IEEE TRANSACTIONS ON

CONSUMER ELECTRONICS

NOVEMBER 2008 VOLUME 54 NUMBER 4 ITCEDA (ISSN 0098-3063)

A Publication by the IEEE Consumer Electronics Society

Administrative Committee i
Officers and Committee Chairmen ii

CAMERA/DISPLAY/TRANSDUCER SYSTEMS

Suppressing Rolling-Shutter Distortion of CMOS Image Sensors by Motion Vector Detection ... *Jung-Bum Chun, Hunjoon Jung and Chong-Min Kyung* 1479
A Paperless Fax Machine with a Single-Touch Panel ..... *Cheng Wen and Chih-Hung Huang* 1488
Lossless and Near Lossless Compression of Real Color Filter Array Data ..... *Andriy Bazhyna and Karen Egiazarian* 1492
Design and Implementation of a Scanner with Stitching of Multiple Image Capture ..... *Ying-Wen Bai and Chien-Chung Cheng* 1501
Real-Time Face-Priority Auto Focus for Digital and Cell-Phone Cameras ..... *Mohammad T. Rahman and Nasser Kehtarnavaz* 1506
Dynamic 3D Human Actor Generation Method using a Time-of-Flight Depth Camera ..... *Ji-Ho Cho, Sung-Yeol Kim, Yo-Sung Ho and Kwan H. Lee* 1514
Reduction of Halftoning Errors due to Line Load in a Plasma Display Panel ..... *Jin-Sung Kim and Hyuk-Jae Lee* 1522
Color Interpolation by Expanding a Gradient Method ..... *Se-Hwan Yun, Jin Heon Kim and Suki Kim* 1531

COMMUNICATIONS

Interactive Data Broadcasting Services based on Middleware Technology in T-DMB ... *Gwang Soon Lee, Kwang-Yong Kim, Nam-ho Hur and Soo In Lee* 1540
Enabling Dual Branch Switched Diversity in Compact Wireless Devices ..... *Gill R. Tsouri, Dov Wulich and Lev Goldfeld* 1545
A Novel Symbol Synchronization Method for OFDM Systems in SFN Channels ..... *Junling Zhang* 1550
Reduced Front-End Reception Requirements for Satellite Broadcast using Interference Processing ..... *Enrico Casini, Gennaro Gallinaro and Joel Grotz* 1555
A Memory-Reduced Direct Digital Frequency Synthesizer for OFDM Receiver Systems ..... *Xiaojin Li, Linhui Lai, Ao. Lei and Zongsheng Lai* 1564
Frequency Domain Feed-forward Filter Combined DFE Structure in Single Carrier Systems over Time-varying Channels ..... *Bo Liu, Lin Gui, Wenjun Zhang and Jian Xiong* 1569
Robust Channel Estimation Scheme for the TDS-OFDM Based Digital Television Terrestrial Broadcasting System ... *Zi-Wei Zheng and Zhong-Gao Sun* 1576
Novel Channel Estimation Method Based on PN Sequence Reconstruction for Chinese DTTB System ..... *Fang Yang, Jintao Wang, Jun Wang, Jian Song and Zhixing Yang* 1583
Design of a Mixed Prime Factor FFT for Portable Digital Radio Mondiale Receiver ..... *Dong-Sun Kim, Sang-Seol Lee, Jae-Yeon Song, Kyu-Yeul Wang and Duck-Jin Chung* 1590
Robust Frequency Synchronization Scheme for Digital FM Broadcasting Systems Using Cyclic Delay Diversity ..... *Eu-Suk Shim, Man-Geun Cho and Young-Hwan You* 1595
Timing Estimation for OFDM Systems by using a Correlation Sequence of Preamble ..... *Yeonsu Kang, Sooyoung Kim, Doseob Ahn and Hojin Lee* 1600
Layered Resource Allocation for Video Broadcasts over Wireless Networks ..... *Junu Kim, Jinsung Cho and Heonshik Shin* 1609
Low-Cost Reconfigurable VLSI Architecture for Fast Fourier Transform ..... *Hao Xiao, An Pan, Yun Chen and Xiaoyang Zeng* 1617
Combinational Spreading and Modulation Technique for Mobile Satellite DMB Services ..... *Sang-Jin Lee, Sangwoon Lee and Jong-Soo Seo* 1623
Cooperative Diversity Technique for MIMO-OFDM Uplink in Wireless Interactive Broadcasting ..... *Myung-Sun Baek and Hyoung-Kyu Song* 1627
A Dual Combining Based Decoding Scheme for Two-Branched Mobile STBC ..... *Jungwook Wee, Wongi Jeon, Kyungwon Park, Hyunsik Kim and Yongsoo Cho* 1635
Enhancing MB-OFDM Throughput with Dual Circular 32-QAM ..... *Runfeng Yang and R. Simon Sherratt* 1640
A Parallel Convolutional Coder Including Embedded Puncturing with Application to Consumer Devices ..... *R. Simon Sherratt* 1647

NETWORKS

UPnP IPv4/IPv6 Bridge for Home Networking Environment ..... *Chung-Sheng Li, Yueh-Min Huang and Han-Chieh Chao* 1651
The DOG Gateway: Enabling Ontology-based Intelligent Domotic Environments ..... *Dario Bonino, Emiliano Castellina and Fulvio Corno* 1656
VoIP and IPTV Distribution over Wireless Mesh Networks in Indoor Environment ..... *Mikael Gidlund and Johan Ekling* 1665
Three-Dimensional Absorbing Markov Chain Model for Video Streaming over IEEE 802.11 Wireless Networks ... *Azfar Moid and Abraham O. Fapojuwo* 1672
Improvements in Home Automation Strategies for Designing Apparatus for Efficient Smart Home ..... *K. Balasubramanian and A. Cellatoglu* 1681
A Wireless Power Outlet System for Smart Homes ..... *Guangming Song, Fei Ding, Weijuan Zhang and Aiguo Song* 1688
Architecture of Home Gateway for Device Collaboration in Extended Home Space ..... *Hojin Park, Ilwoo Lee, Taein Hwang and Nam Kim* 1692
An Evaluation Method for Dynamic Combination among OSGi Bundles based on Service Gateway Capability ..... *Jung-Eun Lim, O-Hoon Choi and Doo-Kwon Baik* 1698
Error Concealment Aware Streaming Video System over Packet-based Mobile Networks ..... *Jae-Young Pyun* 1705
Flexible and Reliable Transmission Techniques for Scale-free UWB System ..... *So-Young Yeo and Hyoung-Kyu Song* 1714
Design and Implementation of a Universal Appliance Controller Based on Selective Interaction Modes ..... *Hyoseok Yoon and Woontack Woo* 1722
A Storage Saving Scheme to Share HD-Quality Content in Community Networks ..... *Sungwook Chung, Eunsam Kim and Jonathan C. L. Liu* 1730

