TECHNISCHE
UNIVERSITÄT
DRESDEN

Vodafone Chair Mobile Communications Systems, Prof. Dr.-Ing. G. Fettweis

vodafone chair

Guided Design Space Exploration of Heterogeneous MPSoCs

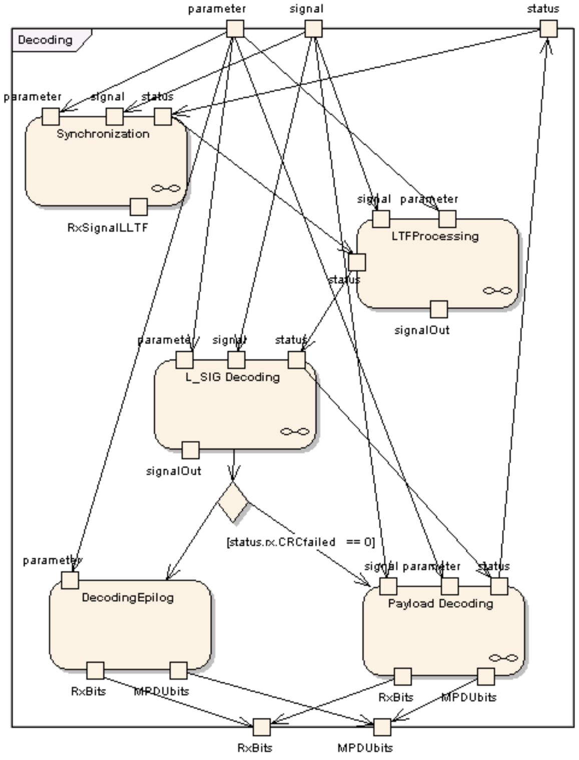
Bastian Ristau

1st Workshop on Mapping Applications to MPSoCs, Schloss Rheinfels
June 17, 2008

The Design Space Exploration Problem: Finding the Right System and Mapping

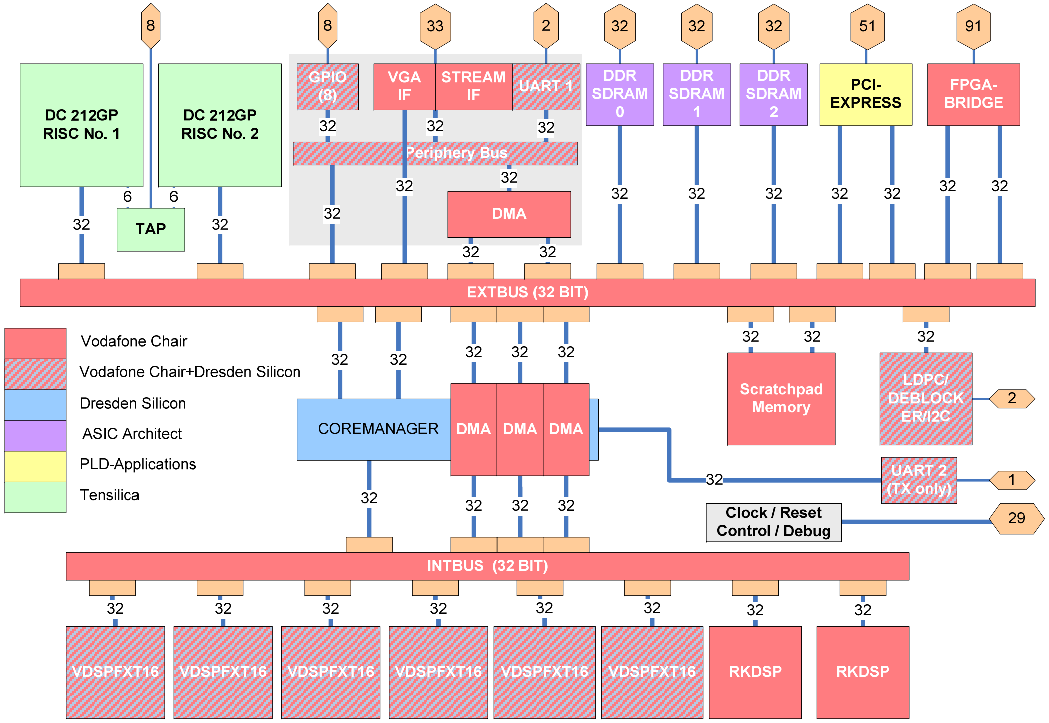
Application

(e.g. 802.11a)



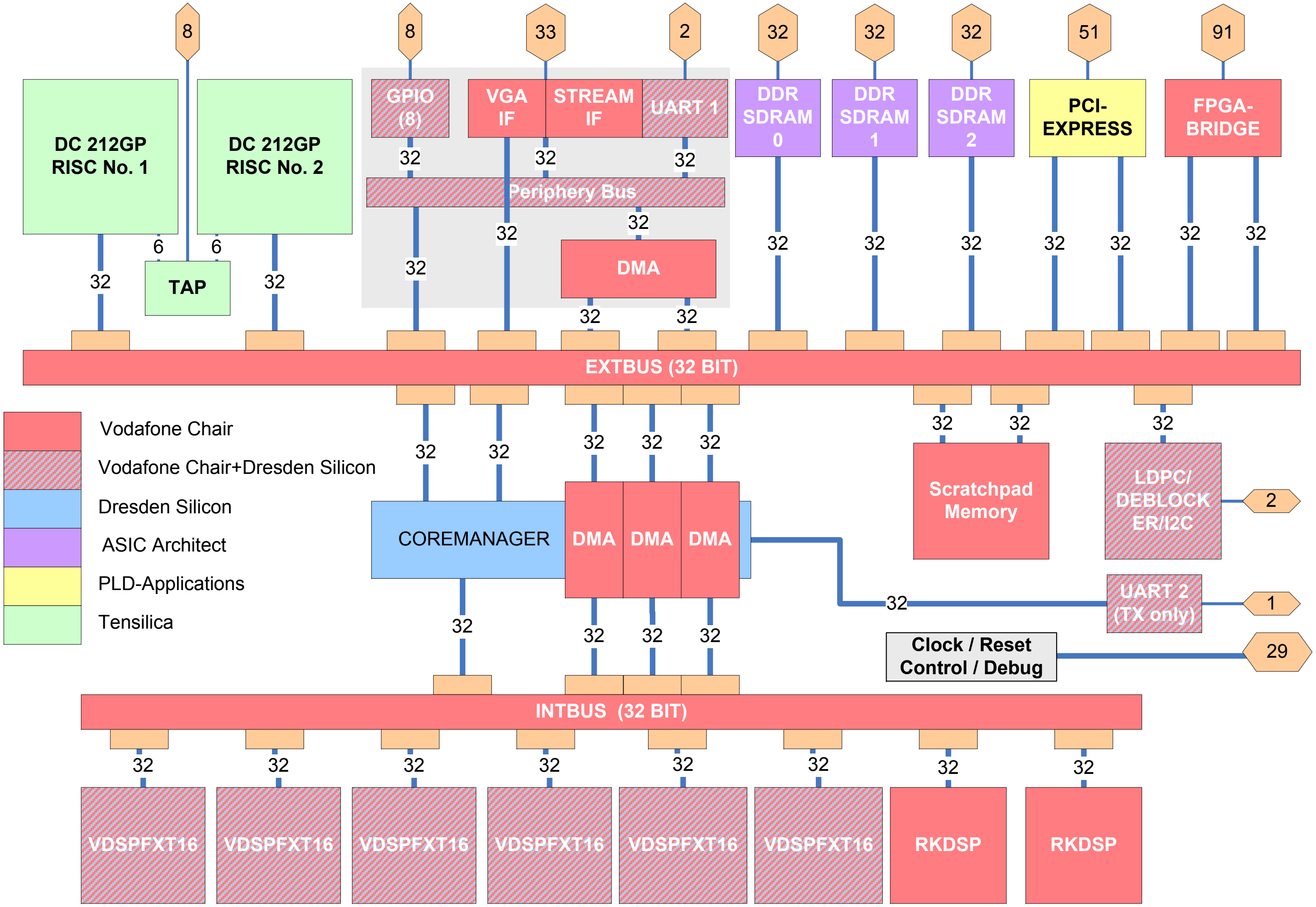
MPSoC

(e.g. Tomahawk)

