Parallel Cycle Simulation

Klaus Hering [*]

Institute of Computer Science
University of Leipzig

Abstract

Parallelization of logic simulation on register-transfer and gate level is a promising way to accelerate extremely time extensive system simulation processes for whole processor structures. In this report parallel simulation realized by means of the functional simulator *parallel-TEXSIM* based on the *clock-cycle algorithm* is considered. Within a corresponding simulation, several simulator instances co-operate over a loosely-coupled processor system, each instance simulating a part of a synchronous hardware design. Therefore, in preparation of parallel simulation, partitioning of hardware models is necessary, which is essentially determining efficiency of the following simulation.

A framework of formal concepts for an abstract description of parallel cycle simulation is developed. This provides the basis for *partition valuation* within partitioning algorithms.

Starting from the definition of a *Structural Hardware Model* as special bipartite graph *Sequential Cycle Simulation* is introduced as sequence of actions. Following a cone-based partitioning approach a *Parallel Structural Hardware Model* is defined as set of *Structural Hardware Models*. Furthermore, a model of parallel computation called *Communicating Processors* is introduced which is closely related to the well known *LogP* Model. Together with the preceding concepts it represents the basis for determining *Parallel Cycle Simulation* as sequence of action sets.

[*]khering@informatik.uni-leipzig.de

1 Introduction

Due to challenging technological capabilities the attainable complexity of *VLSI* designs is growing rapidly. Detecting design faults as early as possible prevents from wasting valuable resources. Therefore, the employment of verification processes in all design phases is inevitable. Simulation is a very important *VLSI* design verification method. The background of our work is given by functional simulation on register-transfer and gate level (logic simulation) without consideration of timing aspects. In [9] a simulation strategy is presented with underlying hardware models embodying complete processor structures and simulation stimuli (test cases) being microprogrammes or machine instruction sequences. During system simulation time-extensive simulation runs for final validation of complex designs are considered. Aiming at significant run time reductions for such simulation processes we parallelized the sequential functional simulator *TEXSIM*[1] which operates on the basis of the *clock-cycle algorithm*. *parallelTEXSIM* is documented in [2]. We chose a parallelization approach making use of model inherent parallelism. Within a corresponding parallel simulation, several simulator instances co-operate over a loosely-coupled processor system, each instance simulating a part of a synchronous hardware design. Therefore, in preparation of parallel simulation, partitioning of the whole hardware model is necessary which is essentially determining the run time behaviour of a following simulation.

In [3] a project comprising investigation, development and implementation of model partitioning algorithms in the context of *parallelTEXSIM* is outlined. A hierarchical partitioning strategy is introduced in [4] followed by a special instance called *mixture of experts approach* described in [5]. Based on ideas of D.Zike and W.Roesner presented in [8] we consider *fan-in cones* as elementary components for building model partitions. Related work is reported in [6] and [7].

The model partitioning problem can be formulated as a combinational optimization problem. In this context partitions are related to quantities (costs) which more or less directly express a connection to parallel simulation run time (*partition valuation*). For partition valuation a model of corresponding parallel simulation is needed. Choosing such a model appears as balance between (a necessary degree of) detail and (a sufficient degree of) simplicity.

[1]developed by *IBM*

Beyond the special background of partition valuation the objective of the present report is to summarize a framework of formal concepts for characterization of parallel cycle simulation realized by *parallelTEXSIM*, providing an abstract basis for development and investigation of corresponding model partitioning algorithms. A schematic representation of the key concepts is given in Tab.1.

Structural Hardware Model **(SHM)**
(bipartite graph)
$\Downarrow$
Sequential Cycle Simulation **(SCS)**
(sequence of actions)
$\Downarrow$
Parallel Structural Hardware Model **(PSHM)**
(set of SHMs)
$\Downarrow$
Extended Sequential Cycle Simulation **(ESCS)**
(sequence of actions)
$\Downarrow$
Parallel Cycle Simulation (PCS)
(sequence of action sets)
$\Uparrow$
Unrestricted Parallel Behaviour **(UPB)**
(sequence of action sets)
$\Uparrow$
Communicating Processors **(CP)**
(model of parallel computation)

Table 1: Concepts for modelling parallelTEXSIM simulation

At first a *Structural Hardware Model (SHM)* is defined as directed bipartite graph with a subset of the node set representing design components as for instance gates or latches. The corresponding complementary node set stands for wires. Underlying designs are assumed to be synchronous.

For assigning behaviour to a *SHM* corresponding to the simulation of one cycle a set of (node related) actions is introduced. These actions are considered to be basic simulation components. They are not supplied with semantic details as for instance state transferring functions (for a first approach of detailed semantic description of parallel cycle simulation see [2]). Taking into account a levelizing of nodes, special action sequences are chosen as *Sequential Cycle Simulation (SCS).*

Furthermore, a *Parallel Structural Hardware Model (PSHM)* is defined in relation to a cone-based partition of a *SHM.* The latter is to be interpreted as a representation of a whole design under consideration. A *PSHM* embodies a set of *SHM*s each of them determined by a partition component (set of cones). For *SHM*s which are elements of a *PSHM*, *Extended Sequential Cycle Simulation (ESCS)* is introduced as component behaviour. An *ESCS* is a sequence of actions containing a special action for the representation of communication between *PSHM* components.

Finally, *Parallel Cycle Simulation (PCS)* is defined as behaviour of a *PSHM.* A *PCS* represents a sequence of action sets with the individual actions related to *ESCS*s belonging to components of the *PSHM* considered. As basis for this definition a model of parallel computation called *Communicating Processors (CP)* is introduced. Besides its application for modelling parallel cycle simulation *CP* enables the investigation of various communication mechanisms for asynchronously working processor systems. *CP* behaviour is determined by given (sequential) component behaviour and communication mechanisms involved. As general framework *Unrestricted Parallel Behaviour (UPB)* is defined, which can be restricted to concrete *CP* behaviour by inclusion of synchronization conditions.

2 Structural Hardware Model

Essential components of our structural hardware model are given by the family of sets listed below :

- M_E : logical boxes - representing logical gates, multiplexers, ...
- M_I : input boxes - representing design elements for signal input
- M_O : output boxes - representing design elements for signal output

- M_L : storing boxes - representing clocked elements (latches)
- M_S : nets - representing wires

In the following, $N(x)$ and $N^{-}(x)$ denote the set of immediate successors and predecessors of a node x within a directed graph, respectively (with the usual extension of the definitions to sets of arguments).

