Embedded Software Development for MPSoC: Preliminary HOPES Experience

June 24, 2008

Soonhoi Ha
Seoul National University

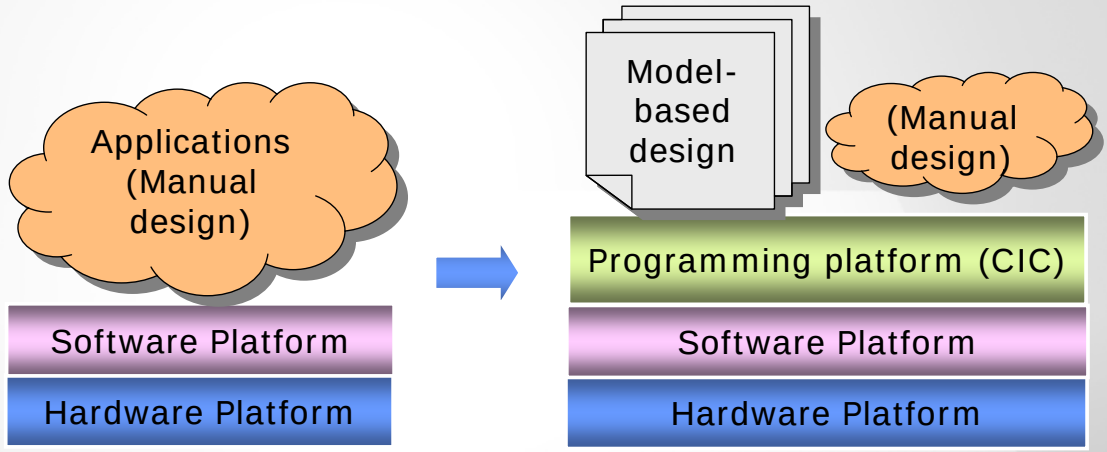
Contents

Problem Statement

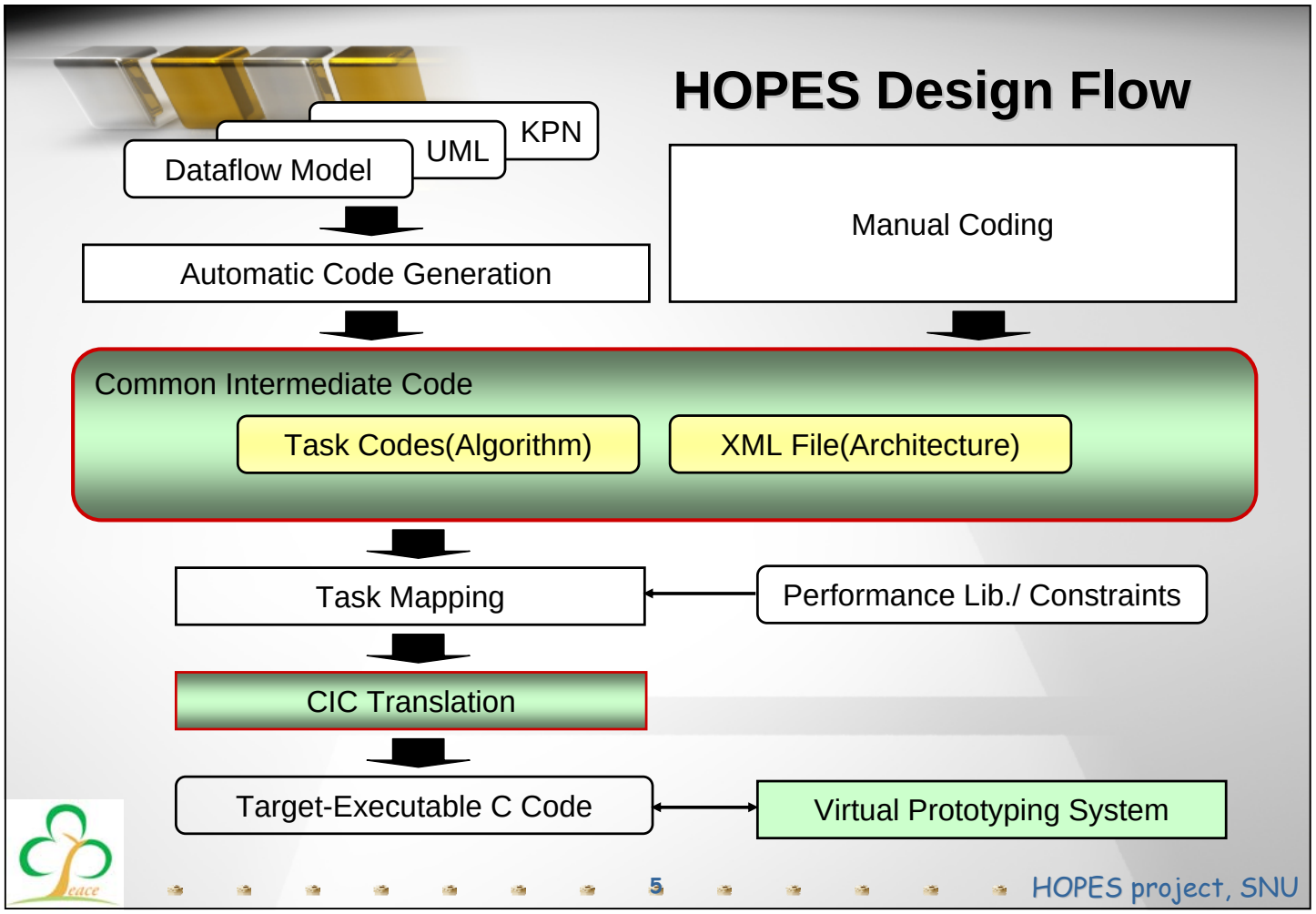
- Target: MPSoC (Embedded Systems) with high degree of parallelism
  - Scalability
  - Heterogeneous processors with diverse communication architecture
  - Power-constrained system
- Problem: parallel programming for MP Systems
  - Parallelism extraction (multiple use case, multi-tasking apps.)
    - Functional parallelism, data-parallelism, temporal-parallelism
  - Partitioning and mapping
  - Parallel code generation: parallel programming is not easy
  - Performance estimation and verification
  - Design space exploration

