CAN-enabled network processor

D.Kenny, F.Morgan, G.Sweeney, M.Glavin, N.Murphy

This paper proposes extending the high performance Network Processor (NP) architecture to meet cost/performance requirements of new emerging high speed and higher cost networks for the automation and automotive industries. The use of NP architectures for implementation of IDB-1394, MOST and Power Ethernet networks, which commonly embed CAN bus control elements, is proposed.

The paper describes the implementation of a CAN enabled NP, incorporating a CAN co-processor within an Intel IXP NP hardware emulation system, and supporting the CANopen protocol. The proposed NP would enable packet transfer between CAN-bus and a range of existing NP communication protocols, including Ethernet, HSS, USB, PCI and UART.

1 Introduction

A Network Processor (NP) is a high performance, configurable and multi-protocol communications packet processor optimised for use in networking applications.

This paper proposes the integration of emerging high-speed in-car networks, e.g., MOST (Media Oriented Systems Transport) and IDB-1394 (Intelligent Data Bus) within a NP architecture.

Likewise, the automation industry offers a range of applications, e.g., a communication upgrade platform for existing CAN installations, providing an interface between legacy CAN installations and recently introduced networks such as Ethernet.

This paper describes the design and integration of a CAN co-processor into a NP. A CAN enabled NP hardware implementation, running CANopen, has been designed and verified. This platform enables integration of developing high performance protocols which commonly embed CAN bus control elements or link with legacy CAN devices.

The structure of the paper is as follows: Section 2 introduces Network Processors. Section 3 outlines a range of potential applications of NPs in automotive and automation applications.. Section 4 describes the IDB and MOST bus protocols. Section 5 details the design of the CAN co-processor. Section 6 describes the implementation of higher layer CANopen software within the NPE. Section 7 describes the hardware implementation and system verification. Section 8 concludes the paper.

2 Network Processors

NPs employ multiple Network Processing Engines (NPE) to perform packet-processing tasks concurrently. NPEs in turn use concurrent hardware networking co-processors to reduce their workload. Each co-processor is independently controllable from the NPE RISC core. The mix of core and co-processors provide unique configurability.

High growth in the application of NPs is predicted for future networking systems, requiring a high degree of flexibility to support evolving networking services. Over the period of second half 2002 and first half 2003, NP revenue was estimated at $61M. The forecast for 2006 is $190M [1].

NP vendors have implemented a variety of hardware assist engines (co-processors),

either hardwired or configurable, having a limited programming interface and normally used for tasks such as table lookup, encryption and communication bus interfacing.

3. NP Applications

NPs offer applications in high performance industrial automation and automotive networks.

The performance model of NPs is in excess of that required by CAN-bus-only applications. However, a number of high performance network protocols are outlined, including IDB, MOST and Power Ethernet which commonly embed CAN bus control elements or link with legacy CAN devices.

3.1 NPs in Industrial Automation

In automation a CAN enabled NP could offer a communication upgrade platform for existing CAN installations, providing an interface between legacy CAN installations and networks newly introduced to automation such as Ethernet and Power Ethernet. The NP would also offer CAN bridge applications to factories using two or more CAN-buses running at different wire speeds.

3.2 NPs in Automotive Networks.

Figure 1 illustrates a typical in-car network, including applications ranging from diagnostics and control via CAN-bus to high performance multi-media.

