Center for Embedded Networked Sensing

Implementing MicaZ Support for the AVRora Simulator

Keith Mayoral - Intern
Ben Titzer - Graduate Mentor , Jens Palsberg - Faculty Advisor

Introduction: AVRora can increase robustness through ability to simulate different types of motes

What is AVRora?

- **Simulation**
  Originally created to simulate Atmel AVR microcontroller-based sensor nodes such as the ***Mica2***, future plans include adding support for TI MSP microcontroller-based sensor nodes such as the ***Tmote Sky***.
- **Analysis**
  Avrora contains a comprehensive set of tools which allow for quick and simple debugging of programs to be used in embedded systems without the hassles that come from testing in the field.
  - Features include: ***Monitors*** which can keep track of anything from I/O register activity to energy consumed by device to a trace of lines being executed.
  - Non-intrusive to measurement of simulation time or results through use of ***probes*** and ***watches***.

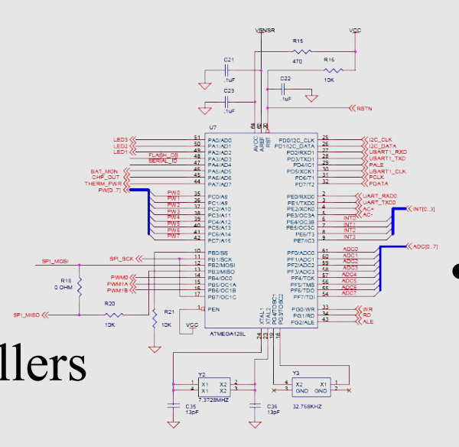
Description of Platforms

- **Mica2 (previously implemented)**
  Two main components of the Mica2 include the Atmel ***ATMega128L*** MCU and the Chipcon ***CC1000*** radio transceiver.

- **MicaZ (requires implementation of CC2420)**
  Uses same microcontroller as ***Mica2***, but carries an upgraded Chipcon ***CC2420*** radio transceiver instead.
- **Tmote Sky (both the MCU and CC2420 need to be implemented)**
  Equipped with a TI ***MSP430*** microcontroller; however, uses the same Chipcon ***CC2420*** radio transceiver as the ***MicaZ***.

Problem Description: Main component which needs to be implemented is the CC2420 Radio

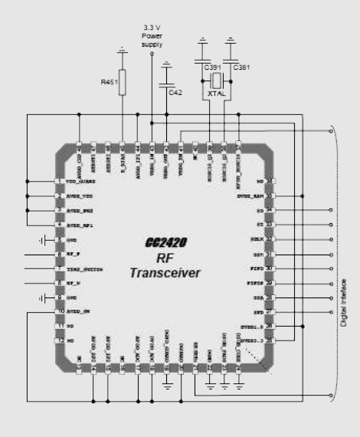
How are Devices described to AVRora?



•Written in Java

-Devices such as the CC1000 Radio and the different on-chip devices on specific microcontrollers such as the ADC, SPI, etc. are all written in Java.

-Implement different basic interfaces such as ***Radio***, ***SPIDevice***, and ***ATmelInternalDevice*** to describe what kind of device they are to the simulator



•Microcontroller described in ISDL

Instruction Set Description Language was originally created by Ben Titzer to describe the instruction set of AVR-based microcontrollers; needed to be expanded to allow for description of other microcontrollers such as the MSP430.

How will the CC2420 implementation interface with AVRora?

•Must interface with existing description of Sensor mote devices

-Just as in the physical representation of the MicaZ, the CC2420 must be able to interface through the ***SPI (serial peripheral device)*** to both configure the radio and transfer data to and from the microcontroller

-The CC2420 must also be able to interface with the simulated ***Air*** in order to transmit and receive data through the "air."

-Specific status/control pins which connect directly to the Microcontroller must be implemented through the ***PinInterface*** class.

Progress Completed: Instruction Set Description of MSP430 and implementation of CC2420

Support for the Tmote Sky

•Described MSP430 in extended ISDL

Implemented a description in ISDL which fully describes the instruction set of the MSP430. New syntax/grammar had to be created in order to completely describe instructions and addressing modes. Having both the MSP430 and CC2420 implementations for Avrora is a necessary initial step to including future support of the Tmote Sky.

•Thing which still need to be implemented for MSP430 support:

Includes description of on-chip devices on the MSP430 such as the ADC, DAC, Timer, Multiplier, USART, etc. Other components such as I/O, interrupts, and SPI can be implemented easily by modifications of existing descriptions.

