Code Generation for Embedded Systems

Real-time Control Using Modelica_DeviceDrivers

Martin Sjölund and Bernhard Thiele

Department of Computer and Information Science
Linköping University

2017-02-06

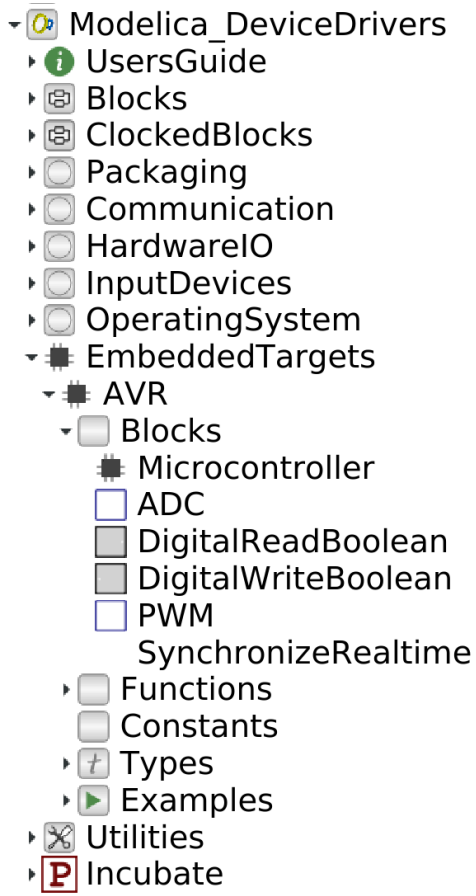
Overview

Part I	Modelica_DeviceDrivers
Part II	Example: Single Board Heater System (SBHS)
Part III	OpenModelica Code Generator for Embedded Systems
Part IV	Conclusion

Part I

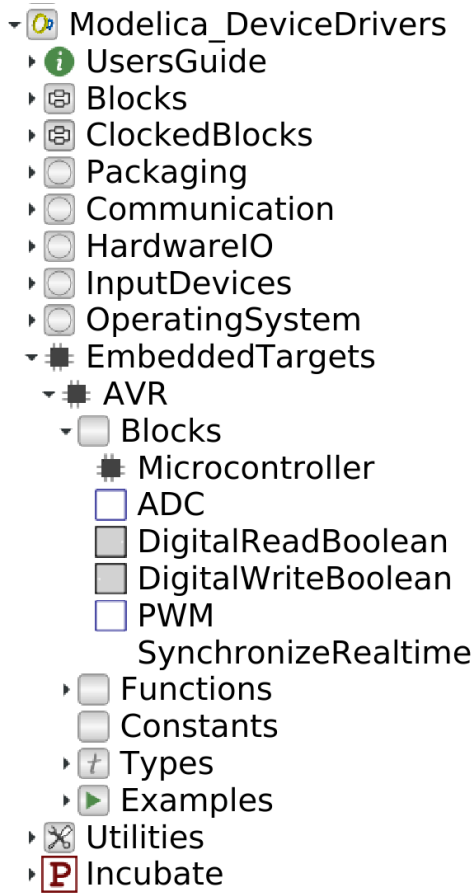
Modelica_DeviceDrivers

Modelica_DeviceDrivers



- **Free library** for interfacing hardware drivers.
- **Cross-platform** (Windows and Linux).
- I/O and communication.
- Supports **interactive real-time** simulations.
- Now also includes code for embedded targets.

Modelica_DeviceDrivers: Embedded Targets



- Explicitly model the hardware available in the microcontroller.
- The library includes external objects that deal with the microcontroller constants and flags.
- The AVR package handles Atmel's ATmega microcontrollers and includes analog and digital I/O as well as real-time synchronization.