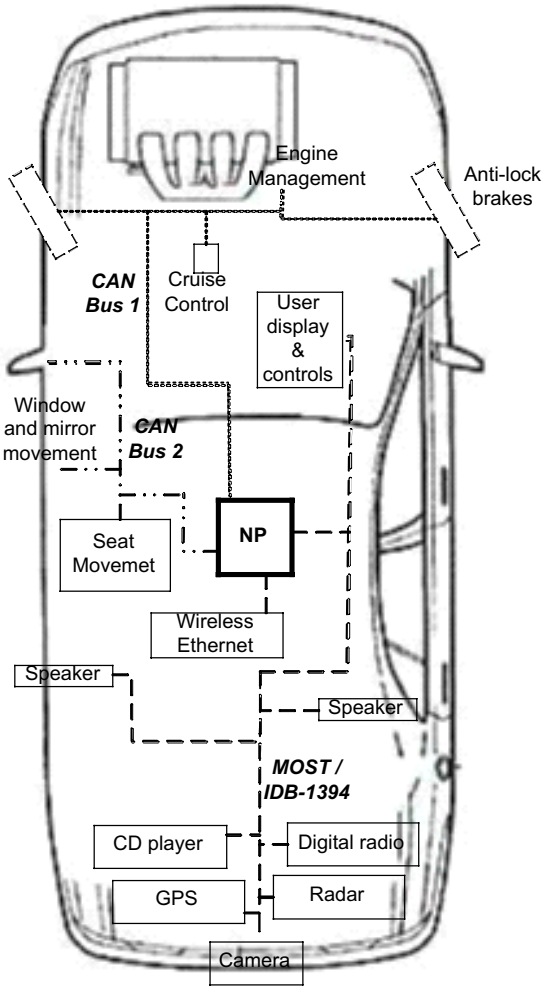

Fig 1 In-car network elements with NP core

No currently developed single in-car bus can adequately handle the entertainment, safety, and intelligent-control equipment of future cars. New driver assistance and telematics systems have increased the annual growth rates of in-car electronics to 16%, according to the IEEE (as quoted in Xilinx Xcell Journal [2]). This also forecasts that electronics would account for 25% of the cost of a mid-size car in 2005.

The highest growth area is telematics systems (the convergence of mobile telecommunications and information processing).

Telematics applications exhibit market conditions of short time to market with changing standards / protocols and high bandwidth. New market conditions differ from the long design cycles of traditional in-car electronics.

The developing IDB-1394 standard supports audio and video and includes a CAN-based control network communicating commands and feedback information. With the changes in market conditions, NPs have been considered for in-car networks.

NPs provide :

- flexibility of a programmable device
- high performance of a fixed function hardware device
- communication-specific functions more efficiently than general-purpose processors (while still providing considerable system flexibility along with high-performance hardware packet processing functions)
- reduced validation and fast time to market
- interoperability
- enhanced system integration
- reuse of software over generations of hardware co-processor design

Extending existing NP architectures to support protocols spanning from CAN-bus to MOST (Media Oriented Systems Transport) and IDB, offers wide application.

A CAN enabled NP emulation system has been developed offering support of packet transfer between CAN-bus and a range of existing NP communication protocols, including Ethernet, HSS, USB, PCI and UART.

This work has been extended to consider incorporation of a NP-based MOST and IDB to CAN-bus gateway.

4 IDB and MOST Bus Protocols

Two major automotive bus protocols, MOST and IDB are considered for future integration onto NP systems. Sections 4.1 and 4.2 describe these protocols.

4.1 MOST Bus

MOST (Media Oriented Systems Transport), an open standard for infotainment networks is based on the D2B [3] optical media network physical layer. It defines the protocol, hardware, and software layers necessary to allow for transport of control, real-time, and packet data at speeds from a few kbps up to 24.8Mbps. MOST offers full support for real-time audio and compressed video. Several European automotive manufacturers are considering MOST as a potential common standard and implementing MOST as an internal vehicle network.

4.2 IDB Bus

The developing IDB-1394 (Intelligent Data Bus) standard, will support audio/video and include a CAN-based control network.

The IDB forum manages the IDB-C and IDB-1394 buses and standard IDB interfaces.

4.2.1 IDB-C Bus

Based on CAN bus, IDB-C targets devices with data rates of 250 kb/s. The reported CAN enabled NP can be further developed to support IDB-C.

4.2.2 IDB-1394 Bus

The IDB-1394 specification is an automotive supplement to the existing IEEE 1394 (Firewire) standards, already well established in the consumer electronics industry. IDB-1394 supports consumer multimedia applications for in-vehicle networking. Existing 1394 portable devices can be plugged directly into the new automotive multimedia bus using an IDB-1394 defined customer convenience port. The IDB forum announced (January 2005 [4]) IDB-1394 as the first automotive networking technology to be approved by the DVD Copy Control Association (DVD-CCA) for distributing copy-protected digital video content on a localised digital network.

This represents a major step forward in bringing high quality all-digital DVD content to vehicles, clearing the way for IDB-1394 technology to be used in end-to-end digital video applications.

5 CAN Co-processor Design

Figure 2 illustrates the CAN co-processor architecture and interfaces (NPE and CAN-bus transceiver).

