Integration of Java processor core JSM into SmartDev(ices)

Frank Golatowski, Stephan Preuss*, Hagen Ploog, Thomas Geithner*,
Clemens Cap*, Dirk Timmermann

Dept. of Electrical Engineering & Information Technology
University of Rostock
R.-Wagner-Str. 31, 18119 Rostock-Warnemuende
Germany

* Dept. of Computer Science
University of Rostock
A.-Einstein-Str. 21, 18059 Rostock
Germany

Abstract – **This article describes the current work to extend the Java processor Java Silicon Machine (JSM) for usage in embedded systems. The JSM is a JavaCard processor supporting all JavaCard bytecodes. The JSM is a fully synthesizable 32bit processor soft core with a very small footprint. The capability of it's integration in small embedded and automation systems is outlined. Special target platform is the SmartDev system which consists of a Java core interfacing to a wide variety of peripherals. SmartDev is intended to be used in mobile embedded systems for administrational, controlling and measurement purposes.**

I. INTRODUCTION

Smart cards, and hence Java Cards, are a special category of mobile embedded systems. Due to that fact, the Java Silicon Machine (JSM) introduced here meets the requirements of a wide range of embedded systems. The JSM brings the following advantages to embedded applications:

- Simplified application design and hence shorter development cycles by the object oriented Java architecture,
- Higher performance and lower resource requirements versus a JVM,
- Flexible and extensible design to natively integrate new functionality (JVM extensions, peripherals),
- Limited real time support.

The Java architecture has been designed to run its bytecode in a virtual machine that uses the functionality of the underlying operating system to manage hardware like memory or peripherals. In the JSM, it is the task of the processor to provide that functionality. New bytecodes were introduced to access the IO-Unit. That makes it easy to extend the IO-Unit with new devices commonly found in embedded systems like CAN controllers or UARTs. Fig. 1 shows a comparison of the software and hardware Java architecture.

The Java Card specification does not require the presence of a garbage collector. The lack of garbage collection results in special constraints in application design and makes the system real time capable. Hence, such systems may be applied in time critical applications.

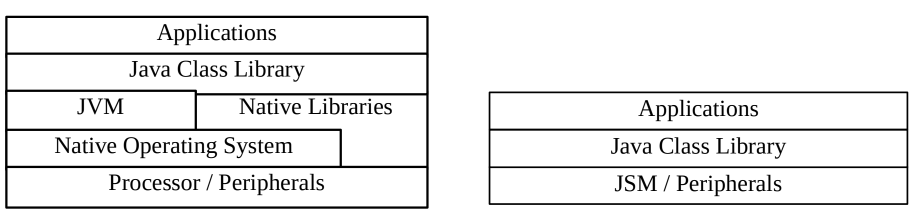
Furthermore a Java card virtual machine does neither have a class loader nor a byte code verifier as online parts, because of the limited hardware resources. These functions are executed offline. Bytecode verification and class loading are done outside the card prior to downloading the applications (cardlets) to the card's memory.

In the case of a Java Card that's not really a loss of functionality because the java card is designed with this poor communication capabilities. Usually, each data transfer is a host initiated request-response sequence.

The situation changes if the Java Card VM is used in embedded systems that may communicate via field busses or even LANs. Here, it could be desirable to load external Java code into the local Java machine to get new functionality or to interface to network services.

A wide spread embedded system architecture is that of the Intel8051 8bit micro controller. It is applied to smart cards as well as sensors or actuators. To make these legacy systems Java capable the architecture is left unchanged and a JVM is set on top of the embedded operating system. Doing so, often leads to a very poor performance of Java applications. To avoid that, the JSM could be applied to systems that have to offer both Java capability and high performance.

Furthermore, the extensibility of the JSM offers new usage scenarios. Currently it is equipped with a DES encryption unit and others are under development.



A: Software Java Architecture.

B: Hardware Java Architecture.

Fig. 1

TABLE I
DIFFERENT HARDWARE SOLUTIONS FOR JAVA AND JAVA CARD VIRTUAL MACHINES

	JSM [7, 3]	**SmartJ** [17]	**TINI^{2}J** [1]	**JSMART** [13]	**JSTAR** [13]	**LavaCore** [4]	**Jazelle** [2]	**Komodo** [11]	**MOON** [12]
Target	Softcore, Xilinx FPGA, ASIC	Integration with ST22 core	Integration with RISC processor, CPLD,ASIC	Softcore	Softcore	Softcore, Xilinx FPGA	Integration with ARM	Softcore, Xilinx FPGA	Altera APEX
Special feature	Hardware security features, Integration of external devices, e.g. I^2C, TDES, Support of JVM stack model	Hardware security features, Combi-nation with micro-controller	Hardware security features, Interrupt control interface	Co-processor	Co-processor	Optional extendable with SPU, DES, GC	Compatible with ARM, Combi-nation with ARM-controller.	Multithreaded Kernel., Interrupt Service Threads	Support of JVM stack model, Co-processor and Single processor solution
Support of additional codes	New bytecodes	Bilingual	Bilingual	Bilingual	Bilingual		Trilingual		
Standard	Javacard	Javacard	Javacard	Javacard	Java	Java	Java	Java	Java

II. RECENT WORK

During the last years some work has been done in developing different kinds of Java processor cores. Table I summarizes different hardware solutions for Java systems.

