Research Report 17/2001
Real-Time Systems Group
University of Technology
Vienna, Austria

An Embedded System Hosting a CORBA and TTP/A Service

Wilfried Elmenreich*, Christian Trödhandl†, and Thomas Losert‡

July 18, 2001

Abstract

The time-triggered fieldbus system TTP/A is probably becoming part of an OMG standard for smart transducer interfaces. This standard will comprise the connection of TTP/A fieldbus clusters to a CORBA ORBit server.

This paper discusses the possibility of implementing an embedded system that contains a TTP/A master, an ORBit server and a TCP/IP interface on a single board computer.

The results show, that the migration of the master task to a separate chip eases the software implementation. The selection of a commercial Linux-friendly single computer board is preferable to the setup of a proprietary hardware setup, because of the effort on the adaption of the Linux operating system.

Keywords: *Smart Transducer Interface, CORBA, TTP/A, embedded Linux.*

1 Introduction

A distributed control system must support predictable performance in the temporal domain. Many of the standard communication protocols, such as General Inter-ORB Protocol (GIOP), have not been designed for temporal predictability. So, in response to a request for a proposal of a smart transducer interface [1] a new standard has been proposed at the OMG [2] that comprises a time-triggered transport service within the distributed smart transducer subsystem and an encapsulated gateway of this subsystem to the CORBA environment.

It is the objective of this paper to discuss the possibility of implementing an embedded system that contains a TTP/A master, a CORBA server and a TCP/IP interface on a single board.

The rest of the paper is organized as follows:

Section 2 discusses the software for the system, namely the operating system, the non-time critical application task and the real-time application. Section 3 discusses the hardware issues and lists the key components for the system. Section 4 lists a sampling of some ready-made Linux-friendly single board computers. The paper is concluded in Section 5.

*wil@vmars.tuwien.ac.at
†christian.troedhandl@vmars.tuwien.ac.at
‡tl@vmars.tuwien.ac.at

2 Software

There are many embedded operating systems on the market, but we decided on embedded LINUX for several reasons [3]:

Support for Multiple Processor Architectures: Linux supports a number of microprocessors, such as PowerPC, 68k, and Intel.

Multiple Bus Architecture Support: It also works with different bus architectures such as VME, PCI, ISA, and others. Device drivers are readily available for multiple hardware devices.

Application Software: Part of our system is a CORBA ORBit server, which is available for the Linux operating system.

Of course this advantages come with some costs: The hardware ressources of Linux are more demanding in comparison to other embedded operating systems such as QNX, VxWorks, CMX-RTX, etc. The real-time capabilities of those other operating systems are also better than of Linux, even under consideration of the RTLinux-extension [4].

3 Hardware Setup

The desired system should be robust and of small size to be applicable in automotive or automation environments. We will examine the hardware components following these goals.

Furthermore the setup will contain

An embedded microcontroller : Although Linux was originally designed for X86 compatible microprocessors, the operating system supports a wide range of microprocessors, such as PowerPC, 68k, Alpha, ARM, and MIPS. While the ARM microcontroller provides a modern design and excellent market chances, development time will be probably shorter when using a traditional x86 core.

Interface units : e. g. a TTP/A bus driver and an ethernet driver.

ROM memory : The ROM memory contains the basic input/output functions and a loader for the operating system. About 128 kB of memory are sufficient for this task.

Working memory : The embedded Linux system kernel needs about 2 Megabytes of RAM during operation. The desired applications need at least 4 Megabytes to work properly.

Persistent memory: This memory contains the basic input/output functions, a loader for the operating system, a compressed image of the Linux kernel and the application software, and a file system. A nonmechanical alternative to a hard disk drive, e. g. a flash card [5] will be favorable.

It would be suitable to add one small TTP/A master node for each TTP/A bus to decouple the time critical master program from the system because of the following reasons:

- in case of a failure, the time to reboot the Linux system is quite long (seconds) in comparison to the start-up time of a small microcontroller (a few milliseconds) running the TTP/A master protocol.
- Linux needs a hard real-time kernel to be able to fulfil the timing requirements of the TTP/A protocol. Such a hard real-time extension exists with Real-Time Linux, but the current edition of RTLinux V3 is only available for x86, PowerPC and Alpha processors, with a MIPS port in beta release.
- There are already working master implementations for Motorola 68k, Atmel ARM7 [6], Atmel AT90 [7, 8], and PIC16 [9]. Working out a new implementation for RTLinux running concurrent to other services (possibly even another master task) would afford even greater development cost and time than the effort for the current master implementations.

It is theoretically possible to pick the hardware parts as described above, build a single board computer and adapt the system software for this hardware. However this would lead to tremendous development costs, especially for the software part.

