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

STMicroelectronics

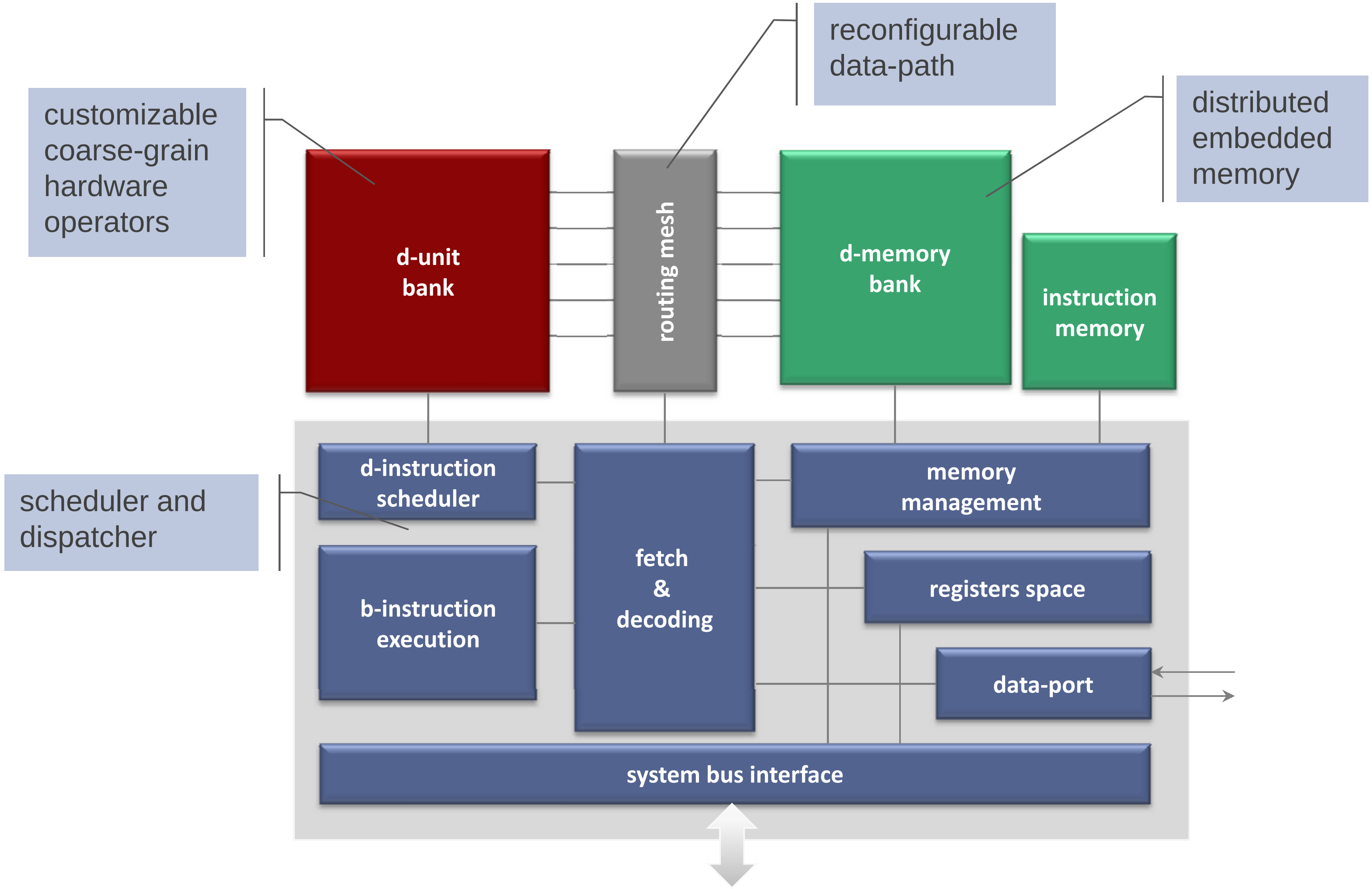
Outline

- 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

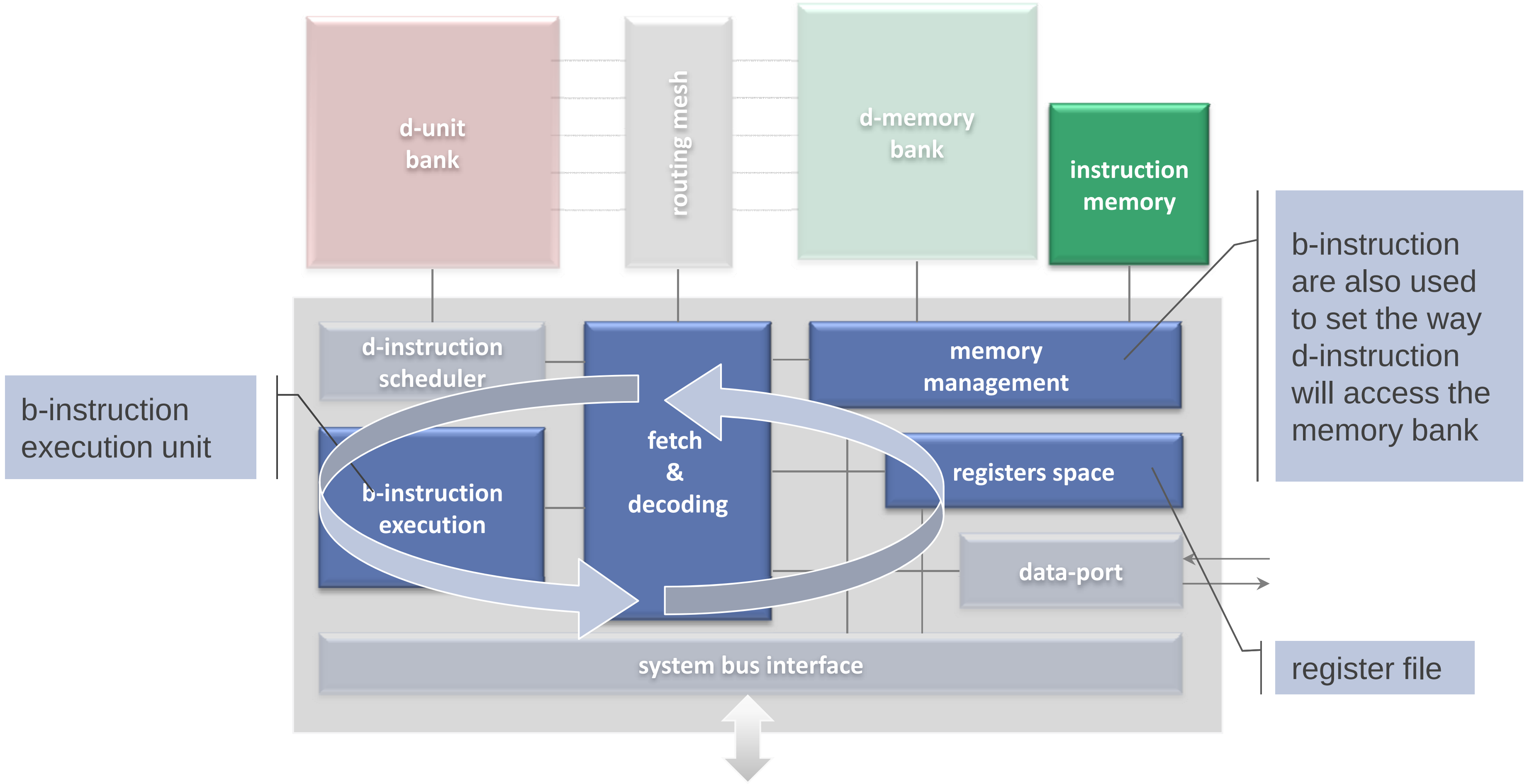
- Requirements
  - Fast access as a priority (latency <50 ms)
  - Mobility (>60Km/h) and Range (~1Km)
  - Robustness and reliability
  - Security
- Applications
  - Vehicle safety (emergency warning systems, Intersection collision avoidance, forward collision warning)
  - Tolling
  - Infotainment
  - 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



