Wearable Personal Network Based on Fabric Serial Bus Using Electrically Conductive Yarn

Hyung Sun Lee, Choong Bum Park, Kyoung Ju Noh, John Sunwoo, Hoon Choi, and Il-Yeon Cho

E-textile technology has earned a great deal of interest in many fields; however, existing wearable network protocols are not optimized for use with conductive yarn. In this paper, some of the basic properties of conductive textiles and requirements on wearable personal area networks (PANs) are reviewed. Then, we present a wearable personal network (WPN), which is a four-layered wearable PAN using bus topology. We have designed the WPN to be a lightweight protocol to work with a variety of microcontrollers. The profile layer is provided to make the application development process easy. The data link layer exchanges frames in a master-slave manner in either the reliable or best-effort mode. The lower part of the data link layer and the physical layer of WPN are made of a fabric serial-bus interface which is capable of measuring bus signal properties and adapting to medium variation. After a formal verification of operation and performances of WPN, we implemented WPN communication modules (WCMs) on small flexible printed circuit boards. In order to demonstrate the behavior of our WPN on a textile, we designed a WPN tutorial shirt prototype using implemented WCMs and conductive yarn.

Keywords: E-textile, wearable computer, personal area network, serial bus topology.

Manuscript received Mar. 12, 2010; revised Aug. 6, 2010; accepted Aug. 9, 2010.

This work was supported by the IT R&D program of MKE/KEIT (2008-F-048, Wearable Personal Companion for u-Computing Collaboration).

Hyung Sun Lee (phone: +82 42 860 6931, email: hslee77@etri.re.kr), Kyoung Ju Noh (email: kjnoh@etri.re.kr), John Sunwoo (email: bistdude@etri.re.kr), and Il-Yeon Cho (email: iycho@etri.re.kr) are with the Software Research Laboratory, ETRI, Daejeon, Rep. of Korea.

Choong Bum Park (email: here4you@cnu.ac.kr) and Hoon Choi (email: hc@cnu.ac.kr) are with the Mobile Distributed Computing Laboratory, Chungnam National University, Daejeon, Rep. of Korea.

doi:10.4218/etrij.10.1510.0084

I. Introduction

Mobile and ubiquitous computing devices have drastically changed our lifestyles and brought out increasing demands for more powerful computing services. However, these electronic devices are weighty and not secure when placed in pockets. Thus, people commonly fear losing these expensive electronic devices. Electronic textiles, or e-textiles, can provide a solution to these problems. Smart clothing made of e-textile technologies provides sufficient space or textile layers for electronic components, and users can easily keep track of their whereabouts. E-textiles have gained much interest in many fields of research such as medicine, military, and mobile entertainment [1]-[3].

Electronic components in e-textiles are normally distributed around/on the garment to gather local personal information and increase comfort. For this reason, a network is needed to connect embedded nodes to exchange control and data information. There are two known types of personal area networks (PANs): wired and wireless. Both types of network have been widely studied. However, wireless PANs such as Bluetooth, Zigbee, and wireless local area network suffer from interference, fading, and low data rates [4], [5].

A number of wired e-textile networks have been proposed. Post and Orth [6] introduced the use of conductive yarns as a simple data bus. Gorlick [7] proposed a bus-type wearable PAN for power and data transfer using a controller area network as a physical layer (PHY). However, mechanical and electrical properties of conductive yarns were not considered. Wade and Asada [8] proposed a similar PAN using DC power line communication (PLC) PHY. They considered the impedance mismatching problem occurring on e-textile

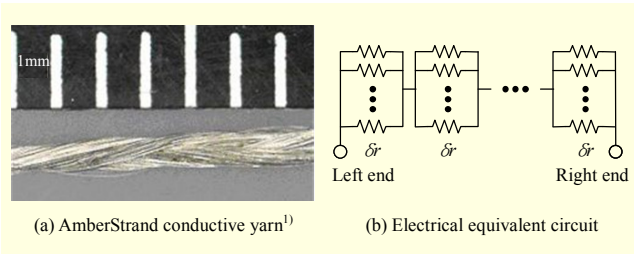
platforms. However, the wearable DC PLC only applies to conductive fabric with high conductivity. The two studies on PAN also suffer from low data rates. Nakad and others [9] and Yoo and others [10] proposed fault-tolerant networks with mesh structures in which conductive yarns are woven into textiles at appropriate locations. However, mesh networks require redundant switches to handle network faults and maintain their robustness. Furthermore, when mesh-type networks are used, the garment design process is tightly coupled with the application design process. This can be problematic for fashion designers with little knowledge on electronics.