Definition 2.1 (SHM) *Let M_E, M_I, M_O, M_L, M_S be pairwise disjoint finite sets, $M_B = M_E \cup M_I \cup M_O \cup M_L$ and M_B, $M_S \neq \emptyset$. Then a directed bipartite graph $M = (M_B, M_S, M_{\mathcal{R}})$ satisfying the following conditions is called* ***Structural Hardware Model (SHM)****:*

1. $\{x \mid x \in M_B \cup M_S \wedge N^{-}(x) = \emptyset\} = M_I$
2. $\{x \mid x \in M_B \cup M_S \wedge N(x) = \emptyset\} = M_O$
3. *Any directed cycle in M includes at least one element of M_L.*

Figure 1 roughly illustrates a *SHM*. Thick arrows represent subsets of M_S.

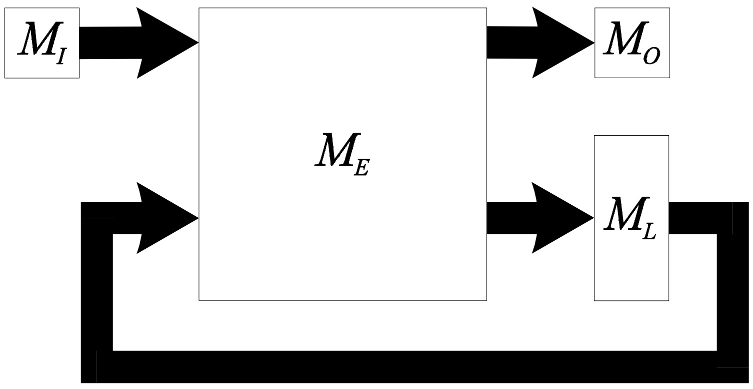


Figure 1: A Structural Hardware Model schematically

Remark 2.1 Condition 3 expresses the exclusion of asynchronous feedbacks in combinational logic (synchrony of the underlying design).

Remark 2.2 There exists a sequence of design description transformations leading to *SHM*s starting from original descriptions given in *DSL* or *BDL/S* (a combination of both languages can be used within one description). So called *protos* as data structures of the *Design Automation Data Base DA_DB*[2] form the last stage before reaching *SHM*s.

For defining a behaviour of *SHM*s a levelizing of logical boxes is introduced. Because of this, logical boxes are concentrated in groups according to the longest distance to storing or input boxes (via input paths). Boxes belonging to the same group are to be interpreted as carriers of simulation activities which can be performed independently from each other.

Definition 2.2 (Levelizing) *Let M be a* SHM *with $M_E \neq \emptyset$ and $\overline{L_i}$ be defined as follows:*

1. $\overline{L_0} = M_I \cup M_L$
2. $\overline{L_{i+1}} = \overline{L_i} \cup \{x | x \in M_E \wedge N^-(N^-(x)) \subseteq \overline{L_i}\}$

Let be

$$k = \min\{j | \overline{L_j} = \overline{L_{j+1}}\}. \tag{2.1}$$

Then

$$\mathcal{L}(M) = \{L_1, \ldots, L_k\} \textit{ with } L_i = \overline{L_i} \setminus \overline{L_{i-1}} \textit{ for } i \in \{1, \ldots, k\}$$

is called **levelizing** of M_E.

Remark 2.3 For justification of the above definition, the existence of $k \geq 1$ with property 2.1 is evident.

Lemma 2.1 *$\mathcal{L}(M)$ is a partition of M_E (in mathematical sense).*

[2]developed by *IBM*

3 Sequential Cycle Simulation

For the representation of basic components of cycle simulation, abstract actions (bound to nodes different from nets) are introduced. They are not supplied with semantic details as, for instance, state transferring functions. With respect to partition valuation, actions are considered as sources of simulation expense. In relation to a *SHM* M we consider an action set

$$\mathcal{A} = \mathcal{A}_E \cup \mathcal{A}_I \cup \mathcal{A}_O \cup \mathcal{A}_L \tag{3.1}$$

with a bijective assignment function $a : M_B \rightarrow \mathcal{A}$ assuming $a(M_\omega) = \mathcal{A}_\omega$ for $\omega \in \{E, I, O, L\}$.

One has to interpret a as an action standing for the evaluation of a logical function for $a \in \mathcal{A}_E$ and as an update action (of an input, output or latch, respectively) for $a \in \mathcal{A}_I \cup \mathcal{A}_O \cup \mathcal{A}_L$.

Definition 3.1 (SCS) *With* $\|$ *denoting sequence concatenation (including the usual extension to sets of sequences),* $\mathcal{A}^+$ *embodying the set of all finite non-empty sequences over* $\mathcal{A}$ *and* $k = |\mathcal{L}(M)|$

$$s_{seq} \in \mathcal{A}^+$$

satisfying the following conditions is called ***Sequential Cycle Simulation (SCS)*** *with respect to* M*:*

1. *Each action of* $\mathcal{A}$ *appears exactly once within* s_{seq}.
2. $s_{seq} \in {\mathcal{A}_I}^+ \parallel a\left(L_1\right)^+ \parallel ... \parallel a\left(L_k\right)^+ \parallel {\mathcal{A}_O}^+ \parallel {\mathcal{A}_L}^+$

Remark 3.1 All boxes imply exactly one action during the simulation of one cycle (*time-driven simulation*). In future work it might be of interest to consider the *TEXSIM* capability of excluding parts of combinational logic from cycle simulation.

Remark 3.2 Levelizing determines a pre-ordering of actions belonging to logical boxes. The order of actions within sub-sequences of s_{seq} belonging to${\mathcal{A}_I}^+$, $a\left(L_j\right)^+$, ${\mathcal{A}_O}^+$ or${\mathcal{A}_L}^+$ is assumed to be without relevance for the simulation result.

4 Parallel Structural Hardware Model

In the following, *PSHM*s are introduced to give a structural representation of design partitions related to *parallelTEXSIM* simulation. Our partitioning approach is based on *fan-in cones* (see [4]). In our context a *fan-in cone* comprises the set of all logical boxes, which are (potentially) able to influence via their box-related actions the respective action of a cone-defining (head) box during simulation of one cycle. We remark that the cone-head itself is an element of the corresponding cone, too.

Definition 4.1 (Fan-in cone) *With M being a* SHM*, the* ***fan-in cone*** *$co(x)$ for $x \in M_E \cup M_L \cup M_O$ is defined as the smallest set satisfying the following conditions :*

1. $x \in co(x)$
2. $y \in M_E \wedge N(N(y)) \cap co(x) \neq \emptyset \rightarrow y \in co(x)$

Obviously, $co(x) \subseteq M_E \cup M_L \cup M_O$ is valid. Input boxes and storing boxes (different from x) immediately feeding the cone $co(x)$ do not belong to its elements.