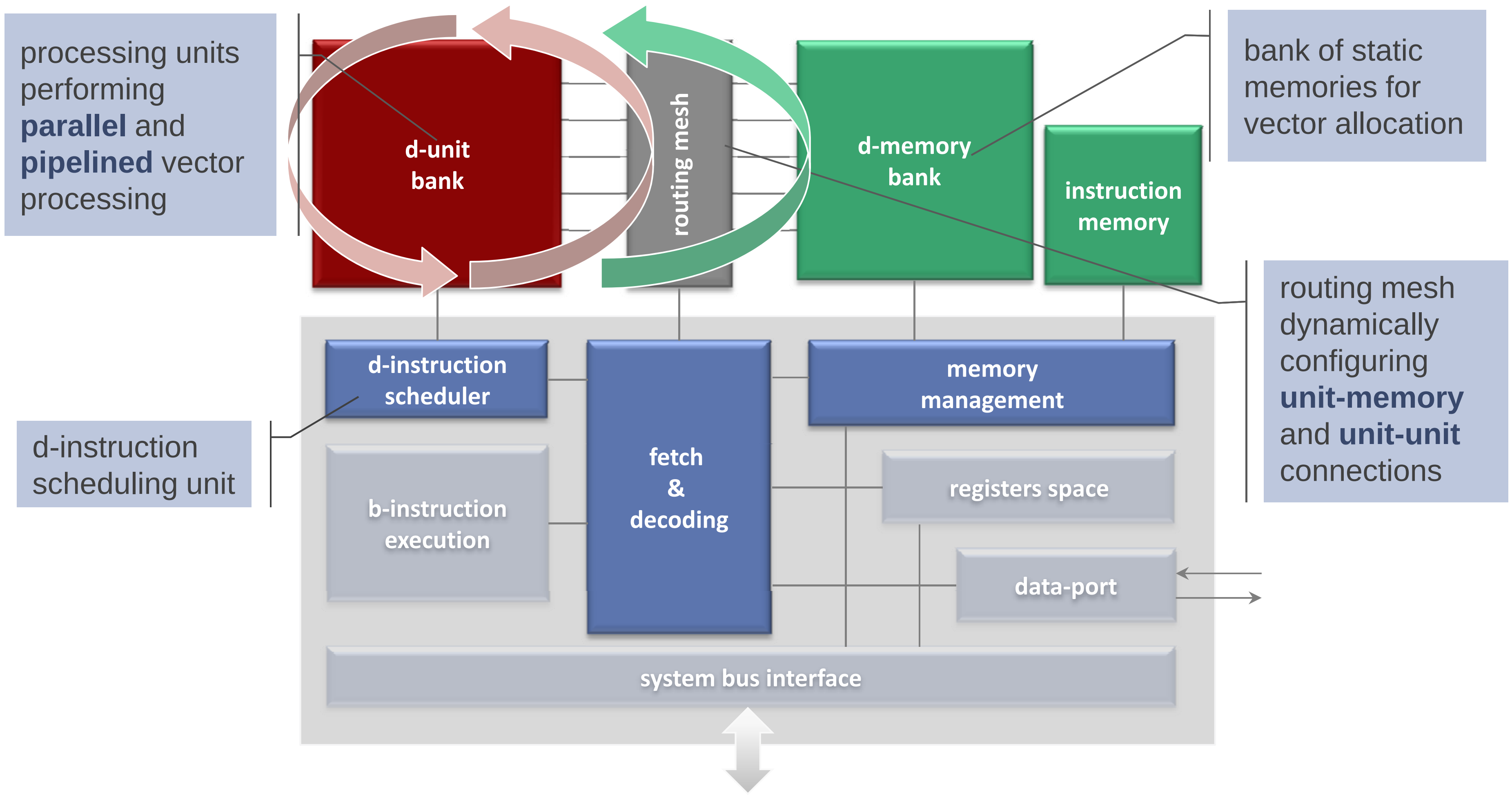
Flow-control: b-instruction

```
out = opcode(in0,in1,in2)
```

