Abstractions for Real Real-time Systems

Mathai Joseph
Tata Consultancy Services
m.joseph@tcs.com

Abstract

From the earliest days, designers of real-time systems have had to decide how to manage the hardware-level mechanisms (such as interrupts and multi-level priorities) that have been provided to share the computing resources between competing demands. One approach is to ignore these mechanisms and use software techniques to schedule program segments; another is to align the software architecture with the hardware features, e.g. by dividing a program into tasks or processes that are scheduled dynamically according to hardware priorities. These are two of many possible engineering solutions to the problem of meeting the timing constraints of real-time programs. Looked at more formally, the problem is to prove that the implementation of the program, by whatever method is chosen, can be shown to satisfy its timing specifications. This is altogether harder to accomplish. Proofs at the program level require guarantees of timing properties at lower levels. If these guarantees are provided independently of the program structure, the guarantees are easier to provide but the design flexibility available at the program level is highly limited. An alternative is to treat timing properties as refinements of an un-timed program [1] and to compute guarantees over the timed version of the program. Another approach is to interpose a layer between the real-time program and its implementation [2]. However, it may be possible to get more effective use of the system resources if the program can directly exercise some control over the use [3]. This paper examines various abstractions over the mechanisms for timing in real-time programs.