Code Generation for Discrete Event Controllers in Ptolemy II

August 17, 2011

The goal of this project is to provide a code generation framework for discrete event (DE) control models for cyber-physical systems (CPS) in Ptolemy II [1]. This framework should be as generic and platform independent as possible. The generated code should be tested on a target platform. Ptolemy offers the possibility to embed the generated code into the simulation environment. The generated code should also be tested using this mechanism.

1 CPS

A simplified Ptolemy model of a CPS is shown in Figure 1. It contains a plant model and a controller model. The controller is described as a DE system as exemplified in Figure 2. C Code should be generated for the controller to be executed on the target platform.

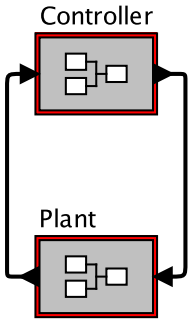

Figure 1: A model of a CPS

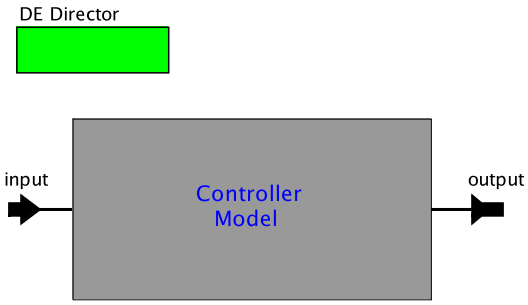

Figure 2: A DE controller model

2 Ptolemy II

Ptolemy [1] is an open source modeling and simulation tool for heterogeneous models of computation. Ptolemy models are actor-oriented. Communication between actors is done via ports. The semantics of the communication is implemented by a model of computation (MoC). Examples for such MoCs are discrete event (DE), continuous time (CT), synchronous reactive (SR), etc.

As shown in Figure 1, we separate a CPS roughly into two parts: the controller and the plant where the plant can be modeled as a continuous system and the controller is modeled as a discrete event system.

Ptolemy II contains a rudimentary code generation framework. The generated code follows the actor-oriented design principle. In the simulation, every actor uses a prefire, fire and postfire method to perform operations on inputs and provide outputs. The same methods should be provided in the generated code. As of now, code is only generated for a small number of MoCs, such as SR or Ptides, and a small number of actors.

3 Testing

The platform used to develop the code generation and run the simulations should be chosen as the target platform. In order to test the correctness of the generated code, appropriate inputs should be generated and the outputs should be monitored and compared with the simulation results.

The generated code can also be tested within the simulation framework. By using an actor that compiles and runs the generated code (the EmbeddedCActor) the generated code can be executed as part of the simulation model (see Figure 3).

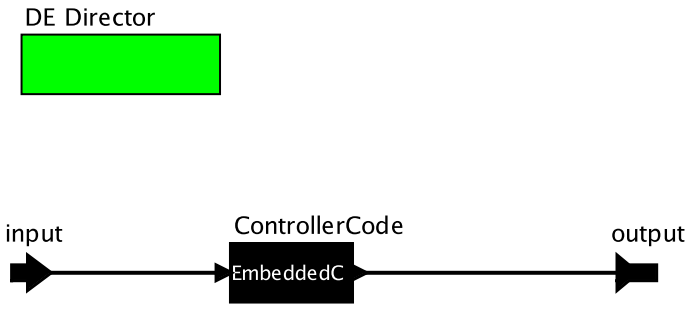

Figure 3: Verification of generated code by simulation

References

[1] J. Eker, J. W. Janneck, E. A. Lee, J. Liu, X. Liu, J. Ludvig, S. Neuendorffer, S. Sachs, and Y. Xiong. Taming heterogeneity—the Ptolemy approach. *Proceedings of the IEEE*, 91(2):127–144, 2003. Available from: http://www.ptolemy.eecs.berkeley.edu/publications/papers/03/TamingHeterogeneity/.