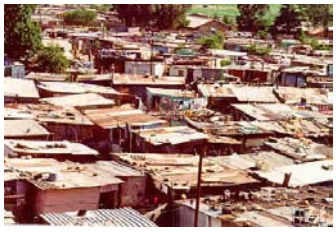
TinyGALS: A Programming Model for Event-Driven Embedded Systems

Elaine Cheong
UC Berkeley Ptolemy Group / PARC

Judy Liebman (LLNL)
Jie Liu, Feng Zhao (PARC)

Fifth Biennial Ptolemy Miniconference
Berkeley, CA, May 9, 2003

Introduction



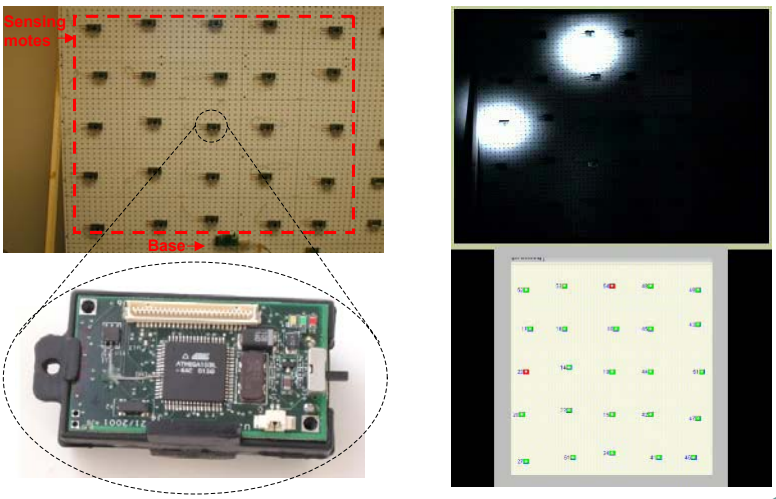
- Embedded software architecture ***today***
  - Inherited from writing device drivers and optimizing assembly code.
  - Poor scalability.
  - Poor common infrastructure.
  - Poor resource management.

- Embedded software architecture ***tomorrow***
  - Reusable, reconfigurable components.
  - Easy to use.
  - Fast prototyping.
  - Software synthesis

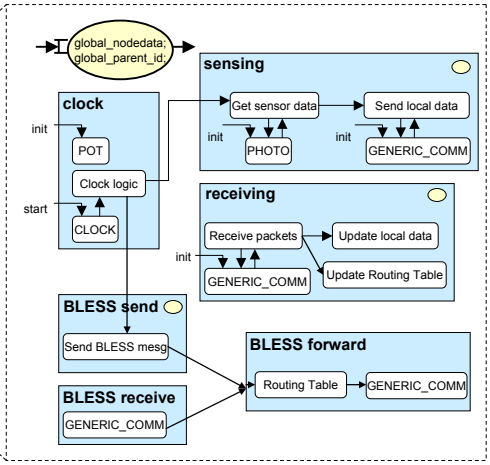
Motivation

- Application characteristics
  - Ad-hoc networked embedded systems
  - Low-power
  - Unstructured, unsynchronized events
  - Collaborative nodes
    - Local communication (peer-to-peer)
    - Global communication (ad-hoc routing)

