An Integrated Concept for Design Project Planning and Design Flow Control

Michael Ryba Utz G. Baitinger
Stuttgart University, IPVR/ISE
Breitwiesenstraße 20 – 22
D-70565 Stuttgart, Germany
Michael.Ryba@informatik.uni-stuttgart.de Utz.Baitinger@informatik.uni-stuttgart.de

Abstract

This paper presents a novel approach supporting administrative tasks within the lifecycle of design projects. Based upon comprehensive models of design environments and design activities it combines techniques known from project management and mechanisms for design flow control. As a result it allows the planning, controlling and reviewing of design projects and supports algorithmic estimation of task durations and automatic computation of plan revisions.

1. Introduction

Although the design of integrated circuits is supported by a large variety of sophisticated CAE-tools based upon open, integrated CAE-systems, several problems have not been solved sufficiently up to now:

- Modern CAE-systems are so complex that administrating and even using them properly has become a problem by itself.
- Due to the complexity of modern integrated circuits their design has to be carried out by groups of cooperating engineers (design teams). The particular activities of the engineers involved in a design project have to be coordinated to guarantee the smooth handling of the entire project. Therefore all design tasks have to be carried-out according to well-defined design methodologies.
- Due to the increasing competition among elctronics companies, it is crucial to the development of new products to save time between the presentation of an idea for a new product and the product's introduction into the market (time to market). Therefore, the detailed planning of design activities based on precise estimation of time and costs is indispensable.

Quite obiously these problems can only be solved by providing tools within design environments, which support the administative tasks in the lifecycle of design projects. In particular the following problem areas have to be addressed:

- administration of complex design environments
- time and resource planning for design projects
- supervision and coordination of design activities
- project reviewing

A homogeneous, integrated solution for these problems requires a combination of techniques known from project management as well as mechanisms for design flow control. Furthermore sophisticated information models of the design environments and design activities have to provided.

1.1 Related Work

While techniques for project management are well known and widely used [6, 10, 11], mechanisms for design flow control are still subject to current research (e. g. [1, 2, 3, 5]; for a detailed discussion of existing systems see [9]). Especially no systems exist, which combine project planning and design flow control in an integrated solution.

1.2 Structure of this document

Within the PLASMA project an integrated concept has been developed, which meets the requirements stated above. The following two sections describe the two information models, which form the basis for this concept: the design environment model and the design activity model.

Based upon these information models a set of mechanisms has been developed, which support the planning, controlling and reviewing of design projects. The mechanims for these three problem areas are covered in sections 4 to 6, whereas section 7 focuses on how these mechanisms have been integrated into one homogeneous concept.

Section 8 briefly describes, how the mechanisms presented in this paper have been implemented. An overview of future extensions is given in section 9.

2. Design Environment Model

Design project planning as well as design flow control require information about the available design environment. Therefore an information model to represent the required information has to be provided.

To define such an information model, we first have to state what a design environment is. Quite obviously, a design environment is the set of all resources, which are directly involved in the design process. Thus an environment for the design of integrated circuits consists of hardware resources (computers, peripherals), software resources (CAD tools), human resources (i. e. the persons working on design projects) and data resources.

A comprehensive information model of the design environment must contain information about all these resources as well as information about the relations and dependencies exisiting between them. For all resources we basically need two kinds of information:

EURO-DAC '96 with EURO-VHDL '96
0-89791-848-7/96 $4.00

- characteristic features
- function within the design environment

While it is quite easy to describe the characteristic features of a resource, it is much more difficult to describe its function. The information model developed for the PLASMA system uses the concept of classification to describe the function of resources, i. e. classes of resources can be defined and inserted into a resource taxonomy. All resources within a class have the same characteristic features and therefore have the same function within the design environment.

As it is impossible to define one resource taxonomy, which is adequate for all design environments, PLASMA does not use a fixed taxonomy. The definition of the taxonomy is a part of the information model itself.

2.1 Description of Software Resources

To illustrate the main concepts used within the design environment model, an example description of CAE tools (software resources) is used. As stated above, it is impossible to define a fixed taxonomy which is suitable for all CAE-tools. Nevertheless there are criteria, which can be used for classification:

- the kinds of data produced or consumed by a CAE-tool
- the design operation performed by the tool

An example taxonomy, which is based upon a model of the design process described in [7], can be found in [9].