Part II

Example: Single Board Heater System (SBHS)

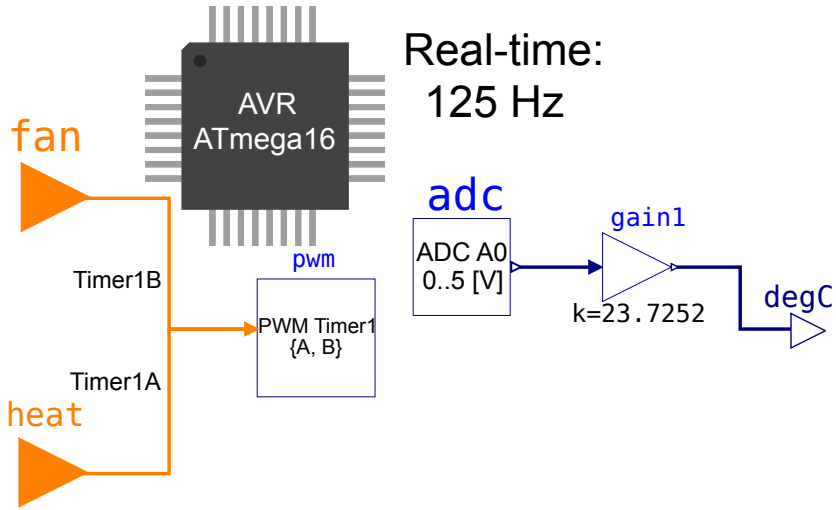
Single Board Heater System (SBHS)

One of the AVR examples included in MDD is the *Single Board Heater System* (SBHS, `http://sbhs.fossee.in/`), which was developed by IIT Bombay and is used for teaching and learning control systems. It consists of:

- Heater assembly
- Fan
- Temperature sensor
- AVR ATmega16 microcontroller
- Associated circuitry

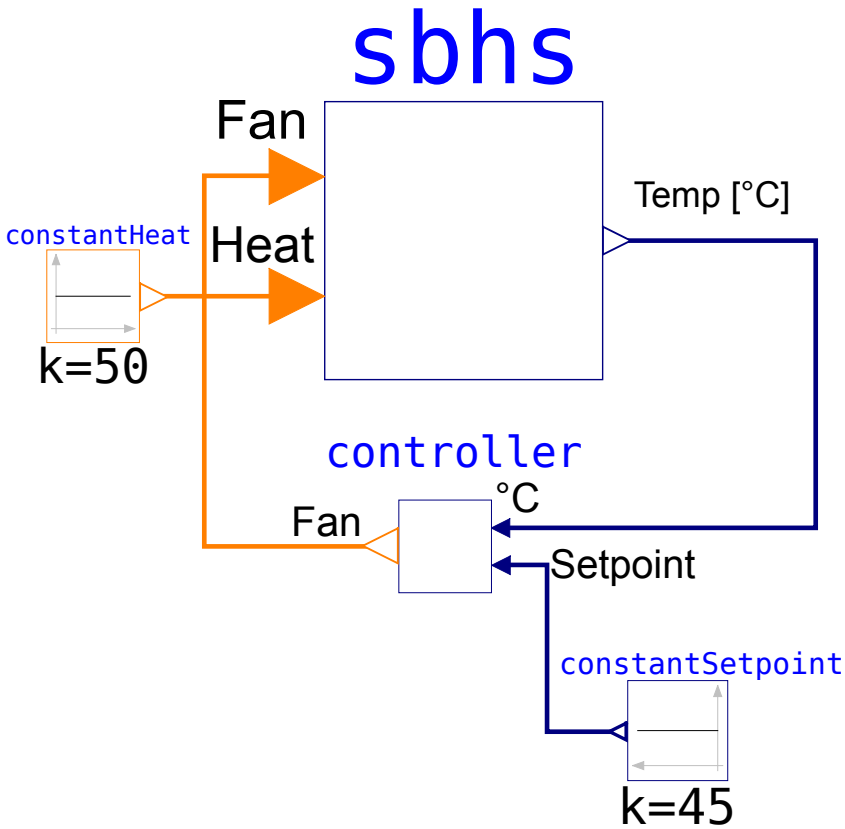
Modeling the SBHS

- Uses a real-time controller (here set @125 Hz).
- Uses pulse width modulation (PWM) to control the heater and fan.
- Uses an analog-to-digital converter (ADC) block to read the temperature (0V=0C, linear gain; the SBHS does the rest in hardware).
- Includes code for the LCD (not shown in the diagrams).



