Modeling Timing Aspects of Software of a Combustion Engine Controller System

Manuel Huber

An important requirement of Real-Time Embedded Systems is not only to achieve correct system behavior depending on the logical results of the computations, but also to meet timing constraints during execution. In the Automotive Domain, this non-functional requirement is essential to guarantee a certain behavior for reasons of safety, which otherwise can cause severe damage and even cost human lives. In this context one talks about Hard Real-Time Systems, and in particular, this timing condition can mean to periodically determine distinct control signals in very short periods of time in a Combustion Engine Control System - partly in dependence on the engine speed.
This work puts its focus on this subject with respect to the Engine Control System of an international automotive manufacturer. Thereby, precise estimates, or even knowledge of Worst Case Execution Times of Software Tasks play a decisive role. To this matter, a part of the Engine Control System has been remodeled using the Timing Definition Language. It puts its focus on guaranteeing time and value determinism by modeling the timing properties. An underlying principle of this Real-Time Programming Language is the specification of Logical Execution Times for tasks to abstract from platform specific details and behavior. Hence, this time interval has to be at least as large as the Worst Case Execution Time of this task. During the initial modeling phase, these Execution Times have been determined based upon Static Code Analysis. This approach resulted in a very conservative timing estimation policy. Consequently, the behavior of the modeled system diverged strongly from the legacy system's behavior. In the meantime, for the retrieval of more convenient execution times, an Integrated Circuit Emulator has been provided for measurements of Code Executions on the Hardware Platform.
Starting from this result, the model of the current TDL based Engine Control Software can be re-engineered. To this process, furthermore, two different versions of the TDL Runtime System have to be redesigned and applied. The TDL Runtime System deals with timely and correctly activation of the tasks and processing its results.
The outcoming systems will be subject to a quantitive comparison to each other and to the legacy system through Software-in-the-Loop simulations. The results are henceforth expected to hardly differ from the latter, since the refinement of the TDL implementation and the remodeling of the Engine Control System are expected to enable higher efficiency and smaller delays introduced by Logical Execution Times.