The taxonomy of software resources as well as the information about specific CAE-tools can be described using RDL (Resource Description Language). Figure 1 shows an example RDL description of three software classes (generator, simulator, circuit-simulator) and one CAE-tool (spice).

The description of software classes mainly consists of the definition of input and output data of the CAE-tools. The data types used to characterize the inputs and outputs (e. g. circuit-netlist) are also defined in the design environment model. In addition hardware requirements can be described by referencing an adequate hardware class and specifying additional constraints (see software class circuit-simulator).

The description of a specific CAE-tool contains all information necessary to invoke this program (e. g. command line arguments). The information which can be described using RDL is similar to the information covered by the CFI Tool Encapsulation Specification (TES; [4]), although the syntax is quite different. In addition to the information covered by TES, RDL can be used to describe the taxonomy of CAE-tools as well as information about the types of design data manipulated by the CAE-tools, available hardware resources and persons involved in the design process.

Figure 2 shows an EXPRESS-G diagram of the global structure of the information model, which forms the basis of RDL.

```
software-class generator
  inherit program
  output generic-out : design-data;
end software-class generator;

software-class simulator
  inherit generator
    rename generic-out as sim-out,
    redefine sim-out;
  input  sim-model : circuit-description;
  input  sim-stimuli : stimuli;
  output sim-out : simulation-output;
end software-class simulator;

software-class circuit-simulator
  inherit simulator
    redefine sim-model;
  input sim-model : circuit-netlist;
  hardware-requirements
    computer
      where (mem-size >= 8, specfp >= 10);
  end hardware-requirements;
end software-class circuit-simulator;

software spice is circuit-simulator
  redefine sim-model, sim-stimuli, sim-out;
  version "3e2"
  input sim-model : spice-netlist;
  input sim-stimuli : spice-cmd-file;
  output sim-out : spice-raw-file prefix "-r ";
  parameter no-init : boolean := false
    prefix "-r ";
  parameter term-type : string
    prefix "-t ";
  parameter batch-mode : boolean := false
    prefix "-b ";
  parameter server-mode : boolean := false
    prefix "-s ";
  parameter interactive-mode : boolean
    := true prefix "-i ";
  command-structure
    [no-init] [term-type] [sim-out]
    [batch-mode | server-mode |
     iteractive-mode]
    (sim-model | sim-stimuli)*
  end command-structure;
  environment-variable
    PATH := "/opt/spice/bin";
end software spice;
```

Figure 1: Example RDL description

3. Design Activity Model

The information about the static structure of a design environment contained in the information model presented in the previous section is not sufficient for planning and controlling design projects. Additional information about the dynamic aspects of design are needed, i. e. information about how the design environment is used within a design project has to be described.

Therefore information about the work flows within a design project have to be formally described. In particular two kinds of information are needed:

- all activities within design projects
- the dependencies between these activities

3.1 Basic Structure of Work Flows

PLASMA uses flow graphs to represent all kinds of work flows. The nodes of these graphs represent activities