(Continued on back cover)

CONTENTS *(Continued from front cover)*

Provision of the Multimedia Service Framework in the Ubiquitous Home Network
Jung-Tae Kim, Sangwook Park, Jong-Hoon Lee, Eui-Hyun Paik and Kwang-Roh Park 501
Novel Bridge for Networking IEEE 1394 Clusters via UWB over Coaxial Cable for HANA & AV/C Digital Multimedia Networks
Sergio Guillén Servent, Michal Duzynski, Petra Nauber, Michael Scholles and Uwe Schelinski 507
RECORD/PLAYBACK/STORAGE SYSTEMS
Fast Play: A Novel Feature for Digital Consumer Video Devices *Marco Furini* 513
A Video Summarization Tool using Two-Level Redundancy Detection for Personal Video Recorders *Yue Gao, Wei-Bo Wang and Jun-Hai Yong* 521
Convenient Teletranscription Method based on EIT in Video Digital Recorder *O-Hoon Choi, Chang-Hun Sung, Jung-Eun Lim, Hong-Seok Na and Doo-Kwon Baik* 527
The Design of Video Segmentation-aided VCR Support for P2P VoD Systems *Xin Wang, Changyi Zheng, Zhenyuan Zhang, Hong Lu and Xiangyang Xue* 531
A Controller Design Method for Constructing a Robust Track-Following System *Moon-Noh Lee and Kyoung Bog Jin* 538
Adaptive Fuzzy Logic for Focusing in Optical Disk Player *Yong C. Choi and Chang-Hun Kim* 545
Replacement and Swapping Strategy to Improve Read Performance of Portable Consumer Devices Using Compressed File Systems
Ohhoon Kwon, Yunjung Yoo, Kern Koh and Hyokyung Bahn 551
SECURITY SYSTEMS
Secure Media Content Distribution Based on the Improved Set-Top Box in IPTV *Shiguo Lian and Zhongxuan Liu* 560
A Low Cost GSM/GPRS Based Wireless Home Security System *Yanbo Zhao and Zhaohui Ye* 567
On the Design of an Embedded Biometric Smart Card Reader *Dong-Sun Kim, Seung-Yerl Lee, Byung-Soo Kim, Sung-Chul Lee and Duck-Jin Chung* 573
SIGNAL PROCESSING
Using Active and Passive RFID Technology to Support Indoor Location-Aware Systems *R. Tesoriero, J. A. Gallud, M. Lozano and V. M. R. Penichet* 578
Robust Mandarin Speech Recognition in Car Environments for Embedded Navigation System *Pei Ding, Lei He, Xiang Yan, Rui Zhao and Jie Hao* 584
A Flying-Adder PLL Technique Enabling Novel Approaches for Video/Graphic Applications *Liming Xiu* 591
An Integrated Architecture for Adaptive Image Stabilization in Zooming Operation *Angelos Amanatiadis and Ioannis Andreadis* 600
Research of Material Properties Reliance for Numerical Analysis *Tohru Nakanishi* 609
Lightweight Content-Adaptive Coding in Joint Analyzing-Encoding Framework *Yayu Zheng, Fan Zhou, Xiang Tian and Yaowu Chen* 614
Micro Power Battery State-of-Charge Monitor *Tanvir Singh Mundra and Ajay Kumar* 623
An Efficient and Advanced Space-management Technique for Flash Memory using Reallocation Blocks *Se Jin Kwon and Tae-Sun Chung* 631
Real-Time DVB-MHP to Blu-ray System Information Transcoding *Zicong Mai, Panos Nasiopoulos, Rabab K. Ward and Sergio Infante* 639
Design Considerations of Channel Buffer Amplifiers for Low-power Area-efficient Column Drivers in Active-Matrix LCDs *Young-Suk Son and Gyu-Hyeong Cho* 648
Dynamic Motion Estimation for Transcoding P Frames in H.264 to MPEG-2 Transcoders *Hari Kalva and Phil Kunzelmann* 657
Effective Bass Enhancement Using Second-Order Adaptive Notch Filter *Junho Lee, Eunjung Song, Youngcheol Park and Daehee Youn* 663
Backlight Power Reduction and Image Contrast Enhancement Using Adaptive Dimming for Global Backlight Applications *Chih-Chang Lai and Ching-Chih Tsai* 669
Memory Bandwidth Efficient Hardware Architecture for AVS Encoder *Dandan Ding, Shuo Yao and Lu Yu* 675
Efficient Pipelined CABAC Encoding Architecture *Wei Zheng, Dong-Xiao Li, Bing Shi, Hoang-Son Le and Ming Zhang* 681
A Novel VLSI Architecture of Motion Compensation for Multiple Standards *Junhao Zheng, Wen Gao, David Wu and Don Xie* 687
A Novel high-quality YUV-based Image Coding Technique for Efficient Image Storage in Portable Electronic Appliances *R. Mancuso, S. Smorfa and M. Olivieri* 695
Enhanced Transform Domain Intra Prediction for MPEG-2 to H.264/AVC Transcoding *Sang-Jun Yu, Jin-Su Myung, Dong-Gyu Sim and Seoung-Jun Oh* 703
Proposed Architecture and Algorithm for Personalized Advertising on iDTV and Mobile Devices *Toon De Pessemier, Tom Deryckere, Kris Vanhecke and Luc Martens* 709
Applying Data Carousel to Improve Search Efficiency in P2P System *Wu Guobin, Ni Hong, Wu Gang and Pan Liang* 714
A Sequence-Action Recognition Applying State Machine for User Interface *Jonghun Baek and Byoung-Ju Yun* 719
Providing Entertainment by Content-based Filtering and Semantic Reasoning in Intelligent Recommender Systems
Yolanda Blanco-Fernández, José J. Pazos-Arias, Alberto Gil-Solla, Manuel Ramos-Cabrer and Martín López-Nores 727
Layer-Weighted Unequal Error Protection for Scalable Video Coding Extension of H.264/AVC *Hojin Ha and Changhoon Yim* 736
Scalable Multiple Description Video Coding for Stereoscopic 3D *H. A. Karim, C. T. E. R. Hewage, S. Worrall and A. M. Kondoz* 745
Low-Complexity Video Error Concealment for Mobile Applications Using OBMA *Tanaphol Thaipanich, Ping-Hao Wu and C.-C. Jay Kuo* 753
Saturation Enhancement of Blue Sky for Increasing Preference of Scenery Images *Ju-Yeon You and Sung-Il Chien* 762
Hardware Architecture for AVS Entropy Encoder *Long Xu, Lei Deng, Xiangyang Ji, Xiaoming Peng and Wen Gao* 769
A Display-Mounted High-Quality Stereo Microphone Array for High-Definition Videophone System
Yusuke Hioka, Manabu Okamoto, Kazunori Kobayashi, Yoichi Haneda and Akitoshi Kataoka 778
Low Complexity, Efficient and Embedded Color Image Coding Technique *Athar Ali Moinuddin, Ekram Khan and Mohammed Ghanbari* 787
Evaluation of the Interference of Pulsed UWB Signals over Current Services in CATV Installations *Julen Ugalde, Iñaki Val and Pedro Crespo* 795
Fast Speech Recognition to Access a Very Large List of Items on Embedded Devices *Hoon Chung, Jeon Gue Park, Yun Keun Lee and Ikjoo Chung* 803
Low Power H.264 Deblocking Filter Hardware Implementations *Mustafa Parlak and Ilker Hamzaoglu* 808
Trick Play Method for HD H.264 Set-Top Box *Jin-Hwan Jeong, Yong-Ju Lee, Hag-Young Kim and Myung-Joon Kim* 817
Feature and Noise Adaptive Unsharp Masking Based on Statistical Hypotheses Test *Yeong-Hwa Kim and Yong Jun Cho* 823
Real-Time Highlight Detection in Baseball Video for TVs with Time-Shift Function *Hyoung-Gook Kim, Jinguk Jeong, Jang-Heon Kim and Jin Young Kim* 831
TV Settop Box Design for Processing Text Messages Based on Graph Theory *Cheng Wen, Chih-Hung Huang, Ming-Feng Yeh and Kuang-Chiung Chang* 839
A Flexible Template for H.264/AVC Block Matching Motion Estimation Architectures
Sebastián López, Gustavo M. Callicó, Félix Tobajas, José F. López and Roberto Sarmiento 845
Sound Source Localization with the Aid of Excitation Source Information in Home Robot Environments *Keun-Chang Kwak and Sung-Suk Kim* 852
Fine Directional De-interlacing Algorithm Using Modified Sobel Operation *Soonjong Jin, Wonki Kim and Jechang Jeong* 857
A Novel Spatial and Temporal Correlation Integrated Based Motion-Compensated Interpolation for Frame Rate Up-Conversion
Yang Ling, Jin Wang, Yunqiang Liu and Wenjun Zhang 863
The Design of a Speech Interactivity Embedded Module and its Applications for Mobile Consumer Devices
Jhing-Fa Wang, Jia-Ching Wang, Ming-Hua Mo, Chuan-I Tu and Shun-Chieh Lin 870
Computationally Efficient Mode Selection in H.264/AVC Video Coding *Jianfeng Ren, Nasser Kehtarnavaz and Madhukar Budagavi* 877
Joint Source and Channel Coding for 3D Video with Depth Image-Based Rendering *B. Kamolrat, W. A. C. Fernando, M. Mrak and A. Kondoz* 887
Design of an Efficient Scalable Vector Graphics Player for Constrained Devices *Cyril Concolato, J. Le Feuvre and J.-C. Moissinac* 895
Tone-Mapping Functions and Multiple-Exposure Techniques for High Dynamic-Range Images *S. Cvetković, J. Klijn and P. H. N. de With* 904
Reducing IPTV Channel Switching Time Using H.264 Scalable Video Coding *Yonghee Lee, Jonghun Lee, Inkwon Kim and Heonshik Shin* 912
Highly Efficient Transmission System for Digital Broadcasting Redistribution Services over IP Multicast Networks
Tetsuya Yamaguchi, Tomoyuki Kanekiyo, Motoyuki Horii, Katsuhiko Kawazoe and Fumio Kishino 920
System Architecture for Free-Viewpoint Video and 3D-TV *Yannick Morvan, Dirk Farin and Peter H. N. de With* 925
SOFTWARE FOR CONSUMER PRODUCTS
VLIW-Aware Software Optimization of AAC Decoder on Parallel Architecture Core DSP (PACDSP) Processor *Tsung-Han Tsai, Chun-Nan Liu and Jui-Hong Hung* 933
Autonomic System Management for Personal Digital Assistants *Yong-Duck You, Hoon Choi and Dongwon Han* 940

