MAXIM WELLNESS APP USER GUIDE & WELLNESS LIBRARY INTEGRATION GUIDE

Version 4.0.0

April 15, 2021

maxim integrated™

CONTENTS

TABLES

FIGURES

1 Introduction

The purpose of the Wellness Suite application is to provide a single Android GUI to evaluate all algorithms in Maxim portfolio. Algorithms consuming outputs of the ones running on the MRD103 modules will be executed on the Android platform. From the user perspective, the Wellness Suite is the interface to all Maxim algorithms.

This user guide covers these algorithms: WHRM, SpO2, HRV, Respiration Rate, Stress, Sleep and Sports Coaching. The first two of them runs by the MRD103, the rest runs by the mobile application. In addition to these algorithms, the application provides auxiliary functions such as Flash Log Parser and HR Alignment.

NOTE: The application has not supported the Android 11 yet. The minimum supported Android version is 5, the targeted Android version is Android 10.

2 Wellness Suite Application

2.1 System Architecture

The mobile application will communicate with the MAXREFDES103/4/5 via BLE and feed the bio-sensing algorithms included in the application. The complete architecture can be viewed in Figure 1.

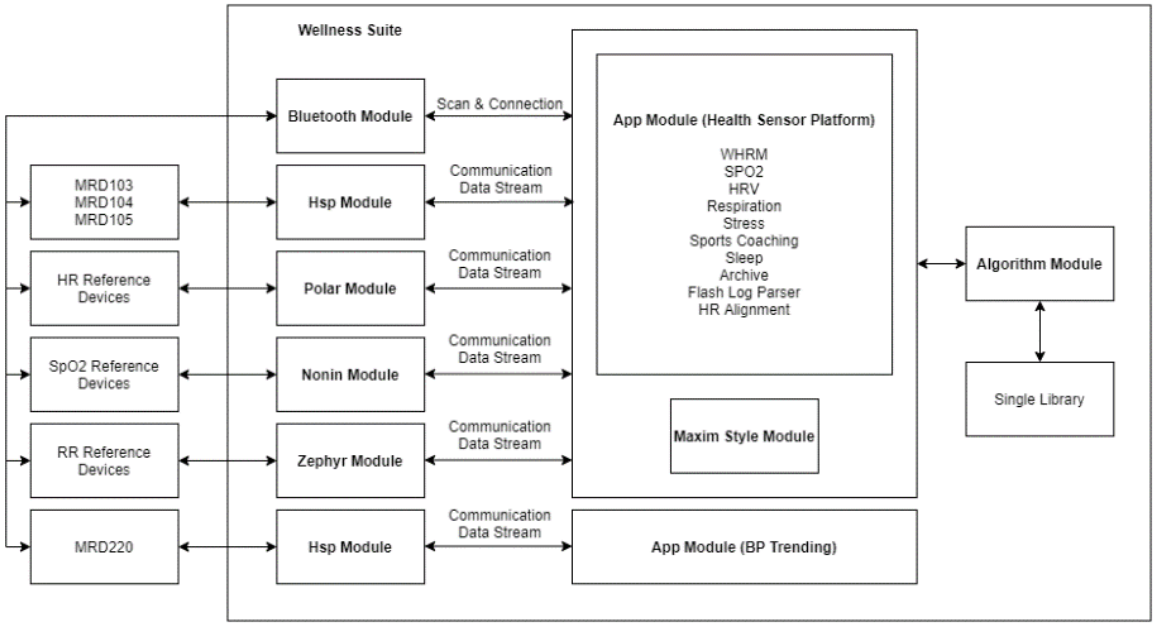

Figure 1: System Architecture

2.2 Application Details

Wellness Suite application runs on Android operating systems. The minimum supported android version for this application is Android 5.0. Besides, the mobile phone running the application should have at least BLE version 4.2.

The app flow diagram can be seen in Figure 2.

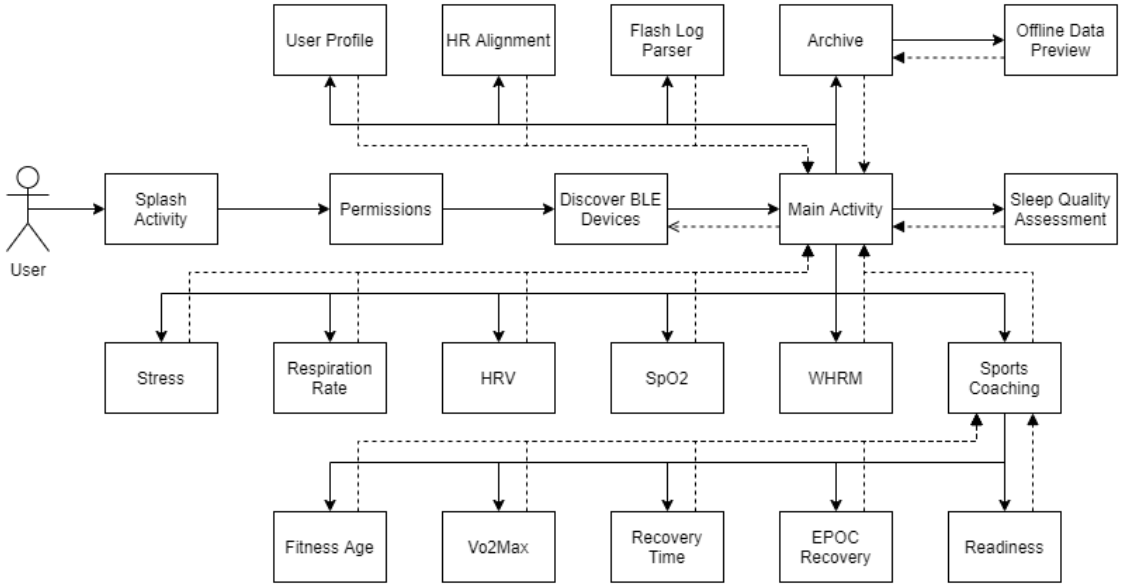

Figure 2: Android App Flow Diagram

2.2.1 Permissions

After opening the application for the first time, the application asks the user to grant the necessary permissions to run as it is supposed to. The application requires the below permissions:

- Location Access
- Storage Write
- Storage Read
- Enabling Bluetooth

The location access permission is needed for the application to discover the nearby Bluetooth Devices. The Storage Write/Read permissions are needed to log the sensor data and read them back. The user should also enable the Bluetooth to discover devices and communicate with them.

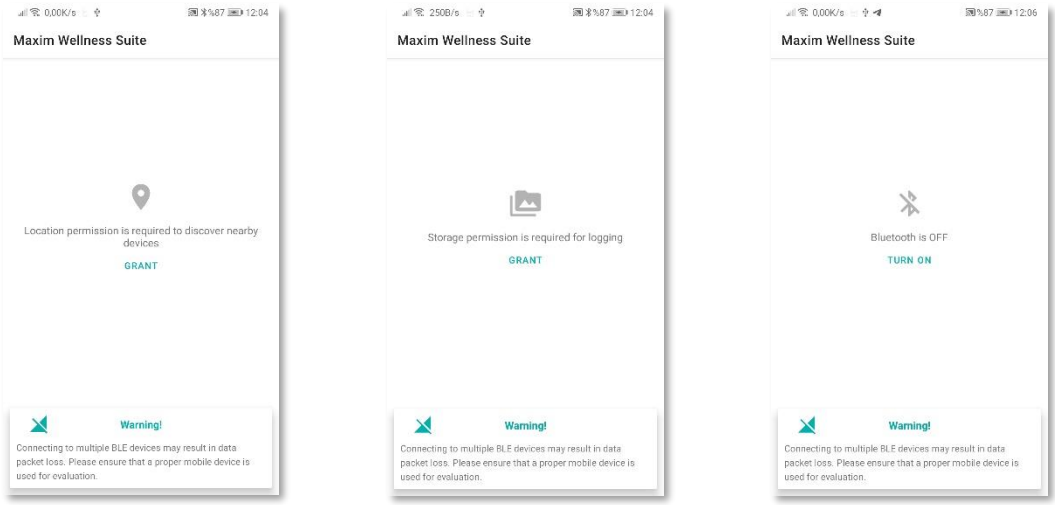

Figure 3:Permissions

2.2.2 Discover BLE Devices

After granting permissions, the available Maxim devices are listed automatically. The user can start the scanning and stop it by pressing the button located in upper-right position.

Every item in the list shows the device name, device MAC address and its signal strength.

NOTE: For some mobile phones, the GPS should be open to discover nearby BLE devices. If no device is listed in the app, please open the GPS to see the available devices.

Figure 4: Available Devices

When the user selects the appropriate device by pressing on it, the application shows the Landing Page (Main Activity) to the user.