Myung and others [11] identified various requirements of the e-textile network and performed an analysis on commercial field-bus protocols. They proposed that as a part of clothing, e-textile network protocols should operate on flexible and elongable conductive textiles and withstand wear and tear that regular garments experience. They also held that e-textile technology should not impose restrictions on materials and fabrication methods of e-textile garments. Finally, to provide efficient networking services for e-textiles, they proposed that network topology should minimize the number of physical connections (often the cause of line failure), support various types of data traffic, and be fault-tolerant against gradual line damage. They concluded that no commercial field-bus protocols meet the requirements of the e-textile network.

In order to maintain their mechanical flexibility and elongability, most conductive yarns are manufactured by twisting a number of filaments. Hence, the overall conductivity of a conductive yarn is determined by the conductivity of its filaments. Figure 1 shows the electrical equivalent circuit of a piece of conductive yarn.

As the body-worn e-textiles are scratched and rubbed in real life, conductive coatings are abraded over time, gradually losing conductivity. Washing test results in research show that resistance of conductive yarns and fabrics linearly increase as they are washed [12]-[16]. The fabric serial-bus (FSB) interface proposed by Lee and others [17], which can detect the variation in line conductivity, allows the upper layers to use that information to control network parameters such as baud rates. However, details of its data link layer and application results on actual e-textiles were not introduced.

In this paper, a set of wearable PAN protocols will be proposed. The data-link-layer protocol is capable of discovering and managing up to 64 embedded nodes and provides reliable and best-effort options to allow the trade-off between transmission success rate and time. It can also handle notification of signal line variation by adjusting network parameters accordingly. The profile layer protocol provides a set of application programming interfaces (APIs) to allow easier development of e-textile applications. Furthermore, we designed our network protocol stack to be light enough for microcontrollers with limited memory resources. The resulting code size of our software protocol stack is less than 42 kB. To verify the operation of our network protocol, we implemented FSB hardware on a flexible printed circuit board (PCB) and designed an e-textile shirt with five embedded nodes.



(a) AmberStrand conductive yarn[1)] (b) Electrical equivalent circuit

Fig. 1. (a) Photo of conductive yarn and (b) its electrical model.

1) http://www.amberstrand.com/

II. Wearable Personal Network Protocol

1. Wearable Personal Network Protocol Architecture

The wearable personal network (WPN) has a wired network protocol for devices embedded in a garment. The embedded devices may have different functions: biological and environmental sensing, user input/output, storage for large multimedia data, and wireless network interfacing [18]. The WPN is designed to provide a set of APIs for easy development of various embedded nodes. WPN can also handle various types of network traffic. This subsection describes the overall architecture of the WPN protocol.

The WPN is based on a dual-lined, serial bus topology as shown in Fig. 2. It is advantageous in scalability because it is easy to append a new route or add new devices to the existing network.

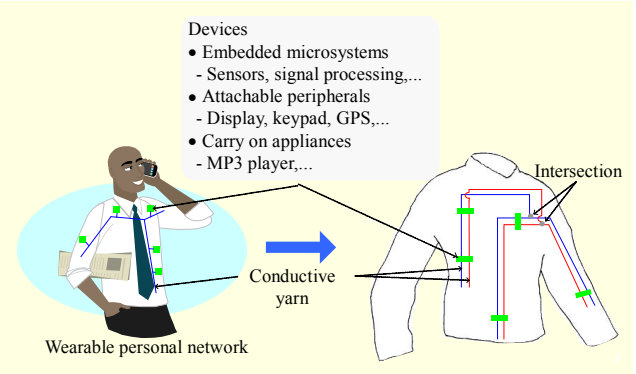


Fig. 2. WPN based on FSB.

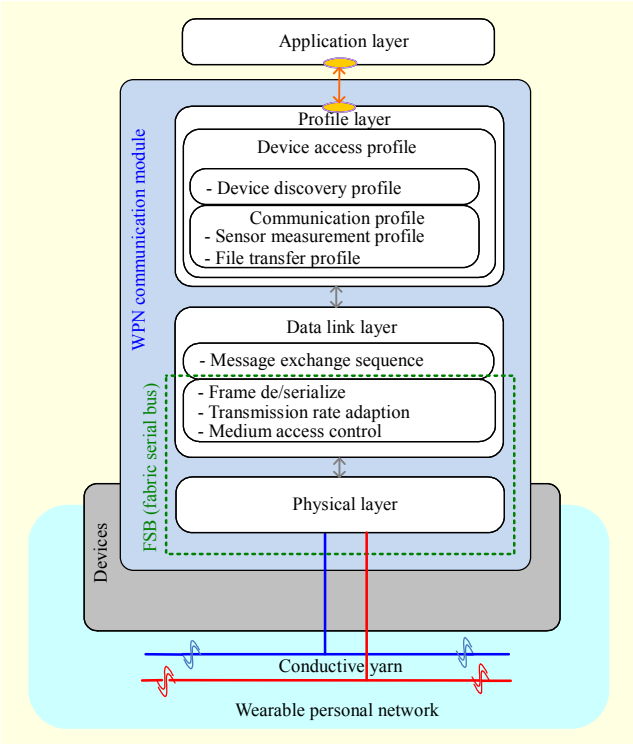


Fig. 3. WPN protocol architecture.