➡ Need of sound (scalable and robust) methodology

Basic Idea



HOPES Design Flow



Design Steps and Techniques

- Algorithm development / Behavior specification
  - (Option 1) PeaCE modeling: FSM + Dataflow tasks
    Automatic clustering into CIC tasks
  - (Option 2) CIC task specification
- Functional Simulation
  - Multithreaded code generation in the host machine
- Mapping to the processing elements
  - Automatic mapping considering functional/data/temporal parallelism.
  - Manual mapping
- (Architectural design space exploration)
- CIC translation to generate target C codes
- Virtual prototyping / system verification

Contents

CIC (Common Intermediate Code)

- .CIC

CIC (tasks)

Interfaces:
- *_init()
- *_go()
- *_wrapup()

Depends on:
- Generic APIs
  - (including comm. APIs)

Additional information:
- OpenMP pragmas
- Hardware pragmas

CIC (architecture) - .xml files

Hardware information:
- Processor information
- H/W accelerator info.
- OS, Memory maps

Constraints:
- Timing requirement
- Memory constraint
- Power consumption

Structure:
- Task structures
- Channel & Task mapping

CIC Code Generation

- Separate CIC task codes for partitioned tasks
  - Explicit functional parallelism → task_name.cic
- openMP programming for data parallelism
- Generic API for platform independent programming

```
void h263decoder_go (void) {
        ...
    l = MQ_RECEIVE("mq0", (char *)(ld_106->rdbfr), 2048);
    ...
    # pragma omp parallel for
    for(i=0; i<99; i++) {
        //thread_main()
            ....
    }
     // display the decode frame
       dither(frame);
}
```