Seamlessly Interconnecting Legacy IEEE 1394 Devices over WiMedia UWB Network: The Mirroring Bridge

Jonghun Yoo, Jiyong Park, Seongsoo Hong, *Member*, IEEE

Abstract — *Bridging IEEE 1394 buses is becoming important since it can be used to provide wireless connectivity among 1394 devices. Unfortunately, existing bridge mechanisms, such as the IEEE 1394.1 bridge and the transparent bridge, have practical limitations. The former does not support interoperability with legacy 1394 devices and the latter requires a new hardware chipset for bridge implementation. We thus propose a new bridge mechanism called a mirroring bridge to overcome these limitations. It supports interoperability with legacy 1394 devices by emulating remote nodes inside a bridge and via packet address translation that can be implemented through software. We have implemented the proposed bridge mechanism and have succeeded in interconnecting legacy 1394 devices over an experimental WiMedia UWB network. The experimental result showed that the average throughput of the mirroring bridge is 188.7 Mbps, which is 94.4% of the maximum throughput of the UWB chipset used.*[1]

Index Terms — 1394, FireWire, bridging, mirroring bridge, wireless, UWB, WiMedia

I. INTRODUCTION

The IEEE 1394 bus [1] has become the de facto standard for wiring high-speed AV devices in a home network environment. For instance, almost all digital camcorders on the market support 1394 interfaces. Such widespread use of the 1394 bus is due to its distinctive features, such as high bandwidth, plug-and-play and QoS guarantee.

