A Testing Method for Vital Sign Monitoring Platform

Shreyas Rao[1], Dr. Shilpa DR[2]

[1]*Student, Dept. of Electronics and Communication Engineering, R.V. College of Engineering,Karnataka, India*

[2]*Associate Professor, Dept. of Electronics and Communication Engineering, R.V. College of Engineering, Karnataka, India*

[1]*shreyasrao.ec17@rvce.edu.in*, [2]*shilpadr@rvce.edu.in*

Abstract: *Chronic conditions including sleep apnea, diabetes, and heart disease necessitate continual monitoring of vital indicators, which is sometimes only available in a hospital. Comprehensive vital sign monitoring in an outpatient setting is now achievable thanks to recent technological advancements.Various monitoring devices are available that measure vital signs such as heart rate, oxygen saturation, respiration rate, blood pressure, and temperature in an outpatient setting with minimal constraints on the patient's typical lifestyle.Testing is an important part of the product development cycle since it detects any faults or defects before the product is delivered to the customer, ensuring product quality. This study presents a way for validating the various functionalities of vital sign monitoring platforms.*

Keywords: Vital Sign Monitoring (VSM), Health Monitoring, Software Testing

1. Introduction

During the past few years, technology has grown to limitless extents and has made human lives easier and sophisticated. The improvements in technology have helped in diagnosing various diseases and also in their treatment. It has been proven by many studies that earlier detection of disease may lead to more cures or longer survival [1]. Earlier there existed periodic screening examinations for detecting specific chronic diseases such as diabetes, cancer and cardiovascular diseases but with the help of current technology, wearable health monitoring devices can be used for such diagnosis. Various wearable health monitors were proposed and developed to combat the limitations of a standalone physiological health monitor.Wearable health monitoring systems are designed to make the necessary measurements to track different body parameters cost-effectively. Different parameters include but not limited to blood pressure, temperature, pulse/heart rate, muscle activity, blood sugar level, oxygen content in the blood (SPO2), brain activity, motion. With such rapid development in the wearable vital sign monitoring sector, it is important to have a robust test infrastructure to validate the functionality of these platform before they are tested clinically. Testing ensures the reliability and performance of the product thus; it is crucial that this step needs to be carried out properly. Testing helps in detecting some bugs or defects that occur in some specific cases known as corner/edge cases, thereby providing the developers with information that will help them deal with such cases. Having well-built test cases increase the overall quality of the product as it ensures that every product is reliable and meets certain standards. Also, during the test phase, it is possible that one can detect a missing functionality, inclusion of which would have a great impact on the user experience. Tests are carried out at different levels to ease the process of testing. A common test infrastructure is proposed in this study that can be used for different vital sign monitoring platform with minimal changes and hence speeding up the process of testing.

2. Overview of vital sign monitoring platform

In order to develop a standard test infrastructure for the vital sign monitoring platform, it is necessary to understand the core working of the platform itself. Wearable type of vital sign monitoring platforms is the most common type of device that is available to a user, popularly these devices are known as fitness trackers and are capable of measuring some crucial vitals such as heart rate, blood oxygen saturation (SPO2) and temperature. These devices often have a display that is used for displaying the vitals measured without the need for any other external computing source to process the sensed data. Several different proposed architectures [2],[3],[4] are analyzed to develop a common platform for which the test infrastructure will be developed. Figure – 1 shows a block diagram of the system that is used as a reference for the development of the test infrastructure.

Figure 1. A Typical Vital Sign Monitoring System

The proposed system consists of five basic blocks namely microcontroller, PPG sensor, ECG sensor, temperature sensor and accelerometer. Depending on the actual sensing mechanism used by these sensors their accuracy and throughput may vary, but the core functionality of sensing a change and converting it into a processable signal remains the same. It is assumed that most of the platforms have some kind of API that allows for it to communicate with an external computer where the test will be written.

2.1 The core – microcontroller

The microcontroller is the core of the entire vital sign monitoring platform as it controls and coordinates all the sensors/modules attached to it. The microcontroller usually has two sets of communication links, one that is used for communicating with the sensors such as SPI or I2C and another for communicating with external devices (smartphone or a desktop computer) like BLE, Zigbee or USB. The microcontroller is also capable of performing a reasonable amount of computation for processing the raw signal from the sensors.