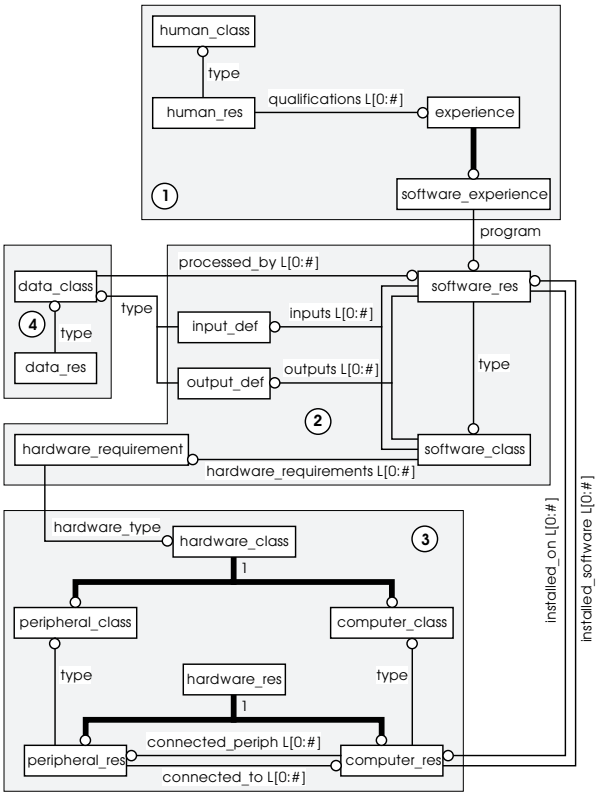

Figure 2: Design Environment Information Model

of arbitrary complexity. Each activity describes an operation, which produces output data from given input data and requires a set of resources to execute this operation.

By connecting the inputs and outputs of design activities arbitrary complex activities can be described. Based on the granularity of the operations PLASMA distinguishes between different kinds of work flows.

3.2 Design Methodologies and Design Processes

As stated in the introduction, design projects can only be successful, if all design tasks are carried out according to well-defined design methodologies. To enforce the accordance to these design methodologies they have to be formally described in a machine-processable form.

The PLASMA system distinguishes two kinds of design flows: design processes and design methodologies. Both are used to represent detailed descriptions of recurrent design tasks.

The description of design processes is based on the execution of specific CAE tools. Hence they describe concrete design flows, which can be used to control the processing of design tasks. Figure 3 shows a simple design process, which describes the actions which have to be carried out when the schematic of a circuit is entered.

Since design processes are directly based on specific CAE-tools they are not portable from one design environment to another. To ensure the portability of design flows an abstraction of design processes, called design methodology in PLASMA, is needed. Such design methodologies

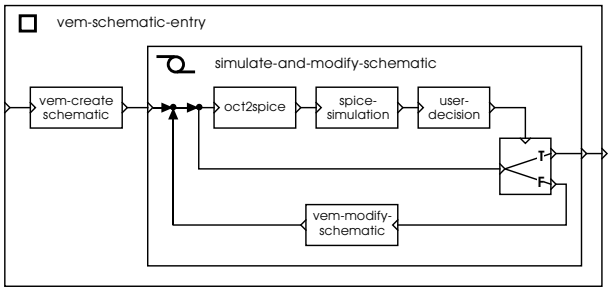

Figure 3: Example Design Process

describe design flows based on abstract design operations, which are derived from the taxonomy of CAE tools. Figure 4 shows the design methodology, which abstracts the design process in figure 3.

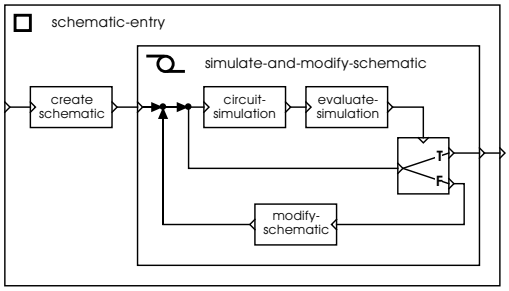

Figure 4: Example Design Methodology

To make the porting of design methodologies as easy as possible, PLASMA contains a tool which derives design processes from design methodologies semi-automatically.

3.3 Design Project Plans

In contrast to design methodologies and design processes, which describe recurrent design tasks, a project plan describes the unique structure of a single design project. It describes the partitioning of a project into subprojects and tasks. As project plans are used to compute the time plan for a project they must not contain cycles. Therefore the task nodes in a project plan are relatively coarse grained.

Since iterative techniques are typical for the design of integrated circuits, the tasks in a project plan are described in terms of design methodologies and design processes, which allow the description of design cycles. Figure 8 shows an example of a simple project plan.

4. Design Project Planning

The main task of project planning is to determine reliable information about how much time the project will take and which resources will be needed, before the project actually starts. In particular the following steps have to be taken:

- definition of project structure
- estimation of task durations
- time planning
- resource planning

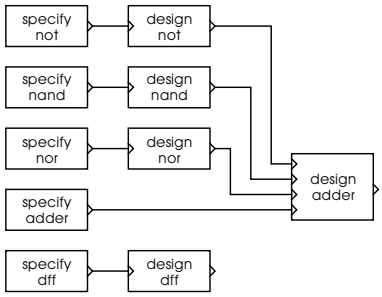

Figure 5: Example project flow graph

The following sections describe how these steps are performed in PLASMA using the two information models presented in the previous sections.

