>Paper ID <

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

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

***Abstract*—Bridging IEEE 1394 buses is becoming important since it enables wireless connectivity. Unfortunately, existing bridging schemes such as the IEEE 1394.1 and transparent bridges 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 mirroring bridging scheme 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 in software. We have successfully implemented the mirroring bridge scheme and conducted extensive experiments. The average throughput of the mirroring bridge was 188.7 Mbps that is 99.3% of the maximum throughput of the UWB network.**

I. Introduction

The IEEE 1394 bus is widely used for interconnecting home AV devices since it can support high bandwidth, plug and play, and QoS guarantee. Recently, there have been growing demands for wireless connectivity among IEEE 1394 devices. One of viable ways to implement the wireless 1394 is to connect devices in proximity via a separate IEEE 1394 bus and then connect multiple 1394 buses via a wireless network. In such a scheme, bridges are required to link IEEE 1394 buses with a wireless network.

While the IEEE 1394.1 bridging scheme [1] and the transparent bridging scheme [2] can be used for this purpose, they possess practical limitations. The former loses interoperability with legacy 1394 devices since it requires the PHY and LINK layers should be changed from the original IEEE 1394 standard. The latter supports the desired interoperability but requires a new hardware chipset for bridge implementation.

In this paper, we propose a new bridging scheme to address these problems. We name it a *mirroring bridge*. Our mirroring bridge scheme emulates a remote device inside a bridge so that the remote device can be transparently accessed by a local device. As a result, a 1394 device can communicate with others regardless of their physical locations. In order to perform the remote device emulation, the mirroring bridge maintains a mapping table and translates packet addresses using this table. Since this mechanism can be implemented entirely in software, the mirroring bridge can be constructed only with existing 1394 chipsets.

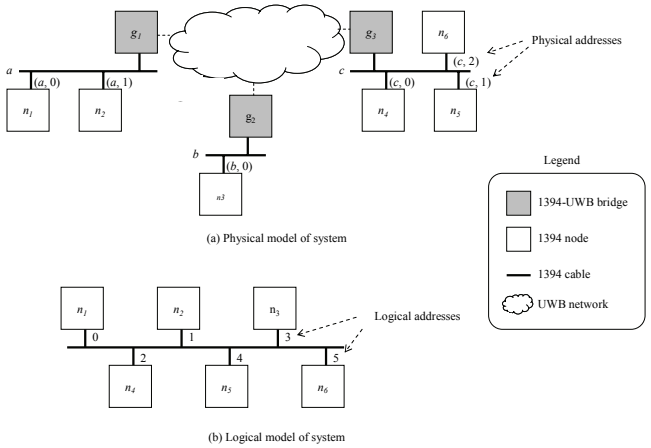

Fig. 1. Physical and logical models of the target system.

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

II. Address Mapping Problem of IEEE 1394 Bridges

In order to formulate and evaluate IEEE 1394 bridging schemes, we first model the target system from two different viewpoints. One is the physical model as shown in Fig. 1 (a) and the other is the logical model as shown in Fig.1 (b). In the physical model, the target system consists of multiple 1394 buses interconnected with a wireless network (UWB) via bridges. Each node is associated with a physical address which is represented by a tuple (bus ID, node ID). A bus ID uniquely identifies a physical 1394 bus in the system and a node ID uniquely identifies a node in a given 1394 bus.

In the logical model, the system is modeled as a single 1394 bus. The bridges and wireless network are completely hidden and all remote nodes are seen as logically connected to the local bus. A node in the logical bus is associated with a unique logical address. If compatibility with the original 1394 standard is needed, a logical address should be 6-bit long.

A bridging scheme should provide 1394 devices with the logical view of the system and bridges with the physical view. To do so, a bridging scheme should define a mapping with which a logical address is translated into a corresponding physical address. Such a mapping is simply defined as

$$f : L \rightarrow (B \times P) \cup \{NIL\},$$

where *L, B,* and *P* are respectively the set of logical addresses, the set of bus IDs, and the set of node IDs. For example, $f(m) = (b, p)$ indicates that a logical address m is mapped to a physical address (b, p).

The IEEE 1394.1 bridging scheme extends the logical address to 16 bits where the upper 10 bits are used to identify a physical bus and the lower 6 bits are used to identify a node in a given physical bus [1]. Obviously, such address space extension requires nontrivial modifications to the PHY and LINK layers of the original 1394 standard. This, unfortunately, leads to the loss of interoperability with legacy 1394 devices.

The transparent bridging scheme supports legacy 1394 devices since it uses a 6-bit node ID as a logical address. In order to simplify the logical-to-physical address translation, it

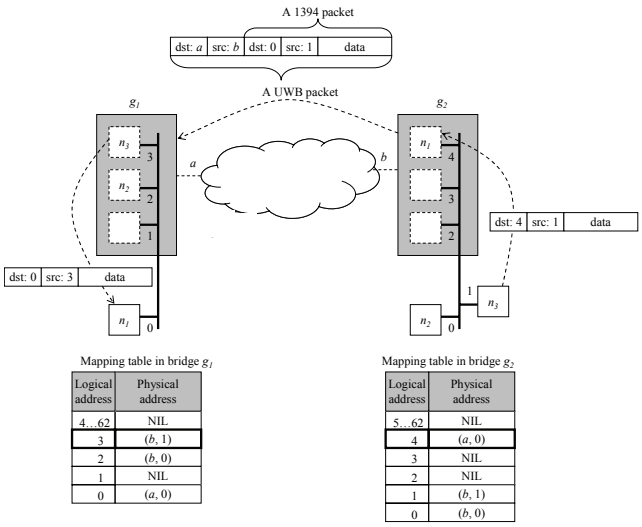

Fig. 2. An example of the address translation process and the structure of mapping table.

mandates that a device should have the same node ID in all local buses. Unfortunately, this address assignment scheme requires a new hardware chipset for bridge implementation since node IDs are determined by the bus reset mechanism in the original IEEE 1394 bus.

