Distributed Computing System Based on Microprocessor Cluster for Wearable Devices

Xin Liu
College of Information Science and Engineering Ocean University of China
Qingdao, China
E-mail: liuxinouc@126.com

Zhiqiang Wei*
College of Information Science and Engineering Ocean University of China
Qingdao, China
*The corresponding author
E-mail: liuxinouc@126.com

***Abstract*—Wearable equipment in recent years has been rapid development. But the hardware manufacturing complexity and the high cost is a real problem. This paper introduces a microprocessor cluster with both hardware design principle and related distributed software design methods. This cluster has the characteristics of low cost, high reliability, flexible hardware and software system structure, low power consumption, simple equipment manufacturing process and so on, especially suitable for wearable equipment. This article discusses the hardware and software design methods in detail, as well as the complete process of the across-node communication module. In order to verify the principle of the design, we created a prototype test machine which consists of an ARM Cortex-M4 core microprocessor and 10 ARM Cortex-M0 core microprocessors through the UART serial interconnection to form a star network and carried out an experimental about the ECG feature extraction operation. Experimental results show that the performance of the cluster can be compared with a Cortex-A7 high-performance embedded processor, but the microprocessor cluster system is less expensive and has a superior cost-effective.**

Keywords-Microprocessor; Cluster; Distributed software; Wearable equipment; ECG feature extraction

I. INTRODUCTION

Wearable equipment in recent years has been rapid development. Whether in the consumer electronics market or the traditional sports apparel market, a variety of wearable devices continues to emerge. Such as an electronic wristband capable of detecting an exercise amount, a smart wristwatch capable of detecting blood oxygen and pulse rate, a sports vest with a heart rate detecting function, and a running shoe capable of detecting the sole pressure. These devices through a long time to detect human life parameters, access to traditional medical instruments can not be collected for a long time continuous data. By digging deeper into the data, the researchers found a series of hidden data features and were able to predict the health of the user for a long time.

Researchers have put forward a variety of solutions for wearable devices. On the one hand, the device is made thinner, lower power consumption, longer standby time, the sensor cable connection between the wireless connection is replaced. On the other hand, the computing power of the device is stronger, and some algorithms that need to be calculated by means of the server, after optimization, can run directly on the device.

There are many researchers working on computational methods. Through a variety of pretreatment techniques, the researchers improved the accuracy of machine learning classification to a practical level. Shereena Shaji et al. [1] increased the accuracy of certain motion recognition to 96.66%. Chin-Teng Lin et al. [2] designed an algorithm that can run on a mobile phone and detect ECG atrial fibrillation in real-time. Shigenori Shirouzu et al. [3] used a wearable device to study the relationship between electrocardiogram and sleep quality. However, the ability to enhance the software is very easy to encounter the ceiling. When some algorithms are further simplified, their accuracy will also decrease. This is an unquestionable fact.

To improve the hardware, for example, the traditional ECG acquisition requires a signal cable up to 10 cables, carrying these wires is very inconvenient for patients. Geng Yang et al. [4] proposed a sensor-based data aggregation method based on the serial bus, which solves the multi-point ECG signal acquisition problem with a single cable. In addition, how to reduce the volume, improve the integration of researchers is also concerned about the direction. G. Kavya and V. ThulasiBai [5] used an Altera FPGA chip to simulate a dual-processor dedicated system and used a parallel computation method to process the ECG signal acquisition and analysis. Jia-Hua Hong et al. [6] designed two special-purpose integrated circuit chips. One of the wireless sensor chip power consumption is very low, access to a very long battery standby time. Another receiver chip is integrated DSP processing heartbeat detection and signal classification. Shih-Lun Chen [7] designed a dedicated chip that uses less gate count and chip area to achieve higher performance.

Based on the conclusions of the researchers, we can see that highly integrated custom chips have an overwhelming advantage in terms of power consumption and performance. However, the cost of custom chips is extremely expensive. Although the traditional general-purpose microprocessors, relatively large size, and power consumption, but the technology is mature, the risk is small, and has a wealth of software and hardware resources, can reduce R & D risk. The scheme proposed in this paper is different from the traditional single-chip solution: the data processing and analysis functions are distributed to dozens of single-chip microprocessors to form a distributed microprocessor cluster. As long as the appropriate software design, the cluster can play a comparable performance of highly integrated microprocessors, and software development will not be significant difficulties.

