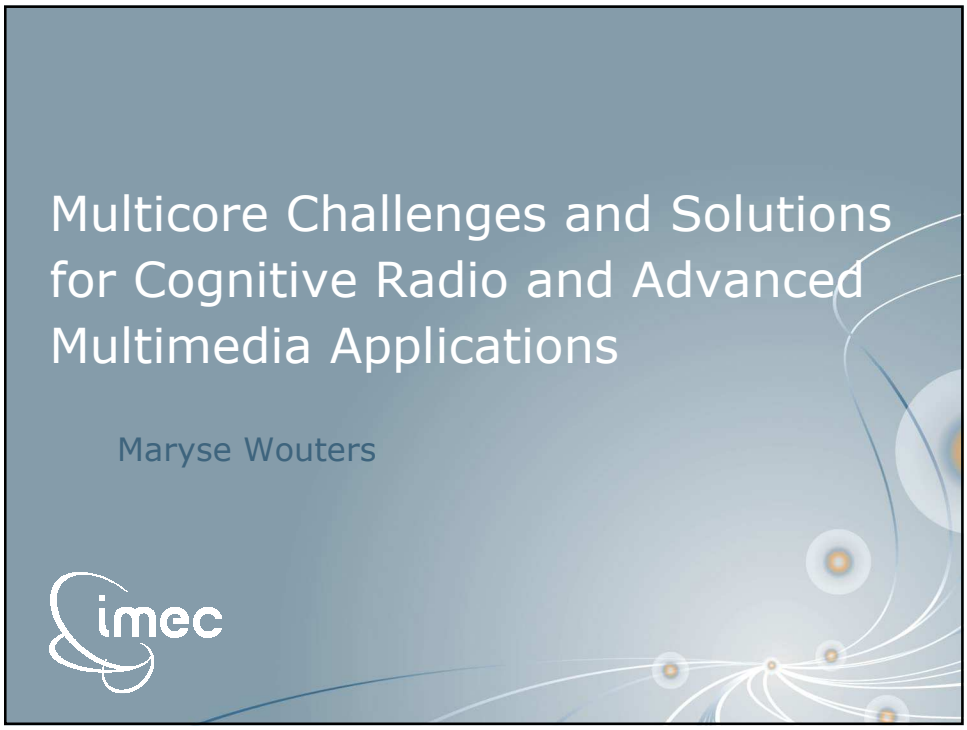

imec
Designing for Embedded Parallel Computing Platforms: Architectures, Design Tools and Applications
April 24, 2009


Multicore Challenges and Solutions for Cognitive Radio and Advanced Multimedia Applications
Maryse Wouters
imec

Outline

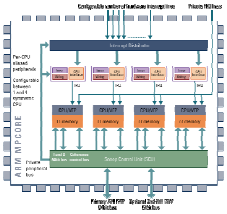
- Programming multiprocessor platforms
  - Multiprocessor platforms are here today and they are here to stay!
  - Current mismatch between platform and programming model
  - How to map sequential code on a parallel platform?
- IMEC's MPSoC tool suite
  - Eclipse *Code Cleaning* plug-in to get your C code in parallel shape
  - MH tool: **M**emory **H**ierarchy optimization tool
  - MPA tool: a **M**ultiprocessor **P**arallelization **A**ssistant
- A concrete case: towards Gbit/s Cognitive Reconfigurable Radio and MPEG4-ENC
- Run time management

Outline

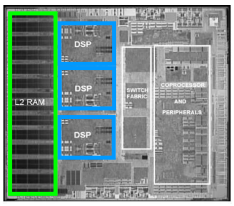
- Programming multiprocessor platforms
  - Multiprocessor platforms are here today and they are here to stay!
  - Current mismatch between platform and programming model
  - How to map sequential code on a parallel platform?
- IMEC's MPSoC tool suite
  - Eclipse *Code Cleaning* plug-in to get your C code in parallel shape
  - MH tool: **M**emory **H**ierarchy optimization tool
  - MPA tool: a **M**ultiprocessor **P**arallelization **A**ssistant
- A concrete case: towards Gbit/s Cognitive Reconfigurable Radio

Multiprocessor platforms are here today …
… and they are here to stay!