4.1 Defining the Project Structure

As described in section 3.3 design projects are defined using flow graphs. Each task node in this graph is linked to a design methodology and a design process, which define how the task has to be carried out. The resources needed for a specific task directly result from the design process used. Thus the selection of adequate resources can be performed automatically using the information contained in the design environment model. Nevertheless the project leader has the possibility to assign resources by hand, which is especially senseful for the selection of human resources (designers). The result of this first step is a so-called project flow graph as shown in figure 5.

4.2 Estimating Task Durations

To compute a time plan for a design project the duration of all tasks has to be estimated. This is usually done manually, i. e. by asking an experienced designer. PLASMA however uses a different strategy. As all tasks are formally defined by design processes, estimates for the task durations are computed using this knowledge.

The design activity model contains estimation functions for each atomic design step (i. e. tool run). The input of these functions are parameters characterizing the complexity of input data (e. g. the number of logic gates in a netlist) and parameters describing the performance of the resources used (e. g. specFP rate of a computer). Furthermore estimation functions exist, which compute the expected complexity of the data produced by an atomic design step.

In most cases these estimation functions cannot be derived analytically. Therefore they are derived from empirical data using statistical methods (see section 6).

Using these functions the time consumption and expected output complexity of each atomic design step in a design process can be estimated. The overall time consumption of a task, i. e. the whole design process, can then be computed by finding the maximum cost path in a directed graph.

4.3 Time and Resource Planning

Based upon the estimated task durations it is now possible to compute the time and resource plan for the design project. For this computation the project flow graph is transformed into a network diagram. The temporal dependencies between the task nodes in the resulting project network are directly derived from the data dependencies.

The network diagram is then used to compute an initial time plan, which does not take resource constraints into consideration. Starting from this initial time plan resources are reserved for the project to be planned. If a needed resource cannot be reserved for the time interval computed for a task, PLASMA's planning algorithm first tries to solve the problem locally by using an adequate other resource. If this is impossible, too, several actions have to be carried out:

- PLASMA queries the design environment model, when an adequate resource will be available
- a special allocation task with a fixed start date is inserted into the network diagram
- a dependency between the task requiring the resource and the allocation task is inserted
- a new time plan is computed

The iteration consisting of resource reservation, modification of the network diagram and time planning is repeated until all required resources have been successfully assigned. For the computation of the initial time plan as well as the time plans containing resource constraints a modified version of the Metra-Potential-Method (MPM; [6]) is used.

5. Design Flow Control

The time and resource plan of a design project are directly used to control the execution of the project. To guarantee the smooth handling of the entire project the following mechanisms are needed:

- observation of project progress
- activation of design tasks
- execution of design processes

Furthermore the whole course of the project is logged to gather information needed to review the project and draw conclusions for future projects.

5.1 Supervising Project Progress

Although the described mechanisms for project planning deliver accurate time and resource plans for a project, it is quite unusual that these plans can be kept during project execution. Typically two kinds of problems lead to violations of the project plan:

- task durations have been estimated too optimistic or pessimistic
- reserved resources cannot be allocated because they are either damaged or blocked by other tasks which are out of schedule

As these problems invalidate the existing project plan it is necessary to revise the plan whenever such problems occur. Therefore the project progress has to be monitored continuously.

PLASMA uses an algorithm based on fixed time intervals, which are derived from the granularity of the time plan, to do this. At each time step this algorithm checks, whether all tasks which should have been started are actually being worked on. If the early start of a task has elapsed, those tasks that prevent the task under considera-

tion to be started, are computed and the responsible designers are notified. If the late start of a task has elapsed the project leader is informed and a plan revision will be computed as soon as possible.

While this algorithm only monitors the progress of the project, the information contained in the project plan can also be used to actively control the course of the project. This is done by an event-driven scheduling system, which works on two levels of granularity: the task level and the design step level.

5.2 Controlling Design Tasks

A task can be carried out if and only if all required input data is available. This situation can only occur, if another task has been successfully completed. Whenever this situation occurs, PLASMA's macro scheduling algorithm computes all tasks, which depend on the task that has just ended. If all input data for one of these tasks exist, a dispatcher process for the task (macro dispatcher) is started.