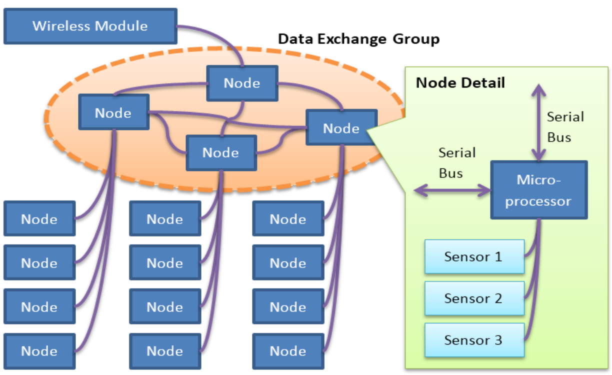

Figure 1. Block diagram of microprocessor cluster.

II. BLOCK DESIGN OF THE MICROPROCESSOR CLUSTER

A. *Hardware Block*

The hardware structure of the microprocessor cluster as shown in Fig .1. A cluster consists of many nodes. The core of each node device is a microprocessor. At least one UART serial port is reserved for each processor. The other ports can be used to connect a variety of sensor devices. There are also nodes located in the "Data Exchange Group", which participate in the calculation in addition to data exchange, GPIO and AD ports can be connected to a number of sensors. Data Exchange Group node compared with ordinary nodes, it needs to have a lot of UART serial port to form a Mini network. If there are enough serial ports on the chip, it is best to have a direct communication link between every two nodes. Otherwise, some inter-chip communication to be relayed through the intermediate nodes will cause performance degradation. For the microprocessor communication overhead is not negligible.

B. *Software Design*

Compared with the difficulty of hardware design, cluster software design is much more difficult. The same is true for clusters of supercomputers made up of rack-mounted servers, which are far more complex than the average desktop software.

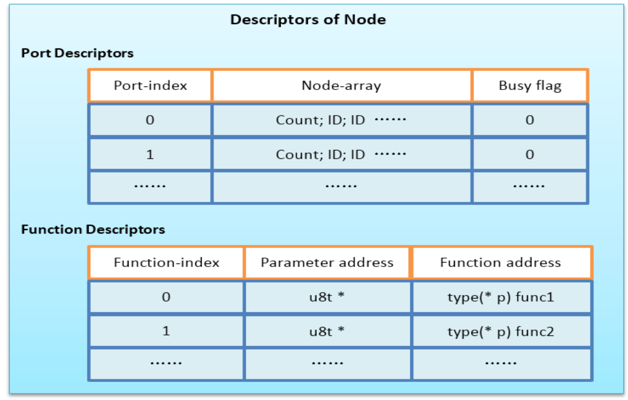

Figure 2. Node data model.

TABLE I. PORT DESCRIPTION FORMAT SPECIFICATION.

Field name	Field format	Remarks
Port-index	Single-byte data, the range of 0 ~ 255	
Node-array	The Node-ID queues which can arrive from this port. It fixed the length of 15 bytes. Byte 1: the number of nodes n, or, queue length. Bytes 2~15: an array of node ID, only the front “queue length” IDs is available.	Because microprocessors have no more than 14 serial ports, thus reserved 14 queue length is enough.
Busy flag	Using a byte to indicates the Busy State	1 means busy and 0 means available

Simply put, the cluster nodes in the device is divided into two categories: communication nodes and computing nodes. On a server cluster, communication functions are handled by multi-layer network switches and load balancing machines, and computing functions are handled by the rack server. In this paper, the microprocessor cluster, each microprocessor has both communication nodes and computing nodes of the two functions. This paper presents a common data model to manage the functions of these two nodes, as shown in Fig .2.