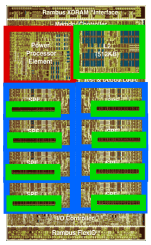
ARM

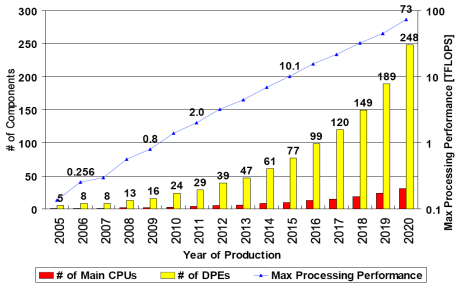


Texas Instruments



IBM Cell





How to efficiently program them?

What are the fundamental issues?

Platform Evolution – Embedded System

Single processor, sequential programs

- Designer uses sequential C code and assumes a single (shared) memory space
- Platform consist of a single processor with a cache (hierarchy?) and a main memory, interconnected by a bus.
- There's a good match between programming model and computing platform
- Designer focuses on algorithmic development and code optimizations

Multi processor, parallel programs

- Memory access will not scale up
- Communication and synchronization becomes problematic
- Debugging is a nightmare
- Existing programming model will break
- Parallelization, componentization and composability
- Keep scalability across platform generations (#processors/amount of memory/available bandwidth)
- Keep retargetability to different vendors
- Multi-application predictability

How can we program multi processor platforms?

- Buying a commercial MP-RTOS and use the primitives available
  - Is just an abstraction layer on top of the MPSoC platform … all problems remain
- Pre-parallelized libraries
  - Limited flexibility and no context awareness
  - Platform specific
- Create parallel threaded code using explicit data communication with message passing (MPI)
  - Parallelization is manual and error-prone
  - Explicit data communication needs to be added by designer.
  - A lot of existing algorithms do not map efficiently to a pure message passing model
- Create parallel threaded code assisted by tools like OpenMP.
  - Less manual work, but still error-prone.
  - Coherency problem still exists and is assumed to be solved in HW

This approach is an MPSoC programming showstopper!
We want sequential C programming with a single (shared) memory space

How do we program multi processor platforms now?

- How do we guarantee that it works for a single application
  - Simulation … over and over and over again
- And if it does not work or does not reach the required performance
  - Do it all again
  - And then: Simulation … over and over and over again

Is this how we want to continue?
We want sequential C programming with a single (shared) memory space

What do we need to get back to sequential C programming with a single (shared) memory space?

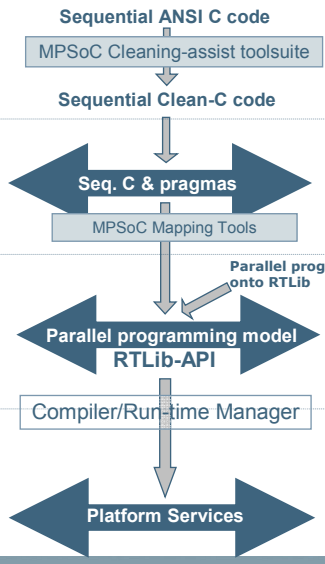
- Tool that performs parallelization and also adds synchronization, inter-task communication and manages the memory hierarchy
- Explicitly controlling (data) communication to avoid hardware cache coherency problem/scaling bottleneck
- Embedded SW designer remains on the "sequential code with single memory space" level. Designer is responsible for giving parallelization *hints*.
- Tools make the code-transition to the parallel world and make sure the transition is functionally correct and optimized for a given set of platform parameters.
- Tools that provide fast feedback on a sequential level!

Such tools minimize the error-prone and time-consuming work … over and over and over again.