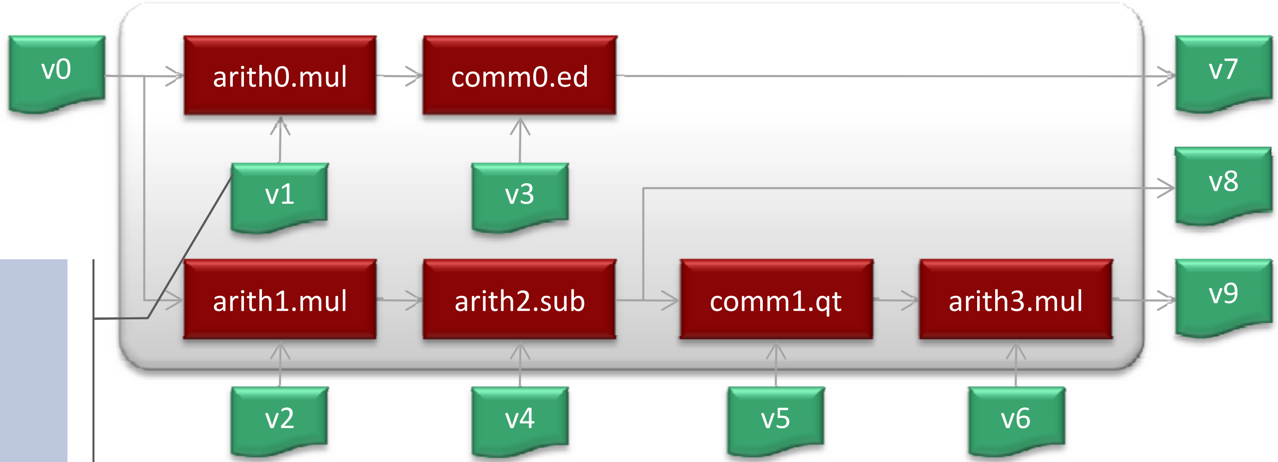


Vector processing: d-instruction

```
out = unit1.opcode(unit0,in0,in1)
```

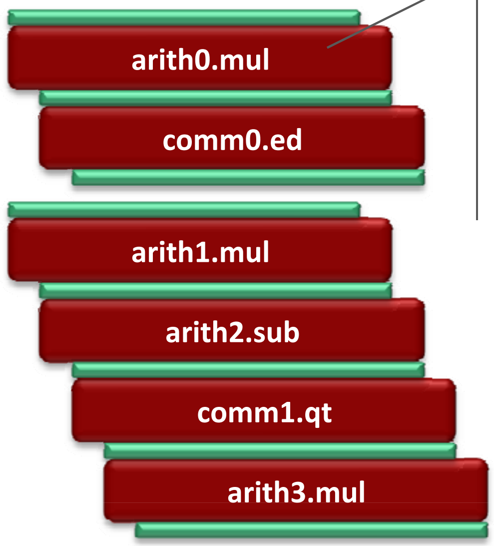


Algorithm mapping: macros



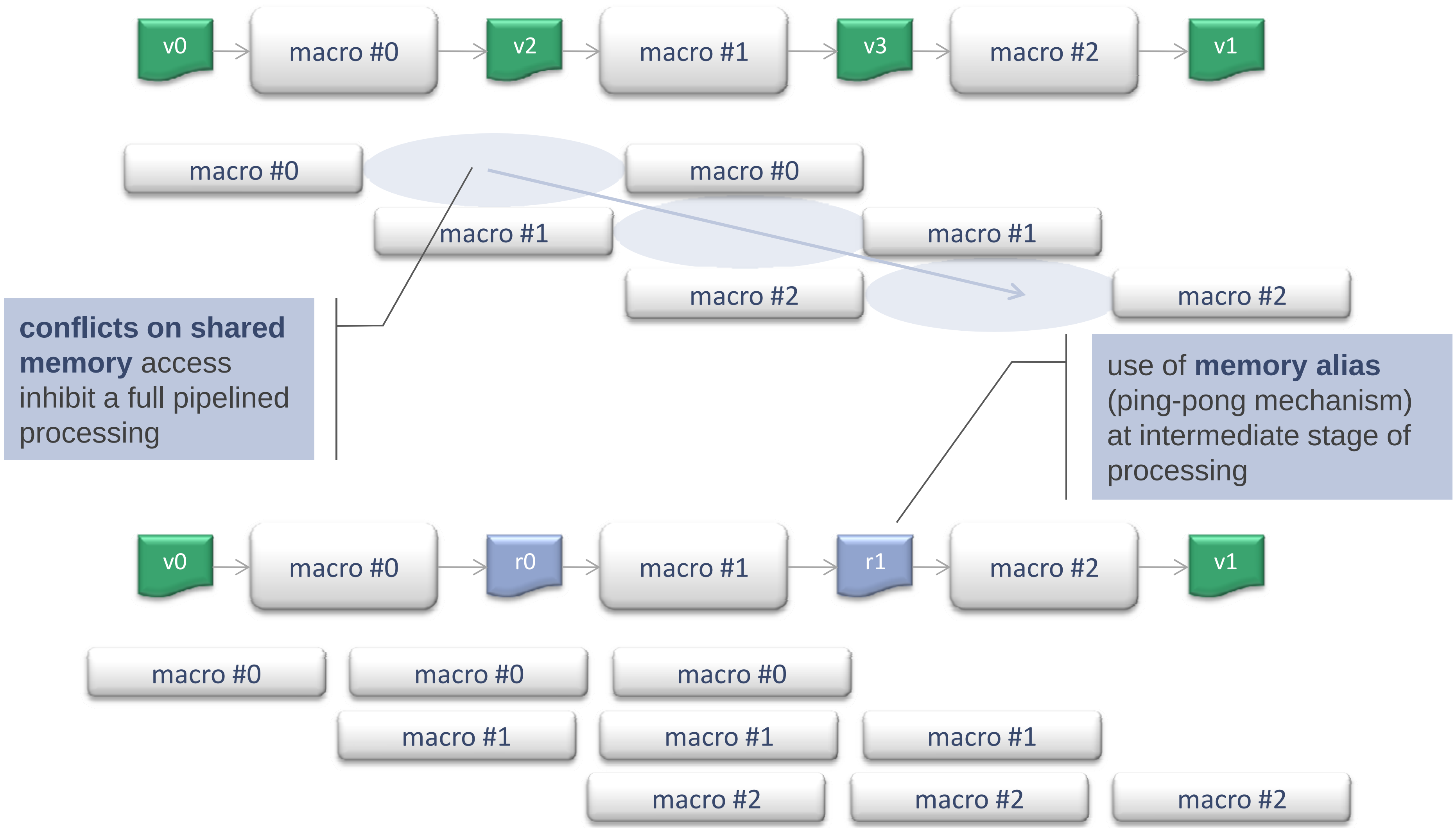
