Some SDR Research at Virginia Tech

Jeffrey H. Reed
Mobile and Portable Radio Research Group (MPRG)
Virginia Tech
reedjh@vt.edu, (540)231-2972
June 22nd, 2004
Research Overview

- Configurable Computing
- Analysis of SDR with Game Theory
- Testbed prototypes
  - Overloaded array processing
  - Single-channel DF
- Distribute MIMO for Cooperative Communications
- Interference management
- SCA Research
SCA Research

- Open-source CF development
  - OSSIE
- Smart Antenna API
- Integration of test equipment for design and debugging
- Power-aware SDR
- Integrated development environment
Impact on Education

- Classes
  - Software Defined Radio Class
  - DSP Implementation of Communication Systems

- Student Projects
  - Open Source SCA

- Development of Supporting Materials
  - Books
  - Short Course
  - Web downloads
Configurable Computing for Software Radio Handsets

A chip that is a collection of dynamically interconnectable and programmable processing cores. (Like an FPGA, but with a coarser granularity)

\[ g(D) = 1 + D + D^2 \]
Objectives of Configurable Computing for Software Radio

- Identify and Evaluate “Ideal” Custom Computing Machine (CCM) architecture for handsets targeting CDMA2000 and UMTS
  - Method for Evaluating Disparate Chip Architectures
  - Dynamic CCM Simulator
  - Attributes of Optimal CCM for UMTS / CDMA2000 handsets
  - Comparative Evaluation of Developed CCM, TI 6701 DSP, and ASIC
  - High-Level Design of Compiler for Developed CCM
In Many Envisioned Applications, Software Radios Will Dynamically Adapt Its Waveform To Perceived Changes In The Environment

- DARPA’s xG Project
- Cognitive Radio
- Spectrum Filling
- Adaptive Interference Avoidance
- Mobile Ad-hoc Networks

Considering An Individual Link, This Adaptive Scheme Is Clearly Advantageous, But With Many Software Radios In A Network Adapting Their Waveforms Problems Can Arise

- Oscillatory Network Behavior
- Undesirable Steady States
  - Distributed power control where there is an incremental benefit to increasing each mobile’s power level leads to each mobile transmitting at its maximum power level
  - Less than optimal allocation of resources
- Overlapping Contentious Networks
- High Sensitivity to Small Environmental Changes
Game Theory Serves As An Analytic Framework for Modeling SDR Behavior

Software Radio
- Software Radios in Network
- Available Waveforms
- Adaptation Algorithm
  - Decision Criteria Function
  - Decision Update Algorithm

Game
- Player Set
- Action Set
- Utility Function
- Learning Process
Power Control Case Study

- 22 Mobiles, Uplink, Single Cell,
- DS-SS, $N = 63$
- Mobile Power Range $(-\infty, 20]$ dBm
- No Coordinated Behavior (etiquette)
- Better Response Dynamic

1. **BER Minimization**
   - Steady state $p = \{20, 20, 20, \ldots\}$
   - Range of values: Closest virtually no errors, furthest, $P_e \approx 0.5$

2. **Target BER ($10^{-2}$)**
   - Steady state $p \propto 1/d^2$
   - All mobiles achieve target
Case Study Comments

- Uncoordinated behavior can either lead to desirable or undesirable results – must understand algorithm before implementing.
- Game theory provides tools to analyze the algorithm.
- Algorithm selection should consider:
  - Steady-state performance
  - Convergence dynamics (Network Complexity)
  - Stability (Robustness)
Hardware setup for software radio test bench

- SDR 3000 – A SCA compliant integrated platform with upto 4 A/D and D/A converters, 4 Virtex II FPGAs and 2 G4 7410 PowerPCs for signal processing
- Signia-IDT 2 channel 20 MHz wideband RF receiver
- MPRG-developed wide band RF transmitter.
Projects on SDR 3000 system

- 802.11b base band processing MIMO demo – space time coding for real-time video evaluation
- 802.11a full system demonstrating MIMO and overloaded array processing algorithms
- To be used for projects on software radio class.
Array Processing on an 802.11a Software Radio Test Bed

- **Primary Objective**
  - Facilitate study of multi-user Smart Antenna algorithms in an overloaded environment.
  - Utilize the superior development capability of SDR-3000.