Outline

- Programming multiprocessor platforms
  - Multiprocessor platforms are here today and they are here to stay!
  - Current mismatch between platform and programming model
  - How to map sequential code on a parallel platform?
- IMEC's MPSoC tool suite
  - Eclipse *Code Cleaning* plug-in to get your C code in parallel shape
  - MH tool: **M**emory **H**ierarchy optimization tool
  - MPA tool: a **M**ultiprocessor **P**arallelization **A**ssistant
  - Application on MPEG4-ENC
- A concrete case: towards Gbit/s Cognitive Reconfigurable Radio
- Run time management

MPSoC Programming Model & Flow



IMEC Programming model

- Sequential C
- Shared memory

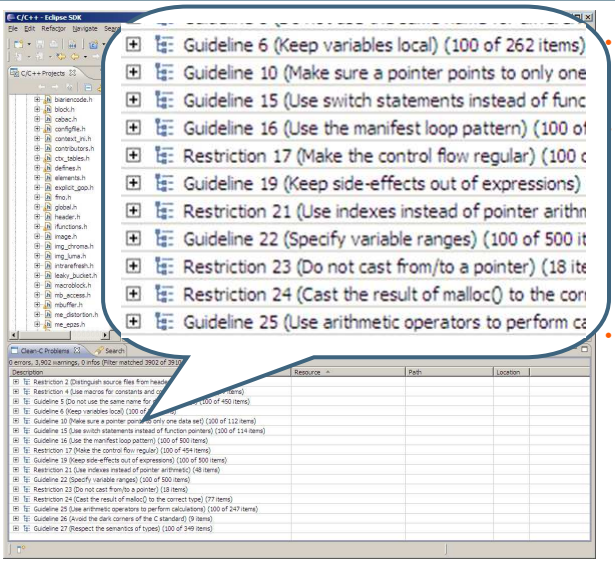
- Sequential C + pragmas
- Shared memory

- Threaded parallel programming on RTLib API
- Memory model:
  - Shared memory and message passing
  - Mixed to have best of both worlds

Platform services

Hardware components

CleanC: promoting an MPSoC-friendly coding style



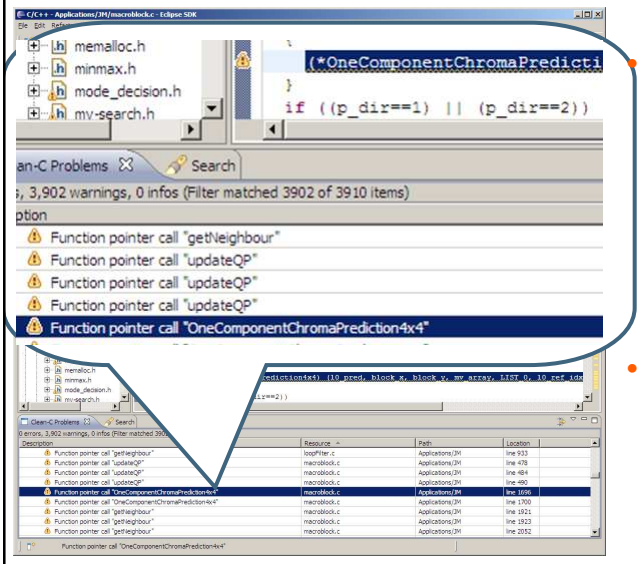
- Static analysis of arbitrary C code has its limitations

IMEC promotes a coding style that is MPSoC-friendly → CleanC

- 28 guidelines and restrictions on how to write C code
- Can analyze code for parallelization purposes

- To help convert arbitrary C code to CleanC code, IMEC develops a tool suite
  - Analyzes code for adherence to CleanC coding style
  - Provides code transformation support to "clean" C code
  - Integrated in Eclipse 3.3 / CDT 4.0 IDE