Linux has been originally designed for PC-compatible desktop computers. The more given hardware differs from PC-compatible, the more problems arrive when you try to put Linux on such a system. Even knowing that Linux drivers or in-kernel support exists for the employed chips is encouraging but not sufficient. [10]

Instead there exist some "Linux-friendly" single board computers. Some vendors offer an integrated product consisting of a single board computer, a taylored embedded operating system based on the free Linux distribution, and development tools to ease the configuration and set-up.

The next section will give an overview on available Linux-friendly single board computers.

4 Linux-friendly Single-Board Computers

For small Single-Board Computers below the size of PC/104 modules there are currently no form-factor standards. Here is a sampling of some small embedded Single-Board Computers that support embedded Linux:

Single-Board Computers with x86 microcontrollers

CompuLab 486CORE – a tiny (2.9" x 1") PC-compatible SBC based on the AMD Elan SC400. Includes: DRAM, Flash, LCD controller, serial, IrDA, digital I/O, RTC and matrix keyboard controller, plus ISA, VL and PCMCIA expansion buses. Other options include solid-state disk, floppy and IDE drive interfaces, parallel port and Ethernet[1]

[1] http://www.compulab.co.il

JUMPTec DIMM-PC/486 – a "DIMM form-factor" (2.7" x 1.6") 66MHz 486-based PC-compatible SBC. Includes: 16MB Flash and 16MB DRAM memory, plus interfaces for serial, parallel, floppy and IDE [2]

Single-Board Computers with ARM microcontrollers

ADS Bitsy – this 3" x 4" SBC is based on a 206MHz Intel StrongARM SA-1110 processor (plus SA-1111 companion chip) and consumes just 450 mW. Includes: serial, USB, audio, digital and analog I/O, a Type II PCMCIA slot, plus a 1024 x 1024 resolution color LCD controller[3]

intrynsic CerfBoard – a tiny (2.2" x 2.4") SBC based on a 133 or 206MHz Intel StrongARM 1110 CPU. Includes: up to 16MB Flash, up to 8MB SDRAM, 16 digital I/O lines, 10MB Ethernet, USB, serial port, audio CODEC, LCD interface and CompactFlash+ socket [4]

LART – an "open licensed", small SBC (4" x 3") design from the Technical University of Delft (The Netherlands). It is based on a 220MHz Intel SA-1100 StrongARM and includes: 4MB Flash and 32MB DRAM memory, serial and parallel ports, and bus expansion. Expansion boards provide: Ethernet, USB, keyboard, mouse, touch input and video[5]

5 Conclusion

Building a concept for an integrated node hosting a TTP/A and TCP/IP interface and a CORBA ORBit server is a demanding task. While embedded Linux as operating system enables the usage of ready-made software such as the CORBA ORBit, the real-time capabilities for high speed real-time systems might be insufficient. A circumvention of this problem could be the employment of an extra microcontroller running the TTP/A protocol. As a hardware platform, we recommend a Linux-friendly single-board computer. Using a ready made-board saves the costly and time-consuming task of designing a custom embedded computer. Since some vendors also provide an embedded Linux operating system adapted to their hardware, time and cost for porting the Linux software to non-PC-compatible hardware can also be saved.

References

[1] Object Management Group OMG. Smart transducers interface request for proposal. *OMG TC Document orbos/2000-12-13*, Dec. 2000. Available at http://www.omg.org.

[2] Objective Interface Systems, TTTech Computertechnik, and VERTEL Corporation. Smart transducers interface. *OMG TC Document orbos/2001-06-03*, July 2001. Supported by Technische Universitt Wien. Available at http://www.omg.org.

[3] G. Tyler. Linux is a great prototyping tool for embedded-hardware designs. *Electronic Design*, 47(13), June 1999.

[2] http://www.jumptec.de
[3] http://www.applieddata.net
[4] http://www.intrinsyc.com
[5] http://www.lart.tudelft.nl

[4] E. F. Hilton and V. Yodaiken. Real-time applications with rtlinux. *Embedded Linux Journal*, Jan. 2001.

[5] J. Sissom. Using compactflash cards in your embedded linux system. *Embedded Linux Journal*, May 2001.

[6] R. Kapeller. Design and implementation of a ttp/a master and gateway controller on a 32-bit microcontroller. Master's thesis, Technische Universität Wien, Institut für Technische Informatik, Vienna, Austria, 2001.

[7] P. Peti. Monitoring and configuration of a TTP/A cluster in an autonomous mobile robot. Master's thesis, Technische Universität Wien, Institut für Technische Informatik, Vienna, Austria, 2001.

[8] L. Schneider. Real time robot navigation with a smart transducer network. Master's thesis, Technische Universität Wien, Institut für Technische Informatik, Vienna, Austria, 2001.

[9] R. Obermaisser and A. Kanitsar. Application of TTP/A for the Otto Bock Axon bus. Technical Report 27, Technische Universität Wien, Institut für Technische Informatik, July 2000.

[10] R. Lehrbaum. All about linux-friendly single-board computers. *Embedded Linux Journal*, March 2001.