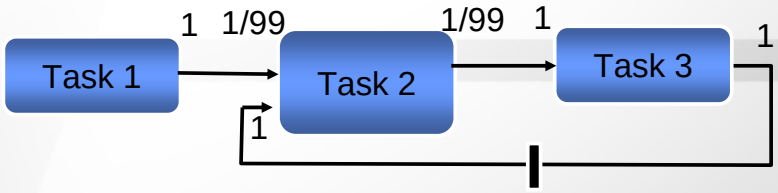

DivX player CIC xml (Structure)

```
<structure>
 <mode name="default">
  <task name="AviReaderI0">
   <subtask name="arm926ej-s0">
   <procMap>0</procMap>
   <fileName>AviReaderI0_arm926ej_s0.cic</fileName>
   </subtask>
  </task>
  <task name="H263FRDivxI3">
   <subtask name="arm926ej-s0">
   <procMap>0</procMap>
   <fileName>H263FRDivxI3_arm926ej_s0.cic</fileName>
   </subtask>
  </task>
  <task name="MADStreamI5">
   <subtask name="arm926ej-s0">
   <procMap>0</procMap>
   <fileName>MADStreamI5_arm926ej_s0.cic</fileName>
   </subtask>
  </task>
 </mode>
```

```
 <queue>
  <name>mq0</name>
  <src>AviReaderI0</src>
  <dst>H263FRDivxI3</dst>
  <size>30000</size>
 </queue>
 <queue>
  <name>mq1</name>
  <src>AviReaderI0</src>
  <dst>MADStreamI5</dst>
  <size>30000</size>
 </queue>
</structure>
</CIC_XML>
```

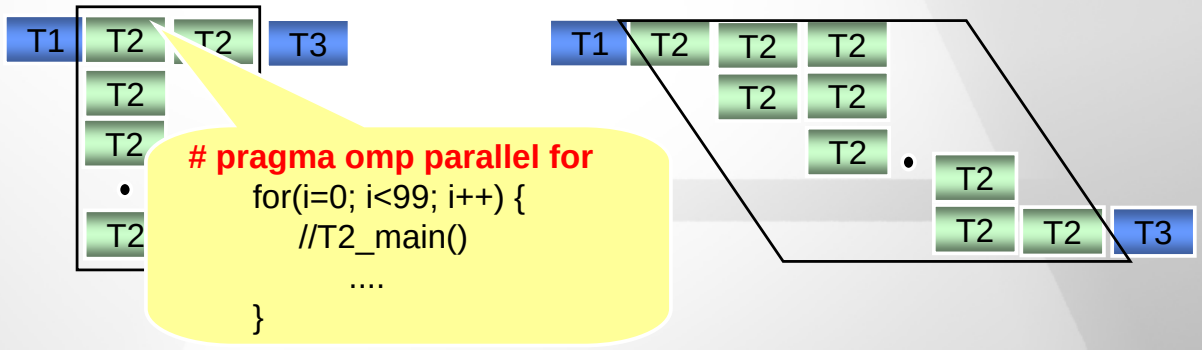
Task Model in CIC

- Task execution semantics
  - Process networks: blocking read, non-blocking write if queue space is available
  - Possibility of data parallel execution is specified
    - If there is no internal state inside a task, it can be executed in parallel
- Two types of channels
  - FIFO channel of finite size FIFO queue (default)
    - *MQSend(data), MQRead(data)*
  - Array channel of finite size array
    - *ACSend(index,data), ACRead(index, data)*



Data Parallel Execution

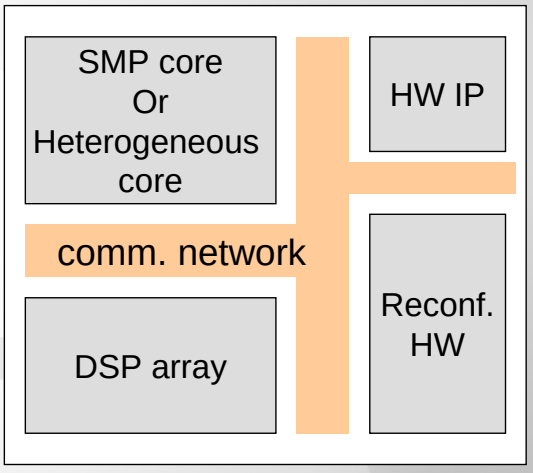
- Default parallelism
  - (example) Macroblock decoding of H263 decoder
  - Specification method: openMP pragma
- Wavefront parallelism
  - (example) Macroblock analysis of H264 encoder
  - Specification method: CIC definition of wavefront vector