CleanC tool box

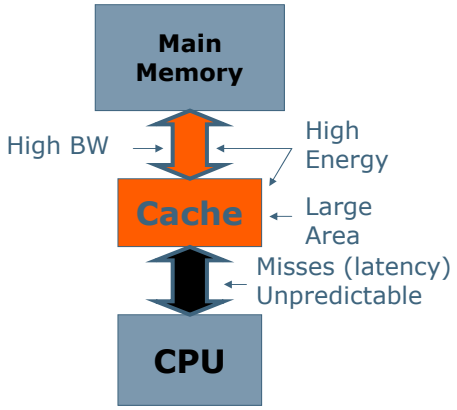


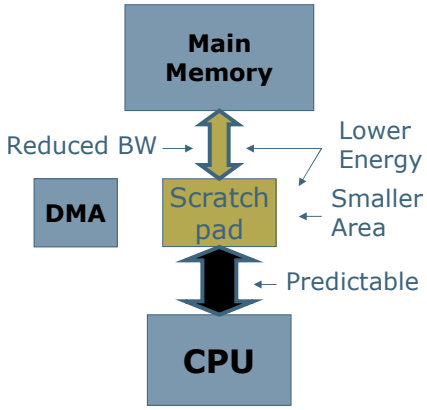
- Analysis tool implemented
  - Eclipse 3.3 / CDT 4.0 IDE plug-in
  - Restrictions implemented
  - Number of warnings reduced
  - First transformation implemented
  - Lots of visibility in the press
    - Free download available
- Future work
  - Support user specified "regions of interest" to focus on most relevant parts of code
  - Further implement code transformation support

Multi-Core Association

- Mutlicore Programming Practice (MPP)
  - "[…] *has a goal to develop a multicore software programming guide for the industry that will aid in improving consistency and understanding of multicore programming issues. Initially the group is working on best practices leveraging the C/C++ language to generate a guide of genuine value to engineers who are approaching multicore programming*. […]".
- Officially Working Group Member since end of September
  - Objective is to promote the CleanC guidelines and rules

Memory hierarchy: cache versus scratchpad

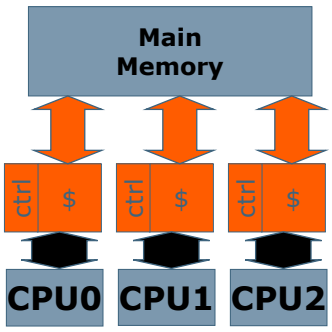


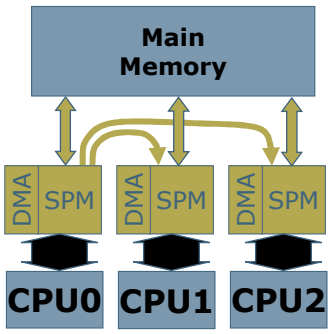


- Costly (power, cycles, BW, …)
- Unpredictable access latency
- Poor scaling to MP
- Simple to program

- More efficient (power, cycles, BW, …)
- More predictable
- Better scaling to MP
- Harder to program ➡ **Tool support needed!**

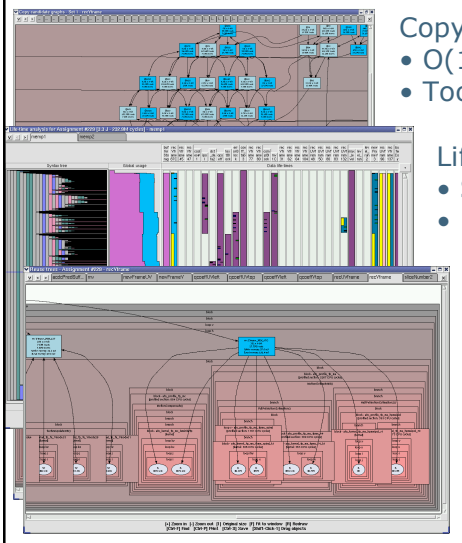
Why is SPM even more important for multi processor systems





- Cache requires more BW than SPM
- Cache coherency → write through → more BW
- No direct L1-L1 communication →more BW
- Main memory becomes central bottleneck → Bad scalability
- Cache usage is implicit for the designer
  - Does not need programming
  - Hard to control → need for simulation/trial and error

MH: A compiler like tool exploiting scratchpad



Copy candidate graphs:
- O(1000) copy candidates
- Too much to manually select and map

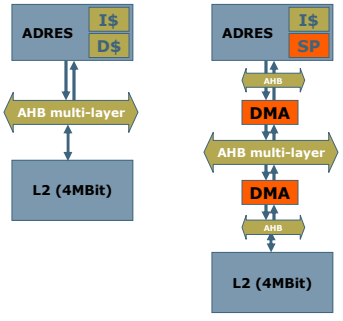
Life-time Analysis:
- SPM utilization (in-place mapping)
- Block transfer scheduling (pre-fetch)

