SOFTWARE IMPLEMENTATION OF THE IEEE 802.11A/P PHYSICAL LAYER

SDR`12 – WInnComm Europe
27 – 29 June, 2012 ~ Brussels, Belgium
T. Cupaiuolo, D. Lo Iacono, M. Siti and M. Odoni
Advanced System Technologies
STMicroelectronics, Agrate Brianza, Italy

Daniele Lo Iacono
Outline

- The system
  - Wireless Access in Vehicular Environments (WAVE): IEEE 802.11p
  - Comparison with IEEE 802.11a/g
- A Software Defined Radio (SDR) implementation approach: the BPE baseband communication platform
- Digital baseband implementation
  - Reference system model
  - 802.11p: Data-aided channel estimation
  - Customization and code profiling
- Results
IEEE 802.11p WAVE

- Requirements
  - Fast access as a priority (latency <50 ms)
  - Mobility (>60Km/h) and Range (~1Km)
  - Robustness and reliability
  - Security

- Applications
  - Vehicle safety (emergency warning systems, Intersection collision avoidance, forward collision warning)
  - Tolling
  - Infotainment
  - Traffic management
  - Cooperative Adaptive Cruise Control

- Comparison with 802.11a/g
  - 10 MHz OFDM bandwidth (vs 20 MHz): max PHY data rate 27 Mbit/s (vs. 54)
  - 5.9 GHz carrier frequency
  - digital baseband: added support for mobility → Data-aided channel estimation
The BPE baseband communication platform

customizable coarse-grain hardware operators

d-unit bank

reconfigurable data-path

distributed embedded memory

scheduler and dispatcher

d-unit bank

routing mesh

instruction memory

d-memory bank

system bus interface

fetch & decoding

memory management

registers space

data-port

b-instruction execution

d-instruction scheduler

fetch & decoding

fetch & decoding
Flow-control: b-instruction

\[ \text{out} = \text{opcode}(\text{in0}, \text{in1}, \text{in2}) \]

b-instruction are also used to set the way d-instruction will access the memory bank.

Register file

System bus interface

Fetch & decoding

Instruction memory

Memory management

Registers space

Data port

B-instruction execution

B-instruction scheduler

D-unit bank

Routing mesh

D-memory bank

B-instruction execution unit
Vector processing: d-instruction

\[ \text{out} = \text{unit1}.\text{opcode}(\text{unit0}, \text{in0}, \text{in1}) \]

- Processing units performing parallel and pipelined vector processing
- Bank of static memories for vector allocation
- Routing mesh dynamically configuring unit-memory and unit-unit connections
- D-instruction scheduling unit
- D-unit bank
- Routing mesh
- D-memory bank
- Instruction memory
- D-instruction scheduler
- Fetch & decoding
- Memory management
- Registers space
- Data-port
- System bus interface
Algorithm mapping: macros

maco made by two parallel branches each performing pipelined processing among different units

parallel and pipelined processing to reduce execution time and memory accesses

v9 = arith3.mul(comm1, v6)
comm1.qt(arith2, v5)
v8 = arith2.sub(v4, arith1)
v7 = comm0.ed(arith0, v3)
arith0.mul(v0, v1);
arith1.mul(v0, v2)
Pipeline of macros

conflicts on shared memory access inhibit a full pipelined processing

use of memory alias (ping-pong mechanism) at intermediate stage of processing
Multi-thread

Single-thread execution

OFDM symbol #1
function #1
function #2
function #3

OFDM symbol #2
function #1
function #2
function #3

Multi-thread (3) execution

function #1
OFDM #1
OFDM #2
OFDM #3
OFDM #4

function #2
OFDM #1
OFDM #2
OFDM #3
OFDM #4

function #3
OFDM #1
OFDM #2
OFDM #3
OFDM #4
IEEE 802.11a/p reference system model

source bits

encoder → puncturer+interleaver → mapper → IFFT → upsampling/filtering → D/A

channel

A/D → filter → FFT → equalizer → de-map → de-int de-punct → Viterbi decoding

decoded bits

discard pilot and virtual sub-carriers

channel estimation

802.11p: data-aided channel estimation
Data-aided channel estimation basic idea:

1. Data detection of the current received OFDM symbol using channel estimation corresponding to the previous OFDM symbol.
2. The channel corresponding to the current OFDM symbol is estimated by using the estimated data QAM symbols.

→ Data detection through simple hard decision detection (HDD):
   → Low extra complexity
   → Low latency compared to 802.11a/g
Data-aided channel estimation (2/2)

Initial CE based on the LTS field

For successive OFDM symbols (SIG and DATA) CE tracked exploiting both pilot and the estimated data symbols
Multimode 11a/p receiver data pipeline

- Filter
- FFT
- Equalizer
- De-map
- De-int de-punct
- Viterbi decoding
- Synchronizer
- Channel estimation

**Recursive update:** processing bottleneck which does not allow to build a pipeline as for 802.11a

**Symbol processing time**

11a

11p
Multimode 11a/p receiver code profiling

<table>
<thead>
<tr>
<th>function</th>
<th>11a/g (clock cycles)</th>
<th>11p (clock cycles)</th>
</tr>
</thead>
<tbody>
<tr>
<td>filter</td>
<td>162</td>
<td></td>
</tr>
<tr>
<td>synchronizer</td>
<td></td>
<td>1536 (latency)</td>
</tr>
<tr>
<td>FFT (iFFT)</td>
<td>200 (radix-4)</td>
<td></td>
</tr>
<tr>
<td>channel estimation</td>
<td>64 (@LTS)</td>
<td>759 (data-aided, hard-detection)</td>
</tr>
<tr>
<td>equalizer</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>de-mapper</td>
<td>348 (MCS #7)</td>
<td></td>
</tr>
<tr>
<td>de-interleaver / de-puncturer</td>
<td>408 (MCS #7)</td>
<td></td>
</tr>
<tr>
<td>OFDM symbol single-thread (@250MHz)</td>
<td>1432 (5.7µs)</td>
<td>2150 (8.6µs)</td>
</tr>
<tr>
<td>OFDM symbol multi-thread (@250MHz)</td>
<td>596 (2.4µs)</td>
<td>1490 (5.9µs)</td>
</tr>
</tbody>
</table>
Conclusions

- The BPE software programmable architecture has support for:
  - macro building
  - (macro-) instructions pipelining
  - emulate memory ping-pong access
  - Multi-threading

- Algorithm profiling on the BPE
  - Translate the algorithm steps into macros
  - Build the macro-pipeline

- PHY profiling on the BPE (MCS #7, @250 MHz)
  - 802.11a/g: 5.7 μs (single thread) ~ 2.4 μs (three threads) (i.e. 54 Mbit/s)
  - 802.11p: 8.6 μs (single thread) ~ 5.9 μs (two threads) (i.e. 27 Mbit/s)

- Future steps
  - 802.11p 20 MHz optional mode
  - Soft decision directed DA CE (FEC based, i.e. Viterbi decoding)
  - to address these and other issues: investigating architectural enhancements
    (including the idea of a “cluster of BPE”)