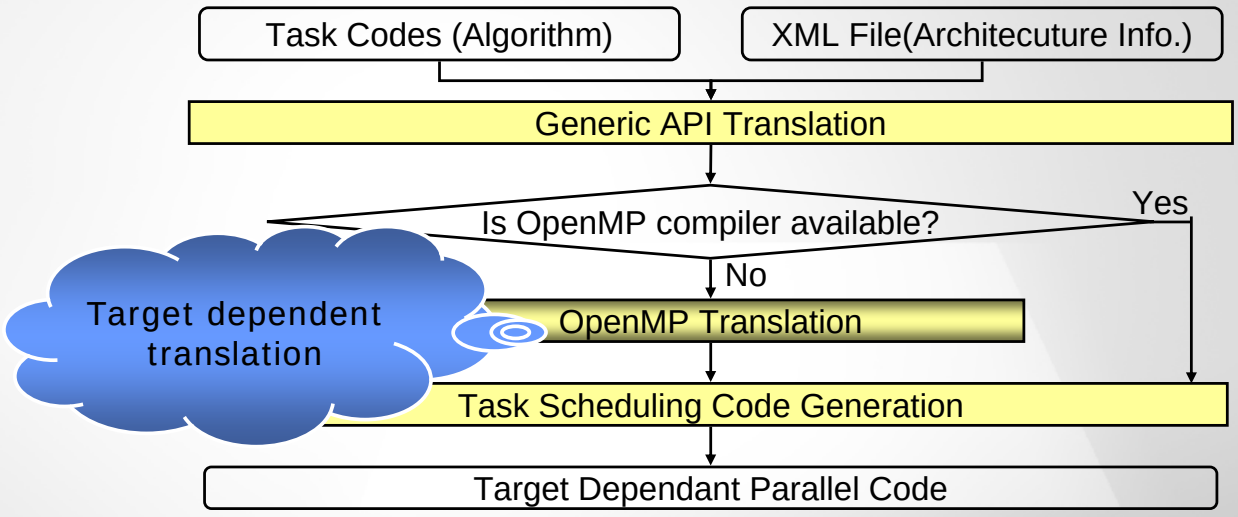
CIC Translation

- CIC to Multi-thread codes for functional simulation
  - Generated codes are run on a host machine
- CIC to target C codes
  - For virtual prototyping
  - For MPCore
  - For Cell processor