2.2.3 Landing Page

When the user arrives to this page, the application is already connected to the selected device and starts communicating. It receives the device information including hub version. On the upper-left corner, Bluetooth icon

can be seen. It indicates the connection status of the selected device. On the lower-left corner, the app version is shown. On the lower-right corner, the device name and the hub version are shown.

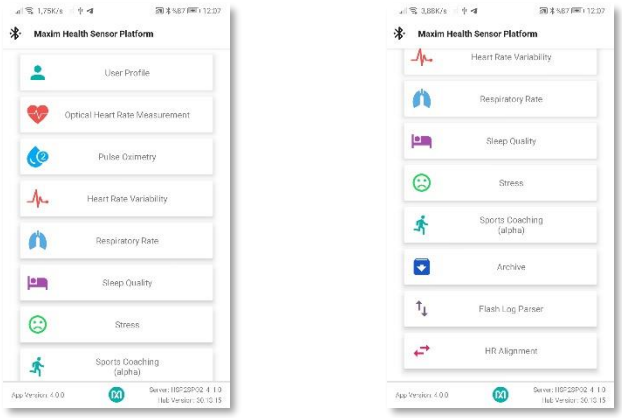

Figure 5: Landing Page

This page contains buttons directing user to the related pages. For example, when the User Profile button is pressed, User Profile page opens.

2.2.4 User Profile

The algorithms that are developed for both the watch itself and Android require a few user information to give more concise result. These user data can be cited as birth year, weight, height, initial heart rate and the gender. In addition to this information, the application also needs the username to maintain different user profiles and allow users to select their own personal information dataset.

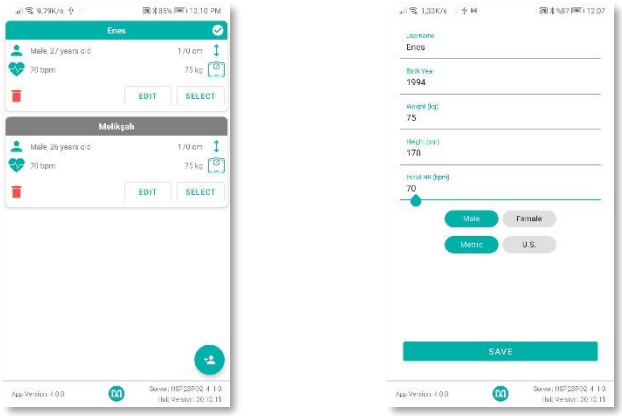

Figure 6: User Profile Page

In the first page of the User Profile, all the available users are listed. The selected user is differentiated from others by its green header and the check mark. The selected user profile is used for both the device algorithms and the android algorithms. If none of the available users is selected, the default user profile is going to be used for them.

With the floating action button located at bottom right of the page, user can create a new user profile.

NOTE: This collected data is used to improve the accuracy of Maxim algorithms that run on this device. It is not shared with any 3rd parties.

2.2.5 Optical HRM

Heart rate is the frequency of heart contractions that is measured in beats per minute (BPM). Maxim Wearable Heart Rate Monitoring (WHRM) algorithm utilizes synchronized 3-axis accelerometer and PPG data to provide several end-user friendly fitness tracking outputs like heart rate and activity related statistics e.g. step count, burned calories and activity class of the user. The PPG (photoplethysmography) signal can be gathered from contractions of ventricles in finger, wrist, ear, toe, chest etc.

Maxim WHRM solution includes HR and activity tracking algorithms, especially for the heart rate detection, cardiac beat detection, step measurement and activity classification. The envisioned product targets health and wellness applications for continuous 24/7 monitoring with optimized power management and the provision of accurate HR and IBI detection, step detection and sample by sample activity classification.

As seen in the Figure 7, this page is responsible to show the WHRM Outputs such as Heart Rate (HR) and activity related outputs. The application also shows the PPG data on the chart.

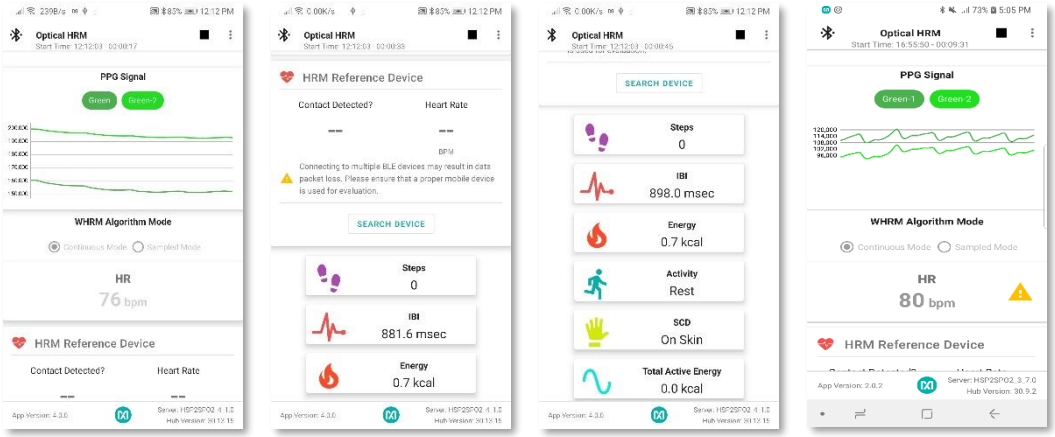

Figure 7: Optical HRM

From the upper right corner, the user can enable/disable the features such as “Log to File”, “Log to Flash”, “Enable SCD”, “Enable MFIO”, “Enable SCD SM”, “Low Power Mode”. Also, the user can open the settings.

- Log to File: If it is enabled, the received sensor data is recorded and saved to a file under “MaximSensorsApp/RAW” folder. The app also logs a 1 Hz HR data and saved to a file under “MaximSensorsApp/1Hz” folder.
- Log to Flash: If it is checked, the device stores the data in its flash, and it does not send the data to the app.
- Enable SCD: If it is checked, the algorithm detects if the watch is worn.
- Enable MFIO: If it is checked, the app enables reporting of events from SensorHub via interrupt.
- Enable SCD SM: If it is checked, Skin Contact Detection State Machine is enabled.
- Low Power Mode: If it is checked, the data stream rate becomes 1Hz instead of 25Hz. Internally, we are generating 25Hz data from 1Hz data to support logging and algorithm functionality.

From the settings dialog (Figure 8), the user can change the following:

- Sample mode cycle time
- HR Confidence Level
- HR Expiration Threshold

The algorithm has 2 modes: Continuous mode and Sampled mode. In the continuous mode, the data streaming does not stop until the user stops it. In the sampled mode, the algorithm stops when the calculation finished, and restarts again after certain time. This time is determined from the Settings Dialog as seen in Figure 8. According to HR Confidence Level and HR Expiration Threshold settings, If the low confidence condition is met, the app shows a warning to the user as seen in Figure 7.

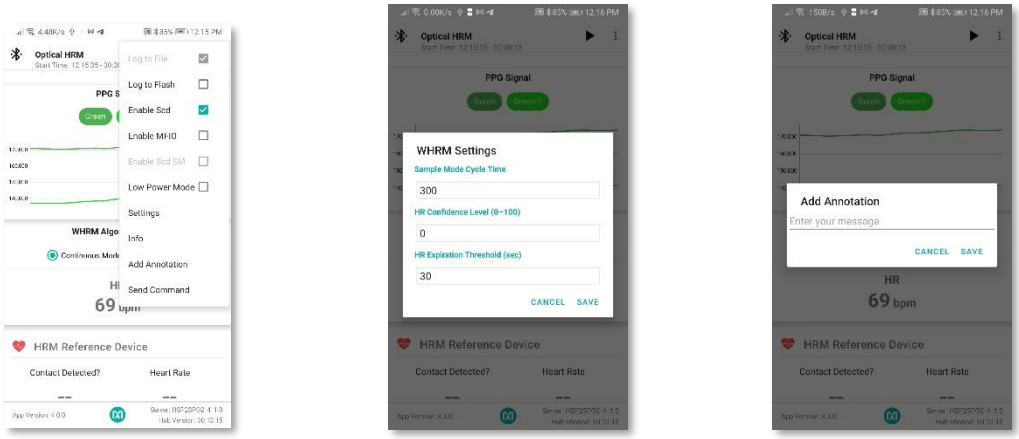

Figure 8: Optical HRM Menu

The user starts and stops monitoring by pressing the play and stop button, respectively. Also, he/she can add an annotation while measuring. It will be saved to “MaximSensorsApp/ANNOTATIONS” folder.

While receiving data from the device, if packet loss occurs, the app records an error log file under “MaximSensorsApp/ERROR” together with a warning message.

