2017 6th MEDITERRANEAN CONFERENCE ON EMBEDDED COMPUTING (MECO), 11-15 JUNE 2017, BAR, MONTENEGRO

Prototype of Group Heart Rate Monitoring with NODEMCU ESP8266

Andrej Škraba, Andrej Koložvari, Davorin Kofjač
University of Maribor
Cybernetics & Decision Support System Laboratory
Faculty of Organizational Sciences
Kidričeva cesta 55a, Kranj, Slovenia

Radovan Stojanović
University of Montenegro
Faculty of Electrical Engineering
Džordža Vašingtona bb., Podgorica, Montenegro

Vladimir Stanovov, Eugene Semenkin
Siberian State Aerospace University
Computer Science and Telecommunications Institute
31, Krasnoyarsky Rabochy Ave., Krasnoyarsk,
Russian Federation

Abstract — **Paper describes the development of prototype that enables monitoring of heart rate and inter beat interval for several subjects. The prototype was realized using ESP8266 hardware modules, WebSocket library, nodejs and JavaScript. System architecture is described where nodejs server acts as the signal processing and GUI code provider for clients. Signal processing algorithm was implemented in JavaScript. Application GUI is presented which can be used on mobile devices. Several important parts of the code are described which illustrate the communication between ESP8266 modules, server and clients. Developed prototype shows one of the possible realizations of group monitoring of biomedical data.**

Keywords- heart rate; pulse sensor; bluetooth; nodejs; cloud; ESP8266

I. Introduction

Monitoring of patients' heart rate in the hospitals often needs to be done remotely. This provides higher level of control and enables more rational work. Detecting heart rate is one of the main tasks that has been addressed historically [1] as well as recently [2–5]. While the algorithms and analog electronics solutions have been provided and are well developed, the new technologies from the field of Internet of Things (IoT) provides new possibilities to ensure the connectivity to the web and the cloud. In our previous research, we have analyzed several setups to provide data to the cloud such as Bluetooth, Bluetooth LE (BLE) and ESP8266 [6]. It has been demonstrated, that Bluetooth LE is best choice regarding the energy consumption while the application of ESP8266 module to stream the heart rate pulse data to the cloud was technically most convenient to realize. This is of prime importance if one needs to provide monitoring of, for example, 10 patients remotely since the development of software to provide control gets complex as the number of monitored devices increases.

In the present paper, the prototype for monitoring several patients heart rate on the web will be described. ESP8266 module was used, not only due to the issues of simple implementation but also due to the fact, that the module is extremely affordable regarding the price. It is therefore possible to upgrade the existing professional equipment with affordable solution to stream the data to the cloud and provide remote monitoring. The system should therefore provide the following capabilities:

a) Capability of reading the signal from the low cost PPG (photoplethysmographic) ear clip sensor

b) Provide the raw data over analog pin wirelessly over web socket

c) Provide real-time data processing computing of heart rate and Inter Beat Interval (IBI)

d) Provide cloud infrastructure to serve the code with algorithms and user interface to the clients which could be different devices (PC, phone, Smart TV etc.)

e) Multiple devices could be connected to provide efficient monitoring

f) Provide uniform GUI with interactive graphics of main parameters.

II. System Architecture

Fig. 1 shows the system architecture where several subjects are equipped with ear clip PPG heart rate sensors. The sensor output is amplified and filtered and then passed to the analog ADC0 pin of the ESP8266 NODEMCU module. With the battery and analog amplifier and filter circuit, this is an entire setup that is on the side of the subject for which we monitor biomedical data. ESP8266 module has integrated WiFi which provides connection to the WiFi router, gathering the signals from the sensors. For monitoring, the device should be connected to the IP of the code hosting server. There are two possibilities to perform signal processing and provide GUI to the user: a) the code could be run directly from the phone or tablet, PC or other device. Here we only start an html page directly from the phone and in this case the server is not needed. However it is much more convenient, that the signal processing and GUI code is provided by the server. This is also convenient for device detection and management. The MAC addresses from the ESP8266 modules could be fixed in the router in order to have better control over devices. There are also other possibilities to identify the devices [7; 8].