Communication is relatively independent, we first introduce the communication function. The communication function module does not generate new data and does not initiate data transmission. The new data transmission process is initiated by the calculation function module or the timer task initiative.

A Port Descriptor binds a port together. First, for each node in the cluster an ID should be assigned. Note that the ID 0 is called "the Debug Node", dedicated to using for debugging and testing the whole cluster, we will discuss the details later. Node-ID starting from 1. In general, the cluster system can't possess more than 254 microprocessors, so with 1 byte of data to store the Node-ID is enough. The value 255 is used to represent "invalid Node-ID", in some cases it may be used. Port Descriptor indicate the port linked to which nodes, and whether the port was occupied by a Communications Task, in other words, “a Busy State”. The Descriptor is shown in Table I.

So, how to deal with data relay? For example, in the simple example shown in Fig .3, the Node A send a datagram to the Node C through the Node B.

The effect of the Port Descriptors is the same as "phone book". It is stored in RAM and the microprocessor may access it at any time. When the microprocessor receives a datagram, it must first check the head of the datagram to see if the destination address is the current node. If the destination is this node, then deal with it. If not, then check the "phone book" again and select a port to send the datagram out. In the head of the datagram, therefore, requires a specific format to indicate the destination Node-ID and other information. The format of the datagram is shown in Table II.

Because the hardware connection between microprocessors has a high reliability, we need no redundancy check mechanism. We should save the communication flow. Datagram header information specified that on which function process the data, and transmit the results to which node which function. This mechanism separates the function parameters and the function returns, which can realize good flexibility, such as data collection, data preprocessing, data compression, and

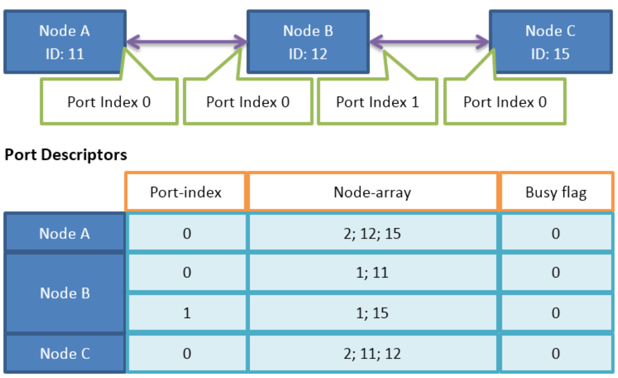

Figure 3. Examples of Port Descriptors.

TABLE II. THE FORMAT OF THE DATAGRAM.

Field name	Field format	Remarks
Node-ID	Single-byte data, the range of 0~254	Destination Node-ID
Function-index	Single-byte data, the range of 0~255	Indicate which function is responsible for this data processing
Node-ID-return	Single-byte data, the range of 0~255	The Node-ID of the result receiving node. 255 means "No need to deal with the return value"
Function-index -return	Single-byte data, the range of 0~255	The function index in the receiving node which is used to handle the returned data.
Data-length	Double-byte, the range of 0~4096	The value of Data-length can be 0. As the microprocessor's RAM is very small, we limit the value up to 4096, to avoid memory overflow.
Data-data	Binary queue	Length of Data-length byte streams

so on. After processing the input data, a function does not need to return the results to the source Node but comply with the Node-ID-return sent to the destination node directly. Thus avoid the unnecessary reciprocating data transmission.

As mentioned before, we called the ID 0 Node the Debug Node. When we debugging on a Node in the cluster, we can create a datagram and set its Node-ID-return value to 0 and send the datagram to the Test Node. Then, when the Test Node completes the calculation, the results will be sent to the Debug node automatically. We can install a special port in the Debug Node, such as a USB-to-Serial Bridge, to forward the datagram to PC, so that we can monitor the results easily.

C. *Distributed Computation*

When the above-mentioned communication system is constructed, the design of the calculation method becomes very simple. In practical engineering, we must first consider two issues:

1, the parameters of which nodes?

2, in which node function?

In general, the parameters are sent from a sensor node, and the function is distributed in many processors. We need a series of strategies to ensure that the parameters can be passed to the appropriate function node.