This page also allows user to connect reference HR device to compare its results with Maxim’s results. The received data is recorded and saved to a file under “MaximSensorsApp/HR_REF” folder. After stopping the measurement, the app tries to align the Maxim HR results with Reference HR data. If the alignment is done successfully, the app saved the aligned HR Results to a file under “MaximSensorsApp/ALIGNED” folder.

Note: The Annotation feature and Error log feature are available for all the measurements.

2.2.6 Pulse Oximetry

Oxygen plays vital role for every cell in human body to produce energy and all organs require oxygen for metabolism. Pulse oximetry is an essential technique used for many decades to monitor the amount of oxygen carried in blood. Maxim has developed high-sensitive finger-based optical sensors with optimized algorithm to make the pulse oximeters portable and easily applicable.

Pulse oximeter is a lightweight, small and noninvasive optical devices to monitor the amount of oxygen carried in the blood. Pulse oximeter devices measure oxygen saturation by illuminating the skin with red and near-infrared (IR) lights and measure changes in light absorption of oxygenated (oxy-hemoglobin, HbO2) and deoxygenated blood (deoxy-hemoglobin, RHb)) using two light wavelengths.

As seen in the Figure 9, this page is responsible to show the SpO2 value together with Heart Rate. The application also shows the PPG data on the chart.

The algorithm has 2 modes: Continuous mode and One-shot mode. In the continuous mode, the data streaming doesn't stop until the user stops it. In the one-shot mode, the algorithm stops when the calculation finished.

The user starts and stops monitoring by pressing the play and stop button respectively.

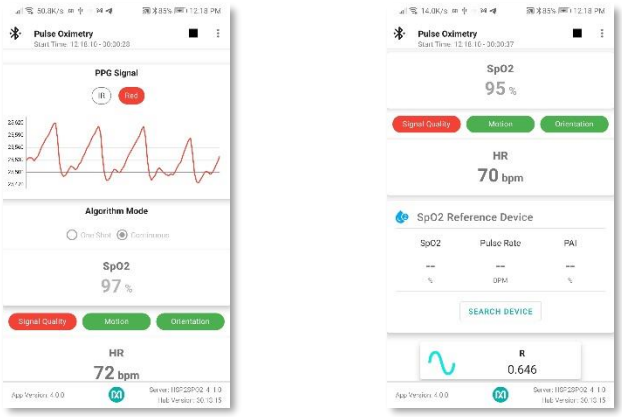

Figure 9: Pulse Oximetry

The SpO2 calculation requires "no movements and talk". When the user measures his/her SpO2 rate, he/she needs to stay still and should not talk. The application gives visual indication for the parameters that affect the measurement such as Signal Quality, Motion, and Orientation. If these indicators are red, it means there is a problem in the corresponding parameter.

This page also allows user to connect reference SpO2 device to compare its results with Maxim's results. The received data is recorded and saved to a file under "MaximSensorsApp/SPO2_REF" folder.

NOTE: The user should adjust the wristwatch on the wrist such that Red PPG peaks and troughs are clearly visible on the chart. After every valid SpO2 value, the color text that represents the SpO2 value is changed to black, and then it fades out until a new value is received. If a new SpO2 measurement is not calculated in 10 seconds after the last measurement, the value becomes obsolete and app shows "No Report" instead of old value.

2.2.7 Heart Rate Variability

Heart Rate Variability (HRV) provides a measure of the time difference between successive heartbeats; the variation is controlled by the autonomic nervous system (ANS). This is the system that regulates our heartbeat, blood pressure, and breathing. Our ANS consists of two main components:

- Our "fight-or-flight mechanism" (the sympathetic nervous system) is our response to external stress factors. This is what makes us energized and ready to face challenges.
- Our relaxation response (the parasympathetic nervous system) is the opposite. This is what turns us toward a rest and recovery mode.

HRV tends to be low when we are in a fight-or-flight mode and high when we are more relaxed. High HRV is generally associated with greater cardiovascular fitness. The electrocardiogram test is the gold standard for

assessing HRV. However, wearable devices with sophisticated algorithms are now making this data more accessible to more people. Since HRV provides insights into our ANS, which regulates our heart rate (HR) and blood pressure, it can be considered as the most important general wellness parameter.

HRV gives us some clues about potential cardiac conditions. Over time, higher HRV can indicate increased resilience, while lower HRV can point to chronic stress. HRV can be increased by changes in behavior: quitting smoking, losing weight if needed, exercising, and managing stress, for example. HRV analysis techniques that fall into three categories provide useful health insights:

- **Time-domain** analysis based on standard deviation and root mean square (RMS) of normal-to-normal intervals provides powerful differentiators of stressed versus relaxed conditions. Increased HRV is an indicator of fitness.
- **Frequency domain** analysis shows the ratio of parasympathetic and sympathetic activity.
    - High frequency pertains to the parasympathetic system and the vagus nerve, which controls the parasympathetic nervous system
    - Low frequency pertains to sympathetic activity
    - Very low frequency pertains to the sympathetic nervous system, chemoreceptors, thermoreceptors, and the renin-angiotensin system (i.e., hormones)
- **Non-linear** analysis can point to underlying cardiac conditions and includes:
    - Detrended fluctuation analysis (DFA), which looks for self-similar patterns by analyzing the power spectral density (PSD). Peaks in PSD indicate repetitive patterns.
    - Entropy analysis, a measure of randomness over time. Decreased HRV and increased randomness of HR are independently associated with high-risk conditions.
    - Poincare plot analysis utilizes scatter plots of consecutive pulse interval points. Consecutive pulses that vary by large amounts will have larger scattering around the diagonal. Low HRV will shrink and cluster around the diagonal. Unbalanced HR behaviors such as fast acceleration and slow deceleration will generate asymmetric plots. Large off-diagonals show skipped heartbeats, which usually indicate an arrhythmia problem.

As seen in the Figure 10, this page is responsible to show the HRV algorithm results together with IBI values. The application receives the data stream and passes them to the algorithm library. Then, it displays the algorithm results.

The meanings of the HRV metrics are listed below:

- **Time Domain Metrics**
    - **AVNN:** Average of normal to normal (NN) intervals in ms
    - **SDNN:** Standard deviation of NN intervals in ms
    - **RMSSD:** RMS value of successive differences in ms
    - **PNN50:** Percentage of successive differences greater than 50 ms
- **Frequency Domain Metrics**
    - **ULF:** Power in Ultra Low Frequency band in ms^2
    - **VLF:** Power in Very Low Frequency band in ms^2
    - **LF:** Power in Low Frequency band in ms^2
    - **HF:** Power in High Frequency band in ms^2
    - **LF/HF:** LF/HF ratio
    - **TOTPWR:** Total power in ms^2

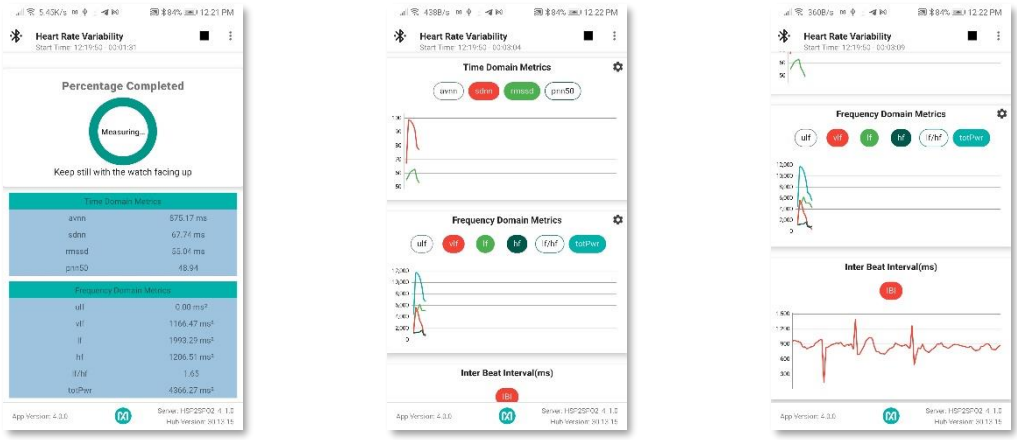

Figure 10: Heart Rate Variability

2.2.8 Respiratory Rate

Respiration rate is a vital sign indicating health and wellness and can be an input for several applications such as sleep quality assessment, stress level estimation and energy exposure estimation etc. The available methods to measure respiration rate base on capnography, ECG and PPG. Although capnography has the most precise results, it needs oral/nasal cannula which makes impossible to use it under movement. ECG based solutions need multiple probs at different locations on human body, therefore they are also not convenient for a watch-based design. Maxim's solution is based on PPG and compatible to use on wrist, finger and forehead that make it best solution for wearable devices.