- **Secondary Objective**
  - Verify power amplifier distortion cancellation measures in a wideband scenario.
Overloaded Signal Environment

- Overloaded Array: more signals than elements.
- Conventional Array Processing breaks down.
- Can extract signals from the environment if can exploit known signal properties.
- OLAP hardest when all signals are cochannel, have little excess bandwidth (e.g. narrow-band) and are near-equal power.

\[ D_u \geq M \]

\( D_u \) Num. Sigs. \( M \) Num. Elements.
Overloaded Array Scenario

- Example: Airborne communication node is under consideration by commercial and military organizations.
  - Communications in Disaster Relief Scenarios
  - Military Communications
- Developed Spatially Reduced Search Joint Detection (SRSJD) Algorithm capable of OLAP in twice-overloaded environments
Single-user MIMO Implementation on 802.11a-like Signal
Single Channel Direction Finding System

- Sponsored by: DRS Technologies

- Objectives
  - To develop an algorithm that can be implemented on a single receiver to detect the DOA of a signal.

- Deliverables
  - Matlab simulation of the algorithm
  - Implementation of the algorithm on the DRS Sunrise Software Radio Receiver.
By using a single receiver and an antenna array, the DOA of a plane wave can be determined.

Common methods:
- Watson-Watt
- Pseudo-Doppler Method
- Others
Key Results - Implementation

- Measurements
  - homemade anechoic chamber
  - Tx=14 dBm
  - Tx-Rx separation of 4 feet.

- Robust estimate
  - Similar accuracy:
    - Tx=-70 dBm
    - Tx-Rx separation of 4 feet
    - No anechoic chamber

<table>
<thead>
<tr>
<th>True DOA (approx.)</th>
<th>Estimated DOA</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0°</td>
<td>-0.3°</td>
<td>0.3°</td>
</tr>
<tr>
<td>45°</td>
<td>45.6°</td>
<td>-0.6°</td>
</tr>
<tr>
<td>90°</td>
<td>88.8°</td>
<td>1.2°</td>
</tr>
<tr>
<td>135°</td>
<td>132.4°</td>
<td>2.6°</td>
</tr>
<tr>
<td>180°</td>
<td>175.6°</td>
<td>4.4°</td>
</tr>
<tr>
<td>225°</td>
<td>222.1°</td>
<td>2.9°</td>
</tr>
<tr>
<td>270°</td>
<td>267.7°</td>
<td>2.3°</td>
</tr>
<tr>
<td>315°</td>
<td>315.1°</td>
<td>-0.1°</td>
</tr>
</tbody>
</table>
Characterization & Management of WLAN Interference

- ISM band is virtually unmanaged wireless medium.
  - 802.11 WLANs must contend with disparate numbers and varieties of interferers
    - microwave oven, cordless phones, VoWiFi phones, Bluetooth devices, and adjacent 802.11 networks

- Cost of maintenance of WLAN networks growing rapidly
  - Current WLANs typically have limited interference characterization and management capability

- Strong need exists for sophisticated network-based tools to characterize and manage WLAN interference
  - Provides ability to optimize network operation
    - i.e.: throughput, coverage, QoS
Objectives and Techniques

- Develop novel and effective network-based means to detect, classify radio network interference
  - Employ offline processing of snapshot data

- Develop advanced interferer geolocation techniques
  - Employ RF fingerprinting and spectral coherence of the signal(s)

- Develop network evaluation and refinement techniques to optimize overall performance of radio network
  - Employ interference avoidance and management techniques

