System-level compilation of parallel real-time systems

Abstract: Our work is at the frontier between real-time scheduling and compilation. Our objective is to build parallel software that respects hard real-time requirements. Historically, the construction of real-time systems relied on the use of very abstract models - known as task models - of both software and the execution platform. Building a task model involves high abstraction overheads and is seldom, if ever, done soundly. Once the task model is built, its relatively small size enabled the use of "exact" constraint solving techniques. But this solution does not scale to the size and complexity of next-generation embedded software and hardware.

Based on previous work in the AOSTE team, our objective was to apply a compilation-like approach to this implementation problem using strong hypotheses on the capabilities of the execution platform and on the form of input specifications. We rely on low-complexity allocation and scheduling heuristics based on list scheduling. This low complexity allows the use of precise models of the platform and executing software, detailing aspects such as synchronization, memory coherency, and timing. Code synthesis covers all aspects of code generation: allocation, scheduling, the construction of thread code, and the construction of linker scripts.

The resulting compiler carefully integrates multiple tools - dataflow compiler, mapping and code generation tool, C compiler and linker, static analysis tool.