Embedded devices in the WPN communicate with other devices using the WPN communication module (WCM). The protocol stack of WCM consists of a profile layer, a data link layer, and a PHY (see Fig 3). FSB forms a part of the data link layer, including medium access control (MAC) and a PHY. FSB is a serial bus interface that uses conductive yarns as a communication medium [17].

Both the profile layer and the data link layer of WCM provide a set of data communication and network management APIs for applications on each embedded devices. The profile layer is designed to provide convenience and interoperability for device applications. The data link layer performs packet encapsulation/decapsulation and controls packet exchange sequences using a master-slave mechanism.

2. Profile Layer

In a WPN environment, various devices can be implemented as host or peripheral nodes. Thus, different application programs need to be ported on each device. In order to successfully communicate among these embedded devices, their application programs need to use the correct communication API at the right moment.

In order to improve convenience in application design and interoperability among applications, a guideline for data link protocol usage must be provided. In this study, we adopted a method to use profiles in well-formed templates. Profiles are code templates for application developers as shown in Table 1. They can produce application programs by adding desired functions to the code templates. The profile method has already been applied to Bluetooth and Zigbee.

Table 1 Example of a profile.

Device discovery profile		
Host		
Function	Short DiscoveryNode (unsigned char addr, unsigned char group)	
Description	Periodically broadcasts the Discovery Message.	
Argument	Addr	Logical address to be allocated to a node
	Group	A node processes Discovery Message if "group" matches the last 8 bits of the node's serial number
Peripheral		
Function	Unsigned short DSEventHandler (void)	
Description	Recognizes the Discovery Messages	
Function	Unsigned short DSEvent (char* framebuffer)	
Description	Analyzes and processes the Discovery Message	
Argument	Framebuffer	Address & group information of Discovery Message

```
struct DeviceDescriptor {
        unsigned int deviceType;
        unsigned long serialNum;
        char deviceName[20];
        char description[256];
        char date[10];     // manufacture date
        char vendor[30];
}
```

Fig. 4. Structure of device descriptor.

Two device access profiles were defined for WPN: the device discovery profile defines a discovery procedure and the communication profile defines a data communication procedure between host and peripheral nodes. The communication profile is further classified into the sensor measurement profile for collecting sensory data from sensor nodes and the file transfer profile for providing file transfer protocol service. Two different versions of each profile were developed for host and peripheral nodes.

The device discovery profile describes a sequence of device recognition on the e-textile. It is executed when a device is newly attached to the e-textile or when the e-textile is powered. As shown in Fig. 4, each device has a device descriptor to describe its hardware specification. The host node obtains information about the peripheral node by reading the node's device descriptor.

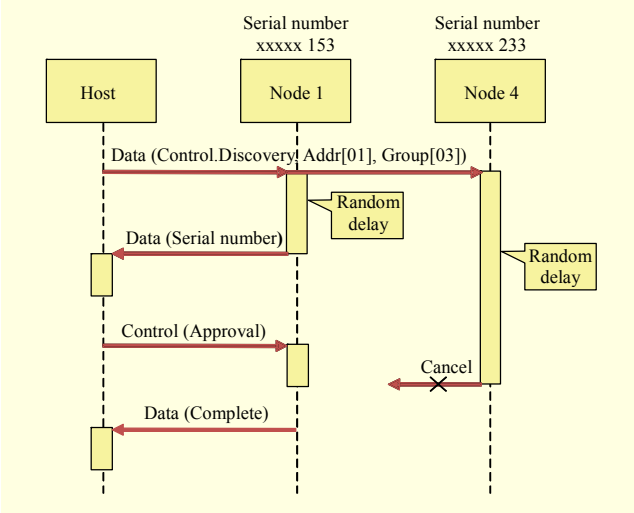


Fig. 5. Procedure of device discovery profile.

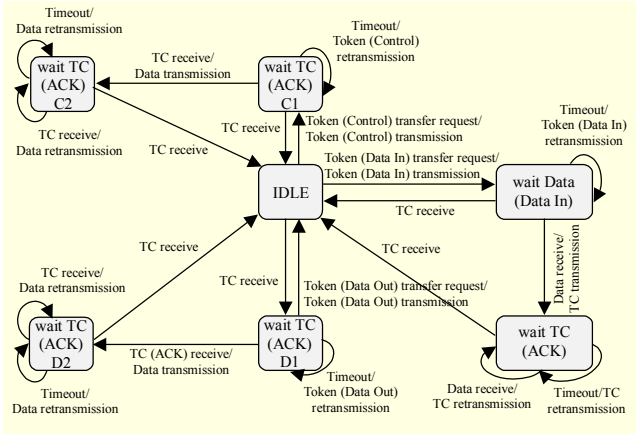


Fig. 6. State diagram of host node.