- Develop device and radio network architectures that can implement above-mentioned techniques
  - Develop foundation for next-generation WLAN network
2.4 GHz Interference Environment

- Burst 802.11b clients (CCK)
- Wireless VoWiFi phone, cordless phones (DSSS, FHSS)
- Out-of-network 802.11 STAs (DSSS)
- High-rate 802.11g OFDM streaming data (emerging)
- Microwave ovens (chirp)
- Bluetooth (FHSS) everywhere
- 802.11bg AP and STAs

Simulated spectrum of Microwave oven emission 2420-2463MHz
Simulated Microwave oven emission envelope at 2462MHz
Software Communication Architecture (SCA) Research Projects

■ Background
  ○ Joint Tactical Radio System (JTRS) program
  ○ SCA

■ Overview of research on SCA at MPRG
  ○ Open-source CF development
  ○ Smart Antenna API
  ○ Integration of test equipment
  ○ Power-aware SDR
  ○ Integrated development environment
JTRS

- Principal communications initiative for U.S. DoD
  - Mandate exists for <2GHz systems
    - Likely to be expanded
  - Partitioned into four clusters
    - Cluster 1: Rotary aviation, ground vehicles
    - Cluster 2: manpack (SOCOM)
    - Cluster AMF: airborne, maritime, and fixed
    - Cluster 5: embedded and handheld
JTRS Fundamentals

- All systems based on SCA
  - Software Communications Architecture
    - Class inheritance structure
    - Class relationship/functionality
    - API description
    - Security supplement
      - Black vs. red system partition
SCA Class Relationship

- Resource-centric
  - System designed to create, connect, and destroy resources
  - Resources are software representation for components
- Largely mimics OS
  - Biggest difference is thread/process management
SCA Usage
Communications Paradigm

- **SCA v2.2**
  - Relies on CORBA
    - Common Object Request Broker Architecture
  - CORBA used to maintain platform and language independence
  - Workarounds available to bypass CORBA
    - Namely allocateCapacities() call on Device

- **SCA v3.0 (to be released soon)**
  - Provides structures to bypass CORBA
    - Guidelines for interface description for DSPs, FPGAs, and ASICs
API

- Based on building blocks
  - Allows API re-use
- Supports PHY and higher functionality
- Major component missing is smart antenna support
  - Current antenna interface inadequate
    - Does not provide support for advanced functionality and inter-layer control and communication necessary for Smart Antenna
SCA-Related Research

- Open-source CF development
- Smart Antenna API
- Integration of test equipment
- Power-aware SDR
- Integrated development environment
Open-Source CF Development

- Only currently-available open-source CF in Java
  - Provided by CRC (Canadian)
  - Most electrical engineers familiar with C/C++, not Java
- Multiple C++ CF available
  - None are open-source
- Open-source C++ necessary
  - Universities and other small-budget research centers do not have means to purchase C++ CF
  - Open-source effort provides forum for whole community to pool knowledge-base
Open-Source SCA Implementation:: Embedded

- Open-source C++ SCA v2.2 implementation
  - Covers major functionality necessary to support waveform
- To be released as open-source on July 2004
- Largely volunteer effort by VT/MPRG students and researchers
- First functional demonstration on June 10th, 2004
  - Integrated C++, embedded device (68HC11), and MATLAB into single waveform
Smart Antenna API

- API provides a standardized entry point into the implementation
  - Generally described in terms of IDL
    - Can support abstract concepts such as inheritance and "wrappers"
      - Not necessarily the same as in C++
  - Relatively easy to create API to support a small set of smart antenna implementations
    - Difficult to create API for comprehensive smart antenna support
Smart Antenna API Issues

- Controls broad information set
  - Hardware
    - Antenna (including calibration info)
    - RF (related to cositing information)
    - Modem
    - Algorithm (MMSE, MSINR, switch, selected, others)
  - Network
    - Acquisition
    - Coordination of nodes
    - Control
    - Power control
    - Channel allocation
    - MAC
    - QoS (application-level or lower)
Research Goal

