MICAz and nesC Language

Aly El-Osery
Electrical Engineering Dept.
New Mexico Tech

Last Modified Dec. 30th 2005

MICAz

- 2.4GHz IEEE 802.4.15 compliant
- 250kbps
- Direct sequence spread spectrum radio
- Runs TinyOS 1.1.7
- 51-pin expansion connector
  - Analog inputs
  - Digital I/O
  - I2C, SPI, UART interfaces

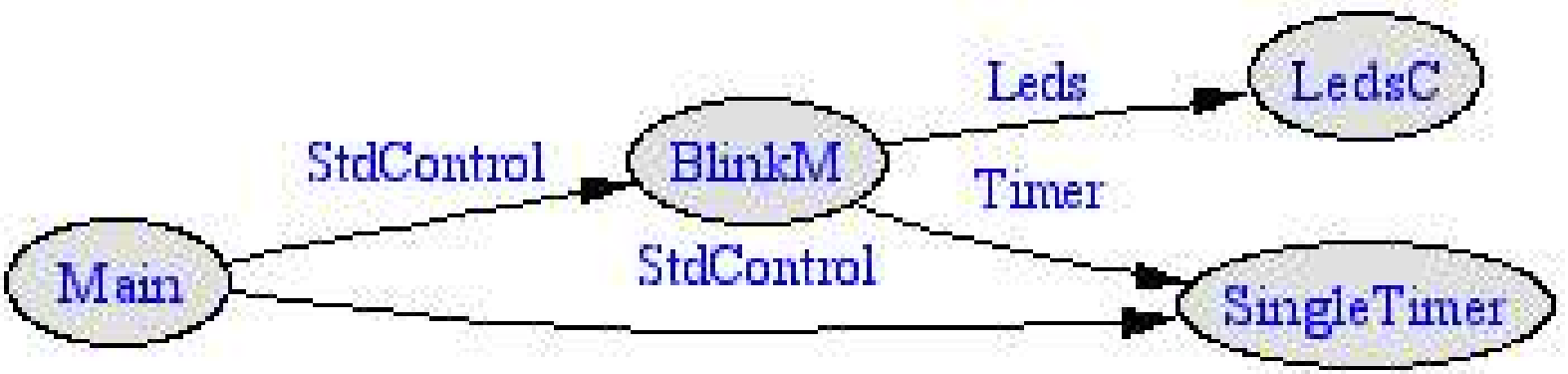
MICAz Radio

- 2.4 Ghz Chipcon CC2420
- Low power
- Built-in Security and Encryption
- Digital RSSI/LQI support
- CC2420RadioControl
  - Channel selection
  - RF Power

Introduction to TinyOs

- Event-driven operating system.
- System, libraries, and applications are written in nesC.
- Light weight and efficient
- In-line code
- Modular

Sample Application



TinyOS Structure Overview

- Configuration
  - **wiring** of components
- Component
  - A nesC application consists of *components*
  - **provides** and **uses interfaces**
  - Implemented in **module**, or
  - "Wired" up of other components in a **configuration**

Interfaces

- Bidirectional
- Specify a multi-function interaction channel between two components, the **provider** and the **user**
- Specifies a set of name functions
  - **commands** implemented by the interface's provider
  - **events** implemented by the interface's user

Filename Convention

- Configuration
  - myApp.nc at top level
  - myComponentC.nc at lower level
- Interfaces
  - myInterface.nc
- Implementation
  - Modules-myComponentM.nc
  - Configurations-myComponentC.nc

Modules

- **Implemention** of a component specification with C code

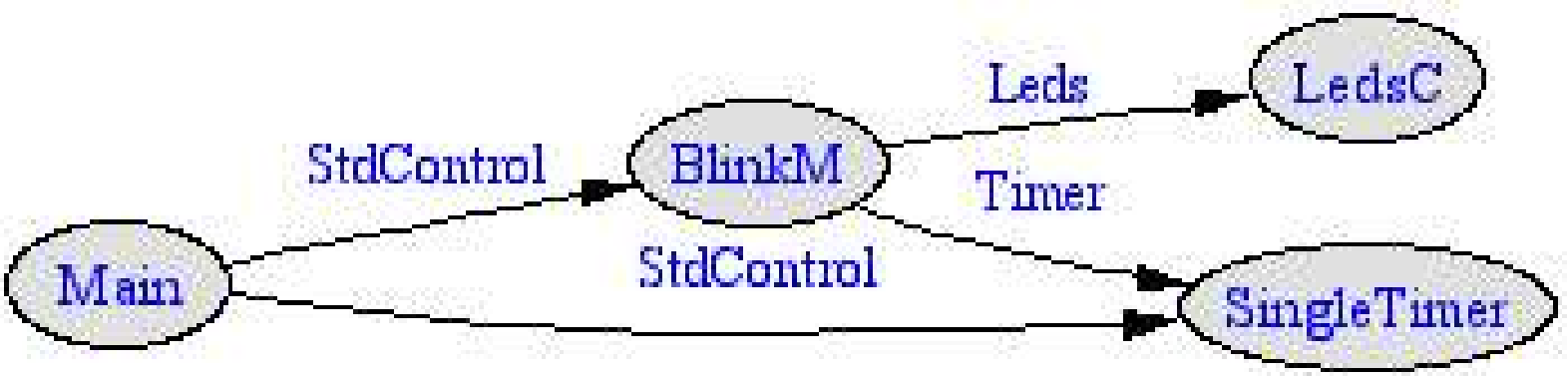
Configurations

- Implement a component specification by connecting, or wiring, together a collection of component

