

# Using TinyOS on BTnodes

A little more than porting to another platform...

Jan Beutel



# Outline

## Devices

- BTnode rev3 architecture details

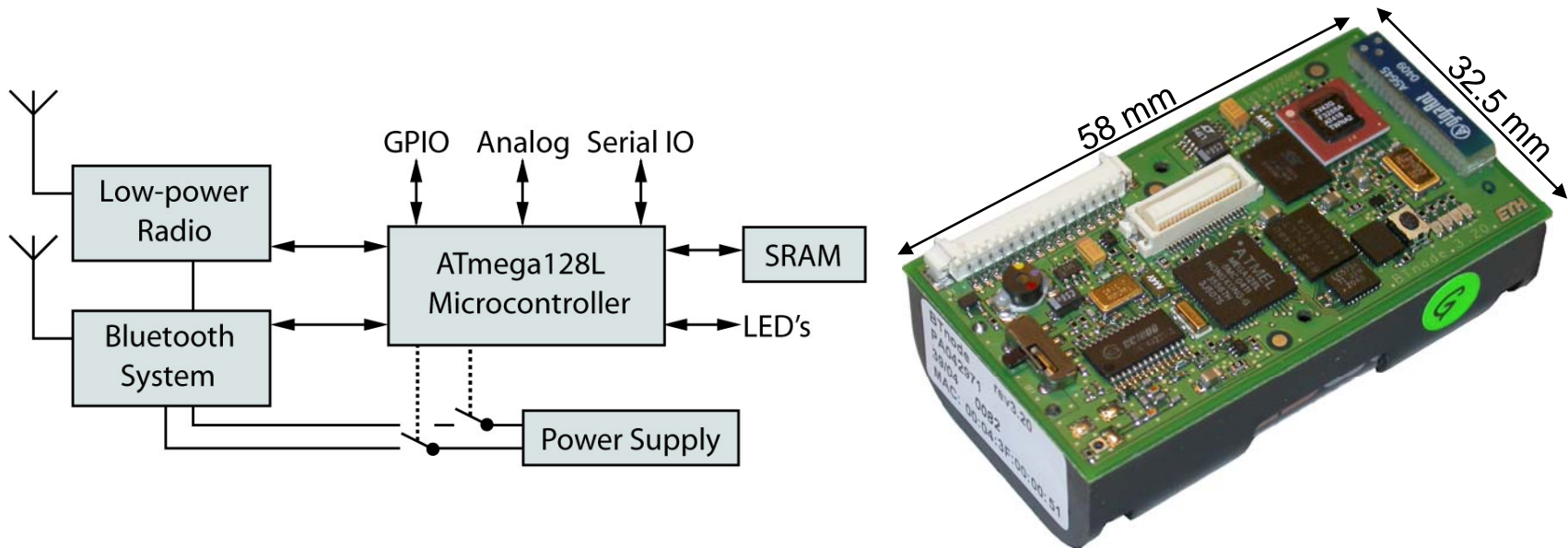
## Software

- Current state of TinyOS 1.x
- Porting TinyOS 1.x onto the BTnode rev3

## Cross platform applications

- Interfacing Motes and BTnodes using Surge
- Experiences with TinyOS – Outlook to TinyOS 2.x

# New BTnode rev3: a lightweight dual radio platform



## BTnode success story

- Running both, TinyOS and the BTnut system software
- Prototyping Wireless Sensor Networks with BTnodes [EWSN2004]
- 3rd generation node commercialized with industrial partners AoT and Iftest
- Open-source policy has led to commercial replicas (Cobalt Blue by Vitronics)