Controlling temperature using the fan

- The example feeds the heat assembly a constant (PWM) voltage.
- It then includes a PID controller with a fixed setpoint, trying to keep the temperature at a constant 45 °C by sending a PWM signal to the fan.



Part III

OpenModelica Code Generator for Embedded Systems

Code Generator

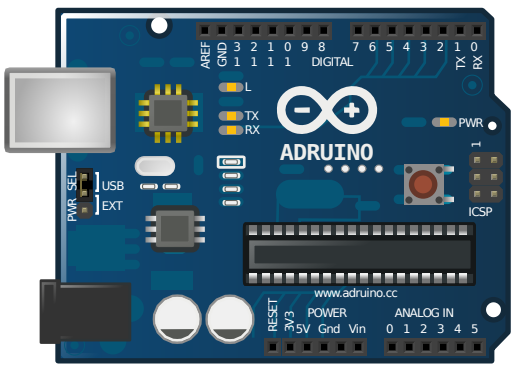
- Designed to support as many targets as possible.
- Supports few Modelica constructs.
- Focuses on generating good code with small footprint.
- Unsupported constructs such a linear systems are rejected.
- Reasonably predictable execution times.
- FMU-like interface (statically linked).

Code Generator Limitations

- No initialization.
- No strongly connected components.
- No events.
- No clocks.
- No functions (except some built-in and external C).

Target Agnostic

- No support for Atmel AVR or Arduino in the compiler.
- Compiler generates simple C code without use of OS or C library.
- Not a single malloc call, even during initialization.
- All hardware I/O and clocks is handled by the Modelica_DeviceDrivers library.



Using the Code Generator

Listing 1: Command sequence to use the code generator

```
# Generate a generic C-file
omc --preOptModules+=evaluateParameters --
    evaluateFinalParameters --evaluateProtectedParameters --
    replaceEvaluatedParameters -s --simCodeTarget=
    ExperimentalEmbeddedC M.mo
# Compile the C-code, targetting an ATmega328P clocked at @16
     MHz
avr-gcc -Os -std=c11 -ffunction-sections -fdata-sections -
    mmcu=atmega328p -DF_CPU=16000000UL -I ~/OpenModelica/
    build/include/omc/c -Wl,--gc-sections M_main.c -o M_avr -
    I ~/dev/Modelica_DeviceDrivers/Modelica_DeviceDrivers/
    Resources/Include /home/marsj/dev/SBHS/ModelicaLibs/
    libModelicaExternalC.a
# Create a hex-file used by avrdude
avr-objcopy -O ihex -R .eeprom M_avr M.hex
# Upload the hex-file corresponding to the controller using
    the Arduino USB protocol. Assume the processor is
    ATmega328P
avrdude -F -V -c arduino -p ATMEGA328P -P /dev/ttyACM0 -b
    115200 -U flash:w:M.hex
avr-size M_avr
```