- Develop comprehensive API
- Submit API to relevant bodies
  - JTRS (SCA v. 3.0+)
  - OMG
    - Controls civilian version of SCA
- Likely that submitted API will become standard API
  - Current community having difficulties in establishing adequate API
Integration of Test Equipment

- Desirable to make test an integral part of SDR development
  - Tektronix attempting to resolve this problem
- MPRG currently integrating multiple test equipment platforms into OSSIE
  - Arbitrary waveform generator
  - Wireless communication analyzer
  - Digital phosphor oscilloscope
  - Logic analyzer
Power-Aware SDR

- Integration of power-control technologies desirable in SDR
  - Easy for SDR to consume far more than hard-coded/wired application
    - Need for power to be addressed early

- Different approaches under study
  - Power management abstractions
  - Thread control for reduced power
  - Embedded Interface generation for power efficiency
Power Management Abstractions

- Power management data structures need to be integrated into software structure

Points to resolve

- Location
  - Implement as additional class, concurrent service, or leave up to waveform?

- Portability
  - Translation of power consumption from one platform to the next
  - Mapping algorithms to power
    - Mapping additional hardware to overall system consumption (overhead)
  - Mapping of power-saving techniques to waveform description
Thread Control for Reduced Power

- Radio will need to support concurrent functionality
  - System-wide “thread” management needed
    - Not limited to scope of OS
  - Appropriate thread management has multiple benefits
    - Limits problems with semaphores (signaling)
    - Allows the use of variable device clocks
      - Reduces power consumption
    - Can be extended to implement sleep cycles
      - Set thread priority to zero to set device to sleep
Embedded Interface Generation for Power Efficiency

- SCA v. 3.0 allows for embedded bypass of CORBA
  - Provides detailed API for interface generation
  - Automated generation of interface would significantly reduce development complexity
    - System-optimized code would reduce power consumption
      - Can be integrated into thread management for system
    - Automation of process can allow for standardized partitioning
      - Optimal selection of run-time versus compile-time features
Integrated Development Environment for Efficient Design

- Sequential development (traditional)
  - Strong partition between simulation and development
- Difficult to reconcile designs after hardware problems found
  - Examples
    - Non-linearities in RF hardware
    - Additional tracking structures for symbol synchronization

![Flowchart Diagram]

1. **User requirements and initial system specifications**
2. **Simulate and verify performance**
3. **Performance specifications satisfied?**
   - Yes: Hardware/software development
   - No: Loop adjusting HW/SW parameters until specifications met
4. **Performance specifications satisfied?**
   - Yes: Test system design
   - No: Adjust system requirements and specifications
5. **Performance specifications satisfied?**
   - Yes: Validate initial simulation model against HW and SW (final sanity check)
   - No: Find errors (debug)
6. **Validation check successful?**
   - Yes: Design process complete
   - No: Return to previous step
Development with SCA

- SCA allows for MATLAB-only system
  - Specific components can be swapped with functional hardware
    - Functional hardware is now part of “simulation”
  - Process blurs simulation and design/development

![Diagram of microprocessor operating environment (operating system + OSSIE)]

![Diagram showing extraction of simulated entity into SDR implementation]
**Integrated Approach**

Integration of simulation and development

- Allows for simulation-level treatment of hardware issues
- Makes reconciliation of theory and implementation far easier
- No final validation required
  - Final validation integral part of development process
  - Channel included in evaluation
Conclusion

- Software-Defined Radio research has wide scope
  - Reconfigurable hardware
  - Network management
  - Interference management
  - Antenna implementation and application
  - Software development
  - Node/System management
  - System development

- JTRS has been a catalyst for research
  - Set down architectural foundation
    - Provides starting point for resolution of SDR issues