maxim integrated™

EMPOWERING DESIGN INNOVATION FOR HEALTHCARE WEARABLES

MOUSER ELECTRONICS

CONTENTS

CONNECT WITH MOUSER

Terms Of Use: ©2020 Mouser Electronics, Inc., a TTI Berkshire-Hathaway company. Mouser and Mouser Electronics are registered trademarks of Mouser Electronics, Inc., Maxim Integrated, logo, product names, service marks, are trademarks, registered trademarks of Maxim Integrated. All other trademarks are the property of their respective owners. Names, products, specifications, links, resources, and similar were verified at the time of publication. Article content, reference designs, datasheets, conceptual illustrations, and other images provided herein are for informational purposes only.

FOREWORD

In an unprecedented time like this, when healthcare resources are stretched in ways we've never experienced in modern times, patients can face challenges in accessing our traditionally centralized healthcare services. In some cases, they might suffer additional discomfort if care is delayed or face risks if they visit a facility where care is also being given to COVID-19 patients.

The time is now to embrace and accelerate the deployment of remote patient monitoring to ensure that chronic conditions are managed to improve outcomes and reduce the need for regular healthcare facility visits. In addition, in the event of pandemics such as COVID-19, remote monitoring can be a valuable tool to help healthcare professionals triage incoming cases based on each patient's condition.

We certainly have the technology to support a more decentralized yet personalized model of healthcare. From sophisticated sensors to advanced algorithms and artificial intelligence (AI), the underlying technologies are now available to enable accurate wearable medical devices that collect and transmit data to support remote patient monitoring.

Challenges with deploying technology at the point of use are certain. And as society becomes more technology-oriented and if individuals are not able to apply the solution to themselves, a care network of family members or professionals can often assist. Connectivity challenges and ensuring that data is being collected reliably can be solved with careful product design and simple set-up requirements. Recently, the Centers for Medicare & Medicaid Services (CMS) approved four Current Procedural Terminology (CPT) codes for the reimbursement of remote patient monitoring (RPM). (CPT is a medical code set used to report medical, surgical, and diagnostic procedures and services to entities such as physicians, health insurance companies, and accreditation organizations for reimbursement.) So, the incentive is there to move toward more remote monitoring.

Wearable devices in varied forms—rings, wristbands, earpieces, and patches—can provide continuous remote monitoring in a convenient, unobtrusive manner for key biometric signals, such as heart rate, blood-oxygen levels (SpO2), and body temperature. In the future, blood pressure and blood-glucose levels will likely be added to the list. Continue reading to learn how Maxim is delivering more personalized healthcare solutions to improve outcomes and at a lower overall cost—to create a healthier world.

Andrew Baker
Managing Director, Advanced Sensor Products, Maxim Integrated

CREATING A WEARABLE HEART RATE MONITOR

JOSEPH DOWNING, *Mouser Electronics*

This project introduces the MAX32630FTHR and MAXREFDES117 and provides instructions for creating a basic heart-rate monitor wearable device. You'll learn how to program the device, wire the development boards, compile and load the Android app, and transmit data to a cloud service.

In recent years, the use of wearable devices has exploded across multiple markets, in large part because of their convenience and the plethora of information they provide. Activity trackers such as Samsung's Gear Fit2, medical devices like the Qardio®Arm blood pressure cuff, and even Under Armor's UA SpeedForm® Gemini 3 Record-Equipped shoes are a few examples. These devices can provide users with various feedback, including sleep quality, VO_2 levels, activity levels, and walking and running cadence, among other data points.

Designing wearable devices requires adding peripherals to sense and display various types of data as well as store and retrieve data in the cloud. This project uses the Pegasus Rapid Development Platform, from Maxim Integrated™ (Maxim), which eases development by integrating key peripherals into the development board along with Maxim's 700-MAXREFDES117# heart-rate monitor reference design. Rounding out the project technologies, we use an Mbed operating system (OS) for cloud-based programming, Ubidots for cloud services, and Android Studio software for the cloud interface.

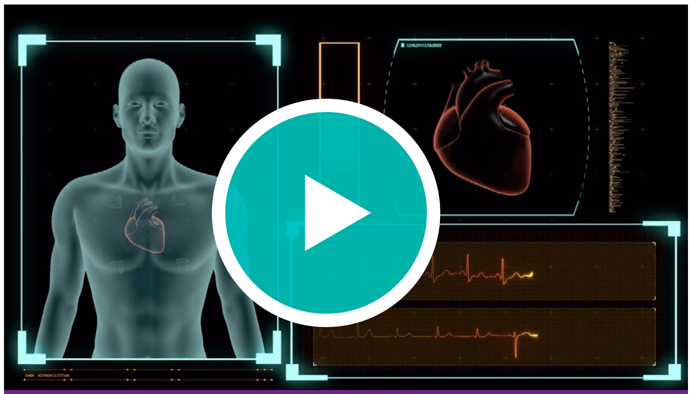
If you're a designer (or a DIYer), using these integrated features, reference designs, and cloud programming tools will put you a step ahead at the starting line. The following sections identify necessary project materials and help you program the device, wire the development boards, compile and load the Android app, and transmit data to a cloud service.

Project Materials and Resources

We recommend gathering the following materials and resources before starting this project:

Project's Bill of Materials (BOM)

- Access the project's BOM on Mouser.com for these required components:
- Maxim's 700-MAX32630FTHR# development platform, powered by the MAX32630 Arm® Cortex® M4F microcontroller and accompanied by the MAX14690 PMIC battery charge management device
- Maxim's 700-MAXREFDES117# heart-rate monitor with heart rate/pulse oximeter sensor, a step-down DC-DC converter, and logic-level translator
- Lithium Polymer Battery



How to Get Your Healthcare Wearable Off the Ground Faster

Project's Code

- Mouser HRM Mbed repository
- Mouser HRM GitHub repository

Hardware

- Soldering iron
- Jumper wire or regular wire
- Flux
- Header pins
- Breadboard
- Digital multimeter (optional)
- Oscilloscope (optional)
- Expansion peripherals (optional)

Accounts and Software

- A Ubidots Cloud service account at Ubidots.com
- An Mbed.org account
- Android Studio software

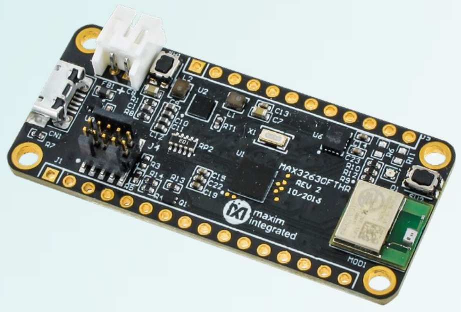


Figure 1: Maxim's MAX32630FTHR Pegasus Development Platform is powered by an Arm® Cortex® M4F microcontroller and comes with a PMIC power management device

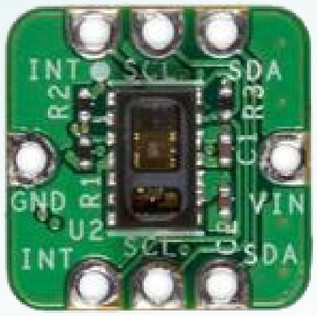


Figure 2: Maxim's popular MAXREFDES117# heart rate module reference design is tiny yet comes equipped with a pulse rate/oximeter sensor, a step-down DC-DC converter, and a logic-level translator

Project Technology Overview

This is an intermediate to advanced project geared toward engineers and DIYers who have programming and soldering experience. We've designed this project using the following technologies:

Maxim MAX32630FTHR Pegasus Development Platform

Driving this project is Maxim's new MAX32630FTHR Pegasus Development Platform (**Figure 1**). Powered by the MAX32630 Arm® Cortex® M4F microcontroller and accompanied by the MAX14690 PMIC battery charge management device, this platform can assist engineers in rapid prototyping. This feature-packed board sports several integrated peripherals, such as an accelerometer/gyroscope and dual-mode Bluetooth as well as SPI, I²C, UART, 66 GPIO, and much more.

The MAX32630FTHR small-form factor is compatible with several off-the-shelf expansion boards as well as standard breadboards, presenting countless possibilities. If you are curious about some of the available options, we've provided a link under the expansion peripherals.

Maxim MAXREFDES117# Heart Rate Module Reference Design

Maxim's popular MAXREFDES117# heart-rate module reference design (**Figure 2**) is tiny yet comes equipped with Maxim's MAX30102 heart rate/pulse oximeter sensor, MAX1921 step-down DC-DC converter, and MAX14595 logic-level translator. This versatile design can be used within both the Arduino and Mbed platforms for quick integration. Example firmware is available for both platforms and provides the user with a very basic algorithm for determining heart rate and SpO², to help get them off the ground.

Mbed OS for Cloud-Based Programming

Mbed OS gives a convenient cloud-based programming tool to help simplify and speed up the creation of Internet of Things (IoT) platforms. Mbed provides the tools to help collaborate on, contribute to, and publish software for existing code, all while helping to maintain a detailed revision history. Starting with Mbed is as simple as creating an account and finding and selecting the hardware you wish to use. If you have an existing account, you can go to the Mbed repository and import the provided code into your compiler to begin.

Ubidots for Cloud Servicing

Ubidots offers a great jumping-off point for anyone wanting to begin an IoT or cloud project. In addition to having tutorials available for several development platforms, Ubidots offers tutorials on creating an interface between their service and Android apps created in Android Studio. The credit system allows for a simple and affordable way to develop and maintain your projects and offers several methods for obtaining more credits when necessary.

Android Studio for Cloud Interfacing

You'll need an interface to transfer data from your Maxim board and sensor to the cloud. This can be carried out through a mobile device, such as a tablet or cell phone. For this project, we stayed within the confines of the Android framework and used Android Studio for app creation to aid in visualizing the information our sensor provides.

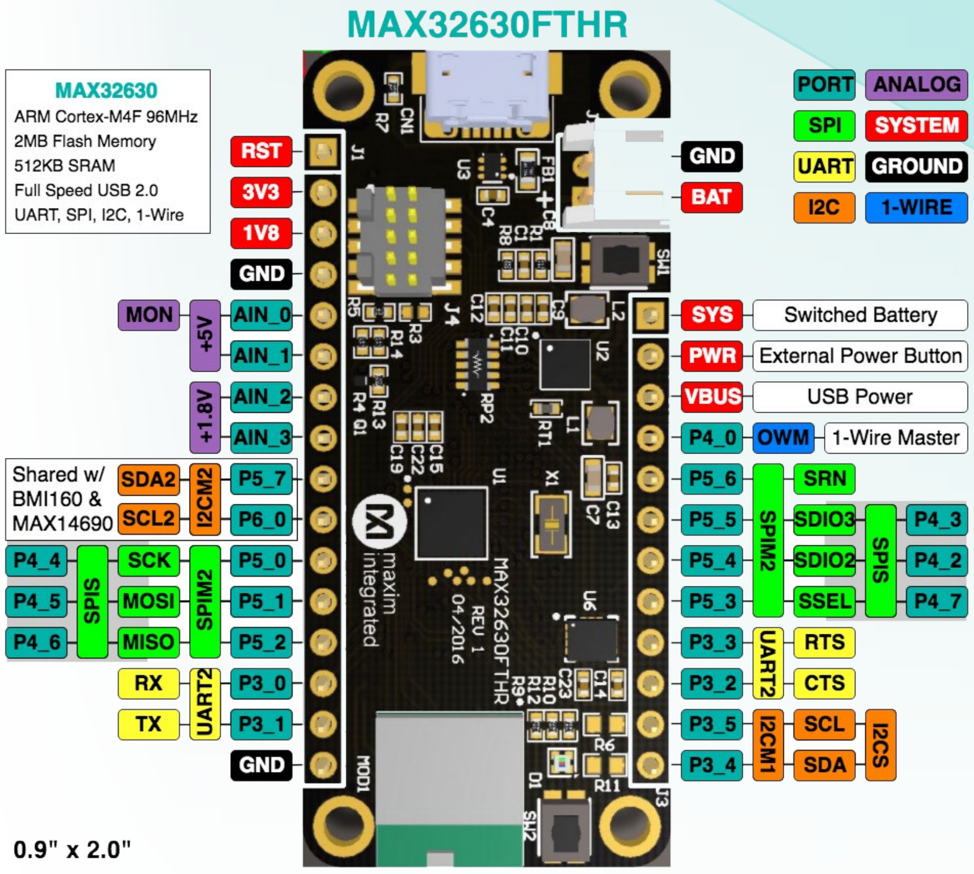


Figure 3: Pin mapping from Maxim MAX32630FTHR datasheet

> Programming the board is as easy as dragging and dropping a file from one folder to another.