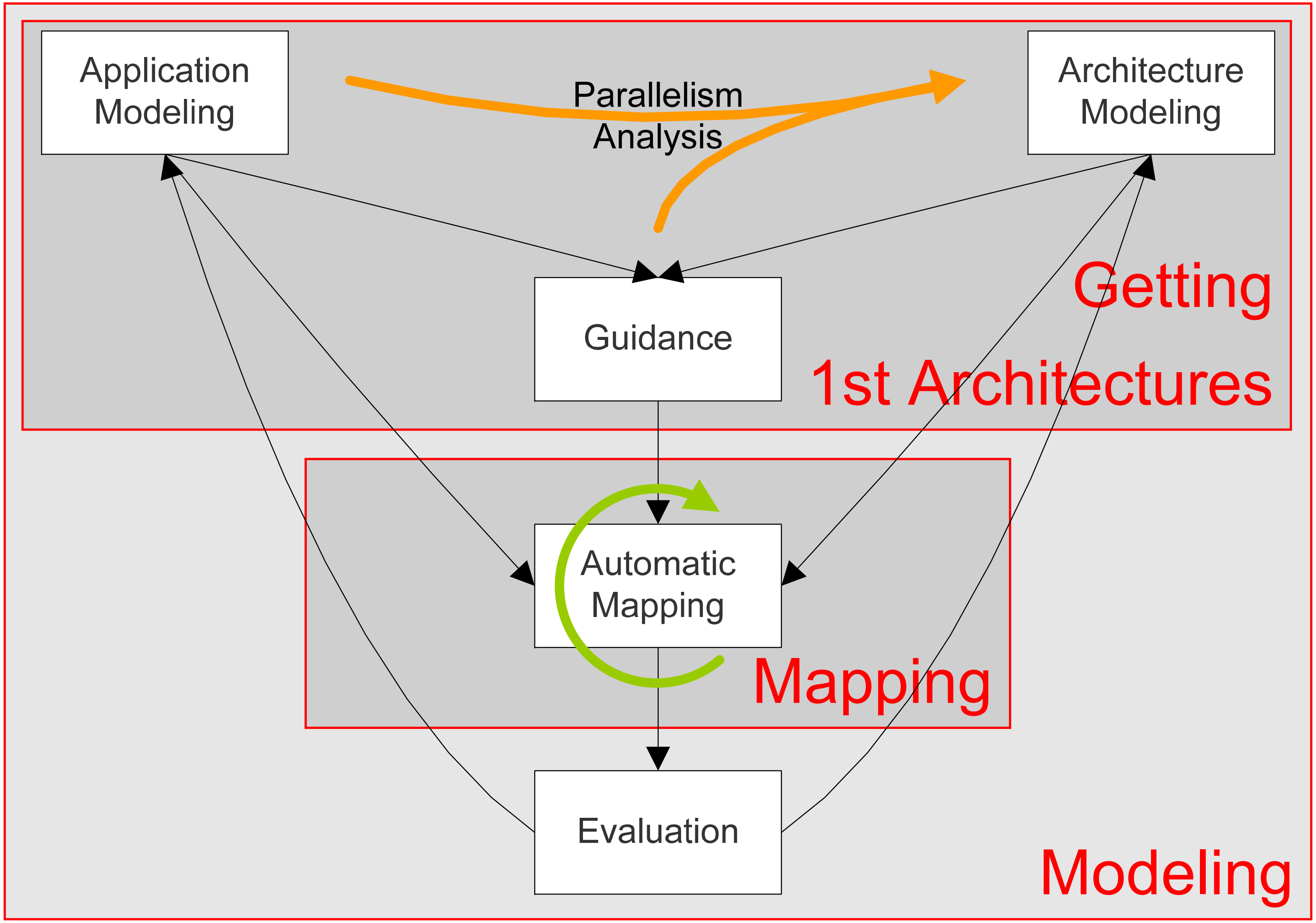


Objectives: Power, Performance, Flexibility, Area, …

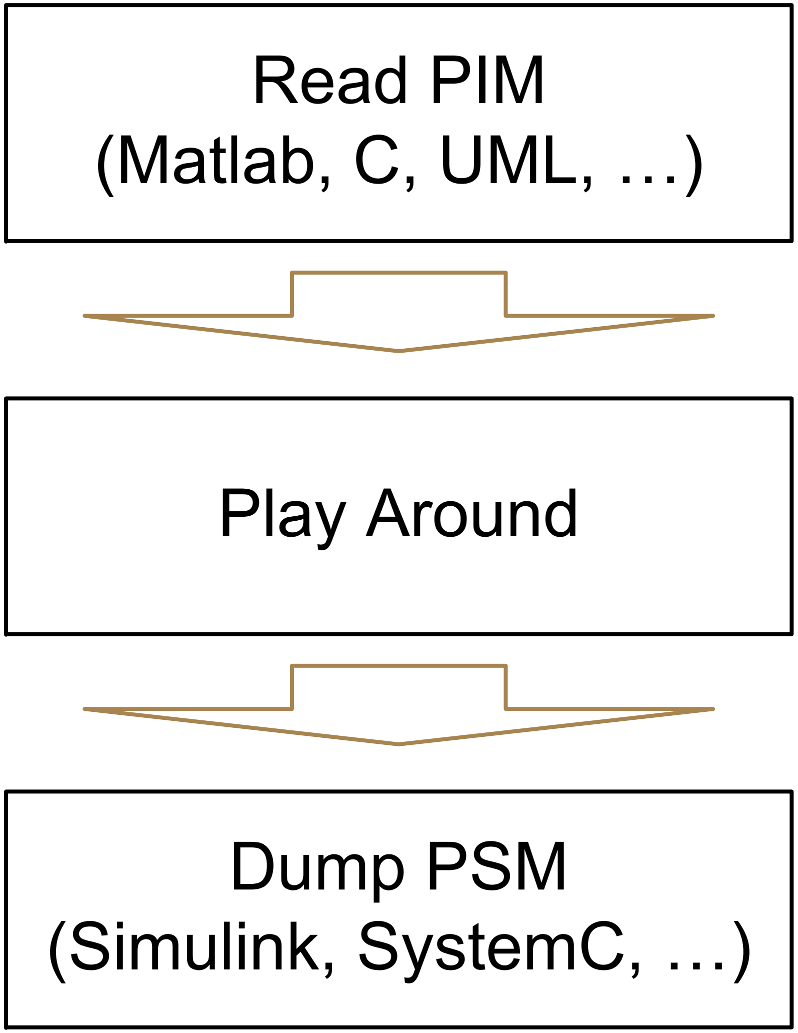
Platform Concept



Design Space Exploration Approach



Kind of Big Picture



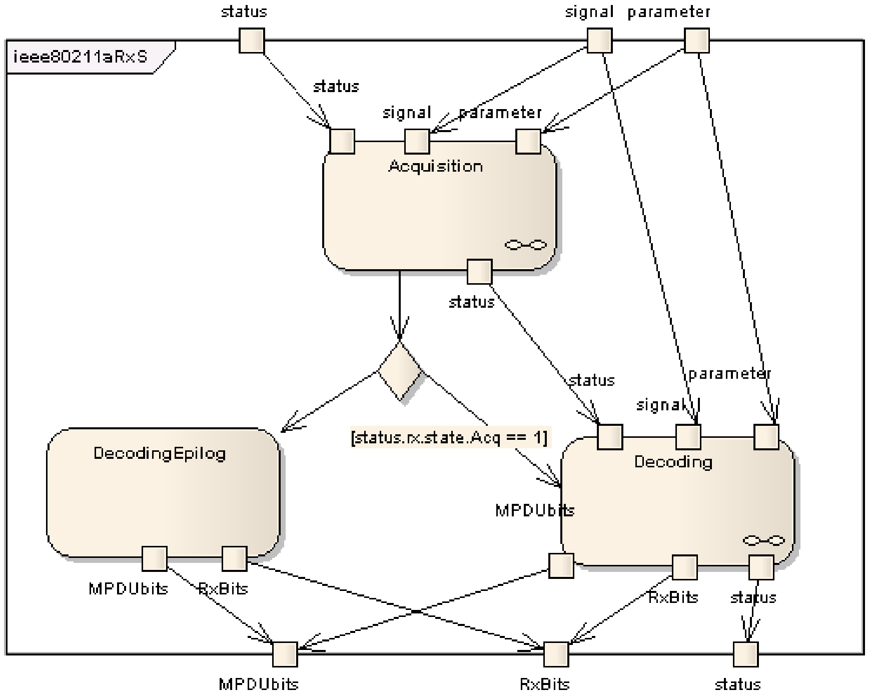
Read PIM
(Matlab, C, UML, …)

Play Around

Dump PSM
(Simulink, SystemC, …)

Modeling

Concept: Nodes communicating via ports over edges



- Nodes = Tasks | Processors
- Ports = Input and output data | Ports
- Edges = Ctrl and data flow | Interconnect

Application Modeling

- Task = Fetch → execute → write back (no side effects)
- Loops & option blocks wrapped into single node

→ Multiple abstraction levels (nodes can contain nodes)

