Communicating Process Architectures 2007
Alistair McEwan, Steve Schneider, Wilson Ifill and Peter Welch (Eds.)
IOS Press, 2007

Compiling occam to C with Tock

Adam T. SAMPSON
Computing Laboratory, University of Kent
Canterbury, Kent, CT2 7NZ, UK
ats@offog.org

Abstract. Tock is a new occam compiler from the University of Kent, the latest result of many years' research into compiling concurrent languages. The existing occam compiler generates bytecode which is then translated into native instructions; this reduces opportunities for native code optimisation and limits portability. Tock translates occam into C using the CIF concurrent runtime interface, which can be compiled to efficient native code by any compiler supporting the C99 language standard. The resulting programs combine the safety and featherweight concurrency of occam with the performance and portability of C. Unlike previous attempts at translating occam to C, Tock's output resembles hand-written CIF code; this eases debugging and takes better advantage of the C compiler's optimisation facilities. Written in the purely functional language Haskell, Tock uses monadic combinator parsing and generic data structure traversal to provide a flexible environment for experimenting with new compiler and language features.

Keywords: occam, C99, Haskell, CIF, compilers, concurrency, optimisation