Developing the Heart Rate Monitor

Working with Mbed

If you have an existing Mbed account, use the Mouser HRM Mbed repository and import the provided code into your compiler.

For first-time users:

1. Go to www.Mbed.org.
2. Click the Mbed OS link, which will take you to the developer site.
3. Click Log in/Sign up and provide the required information.

Once registered, you can either begin a new project by selecting the highlighted Compiler link or by accessing the Mouser HRM Mbed repository and importing the code to the compiler. You can select, review, add, or change platforms within the Compiler screen by clicking the link in the top right corner. You will find several other Maxim products and software samples also supported through Mbed.

Wiring the Board and Reference Design

Wiring the MAX32630FTHR and MAXREFDES117 together requires only a few connections and can be breadboarded to make this easier. To do so, use the following steps:

1. Connect the SCL and SDA lines from the heart rate sensor with P3_5 and P3_4 on the MAX32630FTHR and the INT on the heart-rate device to pin P3_0 (**Figure 3**).
2. The V_{IN} for the heart rate sensor needs between 2.5V to 5.5V, but I recommend deriving power from the 3V3 or SYS connected with the battery. I've had several issues between the two when attempting to use 5V.
3. Connect your ground pin.

Programming the Board

Programming the board is as easy as dragging and dropping a file from one folder to another. The MAX32630FTHR ships with a debug interface used for uploading your code to the board.

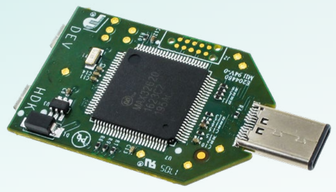


Figure 4: DAPLINK with multiple inputs

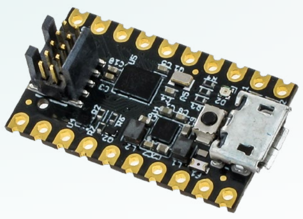


Figure 5: DAPLINK with single input

To begin:

1. Connect the polarized 10-pin connector from the interface board to the MAX32630FTHR. Pay particular attention to Pin 1 if the connection is not shrouded.
2. Connect the micro USB connector labeled HDK (on the interface board) to your PC USB port, which should automatically start driver installation. Depending on which DAPLINK you've received, you might have multiple connections or only one (either **Figure 4** or **5**, depending). Once the drivers have finished installing, a new drive labeled DAPLINK will be available in your File Explorer.
3. Provide power to the MAX32630FTHR either through the micro USB or JST battery connector.

Once you've made your connections and interfaced to the board, you'll need to compile the code to create the `.bin` file necessary to program the MAX32630FTHR. From the Mbed Compiler screen toolbar:

1. Click Compile or use the drop-down menu and select the option you want to use. After a few moments, the software will add a file to your Downloads folder labeled `BLEHeartRateMon_MAX32630FTHR.bin`, which you will want to locate.
2. Go ahead and locate that `.bin` file in the Downloads folder.
3. Open a new Explorer window, which will now display a new, available DAPLINK drive. This is your interface board.
4. Drag or copy the newly created `.bin` file from its download location to the DAPLINK folder to begin programming the board.

The DAPLINK board LEDs should blink rapidly while the programming is taking place. Once completed, the LED will either remain off or blink slowly and steadily on DAPLINK.

5. Press the reset button on the MAX32630FTHR to start the programmed code.

Once the board resets, a red LED on both boards will turn on and remain steady.

The Cloud

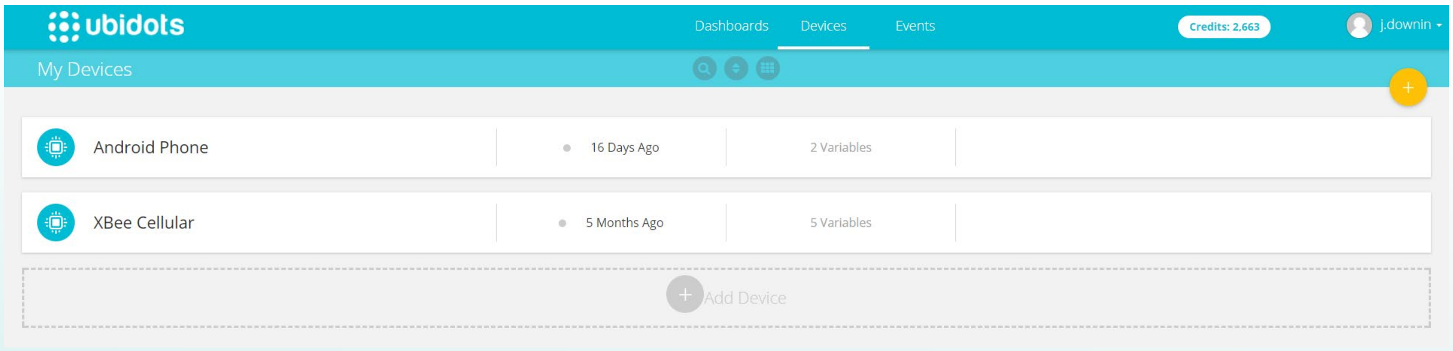
Once signed in, you should find yourself on the Dashboard page, and from here you can select several options such as view created devices, view available events, review your profile, and see how many credits are available on your account. A few things to note, which will be necessary for any project, are the API tokens, devices, and variable IDs:

Viewing API Tokens

To view the API token, click the Profile icon in the top right of the screen and select API Credentials, this will drop down a field displaying the API token on the left. The API token is unique to each user and is created when the account is created. This is what stops data transmitted from your device from going to the wrong account.

Adding Devices

Variable IDs are created within each new device when a variable is added. To begin, start by adding a device:



Android Phone	16 Days Ago	2 Variables
XBee Cellular	5 Months Ago	5 Variables

Figure 6: Start with the "Devices" screen

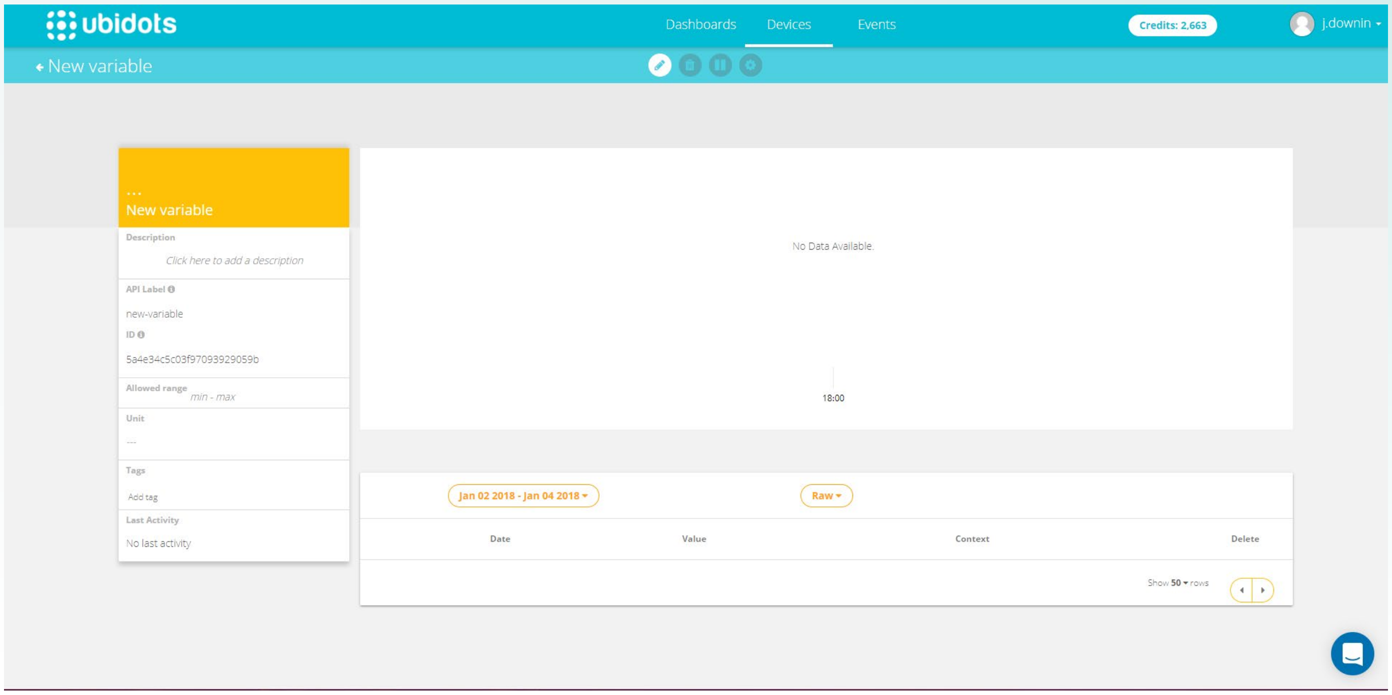


Figure 7: Ubidots "Variable" screen

1. Verify that you're on the Devices screen (**Figure 6**).
2. Click the yellow circle with the plus symbol in the top right corner.
3. Rename the device from My Data Source to whatever you choose.
4. Click anywhere in the window to finalize the device. You will see a brief pop-up message indicating the successful creation.

The device represents your project as a whole, allowing you to have multiple projects under one account. The variable ID is what helps each created device to identify the specific sensors and inputs assigned to, or built into, your device and what points them to the display in the correct location. This helps you maintain multiple projects and sensors that are separate or shared between devices.

Creating Variable IDs

Creating variable IDs is much like creating the device:

1. Click the New Device icon, which will open this device and enable you to create individual variables.
2. Click the yellow circle with the plus symbol in the top right corner to create your new variable.
3. Rename the variable.
4. Click anywhere within the window. A brief pop-up message will indicate that you've successfully created the ID.

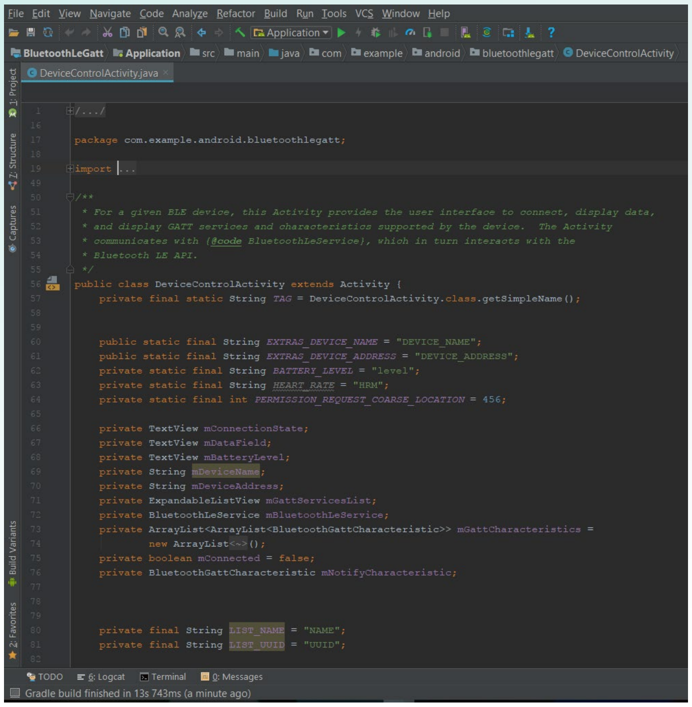
Clicking this new icon will bring up the Created Variable screen (**Figure 7**), which will display several pieces of information along the left-hand side. You can edit much of the information along the left column, which includes API label, allowed range, unit of measurement, and similar (The important item to note is the ID, which will identify where the data from your program should be placed.).