# BTnode rev3 architecture details

## System core

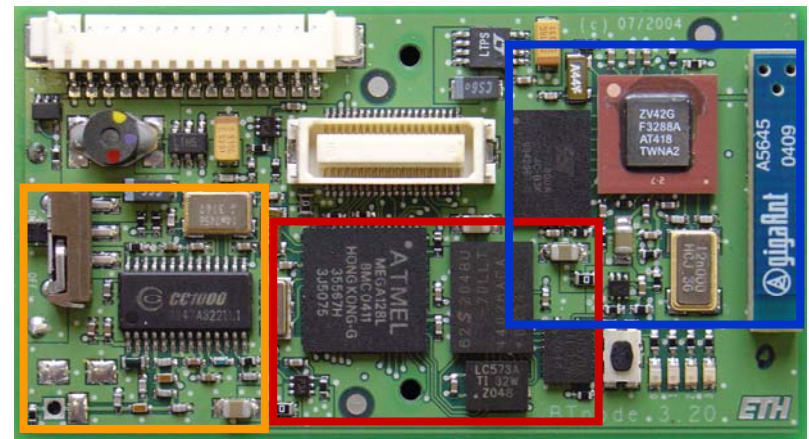
- Atmel ATmega128, AVR RISC, 8 MIPS @ 8 MHz, 128 kB Flash, 64 kB SRAM, 180 kB data cache, 4 kB EEPROM
- 4 LEDs, reset button
- External DC supply or 2 AA cells with on/off switch
- Generic sensor interfaces

## Bluetooth radio

- Zeevo ZV4002, supporting AFH/SFH Scatternets with max. 4 Piconets/7 Slaves, BT v1.2 compatible

## Low-power radio

- Chipcon CC1000 operating in ISM Band 433-915 MHz



# Related development on TinyOS 1.x

TOS 1.x radio interfaces are bit-stream oriented

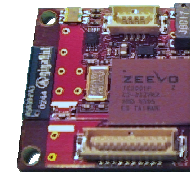
## Tinybt – TinyOS on BTnodes rev2

- Proof of concept and benchmarks



## Imotes – TinyOS with Bluetooth and 802.15 radios

- Using ARM7 and PXA273 processors

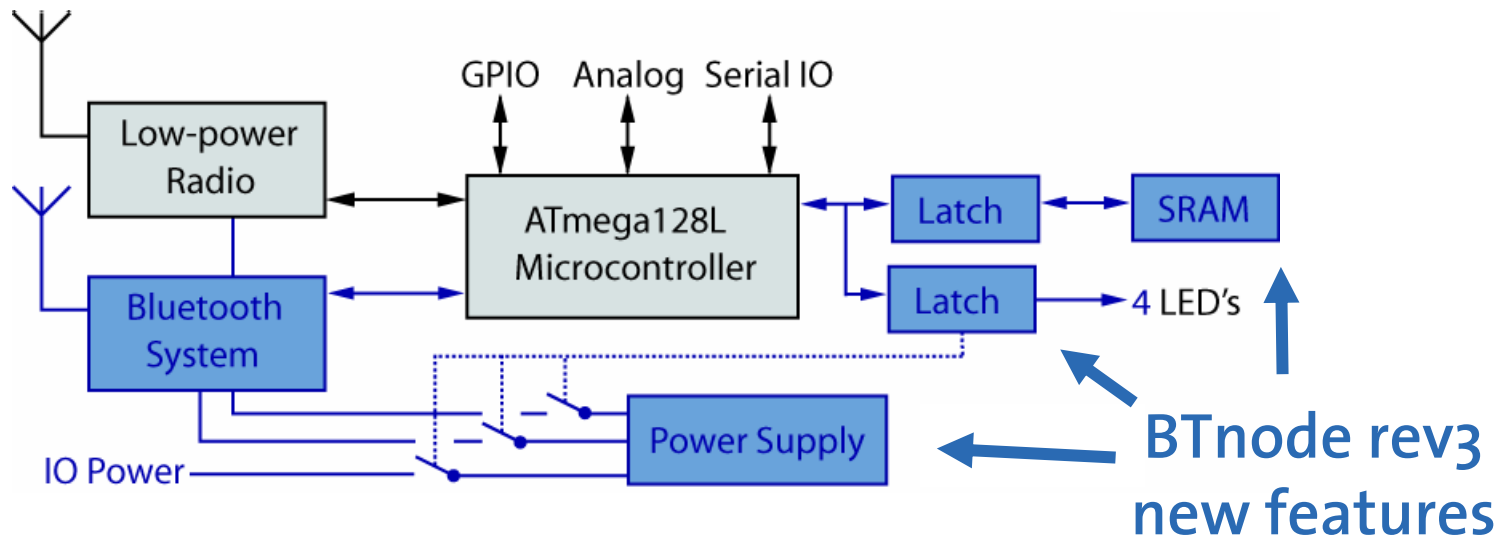


## Telos – 802.15 radio

- Link oriented stack



# TinyOS Integration: BTnode rev3 platform definition



## TinyOS 1.x prerequisites

- Platform definitions, radio stacks and applications separated
- New platform definition and drivers integrated into build system
- Driver functions often application dependant