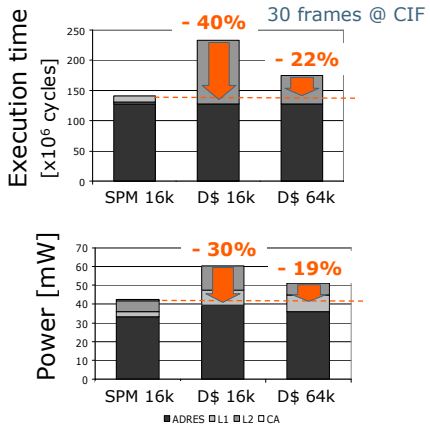
Selected Copy Candidates:
- Reuse buffers in SPM

MPEG-4 part 2 SP encoder results:
- 24 copies selected
- 42 block transfers introduced
- 80% of transfer latency hidden
  (i.e. in parallel with processing)

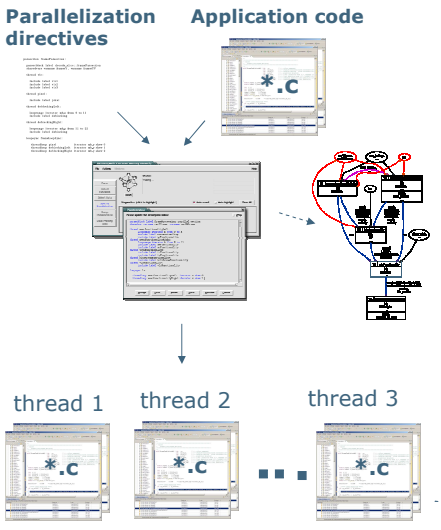
MH Results

- MPEG-4 p2 SP encoder (±8950 lines of C code)

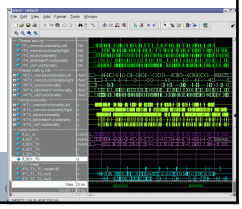




MPA
user assisted parallelization tool



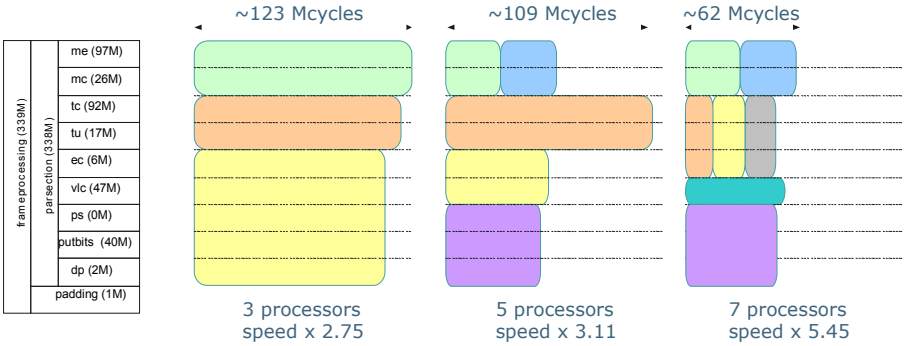
- Parallelizes sequential Clean-C source code
  - *Correct-by-construction multi-threaded code*
  - Higher level than openMP (=less work, exploration ease)
  - Directives in separate file
- Supported types of parallelism
  - Functional split
  - (Coarse) Data-level split
  - Combinations
- Dumps parallel code
- Dumps parallel code
- Sets up communication
  - Communication by means of FIFO's
  - FIFO sizes determined by tool



imec

MPA on MPEG-4 p2 SP encoder

- Prototype tool used to explore different parallel software architecture for MPEG-4 p2 SP encoder
  - 10 parallelization alternatives explored in half a day
  - 20 to 30 lines of parallelization directives