Using PPG, we can measure respiration induced features. One of these features is Respiratory Sinus Arrhythmia (RSA) which causes heart rate change during respiratory events. Another feature we can measure is that the effect on the venous volume of intrathoracic pressure changes during respiratory cycle. After processing these features, our solution can report very accurate respiration rate estimates.

As seen in the Figure 11, this page is responsible to show the Respiration Rate Algorithm result. The application also shows the Respiration Rate on the chart. The application receives the data stream and passes them to the algorithm library. Then, it displays the algorithm results.

This page also allows user to connect reference RR device to compare its results with Maxim's results. The received data is recorded and saved to a file under "MaximSensorsApp/RR _REF" folder.

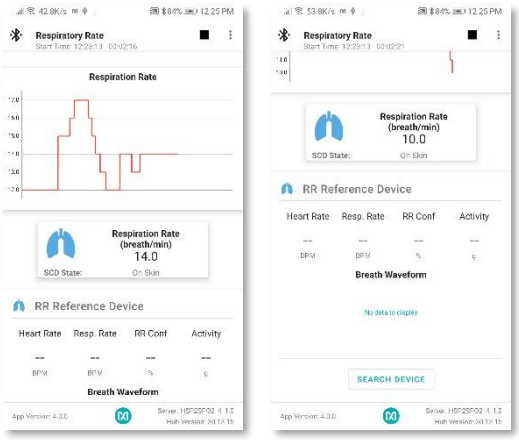

Figure 11: Respiratory Rate

2.2.9 Stress

Daily life stress is one of the important issues of modern life. Mainly, two kinds of stress phenomenon exist as acute and chronic stress. American Psychological Association noted that the causes of acute stress are pressure from recent past and near future. In example, exercise challenges or any kind of sudden anxiety can induce acute stress. On the other hand, chronic stress results from the long-term pressures like socioeconomic conditions, ongoing problems in relations etc. The scope of this study is focused on the detection of acute stress.

Stress releases a hormone called adrenaline that quickens breath increases HR and Blood Pressure, known as "fight or flight response". Deep breathing exercise therefore can be suggested to shift body and mind into more relaxed state.

As there is no single gold standard for measuring acute mental stress, several measurement methods were employed in related works like: psychological tests, blood tests or salivary cortisol level tests. In literature, most of the wearable stress monitoring studies use galvanic skin response, muscle tension, blood pressure and skin temperature devices as well as HR monitoring devices. In this study, Maxim's stress estimation methods use accelerometer and PPG data (HR, HRV) solely.

Quick Stress Assessment for acute stress is in the scope of this solution where user provides a considerably small amount of data (using wearable device) in a resting state and receives his stress level information.

As seen in the Figure 12, this page is responsible to show Stress Algorithm result together with the HR value. The application receives the data stream and passes them to the algorithm library. Then, it displays the algorithm results.

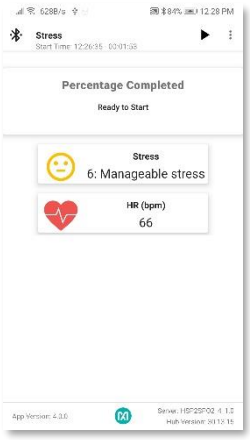

Figure 12: Stress

The stress score is in the range of 1 to 18.

Score	Meaning	Color of the Emoji	Status of the Emoji
1-2	Overwhelmed	RED	☹
3-5	Frustrated	ORANGE	☹
6-7	Manageable Stress	ORANGE	😐
8-9	Feeling Okay	GREEN	😐
10-11	Doing Great	GREEN	😊
12-14	Very Relaxed	ORANGE	😊
15-18	Uncomfortably Relaxed	RED	😊

Table 1: Stress Score

2.2.10 Archive

The archive page lists all the log files. It reads the file names and parses them to show to the user in a more user-friendly way.

Figure 13: Archive

When the user presses to list item, the application reads the recorded sensor data and passes them to the algorithm library. Every list item contains the below buttons:

: It deletes the related file.

: It allows user to share the file.

: It runs the sleep algorithm on the related file. After calculation, the app warns the user whether the algorithm finds sleep data in the log file or not. If it finds, the app saves the sleep algorithm result under "MaximSleepQA" folder.

As seen in Figure 14, the user can select the graph from the Spinner to visualize it. The spinner contains: HR, SpO2, IBI, Steps, Motion, Resp. Rate, HRV metrics and Stress Score. If the selected graph is the result of an algorithm running in the app, the Export CSV button is going to be available for the user to export the algorithm result as csv.

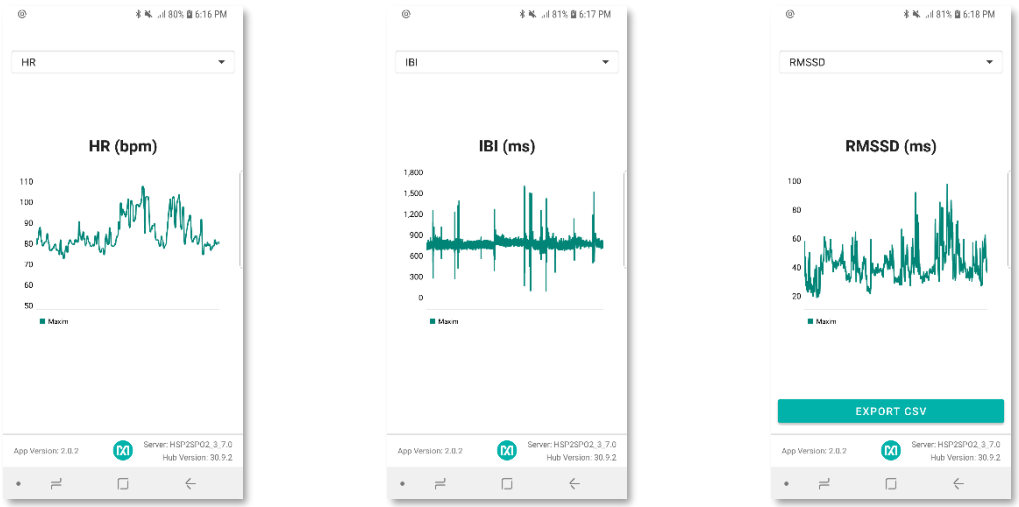

Figure 14: Archive – Preview Screen

2.2.11 Sleep Quality

Sleep is a crucial element for human's mental and physical. It is a highly active state of the body where memory formation and restoration, removal of the unimportant connections and damages during wake state such as oxidative stress, death of neurons in the hippocampus etc. are held. As a result, a healthy sleep is one of the key elements to able to use the cognitive skills. It is stated that not only lack of sleep cause severe losses in cognitive abilities, but also cause cardiovascular diseases.

Sleep-Wake cycle is dictated by the hippocampus through the circadian rhythm. This process stimulates the sleep-promoting and arousal centers in the brain through neurochemical processes. Sleep is characterized by the cyclic occurrences of two distinct states known as rapid eye movement (REM) and non-rapid eye movement (NREM). Furthermore, the NREM consist of slow wave sleep so called Deep Sleep, and lighter sleep stages so called Light Sleep. During the REM dreams occur and activity of the brain excites like the wake state as of other biological traits. As the name suggests, during Deep Sleep brain reflects slower.

The gold standard for the tracking of known as polysomnography (PSG). PSG generally comprises a human annotator, classifying the non-overlapping epochs of 30s to a sleep stage via analyzing electroencephalogram (EEG), electrocardiogram (ECG), electromyogram, electrooculogram, oxygen saturation, respiratory effort, and movement signals. However, not only PSG requires expensive laboratory facilities but also disrupts the sleep of the subjects due to bulky devices. Maxim Integrated has developed an affordable and non-disruptive sleep tracking solution for wrist worn devices.

The Sleep Quality Assessment application is integrated into Wellness Suite App v3.0.0. The app looks under the "MaximSleepQA" folder to visualize the results.

In the main page of the Sleep Fragment, the bar chart lists the sleep time date by date. Below the bar chart, there are pie charts showing the stages of the sleep data for the day in which the measurement is taken.

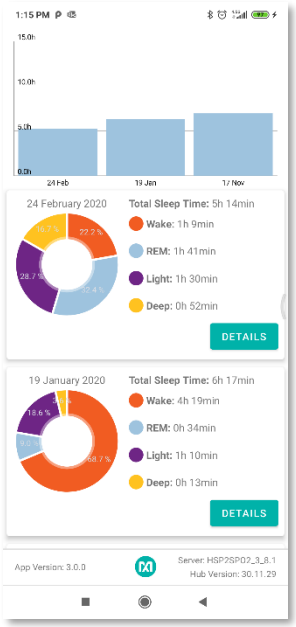

Figure 15: Sleep Quality Assessment

The “Details” button takes the user to a more detailed page about the corresponding sleep data. At that page, the user can see his/her sleep score, sleep stage durations, movements, Heart Rate and IBI values.