Code Generator Comparison, Full vs Simple

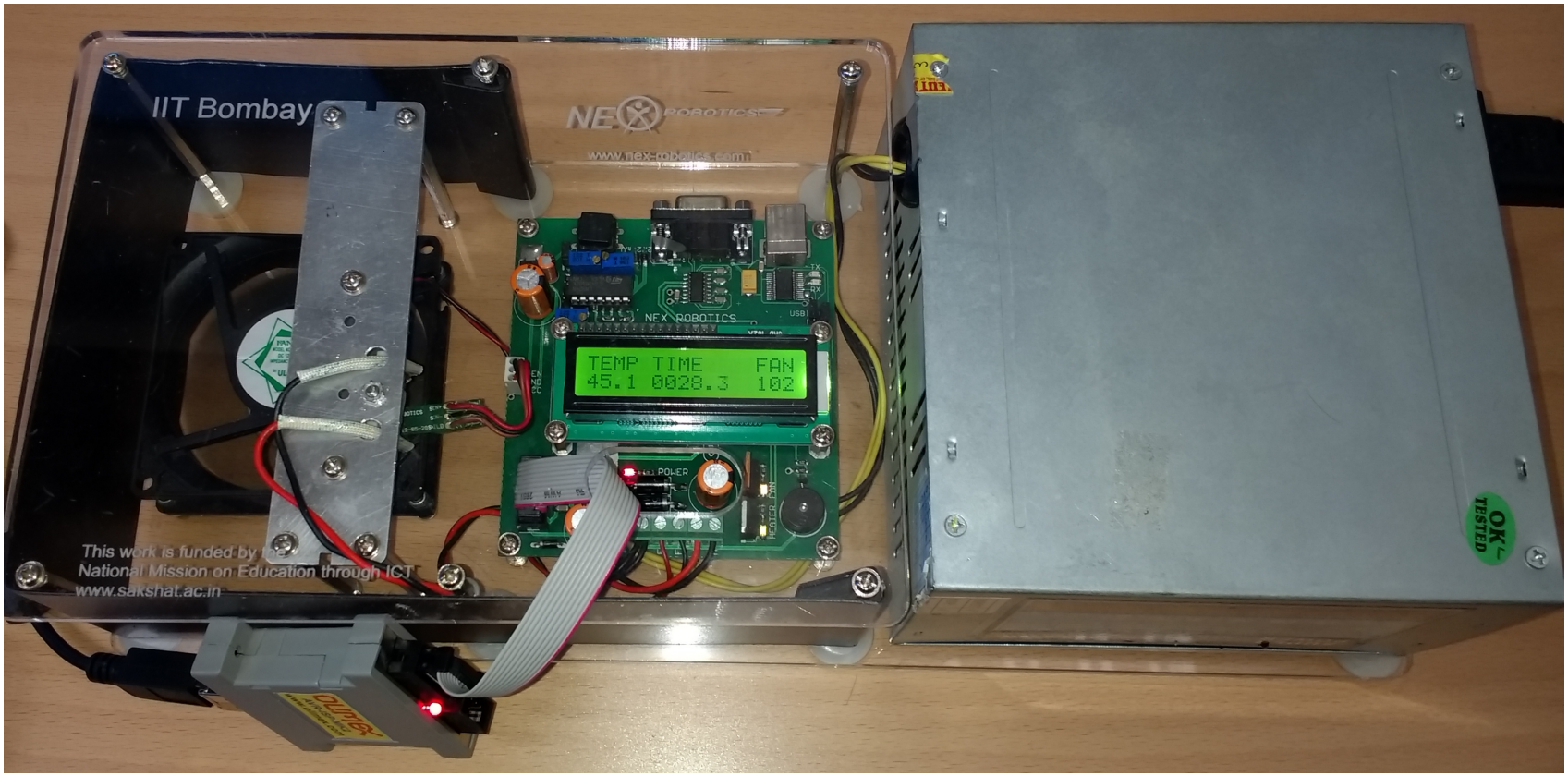
	Old code generator: Full source-code FMU targeting 8-bit AVR	Simple code generator targeting 8-bit AVR
Hello World (0 equations)	43 kB flash memory 23 kB variables (RAM)	130 B flash memory 0 B variables (RAM)
SBHS Board (real-time PID, LCD, etc)	68 kB flash memory 25 kB variables (RAM)	4096 B flash memory 151 B variables (RAM)

Table: The full code generator has a high overhead due to large strings, etc. being embedded in the executable whereas the simple code generator only contains code that is necessary to simulate the model. It also consumes a lot more program memory when more equations are added to the system. The largest 8-bit AVR processor MCUs (Micro Controller Units) have 16 kB SRAM. The common ATmega328p (Arduino Uno) has 2 kB SRAM. The ATmega16 we target has 1 kB SRAM available (stack, heap, and global variables).

Part IV

Conclusion

SBHS controller using MDD and the new code generator



Future Work

- Initialization.
- Strongly connected components.
- Still no event support planned.
- Instead... clocks (synchronous Modelica), and rewriting MDD to use it.
- Function support.
- Most of this after we simplify the code generators in OpenModelica.

www.liu.se