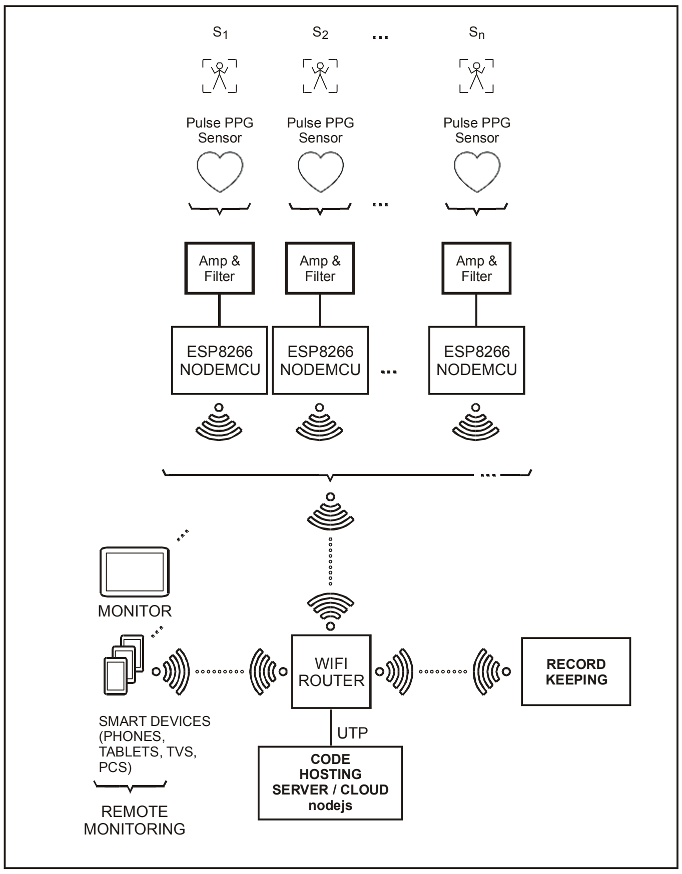

Figure 1: System structure for gathering multiple biomedical data from several subjects by ESP8266 modules and provision of processing with GUI

The ESP8266 board used was NodeMCU 1.0 (ESP-12E Module) with CPU Frequency: 80MHz and Flash Size: 4M (3M SPIFFS).

III. SOFTWARE CONSIDERATIONS

The code consisted of three parts: a) Code on ESP8266 module b) Code for signal processing and GUI and c) server code to provide the code to clients. Code on ESP8266 was uploaded using serial protocol with upload speed 115200 over port /dev/ttyUSB0 in Linux Ubuntu 12 by the application of Arduino IDE 1.6.8. The main part of the c code on the ESP8266 module is inclusion of h file with WebSocket Server (`#include "WebSocketsServer.h"`). Code for signal processing and GUI is shown in Fig. 2. This is JavaScript code that is part of the .html file. As mentioned, this part of the code could be run from the phone and the connection over WebSocekt would instantly be established. One can observe, that we have two connections, for two sensors that were used at the development of prototype. Each of the sensors has it's own IP on which the pulse signal is streamed. The part for the main code is executed on the receipt of new signal in the part marked as: `<... PPG Sensor processing code ...>`.

```
var connection1 = new
WebSocket('ws://192.168.1.112:81/', ['arduino']);
var connection2 = new
WebSocket('ws://192.168.1.114:81/', ['arduino']);
...
connection2.onopen = function () {
setInterval(function() {
   tempval2++
   if(messageReceived2)
      connection2.send('tempval: ' + tempval2);
      messageReceived2 = false;
    }, 40);
};
connection2.onerror = function (error) {
   console.log('WebSocket Error ', error);
};
connection2.onmessage = function (e) {
<... PPG Sensor processing code ...>
}
```

Figure 2: Part of the client side code where two connections are established over WebSocket for communication with two ESP8266 modules.