Lemma 4.1 *For $x, y \in M_E \cup M_L \cup M_O$ the following relation holds:*

$$co(x) = co(y) \Longrightarrow x = y$$

Therefore, calling x *the* head of $co(x)$ is justified.

Basic elements for building partitions of a hardware model M are given by the following set (cone set of M) :

$$Co(M) = \{co(x) | x \in M_L \cup M_O\} \tag{4.1}$$

Lemma 4.2 $M_E \cup M_L \cup M_O = \bigcup_{c \in Co(M)} c$

Definition 4.2 (Partition) *Let M be a* SHM. *Then a partition* Π *(in mathematical sense) of $Co(M)$ is called a* ***partition*** *of M.*

Remark 4.1 Different elements (cones) of $Co(M)$ may have common boxes (*cone overlapping*). If we assume, that a partition component determines the model part to be handled by a simulator instance on a single processor during parallel simulation, then overlapping cones as elements of different partition components stand for replication of simulation work. Besides this drawback the cone-based partitioning approach bears the advantage that interprocessor communication during parallel cycle simulation is necessary only at cycle boundaries.

In the following, the objective is to construct *PSHM*s with respect to partitions of *SHM*s on the basis of "sub-models" defined by partition components. Later on, sub-model behaviour will be combined to behaviour of the whole parallel model characterizing parallel cycle simulation. For sub-model definition some concepts related to partition components (cone sets) are introduced.

Let M be a *SHM*. For arbitrary cone sets $\mathcal{C} \subseteq Co(M)$ we define:

- $B^{\mathcal{C}} = \bigcup\limits_{c \in \mathcal{C}} c$
  (set of all boxes belonging to at least one cone of $\mathcal{C}$)

- $\text{head}(\mathcal{C}) = \{x | co(x) \in \mathcal{C}\}$
  (set of all heads belonging to cones of $\mathcal{C}$)

- $\text{feed}(\mathcal{C}) = \{s \mid s \in M_S \wedge N(s) \cap B^{\mathcal{C}} \neq \emptyset \wedge N^-(s) \nsubseteq B^{\mathcal{C}}\}$
  (set of all nets feeding $\mathcal{C}$ from "outside")
  Corresponding nets have at least one sink box within a cone of $\mathcal{C}$ and one source box lying outside all cones of $\mathcal{C}$.

- $\text{leave}(\mathcal{C}) = \{s \mid s \in M_S \wedge N^-(s) \cap \text{head}(\mathcal{C}) \neq \emptyset \wedge N(s) \cap B^{Co(M) \setminus \mathcal{C}} \neq \emptyset\}$
  (set of all nets leaving $\mathcal{C}$ via cone-heads)
  Corresponding nets have at least one cone-head of $\mathcal{C}$ as a source box

and one sink box belonging to a cone outside $\mathcal{C}$. Due to possible cone overlapping this does not exclude the existence of a cone in $\mathcal{C}$ covering the corresponding sink box as well.

Lemma 4.3 $s \in \text{feed}(\mathcal{C}) \wedge b \in N^{-}(s) \setminus B^{\mathcal{C}} \longrightarrow b \in M_I \cup M_L$

Remark 4.2 The elements of $\text{feed}(\mathcal{C}) \cup \text{leave}(\mathcal{C})$ are to be interpreted as carriers of information at cycle boundaries with respect to $\mathcal{C}$. Nets belonging to $\text{feed}(\mathcal{C})$ can have a source box within a cone of $\mathcal{C}$ and nets belonging to $\text{leave}(\mathcal{C})$ can have a sink box within a cone of $\mathcal{C}$. Remark, that there may exist nets not included in $\text{leave}(\mathcal{C})$ having a source box within a cone of $\mathcal{C}$ and a sink box lying outside of all cones belonging to $\mathcal{C}$. These nets are not related to communication at cycle boundaries.

Now, models related to cone sets, embodying components of a partition of a *SHM*, are introduced.

Definition 4.3 (Sub-model) *Let M be a* SHM , Π *a partition of M and $\mathcal{C} \in \Pi$. $M_{\Pi}^{\mathcal{C}} = \left(M_B^{\mathcal{C}}, M_S^{\mathcal{C}}, M_{\mathcal{R}}^{\mathcal{C}}\right)$ with $M_B^{\mathcal{C}} = M_E^{\mathcal{C}} \cup M_I^{\mathcal{C}} \cup M_O^{\mathcal{C}} \cup M_L^{\mathcal{C}}$ is called **sub-model** of M with respect to Π, if the corresponding components satisfy the following conditions :*

1. $M_E^{\mathcal{C}} = B^{\mathcal{C}} \cap M_E,\ M_L^{\mathcal{C}} = B^{\mathcal{C}} \cap M_L$
2. $M_I^{\mathcal{C}} = M_{I,I}^{\mathcal{C}} \cup M_{I,L}^{\mathcal{C}}$
   - $M_{I,I}^{\mathcal{C}} = N^{-}(\text{feed}(\mathcal{C})) \cap M_I$
   - $M_{I,L}^{\mathcal{C}} = \{(\mathcal{C}', s) | s \in \text{feed}(\mathcal{C}) \wedge N^{-}(s) \cap \left(B^{\mathcal{C}'} \setminus B^{\mathcal{C}}\right) \neq \emptyset \wedge \mathcal{C}' \in \Pi \wedge \mathcal{C}' \neq \mathcal{C}\}$
3. $M_O^{\mathcal{C}} = M_{O,O}^{\mathcal{C}} \cup M_{O,L}^{\mathcal{C}}$
   - $M_{O,O}^{\mathcal{C}} = B^{\mathcal{C}} \cap M_O$
   - $M_{O,L}^{\mathcal{C}} = \{(s, \mathcal{C}') | s \in \text{leave}(\mathcal{C}) \wedge N(s) \cap B^{\mathcal{C}'} \neq \emptyset \wedge \mathcal{C}' \in \Pi \wedge \mathcal{C}' \neq \mathcal{C}\}$
4. $M_S^{\mathcal{C}} = \{s \mid s \in M_S \wedge N(s) \cap B^{\mathcal{C}} \neq \emptyset\} \cup \text{leave}(\mathcal{C})$

5. $M^{\mathcal{C}}_{\mathcal{R}} = [M_{\mathcal{R}} \cap [((B^{\mathcal{C}} \cup M^{\mathcal{C}}_{I,I}) \times M^{\mathcal{C}}_{S}) \cup (M^{\mathcal{C}}_{S} \times B^{\mathcal{C}})]] \cup$
   $\{((\mathcal{C}', s), s) | (\mathcal{C}', s) \in M^{\mathcal{C}}_{I,L}\} \cup$
   $\{(s, (s, \mathcal{C}')) | (s, \mathcal{C}') \in M^{\mathcal{C}}_{O,L}\}$

In all cases, N and N^- are related to M.