  [planned]
  - DSP array
  - Reconf. hardware



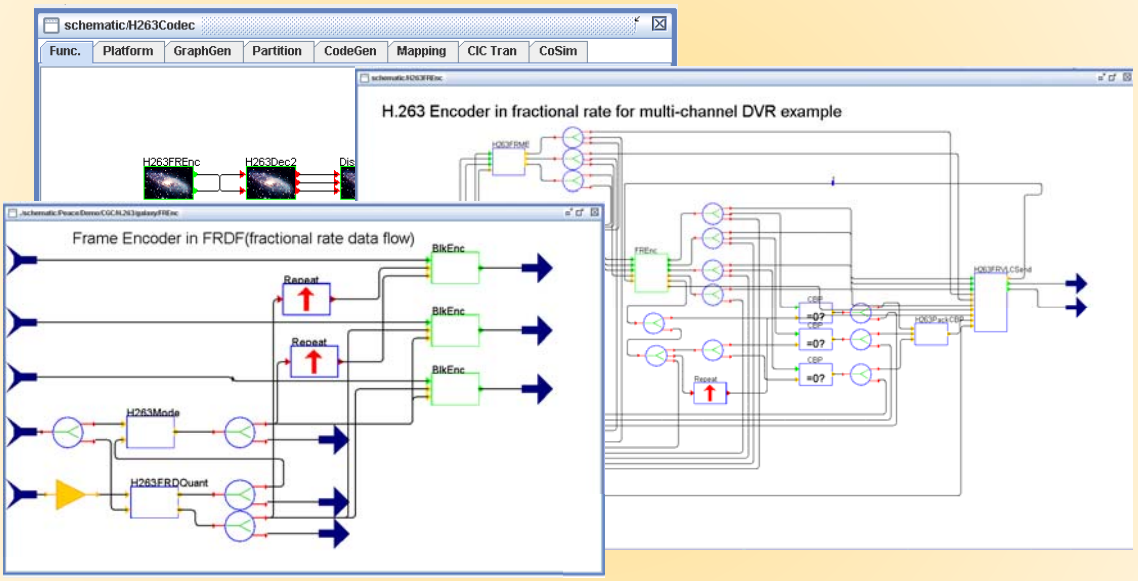
CIC Translator



Contents

Model-based Design: H.263 CODEC

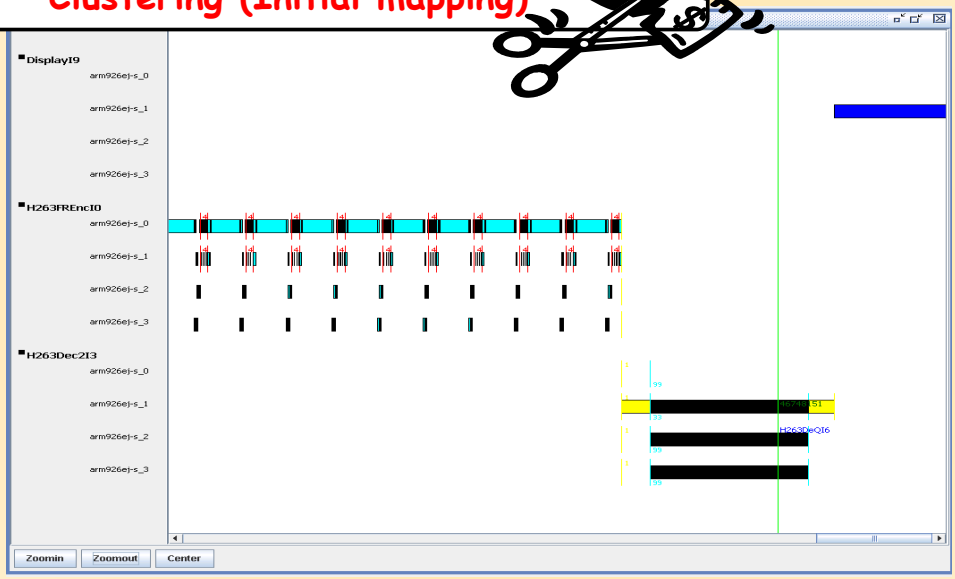
Programming



H263 Encoder/decoder in extended SDF (FRDF) specification

Application Clustering

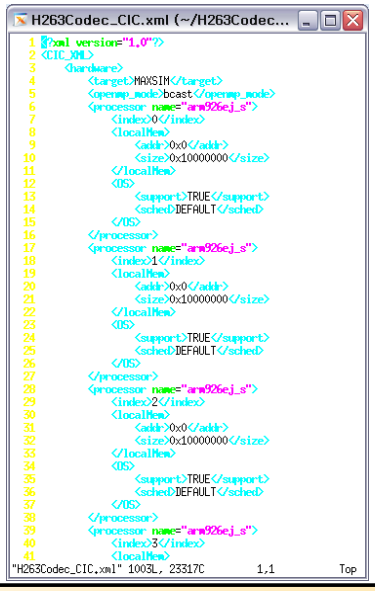
Clustering (Initial mapping)



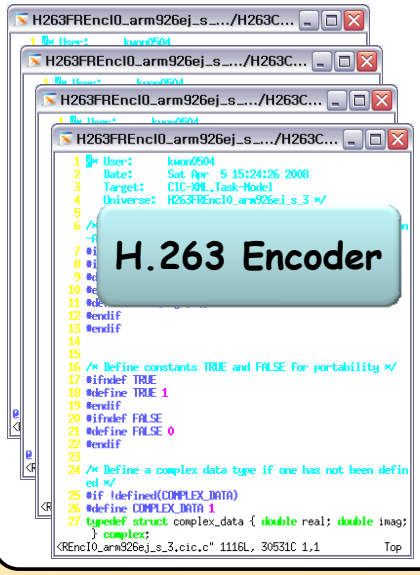
Clustering SDF nodes into CIC tasks to keep functional parallelism as much as possible. (Note data parallel execution of a task in H.263 decoder)