•Problems encountered:

It was obvious early on that the original ISDL could not effectively describe the MSP430, therefore after the new description of the MSP430 had been created, the language was modified to accommodate for the newly created grammar used.

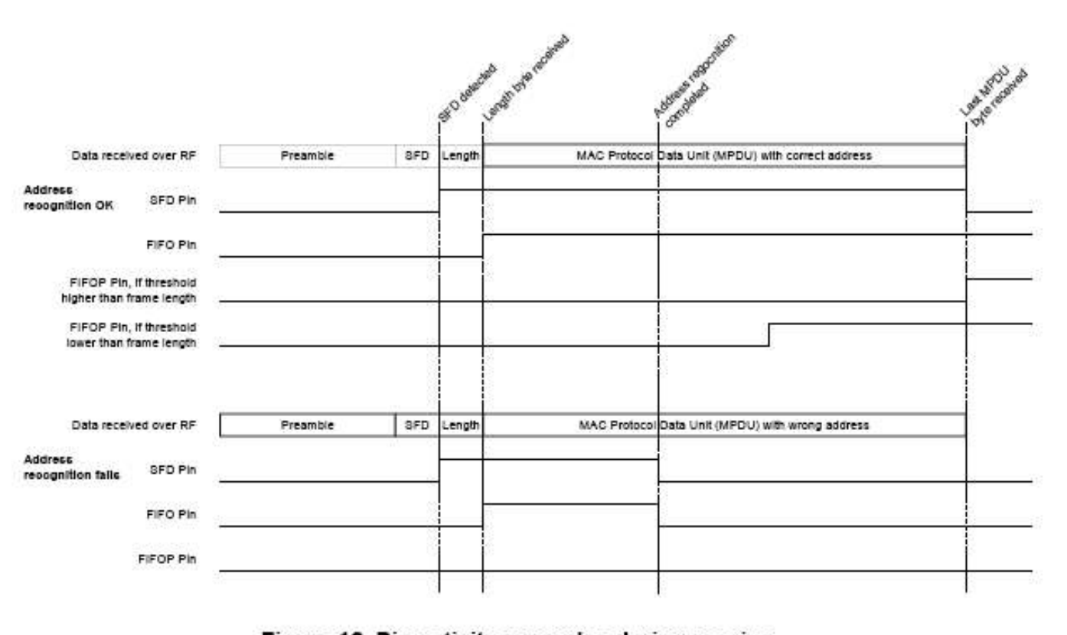


Figure 12. Pin activity examples during receive

Implementing the MicaZ in AVRora

•ATMega 128L microcontroller

Since the MicaZ uses the same MCU as the Mica2 and support for the Mica2 already exists in the Avrora simulator, The first half of this implementation was already completed. Using the Avrora-defined ***Platform*** class, it was possible to tie together different defined devices such as the ATMega128 to a radio like the CC2420 or CC1000 effectively creating the MicaZ and Mica2 platforms respectively.

•CC2420 Radio Transceiver

A large amount of time was devoted to implementing the CC2420 Radio Transceiver, and work still remains to simulate radio activity correctly in Avrora. Before work could begin on the implementation of the CC2420 radio, several abstractions had to be made for devices which would interface with the CC2420, since until then they had be defined to interface solely with the CC1000.

Specifics on the CC2420

•What works

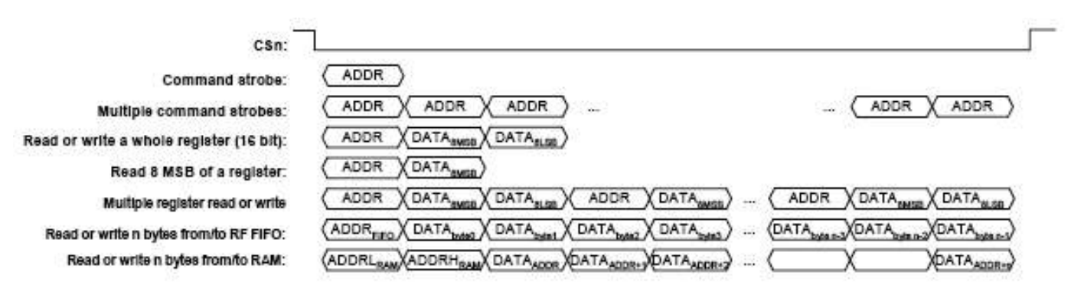
Basic registers, command strobes, RAM access, and RX and TX FIFO buffers are implemented and can be accessed, reset and set as they should be.

