RADIO, RF Concepts, and TOS
Radio Stack

- Mote Radio Architecture
- RF Propagation
- Information Transport
- TOS Messaging & Radio Stack
- Mote Radio Control

---

ChipCon Radio System

- Single IC Transceiver
- Dual Band 433MHz and 915MHz ISM Bands
- 38.4KBit/sec Data Rate
- Low Power 16mA Tx / 9mA Rx
- Programmable Configuration
Atmega uP
- 32Khz crystal and 4Mhz crystal.
- 10 bit ADC
- UARTS (Mica2/Mica2Dot have 2)
- SPI bus
- I2C bus (hardware for mica2/mica2dot)
- Radio (Chipcon 1000)
- External serial flash memory (512K byte)
- Connectors for interfacing to sensor and programming boards
- 1 programmable led (Mica2Dot)
The CC1000 Radio Interface

- Dedicated cpu bus (lines) to configure radio registers for radio frequency, power,.....
- Dedicated SPI bus for data transfer. CC1000 is bus master.
- Radio generates one interrupt every 8 bits when in receive mode.
- Runs usually at 38K or 19K bit rate (default) Manchester (2x bit)

<table>
<thead>
<tr>
<th>Baud Rate</th>
<th>Xmt or Rcv Time(*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>19K</td>
<td>~40msec</td>
</tr>
<tr>
<td>38K</td>
<td>~20msec</td>
</tr>
</tbody>
</table>

(*) Does not include random delay

Chipcon-ATMega Interface

- Configuration Interface
- Communication Interface
  - Data uses ATMega SPI Port
  - CC1000 is SPI Master
- SPI Port
  - Rx Double Buffer
  - Tx Single Buffer
RF Frequencies & Channels

- ISM Bands
  - 902 – 928 MHz US
  - 433– 434.8 MHz Europe
- RF Channel Programmable within a band
- Mote is manufactured for specific band

---

RF Modulation - FSK

- Frequency Shift Keying (FSK)
  - 1/0 represented by two different frequencies slightly offset from carrier frequency

![FSK Waveform Diagram]

64KHz Separation
Manchester Encoding

- Every Bit (Zero or One) has a transition
  - Guarantees no run of NRZ ones or zeros
- Ensures Stable Clock Recovery at receiver
  - Clock determines sampling time of data bits
- Implemented in CC1000 hardware
  - Reduced ATMega128 overhead

NRZ Code: 0101100
Manchester Code:

Monopole Antenna

Simple ¼ wave whip is sufficient for most applications:
- 916Mhz => 3.2” wire length
- 433Mhz => 6.8” wire length

Figure 2. (a) Quarter-Wave Monopole Antenna  (b) Equivalent Half-Wave Dipole Antenna
Antennas and Radio Transmission

- **Polarization:**
  - Vertical orientation of all antennas in a system is best.
  - $1/10^{th}$ distance if some antennas are vertical, some horizontal

- **Transmission Near the Ground**
  - Mica2 916Mhz, 3’ above ground ->’300’ line of sight, 30’ on the ground
  - Mica2 433 Mhz, 3’ above ground ->500’ line of sight, 150’ on the ground

---

RF Propagation

- **Line of Sight**
  - Direct path from Transmitter to Receiver
  - Free space attenuation $1/d^2$
    - double distance needs 4x power

- **Reflection**
  - Off objects large compared to Wavelength
    - walls, buildings

- **Scattering**
  - Off objects smaller than Wavelength
    - foliage, chairs
**MultiPath**

- Path lengths => Delayed version of signal
- Path Attenuation => Various signal strengths
- Result looks like Distortion / Interference at Receiver
- Out of Phase Signal Interference creates NULLS – Zero signal strength

![Diagram of MultiPath](image)

**Indoor Propagation**

- Rapid Signal Attenuation $1/d^3$ or $1/d^4$
- People moving around cuts range to $1/3$
- Concrete/Steel Flooring to $1/4$
- Metallic Tinted Windows to $1/3$
### Dynamic Fade Effects