Model to CIC Generation

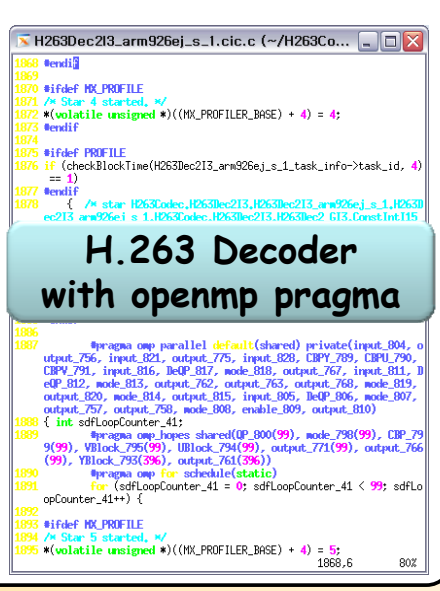
Arch. XML



CIC Task Files



H.263 Encoder

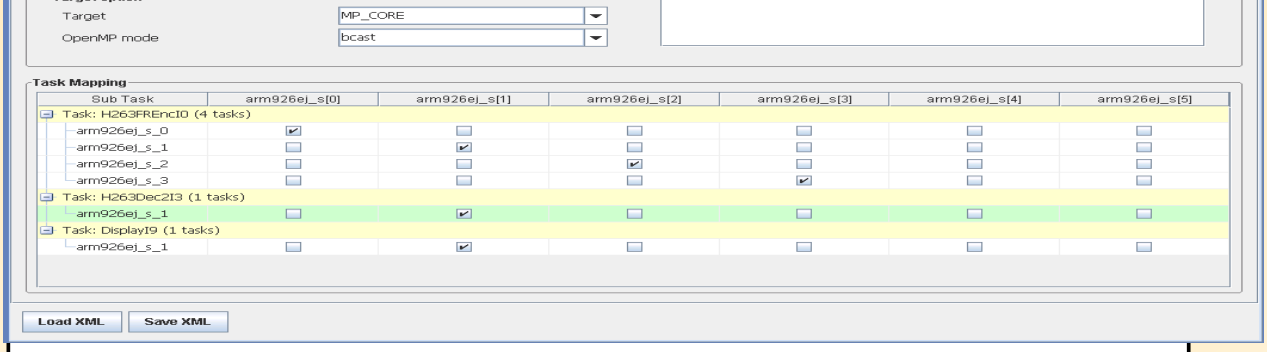


H.263 Decoder with openmp pragma

Generate CIC files from the clustered SDF graph automatically.

CIC Mapping (Manual Mapping)