2.2 PPG sensor

The photoplethysmogram (PPG) is a technique for detecting changes in blood volume in the microvascular bed of tissue. A PPG signal is typically obtained by shining a light on a portion of the skin and monitoring the return signal using a photodetector to determine how much light was absorbed. The most common location for measuring PPG is on fingertip or wrist.The measured PPG waveform changes from person to person. By analyzing the PPG waveform several vitals can be measured such as blood oxygen saturation (SPO2), blood pressure and respiration rate. As this sensor is easy to fabricate and inexpensive, it the most common sensor seen in most vital sign monitoring devices.

2.3 ECG sensor

Electrocardiogram (ECG) is thepainless measurement of the electric signals that are produced due to the activity of heart by the use of 2-4 electrodes in contact with the skin of one's body. It is a quick and easy method to monitor the heart's activity and health. Wearable devices usually incorporate a 2-3 pole measurement technique with one of the poles being a reference to measure the changes in skin impedance. The basic principle followed by this sensor is that they apply a known voltage to the skin and measure the return current, the ratio of which determines the skin impedance. Continuous monitoring of ECG signals can be quite helpful as they help in identifying heart related diseases at an early stage. Another use of such an arrangement of electrodes is that they can be used to check if the monitoring platform is in contact with the skin.

2.4 Temperature sensor

Another vital that can be easily measured with the help of an inexpensive sensor is body temperature, but most of the vital sign monitoring system skip this sensor. A simple thermistor can be used to obtain body temperature with reasonable accuracy. A change in body temperature is the early indicator of some diseases like influenza and fever, hence having an option to measure body temperature continuously help in identifying these diseases.

2.5 Accelerometer

An accelerometer is a device capable of measuring the rate of change of velocity (acceleration) of a body in its rest frame. An accelerometer is used for a different purpose in a vital sign monitoring platform as they do not necessarily measure any vital parameters but can help in improving the accuracy of other measurements. One of the applications is to keep track of the number of steps i.e., an accelerometer along with some suitable algorithm can function as a pedometer in fitness trackers. The accelerometer can also provide correction values to PPG and ECG sensor based on the current state of motion of the device.

3.Proposed test infrastructure

Based on the reference vital sign monitoring platform proposed in the previous section a suitable test infrastructure is developed for validating the basic functionalities of the platform. There are various levels of testing [5] such as unit testing (Performed on individual blocks or functions), Integration testing (Used for checking communication between blocks) and System testing (Used to validate the functionality of the system as a whole). For this study, system-level testing is considered as the combined functionalities of all the sensors together needs to be tested. A model based on grey box testing, which has advantages of both black box and white box testing is proposed in this section. The test infrastructure as shown in Figure – 2 has four main components namely, test engine, test files, report generator and pass/fail handler. This test infrastructure built assuming that the vital sign monitoring platform is connected to a desktop computer via USB.

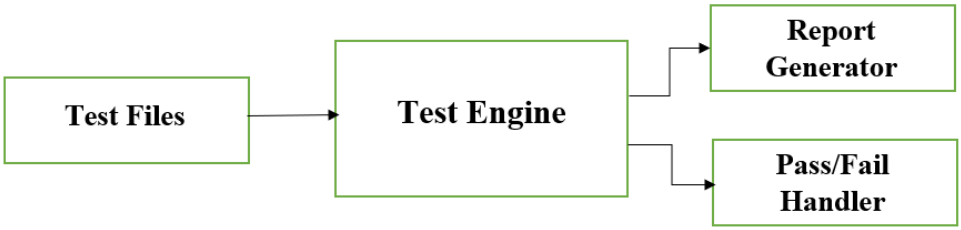


Figure 2. Testing Architecture

3.1 Test engine

The test engine is the heart of the test infrastructure as coordinates the operation of all other blocks. Test engine performs three main tasks namely, initialization of connection, execution of test cases and termination of the connection. Initialization of connection starts with the opening of a specific COM to which the platform is attached and locking the COM port to the test program. Once the COM port becomes accessible, all the test cases in different test files are executed in a sequential manner. After all the tests are executed, a brief cleanup is performed and the COM port is released.

3.2 Test files

Test files contain the various test cases for different scenarios sorted by the application and sensor they belong to. A total of 45 different test cases are developed to test different functionalities of the proposed hardware as per Figure - 1. A sample of what an individual test case looks like is shown in Figure -3.

```
testcase_1(COMPORT object)
{
    datatype testdata;
    datatype returndata;

    returndata = function_under_test(testdata);
    check_if_returndata_is correct(returndata);

    test_result(test_name, exec_status, err_str);
}
```

Figure – 3. A Sample Test Case

Each case takes an input that is the COMPORT object, which in turn allows the test case to sends out commands that can be used for interacting with the hardware. Initially, each test case starts by listing a set of data that will be required of the testing process such as a specific register value that needs to be set for a particular operation or the name of some file that will be accessed during some part of the test. After the initialization of necessary data, the actual tests are run and their output is compared with the expected value to determine if the test is a pass or a failure, which is handled by the pass/fail handler. At end of each test, the test_result () function is called which internally handles the generation of the report.