To illustrate this problem, we designed an example of the need for high-intensity computing - ECG Feature

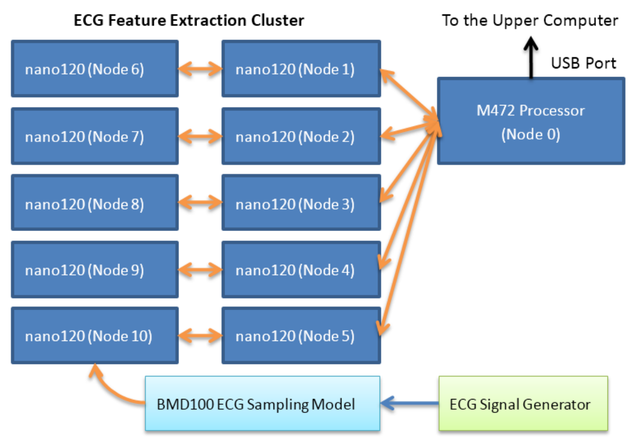

Figure 4. Block diagram of the ECG Feature Extraction Cluster.

Extraction cluster. It consists of a Nuvoton M472 microprocessor and 10 Nuvoton NANO120LD3NA microprocessors (nano120) form. M472 does not participate in Feature Extraction calculation, it is only responsible for calculating the results of a summary upload. It has six UARTs and one USB port. USB interface for communication with the host computer. One of the six serial ports is used for debugging and the remaining five serial ports are connected to the serial ports of five nano120 processors (Node 1 to Node 5). Each nano120 is also connected to another nano120 (Node 6 to Node 10). ECG sampling chip Neurosky BMD100 connected to one of the nano120 (Node 10) serial port. The ECG signal comes from an ECG signal generator. The hardware structure of the whole verification system is shown in Fig .4.

Node 10 is a parameter node. Node 1 to Node 9 is function nodes. M472 is the data summary node. The serial baud rate between Node 10 and BMD100 is 57600. While the microprocessor is used at 115200 baud rate.

Each function node at the beginning of the calculation, the first statement to the M472 himself in a busy state, after the completion of the function calculation, and then notify the M472 to cancel their busy state. The Node 10 node first asks the M472, "Who will process the data", and sends the parameter data to the designated node after obtaining the reply from M472.

The interaction timing of each node is shown in Fig .5. Node 10 prepares a data window queue, which has three windows, in order to fill in the received ECG data. The data length of each window is two heartbeat cycles plus 120 sampling points (about 0.5 seconds of data), and the window length will vary depending on the heart rate. This length is to ensure that there are two complete ECG waveforms in each window. These three data windows are followed by a delay of one heartbeat cycle distance. When a window fills up the data, Node 10 initiates a communication process to send the window data to an idle nano120 processor. As long as the remaining 9 nano120 are not in a busy state, the whole system will not miss the ECG data. When the window data transmission is complete, the window is cleared and moved to the end of the windowing queue. Since Node 10 detects the QRS Complex and the task of allocating data to the window is heavy, it no longer assumes the Feature Extraction calculation task.

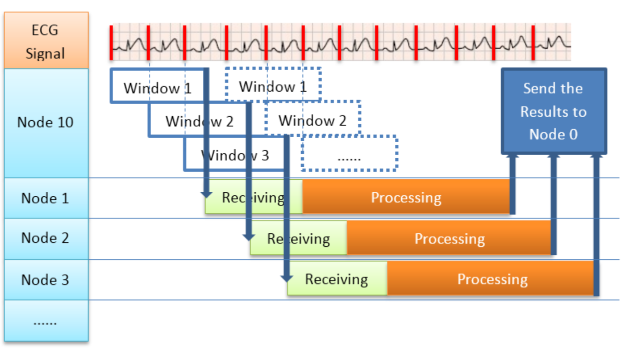

Figure 5. System Collaboration Diagram.