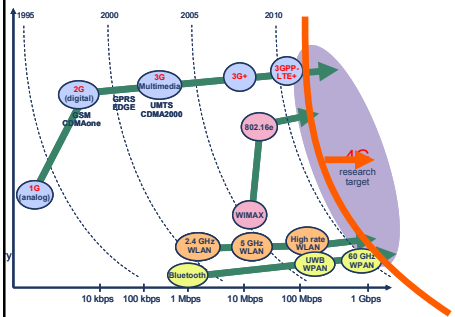
imec

Outline

- Programming multiprocessor platforms
  - Multiprocessor platforms are here today and they are here to stay!
  - Current mismatch between platform and programming model
  - How to map sequential code on a parallel platform?
- IMEC's MPSoC tool suite
  - Eclipse *Code Cleaning* plug-in to get your C code in parallel shape
  - MH tool: **M**emory **H**ierarchy optimization tool
  - MPA tool: a **M**ultiprocessor **P**arallelization **A**ssistant
- A concrete case: towards Gbit/s Cognitive Reconfigurable Radio
- Run time management

The need for reconfigurable radio platforms: functional and techno-economical constraints

Functional constraints



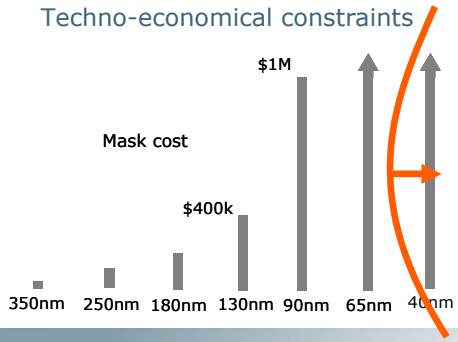
">6 radios/terminal in 2009"

Technology roadmaps (2014 horizon):
Intel, Nokia, TI, Infineon, NXP, ST, Motorola