I highly recommend reviewing the tutorial documentation, located through the drop-down menu by hovering over your Profile icon. A lot of very good information is available on topics such as MQTT and HTTP API interfaces along with tutorials on other IoT devices.

Android Studio

You'll need an interface to transfer data from your Maxim board and sensor to the cloud. This can be carried out through a mobile device, such as a tablet or cell phone. For this project, we stayed within the confines of the Android framework and used Android Studio for app creation to aid in visualizing the information provided by our heart rate sensor.

To save time, I started with the example app, BluetoothLeGatt, which allows you to scan for available Bluetooth devices, connect to those devices, and see the resources available. This can give you a good example for building your own Bluetooth-enabled apps (if you choose to go that route). Ubidots also


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Application
BluetoothLeGatt / Application / src / main / java / com / example / android / bluetoothlegatt / DeviceControlActivity
DeviceControlActivity.java
1: Project
7: Structure
Captures
Build Variants
2: Favorites
TODO
6: Logcat
Terminal
0: Messages

```
/.../

package com.example.android.bluetoothlegatt;

import ...

/**
 * For a given BLE device, this Activity provides the user interface to connect, display data,
 * and display GATT services and characteristics supported by the device.  The Activity
 * communicates with {@code BluetoothLeService}, which in turn interacts with the
 * Bluetooth LE API.
 */
public class DeviceControlActivity extends Activity {
    private final static String TAG = DeviceControlActivity.class.getSimpleName();


    public static final String EXTRAS_DEVICE_NAME = "DEVICE_NAME";
    public static final String EXTRAS_DEVICE_ADDRESS = "DEVICE_ADDRESS";
    private static final String BATTERY_LEVEL = "level";
    private static final String HEART_RATE = "HRM";
    private static final int PERMISSION_REQUEST_COARSE_LOCATION = 456;

    private TextView mConnectionState;
    private TextView mDataField;
    private TextView mBatteryLevel;
    private String mDeviceName;
    private String mDeviceAddress;
    private ExpandableListView mGattServicesList;
    private BluetoothLeService mBluetoothLeService;
    private ArrayList<ArrayList<BluetoothGattCharacteristic>> mGattCharacteristics =
            new ArrayList<~>();
    private boolean mConnected = false;
    private BluetoothGattCharacteristic mNotifyCharacteristic;



    private final String LIST_NAME = "NAME";
    private final String LIST_UUID = "UUID";
```

```
Gradle build finished in 13s 743ms (a minute ago)
```

Figure 8: Android Studio Ubidots API token code

gives a tutorial on connecting and sending data through the app to their service, which I recommend reviewing for instructions on where to put your API and variable tokens.

Importing the Code

1. From Mouser's GitHub, download the files labeled BLEHR.7z necessary for import into Android Studio. Due to file-size restrictions, the project code is archived twice and will require you to extract it.
2. Import the code into Android Studio either by clicking Open an existing Android Studio project from the main start screen or by going to the File→Open menu. From here you can review how I changed and implemented the API for Ubidots (**Figure 8**).

(**Note:** Remember to change the token keys before building your program, as discussed earlier. If the keys are not updated before building, your data will not be displayed on Ubidots.)

Building the App

To test and debug code, you can:

- Build an app to sideload to your device. Use this option if you don't plan to modify the code beyond updating the API and variable tokens.
- Use the Developer's option. Use this option to enable USB debugging. This will also allow you to use your phone or tablet as a virtual test bed by plugging it directly into the PC you're programming on. By a web search or through tutorials in Android Studio, you can find instructions on how to enter the developer mode.

If you decide to use a device as a development tool, you can simply hit Run from the menu bar and select your device to begin building and running the app. Once loaded, the app should appear on your selected device. (**Note:** You can't run this app from a virtual device, because Bluetooth is not supported.)

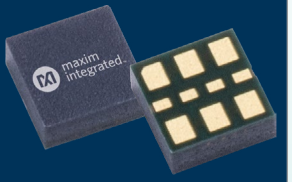
If you plan to simply sideload, you will need to first build the code by selecting Build, then Build APK(s). This will create a new file, which you can then copy to your chosen device to install.

Even though sideloading the app is an option that works well if you don't plan to make changes to the software, I do recommend using a device of your choice as a test bed. Using a device in USB debug mode will allow you to update, rebuild, and run your code repeatedly, which can be very useful in programming and debugging.

Putting It All Together

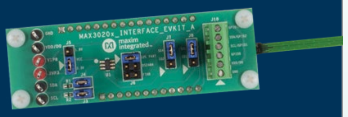
Once you've had an opportunity to load the app and launch it, you can see a Scan icon in the top-right corner that will allow you to scan for any Bluetooth devices in range. If you've wired and programmed the MAX32630FTHR and MAXREFDES117 correctly and the device is powered on, you should see a device labeled HRM in the list of devices that populates.

MAX30208 I2C DIGITAL TEMPERATURE SENSOR



LEARN MORE

MAX30208 EVALUATION SYSTEM



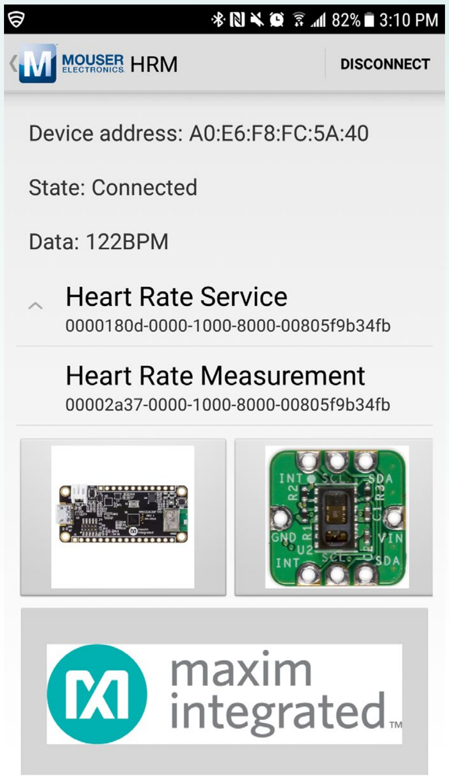
LEARN MORE

Ready to see the results?

1. Verify that location permissions are enabled. If you aren't sure where this option is located, refer to your device's online help or documentation.
2. Select HRM to open a new screen. You'll see several tidbits of data: device address, current state, and data output.
3. Click the Connect button in the top-right corner to allow connection to the device and to see the list of services through a drop-down menu.
4. Lastly, select Heart Rate Measurement, and you'll see data start to populate the screen (**Figure 9**).

If you've updated the code to include the correct API and variable tokens, you should now be able to log into your Ubidots account and view available data within the device, through the Device tab.

You can now create a dashboard for a quick view of the information in several different formats depending on your requirements (**Figure 10**). Ubidots has recently released a beta Android app that you can install to view your dashboard and devices, from either a phone or tablet (**Figure 11**).



Device address: A0:E6:F8:FC:5A:40

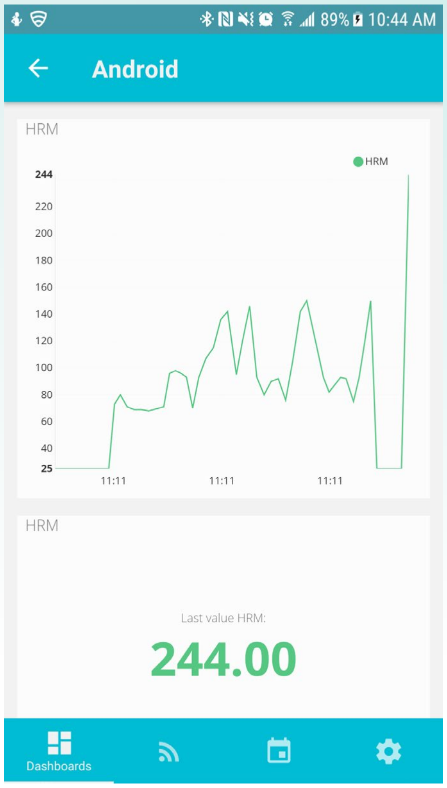
State: Connected

Data: 122BPM

Heart Rate Service
0000180d-0000-1000-8000-00805f9b34fb

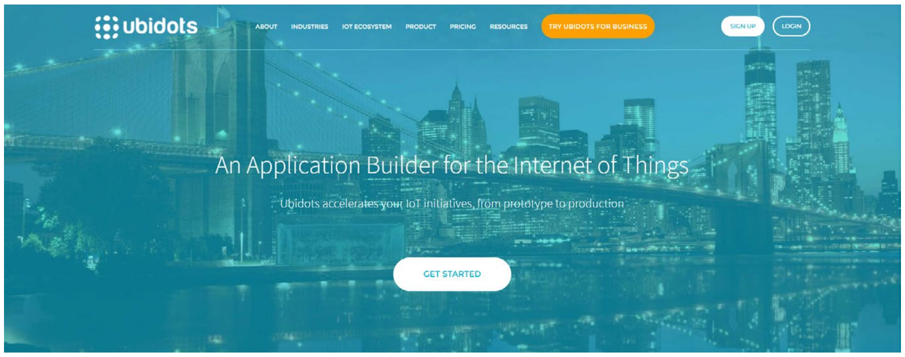
Heart Rate Measurement
00002a37-0000-1000-8000-00805f9b34fb

Figure 9: When you select "Heart Rate Measurement," you'll see data start to populate the screen



Last value HRM:
244.00

Figure 10: Ubidots Android App Dashboard



An Application Builder for the Internet of Things

Ubidots accelerates your IoT initiatives, from prototype to production

Figure 11: Ubidots Website Dashboard

Conclusion

Wearable devices are useful tools that can add convenience to our day-to-day lives. These devices can provide users with a variety of feedback, including sleep quality, VO_2 levels, activity levels, and walking and running cadence, among other data points. What's more, they can help you monitor your health and communicate to health-care providers by sending information, such as daily blood pressure or blood glucose levels.

Designing wearable devices requires adding peripherals to sense, display, store, and retrieve data. The Pegasus Rapid Development Platform has eased development by integrating key peripherals into the development board, and the 700-MAXREFDES117 heart rate monitor reference design is straight-forward to use. Mbed OS, Ubidots, and Android Studio software have rounded out the technologies for cloud-based programming, cloud servicing, and cloud interfacing, respectively. ●

GETTING TO RELIABLE SENSOR DATA: Interaction Between Electronics, Optics, and Mechanical Design

IAN CHEN, *Executive Director, Industrial & Healthcare Business Unit, Maxim Integrated*

This article will illustrate sensor data reliability challenges using an optical heart-rate sensor as an example. However, the complex nature of sensor quality applies to most sensors, not just optical ones.

Getting reliable sensor data is complex. For example, an optical heart-rate sensor is actually sensing the changes in an electric current. Heartbeats cause the volume of arterial blood to change synchronously with each pulse. The change in volume changes the amount of light absorbed and reflected as it travels through live tissue. When that light exits the tissue and enters a photodetector, it changes the output current. By creating a sensing system that carefully sets up a proper series of dominos, a current sensor becomes a heart-rate sensor.

This is how most sensors work. Sensors generally make esoteric electrical measurements (capacitance, impedance, current, voltage). But through an elaborate system construct, a physical event of interest (acceleration, pressure, footsteps, distance) is made to change that measurement. Knowing the system construct, we could then interpret the change into a physical parameter, while assuming everything else in the sensing system stays constant or is at least wellcontrolled.

But what if the dominos aren't all under the designer's control?

This article will illustrate the challenges of sensor-data reliability using an optical heart-rate sensor as the example. However, the complex nature of sensor quality applies to most sensors, not just optical ones.

Understanding the Optical Path

The optical path is the path light travels from the source (emitter) to the detector (receiver). The path spans across one or more media, and any change in those media could interact and affect the light characteristics at the detector. As such, the received light encapsulates all changes in the media along the entire optical path.

For an optical heart-rate sensor, light comes from one or more LEDs and is aimed at live tissue. Before arriving at the tissue, the light has to travel through air and sometimes a layer of cover glass. Air, cover glass, and the surface of the tissue are three different optical media. Similarly, the tissue is not homogenous and could be modeled as successive optical media layers with different refractive indices.

