This class introduces the student into the area of parallel programming. The emphasis is on state-of-the-art programming models, languages and tools, rather than on the conceptual problems and solutions of parallel and distributed processing. A certain level of familiarity with the latter is assumed in the course. The lecture is three hours per week, and is being offered as a special course in Computer Architecture for students in the third or fourth year of their studies (Hauptdiplom). A lab course in parallel programming is recommended to be visited in conjunction with this class.

The lecture begins with an introduction of parallel computers and parallel programming, including problems specific to parallel processing. Data and functional parallelism, the design process and the life cycle of parallel programs are outlined.

The first major part is on parallel programming systems. Parallel programming languages are introduced, with an emphasis on High Performance Fortran (HPF) and Compositional C++ (CC++). For message-passing programming, communication libraries are presented, in particular the Message Passing Interface (MPI) and Parallel Virtual Machine (PVM). A thread library is introduced next as an environment for shared-memory parallel programming. Finally, parallel I/O is dealt with.

The focus of the second part is on tools for the latter phases of parallel program design, i.e. debugging, performance analysis, visualization of dynamic program behaviour, and dynamic load balancing. The tools developed at our Chair over the last years, DETOP, PATOP and VISTOP are described in terms of their concepts, implementation and use. Load balancing strategies are also presented.

### 7.1.4 Industrial Project Management (Industrielle Projektplanung und -durchführung)

*by Hermann Hellwagner*

Based on the experiences made by the lecturer in an industrial research lab, an overview of project planning and management was given. Project management principles and methods were described in a systematic way, and major case studies from industry were presented to illustrate factors for success or failure of large-scale projects.

### 7.1.5 Interconnect Structures (Verbindungsstrukturen)

*by Harald Richter*

The lecture 'Verbindungsstrukturen für parallele und verteilte Systeme' (Interconnect Structures for Parallel and Distributed Systems) deals with the theory and practice of networks. It takes 2 hours per week and each student after the Vordiplom can be examined on request in this topic. It is intended for people from Electrical Engineering or Computer Science with basic knowledge in (parallel) computer architectures. The main topics that are covered are static and

dynamic networks in respect of their topologies, routing schemes and underlying mathematical theories. As commercial examples, the networks of the Cray T3D/E, IBM SP-2, Convex Exemplar, Fujitsu VPP 500, and Hitachi SR2001/2201 are presented.

## 7.2 Classroom Exercises

The lecture *Technische Grundlagen der Informatik (TGI)* is supported by an additional classroom exercise held by Markus Leberecht. The material covers all topics of the original lecture (switching functions, automata, microprogramming, and basics of electrical engineering) by providing exercises that can be performed by the students themselves. Help is given as a means to prepare for the upcoming exam.

## 7.3 Laboratories

### 7.3.1 Parallelrechner-Praktikum

*by Arndt Bode, Hermann Hellwagner, and Thomas Ludwig*

The parallel computer laboratory course describes with architectural considerations, operating system functionality and explanations to compilation techniques, the interdependence of parallel systems components. This course builds mainly upon background in computer architecture and operating systems.

The main objective of this lab is the familiarization with the efficient usage of multiprocessor systems with shared and distributed memory organizations. The principles of parallel processing will be discussed and problems in designing, testing and optimizing parallel programms will will be dealt with in form of projects. Scientific research issues, as currently discussed in debugging, performance measuring, monitoring, visualizing, fault-tolerance or load balancing, offer interested students a wide range of areas to get involved into, e.g., in form of thesis work.

### 7.3.2 Microprocessors and Systems Design Laboratory

*by Arndt Bode, Wolfgang Karl and Markus Leberecht*

The microprocessor laboratory is part of the curriculum for any student who wants to become involved with the technical side of computer science. The course is divided in several parts, comprising every phase of microprocessor systems design and the programming of microprocessor systems.

Each student group (2 - 3 students) has to design and build a substantial project. Modern design tools, e.g. hardware description languages like VHDL, Verilog simulation tools, schematic capture, field programmable gate arrays, as well as evaluation boards and in-circuit emulators