Fig 3. shows the part of the code, that is used on the side of the server. Here nodejs server is used in order to serve the signal processing and GUI code to the clients. This is also useful for providing different user interfaces for example admin/user interface. For this functionality the express library was used.

```
var http = require('http'),
express=require('express'), fs = require('fs'),
app=express(), server;

var server = http.createServer(app).listen(8080);
var path = require("path");

app.use(express.static('public'));
app.get('/',function(req,res){

res.sendFile(path.join(__dirname+"/heart01.html"));
});
```

Figure 3: Part of code on server for serving html page with JavaScript code for signal processing and GUI

On the nodejs server side the entire signal processing could be performed. Weather the processing is done on the client or on the server side depends on the complexity of applied algorithms and wireless network connection speed. In our case, there was no major difference if the processing code was ran on the client or on the server side. Regarding the redundancy in communication it is better, that the processing is performed on the client side.

Important part of the software stack is c code residing on the ESP8266 module. In our case, where we stream pulse data in real time it is important, that WebSocket library provides reliable data transmission. In our tests the ESP8266 proved to be suitable for the task however, additional processing power would contribute to better operation. All signal processing and GUI was coded in JavaScript including the server side on nodejs [10].

IV. HARDWARE REALIZATION WITH RESULTS IN THE BROWSER

Fig. 4. shows the components of the heart rate real time streaming system which consists of: 1) Pulse PPG sensor, 2) 3.5 Jack input to the analog heart rate amp board 3) Operational amplifier LM324 circuit to provide the analog data of the pulse to ESP8266, 4) Node MCU LUA ESP8266 module, 5) AA accumulator battery pack with ON/OFF switch. This is one of the two boards that were used in prototype setup to stream data of two different subjects to the clients.

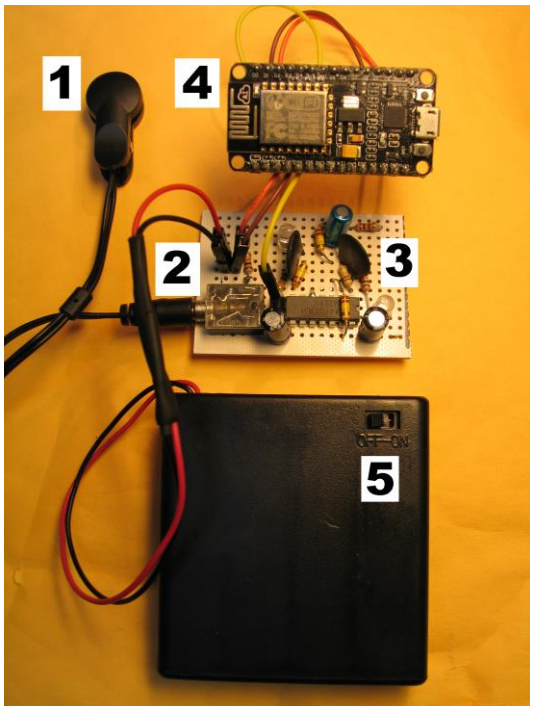

Figure 4: Components of the heart rate real time streaming system, here one of two identical setups is shown

Fig. 5. Shows an example of monitoring two patients at the same time. The data is gathered from two ESP826 modules and feed to the client.

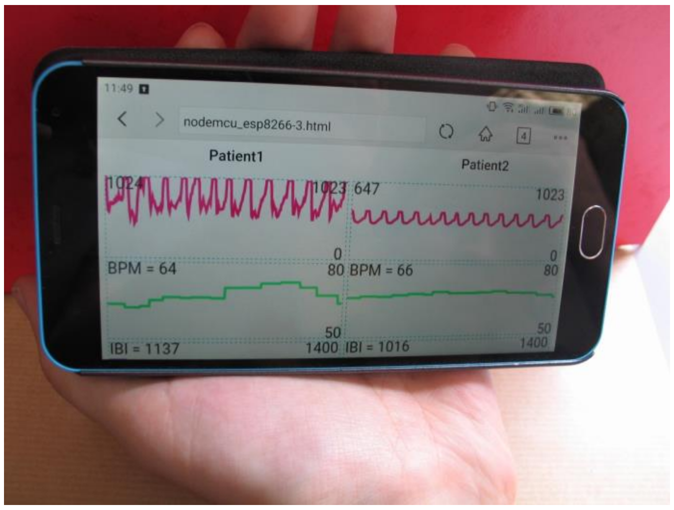

Figure 5: Output of the two sensors from two different patients

One can observe raw data, BPM and IBI on the mobile phone. The monitoring could be expanded to more devices in parallel and in different places. If needed, since the signal processing and GUI is developed in html / JavaScript, the mobile device native applications could easily be developed [16].