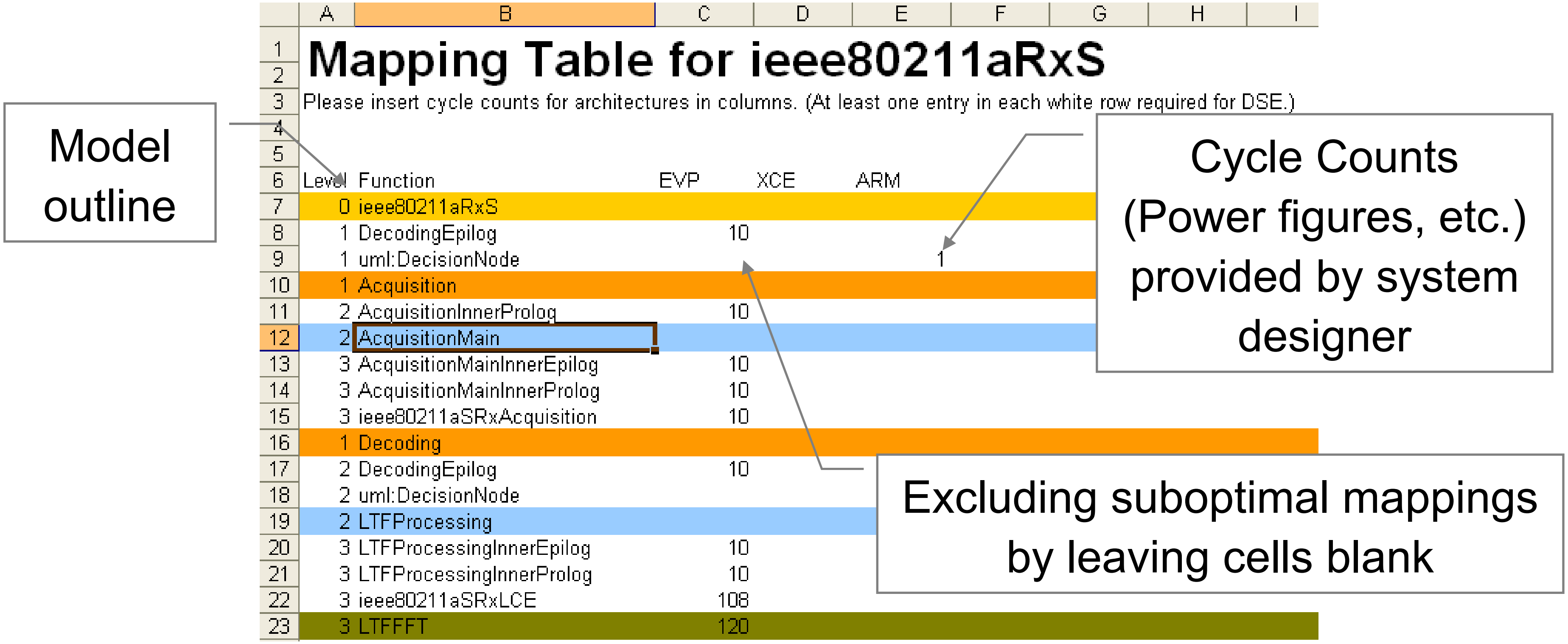
→ Graphs almost acyclic and in SSA form

- Limitations: Streaming

→ Moving to … Data Flow Graphs

Providing Mapping Options

Excel tables generated from application model



Mapping Table for ieee80211aRxS

Please insert cycle counts for architectures in columns. (At least one entry in each white row required for DSE.)

Level	Function	EVP	XCE	ARM
0	ieee80211aRxS			
1	DecodingEpilog	10		
1	uml:DecisionNode			1
1	Acquisition			
2	AcquisitionInnerProlog	10		
2	AcquisitionMain			
3	AcquisitionMainInnerEpilog	10		
3	AcquisitionMainInnerProlog	10		
3	ieee80211aSRxAcquisition	10		
1	Decoding			
2	DecodingEpilog	10		
2	uml:DecisionNode			
2	LTFProcessing			
3	LTFProcessingInnerEpilog	10		
3	LTFProcessingInnerProlog	10		
3	ieee80211aSRxLCE	108		
3	LTFFFT	120		

Model outline

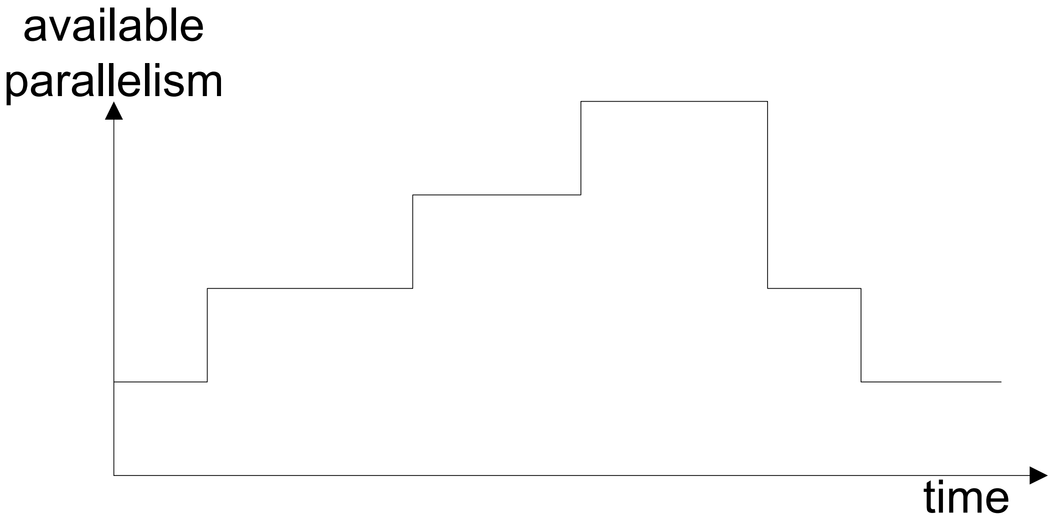
Cycle Counts (Power figures, etc.) provided by system designer

Excluding suboptimal mappings by leaving cells blank

➔ Utilizing Knowledge of System Designer

Parallelism Analysis Basics

Parallelism profiles reveal distribution of parallelism.



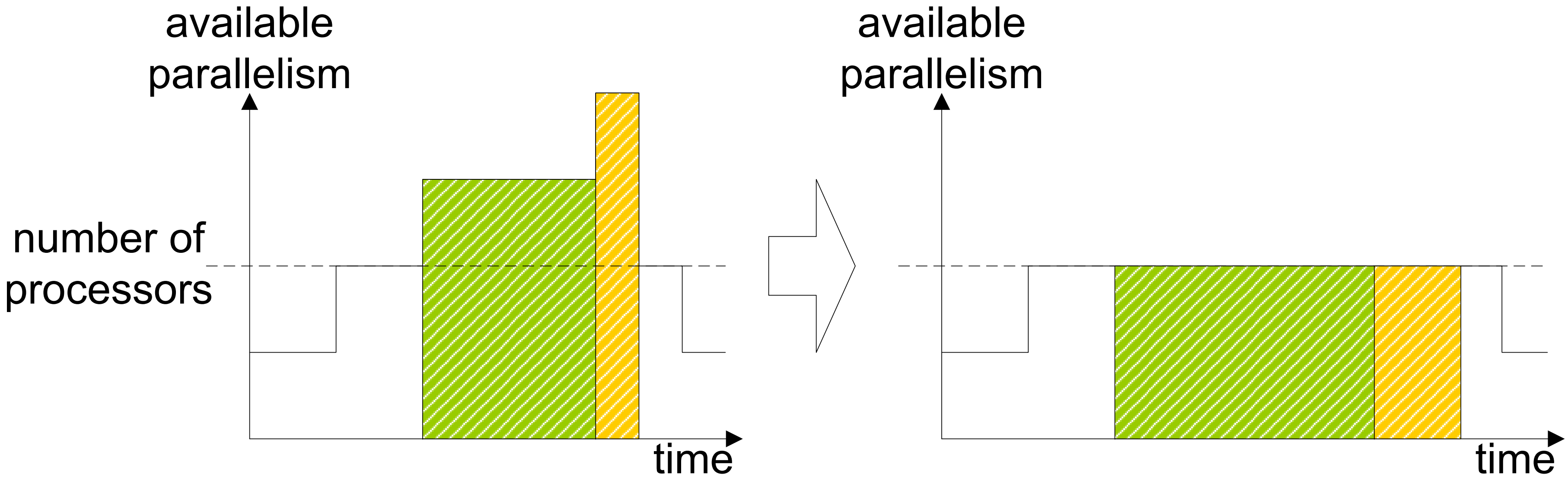
parallelism profile

Useful for determining 1st architecture, if

- Profile is independent of concrete architecture
- Profile is independent of underlying schedule