Remark 4.3 The elements $(\mathcal{C}', s)$ of $M^{\mathcal{C}}_{I,L}$ are to be interpreted as input boxes for $M^{\mathcal{C}}_{\Pi}$. They are related to the set $N^-(s) \cap \left(B^{\mathcal{C}'} \setminus B^{\mathcal{C}}\right)$ of "foreign latches" belonging to the component $\mathcal{C}' \neq \mathcal{C}$ of Π feeding $\mathcal{C}$ via the net s. $N^-(s) \cap \left(B^{\mathcal{C}'} \setminus B^{\mathcal{C}}\right) \subseteq M_L$ is a consequence of Lemma 4.3 and of the exclusion of input boxes from cones. The elements of $M^{\mathcal{C}}_{I,I}$ embody global input boxes of $M^{\mathcal{C}}_{\Pi}$.

Remark 4.4 The elements $(s, \mathcal{C}')$ of $M^{\mathcal{C}}_{O,L}$ are to be interpreted as output boxes for $M^{\mathcal{C}}_{\Pi}$ related to the set $N(s) \cap B^{\mathcal{C}'}$ of "foreign boxes" belonging to the component $\mathcal{C}' \neq \mathcal{C}$ of Π fed by $\mathcal{C}$ via the net s. Due to possible cone overlapping, one element of $N(s)$ can belong to different components of Π. The elements of $M^{\mathcal{C}}_{O,O}$ embody global output boxes of $M^{\mathcal{C}}_{\Pi}$.

Lemma 4.4 *Let M be a* SHM *and Π be the single-block partition$\{Co(M)\}$. Then $M^{Co(M)}_{\{Co(M)\}} = M$ is valid.*

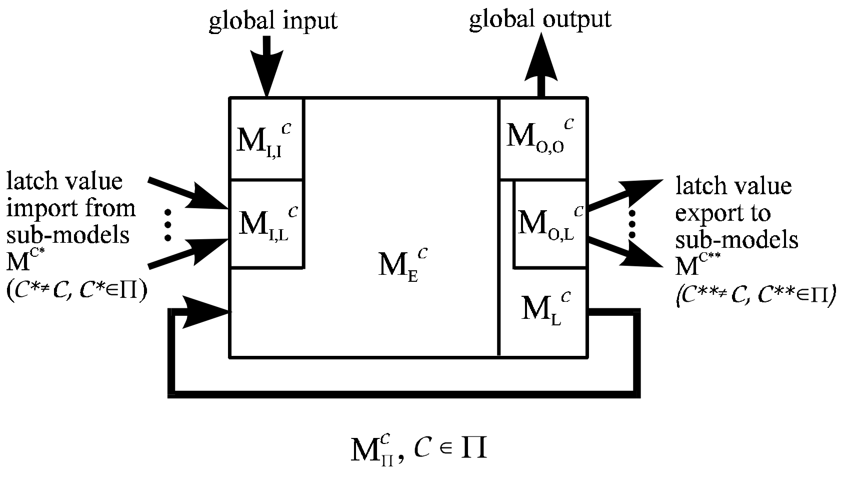
Lemma 4.5 *A sub-model $M^{\mathcal{C}}_{\Pi}$ of M is a SHM.*

Definition 4.4 (PSHM) *Let M be a* SHM *and Π be a partition of M.*

$$M^{\Pi} = \left\{M^{\mathcal{C}}_{\Pi} \mid \mathcal{C} \in \Pi\right\}$$

*is called **Parallel Structural Hardware Model (PSHM)** with respect to Π.*

In Figure 2 a sub-model $M^{\mathcal{C}}_{\Pi}$ in the context of a *PSHM* is represented schematically. M^{Π} implies a binary communication relation over Π. Set



global input
global output
$M_{I,I}^{C}$
$M_{O,O}^{C}$
$M_{I,L}^{C}$
$M_{O,L}^{C}$
M_{E}^{C}
M_{L}^{C}
latch value import from sub-models M^{C^*} $(C^* \neq C, C^* \in \Pi)$
latch value export to sub-models $M^{C^{**}}$ $(C^{**} \neq C, C^{**} \in \Pi)$
$M_{\Pi}^{C}, C \in \Pi$

Figure 2: A sub-model in PSHM context

$$M_{I,L}^{C' \to C''} = \{(C', s) | (C', s) \in M_{I,L}^{C''}\} \text{ and } M_{O,L}^{C' \to C''} = \{(s, C'') | (s, C'') \in M_{O,L}^{C'}\}.$$

$M_{I,L}^{C' \to C''}$contains all input boxes of $M^{C''}$ related to output boxes in $M^{C'}$ and $M_{O,L}^{C' \to C''}$ contains all output boxes of $M^{C'}$ related to input boxes in $M^{C''}$.

Obviously, we have $\bigcup\limits_{C^* \in \Pi} M_{I,L}^{C^* \to C''} = M_{I,L}^{C''}$ and $\bigcup\limits_{C^* \in \Pi} M_{O,L}^{C' \to C^*} = M_{O,L}^{C'}$.

Lemma 4.6 Let Π be a partition of a *SHM* M and $C', C'' \in \Pi$.
Then $\left| M_{I,L}^{C' \to C''} \right| = \left| M_{O,L}^{C' \to C''} \right|$is valid.

Definition 4.5 (Communication relation) *Let* M^{Π} *be* PSHM. *Then*

$$Comm^{\Pi} = \left\{ (C', C'') \mid C', C'' \in \Pi \wedge M_{O,L}^{C' \to C''} \neq \emptyset \right\}$$

is called ***communication relation*** *of* M^{Π}.

Remark 4.5 Regarding the components of a *PSHM* M^{Π} as representations of model parts to be handled on single processors during parallel simulation, $(C', C'') \in Comm^{\Pi}$ means interprocessor communication (directed from the processor handling C' to this handling C'').

5 Extended Sequential Cycle Simulation

After introduction of a parallel hardware model from structural point of view in the previous section, the behaviour of its components is under consideration now. As for *SHM*s not standing in the context of a *PSHM*, the behaviour of *PSHM* components is chosen as action sequence again.