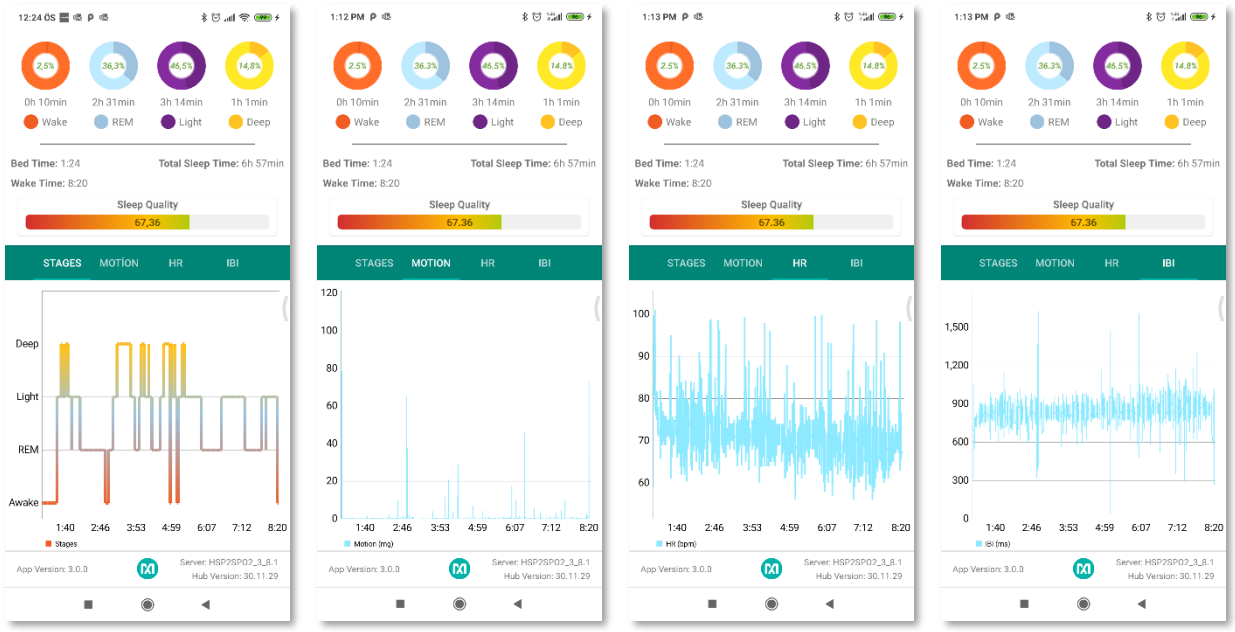

Figure 16: Sleep Quality Assessment-Detailed Page

2.2.12 Sports Coaching

To achieve a healthy life, sport needs to be a part of the individuals lives. People need to plan and keep track of their sports performance and frequency to adjust their lifestyle, so that they can keep their healthy state. With the widespread of wearable devices, people started to use them for keeping track of their vital sign (heart rate, SpO2, etc.) and sports activities (walking, running, biking, etc.). As a next step, people are requesting to keep track of their performance and effort using their wearable devices. Maxim developed bunch of the algorithms to assess users’ performance, effort, readiness during sport activities and bundled them in the Maxim Sports Coaching Algorithms product. Users can enhance their performance and decrease sports related injuries using Maxim Sports Coaching Algorithms.

2.2.12.1 Landing Page

Maxim sports coaching algorithms require the users’ personal information such as birth year, gender, weight, and height for the measurements. If no user is selected in the Main Landing Page, the app does not allow user to continue to Sports Coaching session.

If the user already selected the profile, the app navigates the user to the Sports Coaching Landing page. From this page user can do the followings:

- Training: User can visit all the sports coaching measurements pages.
- History: User can see his/her measurement records.
- Profile: User can change his/her profile.

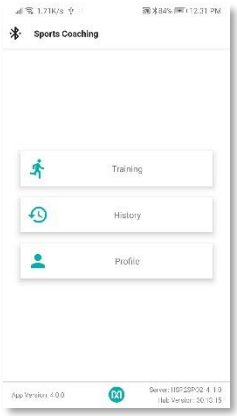

Figure 17: Sports Coaching Landing Page

2.2.12.2 Training

This page lists all the measurements that the Maxim Sports Coaching Algorithm supports. As seen in Figure 18, the user can measure his/her Readiness, Resting VO2Max, EPOC Recovery, Recovery Time, and Fitness Age.

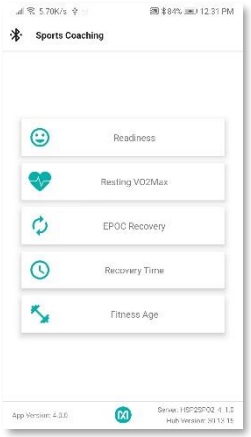

Figure 18: Sports Coaching Training page

2.2.12.3 Readiness

This function is used to estimate how much ready is the user for the next training. It needs 3-6 minutes of data collected from the user in resting state. User can start measuring by pressing the play button. While measuring, the completion percentage is reported in the circular progress bar, and the heart rate is displayed in real-time. When the measurement finishes, the user can see his readiness score together with HR statistics as seen in Figure 19.

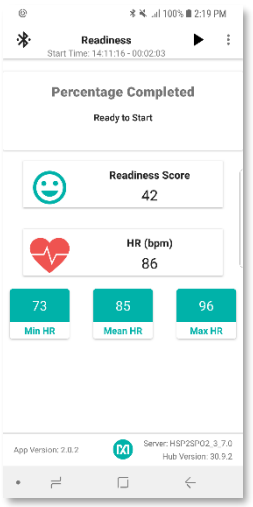

Figure 19: Readiness

2.2.12.4 Resting VO2Max

This function is used to calculate VO2_MAX of the user. This function needs 3-6 minutes of data collected from the user while they are sitting in a relax state. Relax state is where user is fully recovered from last training and not effected by any kind of food, drug, or alcohol. User can start measuring by pressing the play button. While measuring, the completion percentage is reported in the circular progress bar, and the heart rate is displayed in real-time. When the measurement finishes, the user can see his VO2Max score together with HR statistics as seen in Figure 20.

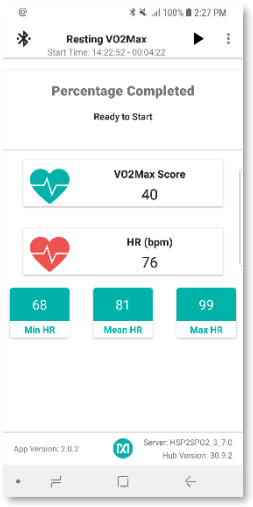

Figure 20: Resting VO2Max

2.2.12.5 EPOC Recovery

EPOC is an abbreviation for Excess Post-Exercise Oxygen Consumption. To estimate the user's EPOC value, user needs to start the algorithm after training and collect data for 5-6 minutes in resting state. User needs to select the

appropriate parameters before starting the measurement. This functionality is dependent on the user information such as exercise duration in minutes, exercise intensity and the time passed since the end of exercise.

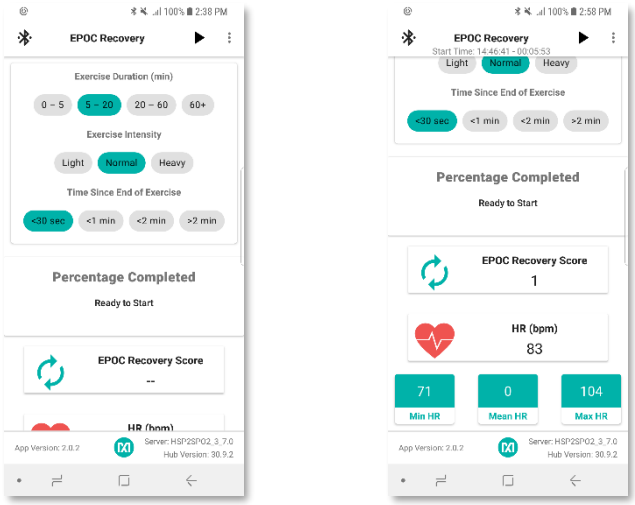

Figure 21: EPOC Recovery

After selecting these parameters, user can start measuring by pressing the play button. While measuring, the completion percentage is reported in the circular progress bar, and the heart rate is displayed in real-time. When the measurement finishes, the user can see his EPOC score together with HR statistics as seen in Figure 21.