Getting 1st Architectures

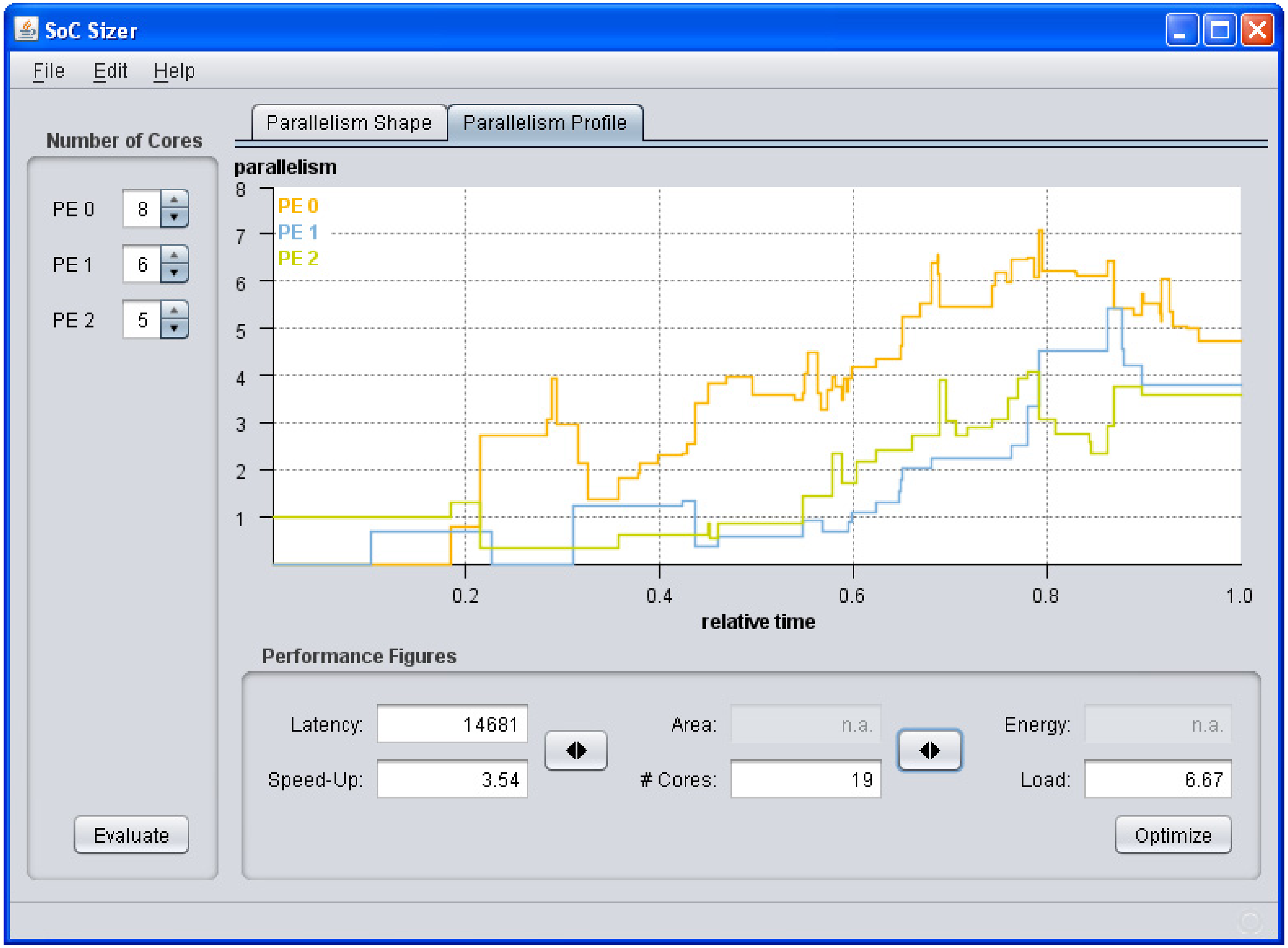
Using parallelism profiles for performance estimation



profile for unlimited processor case

profile for a given architecture

Parallelism Analysis Example



Number of Cores

PE 0: 8

PE 1: 6

PE 2: 5

Performance Figures

Latency: 14681

Speed-Up: 3.54

Area: n.a.

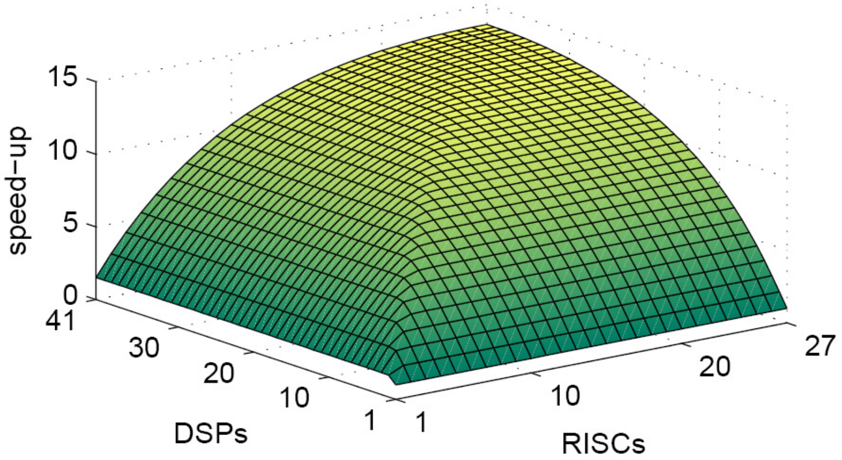
Cores: 19

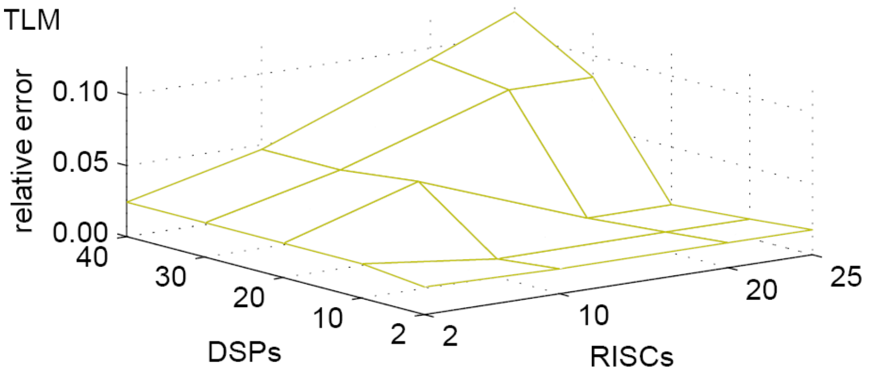
Energy: n.a.

Load: 6.67

Parallelism Analysis Results

(Exemplary results for a task graph w/ 1000 tasks and two processor types)