- ***Registers*** are read or written to by sending 24-bits through the SPI to the radio (one byte for address, two for data) and are used to set different modes, and values used by the radio.
- ***RAM*** access works much like register access, except any address in the memory space can be accessed, and more than two bytes can be written in a given transfer, depending on pin settings.
- ***Command strobes*** activated the same way registers are accessed, by sending the address byte through SPI; however no additional data is required as in register or RAM access.
- ***FIFO buffers*** used for both receiving and transmitting can be accessed through register access, adding or removing a byte to the buffers at a time, or through RAM access by writing to specific addresses within the buffers.

Control/Status pins have been implemented, the Status Byte is set and sent back down the SPI line when required, a basic preamble generator transmits data prior to sending payload.

•Improvements to be made

The timing of transfers must be corrected in order to be assured of proper simulation time. Also, if radio and SPI/MCU are out of sync on timing, transfers may not be completed correctly.



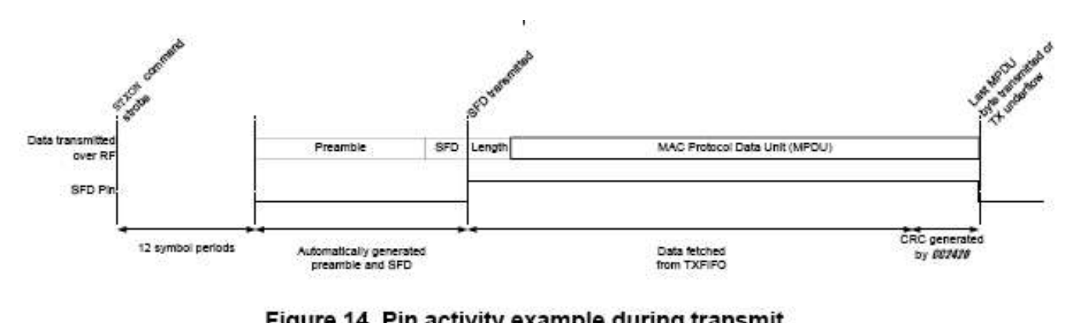


Figure 14. Pin activity example during transmit

Partial implementation of Demodulator for accurate RSSI and LQI readings can also be done.

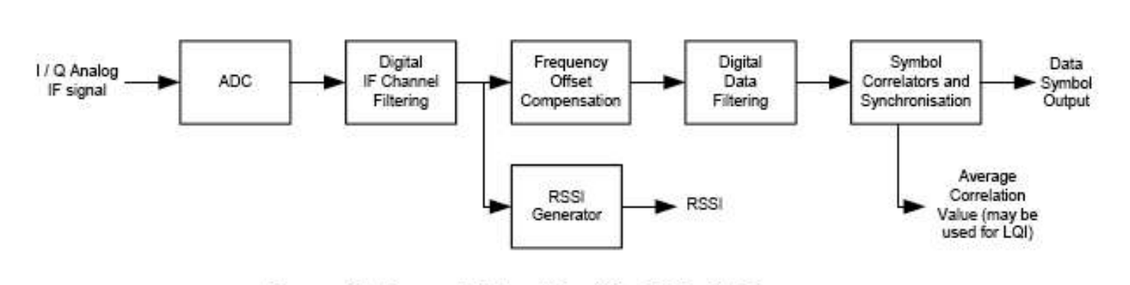


Figure 15. Demodulator Simplified Block Diagram

Future Work

•Fully implement features in CC2420

The CC2420 has many hardware-related features that can still be implemented for a more realistic simulation.

- Real-time RSSI and CCA support
- Full implementation of registers and command strobes
- Full hardware security support; MAC encryption/decryption
- Fix timing issues which affect SPI and RF data transmission

•Create the Tmote Sky platform

Finish defining on-chip devices of MSP430 in order to create a platform which uses the MSP MCU along with the CC2420 Radio.

•Simulation of Heterogeneous networks

Once Avrora becomes capable of simulating a different variety of sensor nodes, in the future it may be possible to simulate a ***heterogeneous network*** through Avrora. This, in turn will allow for a finer degree of optimization when creating sensor networks. For example, by being able to simulate different types of motes, it will be possible to optimize different parts of the network for RF data transfer and energy consumption limitations, among other things, prior to deployment out in the field.

UCLA – UCR – Caltech – USC – CSU – JPL – UC Merced