Note 1: Mean HR is not reported for EPOC. It is a known issue and will be fixed in the next release.

Note 2: To calculate the EPOC Recovery, there should be at least 1 VO2Max calculation in the history.

2.2.12.6 Recovery Time

Recovery Time is the amount of time until user reaches to their homeostasis. Algorithm needs to gather 2-3 minutes of data from the user to estimate his/her recovery time. User can start measuring by pressing the play button. While measuring, the completion percentage is reported in the circular progress bar, and the heart rate is displayed in real-time. When the measurement finishes, the user can see his recovery time together with HR statistics as seen in Figure 22.

Note 1: Mean HR is not reported for Recovery Time. It is a known issue and will be fixed in the next release.

Note 2: To calculate the Recovery Time, there should be at least 1 EPOC calculation in the history.

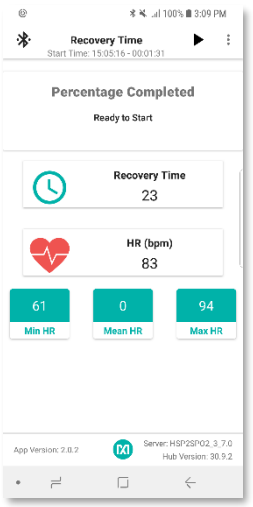

Figure 22: Recovery Time

2.2.12.7 Fitness Age

This page calculates the user's fitness age from the history data and compare it with his/her actual age as seen in Figure 23.

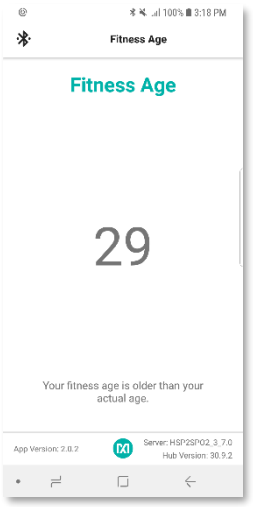

Figure 23: Fitness Age

2.2.12.8 History

The History page shows the list of all the measurements that the user takes. This page is searching for the files under "MaximSensorsApp/SPORTS_COACHING/<username>/" folder. Every item displays the date the measurement is taken, session name and score as seen in Figure 24.

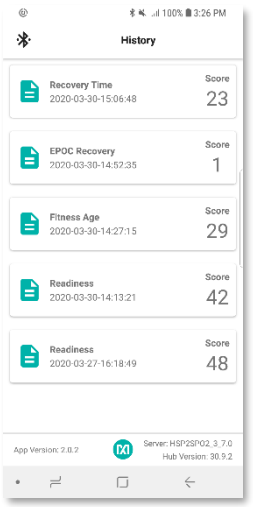

Figure 24: Sports Coaching User History

2.2.13 Flash Log Parser

Logging the sensor data is one of the crucial functions of the wearable devices. MRD103 devices are not only able to send the sensor data to the other devices but also save these data in its flash. The user can obtain these data by turning the mode of the device into storage mode. The extension for these files is “maximlog”.

To visualize this log file in Wellness Suite app, it needs to be converted to a csv file with a predefined format. Wellness Suite app provides Flash Log Parser functionality to the user.

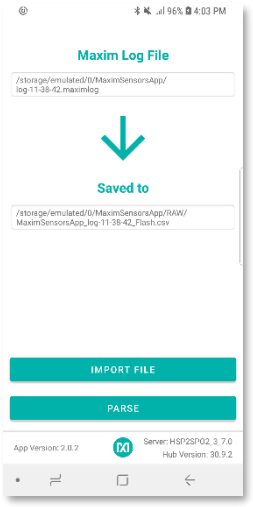

Figure 25:Flash Log Parser

As seen in Figure 25, the user needs to select the log file by pressing the “IMPORT FILE” button. He/she can see the log file location in the screen. To parse the log file, the user should press the “PARSE” button. After parsing is finished, the location to the csv file is displayed in the screen. If any problems occurred during parsing, the output file is not created, and the error message is displayed to the user.

2.2.14 HR Alignment

Wellness Suite app supports automatic Heart Rate Alignment. If the user wants to compare the Maxim heart rate results with a reference device's results, he/she can use this functionality of the Wellness app. There are 3 restrictions to accomplish this:

1) The MRD103 and the reference device should have been used at the same time.
2) Maxim raw data should be used for the alignment.
3) Reference HR log file should contain 2 columns: UNIX timestamp (ms) and Heart Rate (bpm). See Figure 26.

As seen in Figure 26, the user needs to select the Maxim Raw Data File by pressing the corresponding Attachment icon. After selecting the file, the file location is displayed in the screen. Similarly, the user should also select the reference HR file. After selecting both files, the user can align them automatically by pressing the "ALIGN" button. The Wellness Suite app displays the status of the alignment process.

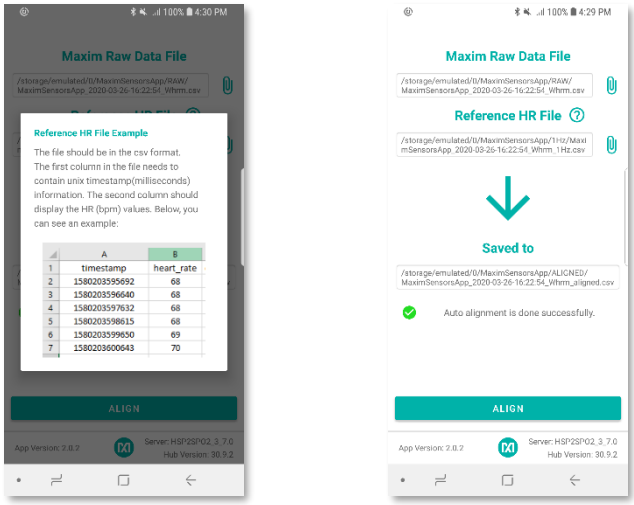

Figure 26: HR Alignment

3 Wellness Library Configuration Definitions

Wellness Library offers single configuration structure which covers configuration parameters for all algorithms included in the application. The config structure is defined in “Wrapper.h file”:

```
struct mxm_algosuite_init_data {
        unsigned char                           enabledAlgorithms;
        MxmHrvConfig                            hrvConfig;
        mxm_respiration_rate_manager_init_str   respConfig;
        mxm_stress_monitoring_config            stressConfig;
        mxm_sleep_manager_config                sleepConfig;
};
```

enabledAlgorithms field is the 8bit long field where Algorithms to be initialized at configuration is selected. Selection is done via configuration bits dedicated to each underlying algorithm:

```
#define MXM_ALGOSUITE_ENABLE_HRV    ( 1 << 0)
#define MXM_ALGOSUITE_ENABLE_RESP   ( 1 << 1)
#define MXM_ALGOSUITE_ENABLE_SLEEP  ( 1 << 2)
#define MXM_ALGOSUITE_ENABLE_STRESS ( 1 << 3)
#define MXM_ALGOSUITE_ENABLE_SPORTS ( 1 << 4)
```

Selected algorithms are initialized within the initialization phase of library and driven by inputs during running phase.

Each underlying library has its dedicated configuration field within the main configuration structure as define below.

3.1 HRV Algorithm Configuration

Heart Rate Variation Calculation Algorithm’s configuration field is described in “mxm_hrv_public.h“ API file distributed with Maxim Wellness Library. It is configuration structure is defined as:

```
struct _MxmHrvConfig {

    float samplingPeriod;
    uint16_t windowSizeInSec;
    uint16_t windowShiftSizeInSec;


}
```

SamplingPeriod: Input sample frequency of the HRV algorithm. It is recommended to set as 40.0 ms when other algorithms within the Library is enabled to run.

windowSizeInSec: Size of the sample buffer processed by HRV algorithm. Determines number of input samples to make an estimation. In sample’s unit it is windowSizeInSec/ samplingPeriod.

windowShiftSizeInSec: Determines resolution of HRV outputs. Smaller in the shift means increase in resolution. In sample’s unit it is windowShiftSizeInSec / samplingPeriod.

3.2 Respiration Rate Manager Algorithm Configuration

Respiration Rate Monitoring Algorithm's configuration field is described in "mxm_respiration_rate_manager.h " API file distributed with Maxim Wellness Library. It is configuration structure is defined as:

```
struct _mxm_respiration_rate_manager_init_str {

        mxm_respiration_rate_manager_ppg_source_options signal_source_option;
        mxm_respiration_rate_manager_led_codes led_code;
        mxm_respiration_rate_manager_sampling_rate_option sampling_rate;
        uint32_t motion_magnitude_limit;
   };
```

signal_source_option: setting for the body part where PPG signal is obtained

led_code: selection of LED source PPG data is excited from. Used fr identifying correct array indices for multiple LED channels

sampling rate: input sample rate selection. 25Hz and 100Hz options are accepted. If other algorithms are enabled to run in library, it is recommended to keep in 25Hz to get correct results or check for the sampling rate of other enabled algorithms within library.

motion_magnitude_limit: The motion magnitude limit in milig.