III. DESIGN OF JAVA CARD-PROCESSORS JSM (JAVA SILICON MACHINE)

The JSM has a modular design (see Fig. 2). The JSM processor architecture was designed to fulfill the demands of different kinds of security applications (high security design). One of the most valuable features is that the core is extendable. Different types of external devices
may be combined with the core using a simple interface.
The processor consists of the following main modules (see Fig.2).

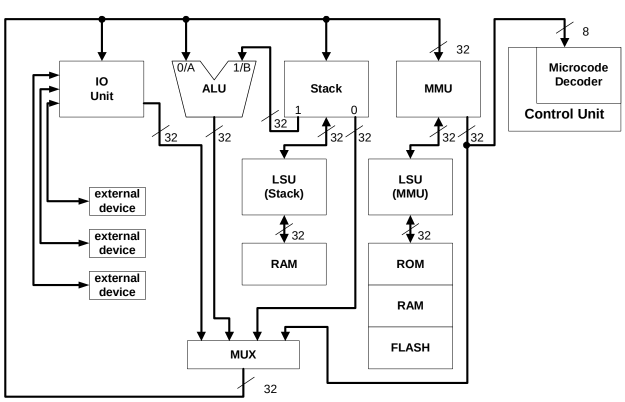


Fig. 2 Design of the JSM-processor

- Control-Unit (CU),
- Arithmetic Logical Unit (ALU),
- Memory Management Unit (MMU),
- Stack, and
- Input/Output-Unit (IO-Unit) (see Fig. 3).

A.IO-Unit

The JSM was intended as a JavaCard processor with the capability to connect external devices and for the use in embedded systems. The IO-Unit serves as an interface between the processor and the corresponding IO-devices which are not defined in the Java standard or can be accessed only by using an appropriate application programmers interface (API). Especially in a JavaCard environment the IO-Unit is necessary for the communication between the card itself and the card reader. Furthermore, the IO-Unit may be used to control additional hardware to accelerate internal computations (e.g. encryption engine). The interface for the attachment of external devices is shown in Fig. 3. Further details about structure, signals, and components of the IO-Unit as well as an example for the integration of an external device (I^2C-device) are explained in [3, 7].

IV. SMARTDEV(ICES)

SmartDev [8] is a Java enabled embedded platform with numerous communication facilities and interfaces to wireless and biometric technologies, positioning and security systems. It's mainly intended to be used in mobile systems and in stationary systems that require wireless access and a high level of security. SmartDev application scenarios range from mobile points of sales (POS) via access control systems to almost any remotely accessible control an measurement system.

Currently, SmartDev works with GSM wireless communication having the possibilities to exchange data over TCP/IP, SMS, FAX or simple serial connection. It's equipped with a smart card reader offering the use as POS or access controller. An enhancement of the security features is the attachment of biometric sensors. The interface for a fingerprint sensor is currently under development. Furthermore, it has a GPS unit attached that is mainly intended to be used to get positioning information in mobile devices. A side effect of the GPS usage is the presence of accurate time information that may be used in securing data transfers.

The whole system is currently built around the TINI (Tiny INternet Interface) [6], a low cost Java capable embedded system from Dallas Semiconductor. TINI's core is a DS80C390 [5], which represents a high performance Intel8051-derivate. The TINI module itself provides 10BaseT Ethernet, I2C, CAN, RS232, 1-Wire and a 4bit wide digital control port. Keyboard and display may be attached via I2C. These features make it suitable to be integrated in a broad range of systems to act as administration, control, or measurement interface (Fig. 4)

The architecture of the TINI module corresponds to that shown in Fig. 1A. On top of it's native operating system (TINI OS) work one or more JVMs. A JVM runs the bytecode directly out of the file system that lies in the battery backed RAM. Class loading and verification is

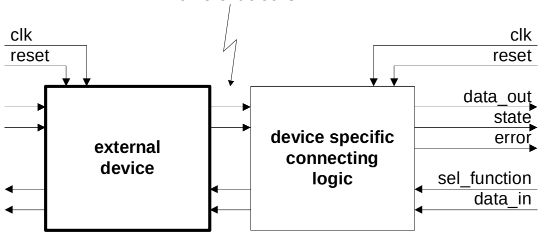


Fig. 3: Interfacing external devices

done offline prior to application downloading. With the exception that the TINI's JVM and class library is partly compatible to the Java 1.1 standard this architecture is similar to that of the Java Card specification.