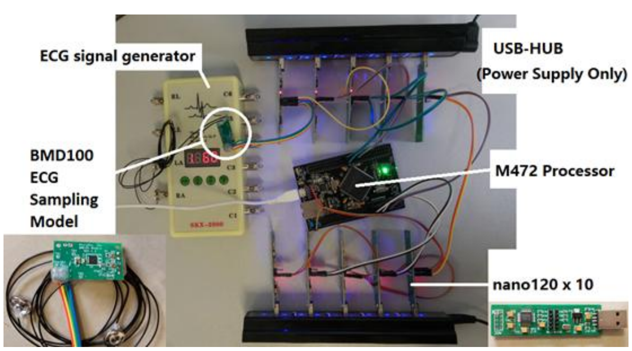

Figure 6. Hardware of the ECG Feature Extraction Cluster.

III. EVALUATION OF DISTRIBUTED COMPUTING PERFORMANCE

The completed ECG Feature Extraction cluster hardware is shown in Fig .6. We have statistics on these 9 nano120 processors. The Receiving process is relatively fixed, in the 68 ~ 75ms range, which is ECG signal generator output signal of the heart rate is fixed. When the communication baud rate is 115200, the time required to receive 620 bytes, in theory, is about 65ms. Most of the processing time-consuming and separate tests of the situation is similar to individual cases, the processing time-consuming process up to 6530ms, this is because Node 0 is sometimes communicating with the host computer, affecting the data transfer. The task assignment of the microprocessor cluster is shown in Fig .7. The numbers on the Timeline bar in the figure indicate the order of the tasks. If more than one nano120 is idle at the same time, the M472 preferentially assigns the task to a smaller number of processors.

In the experiments, we found an interesting phenomenon, the Node 8 and Node 9 has never been assigned for tasks, and Node 7 had less opportunity to be assigned for. This shown that a Cluster with 8 nano120 microprocessors was powerful enough to cope with the current computational tasks. We obtained the calculation result after the heart beat for 3 to 5 seconds, for ECG automatic diagnosis application such a delay is acceptable.

Feature Extraction is a computationally intensive process that involves filtering the signal several times and repeatedly scanning to determine the location of each wave group boundary and extremum. This process takes hundreds of milliseconds on a PC or smartphone. We tested a Spreadtrum SC9830A processor (based on Java) on a cell phone with an ARM Cortex-A7 core at 1.5 GHz and an average time of 112 ms for Feature Extraction for single-channel ECG lead data. We then ran the same test on a nano120 microprocessor (based on a C program), which took about 20 times more time. The test results are shown in Table III and Fig .8.

Compared with the SC9830A processor, nano120 microprocessor computing time is almost 20 times the former. But it's the bulk price of less than 1 US dollars, known as one dollar computer. Nano120 integrated RAM, Flash, does not require external expansion memory. ECG Feature Extraction cluster system hardware costs only 12 dollars, much cheaper than the SC9830A processor, but also to complete the same computing tasks.

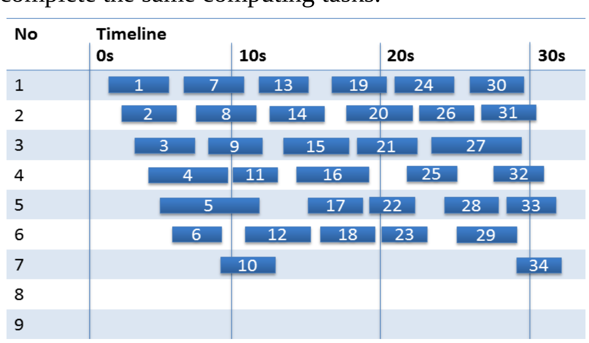

Figure 7. Task Allocation on Microprocessor Cluster.