At the interface of any two media with different optical properties, light could be absorbed (attenuated), reflected (scattered) into the first medium, or transmitted into the second medium. **Figure 1** simplistically shows the many paths light can take after leaving the LED as it makes its way through different media.

We are not concerned with all the possible optical paths; only those optical paths that end at the photodetector are relevant to optical sensing. Heart-rate sensing works because one of these media, the capillaries, changes volume over time synchronous to heart rate. The change affects the amount of light absorbed and reflected. Optical system design must ensure that the path most light takes to go from the LED to the photodetector also interacts with the capillaries.

Data reliability is compromised when light can reach the photodetector without interacting with the capillaries or when something along the optical path changes unexpectedly. Notably, every media above the cover glass is outside the designer's control, and some optical properties could even change over time. Interpreting any change in the photodetector current as a change in heart rate would be a simplification. The following section looks at changes in the media traversed by the optical path and how they could affect the photodetector current.

Impact of Dirt on the Cover Glass and Other Attenuators

Dirt and grime on the cover glass can be unavoidable in practical applications. They largely serve to attenuate the photodetector current by reducing both the LED light that can reach the tissue and the light received by the detector. For heart-rate sensing, the important information is carried in the periodicity and not the signal's overall amplitude. So, as long as the emitter is strong enough, some attenuation should not result in any information loss. However, suppose the sensing configuration uses more than one LED or multiple wavelengths. In that case, it is possible that the same proportion would not affect light intensity for each LED and/or wavelength.

This discussion can be expanded to cover other light-attenuating factors outside of the designer's control. These factors include hair, skin pigment, and color changes of the cover glass. Each medium serves to attenuate light passing through it in both directions, and each may affect some wavelength of light more than others.

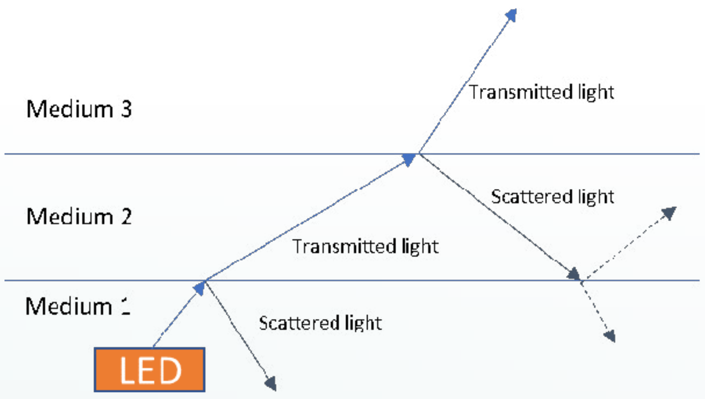


Figure 1: Example of the paths that light can take after leaving the LED and making its way through different media

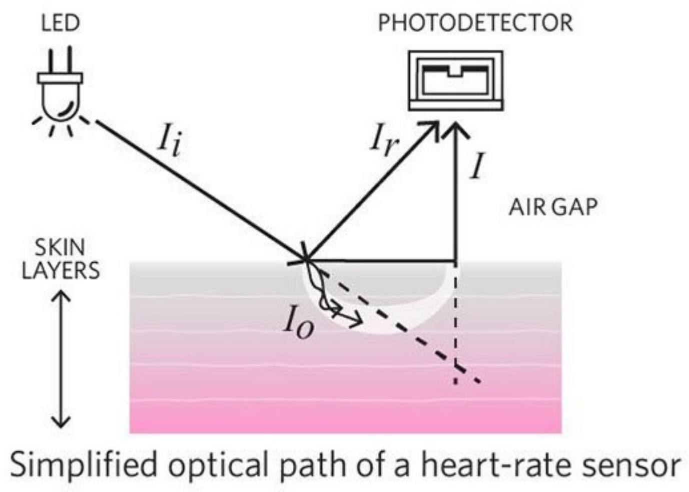


Figure 2: In the optical path of a heart-rate sensor, an air gap can degrade signal-to-noise, making it harder to attain reliable sensor data

Changing Air Gap or Path Length

Figure 2 depicts an optical heart-rate sensor. The distance between the skin surface and the optical components (light source and photodetector) is often called the air gap.

Light could enter the photodetector after being scattered by the skin surface (I_r) or after it has traversed through some layers of tissue (I). Only the optical path of I could have been affected by changes in the tissue and could, therefore, contain useful information. Consequently, a sound heart-rate sensor must minimize the magnitude of I_r by tending to the separation between the light source and the photodetector and their associated mechanical housing design. In doing so, the designers must assume the size of the air gap, which in practice they cannot fully control. When the air gap becomes larger, both I and I_r will be weaker and harder to detect because the photodetector is now farther away. At the same time, proportionately more light directly reflected off the skin surface can now enter the photodetector. Both factors degrade the signal-to-noise ratio of the sensor data, making any derived information less reliable.

Furthermore, unlike dirt and grime, the air gap can change periodically; for example, when subjects are exercising vigorously, the mechanical coupling between the sensors and the target tissues could change with a rhythmic motion. This will introduce a different periodic change in the photodetector current not controlled by capillary pulsation. As a result, heart-rate detection algorithms can become confused.

More than Air in the "Air Gap"

In many wearable applications, water (in the form of, say, sweat or rain) could be present in the air gap. The resulting combinations and variations are numerous, but we could consider some generalities. When the sensing target is live tissue, which is mostly comprised of water, having water in the air gap narrows the difference in refractive index between the air gap and the target. This should allow a proportionately greater amount of light to be transmitted into the tissue, strengthening the sensing mechanism.

Addressing Biological Change in the Subject

An inherent truth in long-term biosensing and monitoring is that the target (living tissue) will grow and change. For example, increasing pressure on the tissue could pinch off blood flow and diminish or otherwise compromise the photodetector's detected signal. Likewise, inflammation and swelling on the tissue change the optical path of the sensor.

Often, these changes are not challenges, but, instead, the objectives behind long-term optical biosensing. Being able to capture biological changes by monitoring changes to light through the tissue makes optical biosensors a useful tool for heart-rate monitoring and the foundation for many different types of non-invasive health and wellness monitoring. Nevertheless, when monitoring for one set of biological change, designers have to remain cognizant of other potential biological changes and how they can interact with the optical path to provide false signals and make the sensing data less reliable.

Maximizing Signal Path Performance

With everything that could affect the optical signal, it becomes increasingly important that the part of the optical data path under the designer's control gives the best signal-to-noise performance. A high-performance design makes discerning the reliability of the sensor data easier.

For example, any light that makes its way to the photodetector but did not enter the target tissue, not just the light emitted from the LED source, adds to the biosensing signal's noise. Some integrated analog front-end (AFE) devices, such as the MAX86140 and the MAX86171, sample any ambient current on the photodetector asynchronous to the LED light source and subtract them from the photodetector current. Indeed, these AFEs even anticipate how ambient light conditions could change in typical use cases so that designers can trust that their effects will have little impact on the biosensing signals.

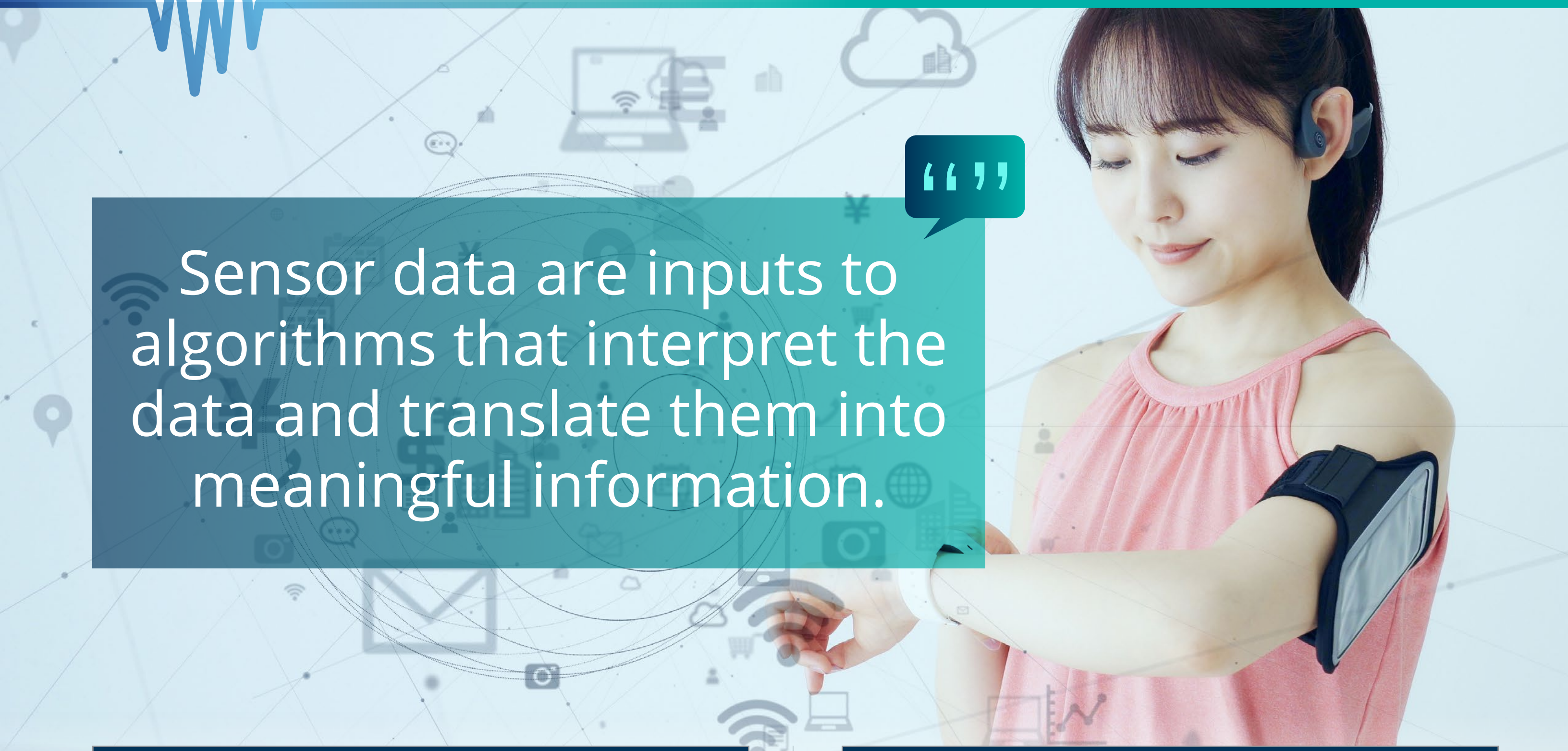
Know When Sensor Data Is Not Reliable

With many things along the optical path potentially changing, designers can include other mechanisms to detect potential changes to ensure sensor data reliability.

One coping strategy is to use sensors unaffected by optical path changes to monitor when such changes might occur. For example, accelerometers could notice moving targets, and pressure sensors could sense increased compression. Because these sensors use a different modality than optical sensors, they can at least warn designers that sensor data can be compromised and, in some cases, could even be used to help mitigate against using data with different optical paths, making the result more reliable.

Another strategy is to use more than one light frequency as lights of different colors are attenuated differently by each optical medium. Therefore, a change in the optical path would attenuate or scatter each colored light with different proportions. By comparing the emitted and the received light's spectral composition, designers can get information on how the optical path has changed.

Sensor data are inputs to algorithms that interpret the data and translate them into meaningful information. Algorithms could use known physical models or the context of the use case and historical sensor data to determine whether new data has become unreliable. In a future article, I'll take a more quantitative look at sensor data interactions and algorithms to provide reliable and actionable information to device end-users. ●



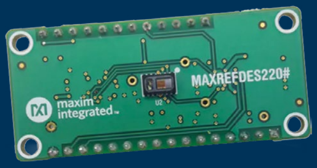
Sensor data are inputs to algorithms that interpret the data and translate them into meaningful information.