As the home network market is rapidly switching to the wireless domain, the demand for wireless connectivity among 1394 devices is also greatly increasing. In response to this, Wireless Firewire was proposed by The 1394 Trade Association in 2004 [2]. It enables the 1394 protocol to operate over the IEEE 802.15.3 High Rate Wireless Personal Area Network (HR-WPAN) [3] while its programming interface remains intact [4]. Unfortunately, Wireless Firewire has a limited applicability in that it can only be applied to newly developed 1394 devices since it requires that the PHY and LINK layers be modified from the original 1394 standard.

Another viable and practical way to support wireless connectivity among 1394 devices is to use bridges. Bridging is a well known technique for transparently interconnecting different types of network segments in the data link layer [5]. Using this technique, one can achieve wireless connectivity as follows. First, 1394 devices in proximity are wired and thus form a separate 1394 network segment. Then, multiple network segments are linked via a wireless network. In each 1394 network segment, there is a bridge device that serves as a gateway to the wireless network. It performs protocol conversion and address translation between a 1394 network segment and the wireless network.

Up until now, there have been two bridge mechanisms for 1394 devices: the IEEE 1394.1 bridge [6] and the transparent bridge of Philips [7]. Unfortunately, they also possess practical limitations. The former loses interoperability with legacy 1394 devices since it requires nontrivial modifications to the PHY and LINK layers of the original 1394 standard. The latter supports the desired interoperability but requires a new custom hardware chipset for the implementation of a bridge device.

In this paper, we propose a new bridge mechanism to address these limitations. We call it a *mirroring bridge*. It employs two techniques. First, in a local 1394 network segment, our bridge device emulates all the other 1394 devices in remote network segments. Thus, a local 1394 device can access remote 1394 devices as if they were directly connected by the same network segment. This guarantees interoperability with legacy 1394 devices. Second, our bridge device performs address mapping so that it can assign arbitrary addresses to virtual devices that emulate remote 1394 devices. It is necessary to support arbitrary address assignment in order to maintain compatibility with the self-ID process of the original 1394 standard. To perform the above address mapping, our bridge device maintains a mapping table and translates packet addresses using this table. Since this mechanism can be implemented entirely through software, our mirroring bridge can be constructed with an existing 1394 chipset.

We have implemented the proposed bridge mechanism in our bridge devices and used them to interconnect legacy 1394 devices via a WiMedia UWB network [8], [9]. A bridge device is equipped with a 400-MHz PowerPC processor, 1394 interface cards and a WiMedia UWB interface card. Linux version 2.6.18 was used as the operating system and all of the bridging functionalities were implemented as three kernel modules. Using the bridge devices, we constructed a wireless network of nine legacy 1394 devices including HDTVs,

[1]This paper is an updated and extended version of the paper "Seamlessly Interconnecting Legacy IEEE 1394 Devices over UWB Network: The Mirroring Bridge" that appeared in the 26th IEEE International Conference on Consumer Electronics 2008.

The work reported in this paper is supported in part by Digital Solution Center, Samsung Electronics, Co. Ltd.

Jonghun Yoo, Jiyong Park and Seongsoo Hong are with the School of Electrical Engineering and Computer Science, Seoul National University, Korea (e-mail: {jhyoo, parkjy, sshong}@redwood.snu.ac.kr).

Contributed Paper
Manuscript received February 17, 2008

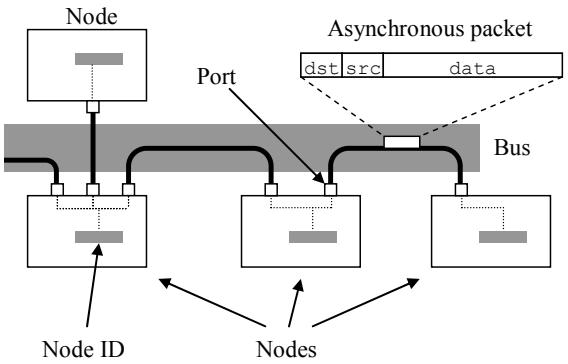


Fig. 1. Core elements of the IEEE 1394 protocol and their relationships.

digital camcorders, hard disk drives and set-top boxes. We performed a series of experiments to measure the performance of the wireless 1394 network. The average throughput of the network was 188.7 Mbps, which is 94.4% of the maximum throughput of the UWB chipset used. In addition, the average round-trip time of a packet measured 2.154 ms. Since the timeout of the round-trip time is set to 100 ms in the 1394 protocol, our mechanism does not cause multiple retransmissions of a packet that has been correctly received.

The remainder of this paper is organized as follows. In Section II, we give an overview of the 1394 protocol and then model the bridging problem as an address translation problem. In Section III, the existing two bridge mechanisms are reformulated with our address translation function so that their practical limitations can be easily understood. In Section IV, we introduce the key idea behind the mirroring bridge mechanism and show the packet handling process in the mechanism. Section V explains the architectural design of the bridge device. Section VI provides the results of an empirical evaluation of our mechanism in the experimental network. Finally, Section VII serves as our conclusion.

II. MODELING THE 1394 BRIDGING PROBLEM

To aid in understanding the rest of this paper, we first give an overview of the 1394 protocol and define the terms that are used throughout the paper. We then model the bridging problem as an address mapping problem. In doing so, we provide the logical and physical models of the target wireless 1394 network and formulate an address mapping function from the two models.

A. Overview of IEEE 1394 Protocol

Fig.1 shows the key elements of the 1394 protocol and the relationships among them. In this figure, a node denotes a distinguishable device that has a unique address called a node ID. A node has one or more sockets called ports. Two nodes can be directly connected with each other through a pair of ports using a cable. The nodes and cables physically form a tree topology. The network of cables can be considered a single bus since the ports internally act as repeaters. This bus is represented as a shaded square behind the cables in Fig.1.

The maximum number of nodes that can be connected via a single 1394 bus is 63 because the size of the node ID is 6 bits and node ID 63 is reserved for a broadcast address [1].

A node is assigned a node ID via a distributed process called the self-ID process. This process is automatically performed by the 1394 chipset when the bus is initialized and a device is plugged in or out. Node IDs are assigned to nodes according to the tree topology of the bus. Thus, it is not possible to program a node with a specific node ID through software alone [1].

Nodes communicate with each other via two types of packets: asynchronous packets for control data and isochronous packets for real-time streaming data. Since the former adopts the client/server model, an asynchronous packet consists of a destination field (`dst`), a source field (`src`) and a payload field (`data`), as shown in Fig. 1. The latter relies on the publish/subscribe model where senders and receivers are unaware of each other. Thus, node addresses are not recorded in isochronous packets and an address mapping problem does not arise for this type of packets. Therefore, we will focus on only asynchronous packets in the subsequent sections.