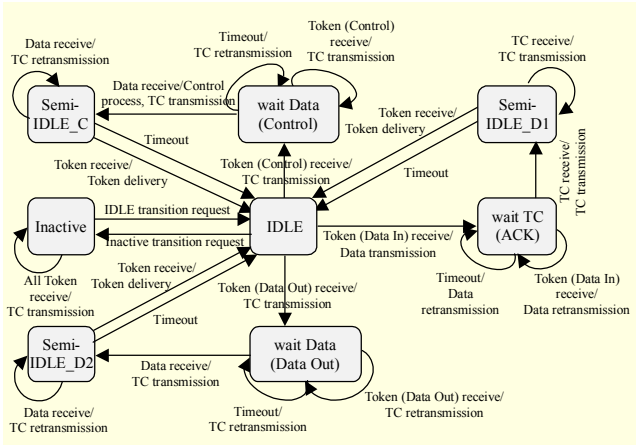


Fig. 7. State diagram of peripheral node.

According to this profile, the host node broadcasts the discovery message that contains the address value and the group value to all the nodes in the bus. The address value is an identifier to be allocated to a node, and the group value is a four bit number that selects the group of devices to respond to the broadcast. Only the devices with matching group bits, the last four bits of the serial number, respond after a random backoff, and only the device that responds first acquires the broadcasted address.

Figure 5 shows an example of the device discovery sequence. The host node broadcasts the discovery message which contains an address value of 1 and a group value of 3. Both node 1 and node 4 have the value 3 as the last four bits of their serial numbers. Hence, both nodes generate a response message that contains their own serial numbers. In order to reduce the chance of message collision, each node broadcasts the response message after waiting through a random delay. If waiting nodes hear a response message from another node during their own random delay period, they cancel the transmission of their response messages. Only the first node that responds proceeds with the discovery procedure, and the other nodes should look for next discovery message.

After the host node receives the response message from a node, they exchange the approval message followed by the complete message to complete the discovery procedure.

3. Data Link Layer

A. Protocol Design

The MAC of the data link layer was designed to work with a bus topology environment with no carrier sense multiple access function. Therefore, the data link layer uses a single host node polling of other peripheral nodes. The data link layer provides both reliable and best-effort services. A transaction to exchange a frame by the reliable data link protocol starts with token frame (TF) transmission by the host and ends with transmission confirmation frame (TCF) reception by the host for the case of reliable communication. If an error or loss of a frame occurs, the transmission of a frame is repeated up to three times, with 10 ms timeout interval to guarantee quality of service (QoS). The timeout values can be set by QoS configuration. The data link protocol with a best-effort option does not use TCF or an error checking mechanism.

State diagrams of the reliable data-link-layer protocols for the host and the peripheral nodes are shown in Figs. 6 and 7, respectively. A peripheral node normally waits to be polled by the host node in IDLE state. If a peripheral node receives a token (Data In) frame, it transmits a data frame (DF) waiting in its transmit buffer and waits for the acknowledgement in the "wait TC (ACK)" state as shown in Fig. 7. When the peripheral node receives a TCF, it also replies with a TCF to the host node and waits in the "Semi-IDLE D1" state. This state is to ensure that the host node has received the reply

correctly. If the host node does not retransmit the TCF within a certain time interval, the peripheral node knows that the host node has ended this transaction successfully and returns to the IDLE state.

B. Frame Structure

Each field in the message frame shown in Fig. 8 was defined to meet requirements of wearable PAN. Applications of WPN exchange sensory and command data as well as multimedia data, such as video and audio streams. Therefore, the data field was designed to handle from zero to 1,024 bytes of payload. The start flag and the end flag fields were designed to have specific bit patterns. This ensures that the baud of any incoming frame can be determined by analyzing the start flag. This also means frame consistency can be checked by analyzing the end flag.

Message frames of the data link protocol are classified into TF, DF, control frame, and TCF. The type of a frame is specified at the header of each frame. TF and TCFs have a payload data of zero bytes.

C. Application Programming Interface

APIs of the data link layer are listed in Table 2.

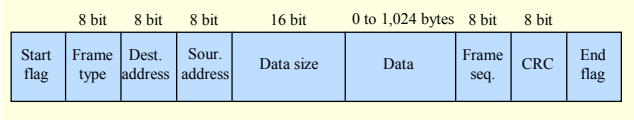


Fig. 8. Frame structure.

Table 2. Data link protocol interface.