3.3 Sleep Quality Assessment Algorithm Configuration

Sleep Quality Assessment Algorithm's configuration field is described "mxm_sleep_manager.h" file distributed with Maxim Wellness Library. It is configuration structure is defined as:

```
struct _mxm_sleep_manager_config
    {

        mxm_sleep_manager_minimum_detectable_sleep_duration mxm_sleep_detection_duration;
        mxm_sleep_user_info user_info;
        bool is_resting_hr_available;
        bool is_confidence_level_available_hr;
        bool is_confidence_level_available_ibi;
        bool is_activity_available;

    } mxm_sleep_manager_config;
```

mxm_sleep_detection_duration: sets the minimum time sleep algorithm will start to produce accurate classifier results.

user_info: age, weight, gender and sleep resting heart rate of user

is_resting_hr_available: set weather value of sleep resting heart rate of user is input during configuration

is_confidence_level_available_hr: Availability of the confidence level for the Heart Rate Measurement

is_confidence_level_available_ibi : Availability of the confidence level for the Interbeat Interval Measurement

is_activity_available: Availability of the activity class identifier

3.4 Stress Monitoring Algorithm Configuration

Stress Monitoring Algorithm's configuration field is described "mxm_stress_monitoring.h" file distributed with Maxim Wellness Library. It is configuration structure is defined as:

```
struct _mxm_stress_monitoring_config {

    uint8_t dummy_config_for_compilation;

}
```

Algorithm's configuration uses a dummy configuration object to obey Maxim Wellness Library algorithm's notation.

4 Wellness Library I/O Interface Attribute Definitions

Maxim Wellness Library input format is defined in “AlgoWrapper.h” file distributed with Maxim Wellness Library:

```
struct mxm_algosuite_input_data {

        uint32_t inp_sample_count               ;
        uint32_t grn_count                      ;
        uint32_t grn2Cnt                        ;
        uint32_t irCnt                          ;
        uint32_t redCnt                         ;
        uint32_t accelx                         ;
        uint32_t accely                         ;
        uint32_t accelz                         ;
        uint32_t whrm_suite_curr_opmode         ;
        uint32_t hearth_rate_estim              ;
        uint32_t hr_confidence                  ;
        uint32_t rr_interbeat_interval          ;
        uint32_t rr_confidence                  ;
        uint32_t activity_class                 ;
        uint32_t r_spo2                         ;
        uint32_t spo2_confidence                ;
        uint32_t spo2_estim                     ;
        uint32_t spo2_calc_percentage           ;
        uint32_t spo2_low_sign_quality_flag     ;
        uint32_t spo2_motion_flag               ;
        uint32_t spo2_low_pi_flag               ;
        uint32_t spo2_unreliable_r_flag         ;
        uint32_t spo2_state                     ;
        uint32_t skin_contact_state             ;
        uint32_t walk_steps                     ;
        uint32_t run_steps                      ;
        uint32_t kcal                           ;
        uint32_t cadence                        ;
        uint32_t timestampUpper32bit            ;
        uint32_t timestampLower32bit            ;

};
```

Input of the Library is based on the extended report output of the WHRM suite running within SensorHub processor, i.e. ME11. **For the definition of field within input structure, please refer to API document of Max32664C.**

IMPORTANT: Input sampling frequency must be a multiple of 25 samples per second. 25Sps is the preferred rate.

Maxim Wellness Library output format is defined in “AlgoWrapper.h” file distributed with Maxim Wellness Library:

```
Struct mxm_algosuite_output_data {

        MxmHrvOutData                           hrv_out_sample;
        mxm_sleep_manager_output_dataframe      sleep_out_Sample;
        mxm_respiration_rate_manager_out_data_str  resp_out_sample;
        mxm_mxm_stress_monitoring_run_output    stress_out_sample;

};
```

Where each algorithm’s output has reserved fields.

Maxim Wellness Library return codes are defined in “AlgoWrapper.h” file distributed with Maxim Wellness Library:

```
Struct mxm_algosuite_return_code {

        mxm_algosuite_hrv_retcode    hrv_status;
        mxm_algosuite_resp_retcode   resp_status;
        mxm_algosuite_stress_retcode stress_status;
        mxm_algosuite_sleep_retcode  sleep_status;

}
```

Where each algorithm's status/return reporting is done in its reserved field.

4.1 HRV Algorithm's Output and Status Definitions

HRV Algorithm's output field is defined in "mxm_hrv_public.h" file distributed with Maxim Wellness Library:

```
struct _MxmHrvOutData {

    TimeDomainHrvMetrics timeDomainMetrics;
    FreqDomainHrvMetrics freqDomainMetrics;
    int percentCompleted;
    bool isHrvCalculated;

}
```

timeDomainMetrics: this is time domain related output field containing following attributes:

- **AVNN:** Average of normal to normal (NN) intervals in ms
- **SDNN:** Standard deviation of NN intervals in ms
- **RMSSD:** RMS value of successive differences in ms
- **PNN50:** Percentage of successive differences greater than 50 ms

FreqDomainHrvMetrics: this is frequency domain related output field containing following attributes:

- **ULF:** Power in Ultra Low Frequency band in ms^2
- **VLF:** Power in Very Low Frequency band in ms^2
- **LF:** Power in Low Frequency band in ms^2
- **HF:** Power in High Frequency band in ms^2
- **LF/HF:** LF/HF ratio
- **TOTPWR:** Total power in ms^2

percentCompleted: Status of algorithm computation

isHrvCalculated: Flag that indicates if the content of the output is valid

HRV Algorithm's status field is defined in "mxm_hrv_public.h" file distributed with Maxim Wellness Library:

```
enum MxmHrvRet {
    MXM_HRV_SUCCESS,
    MXM_HRV_NULL_PTR_ERROR,
    MXM_HRV_INVALID_CONFIG_ERROR,
    MXM_HRV_NON_POSITIVE_SAMPLING_PERIOD_ERROR,
    MXM_HRV_IBI_PREP_ERROR,
    MXM_HRV_METRIC_CALC_ERROR,
    MXM_HRV_ALREADY_INITIALIZED_ERR,
    MXM_HRV_NOT_INITIALIZED_ERR

}
```

4.2 Respiration Rate Algorithm's Output and Status Definitions

Respiration Rate Detection Algorithm's output field is defined in "mxm_respiration_rate_manager.h" file distributed with Maxim Wellness Library:

```
struct _mxm_respiration_rate_manager_out_data_str {

  float respiration_rate; /**< Output respiration rate with unit 'Breaths/Minute'*/
  float confidence_level; /**< Confidence level of the output in range of 0-100*/
  boolean_t motion_flag; /**< True if motion detected*/
  boolean_t ibi_low_quality_flag; /**< True if received IBI values are received with low quality*/
  boolean_t ppg_low_quality_flag; /**< True if input PPG signal is noisy (DC Jumps, spikes) */

}
```

respiration_rate: Output respiration rate with unit 'Breaths/Minute'.

confidence_level: Confidence level of the output in range of 0-100.

motion_flag: True if motion detected.

ibi_low_quality_flag: True if received IBI values are received with low quality.

ppg_low_quality_flag: True if input PPG signal is noisy (DC Jumps, spikes).

