Array Like Runtime Reconfigurable MIMO Detector for 802.11n WLAN: A design case study

Pankaj Bhagawat
Rajballav Dash
Gwan Choi
Texas A&M University-CollegeStation
Outline

- Background
- MIMO Detection as a Tree Search
- Related Work
- Fixed Sphere Decoder and Architecture
- Architectural Space Exploration
- Integration Issues in a Communication System
Standards using MIMO and Requirements

- LAN: 802.11n
- WAN: WiMax, LTE etc
- 1Gbps systems like WIGWAM
- All support multiple modulation and coding schemes (MCS)
- Very high throughput requirements at BB
- Ease of integration
Why MIMO?

- Spatial Multiplexing is especially attractive
- The transmitter is able to send out multiple data streams on the same frequency
- More throughput without extra BW
MIMO System

- Binary data is encoded with a rate R code
- Coded bits grouped (in $\log_2 \eta$) and mapped to a $\eta$-ary QAM symbol
- *Independent* QAM symbols transmitted over multiple antennas
• Each Rx antenna sees *weighted superposition* of (or interference from) signals from all Tx antennas
  - Rx₁ sees $s_1 h_{11} + s_2 h_{21}$ and Rx₂ sees $s_1 h_{12} + s_2 h_{22}$
  - $h_{11}, h_{12} \ldots$ are random channel gains (fading)
  - Noise samples $n_1$ and $n_2$ further corrupts the interference laden signals
• MIMO detection involves removing the interference in presence of noise
• Knowledge of channel gains is assumed (provided by channel estimator)
MIMO System

- Can be written in a matrix form
  \[ y = Hs + n \]

- Best estimate (ML) \( \hat{s} \) of \( s \) is such that it minimizes
  \[ \| y - Hs \|^2 \]

- \( H = QR \), \( R \) is upper triangular matrix. Pick \( s \) such that
  \[ \| \hat{y} - Rs \|^2 \] is minimized
Tree Structure for 4x4-16 QAM MIMO System

- Due to the upper triangular nature of $R$, best estimate of $\mathbf{s}$ can be treated as a tree search
- Incremental metric $|e_i|$, is always positive
- $c_{i+1}$, and hence $|e_i|$, depends only path history and the present QAM symbol $s_i$
- Path corresponding to the leaf node with least $d$ corresponds to the best *hard* estimate of $\mathbf{s}$ (ML)
Related Work

- K-best algorithm takes a BFS approach and retains K “best” paths at each level of the tree
  - Fixed throughput can be achieved
  - Sorting is very expensive
  - Large memory to store intermediate results depending on the value of “K”
  - The value of “K” is modulation scheme dependent => difficult to achieve high resource utilization in a reconfigurable environment

- DFS based approach: Single Tree Search (STS), Repeated Tree Search (RTS)
  - Highly random throughput, throughput not high enough
  - Hard to parallelize and pipeline

- Linear Detectors are low complexity but has poor BER/FER
FSD Algorithm

Each node has $\eta$ children.

\[
d_4(s^{(4)}) = \|y_4 - R_{44} \cdot s_4\|^2
\]
\[
d_3(s^{(3)}) = d_4(s^{(4)}) + \|y_3 - R_{34} \cdot s_4 - R_{33} \cdot s_3\|^2
\]
\[
d_2(s^{(2)}) = d_3(s^{(3)}) + \|y_2 - R_{24} \cdot s_4 - R_{23} \cdot s_3 - R_{22} \cdot s_2\|^2
\]
\[
d_1(s^{(1)}) = d_2(s^{(2)}) + \|y_1 - R_{12} \cdot s_2 - R_{13} \cdot s_3 - R_{14} \cdot s_4 - R_{11} \cdot s_1\|^2
\]
Metric Computation Unit (MCU)

\[ d_1(s^{(1)}) = d_2(s^{(2)}) + \|y_i - R_{12} \cdot s_2 - R_{13} \cdot s_3 - R_{14} \cdot s_4 - R_{11} \cdot s_1 \|^2 \]

- Product Computer: Computes the products \( R_{ij} s_j \)
- Slicer finds the best child, MF[1:0] configures the slicer to operate for different modulation schemes

\[ c_{i+1}(s^{(i+1)}) = \hat{y}_i - \sum_{j=i+1}^{M_f} R_{ij} \cdot s_j \]

\[ |e_i(s^{(i)})|^2 = |c_{i+1}(s^{(i+1)}) - R_{ii} \cdot s_i|^2 \]

\[ d_i(s^{(i)}) = d_{i+1}(s^{(i+1)}) + |e_i(s^{(i)})|^2 \]
Key Features

- Systolic type array of processing elements
- On the fly reconfiguration possible
- Highly pipeline-able
- Data and control flow is forward flowing
- Fixed throughput for a given modulation scheme
- Very high resource utilization
  - MCUs *matched* to level of the tree
  - Pipeline is not broken
• Timing Diagram

• Throughput = \((4 \log_2(\eta)/\eta)f\), where \(f\) is the clock frequency.