- People Moving, Doors Opening & Closing
  - Eg. Closing doors in lab changes Strong (Green) to Weak (Blue) RF Regions

---

### Common RF Link Problems

- Signal Strength
  - Weak, Overload
- Collisions
  - Other Motes (independent of GroupID)
- Interferers
- Multi-path
- Cross-talk
  - Adjacent RF Channels
RF Link Metrics

- Packet Loss
  - Application Layer
- Bit Error Rate
  - Link Layer
- RSSI Received Signal Strength
  - Physical Layer

RF Solutions

Signal Power & Wavelength

- Transmitter Power Level
- Antenna Efficiency
  - Monopole vs. Dipole
  - Antenna Orientation
  - Antenna Placement
    - Ground Plane
    - Off Ground

- RF Band Choice
  - Installation 433MHz vs. 902MHz
- RF Channel selection default
  - static / compile time
- Frequency Hopping
  - dynamic / run time
Radio Packets

Data Transport

- Packetized Data
  - 18 byte Preamble 1010 pattern for clock recovery
  - 2 byte Frame Sync Start of Data Packet
  - 36 bytes of TOS Packet
**Mica2dot**

- **MAC Delay**
- **Preamble**
- **Sync**
- **Packet Transmission**

<table>
<thead>
<tr>
<th>MAC Delay</th>
<th>Preamble</th>
<th>Sync</th>
<th>Packet Transmission</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-128</td>
<td>18</td>
<td>2</td>
<td>36</td>
</tr>
</tbody>
</table>

- **Switch to TX Mode**
- **Switch to RX Mode**

- **Switch to TX Mode**: 250µs
- **Switch to RX Mode**: 250µs

**TOS Message Structure**

- **Header**
  - Address (2bytes)
  - Active Message Type (1byte)
  - Group ID (1 byte)
  - Payload Length (1 byte)

- **Payload**
  - 29 bytes User/Application defined

- **CRC**
  - 2 bytes
Radio Interface

- Data/ Noise streams in from CC1000 at 19.2Kb/s rate
- SPI Input Interrupt every 416uSec (8bits)
- Triggers RadioIntM.nc Module

```
SPI Rx Data  0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
```

Rx Byte Interrupt

```
0 1 0 1 0 1 0 1 0 1
```

Tx Byte

Mica2dot Radio Stack
Mica2dot (CC1000)

Generic Comm

AM

CC1000RadioC

Control (Freq, Power, etc)

CC1000Control

CC1000RadioIntM

SpiByteFifo
RandomLFSR
ADC

Wires the control and data paths: Implementation hidden from app

CSMA Encoding Data Preamble Detect Synchronization

GenericCom Stack
AM Handling & RF/UART
Radio Stack
CC1000Radio

TOS Packet Reception

- RadioIntM.nc SPI Port Interrupt Handler
  - Search for Preamble Pattern (10101)
  - Wait for Frame Sync word
- Assemble TOS Packet
- Check CRC – reject if bad
- Route to Active Message Handler
- Check Group ID – reject if not member
- Signal Application ReceivedMsg Event
TOS Packet Transmission

- Packet is routed
  - GenericCom
  - AM Handler –RF or UART
  - TOS CC1000RadioIntM
- Random Delay (0-15 packet times)
- Check for Collision
- Turn On Transmitter
- Send
  - 18-byte Preamble (10101 pattern) & Frame Sync
  - TOS Packet (34 bytes)
  - CRC (2 bytes)
- Turn Off Transmitter
- Signal TxDone event to Application

CC1000RadioInt State Machine

- Radio SPI Data Port Interrupt triggers every 8 bit times = 416uSec
- Major States
  - IDLE
  - RX Related: Sync, Packet Assemble
  - TX Related: Collision Sense, Preamble, Sync, Packet, CRC, Flush,Done
- Graphically…
IDLE State