MAX32664 ULTRALOW POWER BIOMETRIC SENSOR HUB

LEARN MORE

MAXREFDES220# REFERENCE DESIGN

LEARN MORE



ADD ECG TO YOUR WEARABLE WITHOUT RAISING YOUR HEART BEAT

MAXIM INTEGRATED

In this design solution, we'll review how biopotential theory can be used to measure ECG using wet or dry electrodes connected to a chest-worn device, then consider the challenges in conditioning the ECG signal to allow accurate measurement with minimal power consumption.

Introduction

It looks like the next big thing in health and fitness monitors will be electrocardiogram (ECG) capability, but what if your wearable device doesn't have it? You can feel your stress levels rise as your potential customer base jumps aboard the departing ECG train, and there's nothing you can do about it. Surely, there's no way you can add this feature to your wearable quickly enough to catch up? If only adding ECG to a wearable wasn't such an expensive and time-consuming effort.

Perhaps you can relax after all. In this design solution, we review the theory and practice of measuring an ECG signal in a chest-worn wearable before proposing a composite solution to accelerate design effort and significantly reduce the time it takes to capitalize on this burgeoning technology.

What is Biopotential?

Biopotential measurements require placing two or more electrodes in contact with the skin of a patient's body to detect the small electrical signals generated by the heart. The signals are then conditioned and sent to a microprocessor for storage, calculation, and/or display. An electrocardiogram (ECG or EKG) is the measurement and graphical representation, with respect to time, of the electrical signals associated with the heart muscles. The R-R interval is the time between the peak amplitudes of the heart's periodic electrical signal, also known as R peaks (**Figure 1**).

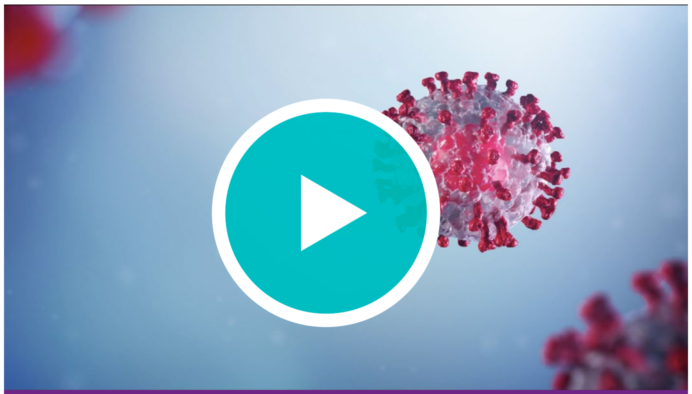
ECG and R-R measurement can be used for heart-rate monitoring to assist in diagnosing specific heart conditions, such as arrhythmias. However, these conditions can be difficult to diagnose because they do not always present themselves in a clinical environment. Wearable devices provide medical professionals with the ability to monitor patients over an extended period outside the hospital environment. This provides them with more information to assist in detection and diagnosis. For serious fitness enthusiasts, ECG can provide insight into peak performance intervals while training.

Measurement with a Chest Strap

Electrodes that come in contact with the skin—wet or dry types—are used to receive the ECG signals. Typically for clinical use, they are wet electrodes that use a sticky gel to adhere to the body. With chest-belt or chest-strap applications, the electrodes are dry. Electrodes are typically two pads manufactured into an elasticated and conductive material that connects to a small, sealed, and battery-powered electronics circuit. The electronics provide ECG signal processing and data conversion before wirelessly communicating to a host device, typically using Bluetooth®. To maintain a lightweight, low-profile device that is comfortable to wear, the sensor electronics will often be powered from a single coin-cell battery. There are several design challenges and important considerations when embarking on an ECG and heart-rate sensor design based on a chest strap.

Electrodes and Input Circuitry

Electrodes require a good connection to the body to provide a reliable signal of sufficient amplitude for detection. Electrode size and material characteristics also influence the signal quality and levels detected. Although dry electrodes are far more convenient than using wet types (you can just take them on and off), dry electrodes present a very high impedance when



How Remote Patient Monitoring Can Help Us Battle COVID-19

initially placed on the body. This means that the ECG signal is likely to be attenuated, resulting in a small signal. This dry-start' scenario typically lasts for a short duration until the wearer has exercised sufficiently to start sweating, lowering the impedance and increasing signal levels. To accommodate dry starts, the input impedance of the ECG channel's analog circuity should be very high so that attenuation is kept to a minimum. Also, while measuring ECG in a hospital environment requires the use of multiple electrodes on a stationary patient, this is not practical for mobile wearers of a portable device. The number of electrodes should be kept to a minimum (ideally no more than two, for example, single-channel).

Motion Artifacts in the Analog Domain

As the body moves during exercise, several factors can interfere with signal quality. For example, while running or cycling, the movement of clothes hitting the body and/or the chest strap, and the electrodes' movement all result in interference to the ECG signal. Removing such interference from these motion artifacts is essential if the ECG signal quality is to be maintained. Typically, such motion artifacts will be

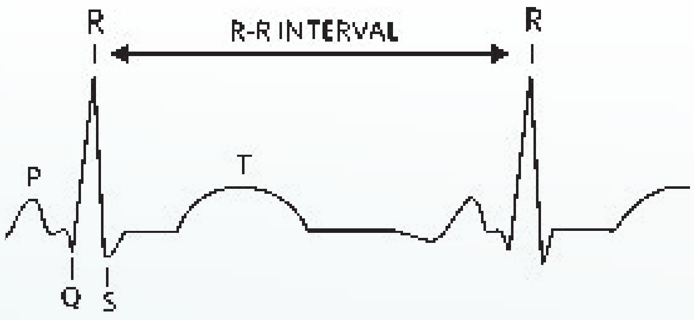


Figure 1: R-R interval in a typical ECG waveform

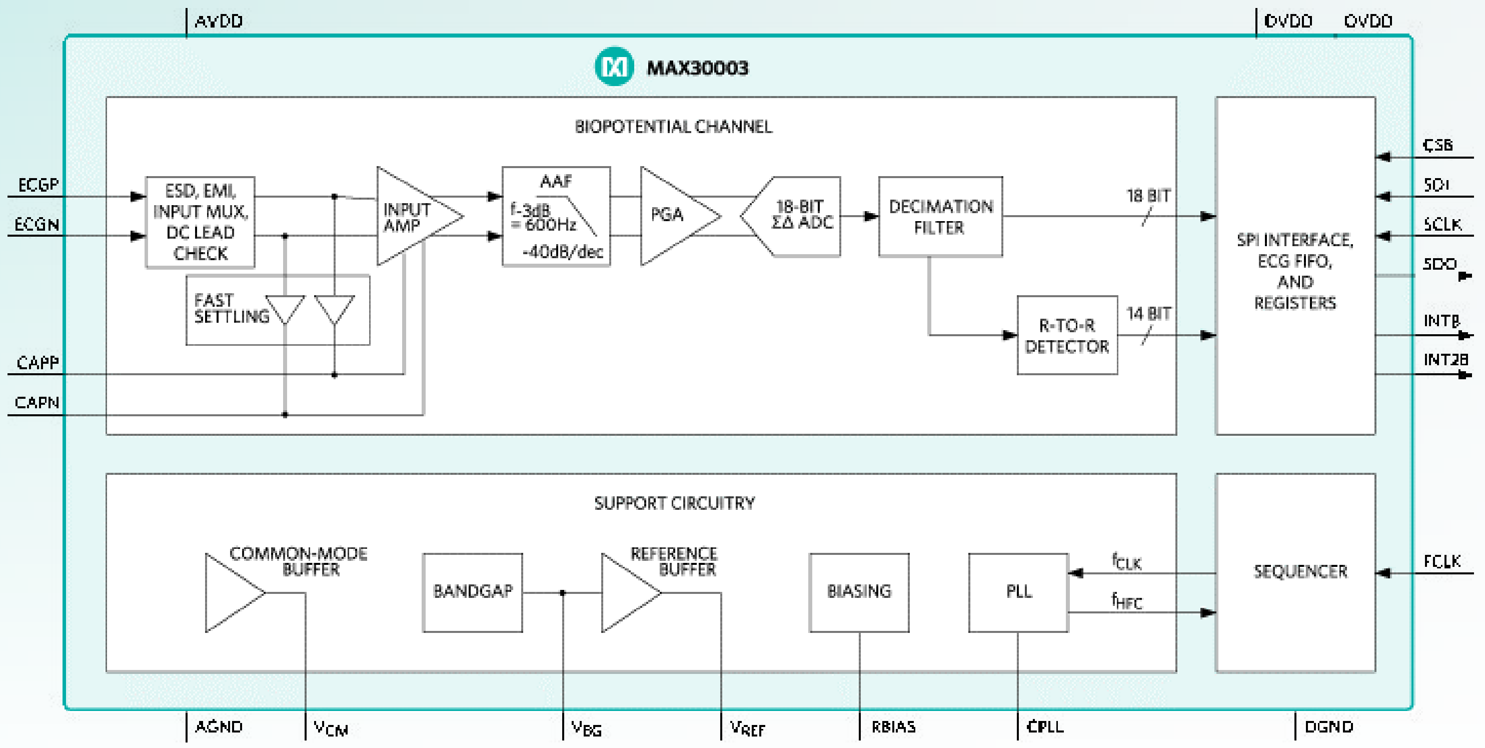


Figure 2: Biopotential AFE IC

present on, or common to, signals from both electrode pads, so the common-mode rejection ratio (CMRR) of the analog front-end needs to be as high as possible. Also, it should be observed that the heavier the sensor electronics, the more likely it is that the unit will bounce around when in use, creating additional motion artifacts.

Power Consumption

To keep the chest strap comfortable and practical, the form factor must be kept as nonintrusive as possible, resulting in minimal space available to host the electronics and power source (ideally a single coin-cell battery). This, in turn, necessitates extremely low-power consumption, as any heat generated could cause discomfort to the wearer while also reducing battery life.

An Integrated Solution

Balancing these key design considerations is challenging. Achieving the level of signal quality necessary for accurate readings while maintaining reliable, low-power operation in a small, durable, lightweight form factor is not trivial. In the following section, we propose a step-by-step approach to adding ECG measurement capability to a chest-worn wearable.

Step 1: The Analog Front-End

The analog front-end (AFE) required for detecting the ECG signal requires several different building blocks. Amongst others, these include an input amplifier with a lowpass filter, a programmable gain amplifier (PGA), and a high-accuracy analog-to-digital converter (ADC) with digital filtering options. Clearly, in a wearable device's confined space, discrete implementation of the AFE is not feasible. Therefore, an integrated approach is required. Some important specifications and features should be considered when selecting an integrated biopotential ECG AFE for a chest-worn wearable. It should ideally use a single input channel with very high series resistance (> 500MΩ) and high CMRR (> 100dB) for the reasons previously discussed. Along with ESD compliance (IEC61000-4-2) and EMI filtering, the integrated circuit (IC) should be able to detect if leads are connected (even in sleep mode) or if they have become detached from the wearer in normal operation while also having the ability to quickly recover from overvoltage conditions (such as defibrillation). This functionality must be provided with the lowest power consumption possible.

Figure 2 shows the functional block diagram for a fully integrated biopotential ECG analog front-end used in wearable designs that meet these requirements. An advantage of this device is that it provides ECG waveforms using one pair of electrodes (single-channel) and performs heart-rate detection in the same package. Similar ECG AFE ICs do not perform heart-rate detection. Instead, they rely on a microcontroller to perform the calculation, which typically consumes an extra 40μW of power. With a typical current consumption of only 150μW (almost 70 percent lower than similar parts), this AFE can be powered using a single coin-cell battery. It meets the IEC60601-2-47 ECG specification, making it suitable for clinical as well as fitness applications.