Consider a sub-model $M_{\Pi}^{\mathcal{C}}$ of M with respect to Π. In this context an action set

$$\mathcal{A}^{\mathcal{C}} = \mathcal{A}_{E}^{\mathcal{C}} \cup \mathcal{A}_{I,I}^{\mathcal{C}} \cup \mathcal{A}_{I,L}^{\mathcal{C}} \cup \mathcal{A}_{O,O}^{\mathcal{C}} \cup \mathcal{A}_{O,L}^{\mathcal{C}} \cup \mathcal{A}_{L}^{\mathcal{C}} \cup \{c\} \tag{5.1}$$

with a bijective assignment function $a : M_{B}^{\mathcal{C}} \rightarrow \mathcal{A}^{\mathcal{C}} \setminus \{c\}$ assuming $a\left(M_{\omega}^{\mathcal{C}}\right) = \mathcal{A}_{\omega}^{\mathcal{C}}$ (ω representing an arbitrary variant of the lower indices appearing in (5.1)) is introduced. Different from *SCS*, a special action c not bound to a special box and representing component communication at cycle boundaries is involved. $\mathcal{A}^{\mathcal{C}}$ reflects the splitting of the sets of input and output boxes within $M_{\Pi}^{\mathcal{C}}$.

Definition 5.1 (ESCS) *Let $M_{\Pi}^{\mathcal{C}}$ be a sub-model of M with respect to Π, $\mathcal{L}\left(M_{\Pi}^{\mathcal{C}}\right) = \{L_{1}^{\mathcal{C}}, \ldots, L_{k^{\mathcal{C}}}^{\mathcal{C}}\}$ be the levelizing of $M_{E}^{\mathcal{C}}$ and $\mathcal{A}^{\mathcal{C}}$ be given as in (5.1). Then*

$$s_{seq}^{\mathcal{C}} \in \left(\mathcal{A}^{\mathcal{C}}\right)^{+}$$

*satisfying the following conditions is called **Extended Sequential Cycle Simulation (ESCS)** with respect to $M_{\Pi}^{\mathcal{C}}$:*

1. *Each action of $\mathcal{A}^{\mathcal{C}}$ appears exactly once within $s_{seq}^{\mathcal{C}}$.*
2. *$s_{seq}^{\mathcal{C}} = s_{cycle}^{\mathcal{C}} \| s_{comm}^{\mathcal{C}}$ with*
   - $s_{cycle}^{\mathcal{C}} \in {\mathcal{A}_{I,I}^{\mathcal{C}}}^{+} \| a\left(L_{1}^{\mathcal{C}}\right)^{+} \| ... \| a\left(L_{k^{\mathcal{C}}}^{\mathcal{C}}\right)^{+} \| {\mathcal{A}_{O,O}^{\mathcal{C}}}^{+} \| {\mathcal{A}_{L}^{\mathcal{C}}}^{+}$ *and*
   - $s_{comm}^{\mathcal{C}} = s_{pre\_comm}^{\mathcal{C}} \| (c) \| s_{post\_comm}^{\mathcal{C}}$,
     $s_{pre\_comm}^{\mathcal{C}} \in {\mathcal{A}_{O,L}^{\mathcal{C}}}^{+}$, $s_{post\_comm}^{\mathcal{C}} \in {\mathcal{A}_{I,L}^{\mathcal{C}}}^{+}$

Remark 5.1 $s_{cycle}^{\mathcal{C}}$ appears as *SCS* with respect to $M_{\Pi}^{\mathcal{C}}$ modified by omitting the actions from $\mathcal{A}_{I,L}^{\mathcal{C}} \cup \mathcal{A}_{O,L}^{\mathcal{C}}$ (assigned to boxes from $M_{I,L}^{\mathcal{C}} \cup M_{O,L}^{\mathcal{C}}$). $s_{comm}^{\mathcal{C}}$ represents 3 phases of communication related work:

- $s^{\mathcal{C}}_{pre\_comm}$: preparation of interprocessor communication under sending aspect with respect to $M^{\mathcal{C}}_{\Pi}$ (extraction of sub-model data with following placement in communication related structures)

- c: (possibly) complex communication action at cycle boundaries

- $s^{\mathcal{C}}_{post\_comm}$: post-processing of interprocessor comunication under receiving aspect with respect to $M^{\mathcal{C}}_{\Pi}$ (extraction of data from communication related structures with following placement in sub-model structures)

The structure of $s^{\mathcal{C}}_{seq}$ reflects the restriction of communication between components involved in parallel cycle simulation to cycle boundaries. For combining the behaviour of *PSHM* components to a behaviour of the whole *PSHM* we make use of a model of parallel computation.

6 Communicating Processors

A *model of parallel computation* embodies a combination of descriptions of a more or less abstract parallel processor structure (consisting of *processing elements*, *memory modules* and an *interconnection network*) and its behaviour. It determines a framework for the investigation of concurrent processes co-operating within the realization of parallel algorithms. A variety of corresponding models has been developed, all of them compromising on a necessary degree of detail (to allow addressing of relevant problems) and a sufficient degree of simplicity (to keep these problems tractable). Working on parallel logic simulation, we were looking for a model of parallel computation

- related to *loosely-coupled* parallel machines without supposing a special architecture, but giving the possibility of introducing architecture dependent properties via parameters,

- allowing different communication mechanisms to consider and

- describing behavioural capabilities of single processes in terms of sequences of abstract actions to have the possibility of relating them to several interpretations (for instance, to simulation time amount as basis for partition valuation).

In [1] a model of a distributed-memory multiprocessor with processors communicating by point-to-point messages is introduced. The model is called *LogP* with the four letters representing the main parameters of the model:

- L as upper bound of the *latency* for communicating "small" messages from source to target
- o as *overhead* in terms of the length of time a processor is engaged in transmission or reception of a message
- g as *gap* representing the minimum time interval between consecutive message transmissions / receptions at one processor
- P as the number of processors and memory modules considered

LogP specifies the performance characteristics of an underlying interconnection network via the parameters given above without consideration of special network topologies. Inspired by *LogP*, we introduce *Communicating Processors (CP)* as model of a loosely-coupled parallel machine providing the possibility of integrating a set of communication mechanisms corresponding to topical needs.

Definition 6.1 (CP) *A model of parallel computation called* Communicating Processors (CP) *is defined as triplet* $\mathcal{P} = (P_P, P_A, P_C)$ *where*

- $P_P = \{P_1, \ldots, P_n\}$ *is a set of (abstract) processors working asynchronously,*
- $P_A = \{\mathcal{A}_1, \ldots, \mathcal{A}_n\}$ *is a family of finite processor-bound action sets and*
- $P_C = \{\mathcal{M}_1, \ldots, \mathcal{M}_l\}$ *is a finite set of communication mechanisms. A communication mechanism is given as an ordered pair with a* qualitative characteristic *as first component and a (possibly empty) set of* quantitative characteristics *as second component. A qualitative characteristic comprises*
  - *the determination of actions related to the corresponding mechanism*

- *the determination of source/target relations within a set of involved processors*
- *the determination of synchronization conditions*

A quantitative characteristic appears as a real function or constant, valuating a communication-related aspect.