3.3 Pass/fail handler and report generator

Pass/fail handler, as the name suggests, it is responsible for handling the status of a test i.e., if a particular test case has passed or failed. This block is tightly integrated with the report generation block, as it eases the process of determining the cause of failure.Once all the tests are completed, it always good to have a detailed report to understand what all tests were run and how long each test took. This functionality is implemented by the report generator. Two types of report are generated by this test infrastructure namely the JUnit XML report and Robot XML report. These are two standard report formats that are widely used and hence allows the test infrastructure to be integrated with an automation tool like Jenkins. JUnit report offers a brief of all the tests conducted and their status. The limitationsof this test report are thatit is difficult to traceback the error as there are no informativestatements in the report except for the cause of failure. A sample Junit report is shown in Figure – 4.

```
<?xml version="1.0" encoding="UTF-8"?>
<testsuites failures="5" tests="10">
  <testsuite name="test_suite1">
    <testcase name="test_case1" class="test_suite1">
      <system-out>Test PASSED</system-out>
    </testcase>
     <testcase name="test_case2" class="test_suite1">
      <system-out>Test FAILED</system-out>
      <failure message="Test failed due to reason">failure description</failure>
    </testcase>
   </testsuite>
</testsuites>
```

Figure 4. A Sample JUnit Report

On the other hand, Robot reports are much more informative and contain logs of every step of the test, hence making it easier to identify the cause of failure. Hence, this test infrastructure is designed to generate Robot XML report by default but also has a provision to generate JUnit XML report.

4. Results

A total of 45 different test cases were written to test the functionality of the proposed hardware and 8 of those 45 tests failed indicating either a bug in the APIs used for communicating with the hardware or a defect in the hardware itself. It is always good to have failures in testing, as it ensures that the test has been run on some corner cases previously not tested by the developer. Some of the tests are listed in Table -1.

Table – 1. Few Test Cases

Sl No	Test Name	Test Description
1	Get_data_PPG	Start the PPG sensor and validate the data obtained from the sensor
2	Get_data_ECG	Start the ECG sensor and validate the data obtained from the sensor
3	Use case 1: PPG + Accelerometer	Check if valid data can be simultaneously obtained from PPG and accelerometer
4	Use case 2: PPG + ECG	Check if valid data can be obtained from PPG and ECG sensors
5	Use case 3: Temperature + ECG	Check if valid data can be obtained from temperature and PPG sensor together

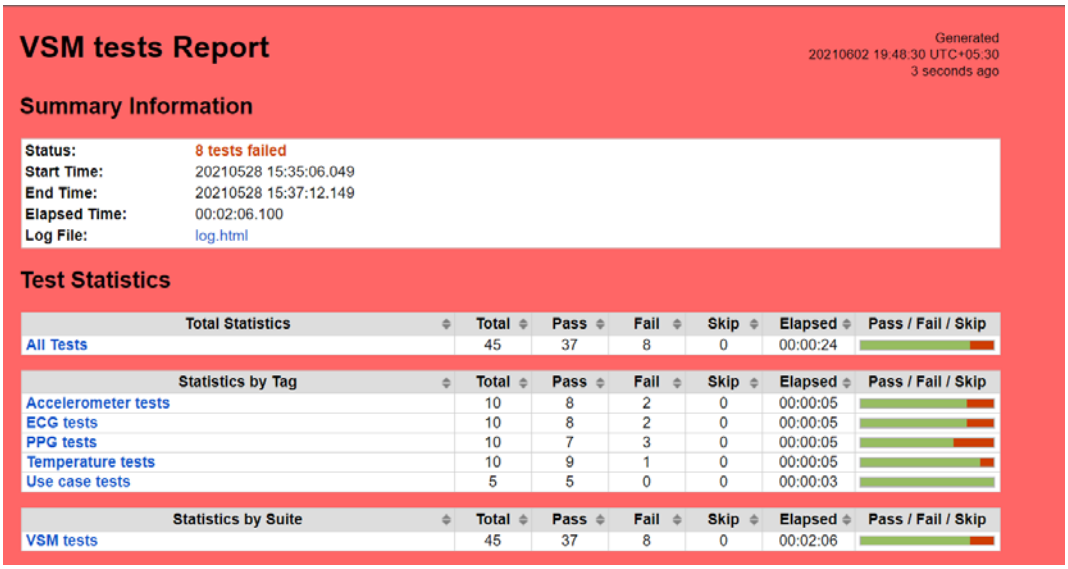
Some use case tests can be observed in Table -1, these are tailored from some specific application that requires data from multiple data simultaneously for better accuracy such as in measurement of blood pressure, some algorithms make use of both PPG and ECG data. Therefore, it is essential that these two data needs to properly synchronized. As every test is executed a brief description and test result is displayed on the command window as shown in Figure – 5. Once all the tests are executed a test report is shown in the command window as depicted in Figure – 6. Along with the command line output, a detailed report along with a log file is generated in Robot format. The Robot report is generated as an XML file which upon parsing with a tool provides HTML report that can be viewed in any browser. A snapshot of the robot HTML report is shown in Figure – 7.

```
==========================
VSM Testing
==========================
Initailzing test...
Connect to COM port - SUCCESS
Start time: 28/05/2021 15:35:06.049
----------------------------------
Test case 1: Get_data_PPG
Start sensor_PPG - SUCCESS
Valid data obtained from sensor? - SUCCESS
Stop sensor_PPG - SUCCESS
Test result - PASS
----------------------------------
```