B. Network Configuration

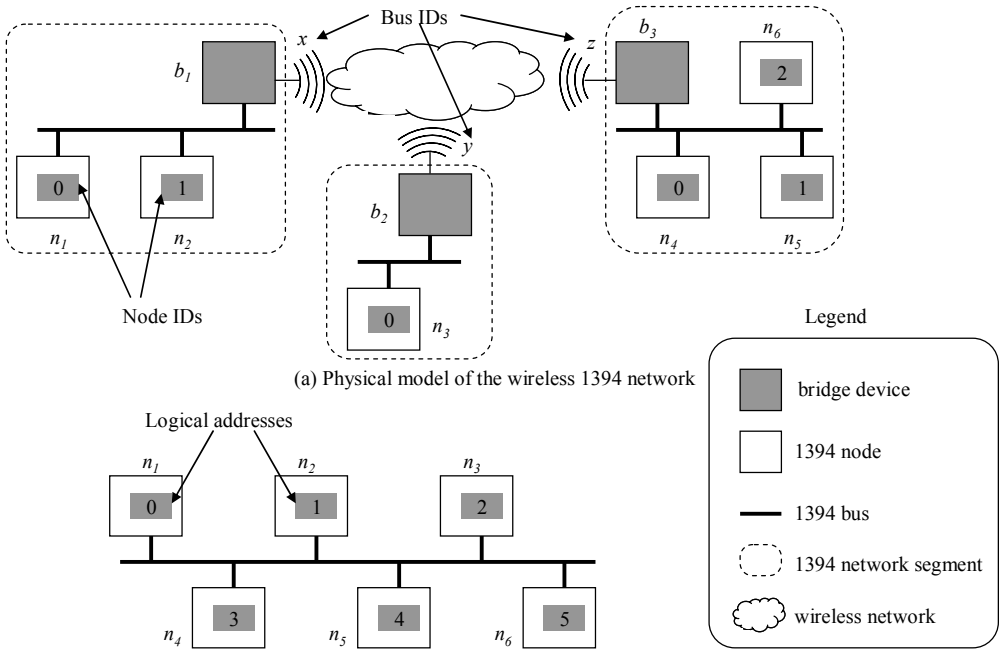
In order to formulate the bridging problem, we have modeled the wireless network of 1394 devices from two different viewpoints. One is the physical view, as shown in Fig. 2 (a), and the other is the logical view, as shown in Fig. 2 (b).

In the physical model, nodes that are connected with the same 1394 bus form a separate 1394 network segment. In each segment, there is a bridge device that connects the 1394 bus and the wireless network. As stated in the previous section, each node has a node ID which uniquely identifies the node in a 1394 network segment. Similarly, each bridge device is assigned a unique address in the wireless network. It can be a specific MAC address depending on the particular protocol used in the wireless network. We call it a bus ID since it can be used to uniquely identify a 1394 bus. Accordingly, in the physical model of the wireless 1394 network, a node is identified by a tuple (bus ID, node ID) and we refer to this as the *physical address* of a node. For example, the physical address of node n_5 in Fig. 2 is $(z,1)$. Note that two nodes in different network segments may have the same node ID, as do nodes n_2 and n_5.

In the logical model, all the nodes are considered to be connected to a single fictitious 1394 bus. The bridge devices and the wireless network are completely hidden in this model. A node in the logical bus is associated with a unique address and we call it the *logical address*. For example, the logical address of n_3 is 2 in Fig. 2.

C. Address Mapping Problem

A bridge mechanism must allow nodes to send and receive asynchronous packets only using their logical addresses. However, in order for a packet to be correctly delivered to a



(a) Physical model of the wireless 1394 network

(b) Logical model of the wireless 1394 network

Fig. 2. Physical and logical models of a wireless 1394 network composed of six nodes and three bridge devices.

destination node across the wireless network, its logical address must be translated into its corresponding physical address. Therefore, a bridge device must have a mapping table and be able to perform a packet address translation according to the table whenever a packet is forwarded. Specifically, the mapping table can be formulated as a logical-to-physical address translation function

$$f : L \to (B \times N) \cup NIL \tag{1}$$

where L, B and N are the set of logical addresses, the set of bus IDs and the set of node IDs, respectively. For example, $f(l) = (b,n)$ indicates that logical address l maps to physical address (b,n) and $f(l) = NIL$ indicates that it does not map to any physical address.

Along with the translation function, we have following two additional design requirements for practical reasons.

- C1: The logical address must be 6 bits long. Otherwise, existing legacy 1394 devices cannot be supported. Maintaining the backward compatibility is very important since a tremendous number of legacy 1394 devices are widely spread.

- C2: A bridge device must be constructed without requiring new custom 1394 hardware chipsets. The reuse of existing 1394 chipsets is preferred as they are low in price and have been extensively tested over a decade.

III. Modeling Existing Bridge Mechanisms

The modeling framework presented in the previous section can serve as an effective tool to formulate and evaluate various bridge mechanisms. Using this framework, in this section we first model the two existing bridge mechanisms and then present our mirroring bridge mechanism in the next section.

The IEEE 1394.1 bridge uses 16 bits for a logical address where the upper 10 bits are used to identify a 1394 bus and the lower 6 bits are used to identify a node in a given 1394 network segment [1]. This implies that $L = B \times N$ and that address translation function f becomes a simple identity function, such that $f((b,n)) = (b,n)$.

While the simplicity of the address translation function leads to an efficient implementation of the IEEE 1394.1 bridge mechanism, the extended logical address space requires nontrivial modifications to the PHY and LINK layer of the original 1394 standard. Therefore, existing 1394 chipsets that conform to the original 1394 standard cannot be used in either 1394 devices or bridge devices. Clearly, the IEEE 1394.1 bridge does not satisfy either requirement C1 or C2.

The transparent bridge uses 6 bits for a logical address just as in the original 1394 standard [7]. Thus, it supports interoperability with legacy 1394 devices and satisfies requirement C1 as well. On the other hand, the transparent bridge additionally mandates that a node ID in a physical address be used as a logical address in each 1394 network segment. This implies that $L = N$ and that the address translation function f again becomes simple, such that $f(n) = (b,n)$.