Events vs. Tasks

- Events
  - Time critical
  - Caused by interrupts
  - Suspend tasks
- Tasks
  - Time Flexible
  - Run sequentially
  - Interruptible

Blink Application



Blink Code

Blink.nc

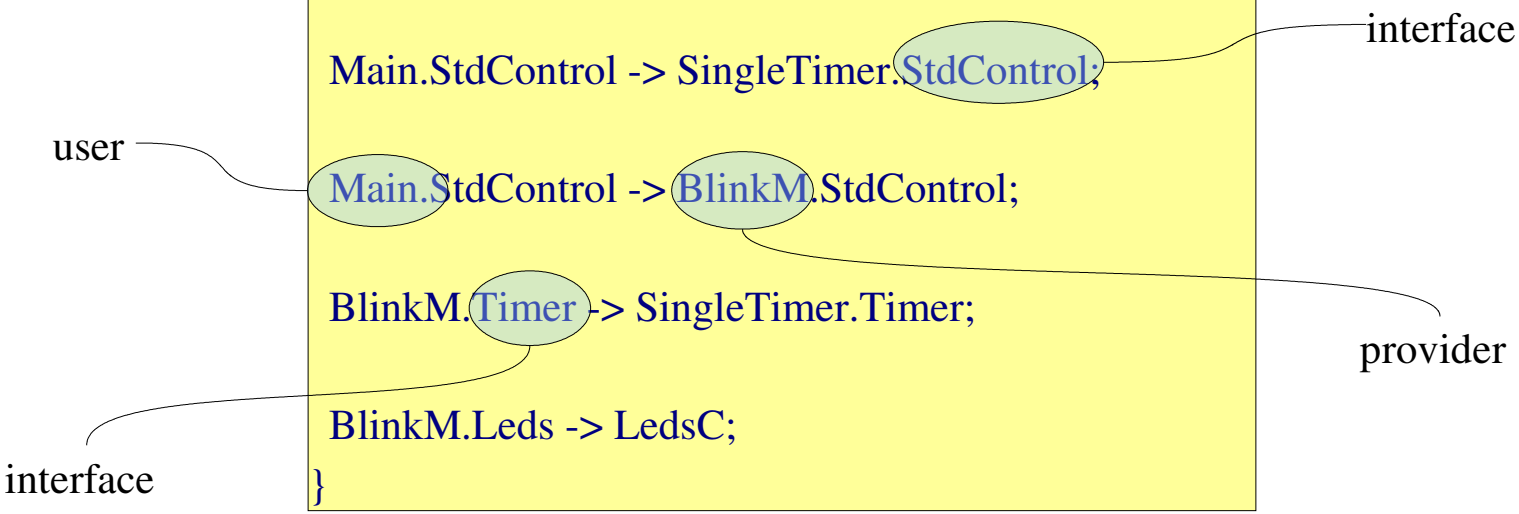
```
configuration Blink {
}
implementation {
  components Main, BlinkM, SingleTimer, LedsC;

  Main.StdControl -> SingleTimer.StdControl;

  Main.StdControl -> BlinkM.StdControl;

  BlinkM.Timer -> SingleTimer.Timer;

  BlinkM.Leds -> LedsC;
}
```



Blink Module Code

```
module BlinkM {
 provides {
  interface StdControl;
 }
 uses {
  interface Timer;
  interface Leds;
 }
}
implementation {
 command result_t StdControl.init() {
  call Leds.init();
  return SUCCESS;
 }
 command result_t StdControl.start() {
  // Start a repeating timer that fires every 1000ms
  return call Timer.start(TIMER_REPEAT, 1000);
 }
 command result_t StdControl.stop() {
  return call Timer.stop();
 }
 event result_t Timer.fired()
 {
  call Leds.yellowToggle();
  call Leds.redToggle();
  call Leds.greenToggle();
  return SUCCESS;
 }
}
```

```
Interface StdControl {
 command result_t init();
 command result_t start();
 command result_t stop();
}
```

Must Provide implementation to commands it provides and events it uses

Building an Application

- To build the application

  make micaz install,<node> mib510,com1

- To generate a document file

  make micaz docs

Reading Assignment

- nesC Language Reference

  http://nescc.sourceforge.net/papers/nesc-ref.pdf