This dispatcher tries to allocates all resources reserved for the task. If this is impossible the algorithm tries to allocate alternative resources. If this is impossible, too, a plan revision has to be computed and the task is blocked until all resources become available.

After all resources have been allocated successfully the design process associated with the task is executed.

5.3 Controlling Design Steps

Design processes are executed using a similar scheduling algorithm as described above, the so-called micro scheduler. Like tasks, design steps can only be carried out if all required input data exists. Therefore the events used by the micro scheduler are based on the termination of design steps.

Whenever a design step ends, those design steps using its output data are computed and it is checked whether they become executable. In this case a dispatcher process for the design step (micro dispatcher) is started.

As all required resources have already been allocated on the task level, the micro dispatcher only has to check, whether the resources allocated are still functional. If problems occur the micro dispatcher performs the same actions as the macro dispatcher.

Using the described two-level scheduling and dispatching system PLASMA is able to determine violations of the project plan as early as possible and to adapt the time plan to the changed situation. Furthermore the automatic scheduling of design processes guarantees that all tasks are carried out according to the recommended design methodologies.

6. Reviewing Design Projects

During project execution various kinds of information are written to a project log. The log includes for example:

- estimated and actual duration of design tasks and design steps
- estimated and actual load of resources
- problems that led to plan revisions

These informations can be used as a detailed documentation of the course of the project. But they form a valuable source of information for the planning of future projects, too. For example they can be used to construct estimation functions for the duration of design steps.

But to make the best use of them, they have to be post-processed, analysed and interpreted. Therefore the final version of PLASMA will contain tools, which support these tasks. In particular tools for the following operations are needed:

- graphical display of resource load protocols
- statistical analysis of the duration of design activities related to the complexity of the design data processed
- deriving functions for the estimation of time consumption and output complexity of design data
- tracking down problems, which led to plan revisions

To accomplish some of these tasks existing programs (e. g. for statistical analysis) can be used.

7. Integrated Planning, Control and Review of Design Projects

In the previous sections a set of mechanisms, which support the administrative tasks in the three phases of a design project have been described briefly. But the overall power of the concept realized in PLASMA only becomes evident, if we have a closer look on how these mechanisms cooperate. Figure 6 illustrates how the components of PLASMA interact.

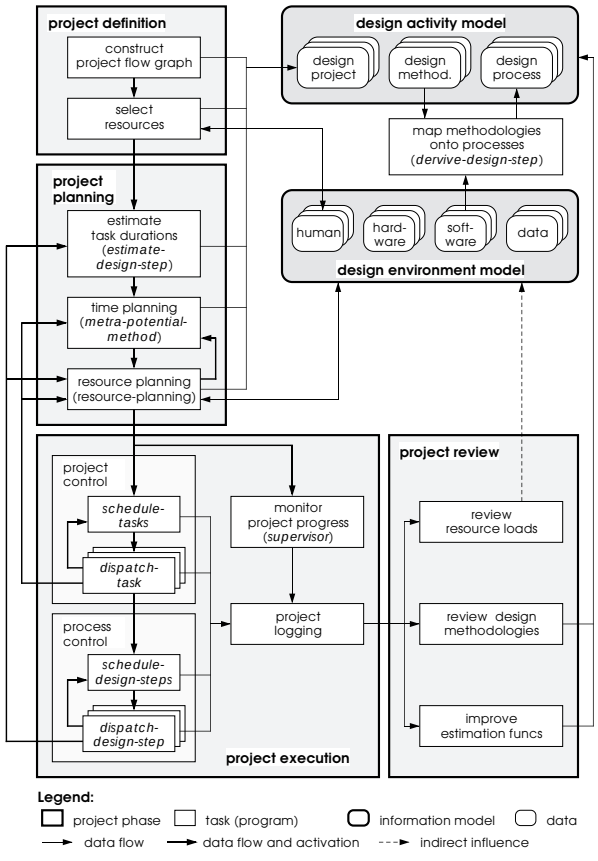

Figure 6: Integration of described mechanisms

From this figure it becomes obvious that PLASMA consists of two nested control loops. The inner one consists of project planning and project execution and enables the automatic computation of plan revisions. The outer loop consists of project planning, project execution and project reviewing, because the information gained by reviewing a project has a direct influence on the design activity model and hence an indirect influence on project planning.