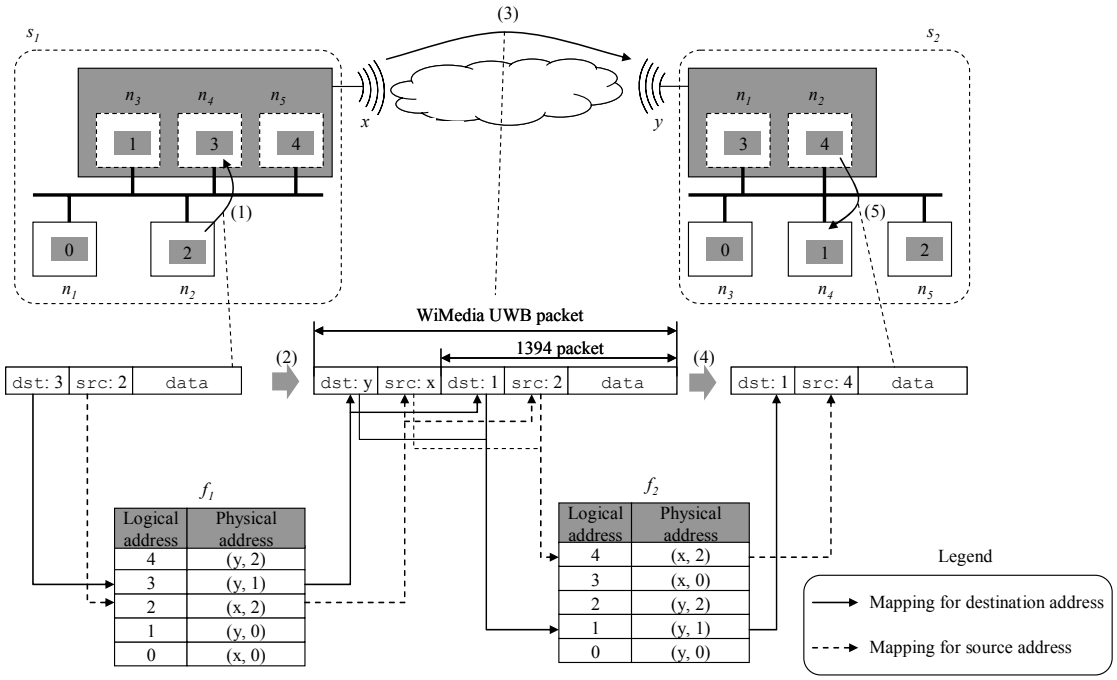


Fig. 3. Packet address translation process of the mirroring bridging mechanism. An asynchronous packet is sent from node n_2 to node n_4. The dotted rectangles inside bridge devices represent proxy nodes for emulating remote nodes. The remote 1394 node that a proxy node emulates is shown on top of the proxy node.

While the address translation function greatly simplifies address mapping, it requires that a bridge device have full control of the node ID allocation process in order that a remote node be able to have a pre-assigned node ID in a local 1394 network segment. As stated in Section II, in the original 1394 standard, node IDs are arbitrarily allocated by a 1394 hardware chipset and cannot be programmed via software. Therefore, the transparent bridge does not satisfy requirement C2.

IV. The Mirroring Bridge Mechanism

In this paper, we formally present our mirroring bridge mechanism using the address translation function and describe how our solution strategies satisfy the two requirements of a practical 1394 bridge. In Section V, we present the detailed architecture design of a bridge device that implements our solution. Before we explain the mirroring bridge formulation, we will first introduce the notion of a proxy node.

A proxy node is a local 1394 node that emulates a remote 1394 node. It is implemented inside a bridge device. Since there is a one-to-one correspondence between a proxy node and its remote node, the node ID of a proxy node can uniquely identify the emulating remote node. Thus, we use the node ID as the logical address of the remote node. This implies that $L = N$ and a logical address is 6 bits long. In Fig. 3, the numbers inside the gray rectangles represent the logical addresses. For example, node n_4 has logical address 3 in segment s_1.

Since the node ID of a proxy node is arbitrarily allocated in each 1394 network segment, the logical address of the same remote 1394 node may vary in different 1394 network segments. In order to resolve this logical address difference, we allow each 1394 network segment to have its own logical address space. This obviously requires a different address mapping function for each bridge device. In Fig. 3, for example, the two tables realize two address mapping functions f_1 and f_2 of the bridge devices in 1394 network segments s_1 and s_2, respectively.

For the bridge device in 1394 network segment s_i, we let f_i and b_i denote its address mapping function and bus ID, respectively. The bridge device constructs a mapping table for function f_i via a three-step process. First, the bridge device waits until all local 1394 nodes, including proxy nodes, are assigned node IDs via the self-ID process of the 1394 protocol. For each node, it then allocates an entry in the mapping table with the node ID being the index of the entry. Second, for each non-proxy local node with node ID l, the bridge device associates bus ID b_i to derive its physical address (b_i, l). In Fig. 3, for example, f_1 maps logical addresses 0 and 2 into physical addresses $(x, 0)$ and $(x, 2)$, respectively. The derived physical address information is broadcast to other bridge devices. Third, for each physical address received from other bridge devices, the bridges device selects an unmapped logical address entry from its mapping table and creates a mapping entry for the physical address.

Using such a mapping table, a bridge device can perform packet address translation. When a packet with destination address l_d and source address l_s is sent from segment s_i to segment s_j, two different types of address translations are

performed.

First, destination address l_d is translated into physical address p_d by the bridge device in s_i using function f_i as below.

$$p_d = f_i(l_d) = (b_j, l_d'). \quad (2)$$

Then, the bridge device in s_j uses the inverse of function f_j to remove its bus ID b_j from p_d as below.

$$l_d' = f_j^{-1}((b_j, l_d')). \quad (3)$$

The l_d' is the logical destination address in segment s_j. The translation from l_d to l_d' can be simplified using a composition of f_i and f_j^{-1} as below.

$$l_d' = (f_i \circ f_j^{-1})(l_d). \quad (4)$$

The source address translation is performed similarly. The bridge device in s_i translates source address l_s into physical address p_s by appending its bus ID b_i using its translation function f_i as below.

$$p_s = f_i(l_s) = (b_i, l_s). \quad (5)$$

Then, the bridge device in s_j uses the inverse of function f_j to translate (b_i, l_s) into logical address l_s' as below.

$$l_s' = f_j^{-1}((b_i, l_s)). \quad (6)$$

The translation from l_s to l_s' can be simplified in a similar manner using a function composition as below.

$$l_s' = (f_i \circ f_j^{-1})(l_s). \quad (7)$$

Fig. 3 depicts an example of the packet address translation process in our mirroring bridge mechanism. In the figure, node n_2 in segment s_1 sends an asynchronous packet to node n_4 in segment s_2. The packet is handled in the order of actions numbered in the figure.