# Interfacing Motes and BTnodes

## Seamless interoperation using TinyOS standard applications

- Blink, RfmToLeds, CntToRfm, CntToLedsAndRfm
- 3 BTnodes and 2 Mica2 Motes interoperate on Surge (no sensors)
- 1-click build from same application code

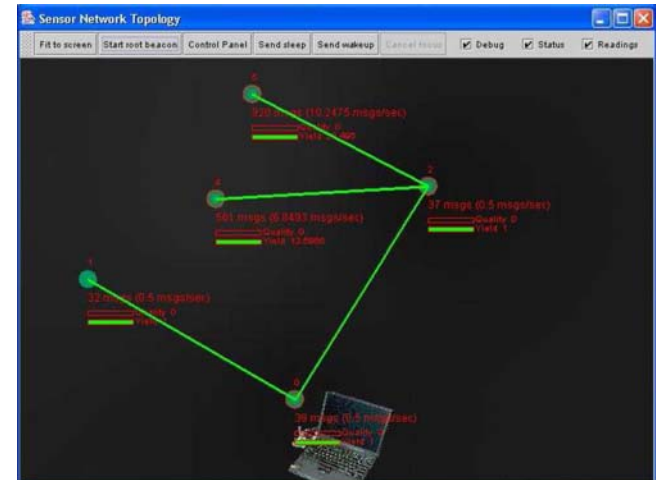
```

module BlinkM {
  provides {
    interface StdControl;
  }
  uses {
    interface Timer;
    interface Leds;
  }

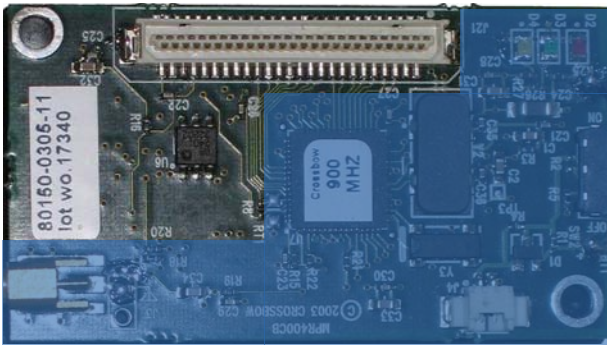
  command result_t StdControl.init() {
    call Leds.init();
    return SUCCESS;
  }
  command result_t StdControl.start() {
    // Start a repeating timer that fires every 1000ms
    return call Timer.start(TIMER_REPEAT, 1000);
  }
  command result_t StdControl.stop() {
    return call Timer.stop();
  }
  event result_t Timer.fired()
  {
    call Leds.redToggle();
    return SUCCESS;
  }
}

```

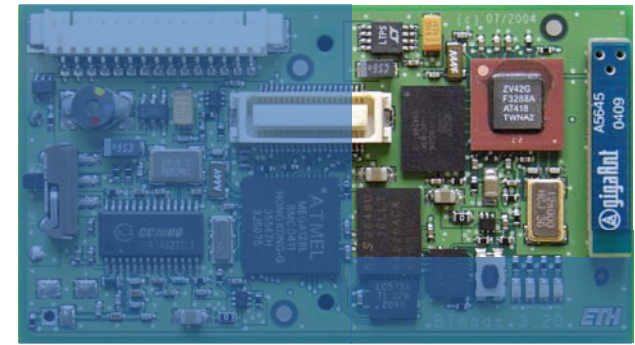
make btnode3  
 →  
 make mica2



# Experiences – TinyOS on the BTnode rev3



Twin architecture  
of Mica2 and  
BTnode 3



**Basics are working:** BTnode rev3 is a Mica2 replacement

- Cooperation with Uni Copenhagen/ P. Bonnet
- BTnode3 platform definition available in **contrib/tinybt**
- Hard to “just port” without an application requirement because of hardware dependencies in the software
- **tinyos-1.x** – 162 MB CVS nightmare
- nesC makes debugging harder, complexity is hidden within



# Experiences 2 – beyond the proof-of-concept

## Emerging (open) standards?

- Monthly TinyOS 1.1.x developer releases
- Stable “even” deployment release: TinyOS 1.2
- nesC 1.1 -> 1.2 major revision with new features is breaking compatibility

