These methods are still developing and should begin to have practical value within several years. The extent to which traditional microprogrammers will turn to formal methods depends in part on the complexity of the microcode and the underlying microarchitecture. We suspect that formal firmware design methods will become important in cases where complex applications are migrated down to highly concurrent, high-performance microarchitectures. We also expect the results of the research described here to influence formal design methods for current and future architectures.

Acknowledgments

This article was considerably improved by the critical comments of many individuals, particularly Subrata Dasgupta and Werner Damm. Thanks also to Vicki Allan, Lothar Kreft, Phil Sweany, Joe Varghese, Steve Vegdahl, Alan Wagner, and many anonymous referees.

We gratefully acknowledge the support of the Abell Center for Computer-Assisted Engineering, the Evans and Sutherland Corporation, and the National Science Foundation. This work was supported in part by National Science Foundation grants MCS-8107481 and DCR-8503941 and by the Evans and Sutherland Corporation.

References


Robert A. Mueller is a guest editor of this issue. His biographical sketch appears on page 5.

Michael R. Duda is a doctoral candidate in computer science at Colorado State University. Previously, he was a software engineer at Texas Instruments and a test engineer at Monolithic Memories. His research interests include microprogramming, languages and compilers, artificial intelligence, signal processing, and computer architecture.

Duda received a BS in electrical engineering from Worcester Polytechnic Institute in 1975 and an MS in electrical engineering from Colorado State University in 1979. He is a student member of the ACM and the IEEE.

Readers may write the authors at Colorado State University, Computer Science Dept., Fort Collins, CO 80523.