API name	Function	Host	Peri.
DLProtocolOpen	Initialization and activation of link layer protocol	○	○
DLProtocolClose	Inactivation of link layer protocol	○	○
DLDataRead	Read data from the node	○	×
DLDataWrite	Write (send) data to the node	○	×
DLFrameSend	Frame transmission (sending)	○	○
DLRecvedFrame	Frame reception	○	○
DLEventHandler	Processing a (read/write) event according to the host request	×	○
DLChangeMachineState	Change of machine condition	×	○
DLChangeLogicalAddr	Change of logical address	×	○
DiscoveryNode	Finds the node which a logical address is not allocated to and then allocates an address	○	×
DSEventHandler	Processing discovery event	×	○

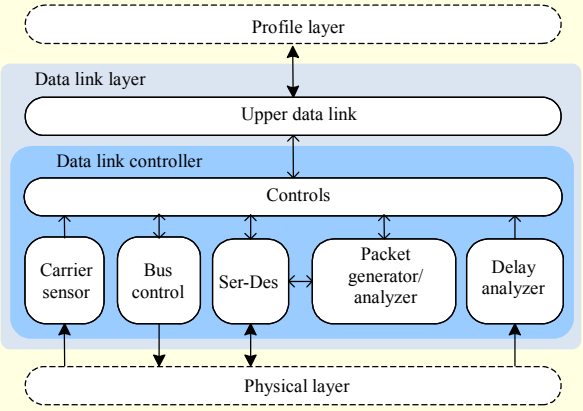


Fig. 9. FSB controller in data link layer.

D. FSB Controller

The lower part of the data link layer is the FSB controller that provides various data controls between the upper data-link layer and FSB transceiver as shown in Fig. 9. It provides carrier sensing, bus control, packet generation and analysis, de/serialization, and signal delay analysis functionalities.

Each embedded device in WPN has different network bandwidth requirements depending on its function and condition of communication medium. The serializer-deserializer (Ser-Des) of the FSB controller allows for the selection of different baud rates for each frame. Its ability to receive an incoming frame with an unknown baud is called auto-packet-baud-detection, and it is achieved by real-time measurement of bit patterns of the start flag in each incoming frame header [17]. The packet generator/analyzer block interfaces with the Ser-Des block to handle the data incoming frame whose structure is defined in Fig. 8.

The carrier sensor and the bus control blocks initiate and terminate data activities on the bus using the PHY. The delay analyzer block measures signal delays of received signals using preprocessed signals from the PHY. Blocks in the upper data-link layer can read the delay value of each incoming frame from registers and use them to adjust network parameters such as the baud rate.

4. Physical Layer

The PHY of WPN uses the FSB transceiver [17]. The FSB transceiver uses a low voltage differential signaling scheme, which allows for low-power high-speed data transmission. In addition, the FSB transceiver has a differential amplifier and a dual-threshold high-speed comparator to process bus signals transmitted through two conductive yarns. The output of each comparator changes at a different voltage threshold, which allows the FSB controller in the data link layer to measure

the rising and falling times of bus signals.

III. WPN Prototype

To evaluate the operation of the developed WPN protocols, we implemented the WCM hardware on small flexible PCB and designed a WPN tutorial shirt with one host and four peripheral nodes.

1. WPN Communication Module

Figure 10 shows a picture of WCM hardware prototype which includes the lower part of the data link layer and the PHY of WPN. The lower part of the data link layer was implemented on a field programmable gate array (FPGA) running at 50 MHz. Upper layers of the WPN and device applications are implemented in a microcontroller. The WCM hardware allows the register access using a serial peripheral interface (SPI) which is common in most microcontrollers and embedded processors.

The prototype module has two conductive snap buttons to be attached on the shirt where a dual-wire bus made of the conductive yarn is sewn. It also has an SPI port to enable interfacing between the microcontroller unit (MCU) and the FPGA. Finally, the prototype module has a removable joint test action group connector for programming the FPGA.

2. Software Protocol Stack

Two versions of the data-link-layer protocol were implemented using C language. A desktop version was developed for PCs with an x86 processor using user datagram protocol broadcast over Ethernet to evaluate frame exchange logics of the protocol. An embedded version was implemented for MSP430 series microcontrollers. A hardware-dependent portion of the data link protocol was implemented with a hardware abstraction layer to help ease development of a protocol stack for other microprocessors. A list of the data-link-layer protocol APIs is shown in Table 2.

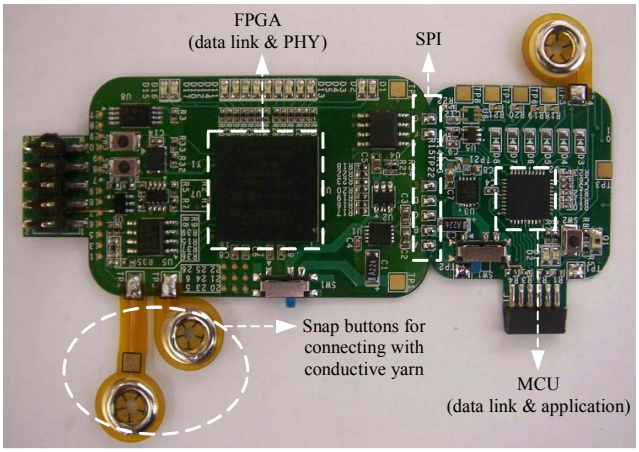


Fig. 10. WCM prototype and MCU.