The implementation of a CAN enabled NP incorporating a CAN-bus co-processor within an Intel IXP [5] NP hardware emulation system, and supporting the CANopen protocol, is described.

The NPE RISC core regulates communication between various embedded communication co-processors through a defined instruction set. The Co-processor Bus Interface handles data exchange between the NPE and co-processors, and also directly between co-processors.

Twelve CAN co-processor instructions are defined within the NPE, e.g.,

1. Write standard frame format message header
2. Write timing registers.

These instructions are coded into 4-bit values on the Co-processor Bus Interface.

Figure 3 illustrates the internal architecture of the CAN co-processor. This includes a CAN Controller and a CAN Interface block.

The CAN Controller incorporates a modified CAN Controller (Saytam [6]) and supports CAN 2.0A and 2.0B, with acceptance filtering. The CAN Controller contains transmitter and receiver units. The former incorporates CRC generator and bit-stuffer. The receiver block performs CAN bus synchronisation, de-serialisation, de-stuffing and error detection of the incoming CAN frame.

The CAN Interface block is designed to interface the CAN controller and the NPE. It decodes the instructions and manages the data accordingly.

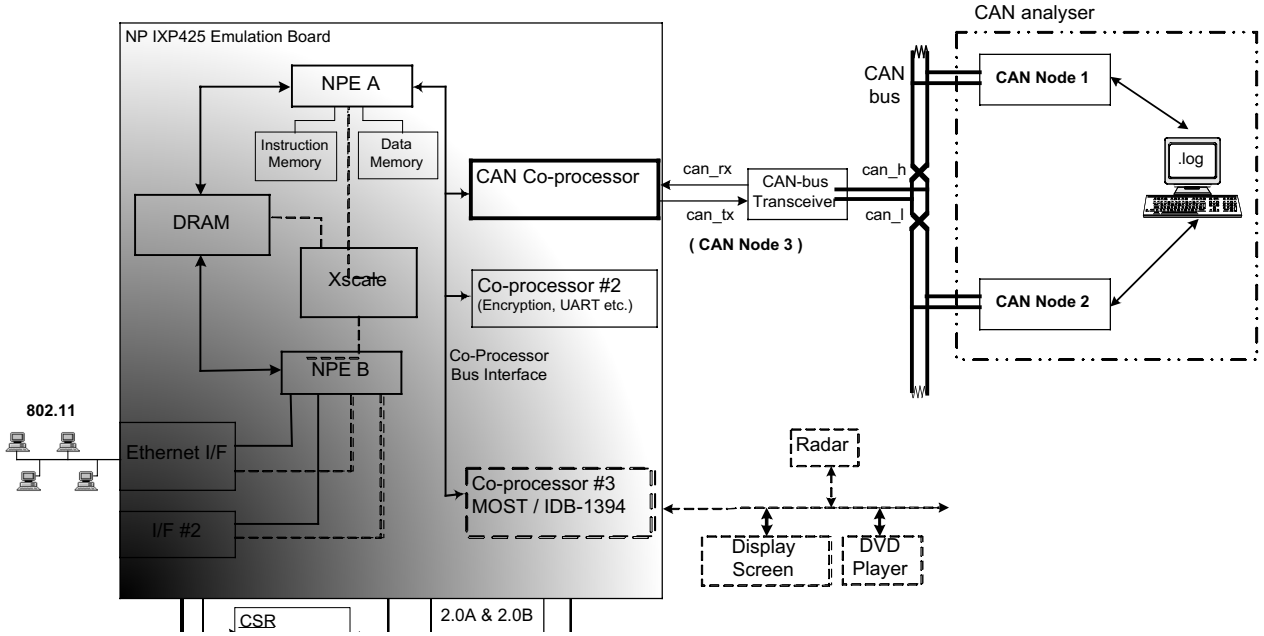

Fig 2 CAN co-processor architecture and interfaces to NPE and CAN bus

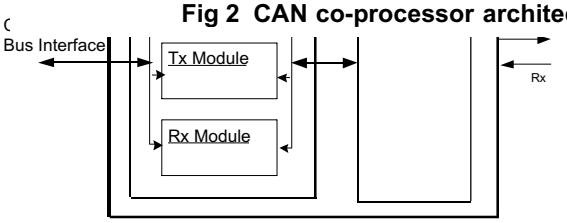

Fig 3 CAN co-processor internal architecture

8-byte message buffering between NPE and CAN Controller. The CAN Interface module consists mainly of three sub-modules: CSR, Transmit Module and Receive Module.