macro made by two **parallel** branches each performing **pipelined** processing among different units

```
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)
```



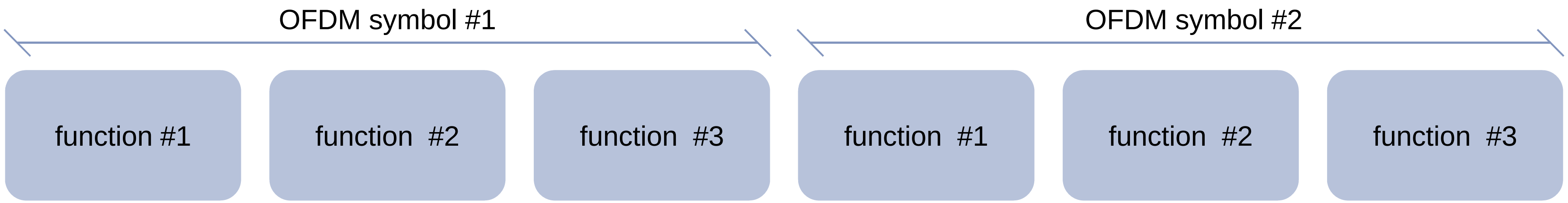
parallel and pipelined processing to reduce **execution time** and **memory accesses**

