Chez Scheme Bibliography

September 9, 1994

[1] R. Kent Dybvig and Bruce T. Smith. *Chez Scheme Reference Manual, Version 1.0.* Cadence Research Systems, Chapel Hill, North Carolina, May 1985. *Precursor to [3].*

[2] R. Kent Dybvig, Daniel P. Friedman, and Christopher T. Haynes. Expansion-passing style: Beyond conventional macros. In *Proceedings of the 1986 ACM Conference on Lisp and Functional Programming*, pages 143–150, 1986. *Preliminary version of [6].*

[3] R. Kent Dybvig. *The Scheme Programming Language.* Prentice-Hall, 1987. *Brief introduction and reference manual for Scheme with some Chez Scheme specifics. Currently being revised.*

[4] R. Kent Dybvig. *Three Implementation Models for Scheme.* PhD thesis, University of North Carolina, Chapel Hill, April 1987. *Chapter 4 describes the essence of the Chez Scheme Version 1 run-time architecture.*

[5] R. Kent Dybvig and Robert Hieb. A variable-arity procedural interface. In *Proceedings of the 1988 ACM Conference on Lisp and Functional Programming*, pages 106–115, July 1988. *Preliminary version of [10].*

[6] R. Kent Dybvig, Daniel P. Friedman, and Christopher T. Haynes. Expansion-passing style: A general macro mechanism. *Lisp and Symbolic Computation 1*, 1, pages 53–75, 1988. *Describes the low-level macro mechanism provided by Chez Scheme Versions 2–4.*

[7] R. Kent Dybvig and Robert Hieb. Engines from continuations. *Computer Languages 14*, 2, pages 109–123, 1989. *Describes the continuation-based engine implementation used by Chez Scheme.*

[8] R. Kent Dybvig, Robert Hieb, and Tom Butler. Destination-driven code generation. Technical Report 302, Indiana Computer Science Department, February 1990. *Describes the basic mechanism used by Chez Scheme's code generator to generate code from abstract syntax trees.*

[9] Robert Hieb, R. Kent Dybvig, and Carl Bruggeman. Representing control in the presence of first-class continuations. In *Proceedings of the SIGPLAN '90 Conference on Programming Language Design and Implementation*, pages 66–77, June 1990. *Describes Chez Scheme's segmented stack representation of continuations.*

[10] R. Kent Dybvig and Robert Hieb. A new approach to procedures with variable arity. *Lisp and Symbolic Computation 3*, 3, pages 229–244, September 1990. *Describes an alternative to the "dot" notation for defining procedures that accept variable numbers of arguments; only the multiple-case notion is adopted in Chez Scheme (as case-lambda).*

[11] R. Kent Dybvig. Writing hygienic macros in scheme with syntax-case. Technical Report 356, Indiana Computer Science Department, June 1992. *Introduces the syntax-case macro system through a series of examples. Some overlap with [13].*

[12] R. Kent Dybvig, Carl Bruggeman, and David Eby. Guardians in a generation-based garbage collector. In *Proceedings of the SIGPLAN '93 Conference on Programming Language Design and Implementation*, pages 207–216, June 1993. *Introduces guardians and describes the implementation of guardians and weak pairs in Chez Scheme.*

[13] R. Kent Dybvig, Robert Hieb, and Carl Bruggeman. Syntactic abstraction in Scheme. *Lisp and Symbolic Computation 5*, 4, pages 295–326, 1993. *Describes the syntax-case macro system and its implementation and argues why it is superior to previous attempts at hygienic macro systems.*

[14] R. Kent Dybvig, David Eby, and Carl Bruggeman. Don't stop the BiBOP: Flexible and efficient storage management for dynamically-typed languages. Technical Report 400, Indiana Computer Science Department, March 1994. *Describes Chez Scheme's storage management system.*

[15] J. Michael Ashley and R. Kent Dybvig. An efficient implementation of multiple return values in Scheme. In *Proceedings of the 1994 ACM Conference on Lisp and Functional Programming*, pages 140–149, June 1994. *Describes Chez Scheme's implementation of the Scheme multiple return values interface.*

[16] Cadence Research Systems, Bloomington, Indiana. *Chez Scheme System Manual, Rev. 2.4*, July 1994. *Describes various aspects of Chez Scheme not covered in [3].*