論文 / 著書情報
Article / Book Information

題目(和文)	リソースアウェアなリアクティブプログラミング言語
Title(English)	
著者(和文)	横山陽彦
Author(English)	Akihiko Yokoyama
出典(和文)	学位:博士(工学), 学位授与機関:東京工業大学, 報告番号:甲第12744号, 授与年月日:2024年3月26日, 学位の種別:課程博士, 審査員:渡部 卓雄,権藤 克彦,西崎 真也,DEFAGO XAVIER,林 晋平
Citation(English)	Degree:Doctor (Engineering), Conferring organization: Tokyo Institute of Technology, Report number:甲第12744号, Conferred date:2024/3/26, Degree Type:Course doctor, Examiner:,,,,
学位種別(和文)	博士論文
Category(English)	Doctoral Thesis
種別(和文)	論文要旨
Type(English)	Summary

（博士課程）
Doctoral Program

論 文 要 旨

THESIS SUMMARY

系・コース： Department of, Graduate major in	情報工学 系 情報工学 コース	申請学位（専攻分野）： Academic Degree Requested	博士 Doctor of （ 工学 ）
学生氏名： Student's Name	横山 陽彦	審査員主査： Chief Examiner	渡部 卓雄

要旨（英文 800 語程度）
Thesis Summary (approx.800 English Words)

Cyber-physical systems (CPS) have evolved in recent years. CPS is the integration of computational and physical processes. Embedded devices, called edge devices, are the boundary between the computational world and the real world. In CPS, the interaction between computation and physical processes is achieved by feedback between them through edge devices.

The key features of CPS include "high certainty" and " quick response". Edge devices are required to follow these two features. Therefore, it is important to design embedded systems that work stably in uncertain environments.

A system that responds to asynchronous inputs while changing its internal state, such as an edge device, is called a reactive system (RS). Functional Reactive Programming (FRP) is a programming paradigm for concisely describing RSs. In FRP, users can easily construct RSs by writing the data flow from input to output using time-varying values (values that automatically change over time). The affinity between FRP and embedded systems has led to several proposals for FRP languages for embedded systems. Emfrp, an FRP language for small embedded systems, is one such FRP language.

However, Emfrp and other FRP languages have difficulty in programming robust embedded systems, such as edge devices, because they lack a mechanism for precisely handling computational resources. In this study, we focus on three important computational resources in embedded systems, analyze the problems of these resources in FRP languages, and propose a solution. Specifically, we analyzed memory resources, CPU resources, and power resources of Emfrp, which is a typical FRP language for embedded systems. For each resource, we proposed a solution to the corresponding problem by proposing a new language that extends the Emfrp language. A summary of these languages we proposed is given below.

(EmfrpBCT) Emfrp prohibits recursive definitions to run in a fixed memory space. This restriction prevented the use of conventional functional programming techniques in Emfrp programs. This problem forced users to write unnatural programs. EmfrpBCT, proposed in this study, is an extension of Emfrp that introduces size-restricted recursive data types (Bounded-Construction Types, BCT). In addition to relaxing the restrictions on recursive definitions, EmfrpBCT introduces syntax such as fit expressions to improve the affinity between BCTs and time-varying values. EmfrpBCT allows the user to determine the amount of memory used by the program while still using recursive data types. After formalizing EmfrpBCT and demonstrating its soundness, we developed a compiler and measured various overheads.

(EmfrpHT) In FRP languages that update time-varying values in a single-threaded fashion, the responsiveness of the entire system degrades even if one time-varying value to be updated involves a heavy-task computation. This phenomenon is called the Reactive Thread Hijacking Problem (RTHP) and has been considered a problem. In this study, we point out the problems with the current RTHP avoidance methods in Emfrp and propose EmfrpHT to solve the problems by presenting an example of a micromouse robot competition. EmfrpHT has a mechanism to define heavy-tasks and handle them as time-varying values, and a runtime to execute heavy-tasks asynchronously. EmfrpHT allows reactive behavior and heavy-task computation to coordinate their actions, which stabilizes the overall responsiveness of the system.

(EvEmfrp/S) In Emfrp, time-varying value updates are performed continuously by a kind of polling to reduce memory consumption. This mechanism causes updates to occur more frequently than necessary, resulting in high power consumption. EvEmfrp, a successor to Emfrp, reduces power consumption by performing time-varying value update processing intermittently. In EvEmfrp, time-varying values have annotations regarding update frequency. The compiler uses this as a hint to schedule the update interval appropriately. However, when describing actions that "start from an asynchronous time and process after a certain time", which is often used in embedded systems, for example, the power saving effect may not be sufficient due to the lack of expressiveness of the annotations. In this study, we proposed EvEmfrp/S, an extended language with a mechanism to dynamically switch the frequency of time-varying value updates by renewing the timing annotations in EvEmfrp. Through a software example of button denoising, we showed that programs in EvEmfrp/S can take advantage of hardware sleep more aggressively than conventional EvEmfrp and can be expected to save power.

The integration of these languages is a future challenge. We expect the results of this study to be a step towards the design of FRP languages for robust embedded systems.