Remark 6.1 In the context of *CP* actions represent the execution of operations on the processors under consideration. There is nothing said about their complexity. The execution of an extensive high-level procedure can be regarded as well as handling a microcode instruction. There is a certain freedom of assigning semantic details to actions corresponding to the requirements of topical objectives. We use actions as basic blocks to build sequences interpreted as behaviour of an underlying processor.

Remark 6.2 Within P_C elemental *point-to-point* communication mechanisms built on *send-* and *receive-* actions can be considered as well as *collective* communication mechanisms realizing broadcasts or related tasks with (usually) more than two actions (on different processors) involved. Specifying synchronization conditions results in a potential blocking of actions from behavioural point of view. This directly restricts possible combinations of component behaviour sequences within *CP* behaviour. An example of reifying P_C is given in relation to the definition of *Parallel Cycle Simulation* in chapter 8.

Remark 6.3 Quantitative characteristics are introduced within P_C for allowing communication properties of real parallel architectures to flow into the *CP* model. For instance, time boundaries of special communication related events (see latency, gap, overhead within *LogP*) could be such characteristics. Another example is given by functions yielding run time estimations for communication processes in dependence of the number of involved processors, message lengths, network load situation and similar arguments. Contrary to the qualitative characteristics the quantitative ones are not intended to influence action based behaviour definitions.

7 Unrestricted Parallel Behaviour

The definition of *CP* behaviour will be based on given component behaviour (as action sequences) and synchronization conditions according to the qualitative characteristics of communication mechanisms described in P_C. At first we want do determine *unrestricted CP* behaviour omitting synchronization conditions specified. We start with some initial definitions:

- Let $\mathcal{B}_i$ be a given set of all behaviour sequences of P_i for a *CP* $\mathcal{P}$ $\left(\mathcal{B}_i \subseteq \mathcal{A}_i{}^+\right)$. We will skip to an "enriched" component behaviour for technical reasons. Thereby, we consider an action a within a component behaviour sequence s together with the corresponding processor index i and the position of a within s as "enriched" action. With $\mathcal{N}$ denoting the set of natural numbers we call

$$\overline{\mathcal{A}_i} = \mathcal{A}_i \times \{i\} \times \mathcal{N} \tag{7.1}$$

  the *enriched action set* of processor P_i. Enriched action sets of different processors within P_P are disjoint. In the following, $\text{act}(a)$, $\text{proc}(a)$ and $\text{pos}(a)$ denote the three components of actions $a \in \overline{\mathcal{A}_i}$. The *enriched component behaviour* $\overline{\mathcal{B}_i}$ with respect to P_i is built from $\mathcal{B}_i$ by skipping in every action sequence from actions to the corresponding "enriched" actions as schematically represented below:

$$\begin{gathered} s = (\ldots, a_k, \ldots) \in \mathcal{B}_i \\ \Downarrow \\ \overline{s} = (\ldots, (a_k, i, k), \ldots) \in \overline{\mathcal{B}_i} \end{gathered} \tag{7.2}$$

  Hence, $\overline{\mathcal{B}_i} \subseteq \overline{\mathcal{A}_i}^+$ is valid. We call

$$\overline{\mathcal{A}} = \bigcup_{i=1}^{n} \overline{\mathcal{A}_i} \tag{7.3}$$

  the *total enriched action set* with respect to $\mathcal{P}$.

- For any finite sequence $s = (s_1, \ldots, s_m)$ we define **expand(s)** as the set of all finite sequences $s' = \left(s'_1, \ldots, s'_{m'}\right)$ satisfying the following condition:

There exists a finite sequence $i = (i_1, \ldots, i_m)$ of immediately consecutive closed intervals $[1, n_1]$, $[n_1+1, n_2], \ldots \left[n_{m-1}+1, m^{'}\right]$ of natural numbers such that $s^{'}_k = s_j$ holds for $k \in i_j$.

For example, with $s = (0,1,0)$ we have $(0,0,0,1,1,0) \in$ **expand(s).** In every case $s \in$ **expand(s)** is valid. For sets S of sequences we define **expand(S)** $= \bigcup_{s \in S}$ **expand(s).**

The background of this definition is given by the multiplication of action occurences in consecutive snapshots of *CP* behaviour.

- Let M_i be n arbitrarily chosen sets with $i \in [1,n]$, $M = \bigcup_{i=1}^{n} M_i$ and $s \in \left(2^M\right)^+$, where 2^M denotes the power-set of M. Furthermore, let $s^i = (s^i_1, \ldots, s^i_m)$ be the maximum sub-sequence of s with $s^i_k \cap M_i \neq \emptyset$ for each component s^i_k of s^i.

  Then $\mathbf{proj}^i(s)$ (for $i \in [1,n]$) is defined as set of all sequences $s^* = (s^*_1, \ldots, s^*_m)$ with $s^*_k \in s^i_k \cap M_i$ for $k \in [1,m]$.

  For example, consider $M_1 = \{0\}, M_2 = \{1\}$ and $s = (\{0,1\}, \{0\}, \{1\})$. Then we have $s^1 = (\{0,1\}, \{0\})$, $s^2 = (\{0,1\}, \{1\})$, $\mathbf{proj}^1(s) = \{(0,0)\}$ and $\mathbf{proj}^2(s) = \{(1,1)\}$.

  The intention of this definition is to identify component behaviour within the system behaviour of a *CP*.

Definition 7.1 (UPB) *Let $\mathcal{P}$ be a* CP *model, $\mathcal{B}$ a family of sets $\mathcal{B}_i \subseteq \mathcal{A}_i{}^+$ of component behaviour sequences and $\overline{\mathcal{A}_i}$, $\overline{\mathcal{B}_i}$, $\overline{\mathcal{A}}$ defined as in* (7.1), (7.2) *and* (7.3)*, respectively. Then the set $\mathcal{U}^{\mathcal{B}}(\mathcal{P})$ of all sequences*

$$\overline{s} \in \left(2^{\overline{\mathcal{A}}}\right)^+$$

satisfying the following conditions is called ***Unrestricted Parallel Behaviour (UPB)*** *of $\mathcal{P}$ with respect to $\mathcal{B}$:*

1. $\overline{s}_k \neq \emptyset$ *for all component indeces k of $\overline{s}$*
2. $\left|\overline{s}_k \cap \overline{\mathcal{A}_i}\right| \leq 1$ *for all indeces k of $\overline{s}$ and $i \in [1,n]$*
3. $\emptyset \subset \ \mathbf{proj}^i(\overline{s}) \subseteq \mathbf{expand}(\overline{\mathcal{B}_i})$ *for all $i \in [1,n]$ with $M_i = \overline{\mathcal{A}_i}$*
4. $\overline{a} \in \overline{s}_k \wedge \overline{a} \notin \overline{s}_{k+1} \longrightarrow \overline{a} \notin \overline{s}_{k+l+1}$ *for all component indeces k of $\overline{s}$, $\overline{a} \in \overline{\mathcal{A}}$ and $l \in \mathcal{N}$*

Remark 7.1 The components of $\overline{s}$ are to be interpreted as maximum sets (snapshots) of simultaneous active actions on different processors. Condition 2 expresses that the contribution of each processor to such a set can be at most one action.