TABLE III. SPEED COMPARISON OF THE FEATURE EXTRACTION PROGRAM ON THE SC9830A AND NANO120

No.	Consumed time (MS)	
	SC9830A	***nano120***
1	109	2398
2	110	2087
3	122	2256
4	112	2443
5	111	2299
6	108	2573
7	112	2019
8	113	2221
9	109	2320
10	109	2702
11	120	2249
12	113	2501
Mean:	112.33	2339

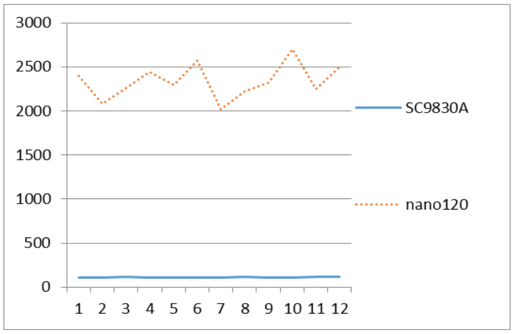

Figure 8. Speed Comparison of the Feature Extraction Program on the SC9830A and nano120.

IV. CONCLUSION AND FUTURE CONSIDERATIONS

As a popular saying goes, two heads are better than one. The same is the case with microprocessors. Described in this article the microprocessor cluster has good price-performance ratio. Its price is low and only need lower production conditions. In the aspect of software development, in order to achieve higher performance, you need to follow certain design rules. A well-designed distributed computing system can solve the problems with quite complex computation.

Outlook to the future, when wearable devices entered into the people's daily life, you can't find a part with the name "mainboard" on these devices. Every sensor node is a microprocessor, they communicate with each other via the textile fiber cable or wireless networks. With low manufacturing cost, small enough volume and long service life, the wearable devices can completely combine with clothing. Just like a mobile phone has cameras today, clothing in the future will own intelligence.

ACKNOWLEDGMENT

This work is supported by Major Projects of Shandong (No.2015ZDZX05002) and Key Research and Development Project of Shandong (No.2016ZDJS09A01)

REFERENCES

[1] Shereena Shaji, Maneesha Vinodini Ramesh and Vrindha N. Menon, “Real-Time Processing and Analysis for Activity Classification to Enhance Wearable Wireless ECG”, Proceedings of the Second International Conference on Computer and Communication Technologies. Springer India, 2016, pp.21-35

[2] Chin-Teng Lin, Kuan-Cheng Chang, Chun-Ling Lin, Chia-Cheng Chiang, Shao-Wei Lu, Shih-Sheng Chang, Bor-Shyh Lin, Hsin-Yueh Liang, Ray-Jade Chen, Yuan-Teh Lee and Li-Wei Ko, “An Intelligent Telecardiology System Using a Wearable and Wireless ECG to Detect Atrial Fibrillation”, IEEE Transactions on Information Technology in Biomedicine, Vol.14, No.3, 2010, pp. 726-733

[3] Shigenori Shirouzu, Yumeka Seno, Ken Tobioka, Tomoko Yagi, Toshiharu Takahashi, Mitsuo Sasaki and Hisanobu Sugano, “For Children’s Sleep Assessment: Can we trace the change of sleep depth based on ECG data measured at their respective home with a wearable device?”, Ieee-Embs International Conference on Biomedical and Health Informatics IEEE, 2016, pp.208-211

[4] Geng Yang, Jian Chen, Ying Cao, Hannu Tenhunen, and Li-Rong Zheng, “A Novel Wearable ECG Monitoring System Based on Active-Cable and Intelligent Electrodes”, International Conference on E-Health Networking, Applications and Services, 2008. Healthcom IEEE, 2008, pp.156-159

[5] G.Kavya and V.ThulasiBai, “Wearable advanced single chip ECG telemonitoring system using SoPC”, Ieice Electronics Express, Vol.11, No.6, 2014, pp.1-10

[6] Jia-Hua Hong, Shuenn-Yuh Lee, Member, IEEE, Ming-Chun Liang, Cheng-Han Hsieh, and Shih-Yu Chang Chien, “A Wireless ECG Acquisition and Classification System for Body Sensor Networks”, Engineering in Medicine and Biology Society IEEE, 2013, pp.5183-5186

[7] Shih-Lun Chen, Min-Chun Tuan, Tsun-Kuang Chi, and Tin-Lan Lin , “VLSI architecture of lossless ECG compression design based on fuzzy decision and optimisation method for wearable devices”, Electronics Letters, Vol.51, No.18, 2015, pp.1409-1411