5.1 Control & Status Register (CSR)

The CSR stores configuration information defined by the NPE, along with diagnostics for the CAN Controller. On completion of the configuration phase, the CAN co-processor transfers messages to and from the NPE for subsequent re-routing or processing by the NP.

Configuration information is stored in the Timing Register and the Acceptance Register. When executing the configuration of the CAN co-processor, the NPE usually assigns values to these registers. The CSR also contains an Error Code register. From which the NPE can determine what type of error has occurred (Bit, Stuff, Form etc), position of the error in the CAN bit stream and whether the error occurred during reception or transmission of a message. The Error Code Register, stores a Received Error Count and Transmission Error Count, from which the NPE can determine CAN co-processor operation mode (Error Active, Error Passive or Bus Off).

5.2 Transmit Module

The Transmit Module decodes and bundles data from the NPE for output on the CAN Bus. The NPE instructs this module to transmit data to the CAN Controller. No messages are stored within the CAN Controller. The NPE stores all co-processor data (within data memory). The Transmit Module informs the NPE when the CAN bus is free. The NPE transfers message data from memory to the Transmit Module.

5.3 Receive Module

The CAN Controller bundles received messages for storage in the Receive Module. The Receive Module informs the NPE which reads the stored data. The receive module buffers only one CAN message since the NPE operates at clock speeds up to 200MHz while the CAN controller operates at 20MHz.

6 CANopen Implementation on CAN enabled NP

The CAN enabled NP hardware implementation described supports first and second layers of the OSI reference model. DeviceNet, SAE J1939, CANKingdom and CANopen were considered to support higher layers.

DeviceNet is used mainly for industrial applications as a low cost solution linking industrial devices to eliminate expensive hard wiring [7].

SAE J1939 was developed for in-vehicle communication in trucks, and has been adopted by other vehicles such as agricultural and forestry vehicles.

CANKingdom is mainly applied to machine control, e.g., industrial robot control, weaving machines, mobile hydraulics etc.

CANopen was initially developed for motion-oriented machine control networks and now has broad application, including medical equipment, off-road vehicles, building-automation etc.

CANopen was selected for implementation on the NP due to its flexible configurability. In addition, Power Ethernet protocol, which is incorporated within the NP, is proven to work well with CANopen.

The CANopen Object Dictionary describes the complete operation of a device by way of communication objects. Communication objects include:

- Process Data Objects (PDO)
- Service Data Objects (SDO)
- Network Management Objects (NMT)

The Object Dictionary for the NP is stored in NPE data memory.

To best utilise NPE hardware features, a low-level programming language has been used to program the NPE. Once compiled the code is downloaded into the NPE's Instruction memory(shown in figure 4).

The availability of a threading mechanism within the NP simplifies the implementation of the CANopen protocol.

A Conditional Co-processor is used during initialisation to configure NPE hardware threading, and reduce NPE workload.

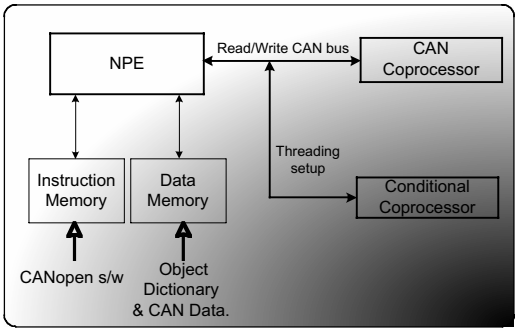

Fig 4 NPE CAN and conditional co-processors

High priority threads are used to service the physical interface with the CAN co-processor, responding to signals such as, Receive Module full or Transmit Module empty. Other high-priority threads are set to trigger on bus errors and network management objects

On receipt of a message from the CAN co-processor, the NPE stores the message in its data memory. CANopen software in the NPE checks the received message type identifier and sets a related flag in the NPE. This in turn triggers a lower priority thread that deals with the message accordingly.

Examples of lower priority threads are:

SDO Download : receives information to change node configuration, writing this information to the Object Dictionary

Receive PDO : depending on message ID, sends message to DRAM for transmission on another interface, or performs some processing and resends onto CAN bus

Heartbeat Consumer : monitors heartbeats of other nodes.