(1) In segment s_1, logical source and destination addresses are 2 and 3, respectively.
(2) The bridge device in s_1 translates logical source address 2 into $f_1(2) = (x, 2)$ and logical destination address 3 into $f_1(3) = (y, 1)$.
(3) The packet is delivered from s_1 to s_2.
(4) The bridge device in s_2 translates the physical source address $(x, 2)$ into $f_2^{-1}(x, 2) = 4$ and the physical destination address $(y, 1)$ into $f_2^{-1}(y, 1) = 1$.
(5) The packet is correctly delivered to destination node n_4.

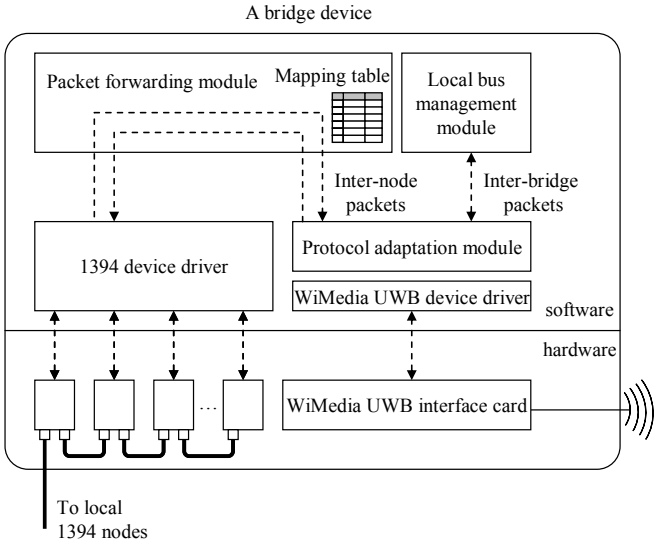


Fig. 4. Architecture of the mirroring bridge. Dotted lines represent packet handling paths.

Observe that the proposed address mapping mechanism satisfies the two requirements of a practical 1394 bridge device. Our mechanism supports interoperability with legacy 1394 devices since it maintains compatibility with the original 1394 protocol by using 6 bits for a logical address. Furthermore, it does not require a custom 1394 chipset for bridge implementation since it works well with the self-ID process of the 1394 protocol.

V. The Architecture Design of a Bridge Device

In this section, we provide the detailed architecture design of a bridge device that implements the proposed mirroring bridge mechanism.

As depicted in Fig. 4, our mirroring bridge hardware consists of multiple 1394 nodes and a WiMedia UWB interface card. Even though a WiMedia UWB interface card is used to exchange packets between bridge devices in this particular design, it can be replaced with any other network interface card since our bridge mechanism does not depend on a specific wireless technology.

Fig. 4 also shows the software architecture that consists of two device drivers and three modules. First, the packet forwarding module transfers packets between a 1394 network segment and the WiMedia UWB network. While transferring a packet, it also translates the destination and source addresses in the packet using the mapping table inside this module according to the process explained in Section IV.

Second, the local bus management module basically manages communication resources and network reconfiguration by exchanging *inter-bridge packets*. The inter-bridge packet is a control packet exchanged between bridge devices. It does not pass through the packet forwarding module. The local bus management modules distributed over all bridge devices jointly

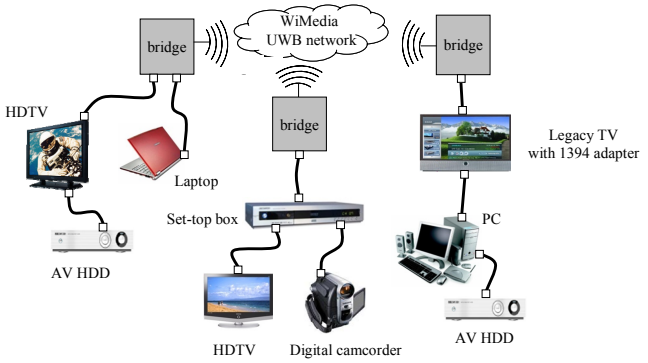


Fig. 5. Experimental network configuration for evaluating the mirroring bridge mechanism.

perform three tasks using inter-bridge packets: (1) they manage isochronous resources such as channel bandwidths; (2) they synchronize the clocks of the 1394 buses in order to prevent buffer overflow and underflow; and (3) they exchange node ID information whenever a 1394 device is plugged in or is unplugged. A local bus management module uses the exchanged information to update the mapping table in the companion packet forwarding module.

Finally, the protocol adaptation module converts a 1394 packet into a WiMedia UWB packet and vice versa. We extract the protocol conversion functionality from the original design and construct it as a separate module in order to increase the portability of our mechanism. Other wireless protocols, such as IEEE 802.11, can be easily supported by implementing an appropriate protocol adaption module.

VI. Experimental Evaluation

We have implemented the proposed bridge mechanism in our bridge device. The bridge device is equipped with a 400-MHz PowerPC processor, 1394 interface cards and a WiMedia UWB interface card. Linux version 2.6.18 was used as the operating system and all the bridging functionalities were implemented as three kernel modules. In order to show the viability and efficiency of our mechanism, we have built an experimental network and conducted a series of experiments in the network. Specifically, we have measured its throughput and latency. Before presenting the experimental results, we will give an overview of the experimental configuration.

Fig. 5 shows the configuration of the experimental network. In the network, we have three 1394 network segments, each of which consists of a bridge device and three legacy 1394 devices including HDTVs, a PC, a set-top box and a digital camcorder. The three bridge devices are connected to a WiMedia UWB network. Although the maximum data rate of the wireless network is defined as 480 Mbps in the specification [9], we set the data rate to 200 Mbps because our prototype UWB hardware chipset could not stably support the maximum data rate.

A. Measuring Throughput

For measuring the throughput, each 1394 device was configured to continuously generate traffic to arbitrary devices across the WiMedia UWB network. We then measured the maximum throughput of a bridge device and the result is shown in Table 1. The resultant throughput was 188.7 Mbps, which is 94.4% of the data rate of the UWB chipset used. The unutilized 5.6% is due to the overhead caused by inter-bridge packets and the additional packet headers for the WiMedia LINK layer [8]. Therefore, we conclude that the throughput loss is negligible when using our mirroring bridge mechanism.