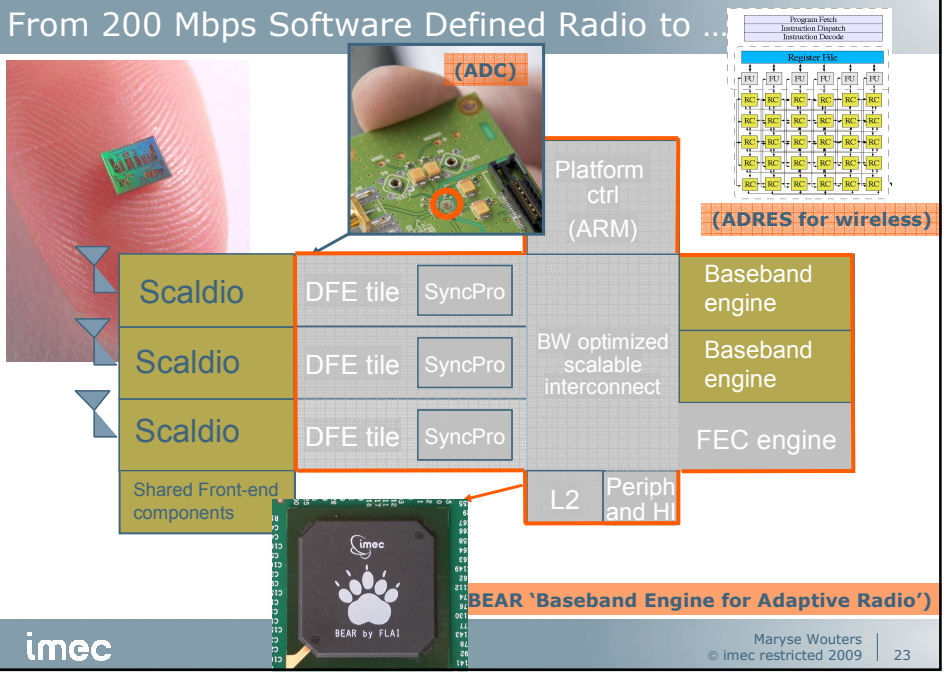
Advantage 1: lower cost

Advantage 2: higher flexibility

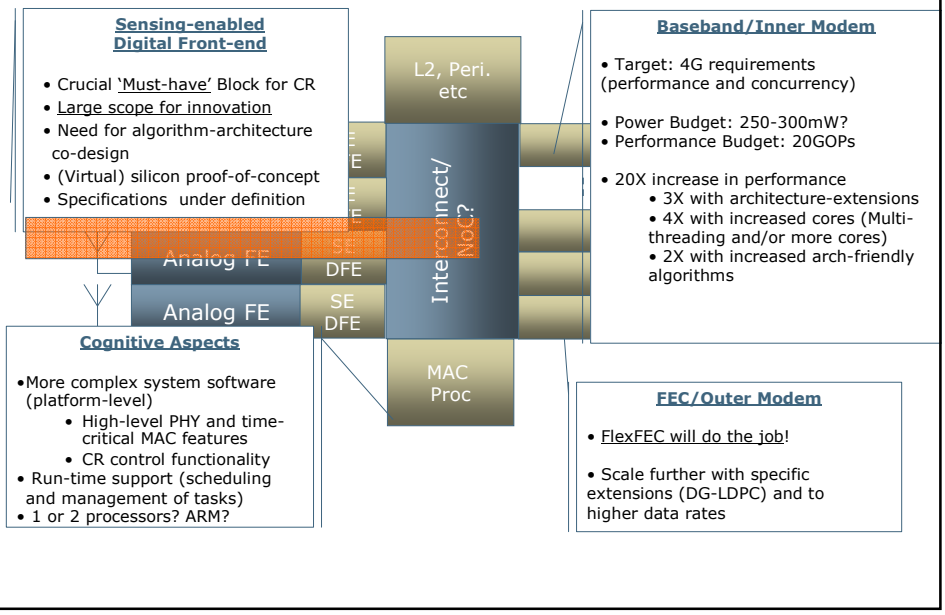
Techno-economical constraints



IMEC's flexible baseband platforms: From 200 Mbps Software Defined Radio to …



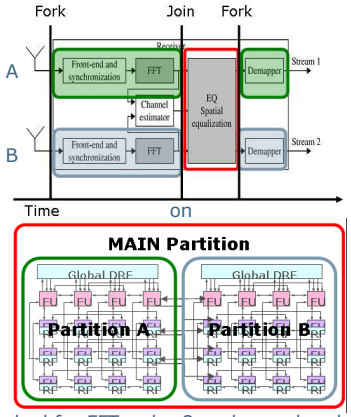
… Gbps cognitive reconfigurable radio



Two ways to higher throughput (Gbps): Exploring multi-threading and multi-processing with MPA

Goal: balanced load and minimal communication overhead

Multi-threading | Split per antenna



Checked for FFT-only: 2 cycle overhead on 694 cycles; IPC=24.2 out of 32

Needed: fast switching between threads

Multi-processing | Split per symbol

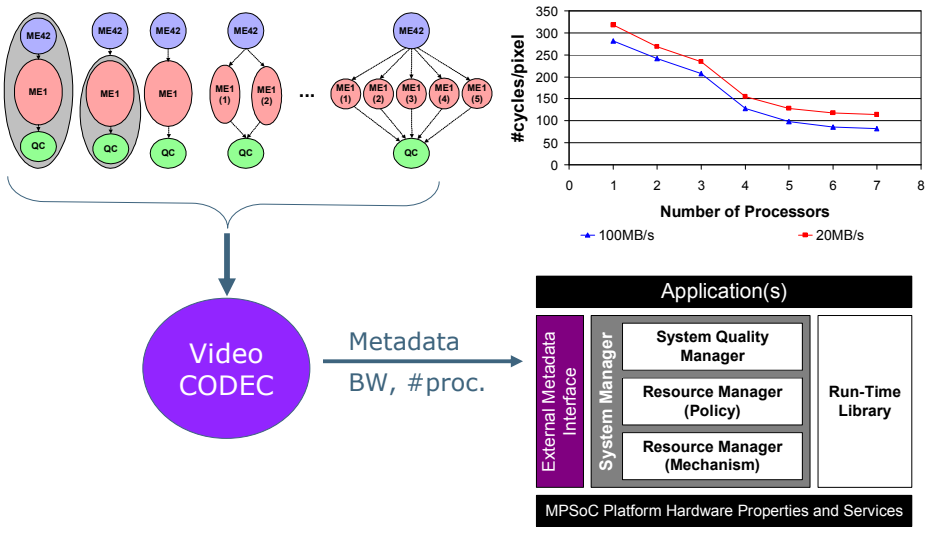
Confidential Until Published (submitted to ISSCC 2009)