Remark 7.2 According to condition 3, within $\overline{s}$ a "line" concerning each processor can be found which is related to a possible "local" behaviour sequence. Due to condition 2, $\mathbf{proj}^i(\overline{s})$ supplies exactly one sequence of actions belonging to $\left(\overline{\mathcal{A}_i}\right)^+$ which has to be an expansion of an "enriched" component behaviour sequence belonging to processor P_i. An action of $\overline{\mathcal{A}_i}$ can occur in several consecutive snapshots. However, condition 4 prevents the existence of gaps between such phases. Remark that within an original behaviour sequence belonging to $\mathcal{B}_i$ consecutive components can embody the same action; in a corresponding "enriched" behaviour sequence all consecutive actions are different from each other due to the inclusion of the sequence position into the actions.

UPB represents a general framework which is to be restricted to concrete *CP* behaviour by inclusion of synchronization conditions according to communication mechanisms integrated in *CP*. In the next chapter such a restriction is performed leading to a definition of parallel cycle simulation on the basis of a collective communication mechanism.

8 Parallel Cycle Simulation

In the following we consider an arbitrarily chosen *PSHM*

$$M^{\Pi} = \left\{ M_{\Pi}^{\mathcal{C}} \mid \mathcal{C} \in \Pi \right\}$$

determined by a partition Π of a *SHM* M as introduced in definition 4.4. It is taken as basis for constructing a *CP* model $\mathcal{P} = (P_p, P_A, P_C)$. Then, the (parallel) behaviour of $\mathcal{P}$ based on the component behaviour of M^Π will comprise sequences of action sets we call *Parallel Cycle Simulation* with respect to M^Π.

Let us assume $\left|M^\Pi\right| = n$. We determine an ordering over M^Π by the introduction of component denotations $M_1, \ldots, M_n$. According to (5.1) each M_i is related to a set of abstract actions which is now called $\mathcal{A}_i$. By $\mathcal{B}_i$ we denote the set of all *Extended Sequential Cycle Simulations* (see definition 5.1) which belong to M_i $\left(\mathcal{B}_i \subseteq \mathcal{A}_i{}^+\right)$.

We set $P_p = \{P_1, \ldots, P_n\}$(a set of abstract processors) and $P_A = \{\mathcal{A}_1, \ldots, \mathcal{A}_n\}$. Furthermore, we want to introduce one communication mechanism $\mathcal{M}$ into $\mathcal{P}$ $(P_C = \{\mathcal{M}\})$. $\mathcal{M}$ does not depend on the concrete *PSHM* under consideration. It is related to the *mpc_index*-command belonging to the *Message Passing Library* of the *AIX Parallel Environment*. This command was used for the implementation of interprocessor communication at cycle boundaries during simulation with *parallelTEXSIM*. The qualitative characteristic of $\mathcal{M}$ in the framework of $\mathcal{P}$ is as follows:

- The only action engaged in $\mathcal{M}$ is the communication action c which is element of every action set $\mathcal{A}_i$.
- The whole processor set P_P is involved in $\mathcal{M}$. Each processor sends to each of the remaining processors individual messages (*all-to-all* personalized communication).
- $\mathcal{M}$ is a collective communication for which n actions c (one at each processor) have to synchronize.

For the definition of *Parallel Cycle Simulation* quantitative characteristics of $\mathcal{M}$ are not taken into account.

Definition 8.1 (PCS) *Let* $M^\Pi = \{M_1, \ldots, M_n\}$*be a* PSHM *with the corresponding family* $\mathcal{B} = \{\mathcal{B}_1, \ldots, \mathcal{B}_n\}$ *of sets of component behaviour sequences (*ESCS*s) and a* CP *model* $\mathcal{P} = (P_p, P_A, P_C)$ *constructed as above. The set* $\mathcal{R}^{\mathcal{B}}(\mathcal{P})$ *of all sequences* $\overline{s} \in \mathcal{U}^{\mathcal{B}}(\mathcal{P})$ *which satisfy the following condition is called* ***parallel behaviour*** *of* $\mathcal{P}$ *with respect to* $\mathcal{B}$*:*

There exists a sequence component $\overline{s}_k$ with $|\overline{s}_k| = n$ such that for all $a \in \overline{s}_k$ $act(a) = c$ is valid.

The elements of $\mathcal{R}^{\mathcal{B}}(\mathcal{P})$ are called ***Parallel Cycle Simulation (PCS)*** *with respect to M^{Π}.*

Remark 8.1 The restricting condition required in the definition above expresses the synchronization effect related to the (collective) communication action c. Note that we regard the simulation of exactly one cycle.

Lemma 8.1 *Under previous definitions for any PSHM M^{Π} we have $\mathcal{R}^{\mathcal{B}}(\mathcal{P}) \neq \emptyset$.*

Lemma 8.2 *Let $\overline{s}$ be a PCS with respect to M^{Π}. For each $i \in [1, n]$ sequences $\overline{s}^i \in \overline{\mathcal{B}_i}$ and $s^{*\,i} \in \overline{\mathcal{A}_i}^{+}$ with $\mathbf{proj}^i(\overline{s}) = \left\{s^{*\,i}\right\}$ and $s^{*\,i} \in \mathbf{expand}\ (\overline{s}^i)$ are unambiguously determined.*

According to (7.2) the action sequence

$$s^i = \left(\ldots, act\left(\overline{s}^i_k\right), \ldots\right) \tag{8.1}$$

built from $\overline{s}^i$ belongs to $\mathcal{B}_i$ and therefore is an *ESCS* (see definition 8.1). s^i identifies the behaviour of M_i within $\overline{s}$ and is denoted by $\mathbf{beh}^i(\overline{s})$ in the following.