Fig. 6. shows the monitoring of two patients in the web browser. The monitoring could be performed on other smart devices as for example Smart TV. The data is organized in two columns one column for each subject. The interface could be reached by entering the IP address and port of the nodejs server into the web browser.

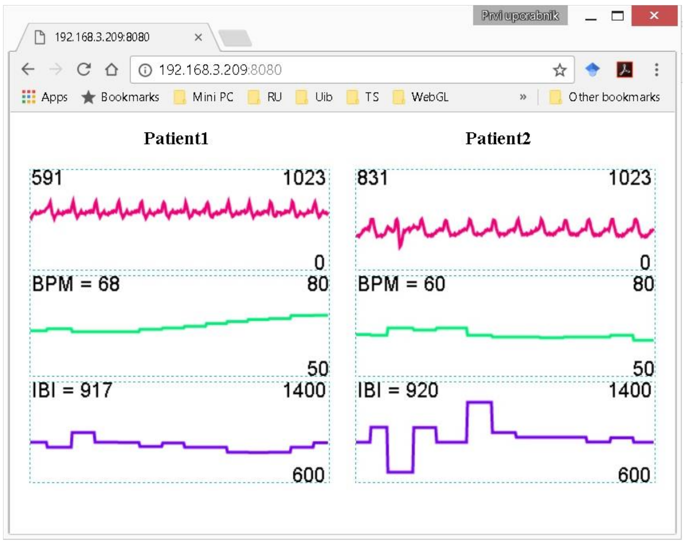

Figure 6: Example of the patient monitoring in the web browser

Hardware realization was straight-forward. The ESP8266 module has one analog ADC0 pin which was used for transmitting raw data. Low cost PPG sensor proved to be useful. Entire setup is mobile, which is important for other applications [11; 12; 13; 14; 15], for example on the wheelchair.

V. CONCLUSION

Monitoring of the several subjects' biomedical data has been successfully implemented by the application of ESP8266 modules. Application of WebSockets IP protocol enables simple realization. The hardest part of the implementation is providing microcontroller c code on the ESP8266 module itself. When connected, all the development was done in JavaScript which is convenient both on the client and server side.

Since the algorithm to extract heart rate from raw data was not too complex, it was convenient, that the processing was done on the client side. By this, the server could be omitted, which might be useful in some cases. JavaScript has been successfully used for monitoring the hardware [16]. Provided prototype shows one of the possible realizations of mass monitoring of biomedical data which is of interest in the era of Internet of Things and Cyber-physical systems. Further development will go in the direction of merging analog circuitry with ESP8266 modules and providing standard interfaces to put the current data from professional hear rate monitoring equipment to the cloud. This are great new opportunities to improve biomedical data acquisition as well as processing and monitoring.

ACKNOWLEDGMENT

This work was supported in part by the Slovenian Research Agency (ARRS) (Research program "Decision support systems in electronic commerce", program No.: UNI-MB-0586-P5-0018), ARRS SI-RF bilateral project "Efficient Control of Cyber-physical Systems & Internet of Things by the Application of Evolutionary and Biologically Inspired Algorithms" Proj. No.: BI-RU/16-18-040, ARRS SI-MNE bilateral project "Development of Speech Controlled Wheelchair for Disabled Persons as Cyber-Physical System" Proj. No.: BI-ME/16-17-022 and and Russian Federation Presidential Scholarship No. 16-in-689.

REFERENCES

[1] J. Pan; W. J. Tompkins, "A Real-Time QRS Detection Algorithm," in IEEE Transactions on Biomedical Engineering, vol. BME-32, no. 3, pp. 230-236, March 1985. doi: 10.1109/TBME.1985.325532

[2] Z. Piotrowski; K. Rózanowski, Robust Algorithm for Heart Rate (HR) Detection and Heart Rate Variability (HRV) Estimation. Acta Physica Polonica A. Vol. 118 (2010) No. 1 pp. 131 – 135.