We have also measured throughput in upward and downward directions. The former is the direction from the local 1394 segment to the WiMedia UWB network and the latter vice versa. As shown in the Table 1, the upward throughput was 63.0 Mbps while the downward throughput was 125.7 Mbps. The downward throughput is about twice the upward throughput. This is because the three bridge devices access the WiMedia UWB network in a TDMA manner. For each bridge, one third of the time is used sending packets and two thirds receiving packets.

TABLE I
Measured Maximum Throughput of Bridge Device

Direction	Maximum Throughput (Mbps)
Upward	63.0
Downward	125.7
Both (Upward + Downward)	188.7

B. Measuring Latency

In the 1394 protocol, a packet is considered lost if a response to the packet is not received within 100 ms. Packet loss triggers the retransmission of the packet, which significantly decreases the net throughput. In order to prevent such packet retransmission due to timeout, a bridge device should not add unacceptable latency. We measured the end-to-end round-trip packet delay to show that our mechanism incurred only negligible bridging delay.

We programmed the PC to send asynchronous packets to the laptop as shown in Fig. 5. We then measured the round-trip delay of the packets. The result is shown in Fig. 6. The average, the maximum jitter and the standard deviation of the delay were 2.154 ms, 2.191 ms and 0.750 ms, respectively. The delay is only 2.2% of the 100 ms timeout value. Therefore, we can safely argue that packet losses seldom occur.

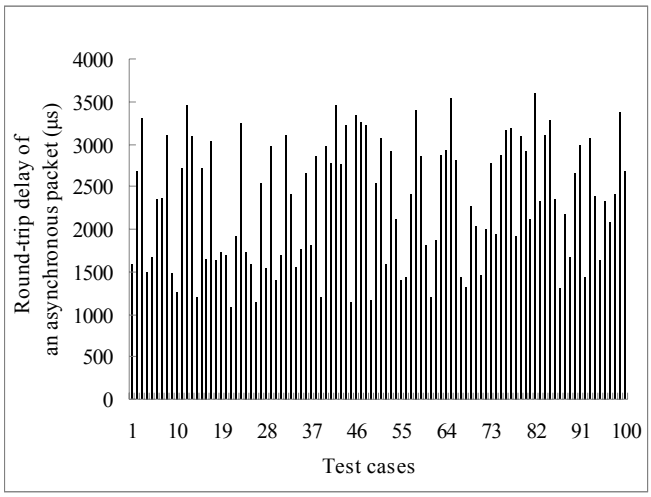


Fig. 6. Round-trip delay of an asynchronous packet.

VII. CONCLUSION

In this paper, we have presented the mirroring bridge mechanism as a means of seamlessly interconnecting legacy 1394 devices over the UWB network. Unlike existing bridge mechanisms, the proposed mechanism supports interoperability with legacy 1394 devices and does not require new chipsets for the implementation of bridge devices. This is made possible through the flexible address mapping function which can be implemented entirely through software.

We have implemented the proposed bridge mechanism in our bridge devices and used them to interconnect legacy 1394 devices via a WiMedia UWB network. By conducting a series of experiments, we have shown that the overhead caused by our mechanism is negligible. Throughput was decreased by only 5.6% and latency was 2.5% of the timeout value.

REFERENCES

[1] IEEE Std 1394-1995, *IEEE Standard for a High Performance Serial Bus*, Dec. 1995.

[2] 1394 Trade Association Wireless Working Group, *Protocol Adaptation Layer (PAL) for IEEE 1394 over IEEE 802.15.3*, Document number 2003010, May. 2004.

[3] IEEE Std 802.15.3-2003, *Part 15.3: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for High Rate Wireless Personal Area Networks (WPANs)*, Sep. 2003.

[4] I. Jang, S. Lee, S. Park and S. Choi, "Design of Protocol Adaptation Layer for IEEE 1394 over IEEE 802.15.3," in *Proceedings of International Conference on Consumer Electronics*, Jan. 2007.

[5] W. Stallings, *Data & Computer Communications*, Prentice Hall, 2000.

[6] IEEE Std 1394.1-2004, *IEEE Standard for High Performance Serial Bus Bridges*, Jul. 2005.

[7] A. Bouffioux, S. Perrot, G. Spalink, D. Mischler and N. Philips, "Transparent Bridges for IEEE 1394 Networks with HiperLAN2 Interconnection between IEEE 1394 Portals," in *Proceedings of International Symposium on Consumer Electronics*, Sep. 2002.

[8] WiMedia Alliance, *WiMedia Logical Link Control Protocol*, Aug. 2007.

[9] ECMA-368, *High Rate Ultra Wideband PHY and MAC Standard*, Dec. 2005.

Jonghun Yoo earned his BS degree in Electrical Engineering from Korea Advanced Institute of Science and Technology, Korea, in 2001. He is currently a PhD candidate at the School of Electrical Engineering and Computer Science of Seoul National University. He is also a member of Real-Time Operating Systems Laboratory at Seoul National University. His current research interests include software engineering for embedded systems, software product lines, component based software engineering and real-time operating systems.

Jiyong Park earned his BS degree in Electrical Engineering from Seoul National University, Korea, in 2002. He is currently a PhD candidate at the School of Electrical Engineering and Computer Science of Seoul National University. He is also a member of Real-Time Operating Systems Laboratory at Seoul National University. His current research interests include customizable software development, aspect-oriented programming, multiprocessor real-time scheduling and embedded real-time operating systems.

Seongsoo Hong (M'96) earned his BS and MS degrees in Computer Engineering from Seoul National University, Korea, in 1986 and 1988, respectively. He received his PhD degree in Computer Science from the University of Maryland, College Park, in 1994. He is currently a professor in the School of Electrical Engineering and Computer Science at Seoul National University. His current research interests include embedded and real-time systems design, real-time operating systems, embedded middleware, and software tools and environments for embedded real-time systems. He served as a general co-chair of IEEE RTCSA 2006 and CASES 2006 and as a program committee co-chair of IEEE RTAS 2005, RTCSA 2003, IEEE ISORC 2002 and ACM LCTES 2001. He has served on numerous program committees, including IEEE RTSS and ACM OOPSLA. He is currently a member of the IEEE and ACM.