Pipeline of macros

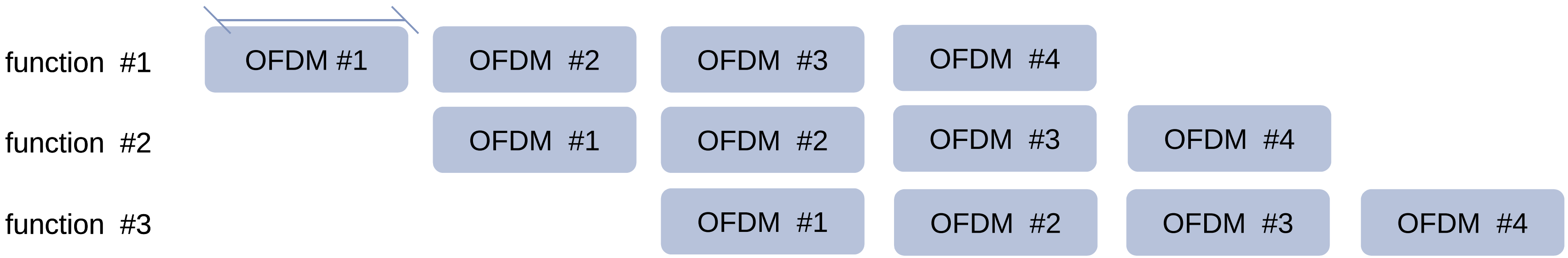


Multi-thread

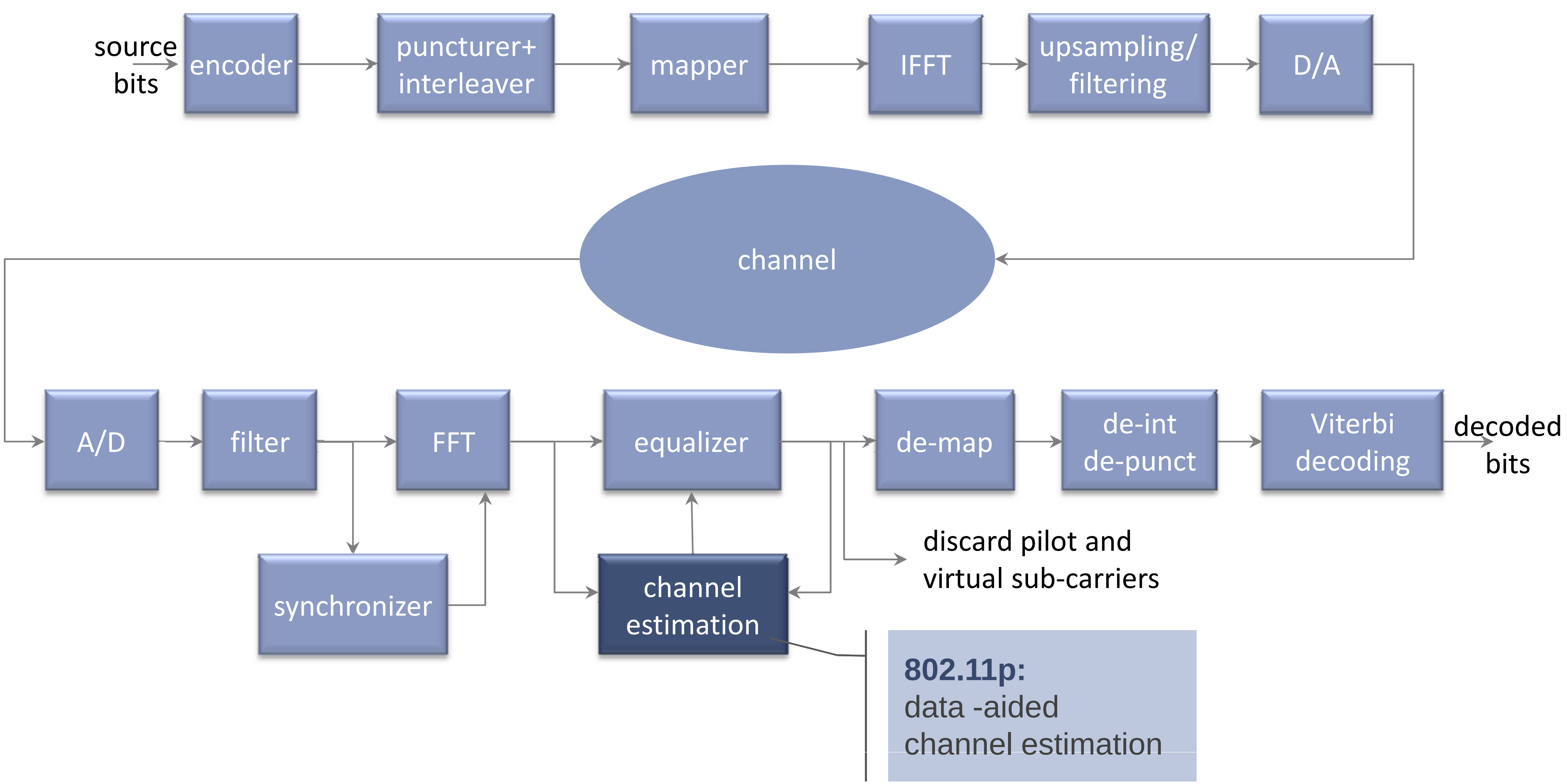
Single-thread execution



Multi-thread (3) execution



IEEE 802.11a/p reference system model



