

The TinyOS of Tomorrow

Phil Levis

NEST Retreat, Jan. 2003

TinyOS Progress

- Data link layer
- Language support (nesC)
- Concurrency model formalization
- Data race detection
- Low-power mechanisms
- Network layer

What's to Come

- Abstract components vs. parameterized interfaces
- Incorporating concurrency model
- Further static analysis
 - Maximum stack depth
 - Model checking

Abstract vs. Parameterized

- Abstract components
 - State replication
 - Complication: shared state
 - One-of-many
- Parameterized interfaces
 - Compile-time binding

Incorporating Concurrency

- Synchronous (task)
- Asynchronous (interrupt)
- Translating between the two
- Timer: Synchronous -> Asynchronous
- Tasks: Asynchronous -> Synchronous

Post Considered Harmful

- Deprecated by concurrency qualifiers
- Abstract components (linkage analysis)
- Task queue length
- Scheduler component

```
interface TaskBasic {  
    acommand result_t post();  
    event void run();  
}
```

```
uses TaskBasic as SendTask;
```

Abstract Tasks

```
abstract component SchedulerBasic() {  
    provides TaskBasic;  
}  
...  
TaskEntry taskQueue[_INSTANCES_];  
...  
components SchedulerBasic();  
...  
TinyDB.SendTask -> SchedulerBasic;  
...
```

Parameterized Tasks

```
interface TaskPriority {
    acommand result_uncombined_t post();
    event void run();
}

component SchedulerPriority {
    provides TaskPriority[uint8_t priority];
}
```

Static Analysis

- Stack depth (full program analysis)
- Model checking (Engler et al., SOSp)
 - Implicit state transitions (MHSR)
 - Read-modify-write races (cross-component)

The Mantra

- HURRY UP PLEASE ITS TIME
- SOSP 2003