sflrtoolsuite

Outline

- Motivation
- Simulator

Break

- Hardware Platforms
- Summary, Q & A

Summary and Pointers to Resources

- Sunflower simulator
  - Intended as a full-system simulation platform for networked embedded systems
  - Microarchitectural simulation, power estimation, battery modeling, analog signal modeling, networking

- Sunflower hardware platforms
  - Sunflower sensor node
  - Sunflower handheld client
  - Sunflower embedded multiprocessor module

- Online resources
  - http://sflr.org

Credits

- Carnegie Mellon University
  - The Sunflower simulator has been used in the graduate course ECE-743 “Energy-Aware Computing” since 2001; feedback from students has been invaluable
  - The Sunflower node platform was originally developed while at CMU
- Technische Universiteit Eindhoven
- Current and past users of the simulator

Thanks

Backup Slides