Welcome to Concurrency in System Design, and welcome to Aizu-Wakamatsu!

The first edition of any scientific gathering is always a source of emotions for the participants: we wonder how many new interesting people and how many old friends and colleagues we will have a chance to meet, how interesting (or boring) we will find the sessions, and how many new topics or areas for research we will discover.

In this case, CSD was born exactly with these objectives in mind. The organizers and the Program Committee felt that there was a deep need to introduce novelty into two related areas: concurrency theory and system design. The goal, stated in the first call for papers, was to bring together two fairly different and partially separated groups of people. On the one hand there are researchers who, due to their formation and background, have developed theory and algorithms to model, analyze, synthesize, verify and so on large, complex systems. On the other hand, there are researchers who have problems coping with the design issues of such systems every day. We think that it is only by fostering or building a long-lasting working relationship between broader and broader portions of these groups that the solutions proposed by the former can be tested and put to work, and the problems posed by the latter can be effectively solved. We hope that the technical and tutorial program that we have put together with the help of a very long list of people (only some of whom are listed below for reasons of space) is a good step in that direction.

As a contribution to the further advancement of knowledge and application in the field, we would like to ask all of the participants to consider and discuss how well the stated goal has been achieved (or why it is either irrelevant or not achievable), and how the effectiveness of future editions of CSD can be improved. Part of the conference program will be devoted, in the form of panels and group discussions, to this purpose of setting the pace of CSD for the future. We would also like, in order to facilitate the starting of this discussion, to outline several areas in which this synergy between theory, algorithms and problems can be particularly effective today.

Timing has been a long-standing problem in system-level analysis (not to mention synthesis), because its introduction in a model immediately causes an explosive growth of the complexity of almost every algorithm that can be applied to it. On the other hand, timing, performance, Quality of Service, and so on, are some of the most important characteristics of any kind of human-designed system, and at the same time are some of the most difficult to satisfy.

In our opinion, theoretical researchers can gain a lot by being exposed to the often informal and ad-hoc techniques that designers use in this area to solve the complex problems that they have to face. This would hopefully help theoreticians to learn how to construct simplified models that can be used to effectively tackle otherwise prohibitively complex problems.

The dichotomy between state-based and event-based models is also another promising area for interesting development. The former have almost always been used in a synchronous context and the latter have almost always been used in an asynchronous context. The reasons for this are not clear, but the problems caused by this dichotomy are clear; for example, in the area of synchronous circuit design: synthesis models are state-based, while simulation models are event-based. This currently causes endless pain, in terms of mismatch between the two, difficulties in optimizing tool performance, and so on. Hence in this area an application of recent interesting results building a bridge between the two domains can be extremely beneficial.

Modularity, hierarchy and reuse are traditionally considered to be the principal means to bridle complexity at all levels in system design. However, it is still not clear — even at the theoretical level — how to truly apply them. One would like to be able to select useful "blocks" from a library of pre-design components, evaluate how well they fit the required specifications, and then seamlessly integrate them together, building new re-usable blocks. However, we are still
very far from this goal since every module, be it hardware, software or any other piece of Intellectual Property, must still be understood in detail by a designer in order to be profitably used. In particular, interface abstraction is still a far goal in most areas (with two noticeable, but fairly special-purpose exceptions in the areas of Boolean gates and operating system services). While one would like to know only that two modules when used together perform all the required functions, one still has to look at their implementation in detail in order to interface or coordinate them properly.

This list, by no means exhaustive, is just an example of the kinds of issues and problems that are dealt with by the papers and tutorials that we have selected for this first edition of CSD. We are counting on the help of the participants, the Technical Committee, and all of the researchers and practitioners in the area to extend and discuss the list, to effectively tackle the issues, and to make CSD an even more successful event.

In the meantime, we would like to thank the Technical Program Committee and all the people and organizations who have helped with this first edition. Among them, our special thanks go to Fukushima Prefecture for the financial and organizational support, to Yuko Kesem and Kazuaki Yamauchi for the local arrangements, and to Regina Spencer Sipple for the publications.

**General Chair:** Shoichi Noguchi

**Program Co-Chairs:** Luciano Lavagno, Wolfgang Reisig

**Conference Co-Chairs:** Sadatoshi Kumagai, Alex Kondratyev