• High resource utilization, pipeline is not broken, MCUs are \textit{matched} to tree levels
Packet structure in 802.11n type systems

**Perels, D. et. al. "ASIC Implementation of a MIMO-OFDM Transceiver for 192 Mbps WLANs." ESSCIRC 2005**
MIMO-OFDM System

- MOS = MIMO-OFDM Symbol
- A detector core has to process 52 tones in 3.6 microseconds.
Detector Array

- Processing time for 52 tones, $T_p$, is given by $T_p = 52 \cdot \lfloor \eta / m \rfloor \cdot C_d / (k + 1)$
- $\eta$ depends on the modulation format used, $\eta = 4$ (QPSK), 16 (16-QAM), 64 (64-QAM).
- $C_d$ is the combinational delay of the un-pipelined array.
Architectural Space

- In actual simulations $T_p = 3000\text{ns}$ (to account for 15-20% pessimism factor).
Power, Delay, and Area Estimation

- Power consumption mainly due to logic and clock network.
- Clock network is modeled as a symmetrical mesh.
  - Global clock network power was estimated using HSPICE
  - Local power is estimated using capacitive load due to number of flops driven by a local clock buffer
- Synopsys DC was used to estimate logic power, area, and timing of the detector
- DC retiming utility was used to introduce and balance the pipes.
Architectural Exploration for Low Area

- Find \((m,k)\) such that it meets the throughput requirements for all modes and has minimum area.
- 64-QAM is most intensive \(\Rightarrow\) find \((m,k)\) to meet throughput requirement for it
- \(M=3,\) and \(k=8\) is able to meet the requirement with lowest area.
Architectural Exploration for Low Power

- Power consumption profiles differ with the modulation modes
- Power consumption has to be optimized over all the modes
- Find \((m,k)\) such that the aggregate power is minimized and detector still meet the throughput requirements.
Aggregate power $Pow_{agg} = Prob(QPSK) * Pow(QPSK) + Prob(16QAM) * Pow(16QAM) + Prob(64QAM) * Pow(64QAM)$

All the probabilities can be assumed $=1/3$, since there is no a-priori knowledge

$Pow_{agg}$ as a function of $(m,k)$ is shown in the figure

- Not all points meet the throughput req.
- Admissible points are shown as stems.
- $m=4$ and $k=5$ => lowest power while meeting throughput req.
## Results

### OUR ASIC IMPLEMENTATION DETAILS

<table>
<thead>
<tr>
<th>Design Parameters</th>
<th>Area Optimized Nangate 45nm PDK</th>
<th>Power Optimized Nangate 45nm PDK</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target Tech. Library</td>
<td>Nangate 45nm PDK</td>
<td>Nangate 45nm PDK</td>
</tr>
<tr>
<td>Pipeline Stages (k)</td>
<td>8</td>
<td>5</td>
</tr>
<tr>
<td>Parallelism (m)</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Gate Equivalent</td>
<td>58.2k</td>
<td>67.7k</td>
</tr>
<tr>
<td>Power Consumption</td>
<td>11.91mW</td>
<td>9.7mW</td>
</tr>
<tr>
<td>Frequency: QPSK</td>
<td>38.8MHz</td>
<td>18MHz</td>
</tr>
<tr>
<td>Frequency: 16-QAM</td>
<td>116.3MHz</td>
<td>71.8MHz</td>
</tr>
<tr>
<td>Frequency: 64-QAM</td>
<td>426.6MHz</td>
<td>287.3MHz</td>
</tr>
<tr>
<td>Throughput Requirement: QPSK</td>
<td>115.6Mbps</td>
<td>115.6Mbps</td>
</tr>
<tr>
<td>Throughput Achieved: QPSK</td>
<td>155Mbps</td>
<td>144Mbps</td>
</tr>
<tr>
<td>Throughput Requirement: 16-QAM</td>
<td>231.1Mbps</td>
<td>231.1Mbps</td>
</tr>
<tr>
<td>Throughput Achieved: 16-QAM</td>
<td>310.13Mbps</td>
<td>287.2Mbps</td>
</tr>
<tr>
<td>Throughput Requirement: 64-QAM</td>
<td>346.7Mbps</td>
<td>346.7Mbps</td>
</tr>
<tr>
<td>Throughput Achieved: 64-QAM</td>
<td>465.38Mbps</td>
<td>430.95Mbps</td>
</tr>
</tbody>
</table>
Thank You!