(Manual) Mapping

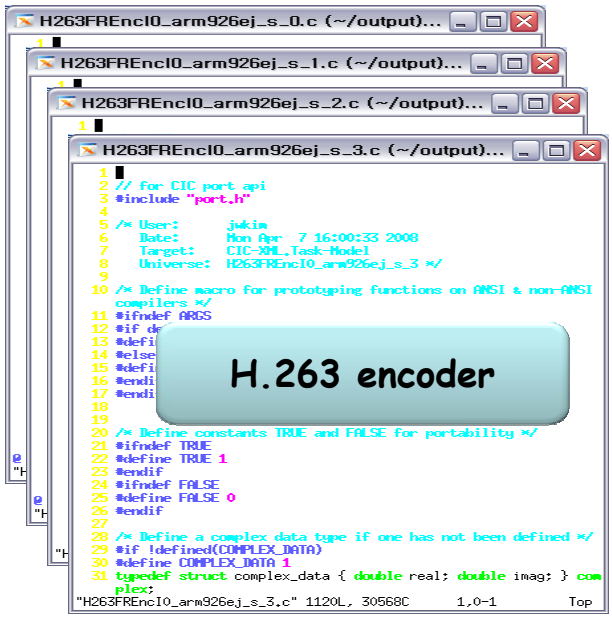


Mapping CIC tasks into the target processing elements

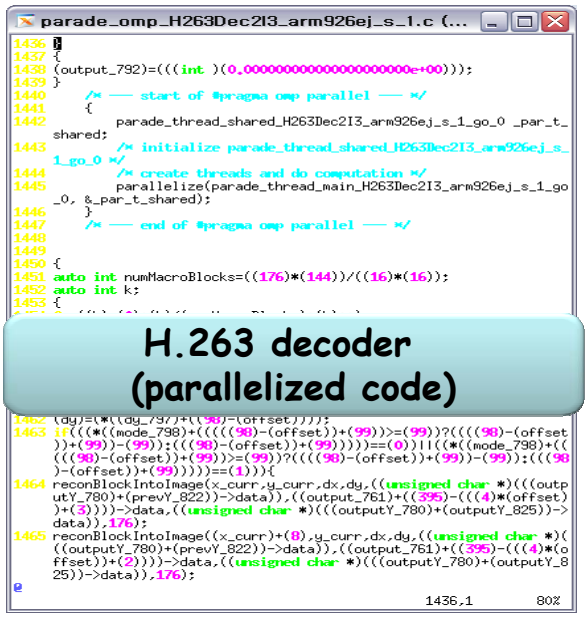
CIC Translation

CIC Translation

Target C codes



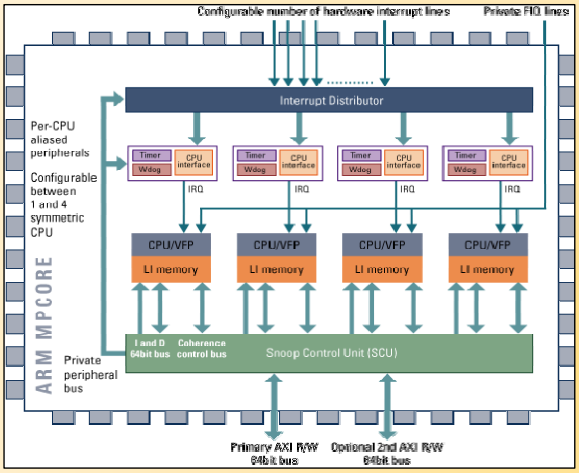
H.263 encoder



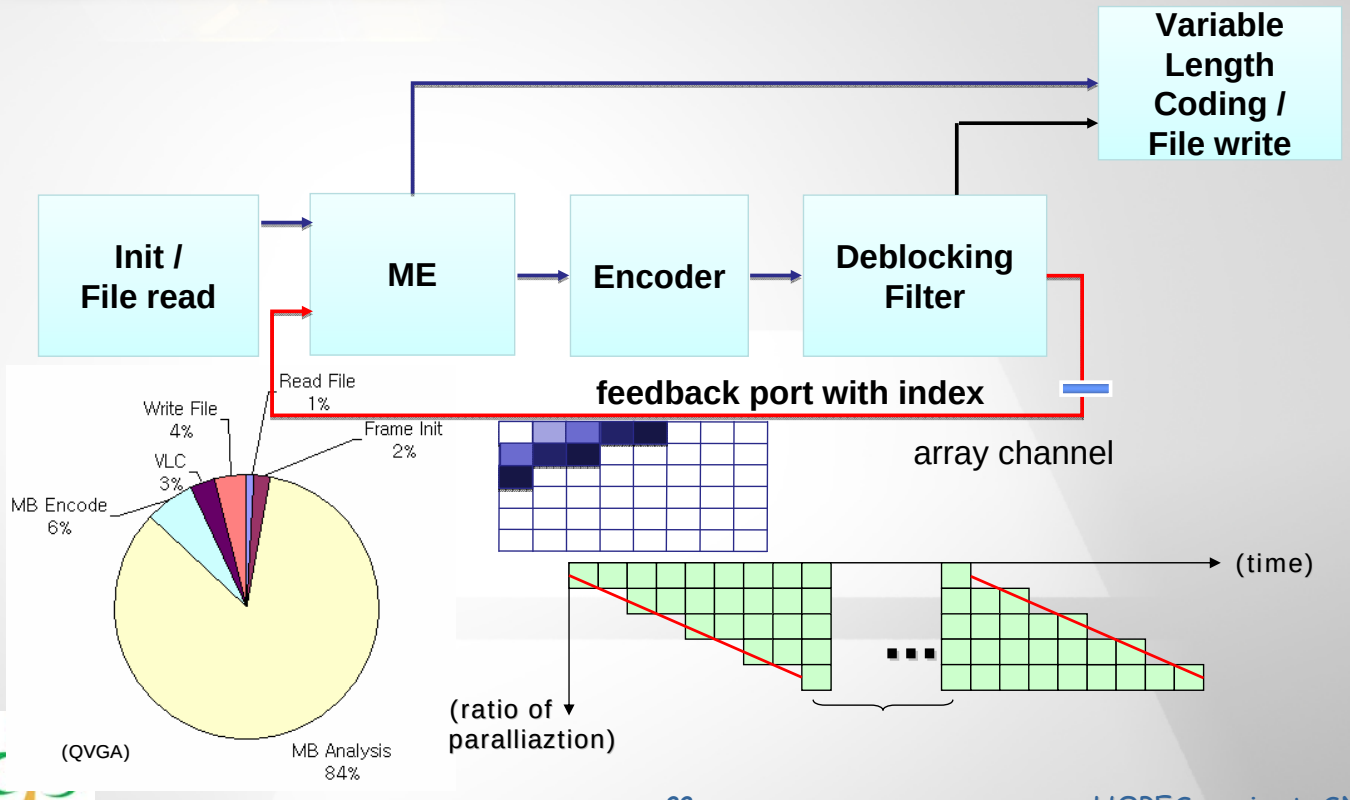
H.263 decoder (parallelized code)