8. Current State of Implementation

All mechanisms presented above have been implemented in a prototype, which consists of four main components:

- CARMA (Computer-Aided Resource Management) provides all tools to construct and maintain the design environment model.
- CAPPLAN (Computer-Aided Project PLANning) contains all tools to construct and maintain the design activity model as well as for planning design projects.
- CAPEX (Computer-Aided Project Execution) provides mechanisms for supervising, controlling and logging design projects.
- CAPA (Computer-Aided Project Analysis) contains utilities for analysing project logs.

Although the PLASMA components have been designed to be implemented as cooperating programs, which interact in a client-server style, the current prototype is a monolithic application. To gain quick and reproducible experimental results, this prototype can be used to simulate design projects.

9. Future Work

The PLASMA prototype showed that the described concepts can be implemented. Furthermore, it became obvious that the integration of project planning and controlling a project's execution has several advantages over the existing separation of these problem areas. Therefore we are currently planning to integrate the PLASMA mechanisms into an existing design environment to verify the experimental (simulated) results in the real world.

Furthermore we are working on several extensions of the concepts presented in this paper, which are mostly related to PLASMA's planning strategies. These extensions include:

- supporting successive refinement of the project plans („evolutionary planning")
- integrate mechanisms for time, resource and cost optimization

Another interesting topic is the adaptation of the concepts described in this paper to other application areas like software engineering or production planning and control.

10. Conclusion

Starting from the increasing administrative problems in the field of electronic circuit design, a new concept supporting administrative tasks throughout the whole lifecycle of design projects has been presented. This concept closes the gap between project planning and design flow control. As a result, planning and controlling form a closed control loop. This concepts shows several advantages compared to the existing separation between project planning and project execution:

- simplified selection of resources and detailed resource planning based on a comprehensive model of the design environment
- algorithmic estimation of task durations enabled by formal descriptions of design processes
- project execution control through direct use of project plans
- automatic computation of plan revisions, which may become necessary due to problems during project execution
- easier inclusion of experiences from former projects and tuning of planning mechanisms by analysing project protocols

The only disadvantage lies in the fact that the information contained in the models of the design environment and the design activities has to be entered into the system. But despite of occasional changes this has to be done only once and is supported by special tools.

11. References

[1] W. Allen, D. Rosenthal, K. Fiduk: The MCC CAD Framework Methodology Management System; Proceedings 28th ACM/IEEE Design Automation Conference (DAC), 1991

[2] K. O. ten Bosch, P. Bingley, P. van der Wolf: Design Flow Management in the NELSIS CAD Framework; Proceedings 28th ACM/IEEE Design Automation Conference (DAC), 1991

[3] J. B. Brockman, S. W. Director: The Hercules CAD Task Management System; Proceedings IEEE International Conference on Computer-Aided Design (ICCAD), 1991

[4] CAD Framework Iniative: Tool Encapsulation Specification, Version 2.0; CAD Framework Iniative, 1995

[5] E. Kupitz: Design Assistance in Concurrent Integrated Environments; in [Rhyne 92], 1992

[6] K. Neumann: Operations Research Verfahren, Bd. III – Graphentheorie/Netzplantechnik; Carl Hanser Verlag, München - Wien, 1975

[7] F. J. Rammig: Systematischer Entwurf digitaler Systeme; Verlag B. G. Teubner, Stuttgart, 1989

[8] T. Rhyne (Hrsg.): Electronic Design Automation Framework; Elsevier Science Publishers B. V. (North Holland), 1992

[9] M. Ryba: Planung und Durchführung methodischer Entwurfsaktivitäten; Verlag Dr. Kovac, Hamburg, 1996

[10] J. Schwarze: Netzplantechnik – Eine Einführung in das Projektmanagement; Verlag Neue Wirtschafts-Briefe, Herne - Berlin, 1990

[11] H.-J. Zimmermann: Netzplantechnik; Verlag Walter de Gruyter, Berlin - New York, 1971