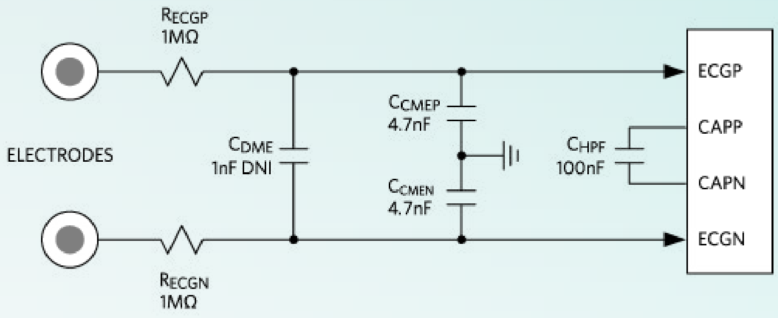


Figure 3: Input analog bandpass filter network

Step 2: Design a Motion Artifact Bandpass Filter

The removal or reduction of motion artifacts is best done in the analog domain, before converting the electrode signals into the digital domain. The primary way to do this is by reducing the bandwidth using highpass and lowpass filters. With the example IC, the single-pole highpass corner frequency can be set by connecting an external capacitor (C_{HPF}) to the CAPP and CAPN pins, as shown in **Figure 3**. The value used should set the highpass corner at 5Hz, especially for high-motion usage such as most sports and fitness applications. For clinical applications, this can be a lot lower, typically down to 0.5Hz or even 0.05Hz. When there is little to no movement, this provides better quality ECG information for diagnosis.

Figure 4 shows the analog bandpass bode plot for a chest-strap application.

A value of 100nF for C_{HPF} sets the highpass corner at a minimum of 5Hz but could go as high as 7Hz for high motion requirements. This would require a 68nF capacitor for C_{HPF}. The components set the lowpass filter to the left of the CAPP and CAPN pins, which are RECGP, RECGN (1MΩ), C_{CMEP}, and C_{CMEN} (4.7nF). This sets the common-mode lowpass corner at 34Hz, which is best for limiting shirt or clothing noise during dry starts. Limiting the bandwidth on the high end is also important to attenuate noise from static electricity and high-frequency signals. The impedance of the series resistors, RECGP and RECGN, should be limited, such that the root-sum-square (RSS) of the resistor thermal noise and the input noise of the ECG channel should not be much more than the input noise alone. The differential mode capacitor, CDME, is not used. Still, it is recommended to run an experiment to compare the performance of the common-mode lowpass filter to the differential mode lowpass filter since each design could have its own noise sources.

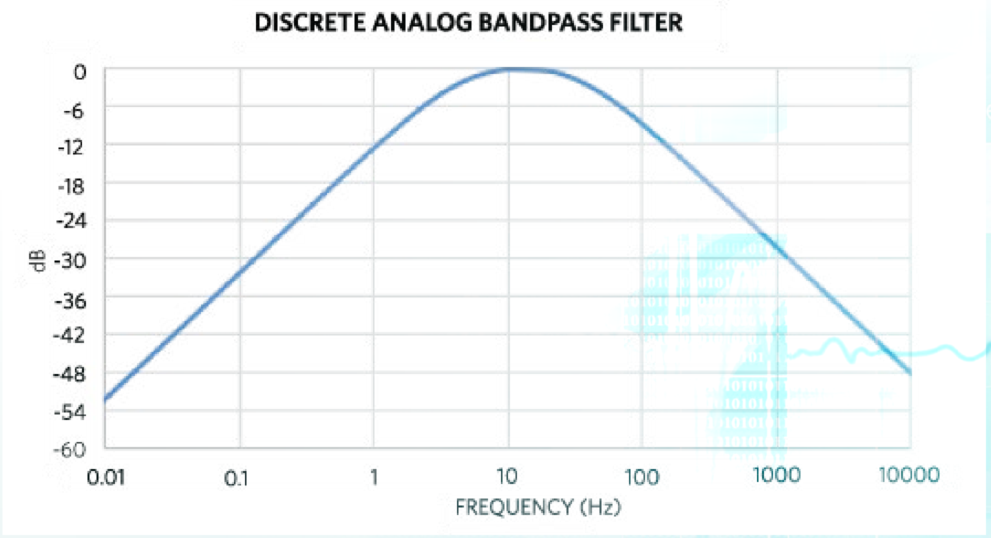


Figure 4: Analog bandpass filter bode plot for chest strap

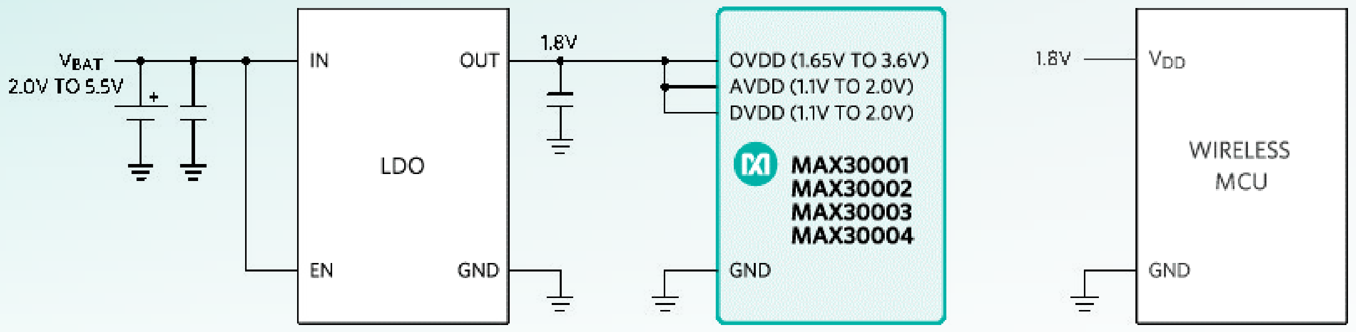


Figure 5: Simple linear dropout regulator power scheme

Recommendations for designing the PCB and selecting components:

- Use C0G-type ceramic capacitors, if possible, in the signal path to reduce signal distortion; for the ECG path, this includes C_{HPF}, C_{DME}, C_{CMEP}, and C_{CMEN}.
- Place discrete components close to the ECG IC and keeping traces as short as possible. The differential signals (E_{CGP}/E_{CGN}) keep the traces of equal length and symmetrical to maintain a high CMRR.
- Use a single ground plane under the device (AGND and DGND should not be split).

Step 3: Power Options

Depending on the battery type being used, there are several options for powering a complete wearable. The simplest option is to use a linear regulator (**Figure 5**) to create a common 1.8V DC rail from a coin cell that typically varies from 3.4V down to 2.2V. However, this approach is not particularly power efficient.

Although using a buck regulator instead of an LDO would improve efficiency, the best solution is to use a power management integrated circuit (PMIC), as shown in **Figure 6**.

The advantage of using this type of solution is that a PMIC can deliver separate power outputs for the microcontroller, analog front-end, and the digital interface.

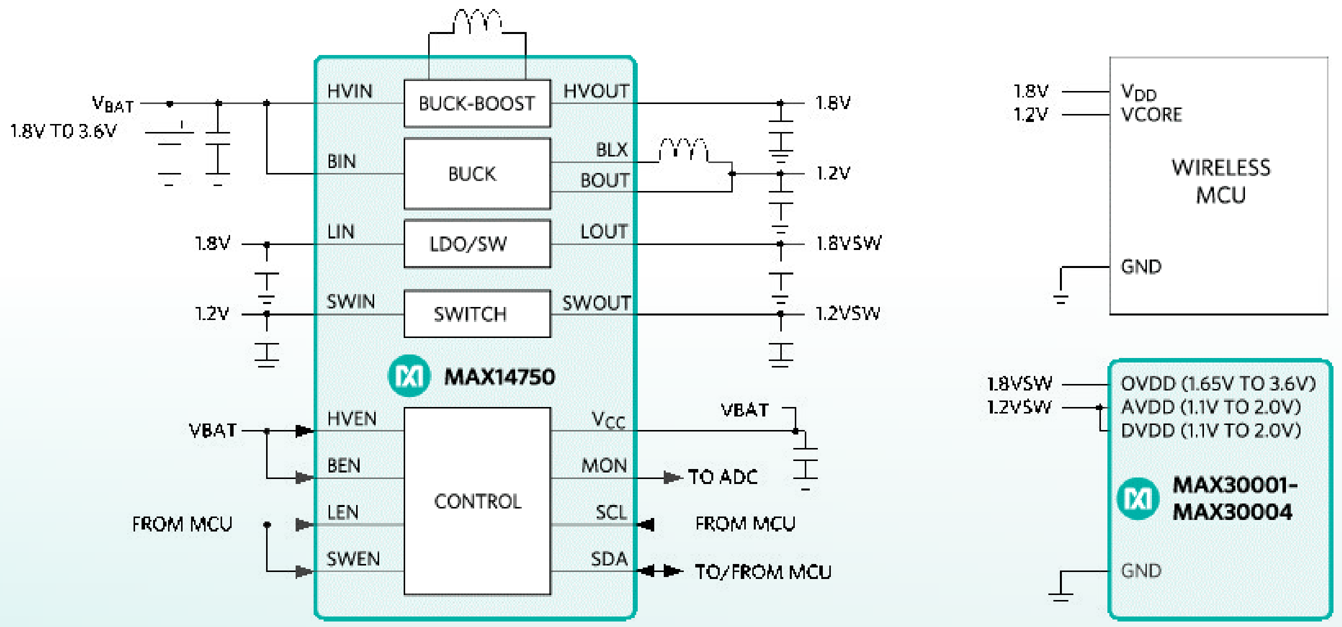


Figure 6: PMIC and a 3VDC coin-cell battery

Conclusion

We have reviewed the biopotential theory and how it is used to measure ECG using wet or dry electrodes connected to a chest-worn device. We then considered the challenges in conditioning the ECG signal to allow accurate measurement with minimal power consumption. Finally, we proposed a composite solution consisting of a biopotential AFE IC, a discrete analog filter, and a PMIC. This solution profiles the components required to quickly and easily integrate ECG functionality into a chest-worn health and fitness wearable. ●

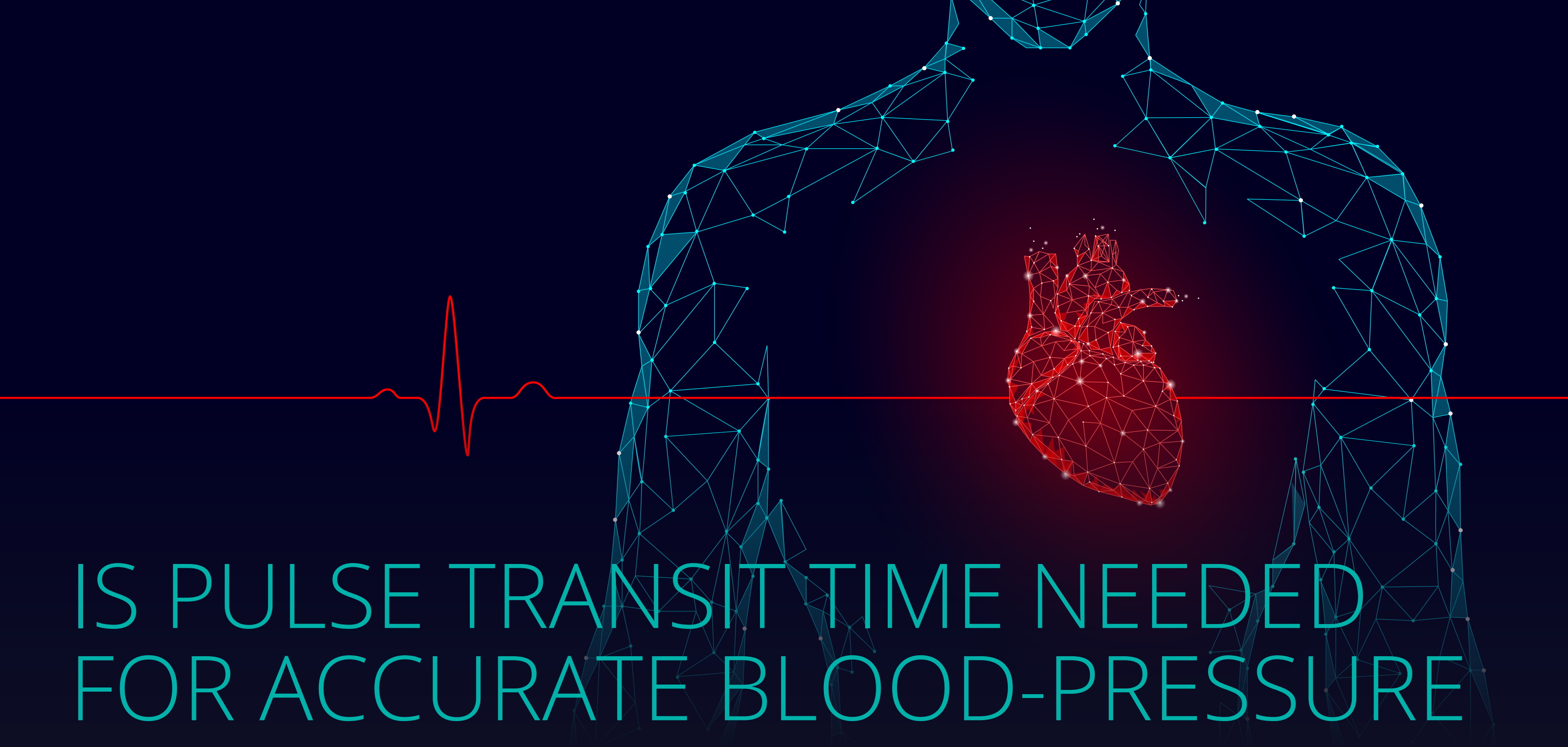
Wearable devices provide medical professionals with the ability to monitor patients over an extended period outside the hospital environment.