[8] Joseph J. Oresko, Zhanpeng Jin, Jun Cheng, Shimeng Huang, Yuwen Sun, Heather Duschl, and Allen C. Cheng, “A Wearable Smartphone-Based Platform for Real-Time Cardiovascular Disease Detection Via Electrocardiogram Processing”, IEEE Transactions on Information Technology in Biomedicine A Publication of the IEEE Engineering in Medicine & Biology Society, Vol.14, No.3, 2010, pp.734-740

[9] Jian Kang Wu, L. Dong, X. Chen, Wee Soon Yeoh, and Isaac Pek, “Ambulatory Examination and Management of CVD Patients”, IEEE Biomedical Circuits and Systems Conference IEEE, 2007, pp. 199-202

[10] Kai Yang, Zhuan He, Wendi Yang, Qi Tang, Dongmei Li, and Zhihua Wang, “Heart Sound Denoising Using Computational Auditory Scene Analysis for a Wearable Stethoscope”, IEEE, International Midwest Symposium on Circuits and Systems IEEE, 2013, pp.1220-1223

[11] Taegyun Jeon, Byoungho Kim, Moongu Jeon and Byung-Geun Lee, “Implementation of aportable device for real-time ECG signal analysis”, BioMedical Engineering OnLine, Vol.13, No.1, 2014, pp. 160

[12] Shuang Zhu, Jingyi Song, Balaji Chellappa, Ali Enteshari, Tuo Shan, Mengxun He, and Yun Chiu, “A Smart ECG Sensor with In-Situ Adaptive Motion-Artifact Compensation for Dry-Contact Wearable Healthcare Devices”, International Symposium on Quality Electronic Design, 2016, pp.450-455

[13] G. Tortora, R. Fontana, S. Argiolas, M. Vatteroni, P. Dario, M.G. Trivella, “A dynamic control algorithm based on physiological parameters and wearable interfaces for adaptive ventricular assist devices”, International Conference of the IEEE Engineering in Medicine & Biology Society, 2015, 4954

[14] Chao-Ting Chu, Huann-Keng Chiang, and Jian-Jie Hung, “Dynamic Heart Rate Monitors Algorithm for Reflection Green Light Wearable Device”, International Conference on Intelligent Informatics and Biomedical Sciences IEEE, 2015, pp.438-445

[15] Sungmook Jung, Jongsu Lee, Taeghwan Hyeon, Minbaek Lee, and Dae-Hyeong Kim, "Fabric-Based Integrated Energy Devices for Wearable Activity Monitors", Advanced Materials, Vol.26, No.36, 2014, pp.6329-6334

[16] Ya-Li Zheng, Bryan P. Yan, Yuan-Ting Zhang, and Carmen C. Y. Poon, "An Armband Wearable Device for Overnight and Cuff-Less Blood Pressure Measurement", IEEE transactions on bio-medical engineering, Vol.61, No.7, 2014, pp.2179-2186

[17] Min Chen, Yin Zhang, Yong Li, Mohammad Mehedi Hassan, and Atif Alamri, "AIWAC- Affective Interaction Through Wearable Computing and Cloud Technology", IEEE Wireless Communications, Vol.22, No.1, 2015, pp.20-27

[18] Gianmarco Angius, Luigi Raffo, "Cardiovascular Disease and Sleep Apnoea: a Wearable Device for PPG Acquisition and Research Aims", Computing in Cardiology, Vol.39, 2012; pp.513-516

[19] Biyi Fang, Nicholas D. Lane, Mi Zhang, Aidan Boran, Fahim Kawsar, "BodyScan : Enabling Radio-based Sensing on Wearable Devices for Contactless Activity and Vital Sign Monitoring", The International Conference, 2016, pp.97-110

[20] A.J. Cook, G.D. Gargiulo, T. Lehmann and T.J. Hamilton, "Open platform, eight-channel, portable bio-potential and activity data logger for wearable medical device development", Electronics Letters, Vol.51, No.21, 2015, pp. 1641-1643