Respiration Rate Detection Algorithm's status field is defined in "mxm_respiration_rate_manager.h" file distributed with Maxim Wellness Library:

```
enum mxm_repiration_rate_manager_return_code {
        MXM_RESPIRATION_RATE_MANAGER_SUCCESS,
        MXM_RESPIRATION_RATE_MANAGER_INIT_NULL_PTR_ERROR,
        MXM_RESPIRATION_RATE_MANAGER_INIT_ERROR,
        MXM_RESPIRATION_RATE_MANAGER_RUN_NULL_PTR_ERROR,
        MXM_RESPIRATION_RATE_MANAGER_RUN_ERROR,
        MXM_RESPIRATION_RATE_MANAGER_END_NULL_PTR_ERROR,
        MXM_RESPIRATION_RATE_MANAGER_END_ERROR,
        MXM_RESPIRATION_RATE_MANAGER_GET_VERSION_NULL_PTR_ERROR,

    };
```

4.3 Sleep Quality Assessment Algorithm's Output and Status Definitions

Sleep Quality Assessment Algorithm's output field is defined in "mxm_sleep_manager.h" file distributed with Maxim Wellness Library:

```
struct mxm_sleep_manager_output_data_str {

        mxm_sleep_manager_sleep_wake_output_status sleep_wake_decision_status;
        mxm_sleep_manager_sleep_wake_output sleep_wake_decision;
        short int sleep_wake_detection_latency;
        float sleep_wake_output_conf_level;
        mxm_sleep_manager_sleep_phase_output_status sleep_phase_output_status;
        mxm_sleep_manager_sleep_phase_output sleep_phase_output;
        float sleep_phase_output_conf_level;
        float hr;
        float acc_mag;
        float interbeat_interval;

    };

struct mxm_sleep_manager_output_dataframe {
```

```
        mxm_sleep_manager_output_data_str * output_data_arr;
        unsigned int output_data_arr_length;
        uint64_t date_info
    };
```

Sleep/Wake Classification related outputs:

sleep_wake_decision_status: This enumeration is a flag indicating whether the sleep-wake decision is updated or not. Moreover, the sleep-wake-output status is set to NOT_CALCULATED during the first N minutes (initialization period) defined by config *mxm_sleep_manager_detection_duration_config*

sleep_wake_decision: Sleep/Wake state classifier output representing the status of sleep time instance:

MXM_SLEEP_MANAGER_WAKE: short-term decision for wake state

MXM_SLEEP_MANAGER_RESTLESS: short-term decision for restless state which corresponds to inconvenient portions during sleep rooting from small and non-periodic wiggles.

MXM_SLEEP_MANAGER_SLEEP: short-term decision for sleep state

sleep_wake_detection_latency:

sleep_wake_output_conf_level: Confidence level for the sleep/wake state classifier output.

Sleep Phases Classification related outputs:

sleep_phase_output_status: Sleep Phases Output status as calculated or not yet calculated

sleep_phase_output: Sleep Phases Classification output, whose states can be:

MXM_SLEEP_MANAGER_SP_UNDEFINED: Sleep Phases Output is not ready

MXM_SLEEP_MANAGER_SP_WAKE: wake condition

MXM_SLEEP_MANAGER_REM: Subject sleeps in in REM sleep

MXM_SLEEP_MANAGER_LIGHT: Subject sleeps in in light sleep

MXM_SLEEP_MANAGER_DEEP: Subject sleeps in in deep sleep

sleep_phase_output_conf_level: Sleep Phases Classification output confidence level.

hr: average heart rate value processed by SleepQA Algorithm

acc_mag: average accelerometer magnitude value processed by SleepQA Algorithm

interbeat_interval: average inter-beat interval value processed by SleepQA Algorithm

Sleep Quality Assessment Algorithm's status field is defined in "mxm_sleep_manager.h" file distributed with Maxim Wellness Library:

```
typedef enum _mxm_sleep_manager_return {
        MXM_SLEEP_MANAGER_SUCCESS,                  // Success return code
        MXM_SLEEP_MANAGER_INIT_NULL_PTR_ERROR,      // NULL pointer error during initialization
        MXM_SLEEP_MANAGER_INIT_INVALID_INPUT_ERROR, // Invalid Input error return code
        MXM_SLEEP_MANAGER_ALGO_INIT_ERROR,          // Algorithm module initialization error return code
```

```
    MXM_SLEEP_MANAGER_ALGO_END_ERROR,           // Algorithm module termination error return code
    MXM_SLEEP_MANAGER_RUN_NULL_PTR_ERROR,       // NULL pointer error during execution return code
    MXM_SLEEP_MANAGER_RUN_ALGO_ERROR,           // Algorithm execution error return code
    MXM_SLEEP_MANAGER_RUN_ALGO_AGE_OR_GENDER_NOT_DEFINED, // Age or gender is not defined flag

}
```

4.4 Stress Monitoring Algorithm's Output and Status Definitions

Stress Monitoring Algorithm's output field is defined in "mxm_stress_monitoring.h" file distributed with Maxim Wellness Library:

```
struct _mxm_stress_monitoring_run_output {

    boolean_t stress_class;
    uint8_t stress_score;
    float32_t stress_score_prc;

}
```

stress_class: Binary stress/non-stress output

stress_score: Integer stress score output in interval [0 18]:

Score	Meaning
1-2	Overwhelmed
3-5	Frustrated
6-7	Manageable Stress
8-9	Feeling Okay
10-11	Doing Great
12-14	Very Relaxed
15-18	Uncomfortably Relaxed

Table 2: Stress Score Provision

stress_score_prc: Stress score output as percentage

Stress Monitoring Algorithm's status field is defined in "mxm_stress_monitoring.h" file distributed with Maxim Wellness Library:

```
enum _mxm_stress_monitoring_return_code {

    MXM_STRESS_MONITORING_SUCCESS = 0,        //SUCCESS code
    MXM_STRESS_MONITORING_NULL_PTR_ERR,       // NULL pointer to configuration structure
    MXM_STRESS_MONITORING_INVALID_CONFIG_ERR, // Configuration error
    MXM_STRESS_MONITORING_RUN_ERR,            // Algorithm process error

}
```

5 Wellness Library API Function Definitions

5.1 Algorithm Related Process

Maxim Wellness Library API functions are defined in “AlgoWrapper.h” file distributed with Maxim Wellness Library:

mxm_algosuite_manager_init( ) : Initialization call of the Maxim Wellness Library.

Input: init_str : pointer to initialization structure of Maxim Wellness Library, described as *mxm_algosuite_init_data*

Output: status: pointer to Maxim Wellness Library return codes, described as *mxm_algosuite_return_code*

mxm_algosuite_manager_run( ) : Method to dispatch processing of Maxim Wellness Library algorithms. Must be called with arrival of every new input sample. Outputs an output sample and a return code per input sample fed to method.

Input: data_in_str: pointer to input data sample of Maxim Wellness Library, described as

mxm_algosuite_input_data

Output: data_out_str: pointer to Maxim Wellness Library output sample structure, described as *mxm_algosuite_output_data*

Output: status: pointer to Maxim Wellness Library status output sample structure, described as *mxm_algosuite_return_code*

mxm_algosuite_manager_end( ) : End call of the Maxim Wellness Library.

Input: tobeDisabledAlgorithms: 8BIT filed to select algorithms to be disabled. Bit fields reserved for algorithms are explained in configuration definitions section.

Output: status: pointer to Maxim Wellness Library status sample structure, described as *mxm_algosuite_return_code*.

mxm_algosuite_manager_get_versions ( ) **:** returns the version information of Wellness Library.

Output: version_str: pointer to Maxim Wellness Library version structure, described as:

```
struct mxm_algosuite_version_str {
        char version_string[20];            //version in XX.XX.XX format
        unsigned short int version;         // The first number in XX.XX.XX format
        unsigned short int sub_version;     //The second number in XX.XX.XX format
        unsigned short int sub_sub_version; // The third number in XX.XX.XX format
}
```

mxm_algosuite_calculate_SQI ( ) : Calculates the Sleep Quality Index related to subject's sleep. This function should be called after the detection of subject's wake-up for post processing of Sleep Quality assessment algorithm's outputs.

Input: deep_time_in_sec : time spent in DEEP state (in seconds) during the subject's sleep period

Input: rem_time_in_sec : time spent in REM state (in seconds) during the subject's sleep period

Input: in_sleep_wake_time_in_sec: total sleep time, time between Bedtime and Wake-Up time, duration in secs

Input: number_of_wake_in_sleep : count of wakes during the sleep period

Output: output_sleep_quality_index: calculated Sleep Quality Index

5.2 Authentication Related Process

Authentication to the SensorHub is a requirement for the Maxim Wellness Library. Authentication must take place before initialization of Algorithms within Wellness Library. Authentication Steps, which are also explained in integration document are:

1. Ask Authentication initials to the SensorHub, which is defined in **Max32664C/B API document**. Authentication initials is 12-character long byte array.
2. Call **mxm_algosuite_manager_getauthinitials()** Wellness Library API function with the Authentication initials stream obtained in step1 from Sensorhub. Function returns out_auth_initials of the Wellness Library which is a byte stream of 12 characters.
3. Send the authentication array of Library obtained in step-2 to SensorHub via command defined in **Max32664C/B API document.**
4. Get the Two authentication keys of SensorHub via commands defined in **Max32664C/B API document.** Those Keys are:
    - 12byte long auth_str1[]
    - 32byte long auth_str2[]
5. Call **mxm_algosuite_manager_authenticate()** Wellness Library API function with the Authentication Keys requested from SensorHub in step-4; auth_str1[] and auth_str2[] to complete authentication process, After completing Authentication, Algorithms within Wellness Library can be initialized and run.