Creating a TDL Editor Suite

Andreas Werner

Overview

TDL (timing description language) is a language that defines the behavior of software in the time domain. With the help of TDL it is possible to define exactly when inputs of tasks are read and when outputs are written. Such definitions hold independently of the platform the software runs on, if it is possible to schedule the tasks with the given time constraints on the platform with the used algorithms.

Important TDL notions:

Mode
: A mode represents a state a TDL program can be in. A set of tasks runs within a mode. Each with it's own frequency relativ to the mode's period. Values are frequently exchanged between tasks.

Task
: A task is a box containing an externally defined function. Such a function is called functionality code and must be written in a language that is not TDL. A task defines how the function is called and how often it is executed within the containing mode's period. TDL assumes that a task is executed in a fixed logical execution time (FLET). The FLET is constant on all platforms on that the WCET of the function is small enough so that the function can be scheduled on the platform.

Mode Switch
: A mode switch defines under which conditions the program changes the mode it operates in. Switching modes can include copying values from one mode to the other.

If the behavior of software is defined in the value and time domain it's simulation will deliver the same sequence of values as the software's execution. Moreover hardware can be exchanged easily without modifying the system's behavior.

This has an important impact on the construction of hard real time systems: A complete system consisting of software and physical equipment can be simulated without the failure that may come from not knowing the exact execution times on the target platform.

In my thesis I design an editor for TDL. It will make it possible to visually design TDL programs and to export them to the industry standard simulation environment Mathlab Simulink or to save TDL code.

I have the following objectives:

1. Integrate TDL and Simulink: Create a visual TDL editor suite that can output TDL code and a Simulink model for simulation of a TDL program.
2. Design and implement a platform editor that supports the TDL annotations.

Important Components of the Suite

Three central components can be identified: One provides an overview of the system's modes and their transitions, one shows the behavior within a mode, and another defines the behavior during a

mode switch:

Mode transition editor

The mode transition editor displays all modes of the system and all possible mode switches in the form of a state chart. Modes are represented as states and mode switches are represented as transitions (see figure 1).

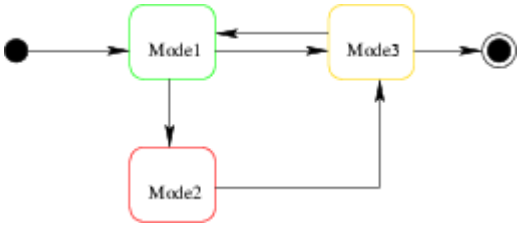

Figure 1: Mode transition editor

Mode editor

The mode editor (see figure 2) displays all tasks that run in a mode and a static abstraction of how ports are copied.

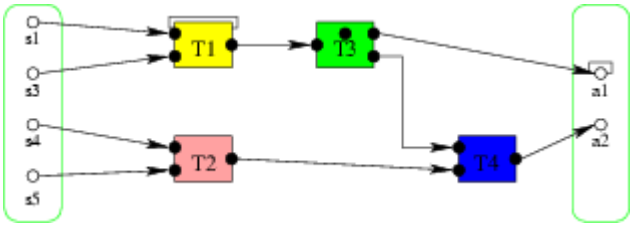

Figure 2: Mode editor: state in task T3; guarded: task T1, actuator a1

Mode switch editor

During a mode switch, values from ports of the old mode need to be copied to ports of the new mode. Which ports are copied whereto is defined in the mode switch editor (see figure 3). The left box represents the old mode, the right box the new mode.

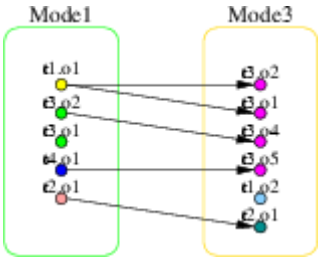

Figure 3: Mode switch editor's view

For a more detailed description please read "Visual Interactive Modeling of TDL Programs", Andreas Werner.

Time Line

Current state

A prototype comprising the mode transition editor and a tree view of the edited TDL program is completed. It allows to create modes and define mode switches and to edit some of their properties.

The architecture for editing TDL code has been refined and will probably be used as it is in the final

version.

Milestones in the next two months

The following features are expected to be supported in extended prototypes:

M1
: Create and edit tasks: until March 11

M2
: Mode editor: until March 25

M3
: Mode switch editor: until April 2

M4
: File I/O: until April 10

Estimated completion of thesis:

end of May 2004

Gerald Stieglbauer 2004-03-18