V. SMARTDEV SUPPORTED BY JSM

Because of similar functionality the TINI may be easily replaced inside the SmartDev system by the JSM making the core more lean with higher performance. The JSM's modular design allows it to integrate almost any peripheral unit on the chip and in this way to customize the chip to the application's needs. That has the effect that the class library may directly operate on top of the hardware (no drivers or native libraries are needed). Performance and reliability are improved because of the reduction of the numbers of software layers. Furthermore, the overall size of the system is reduced because external components may now be integrated.

Currently, the replacement of the TINI module by the JSM is under development. In the first step of this process the JSM core is extended by external components to pro

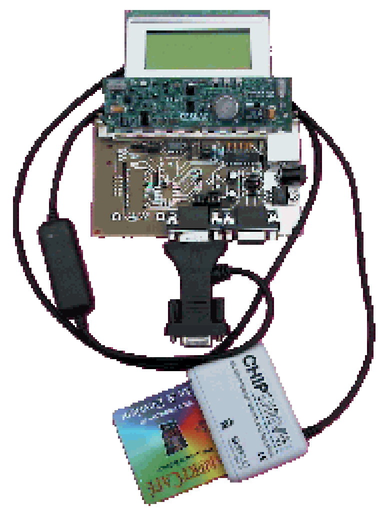


Fig. 4 SmartDev built around the TINI module

vide the functionality that's not available on chip. Most important are serial communication lines to interface to GSM, GPS, card reader, and fingerprint sensor. For an easy integration into the target environment LAN and CAN interfaces are necessary. In the second step all these peripherals will be put on the JSM chip making the system compact and small and hence applicable to almost any target device.

VI. FUTURE WORK

With the ongoing developments in the field of ubiquitous computing the demand for high performant embedded Java systems will increase significantly. They'll drive devices like PDAs, cell-phones, smart badges, as well as automation appliances. The functionality of the JavaCard VM is not suited to meet all these devices' needs. That's why, the JSM development will go on to provide Java1.1 compatibility and features like garbage collection, dynamic class loading, and bytecode verification. On top of these developments the new JSM-powered SmartDev architecture will be enabled to work in Jini [14] environments.

VII. REFERENCES

[1] Advancel Logic Corporation, Tiny2J Micro-*processor Core for JavaCard Applications*, http://www.advancel.com

[2] JazelleTM – ARM® Architectumre Extensions for Java Applications, White Paper, www.arm.com

[3] N. Bannow, Java-processor for SmartCards and small embedded systems. (in german) Diploma thesis, Institute of Applied microelectronics and computer engineering, University of Rostock, Dec. 2000

[4] Bhaskar Bose, M. Esen Tun; LavaCORE - A Configurable Java Processor; http://www.xilinx.com/xcell/xl37/xcell37_20.pdf

[5] Dallas Semiconductor. DS80C390 Dual CAN High-Speed Microprocessor, 1999. http://www.dalsemi.com/datasheets/pdfs/80c390.pdf

[6] Dallas Semiconductor. DSTINI1 TINI Verification Module, 2000. http://www.dalsemi.com/TINI/dstini1.pdf

[7] H. Ploog, R. Kraudelt, N. Bannow, T. Rachui, F. Golatowski, D. Timmermann: A Two Step Approach in the Development of a Java Silicon Machine (JSM), Workshop on Hardware Support for Objects And Microarchitectures for Java. Austin, Texas, October 1999

[8] T. Geithner, Mobile GSM Powered Web Server. (in german) Project Work, Chair of Information and Communication Services, University of Rostock, May 2001

[9] F. Golatowski, H. Ploog, R. Kraudelt, T. Rachui, O. Hagendorf: Java Virtual Machines for resource critical embedded systems and Smart-Cards,. (in german), Java Informationstage JIT 99, ITG/GI-Fachtagung, Düsseldorf, September 1999

[10] F. Golatowski, N. Bannow, H. Ploog, J. Hildebrandt, D. Timmermann: JSM- Java Processor for embedded systems: Design, Implementation and Rapid-Protoptyping, (in german), 10. Maritimes Symposium Rostock, June 2001

[11] J. Kreuzinger, R. Zulauf, A. Schulz, Th. Ungerer, M. Pfeffer, U. Brinkschulte, C. Krakowski, Performance Evaluations and Chip-Space Requirements of a Multithreaded Java Microcontroller, Workshop on Hardware Support for Objects And Microarchitectures for Java. Austin, Texas, October 2000

[12] Moon Java Processor Core, http://www.vulcanasic.com/eda.htm

[13] Java Coprocessor, The JSTAR, Product brief, http://www.nazomi.com

[14] Sun Microsystems. Jini Network Technology, 1998. http://www.sun.com/jini/

[15] Sun Microsystems, Inc., Java Card Technology Home Page, http://java.sun.com/products/javacard

[16] Sun Microsystems Inc., Java Card TM 2.1.1 Virtual Machine Specification, 2000

[17] SGS Thomson, Instant Java for your smartcard, http://www.st.com/stonline/prodpres/smarcard/insc9901.htm