Virtual Platform Generation Using TECS Software Component and SCE

Takuya Azumi
Ritsumeikan University

Yuko Hara-Azumi
Ritsumeikan University

Rainer Dömer
Center for Embedded Computer Systems
University of California, Irvine

***Abstract*—The paper presents our going work on a system-level design framework integrating TECS, which is one of the software component technologies for embedded systems, and SCE, which is a system-on-Chip environment based on SpecC language, for realizing a higher abstraction level design than prior work. Since TECS implementation is based on conventional C language, such as function calls, it is suitable for embedded software developers. Then, our framework supports the transformation from component descriptions and component sources to SpecC specification for using SCE advantages, such as design space exploration and efficient MPSoC implementation. An application for creating a panoramic image removing objects, such as people, shows an example of data partitioning and parallelization by using our framework.**

I. Introduction

Increasing embedded system complexities and strict time-to-market schedules are critical issues in the today's system-level design. To improve the design productivity, designing the systems at a higher abstraction level is necessary [1].

In embedded software domains, meanwhile, software component technologies have become popular to improve the productivity [2], [3], [4], [5]. It has many advantages such as increased reusability, reduced time-to-market, reduced software production cost, and hence, improved productivity [6]. We, therefore, propose a system-level design framework for realizing a higher abstraction level design than prior work. We integrate a software component system for embedded systems, TECS (TOPPERS Embedded Component System [2]), and the system-on-chip environment SCE [7], which is based on SpecC language.

The contribution of this work is to present a system-level design method based on TECS which provides a higher abstraction level design environment than existing works such as [7], [8], [9]. In the existing HW/SW codesign technologies, designers need to manually add or modify HW/SW communication sources (e.g., their size, direction, and allocator) in input behavioral descriptions, which is complex to specify. In contrast, in our framework, the designers can design the overall system at higher abstraction level and have no need to specify the HW/SW communication in the input description because TECS specifically defines the interface between components, and the communication sources are automatically generated.

Fig.1 represents the proposed design flow using our framework. At Stage 1 in Fig. 1, starting from component descriptions and component sources in TECS [2], the SpecC system model, including definitions of *behavior*s and *channel*s, is generated by a TECS generator. At Stage 2, the remaining design flow is the same as a general SCE design flow [7]. SCE supports a top-down system design flow based on a specify-explore-refine paradigm with support for heterogeneous target platforms consisting of custom hardware components, embedded software processors, dedicated IP blocks, and complex communication bus architectures.

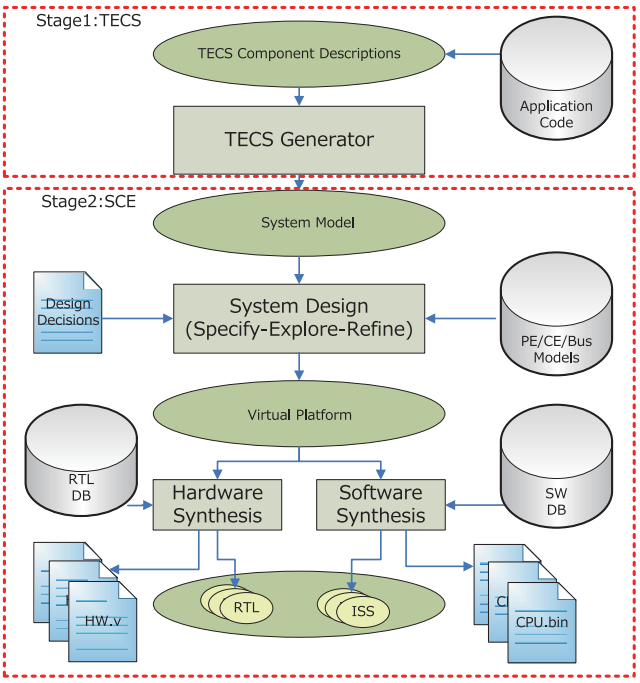

Fig. 1. Design flow for virtual platform generation

II. Target Application

The target application for the framework is an application for generating a panoramic image removing objects, such as people. In the panoramic image view system, such as Google Street View, a user can see images from the street using omnidirectional images. Fig. 2 illustrates the target application. The application creates the image without people as shown in the right image of Fig. 2 based on the algorithm [10] by using a set of panoramic images which are taken at the same position.

Since creating an image by removing obstacles needs a number of original images, each of which has two million pixels the original program is designed only for off-line use.