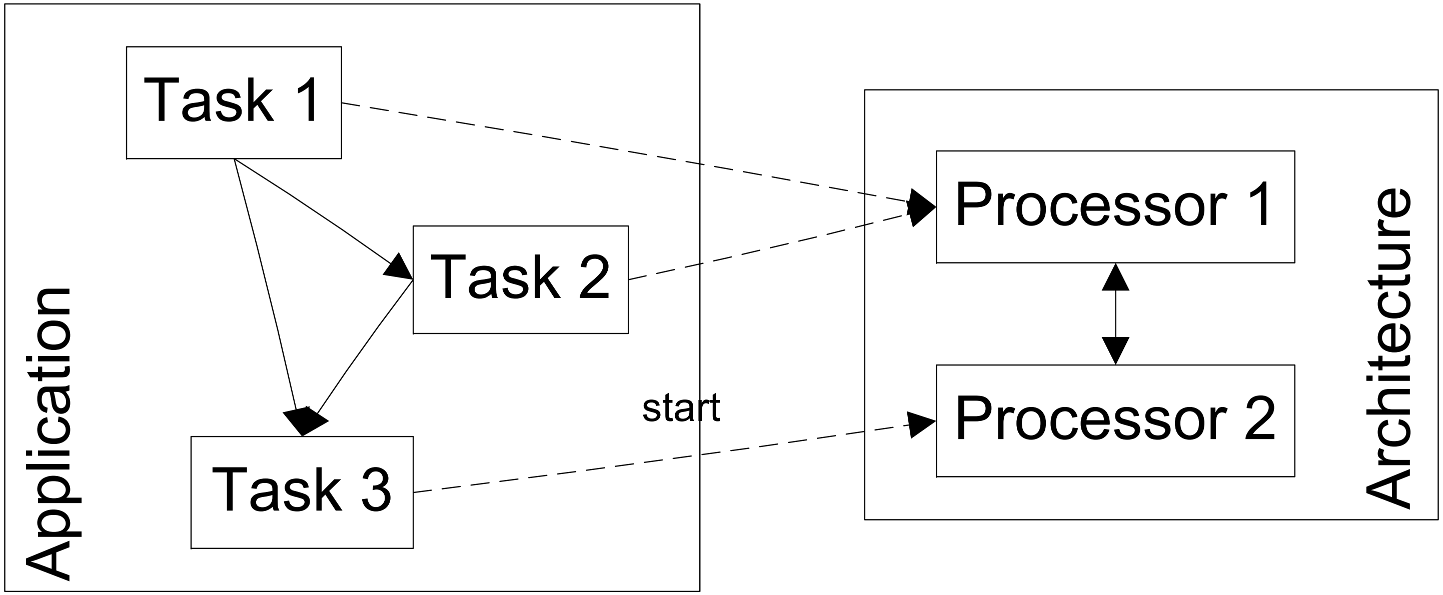




relative speed-up curve

estimation error compared to TLM simulation

Mapping

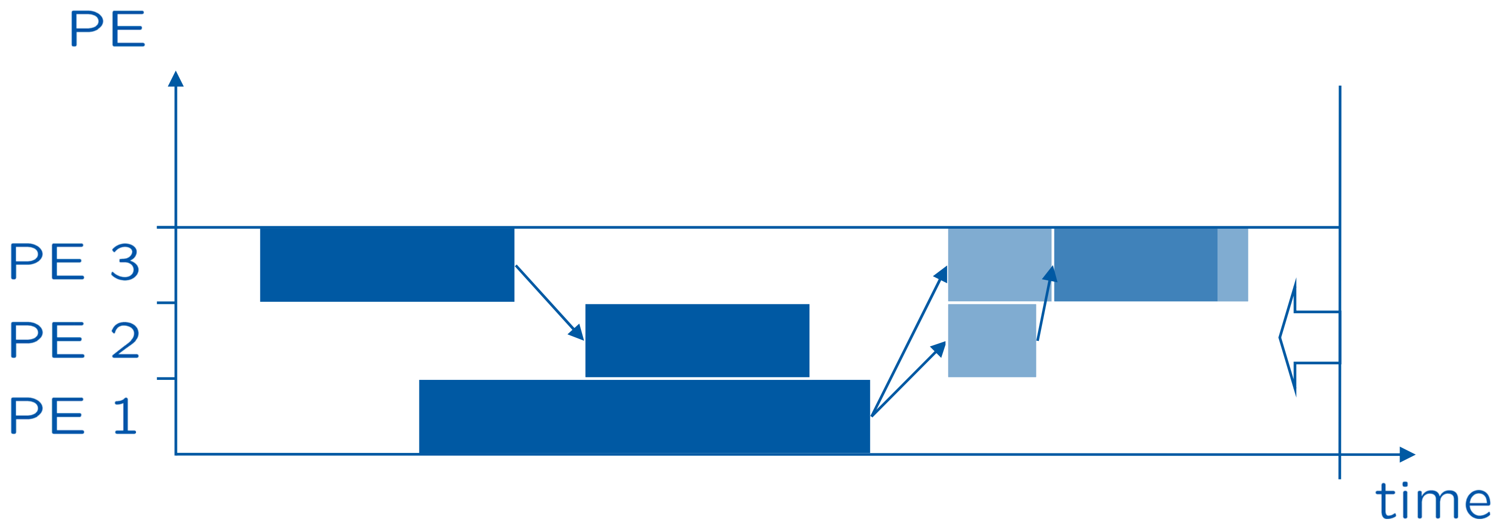


Mapping = Assignment of graph elements to graph elements

Node	→ (Node, start)	(task & data transfer mapping)
Edge	→ (Node, address)	(data mapping)
Port	→ Port	(task & data transfer mapping)

Task Mapping

= Strip Packing



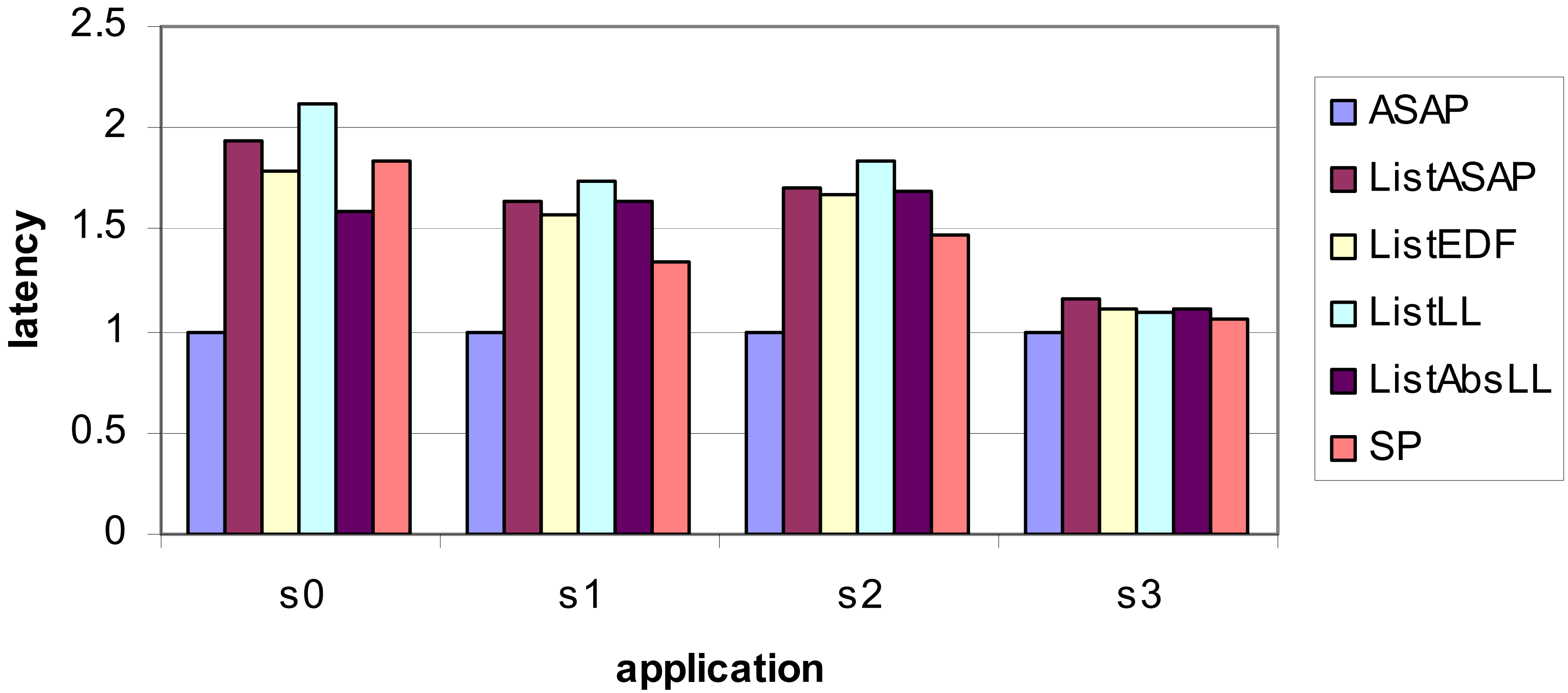
Currently implemented methods:

