Exploring the use of Haskell to Program Microcontrollers used in Robotics Platforms

Student

Olivier Lischer

Introduction: To introduce children in schools to electronics and programming, "educational robotic kits" are being developed.
Traditionally, systems programming languages like C/C++ are used for embedded development and controlling hardware.
To simplify the process and allow children to focus on the general concepts, often a Scratch based IDE is often provided alongside the kit. Python is also often offered as an option for older children and more advanced use cases.
These technologies all follow the imperative programming paradigm.
Children using these technologies are introduced to thinking in an imperative model from an early stage.
Currently, there are no kits available that are targeted for functional programming languages.
This project demonstrates how a beginner-friendly kit can be programmed using Haskell, a functional programming language.

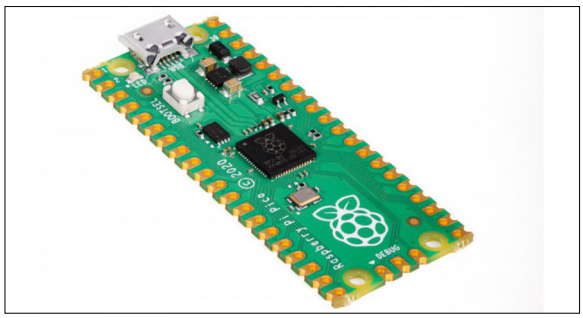
Approach / Technology: To run the Haskell source code on a robotic kit, it must first be compiled to a binary format. Although an operating system was developed using the standard Haskell compiler GHC, it is not suitable for this project. First, GHC normally requires a working operating system. A custom patch was developed to remove this restriction during the operating system development. The second problem is that GHC often produces a binary that is too large for the robotic kit.

An alternative approach could use the idea of "Compiling to categories" by Conal Eliot. This approach was used by Kittyhawk to power their aircraft. They developed "categorifier", a GHC plugin that produces C code after GHC performed the parsing and type checking. This approach was also not an option because the current examples do not compile, and the project is no longer being developed.
Lennart Augustsson started to develop a new Haskell compiler called MicroHs. It produces binaries small enough that it can run on microcontrollers. In this project, MicroHs was extended to support the Raspberry Pi Pico development board.

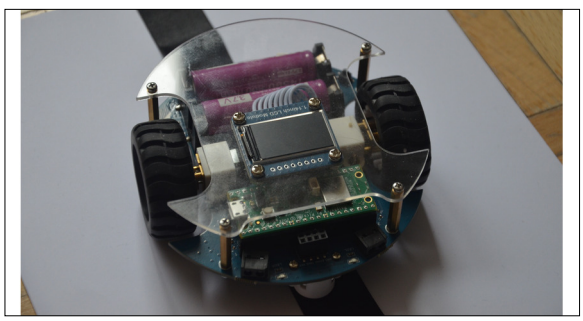
Result: First, the MicroHs project was extended to support the Raspberry Pi Pico RP2040 microcontroller. To ensure that the resulting binary can be run on the microcontroller, a simple application that turns the built-in LED on and off was developed.
After the successful prototype, the goal was to develop a line following robot. The PicoGo was used as a kit. It already comes with motors and sensors and can be controlled using the Raspberry Pi Pico. The Raspberry Pi Pico provides no Haskell SDK, but only C SDK to interact with the hardware. To use the functions from the C SDK, Haskell's Foreign Function Interface (FFI) was used to call the C functions. To ensure type safety and provide a better Haskell experience, wrapper functions and data types were developed for all used functions.

The logo of Haskell – purely functional programming language
http://static.igstan.ro/haskell-logos/png/logo2.png

Raspberry Pi Pico 2040
https://www.pi-shop.ch/raspberry-pi-pico



PicoGo Robot for Raspberry Pi Pico
Own presentment



Advisor
Prof. Dr. Farhad D. Mehta

Subject Area
Computer Science, Software and Systems, Mechatronics and Automation, Sensor, Actuator and Communication Systems