Table 3. Number of test cases and test results.

Test subject	Host		Peripheral	
	Test	Pass	Test	Pass
IDLE status	11	11	11	11
Inactive status	–	–	11	11
Data read	22	22	22	22
Control data write	22	22	22	22
Data write	22	22	22	22

Table 4. Performance evaluation results.

Data read						
Experiment environment	Desktop				MCU	
Commun. type	BE	R	BE	R	BE	R
Target	Node 1	Node 1	Node 2	Node 2	Node 1	Node 1
No. of success	4,999	5,000	4,994	5,000	4,769	4,972
Success rate (%)	99.98	100	99.88	100	95.38	99.44
Data write						
Experiment environment	Desktop				MCU	
Commun. type	BE	R	BE	R	BE	R
Target	Node 1	Node 1	Node 2	Node 2	Node 1	Node 1
No. of success	5,000	5,000	5,000	5,000	4,840	4,979
Success rate (%)	100	100	100	100	96.8	99.58

Note. BE: best-effort option
R: reliable option

To evaluate the operation of the data link layer, we performed a formal protocol test. The test suite was derived from state diagrams of the protocol using ISO tree and tabular-combined notation forms and tested with the tools from Onnet Technologies. The number of test cases and test results are shown in Table 3. The data-link-layer protocol passed all 165 cases.

Communication performances between two desktop PCs and two WPN nodes with WCMs were tested with the implemented software protocol stack. The test consisted of a total of 5,000 data exchanges between a host node and a peripheral node. Each data exchange occurred at a 300 ms interval, and both the reliable and best-effort options were tested. Experiment results in Table 4 indicate that all the data were successfully delivered in the desktop environment using the reliable option. However, the mean success rate of the best-

Table 5. Transaction time (ms).

	Data read			Data write		
	Avg.	Min.	Max.	Avg.	Min.	Max.
BE	10.9	4.8	14.4	<1	–	–
R	20.4	14.4	30.4	20.0	14.4	30.4

Note. BE: best-effort option
R: reliable option

effort data-read operation was reduced to 99.93%. Under an MCU environment, the mean success rates of the best-effort and the reliable operations were 96.09% and 99.51%, respectively.

Table 5 shows the average transaction time results of 5,000 data read and data write transactions between a host node and a peripheral node. For data read and write transactions, 32 bytes of data payload were used. In the case of reliable transaction, we measured the time at the host node from transmitting a token until receiving data/TCF. Since best-effort transactions do not exchange TCFs, their processing times and the transmission delays on the link are excluded in the best-effort transaction times. Hence, its results are shorter than that of reliable ones. Also, for the best-effort case, the transaction time of data write is much shorter than that of data read because the data write operation using the best-effort option is accomplished by only sending one frame from the host node.

3. Verification of WPN Using the Tutorial Shirt

We implemented the WPN tutorial shirt with five embedded nodes as shown in Fig. 11 to demonstrate operation of the proposed WPN protocol and its implementation. Each embedded node consists of a WCM hardware board and a microcontroller board. Peripheral nodes are equipped with an ambient temperature sensor and a light sensor, a textile touch user interface (UI) that can recognize five touch gestures, a character liquid crystal display (LCD) for information display, and an MP3 codec. The host node plays the role of the master. Each embedded device is attached to a serial bus with two conductive yarns sewn on the WPN tutorial shirt as marked in Fig. 11.

The microcontroller on each peripheral node is loaded with an application that manages its sensor and actuator. The host node is loaded with a bridge application which polls and routes packets among peripheral nodes. As shown in Fig. 11, the operation of the WPN tutorial shirt can be verified by the information displayed on the LCD. It displays the ambient temperature and illumination values, the number of WPN

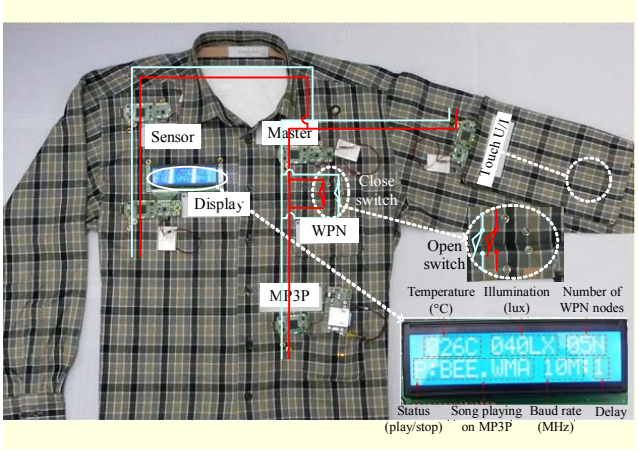


Fig. 11. WPN tutorial shirt.

nodes, the title of the song playing in the MP3 codec, and current network information, such as baud rates and measured signal delay in periods of 20 ns.