III. Solution Approach and Experimental Evaluation

Our mirroring bridging scheme uses a 6-bit node ID as a logical address. Unlike the transparent bridging scheme, each mirroring bridge is allowed to map a remote 1394 device into a different local node ID independently of other bridges (or buses) simply using the bus reset mechanism of the original IEEE 1394 standard. This requires that each mirroring bridge should implement its own address mapping using a table and translate packet addresses using this table. Fig. 2 shows an example of the mapping table. Each local and emulated remote node in a local 1394 bus has an entry in the table and each entry contains a logical and physical address pair.

Packet address translations are performed as follows. Suppose that a node in bus b_1 transmits a packet to a remote node in bus b_2 and that f_1 and f_2 are the mapping functions of local and remote bridges, respectively. First, the local bridge in bus b_1 intercepts a packet and translates logical address m in the packet into physical address $f_1(m)$ and forwards it to

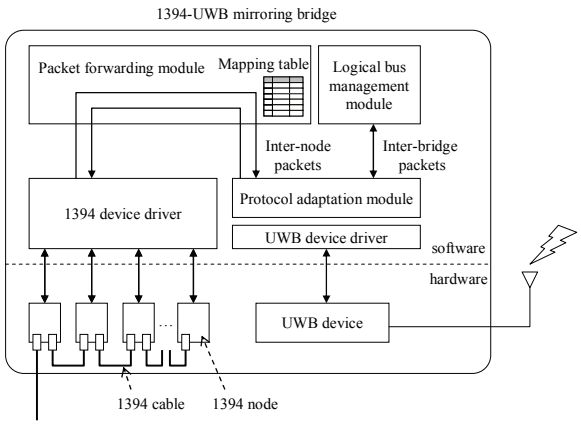

Fig. 3. The hardware and software architecture of the mirroring bridge.

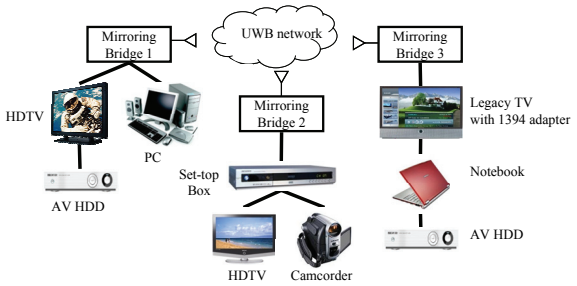

Fig. 4. The experimental environment.

the remote bridge in bus b_2 via the UWB network. Then the remote bridge translates the physical address $f_1(m)$ in the packet into logical address $f_2^{-1}(f_1(m))$ and finally sends it to the destination node. Fig. 2 shows an example of address translation process where a packet is transmitted from the node n_3 to the node n_1.

As depicted in Fig. 3, the mirroring bridge hardware consists of multiple 1394 nodes and a UWB device. Fig. 3 also shows the software architecture that consists of two device drivers and three modules. First, the packet forwarding module transfers packets between a 1394 bus and the UWB network and translates addresses in the packets. Second, the protocol adaptation module performs a protocol conversion between the 1394 and UWB MAC. Finally, the logical bus management module performs global isochronous resource management, global clock synchronization, and bus reset handling.

We have successfully constructed a wireless 1394 bus with three newly implemented mirroring bridges, UWB network, and nine 1394 devices connected via three 1394 buses, as shown in Fig. 4. We have implemented the mirroring bridge using Linux version 2.6.18 and additional three loadable kernel modules which realize the three bridge functions. We have also conducted a series of experiments to evaluate its performance and run-time overhead. The average throughput of our bridge was 188.7 Mbps that is 99.3% of the maximum throughput of the UWB network. The average round-trip delay of a packet was 2.154 ms.

IV. Conclusion

In this paper, we presented the mirroring bridging scheme that can be used to construct a wireless 1394. Unlike existing bridging schemes, it supports interoperability with legacy 1394 devices via a software-implemented address translation mechanism. Although the mirroring bridge was implemented in software, its runtime overhead is extremely small. 99.3% of the UWB bandwidth is utilized.

References

[1] IEEE Computer Society, *1394.1 IEEE Standard for High Performance Serial Bus Bridge*. IEEE, 2005.

[2] Alain Bouffioux, Sebastien Perrot, Gerd Spalink, Denis Mischler, and Norbert Philips, "Transparent bridges for IEEE 1394 networks with Hiperlan2 interconnection between IEEE 1394 portals," in *Proceedings of International Symposium on Consumer Electronics*, Sep. 2002.