MAX8614X PULSE OXIMETER & HEART-RATE SENSOR

LEARN MORE

MAXREFDES103 HEALTH SENSOR PLATFORM

LEARN MORE



IS PULSE TRANSIT TIME NEEDED FOR ACCURATE BLOOD-PRESSURE MONITORING FROM WEARABLES?

ANDREW BURT, *Executive Business Manager, Industrial & Healthcare Business Unit, Maxim Integrated*

Cardiovascular disease has been the leading cause of death for Americans since 1920. Also alarming is the fact that related medical and indirect costs totaled $555 billion (USD) in 2016 are expected to reach $1.1 trillion (USD) by 2035, according to the American Heart Association.

It's a good thing that wearables—with their ability to provide continuous, real-time monitoring of vital signs—are getting increasingly sophisticated.

Designed with advanced sensors and algorithms, along with powerful yet efficient processors, wearable devices that monitor parameters such as heart rate, blood-oxygen levels, sleep quality, and stress levels are already on the market.

Research and work are underway on products for the non-invasive detection or screening of such conditions as diabetes, breast cancer, atrial fibrillation, and ultraviolet exposure. Wearables can empower people to more closely manage chronic conditions and be more proactive about addressing previously undetected health issues (**Figure 1**).

Let's consider blood pressure—an important indicator of cardiovascular health and a vital sign that must be managed. The American Heart Association recommends that people with high blood pressure engage in home monitoring to evaluate whether their treatments are effective. Plenty of home-monitoring tools are already available. However, most are cuff-based and require dedicated time to capture the measurement (and these measurements aren't made continuously).

Fortunately, wrist-based devices that monitor blood pressure continuously and non-invasively are becoming available. For instance, Omron Healthcare's HeartGuide is the first wearable blood-pressure monitor, and it has the approval of the U.S. Food and Drug Administration (FDA). HeartGuide is essentially a smartwatch and uses the oscillometric cuff method, the standard for medical-grade personal blood-pressure measurement. An accompanying app

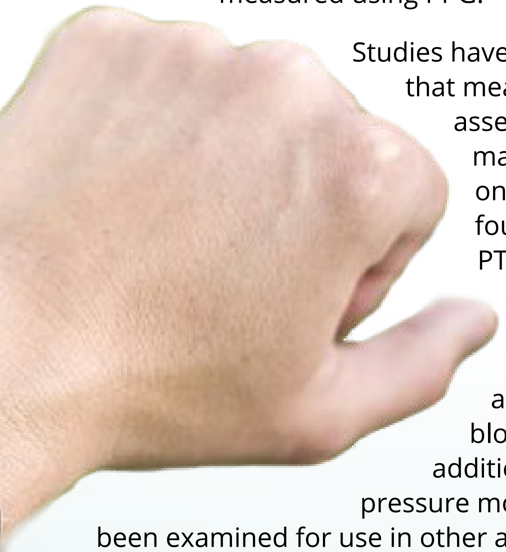
Figure 1: Wearables, such as smartwatches, monitor vital signs, empowering users to be more proactive about managing their health and wellbeing

provides insight on readings and allows sharing with the user's doctor.

Wrist-Based BP Monitoring Challenges

Capturing precise blood-pressure readings from the wrist is challenging because the arteries are more narrow and not as deep under the skin as upper-arm arteries. Also, the arm and wrist must be at heart level to capture a correct reading. Pulse transit time (PTT), which denotes the time that it takes a pulse pressure waveform to propagate through the arterial tree's length, provides a means to measure blood pressure. This pulse pressure waveform occurs when blood is ejected from the left ventricle, and it moves with a greater velocity than the forward movement of the blood itself. In addition to a measurement of blood pressure, PTT also provides an indicator of arterial stiffness.

PTT can be derived from calculations on electrocardiogram (ECG) and photoplethysmography (PPG) signals. It is based on tightly defined characteristics of the shape of the pressure pulse waves in blood vessels. ECG provides a measurement and a graphical representation, in terms of time, of the electrical signals associated with heart muscles. PPG provides an optical measurement of the volumetric change of blood in the tissue during the cardiac cycle. Optical sensors that utilize PPG and ECG are used in wearables to measure heart rate. Measuring PTT involves calculating the time between the ECG's R-peak and a reference point on the pressure pulse wave measured using PPG.



Studies have shown, however, that measuring PTT alone to assess blood pressure may not be sufficient. In one study, researchers found that integrating PTT, heart rate, and a previous blood-pressure estimate results in a more accurate current blood pressure value. In addition to general blood-pressure monitoring, PTT has been examined for use in other applications, such as a means to detect sleep ailments via measurement of respiratory effort or indicate the onset of low blood pressure during spinal anesthesia.

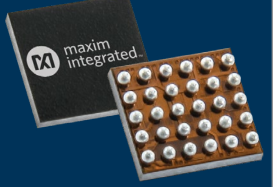
Maxim offers biosensor modules and biopotential analog front-ends (AFEs) for health wearables. One example is the MAX30001 ultra-low-power, single-channel integrated biopotential, and bioimpedance AFE. The solution consists of a single biopotential channel that provides ECG waveforms and heart-rate and pacemaker edge detection and a single bioimpedance channel that measures respiration.

Examinations have been undertaken that utilize a two-chip solution consisting of an ECG AFE and a PPG sensor to measure, via a wearable, pulse arrival time (PAT), equal to the sum of PTT and the pre-ejection period. PAT is simpler to measure than PTT and has been proposed as a PTT surrogate. However, subsequent studies have shown that, because of accuracy issues, PAT is not an ideal replacement for PTT as a blood-pressure marker. However, it could potentially indicate wide trend blood-pressure changes in some cases.

The interesting point here is, a two-chip solution can measure PAT via a wearable. Now, imagine the potential solutions that could emerge by adding the capability to calculate PTT from the wearable form factors' data. Applying sensor fusion to bring together data from multiple sensors and artificial intelligence (AI) to identify patterns and opportunities for action opens up many more possibilities. The sophisticated data analysis would happen in the background. For the wearer, he or she simply has a device that provides continuous monitoring, comfortably and conveniently. ●

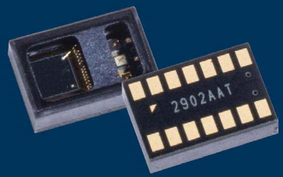
MAX30001 BIOPOTENTIAL ANALOG FRONT-END SOLUTION

LEARN MORE



MAXM86161 OPTICAL BIO-SENSOR

LEARN MORE



REALLY SMART WEARABLES CUSTOMIZE THEIR POWER SUPPLY

Figure 1: Olympic flag

This design solution shows how an alternative power management integrated circuit allows optical sensors used in health and fitness wearables to save power with dynamic voltage scaling techniques, particularly when they're not being used to their full potential.

Introduction

"Citius, Altius, Fortius," or "Faster, Higher Stronger" in its more recognizable form, is the motto of the Olympic Games (**Figure 1**). Of course, not everyone who wears a health and fitness monitor to measure and record vital signs is an aspiring Olympic athlete. For more sedentary individuals, such as your humble scribe, an occasional workout might be the only time a wearable is pushed anywhere close to the level of performance for which it was designed. In addition, everyone (even Olympic athletes) needs rest and sleep. Ultimately, this means that for a large part of their lifespan, wearables are often required to function well below their potential. Is it reasonable to assume that during this time they are consuming as little power as possible to conserve battery life? This might not always be the case.

In this design solution, we will review the operation of optical sensors used in health and fitness monitors and show how they can waste power even in less demanding conditions. We will then present a power management integrated circuit (PMIC) that does more than simply reduce current to save even more power under these circumstances.

Optical Sensing

Optical sensors are commonly used to measure health indicators such as heart rate and blood oxygenation (SpO_2). These measurements are based on a technique called photoplethysmography (PPG). A PPG signal is obtained by illuminating skin using a light-emitting diode and detecting changes in the intensity of the reflected light (**Figure 2**) using a photodiode that generates a current proportional to the amount of received light.

Making reliable measurements to provide accurate heart rate (or SpO_2) information is challenging and affect several different factors.

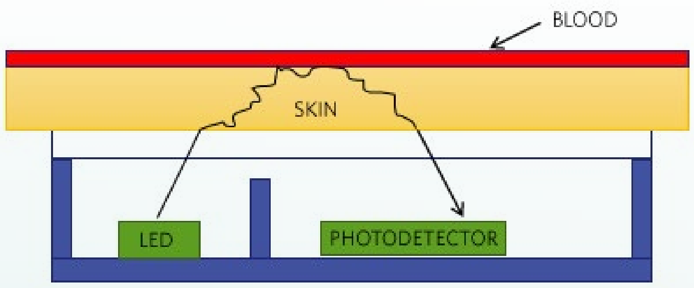


Figure 2: Photoplethysmography (PPG) using LED and photodiode

Measurement Challenges

Making reliable measurements to provide accurate heart rate (or SpO_2) information is challenging and affect several different factors. The first is motion. For example, when a device wearer is asleep (or inactive), LED light pulses of lower frequency and amplitude are required compared to when the wearer is engaged in high-intensity exercise. Ambient lighting conditions have a similar impact. In a dark environment, fewer and lower intensity LED light pulses are required than in bright or sunny conditions. User conditions requiring brighter and more frequent LED pulses drain more power from the battery. In a single day, a user might encounter varying lighting conditions and engage in different activity levels. Ideally, an optical sensor should only use as much power as is required for a given use case. Since a typical user will spend between 7 and 9 hours a day sleeping, reducing power consumption is important. To date, power-saving efforts have primarily focused on reducing current consumption. However, in the following section, we consider the limitations of this approach.

LED Circuit

A simplified example of a circuit used to drive the LED in a wearable device is shown in **Figure 3**. Because some class of lithium battery (which can range from 3.2V to 4.35V) is used as a power source, a boost circuit is used to increase the circuit supply voltage to 5V.

