PUTTING SYNTHESIS

MANUEL D’ABREU
GE Corporate Research and Development

Synthesis is emerging as the new frontier in CAE/CAD. In this issue, we address architectural level and layout level synthesis (see figure below). Architectural, or high-level, synthesis addresses the need of the system designer, who is concerned with algorithm design and mapping the algorithm onto hardware. High-level synthesis, which until now was used only in industrial laboratories and universities, is evolving from its embryonic state to practical use in product design.

Commercial products now address logic-level synthesis, which is rapidly gaining user acceptance. Logic synthesis systems manipulate a Boolean-level or equivalent representation of the design. The role of synthesis at this level is to minimize the logic and optimize the design in terms of area and performance.

Our issue reports on work done at two industrial laboratories and at a European university. A number of universities in the US have been very active in high-level synthesis research, which has provided the enabling technology for current synthesis developments. The systems described in this special issue have been used or will be used soon in design projects.

System specification is currently done using programming languages. It is hard, if not impossible, to describe hardware requirements with this type of language. The difficulty lies in describing

<table>
<thead>
<tr>
<th>Synthesis Stage</th>
<th>Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Develop high-level algorithm</td>
<td>• High-level design language</td>
</tr>
<tr>
<td>Optimize architecture</td>
<td>• High-level simulation</td>
</tr>
<tr>
<td>Implement structure</td>
<td>• Resource sharing</td>
</tr>
<tr>
<td>Generate modules and layout</td>
<td>• Operator scheduling</td>
</tr>
<tr>
<td>Layout level</td>
<td>• Pipeline optimization</td>
</tr>
<tr>
<td>Architectural level</td>
<td>• Operator selection</td>
</tr>
<tr>
<td></td>
<td>• Area/speed tradeoffs</td>
</tr>
<tr>
<td></td>
<td>• Module compilers</td>
</tr>
<tr>
<td></td>
<td>• Layout verification</td>
</tr>
</tbody>
</table>

The stages and tasks of synthesis at the architectural and layout levels.
the parallel nature of hardware. A good hardware design language is essential to the synthesis process, and the V language developed at IBM T.J. Watson Research Center is one of the first languages to address the special needs of hardware synthesis. The article by V. Berstis describes the importance of having well-defined semantics so that valid transformations can be defined to synthesize the hardware. The importance of a well-defined language is also stressed in the article by F. Catthoor et al. on the Cathedral-II synthesis system. The FACE system, described by A. Casavant et al. in our third article, reports on still more positive results of using an application language as the hardware description language.

Several common themes are evident in these articles. For one, all the authors have discovered that general synthesis techniques are inefficient. Targeting synthesis tools to a particular hardware architecture and focusing the tool to a particular technology increases the chance for success. The technology targeted in the Cathedral-II and FACE systems is digital-signal-processing systems. Catthoor et al. discuss how having a well-defined target architecture eases the synthesis of testable hardware.

Another common result from these synthesis efforts is the realization that synthesis needs to be an interactive process, or at least user guided. Experience with the FACE system, for example, shows that the users want to have control over critical decisions during synthesis. Cathedral-II uses the concept of pragmas to help guide the synthesis process, while the FACE system provides a graphical user interface. The V language allows the user to annotate the design description to achieve better design results.

I expect high-level design tools to become quite popular once the commercial CAE/CAD companies embrace the technology. In the meantime the number of believers are few and a lot of missionary work lies ahead of us to make high level synthesis a reality. For those interested in reading more about synthesis and design languages, I have included a list for more reading. It is not too early to begin anticipating the challenges these new developments will bring.
ACKNOWLEDGMENTS

Getting this issue together required a lot of dedication on the part of our authors. I thank them for their diligence and quick response to our deadlines. I also thank the referees for their suggestions and timely response.

FOR FURTHER READING