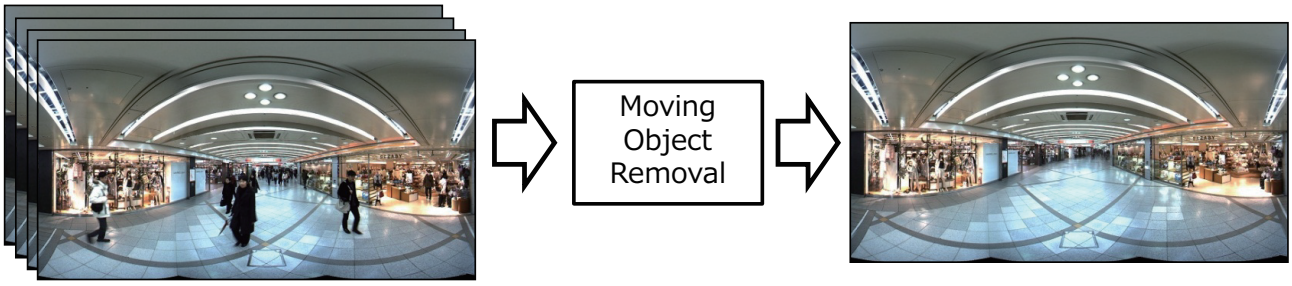

Fig. 2. Target application

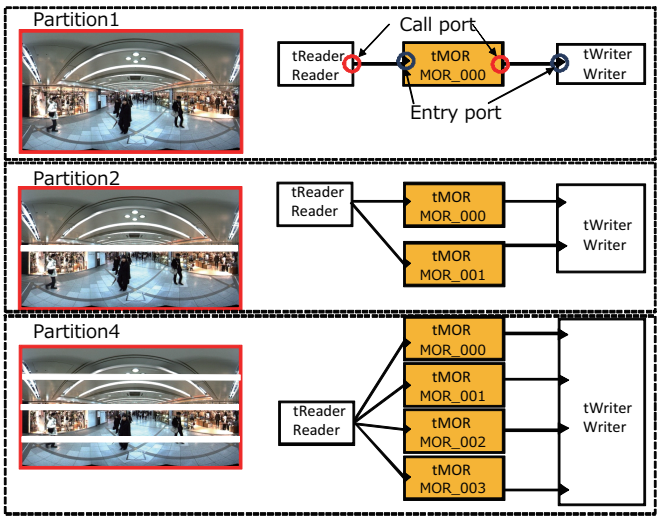

Fig. 3. Component diagram for target application

Because the output image depends on place and environment, we do not know how many source images are needed to create the output image. Therefore, currently, we need extra time to take pictures at each place. Our final goal is to create the image remaining objects in real-time by using our framework.

Fig. 3 shows a TECS component diagram for the application. Each rectangle represents a *cell* which is a component in TECS. The left, middle, and right *cell*s are a Reader *cell*, an MOR (MovingObjectRemoral) *cell*, and a Writer *cell*, respectively. The Reader *cell* reads image files and sends the image data to the MOR. The MOR *cell* collects background colors (RGB) of each pixel based on the input images. The Writer *cell* creates the final image based on the data which the MOR *cell* collected. Here, tReader, tMOR, and tWriter represent the *cell type* name. The *cell type* is the definition of a *cell*, such as the *Class* in an object-oriented language. The triangle in the *cell*s depicts an *entry port*. The *entry port* is an interface to provide services (functions) to the *cell*. The connection of the *entry port* in the *cell*s describes a *call port*. The *call port* is an interface to use the services of other cells. A *cell* communicates with this environment through these interfaces. The *entry port* and the *call port* have *signatures* (sets of services). A *signature* is the definition of interfaces in a cell.

A basic policy of transformation is that a *cell* and an argument of function of *signature* correspond to a *behavior* and a *channel* in SpecC language, respectively. Note that it is easy to duplicate MOR *cell*s for realizing data partitioning and parallelization as shown in Fig. 3.

III. Conclusion

This paper proposed a new system-level framework integrating TECS and SCE for realizing a higher abstraction level design than prior work. The advantage of our framework is to use software components for system-level design without modifying input C sources (component sources). Moreover, since TECS supports data partitioning and SCE supports MPSoC as target architecture, our framework can deal with more complex applications, and can help in parallelizing them for efficient implementation. We plan to implement the TECS generator and will evaluate it using the MOR application.

References

[1] A. Sangiovanni-Vincentelli, "Quo vadis, SLD? Reasoning about the Trends and Challenges of System Level Design," *PROCEEDINGS-IEEE*, vol. 95, no. 3, pp. 467–506, 2007.

[2] T. Azumi, M. Yamamoto, Y. Kominami, N. Takagi, H. Oyama, and H. Takada, "A new specification of software components for embedded systems," in *Proc. 10th IEEE International Symposium on Object/component/service-Oriented Real-Time Distributed Computing*, May 2007, pp. 46–50.

[3] AUTOSAR, "Autosar specification," http://www.autosar.org/.

[4] F. Loiret, J. Navas, J.-P. Babau, and O. Lobry, "Component-based real-time operating system for embedded applications," in *Proc. 12th International Symposium on Component-Based Software Engineering*, Jun. 2009, pp. 209–226.

[5] M. Åkerholm, J. Carlson, J. Fredriksson, H. Hansson, J. Håkansson, A. Möller, P. Pettersson, and M. Tivoli, "The SAVE approach to component-based development of vehicular systems," *Journal of Systems and Software*, vol. 80, no. 5, pp. 655–667, May 2007.

[6] K.-K. Lau and Z. Wang, "Software component models," *IEEE Transactions on Software Engineering*, vol. 33, no. 10, pp. 709–724, Oct. 2007.

[7] R. Dömer, A. Gerstlauer, J. Peng, D. Shin, L. Cai, H. Yu, S. Abdi, and D. D. Gajski, "System-on-Chip Environment: A SpecC-Based Framework for Heterogeneous MPSoC Design," *EURASIP Journal on Embedded Systems*, vol. 2008, pp. 1–13, Jan. 2008.

[8] H. Nikolov, M. Thompson, T. Stefanov, A. D. Pimentel, S. Polstra, R. Bose, C. Zissulescu, and E. F. Deprettere, "Daedalus: Toward composable multimedia mp-soc design." in *Proc. International 45th Design Automation Conference*, Jul. 2008, pp. 574–579.

[9] S. Honda, H. Tomiyama, and H. Takada, "Rtos and codesign toolkit for multiprocessor systems-on-chip," in *Proc.In 12th Asia and South Pacific Design Automation Conference*, Jan. 2007, pp. 336–341.

[10] M. Hori, H. Takahashi, M. Kanbara, and N. Yokoya, "Removal of moving objects and inconsistencies in color tone for an omnidirectional image database," in *Proc. of ACCV2010 Workshop on Application of Computer Vision for Mixed and Augmented Reality*, Nov. 2011, pp. 62–71.