Lemma 8.3 *Let $\overline{s}$ be a PCS with respect to M^{Π} and k be an index of $\overline{s}$ such that $|\overline{s}_k| = n$ and $act(a) = c$ for all $a \in \overline{s}_k$. Consider an arbitrarily chosen $\overline{a} \in \overline{s}_{\overline{k}} \cap \overline{\mathcal{A}_{\overline{i}}}$ with $act(\overline{a}) \neq c$, $\overline{i} \in [1, n]$ and $\overline{k}$ being an index of $\overline{s}$. For $act(\overline{a})$ lying in the cycle- or pre_comm-phase of $\mathbf{beh}^{\overline{i}}(\overline{s})$ (see definition 5.1) we have $\overline{k} < k$. For $act(\overline{a})$ lying in the post_comm-phase of $\mathbf{beh}^{\overline{i}}(\overline{s})$ we have $\overline{k} > k$.*

Parallel Cycle Simulation is visualized concerning a 3 processor variant in figure 3. Actions belonging to input-, output-, latch- and logical boxes are

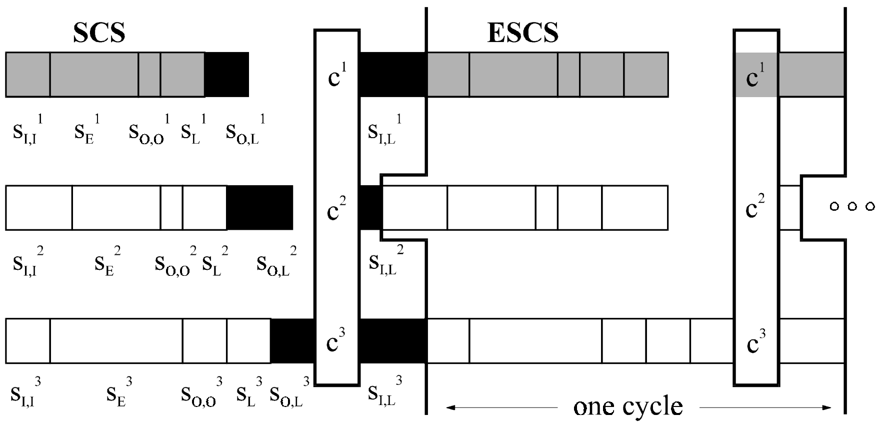

SCS
ESCS
$s_{I,I}^1$ s_E^1 $s_{O,O}^1$ s_L^1 $s_{O,L}^1$
$s_{I,L}^1$
$s_{I,I}^2$ s_E^2 $s_{O,O}^2$ s_L^2 $s_{O,L}^2$
$s_{I,L}^2$
$s_{I,I}^3$ s_E^3 $s_{O,O}^3$ s_L^3 $s_{O,L}^3$
$s_{I,L}^3$
c^1
c^2
c^3
one cycle

Figure 3: Parallel Cycle Simulation based on a model partition with 3 components

concentrated in separated areas. Two sub-sequences with *SCS* or *ESCS* structure, respectively, are shaded grey. Areas represented in black are related to pre- and post-communication sub-sequences. The synchronization effect of the special communication action c is emphasized by vertical bars.

For partition valuation, estimations of run time are assigned to action sequences. An estimation of cycle time can be expressed as follows:

$$t_{cycle} = \max_j \left(t^j_{I,I} + t^j_E + t^j_{O,O} + t^j_L + t^j_{O,L} + t^j_{I,L} \right) + t_{comm} \tag{8.2}$$

Thereby t_{comm} denotes the expected time for one collective communication at cycle boundaries. It is given by a function depending on the number of processors and the maximum length of a message that has to be sent between any two processors. This function is used as a quantitative characteristic of the communication mechanism $\mathcal{M}$ mentioned above within the *CP* model considered. The other time intervals occuring within (8.2) are inquired using average execution times of boxes belonging to certain classes (known from pre-simulation) together with structural model information.

9 Concluding Remarks

The framework of concepts developed here provides a formal basis for the construction, investigation and implementation of model partitioning algorithms. On the one hand the cone-related definitions allow to describe the partitioning subject exactly. They guide to the determination of data structures derived from Structural Hardware Models as for instance *Overlap Hypergraphs* and *Communication Graphs* which are frequently used in partitioning. On the other hand abstract modelling of *Parallel Cycle Simulation* supports partition valuation, thereby leading to a combination of load balancing and communication aspects. In its strongest form, partition valuation appears as performance prediction for corresponding parallel simulation processes. Early performance prediction [10] is one of the challenges of our future work.

Acknowledgements

Heartfelt thanks to Reiner Haupt, Thomas Villmann and Udo Petri for many valuable discussions and reviewing the manuscript.

References

[1] D. Culler, R. Karp, D. Patterson, A. Sahay, K. E. Schauser, E. Santos, R. Subramonian, and T. von Eicken. LogP: Towards a realistic model of parallel computation. *4th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, pages 1–12, 1993.

[2] D. Döhler. *Entwurf und Implementierung eines parallelen Logiksimulators auf Basis von TEXSIM.* Diplomarbeit, Universität Leipzig, Fakultät für Mathematik und Informatik, 1996.

[3] K. Hering. Partitionierungsalgorithmen für Modelldatenstrukturen zur parallelen compilergesteuerten Logiksimulation (Projekt). Technical Report 5(94), Universität Leipzig, Institut für Informatik, 1994.

[4] K. Hering, R. Haupt, and T. Villmann. Cone-basierte, hierarchische Modellpartitionierung zur parallelen compilergesteuerten Logiksimulation beim VLSI-Design. Technical Report 13(95), Universität Leipzig, Institut für Informatik, 1995.

[5] K. Hering, R. Haupt, and T. Villmann. Hierarchical strategy of model partitioning for VLSI-design using an improved mixture of experts approach. *Proc. of 10th Workshop on Parallel and Distributed Simulation*, pages 106–113, 1996.

[6] N. Manjikian. High performance parallel logic simulation on a network of workstations. Technical Report CCNG T-220, University of Waterloo, Department of Electrical and Computer Engineering and Computer Communications Network Group, 1992.

[7] R. B. Mueller-Thuns, D. G. Saab, R. F. Damiano, and J. A. Abraham. VLSI logic and fault simulation on general purpose parallel computers. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 12*, pages 446–460, 1993.

[8] W. Roesner. TEXSIM for loosely coupled multi-processors - performance estimates, sizing. IBM internal, 1993.

[9] W. G. Spruth. *The Design of a Microprocessor.* Springer, 1989.

[10] Z. Xu and K. Hwang. Early prediction of MPP performance : The SP2, T3D and Paragon experiences. *Parallel Computing 22*, pages 917–942, 1996.