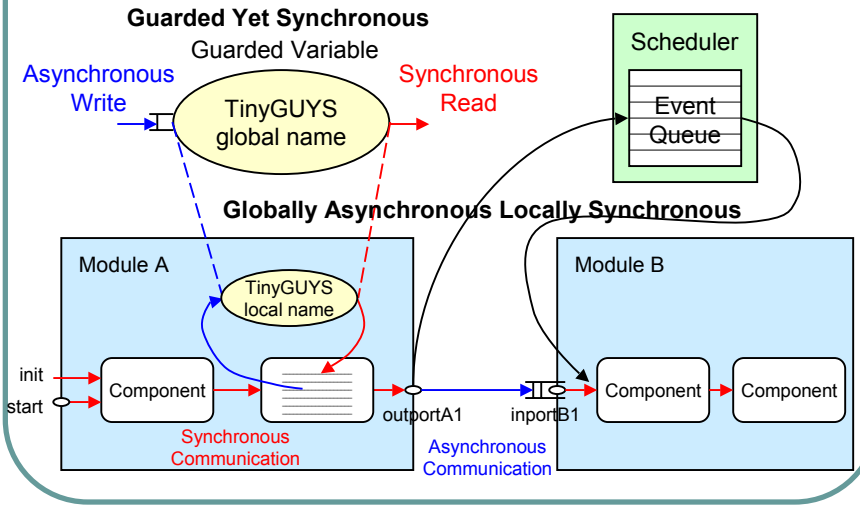
Example: Sensor Networks



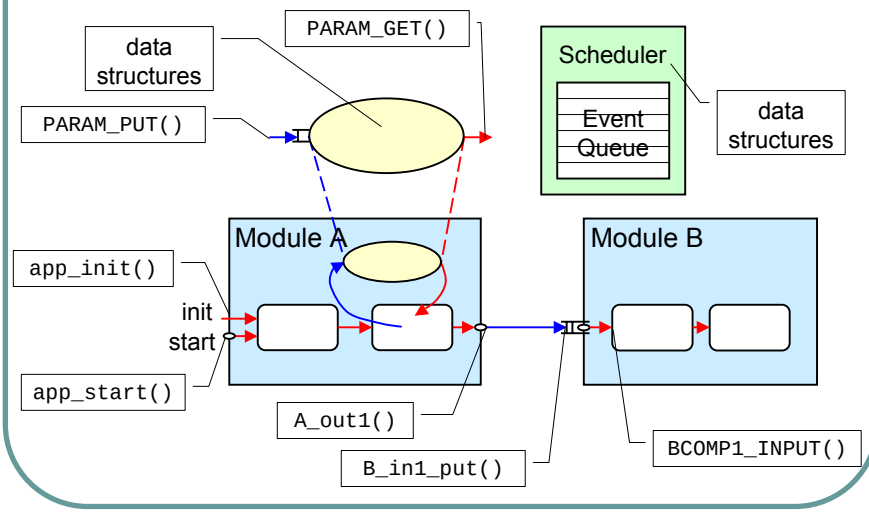
TinyGALS



TinyGALS Architecture

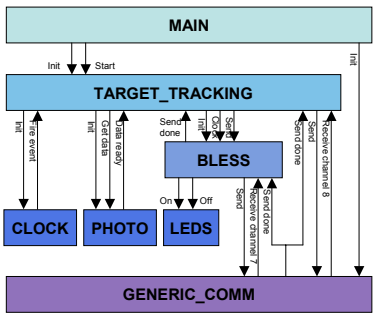


Software Synthesis

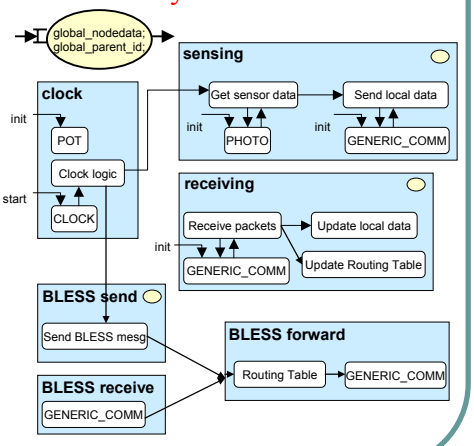


Target Tracking Example: Before and After…

OS View (TinyOS)



TinyGALS View



Memory Usage

Code Size	Scheduler	Target counting application
TinyOS	86 bytes	19929 bytes
TinyGALS	112 bytes	24750 bytes

Future Work

- Port to NesC language (TinyOS).
- Implement as Ptolemy domain?
  - Compare to CI domain.
- Blocking write: retry when queue is full.
- Priority scheduling algorithm with queue insertions.
- Run-time reconfigurability of modules.
- Heterarchy: distributed multi-tasking.

Conclusions

- TinyGALS provides a globally asynchronous, locally synchronous model of computation for event-driven embedded software.
- Allows reuse of software components.
- TinyGUYS provides protected, quick access to global data.
- Software synthesis tools created to generate communication and scheduling code.

http://ptolemy.eecs.berkeley.edu/papers/03/TinyGALS/