## A clean break for a new architecture: TinyOS 2.x

- Recently a lot of “back to the roots” discussion in the 2.0 WG – component based – platform design – hardware independence – compatibility – robustness – reuse – many new platforms
- Fixing many well known problems (multiple threads/tasks, RT issues, abstractions, modularity)
- TinyOS Enhancement Proposals: Open discussion
- Beta is available – looks very promising

# Acknowledgements

**Attila Dogan – term thesis @ ETHZ**

**Manatee Project – University of Copenhagen**

- Philippe Bonnet, Mads Dydensborg, Martin Leopold, Klaus Madsen

**Imote Project – Intel Research**

- Ralph Kling

**TinyOS 2.x Working Group – UC Berkeley...**

**BTnode Team @ ETHZ**

# To probe further...

BTnodes - A Distributed Platform for Prototyping Ad hoc Networks - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

http://www.btnode.ethz.ch/

## BTnodes – A Distributed Environment for Prototyping Ad Hoc Networks

Welcome to the BTnode Platform!


### Overview

The BTnode is an autonomous wireless communication and computing platform based on a Bluetooth radio and a microcontroller. It serves as a demonstration platform for research in mobile and ad-hoc connected networks (MANETs) and distributed sensor networks. The BTnode has been jointly developed at ETH Zurich by the Computer Engineering and Networks Laboratory (TIK) and the Research Group for Distributed Systems. Currently, the BTnode is primarily used in two major research projects: [NCCR MICS](#) and [Smart-Its](#).

The low-power radio is the same as used on the Berkeley Mica2 Nodes, making the BTnode rev3 a twin of both the Mote and the old BTnode. Both radios can be operated simultaneously or be independently powered off completely when not in use, considerably reducing the idle power consumption of the device.

### News

- Project: [Deployment Support Networks \(Scalable Topology Control\)](#) [2004-12-14]
- [BTnode usbprog adapter and debugging information](#) [2004-11-18]
- [New BTnut System Software release 1.1 based on EtherNUT](#) [2004-10-28]
- [BTnode rev3 available in samples through contract manufacturer](#) [2004-09-24]



### BTnode rev3 features at a glance

- Microcontroller: Atmel ATmega128L (8 MHz @ 8 MIPS)
- Memories: 64+180 Kbyte RAM, 128 Kbyte FLASH ROM, 4 Kbyte EEPROM
- Bluetooth subsystem: Zeevo ZV4002, supporting AFH/SPH Scatternets with max. 4 Piconets/7 Slaves, BT v1.2 compatible
- Low-power radio: Chipcon CC1000 operating in ISM band 433-915 MHz
- External interfaces: ISP, UART, SPI, I2C, GPIO, ADC, Timer, 4 LEDs
- Standard C Programming, TinyOS compatible

### Quickstart

To get going is quite straightforward. Before you can start developing applications for the BTnode, you need to:

- Download and install the development tools and the BTnut System Software.
- Buy a hardware programmer. We recommend the Atmel STK 500 programmer.
- Get BTnodes or serial Bluetooth devices that can be used in emulation mode.
- Compile and download your first example application.

last edited 15-Dec-2004 19:30 by [Jan Beutel](#)

Done

#### Overview

- [Old News](#)

#### Support

- [BTnut System Software Reference](#)
- [Hardware Reference](#)
- [General Software Issues](#)
- [Tool References](#)


#### Tools

- [Development Tools](#)
- [Installing the BTnut System Software](#)
- [Demo Application](#)
- [Advanced Tools](#)

#### Projects

#### Links

- [Project at SourceForge.net](#)
- [CVS at SourceForge.net](#)



Copyright (c) 2000-2004 BTnode Project

<http://www.btnode.ethz.ch>

# Backup: BTnode rev3 additional information

## Power Budget on typical AA cells (2x1.2V, 2500mAh)

- 0.5 – 150 mA current consumption
- ~80% efficiency in up-conversion to 3.3V
- Non power-aware apps can last a few days on batteries
- Power optimized apps can last many weeks depending on duty cycle

## Commercialization

- **Availability** samples now, volume Q1/2005
- **Pricing** USD 215/EUR165/CHF255 for samples, larger quantities upon request
- **Contract Manufacturer** Art of Technology, Zurich, Switzerland

Input Current at selected voltages