- Strip packing
- ASAP & ALAP w/o machine restrictions
- List ASAP, EDF & LL

TECHNISCHE UNIVERSITÄT DRESDEN

vodafone chair

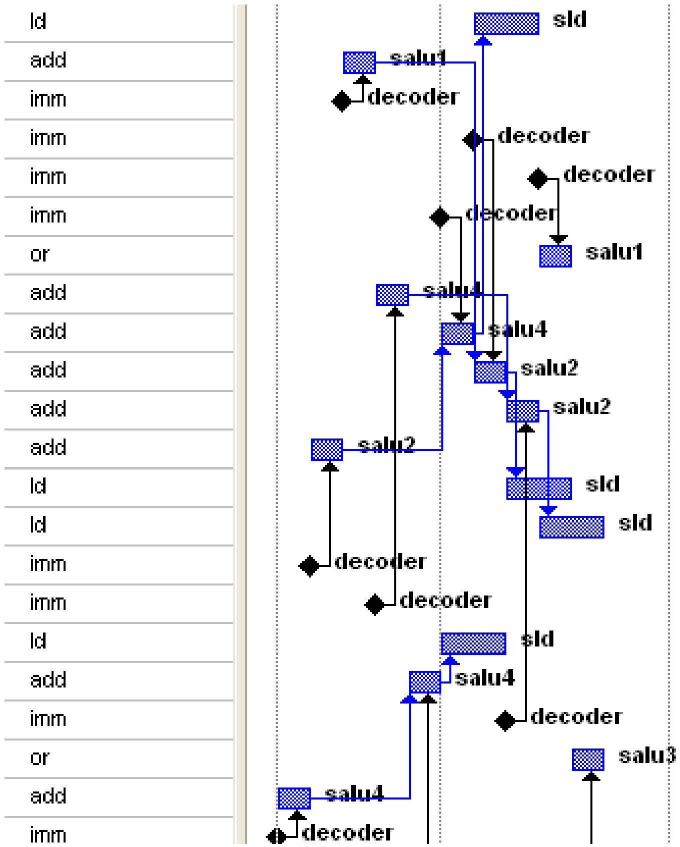
TGFF Task Mapping Results



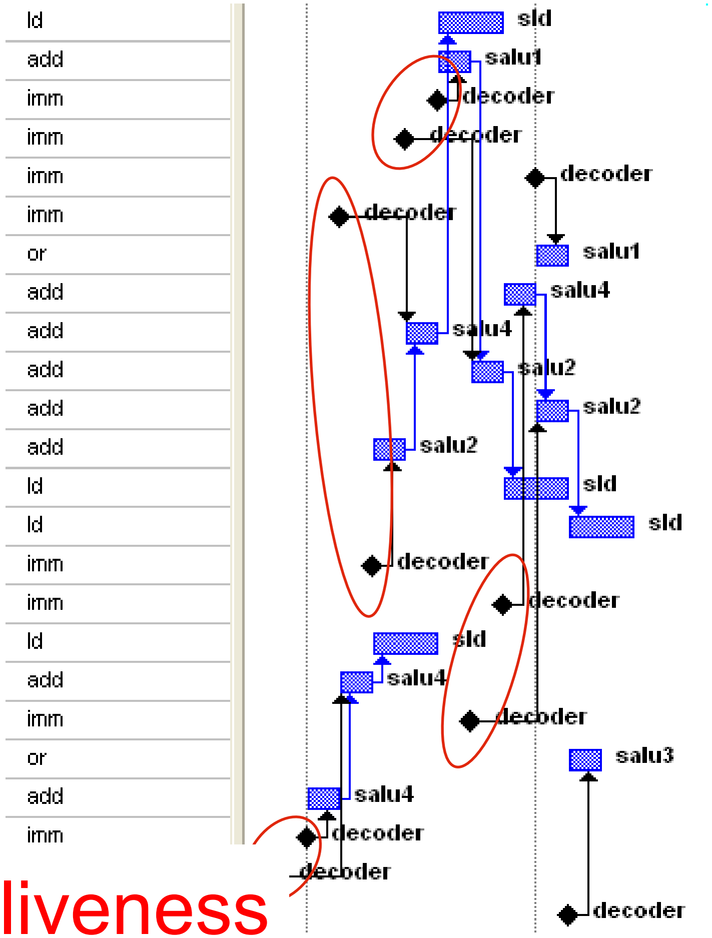
- Application: 32 nodes, 8 node types, max in/out degree = 2
- Architecture: 1 RISC, 2 DSPs

Comparison of ListEDF and ListASAP

ListEDF



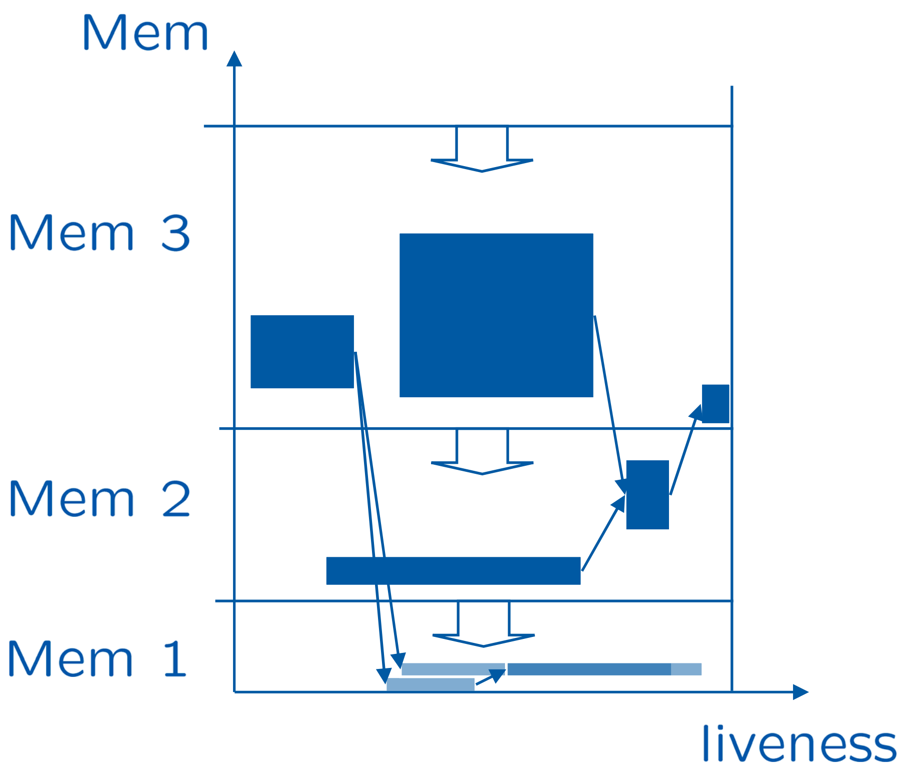
ListASAP



Problem in ListASAP: Data liveness

Data Mapping

= Shelf Packing



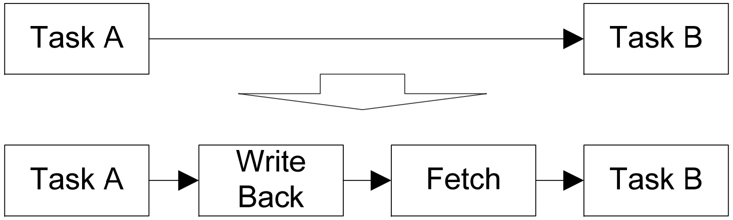
- Fixed total height and width → Minimize heights in each shelf
- Memory selection → weighted shelf selection
- Interference → “order” conditions

Data Transfer Mapping

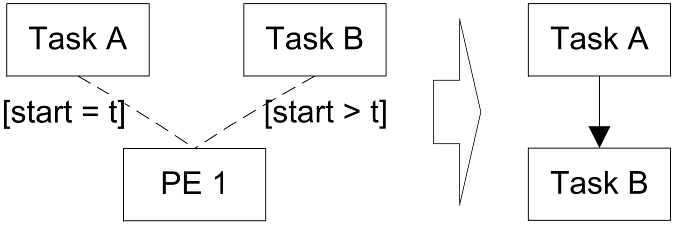
TECHNISCHE UNIVERSITÄT DRESDEN

vodafone chair

1. Two graph transformations
   - Insert transfer nodes, if data transfer is needed



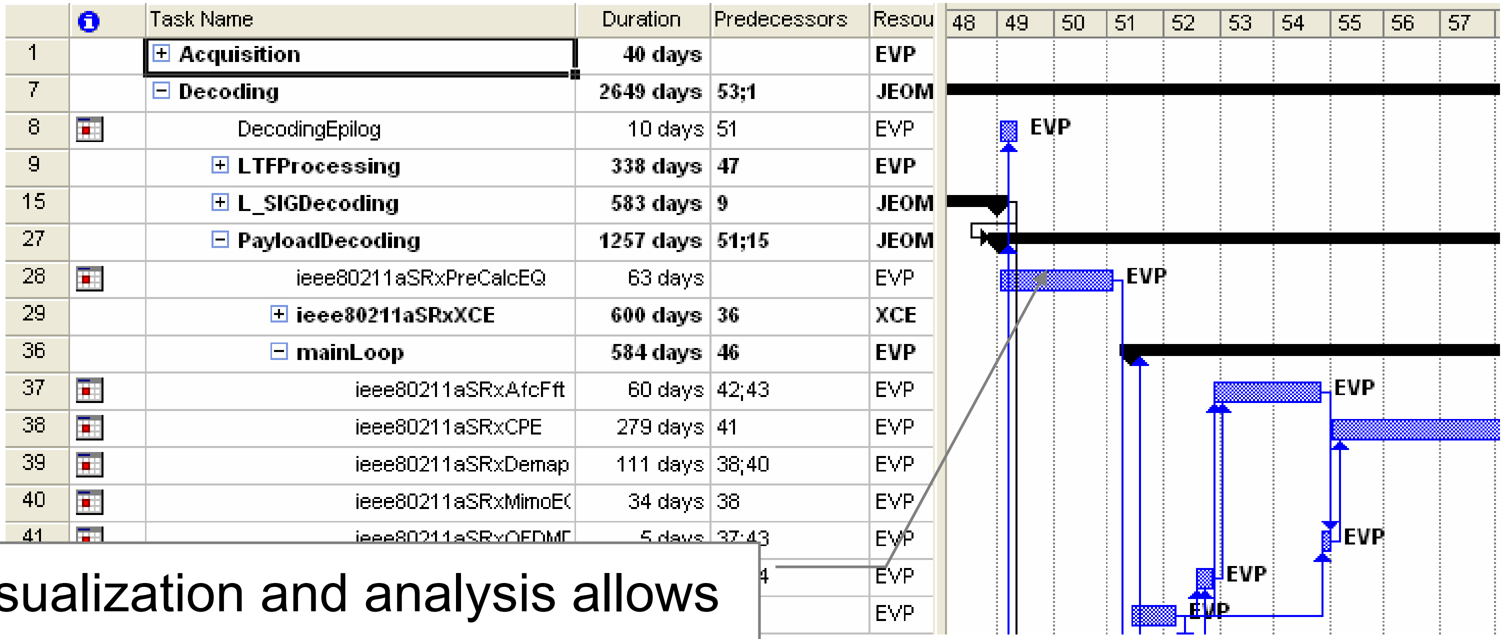
   - Insert task dependencies, if two tasks mapped to same resource