Additionally, to demonstrate the FSB's ability to sense the status of the communication medium and assign appropriate baud rates, we created a worn-out path between the MP3 node and the host node, and we appended a parallel route with snap buttons. Opening of these snap buttons simulates abrasion of the conductive yarn bus, and closing of them simulates repairing damaged routes.

As peripheral nodes on the WPN tutorial shirt are switched on, the host node discovers them and sends the information to the display node to update the node count in LCD display. The temperature and light node periodically senses the current temperature and illumination and sends them to the host node, which forwards the information to the display node for an update. Tabbing the textile touch UI with a finger indicates a play/pause gesture. Upon recognition of a control gesture, the touch UI node sends control information to the host node, which forwards the commands to the MP3 node.

The behavior of WPN to a line failure can be tested with two snap buttons at the upper left chest. When we unsnap these buttons, the connection between the MP3 node and the host node is only made of the worn-out conductive yarn with low conductivity. The FSB hardware detects a change of the signal delay due to the low conductivity of the communication wire as soon as the first frame is sent through the worn-out conductive yarn. This information is stored in registers in the data link layer, and an interrupt is generated. The data link layer at the host node uses this information to adjust the network parameter, and updated baud-rate information is sent to the display node. When buttons are unsnapped, we can observe that the signal delay value displayed in the LCD increases from one to ten, and the baud rate decreases from 10 Mbps to 1 Mbps.

IV. Conclusion

E-textile technology has received much expectation as the next generation wearable computing technology. Some of simple items such as iPod jackets have already reached the market. However, in order to design more powerful e-textiles with intelligent ubiquitous services, basic infrastructures, such as a wearable PAN, must be provided.

In this paper, we proposed a wearable personal network (WPN): a four-layered wearable PAN protocol using FSB. WPN's light design is suitable for microprocessors and provides a profile layer for easy development of application programs. The resulting code size of the software protocol stack was 42 kB, and communication success rates between two WPN communication modules using a reliable mode were 99.44% (read) and 99.58% (write). We implemented FSB hardware which comprises the lower part of the data link layer and the physical layer on a small flexible PCB board. We also designed a WPN tutorial shirt to evaluate network communication performance and to demonstrate the use of the WPN protocol. When the conductivity of the conductive yarn decreased on the shirt, the WPN network adjusted the network baud rate from 10 Mbps to 1 Mbps in real-time. The resulting prototype shirt showed that the WPN protocol can handle various types of network traffic, such as sensory, command, and multimedia data, as well as adapt to communication line faults due to the abrasion of conductive yarns.

Concerning future applications, we are developing a new transceiver circuit that can measure more bus signal information, such as signal attenuation and impedance mismatches. Also, we plan to implement WCM hardware as a system-on-chip or as an intellectual property core for microcontrollers.

Acknowledgment

The authors would like to thank Nohwan Myung, In-Geol Baek, and Kyung-Min Park for their contribution in developing the data-link-layer protocol.

References

[1] R.L. Ashok and D.P. Agrawal, "Next-Generation Wearable Networks," *Computer*, vol. 36, no. 11, Nov. 2003, pp. 31-39.

[2] R. LaRowe and C. Elliott, "Computer Networks for Wearable Computing," *Fundamentals of Wearable Computers and Augmented Reality*, New Jersey, USA: Lawrence Erlbaum, 2001, pp.715-745.

[3] F. Mizuno et al., "Development of a Wearable Computer System with a Hands-Free Operation Interface for the Use of Home Health Caregiver," *Frontiers of Med. Informat.*, vol. 13, no. 4, July 2005, pp. 293-300.

[4] J.A. Gutiérrez, E.H. Callaway Jr., and R.L. Barrett Jr., *Low-Rate Wireless Personal Area Networks: Enabling Wireless Sensors with IEEE 802.15.4*, New York: IEEE Press, 2003, pp. 4-5.

[5] B. Koh and P.Y. Kong, "Performance Study on ZigBee-Based Wireless Personal Area Networks for Real-Time Health Monitoring," *ETRI J.*, vol. 28, no. 4, Aug. 2006, pp. 537-540.

[6] E.R. Post and M. Orth, "Smart Fabric or 'Wearable Clothing'," *Proc. 1st Int. Symp. Wearable Comput.*, Oct. 1997, pp. 167-168.

[7] M.M. Gorlick, "Electric Suspenders: A Fabric Power Bus and Data Network for Wearable Digital Devices," *Proc. 3rd Int. Symp. Wearable Comput.*, Oct. 1999, pp.114-121.

[8] E. Wade and H.H. Asada, "Wearable DC Powerline Communication Network Using Conductive Fabrics," *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2004, pp. 4085-4090.