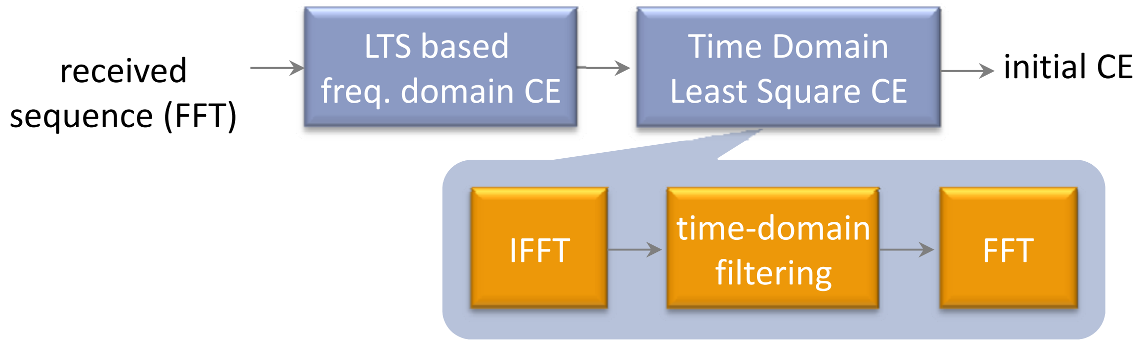
Data-aided channel estimation (1/2)

- 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

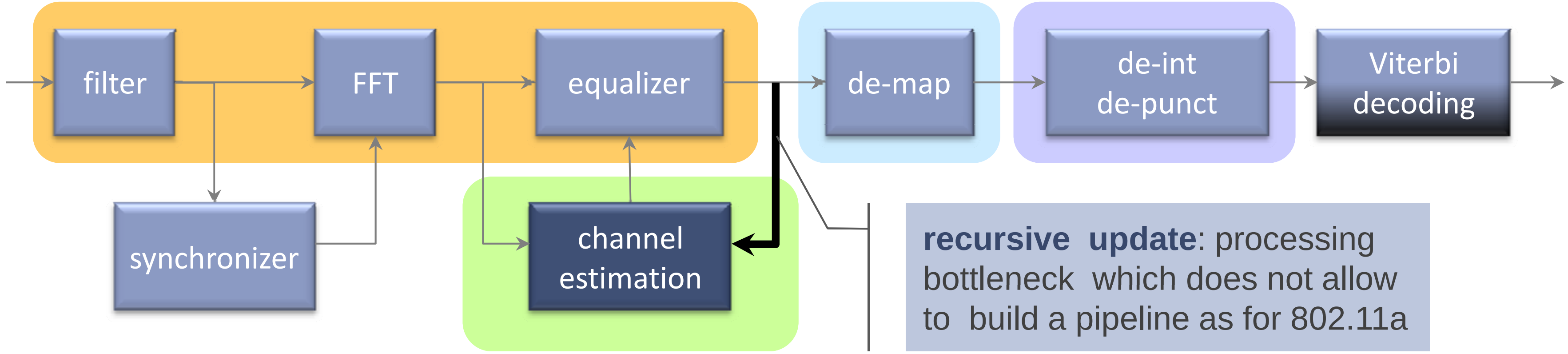
Data-aided channel estimation (2/2)