2. Task mapping (w/ given spatial task mapping)

Evaluation of Mapping Result

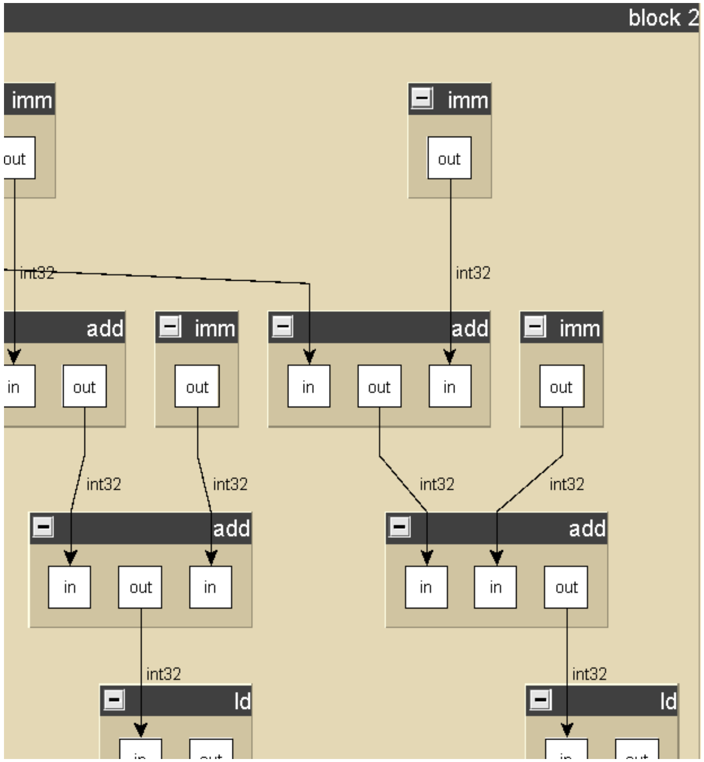
MS Project XML generated from mapping result

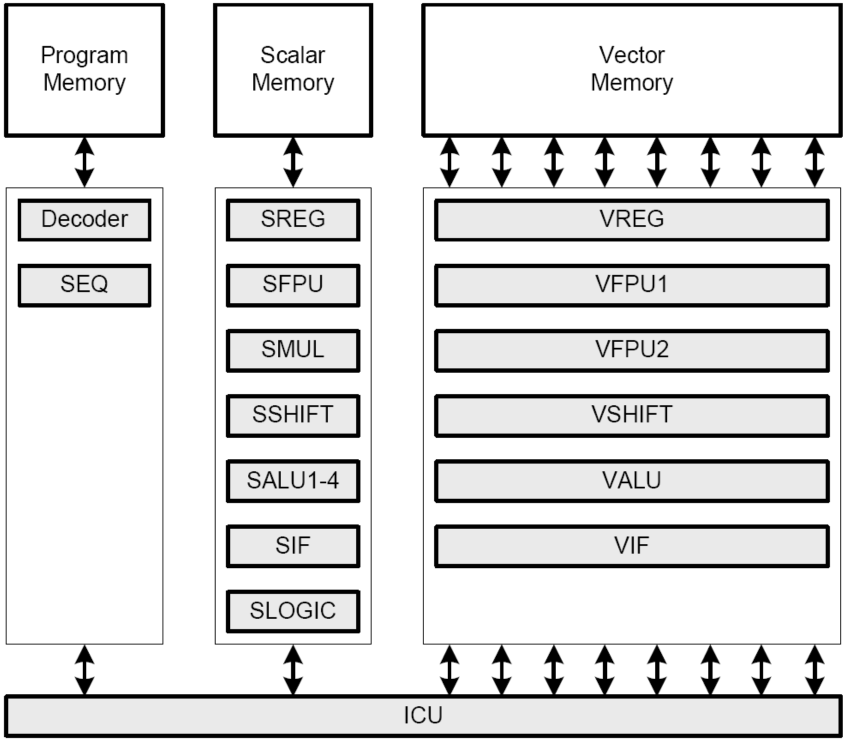


	ℹ	Task Name	Duration	Predecessors	Resou
1		⊞ Acquisition	40 days		EVP
7		⊟ Decoding	2649 days	53;1	JEOM
8	▦	DecodingEpilog	10 days	51	EVP
9		⊞ LTFProcessing	338 days	47	EVP
15		⊞ L_SIGDecoding	583 days	9	JEOM
27		⊟ PayloadDecoding	1257 days	51;15	JEOM
28	▦	ieee80211aSRxPreCalcEQ	63 days		EVP
29		⊞ ieee80211aSRxXCE	600 days	36	XCE
36		⊟ mainLoop	584 days	46	EVP
37	▦	ieee80211aSRxAfcFft	60 days	42;43	EVP
38	▦	ieee80211aSRxCPE	279 days	41	EVP
39	▦	ieee80211aSRxDemap	111 days	38;40	EVP
40	▦	ieee80211aSRxMimoE(	34 days	38	EVP
41	▦	ieee80211aSRxOFDMF	5 days	37;43	EVP
				4	EVP
					EVP

Visualization and analysis allows identification of bottlenecks and unveils possibilities for system improvement

Instruction Level Test Bench





Applications: Instruction Level

→ Large Control & Data Flow Graphs

MP-SoC: SAMIRA DSP

→ 21 Processors, 5 Memories

Example of Utilizing Mapping Result for System Improvement

Load Analysis (for various Signal Processing Algorithms)

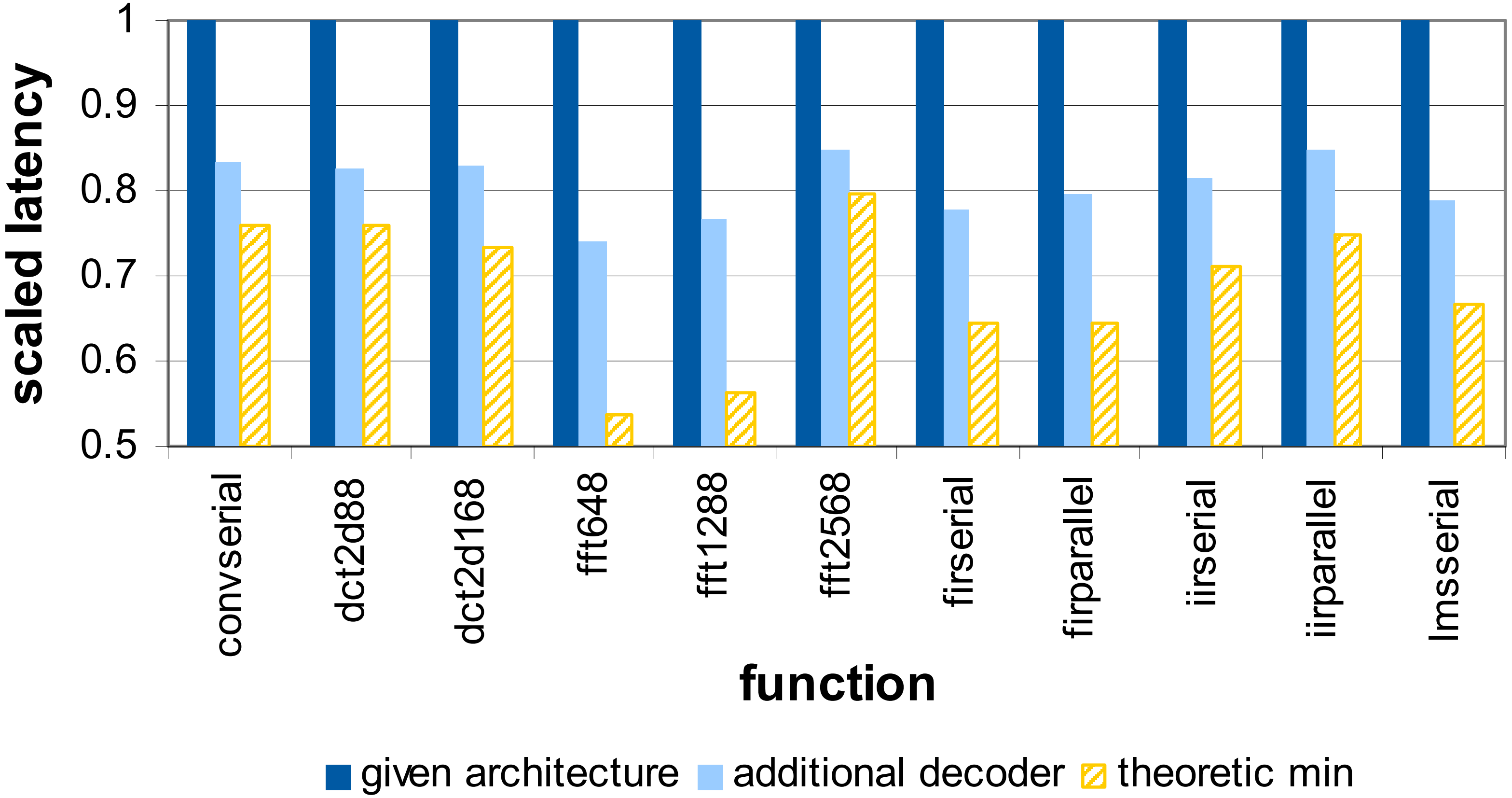
Functional Unit	Average Load
decoder	**67%**
salu1	42%
salu4	16%
sshift	15%
salu2	14%
smem	12%
smul	12%
vmem	9%
…	…