[9] Z. Nakad, M. Jones, and T. Martin, "Communications in Electronic Textile Systems," *Proc. Int. Conf. Commun.*, June 2003, pp. 37-43.

[10] J. Yoo, S. Lee, and H.J. Yoo, "A 1.12 pJ/b Inductive Transceiver with a Fault-Tolerant Network Switch for Multi-Layer Wearable Body Area Network Applications," *IEEE J. Solid-State Circuits*, vol. 44, no. 11, Nov. 2009, pp. 2999-3010.

[11] N. Myung, I. Baek, and H. Choi, "Field-Bus Protocols for FAN," *Proc. Int. Conf. Next-Generation Computing*, Seoul, Rep. of Korea, Nov. 2008, pp. 98-102 (in Korean).

[12] J. Hännikäinen et al., "Conductive Fibres in Smart Clothing Applications," *Mechatronics for Safety, Security and Dependability in a New Era*, Elsevier B.V., 2006, pp. 395-400.

[13] R. Endo et al., "Preparation and Characterization of New Type PVA/CuxS Nano Composite Conductive Fiber," *Proc. 11th Int. Symp. Wearable Comput.*, Oct. 2007, pp. 125-126.

[14] B.S. Shim et al., "Smart Electronic Yarns and Wearable Fabrics for Human Biomonitoring Made by Carbon Nanotube Coating with Polyelectrolytes," *Nano Lett.*, vol. 8, no. 12, 2008, pp. 4151-4157.

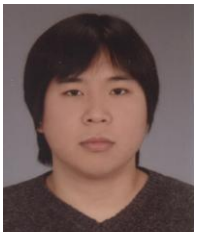
[15] S. Varnaitė and J. Katunskis, "Influence of Washing on the Electric Charge Decay of Fabrics with Conductive Yarns," *Fibres & Textile in Eastern Europe*, vol. 17, no. 5, 2009, pp. 69-75.

[16] J. Slade et al., "Washing of Electrotextiles," *Proc. Mater. Research Soc. Symp.*, vol. 736, 2003, pp. 99-108.

[17] H.S. Lee, J. Sunwoo, and D.W. Han, "Fabric Serial Bus: A Rate Adaptive Serial Bus Network for E-Textile Platform," *Proc. Int. Conf. Next-Generation Computing*, Seoul, Rep. of Korea, Oct. 2009, pp. 238-241.

[18] G. Tröster, "SoT: System on Textile for Wearable Computing," *The World of Electronic Packaging and System Integration*, eds. B. Michels and R. Aschenbrenner, 2004, pp. 114-119.

Hyung Sun Lee received his BS and MS degrees in electrical engineering from KAIST in 2000 and 2002, respectively, and his PhD in electrical engineering and computer science from KAIST in 2007. He is currently a senior researcher at ETRI, Daejeon, Rep. of Korea. His research interests include real-time embedded systems, wearable computing, and intelligent textiles.



Choong Bum Park received his BS in computer engineering from Kongju National University, Gongju, Rep. of Korea, in 2004. He currently works as a member of the research staff at Mobile Distributed Computing Laboratory in Chungnam National University. His research interests include mobile computing, wearable computing, ubiquitous computing, autonomic computing, and mobile learning service systems.

Kyoung Ju Noh received her BE and MS in computer sciences from Chonbuk National University, Rep. of Korea, in 1999 and 2001, respectively. She is currently a senior member of the engineering staff at ETRI, Daejeon, Rep. of Korea. Her research interests include computer network and data communication.

John Sunwoo received his BS and MS in electrical engineering from Auburn University in 2003 and 2005, respectively. Since 2005, he has worked at ETRI, Daejeon, Rep. of Korea, developing wearable computing systems. He is interested in designing a wearable BAN communication controller for e-textiles.

Hoon Choi received his BS in computing engineering from Seoul National University, Rep. of Korea, in 1983 and his MS and PhD in computer science from Duke University in 1990 and 1993, respectively. From 1983 to 1996, he was a senior member of technical staff at ETRI, Daejeon, Rep. of Korea, where he worked on LAN, broadband ISDN, and high-speed network systems. Since 1996, he has been with Chungnam National University. His research area is mobile/distributed computing. He has been involved in several research projects on middleware for distributed computing, mobile computing, and a software platform for cellular phone wireless applications. He is currently interested in autonomic computing and system software for wearable computers.

Il-Yeon Cho received his BS and MS in industrial engineering from Sungkyunkwan University, Rep. of Korea, in 1991 and 1993, respectively. He received his PhD in computer engineering from Chungnam National University, Rep. of Korea, in 2007. Since 1993, he has been working at ETRI, Daejeon, Rep. of Korea, as a senior member of the engineering staff. From 1995 to 1996, he was a visiting researcher at the Open Software Foundation (OSF) Research Institute, US, and conducted collaborative research. Currently, he is the head of the Wearable Computing Research Team at ETRI. He is interested in developing wearable computers and embedded systems.