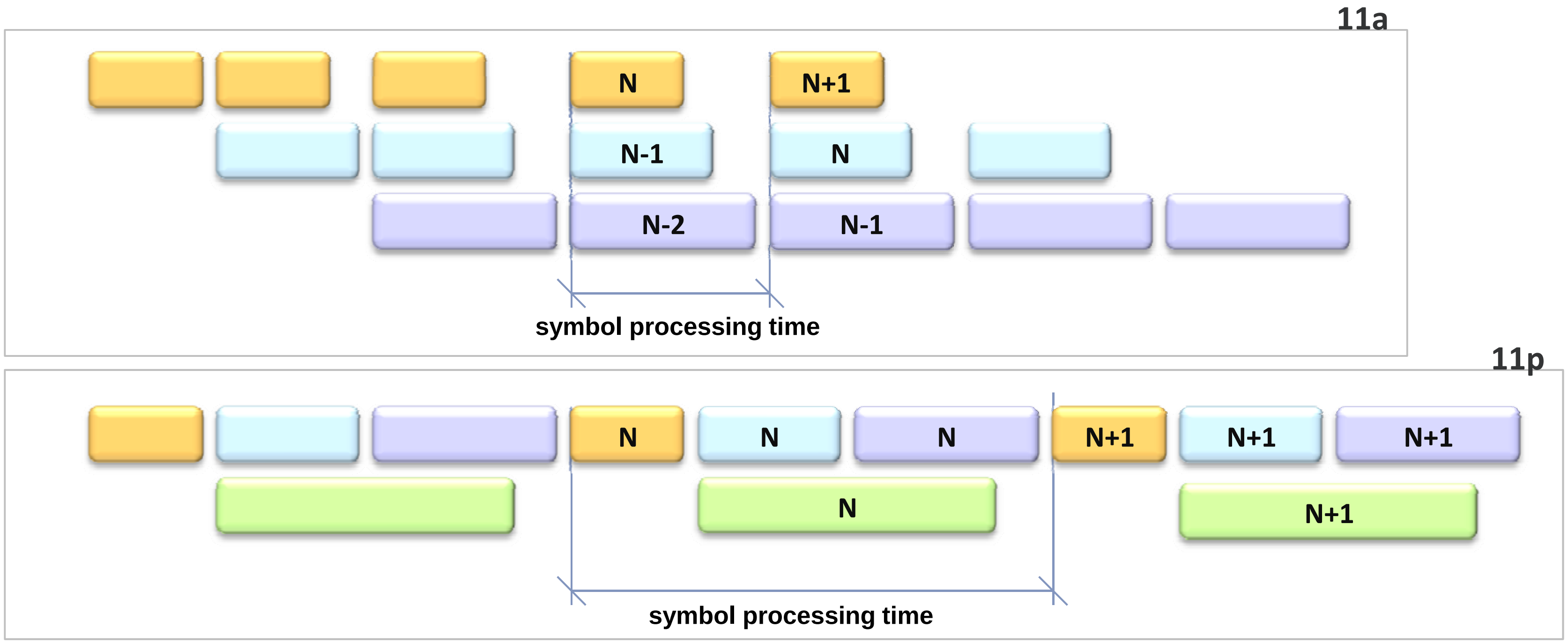
Initial CE based on the LTS field



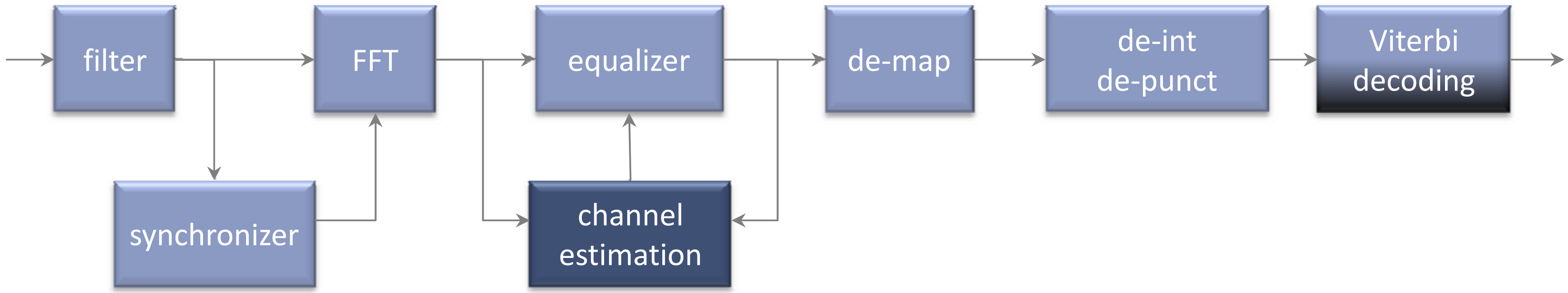
For successive OFDM symbols (SIG and DATA) CE tracked exploiting both pilot and the estimated data symbols

Multimode 11a/p receiver data pipeline





Multimode 11a/p receiver code profiling



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

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")