Example of Utilizing Mapping Result for System Improvement

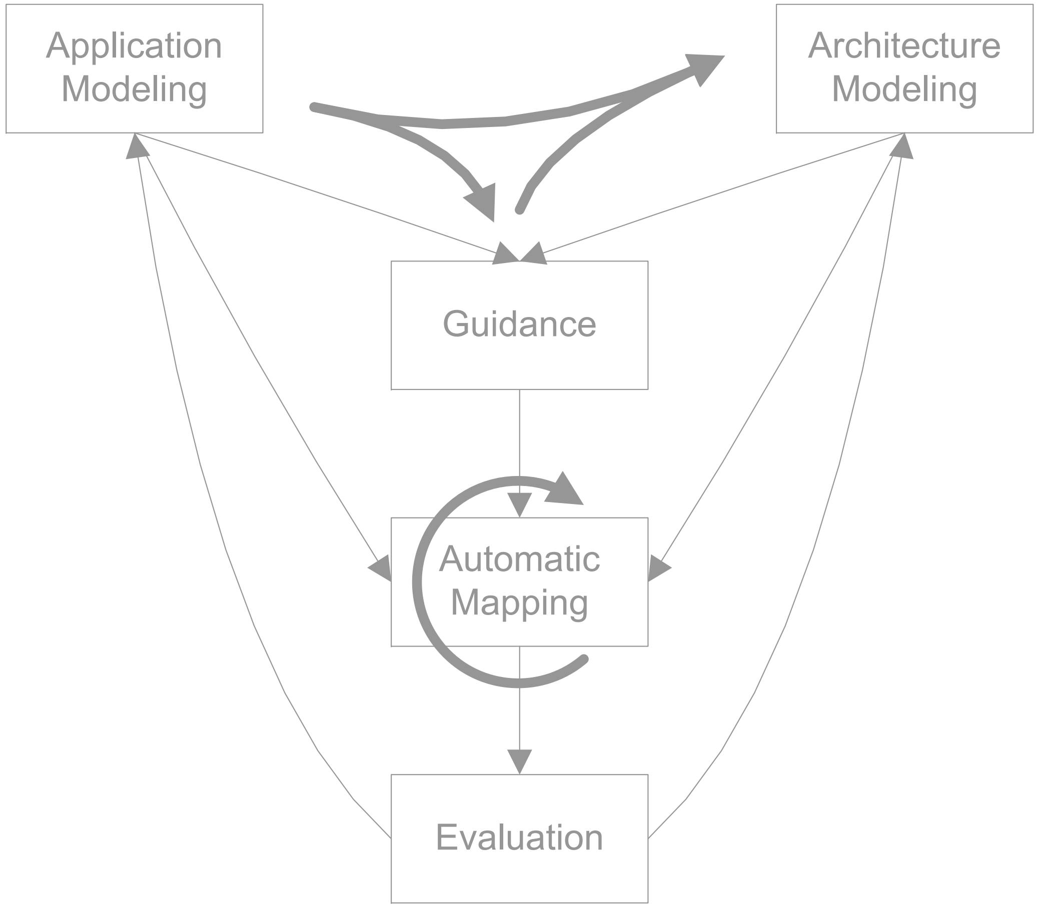
Load Analysis: High Load in Decoder

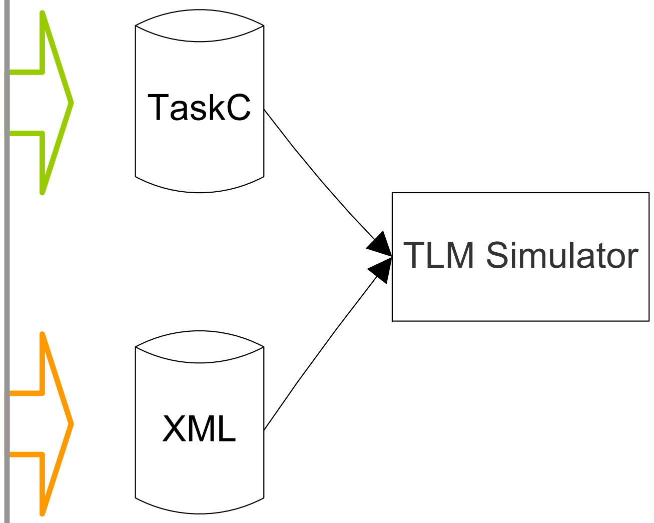
Idea: Add Second Decoder

Result:



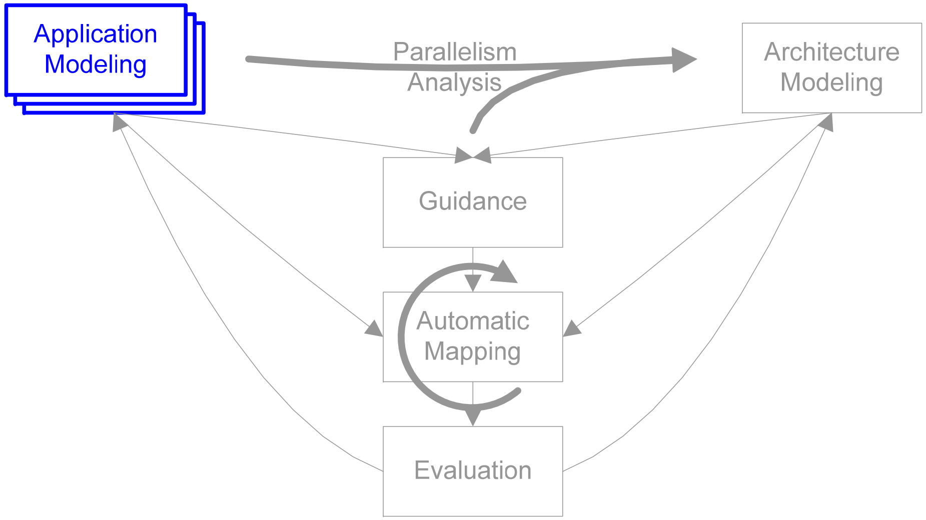
Passing Results to TLM



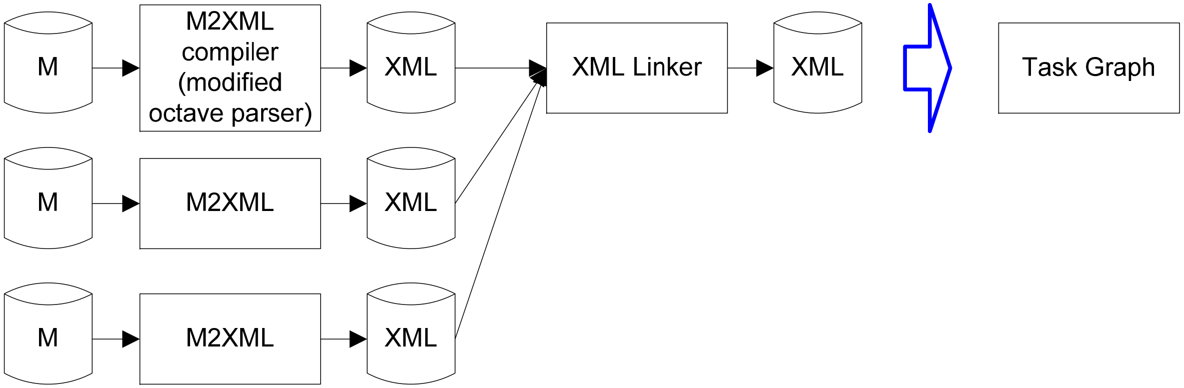


Next Steps

- Head towards multiple standards running in parallel



- Automate generation of task graphs


M
M2XML compiler (modified octave parser)
XML
M
M2XML
XML
M
M2XML
XML
XML Linker
XML
Task Graph