Needed: low communication between symbols due to lower inter-ADRES communication bandwidth than intra-ADRES

Outline

- Programming multiprocessor platforms
  - Multiprocessor platforms are here today and they are here to stay!
  - Current mismatch between platform and programming model
  - How to map sequential code on a parallel platform?
- IMEC's MPSoC tool suite
  - Eclipse *Code Cleaning* plug-in to get your C code in parallel shape
  - MH tool: **M**emory **H**ierarchy optimization tool
  - MPA tool: a **M**ultiprocessor **P**arallelization **A**ssistant
- A concrete case: towards Gbit/s Cognitive Reconfigurable Radio
- Run time management

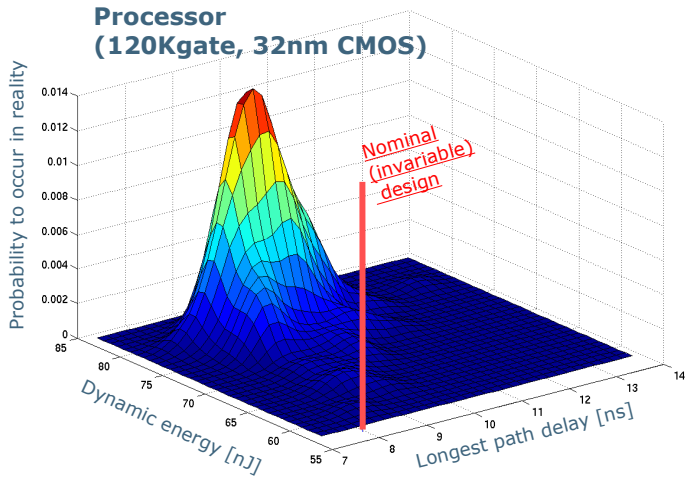
Run-time manager integrated in the flow



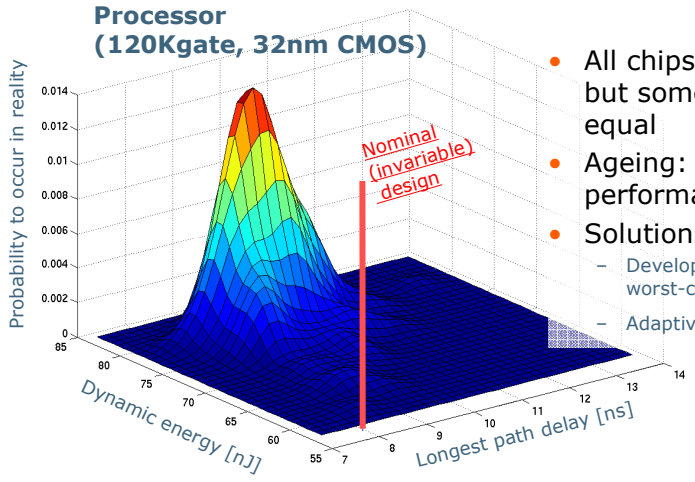
A glimpse of the future

- What the future brings
  - More and more run-time solutions required
    - Multi-core evolves to many-core
    - Applications and usage patterns grow increasingly dynamic
    - Advanced CMOS processing technology becomes a source of un-predictability: run-time mitigation of variability/reliability issues
  - Advances in integration technologies
    - 3D-stacking of devices with Through-Silicon-Via
    - Will radically change the way we build memory hierarchies and hence multi-core architectures

Variability in process technology



Variability in process technology



- All chips are equal – but some will be more equal
- Ageing: degradation of performance over time
- Solutions:
  - Develop software for the worst-case?
  - Adaptive system?

Conclusion

- IMEC's MPSoC tool suite automates the mapping of single-threaded applications on multi-processor platforms
  - Eclipse *Code Cleaning* plug-in to get your C code in parallel shape
  - MH tool: **M**emory **H**ierarchy optimization tool
  - MPA tool: a **M**ultiprocessor **P**arallelization **A**ssistant
- The tool suite is used by IMEC and its partners for multi-processor platforms targeting gigabit/s wireless communication.
- The MPSoC technology is currently being transferred to two semiconductor companies, and is available for transfer to others.