- RXData == Preamble Pattern?
  - Next State = SYNC
- TX Delay (Holdoff) Decrement
- TX Delay == 0?
  - Next State = PRE_TX
**SYNC RX State**

- Shift RX Byte bit-wise into Word Buffer
  - Word Buffer == SYNC PATTERN?
  - Byte Align = Bit Shift Count
  - Next State = RX
- RX Byte Count > MAX LENGTH ?
  - Next State = IDLE

<table>
<thead>
<tr>
<th>Word Buffer (Previous RX Byte)</th>
<th>Incoming RX Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>10011001</td>
<td>11100110</td>
</tr>
<tr>
<td>00110011</td>
<td>11001100</td>
</tr>
</tbody>
</table>

SYNC PATTERN

---

**RX State**

- RXBuffer[RXCount] = RX Data
- RXCount++
- Compute CRC(RXByte)
- RXCount == RXBuffer[Length]+Header?
  - CRC = RXBuffer[CRC]?
    - Post PACKETRECEIVED
    - Next State = IDLE
  - Error ?
    - Next State = IDLE
Signaling of PacketReceived by CC1000Radio

```
... if (rxbufptr->crc == usRunningCRC) {
    call SpiByteFifo.disableIntr();
    RadioState = DISABLED_STATE;
    rxbufptr->strength = usRSSIVal;
    if (!(post PacketRcvd()))) {
        // If FALSE there are insufficient resources to
        process the incoming packet we drop it
        RadioState = IDLE_STATE;
        call SpiByteFifo.enableIntr();
    }
}
```

```
void PacketRcvd() {
pBuf = rxbufptr;
pBuf = signal Receive.receive((TOS_MsgPtr)pBuf);
}
```

PacketReceived EVENT

- Signal Generated in Radio Component.
- Propagates up to AMStandard Component

```
// Handle the event of the reception of an incoming message
TOS_MsgPtr received(TOS_MsgPtr packet) __attribute__((C, spontaneous)) {
    counter++;
    if ( packet->group == TOS_AM_GROUP &&
        (packet->addr == TOS_BCAST_ADDR ||
         packet->addr == TOS_LOCAL_ADDRESS)) {
        uint8_t type = packet->type;
        TOS_MsgPtr tmp;
        // dispatch message
        tmp = signal ReceiveMsg.receive[type](packet);
        if (tmp)
            packet = tmp;
    }
    return packet;
}
```
# TOS Radio Controls - Frequency

- **Frequency Band / RF Channel Choices**
  - `#define CC1K_433_002_MHZ 0x00`
  - `#define CC1K_916MHZ 0x01`

- **Specify CC1K_DEFAULT_FREQ in makefile**
  - `CFLAGS -d:CC1K_DEFAULT_FREQ CC1K_433_002_MHZ`

---

# TOS Radio Controls - Power

- **Power On / Off**
  - Sleep ~ 2uA
  - Radio Signal Strength (RSSI Valid) ~ 20uS
  - Receiver Packet Acquire Time ~3mSec
  - Re-TUNE Radio after a power off/on cycle
  - `command result_t Tune(uint8_t freq);`

- **RF Power Level (@ 915MHz)**
  - `command result_t SetRFPower(uint8_t power);`
  - 0xFF is 5dBm
  - 0x80 is 0 dBm (1mW)
  - 0x09 is −10dBm
**Important RF Issues**

- Different Motes for Different ISM Bands
- Re-tune after Sleep or Temperature changes
- Keep Motes separated >1m
- Watch out for Multi-path effects
- Different Group id’s do NOT prevent RF interference

---

**Radio Debugging Hints**

- Correct Radio Frequency?
  - CC1K_DEFAULT_FREQ
- Correct GroupID?
- GenericBase Hangup
  - Press RESET button
- RF Null Location?
  - Move Mote to different location (+/- 1m)
- RF Overload
  - Separation >3m