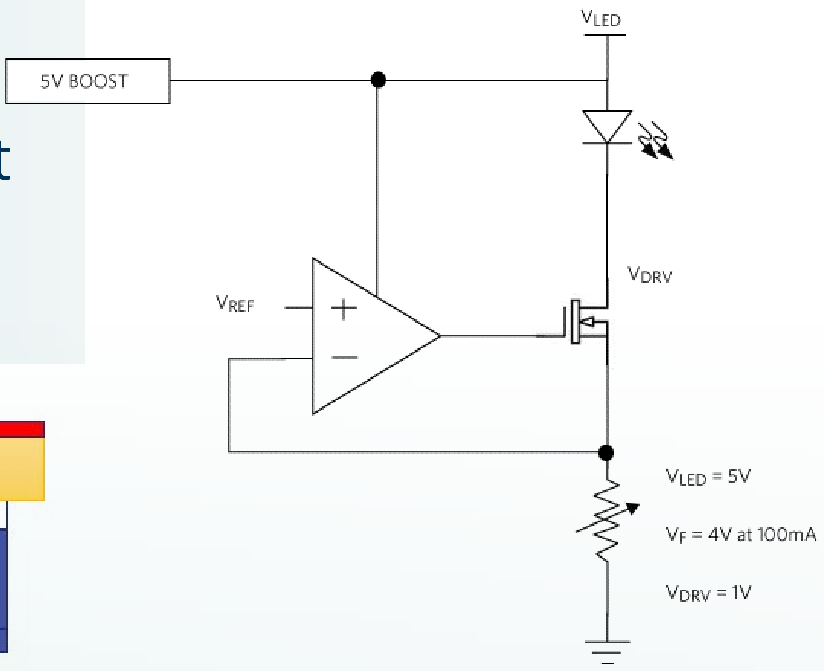


Figure 3: High LED current circuit

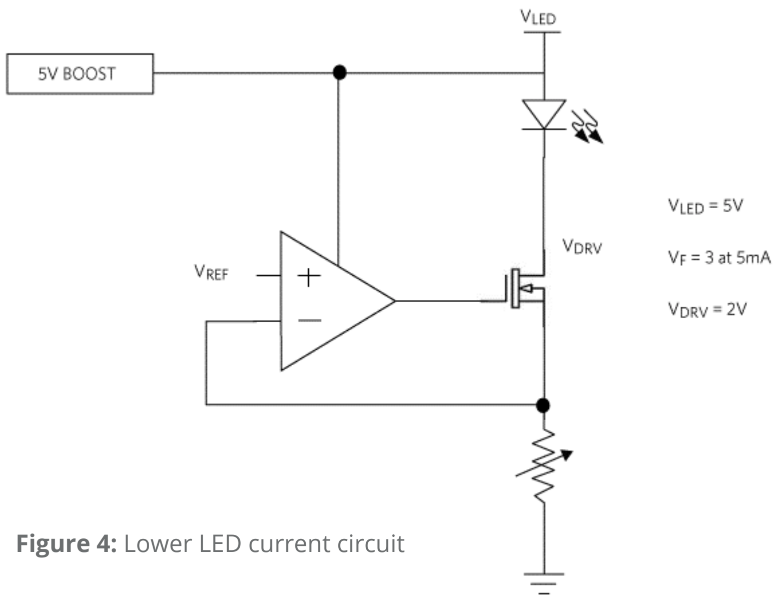


Figure 4: Lower LED current circuit

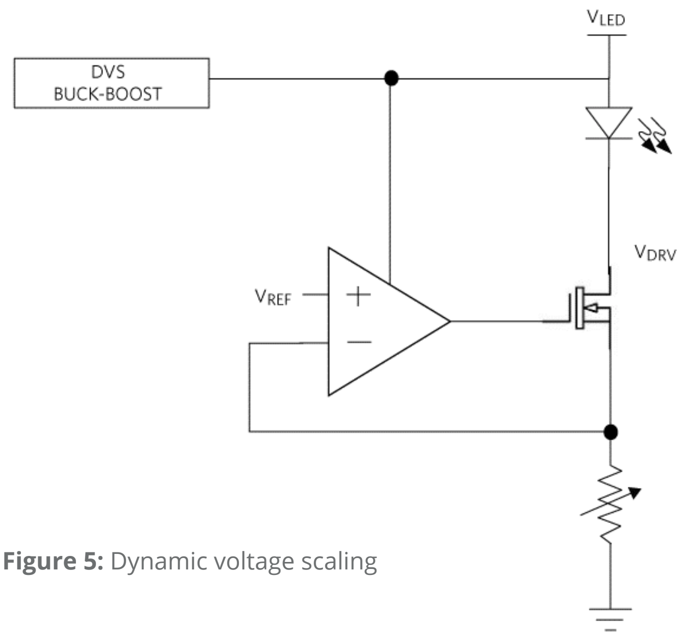


Figure 5: Dynamic voltage scaling

In this example scenario (when the user might be engaged in strenuous exercise), a high LED current is required (such as 100mA) with a voltage drop across the LED of 4V, allowing a drain voltage (V_{DRV}) of 1V for the transistor to maintain operation in the linear region. However, under more favorable measurement conditions (for example, when the user might be asleep), less LED current is required (5mA), and the circuit condition in **Figure 4** exists. Because of the lower current, the voltage drop across the LED is reduced to 3V, leaving the voltage across the transistor to 2V. However, the lower current means that the transistor does not need this voltage level to maintain linear operation, Instead, it requires a V_{DRV} of only 0.2V.

This means an excess voltage of 1.8V and an effective power wastage of 9mW (such as 1.8V x 5mA) in this circuit condition.

Dynamic Voltage Scaling

A preferable scenario would be when the supply voltage provided by the boost circuit changes in response to LED current. The transistor always has sufficient (but not excess) voltage to maintain linear operation. This technique, referred to as Dynamic Voltage Scaling, can be implemented by simply using a lookup table (**Table 1**) to determine the output voltage required by the boost converter, (**Figure 5**). This can provide significant power savings in less demanding measurement conditions.

Table 1. DVS Lookup Table

LED Current (mA)	V_{DRV} (mV)
128	950
96	480
64	320
32	160

For this circuit:

$V_{DVS} = V_{LED} + V_{DRV}$

Many wearable systems manage LED current to reduce power consumption. DVS now provides the option also to manage voltage for even greater power savings. The PMIC in **Figure 6** features a buck-boost converter that uses DVS on its output voltage.

This PMIC also features three buck regulators, three low-dropout (LDO) linear regulators, and a buck-boost regulator, providing up to seven regulated voltages, each with an ultra-low quiescent current. This allows it to power multiple sensors (including the MAX86140 optical AFE) and a microcontroller in a typical design for a wearable device (**Figure 7**). It is available in a 56-bump, 0.4mm pitch, 3.37mm x 3.05mm wafer-level package (WLP).

Summary

The power consumed by an optical sensor in a health and fitness wearable varies widely in response to changing user conditions. Until now, current consumption has been the focus of power-saving techniques used in these devices. Although effective, it neglects the other part of the power equation, namely voltage. As we have shown, power is wasted in an optical sensor that uses a fixed supply voltage in all circumstances. In this design solution, we presented a PMIC that uses a Dynamic Voltage Scaling technique to provide the appropriate (but not excessive) voltage required to supply the current needed by an optical sensor's LED for a given use case. The ability to dynamically adapt voltage in response to current demand offers a new degree of freedom for power-saving approaches in wearable devices. ●

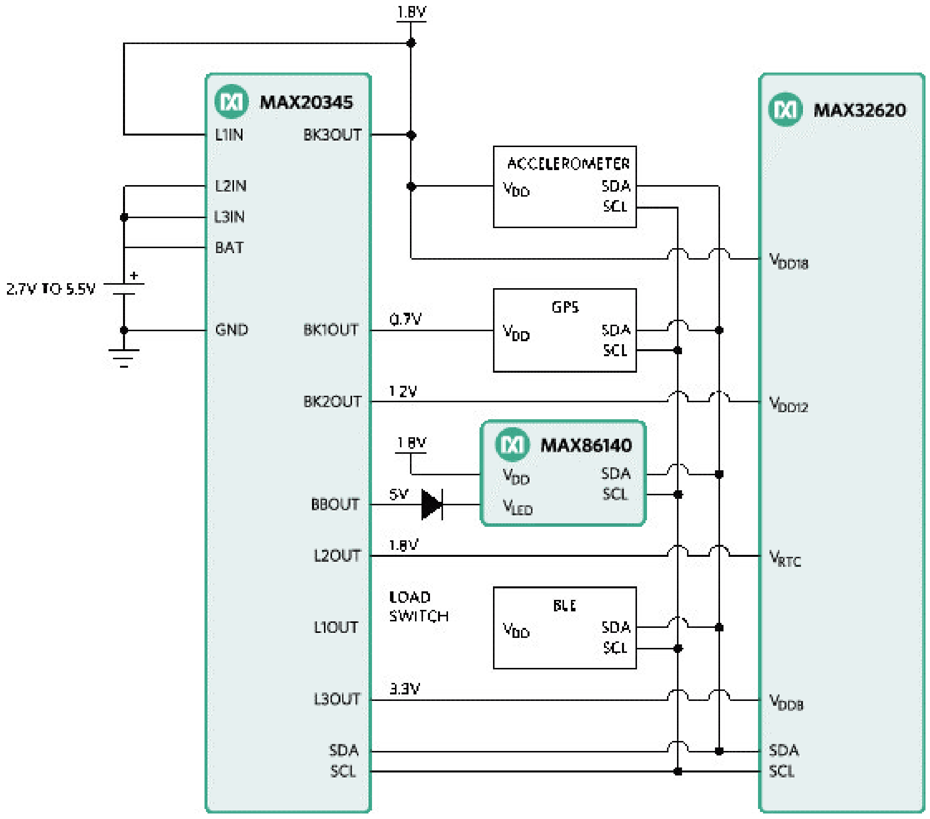


Figure 7: MAX20345 typical application circuit

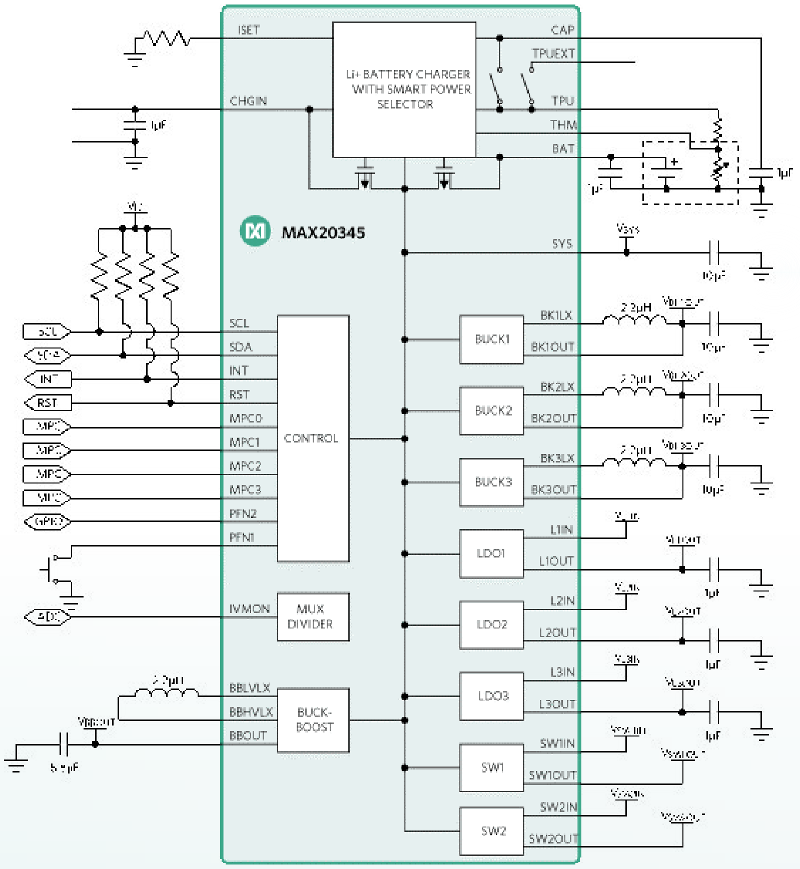
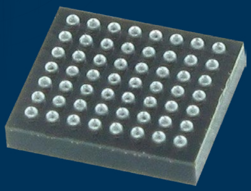


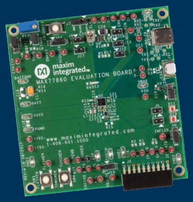
Figure 6: MAX20345 PMIC for wearables

MAX20345 POWER MANAGEMENT IC (PMIC)



LEARN MORE

MAX77860 EVALUATION KIT



LEARN MORE

Mouser stocks the widest selection of the newest products

maxim integrated™

mouser.com/maxim

Worldwide leading authorized distributor of semiconductors and electronic components

MOUSER ELECTRONICS

Mouser and Mouser Electronics are registered trademarks of Mouser Electronics, Inc. Other products, logos, and company names mentioned herein, may be trademarks of their respective owners.