7 Hardware Implementation & System Verification

The CAN co-processor Verilog model simulation testbench incorporates a NPE Bus Functional Model that facilitates extensive verification using Modelsim.

Intel's IXP NP hardware emulator system incorporating the CAN co-processor has been synthesised and implemented using three Xilinx Virtex XCV3200E devices connected to an ARM processor. The CAN transceiver daughter card connects physical CAN devices to the CAN enabled NP system emulator using a Microchip 2551 High-Speed CAN Transceiver (Figure 1).

Extensive hardware prototype testing has been performed in a close to real world environment incorporating a two-node CAN analyser [8], configured to send messages on node 1 (with identifier). The CAN enabled NP receives node 1 data and re-transmits to CAN analyser node 2.

Messages received by the NPE, addressed to other interfaces or networks, (such as Ethernet or USB), are stored in DRAM and the NPE instructs the Xscale processor to process this data.

Messages in DRAM for transmission on the CAN bus are read by the NPE and transferred to the CAN co-processor for transmission onto the CAN bus. Figure 1 illustrates the fully implemented system.

The system has also been verified using the NPE configured as master of a three node CANopen network. The NP modifies the object dictionary of the two CAN analyser nodes using SDOs. Various CANopen communication objects are sent and received by the NP.

A typical vehicle can contain two or three separate CAN buses operating at different transmission rates (shown in Figure 1). Low speed CAN, operating at less than 125kb/s, manages body control electronics, such as seat and window movement controls. Higher speed (up to 1Mb/s) CAN operate more real-time critical functions such as engine management, antilock brakes and cruise control.

The hardware implementation of a NP-based CAN bridge is in progress using a NP with multiple CAN co-processors to enable message transfer between high and low speed CAN buses.

8 Conclusions and Future Work

The paper proposes the application of NPs to meet cost/performance requirements of new emerging high speed and higher cost networks for the automation and automotive industries.

The performance model of NPs is in excess of that required by CAN-bus-only applications.

A number of network protocols have been outlined and proposed, including IDB, MOST and Power Ethernet. These networks commonly embed CAN bus control elements.

This paper describes the design and integration of a CAN co-processor into a NP. A CAN enabled NP hardware implementation, running CANopen, has been designed and verified. This platform also enables integration of developing high performance protocols with CAN bus protocols on a variety of both legacy and new applications.

Extension of this work to CAN gateway, MOST and IDB implementations is now being considered

References:

[1] Chris Rosewarne, "Network Processors: Evaluating Architectures for Leading Edge Applications", TechOnline White Pape, March 2004, www.techonline.com

[2] Karen Parnell, "Put the right bus in your car", Xilinx Xcell Journal, December 2003

[3] Thomas Kibler, Stefan Poferl, Gotthard Böck, Hans-Peter Huber, and Eberhard Zeeb, "Optical Data Buses for Automotive Applications". Journal of Lightwave Technology, Vol. 22, Nn. 9, Sept 2004

[4] "IDB-1394 becomes the first Digital Automotive Network approved to carry "Localized" DVD video", Jan 2005, www.idbforum.org

[5] Intel IXP425 NP, Data Sheet, ftp://download.intel.com/design/network/datashts/25247904.pdf

[6] Satyam Computer Services Ltd. CAN IP Core http://www.satyam.com/solutions/s_tsilides.html

[7] CAN In Automation "CAN higher layer protocols". http://www.can-cia.org/

[8] Vector Informatik "CANalyzer"

http://www.vector-informatik.com

Author 1:	David Kenny
Company:	NUI, Galway
Address:	Electronic Engineering Dept.
Phone:	+353 91 493137
Fax:	+353 91 494511
E-mail:	david.m.kenny@nuigalway.ie
Website:	www.ee.nuigalway.ie
Author 2:	Fearghal Morgan
Company:	NUI, Galway
E-mail:	fearghal.morgan@nuigalway.ie
Author 3:	Gerard Sweeney
Company:	NUI, Galway
Author 5:	Martin Glavin
Company:	NUI, Galway
Author 4:	Noel Murphy
Company:	Intel Communications Europe
Address:	Shannon, Co.Clare, Ireland