Prototyping

Prototyping

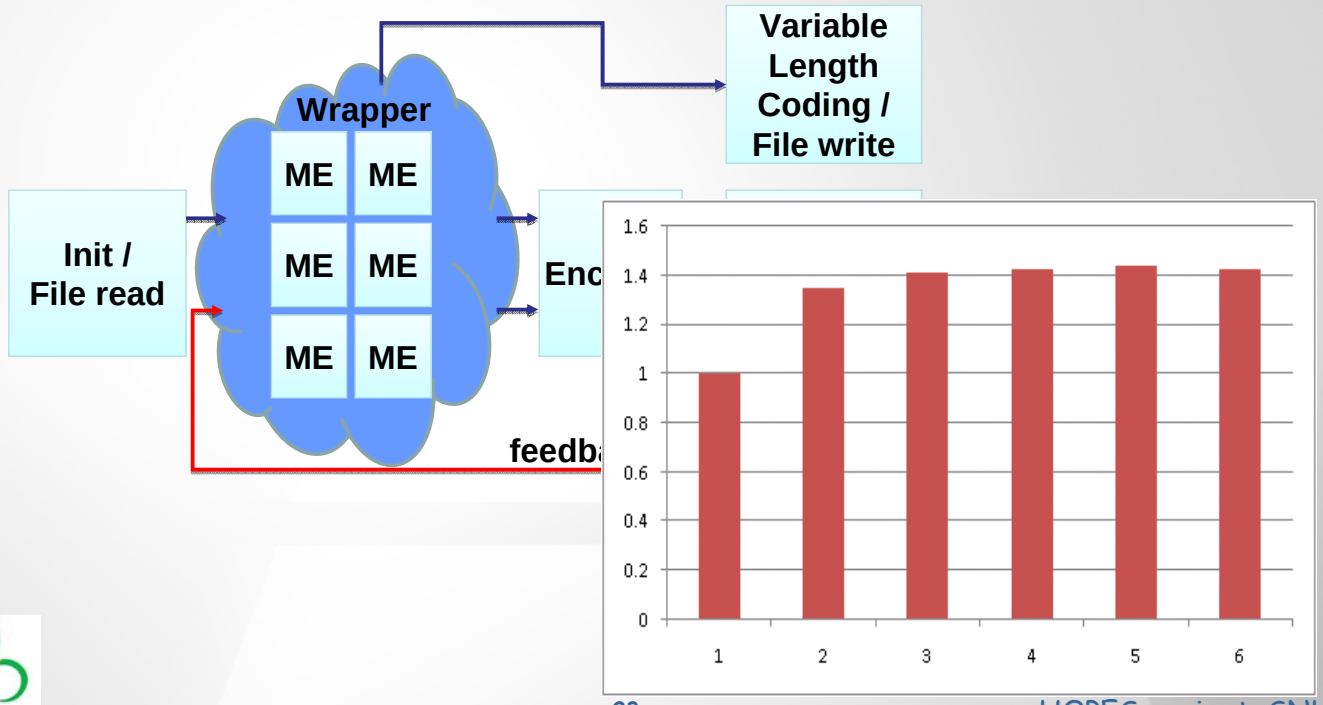


Manual Design: X264 Encoder on CELL



X264 Encoder on CELL(2)

- CIC translator synthesizes wrapper automatically



Contents

Status

- Currently two ways of CIC generation are supported
  - PeaCE models to CIC: FSM model to CIC is under development
  - Manual specification of CIC tasks
  - (Simulink Model to CIC, UML to CIC might be interesting)
- CIC model refinement
  - X264 specification uncovers need of extensions.
  - More examples will do the same
- Automatic mapping
  - Currently considers functional and data parallelism only
  - Will consider pipelining too
- Virtual prototyping
  - Will be developed as a TLM simulator that models the communication architecture accurately.
  - Debugging and monitoring features will be added
  - Fast simulator based on virtual synchronization technique is under development

Conclusion

- HOPES is a newly launched project to make a embedded software development environment for MPSoCs
  - Support of diverse models
  - Target independent environment + target specific libraries
  - Integration of software modules at various stages
  - http://peace.snu.ac.kr/hopes