[3] M. A. Motin; C. Karmakar; M. Palaniswami, "Ensemble Empirical Mode Decomposition with Principal Component Analysis: A Novel Approach for Extracting Respiratory Rate and Heart Rate from Photoplethysmographic Signal," in IEEE Journal of Biomedical and Health Informatics , vol.PP, no.99, pp.1-1, doi: 10.1109/JBHI.2017.2679108

[4] K Perakis, M. Haritou, R. Stojanovic, B. Asanin and D. Koutsouris. Wireless patient monitoring for the e-inclusion of chronic patients and elderly people. Proceedings of the 1st international conference on Pervasive Technologies Related to Assistive Environments. ACM 2008/7/16.

[5] Kovacevic, Jovan; Stojanovic, Radovan; Karadaglic, Dejan; Asanin, Bogdan; Kovacevic, Zivorad; Bundalo, Zlatko; Softic, Ferid, "FPGA low-power implementation of QRS detectors," in Embedded Computing (MECO), 2014 3rd Mediterranean Conference on , vol., no., pp.98-101, 15-19 June 2014 doi: 10.1109/MECO.2014.6862667

[6] A. Škraba; A. Koložvari; D. Kofjač; R. Stojanović; V. Stanovov; E. Semenkin, Streaming pulse data to the cloud with bluetooth LE or NODEMCU ESP8266. Embedded Computing (MECO) 5th Mediterranean Conference on, Bar, Montenegro. 12-16 June 2016, pp. 428 – 431.

[7] Kliem, A.; Koner, M.; Weissenborn, S.; Byfield, M., "The Device Driver Engine - Cloud enabled ubiquitous device integration," in Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), 2015 IEEE Tenth International Conference on , vol., no., pp.1-7, 7-9 April 2015, doi: 10.1109/ISSNIP.2015.7106921

[8] Biswas, J.; Maniyeri, J.; Gopalakrishnan, K.; Shue, L.; Eugene, P.J.; Palit, H.N.; Foo Yong Siang; Lau Lik Seng; Li Xiaorong, "Processing of wearable sensor data on the cloud - a step towards scaling of continuous monitoring of health and well-being," in Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE , vol., no., pp.3860-3863, Aug. 31 2010-Sept. 4 2010, doi: 10.1109/IEMBS.2010.5627906

[9] https://expressjs.com/ (Accesed, 10.3.2017)

[10] nodejs, http://nodejs.org/ Accesed, 10.3.2017

[11] A. Škraba, R. Stojanović, A. Zupan, A. Koložvari, D. Kofjač. Speech-Controlled Cloud-Based Wheelchair Platform for Disabled Persons. Microprocessors and Microsystems, Elsevier, 2015.

[12] A. Škraba A. Koložvari, D. Kofjač, R. Stojanović, Wheelchair maneuvering using leap motion controller and cloud based speech control: Prototype realization. Embedded Computing (MECO) 4th Mediterranean Conference on, Budva, Montenegro. 14-18 June 2015, pp. 391 – 394.

[13] A. Škraba; A. Koložvari; D. Kofjač; R. Stojanović, Prototype of speech controlled cloud based wheelchair platform for disabled persons. Embedded Computing (MECO) 3rd Mediterranean Conference on, Budva, Montenegro. 15-19 June 2014, pp. 162 – 165.

[14] Karagoez, Mehmet Fatih; Turgut, Cevahir, "Design and Implementation of RESTful Wireless Sensor Network Gateways Using Node.js Framework," in European Wireless 2014; 20th European Wireless Conference; Proceedings of , vol., no., pp.1-6, 14-16 May 2014

[15] Carlos, R.; Coyle, S.; Corcoran, B.; Diamond, D.; Tomas, W.; Aaron, M.; Stroiescu, F.; Daly, K., "Web-based sensor streaming wearable for respiratory monitoring applications," in Sensors, 2011 IEEE , vol., no., pp.901-903, 28-31 Oct. 2011 doi: 10.1109/ICSENS.2011.6127168

[16] "Why Intel Loves HTML5," https://software.intel.com/en-us/videos/why-intel-loves-html5 (Accesed, 10.3.2017).