Figure 5. Output of a Test Case

```
End time: 28/05/2021 15:37:12.149
Time elapsed: 00:02:06.100
Total Tests: 45
Number of Failures: 8
Success rate: 82.23%
```

Figure 6. Test Result



VSM tests Report

Generated
20210602 19:48:30 UTC+05:30
3 seconds ago

Summary Information

Status:	8 tests failed
Start Time:	20210528 15:35:06.049
End Time:	20210528 15:37:12.149
Elapsed Time:	00:02:06.100
Log File:	log.html

Test Statistics

Total Statistics	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests	45	37	8	0	00:00:24	

Statistics by Tag	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
Accelerometer tests	10	8	2	0	00:00:05	
ECG tests	10	8	2	0	00:00:05	
PPG tests	10	7	3	0	00:00:05	
Temperature tests	10	9	1	0	00:00:05	
Use case tests	5	5	0	0	00:00:03	

Statistics by Suite	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
VSM tests	45	37	8	0	00:02:06	

Figure7. Robot Test Report

5. Conclusion

Testing is a very important part of the product development cycle, sometimes taking as much as 40% of the product cycle time. Hence, it is important to have a robust testing infrastructure for easy and quick development of different test cases. After analyzing various proposed models for vital sign monitoring platforms, a generalized model was developed to aid in the development of the test infrastructure. Each block of the test infrastructure was built separately and integrated together to validate hardware similar to the generalized model. Some of the test cases failed, which is a good thing in testing as it shows that the test infrastructure is capable of testing different corner cases.

6. References

[1] Lee S, Huang H, Zelen M. “Early detection of disease and scheduling of screening examinations”, Stat Methods Med vol. 6, December**(2004),**pp. 443-56.

[2] D. G. Guo et al., "A Long-term Wearable Vital Signs Monitoring System using BSN",Proceedings of 11th EUROMICRO Conference on Digital System Design Architectures, Methods and Tools, **(2008),** pp. 825-830.

[3] M. A. A. E. Ghany, M. S. Saleab, R. M. Toma and K. Hofmann, "Efficient wearable real-time vital signs monitoring system",Proceedings of IEEE International Conference on Electronics, Circuits, and Systems (ICECS), **(2015)**, pp. 217-220.

[4] M. A. Al-Shaher and N. J. Al-Khafaji, “E-healthcare system to monitor vital signs”, Proceedings of International Conference on Electronics, Computers and Artificial Intelligence (ECAI), **(2017)**, pp. 1–5.

[5] A. Sawant, P. Bari, and P. Chawan, “Software testing techniques and strategies”, International Journal of Engineering Research and Applications (IJERA), vol.2,June**(2012)**, pp. 980–986.

[6] D. Bhatt, “A survey of effective and efficient software testing technique and analysis”, Iconic Research and Engineering Journals, vol. 1, no. 1, **(2017),** pp. 1–4.

[7] A. Orsa and G. Rothermel, “Software testing: A research travelogue (2000–2014)”, Future of software engineering, vol. 4, **(2014),** pp. 117–132.

[8] Y. Bobade and R. M. Walli “A Review of Wearable Health Monitoring Systems”, International Journal of Advanced Research in Electrical,Electronics and Instrumentation Engineering, vol. 4, no. 10, October **(2015),** pp. 8386-8390.