Esperto Watch
Software Design Description (SDD)

Revision History

Revision	Date	Description	By
2.1	29-08-2018	Version 2 Initial draft	Daniel De Sousa

1 Project Introduction

The Esperto Watch is a wearable platform in the form of a smart watch which is powerful enough for researchers and developers to use in their professional work yet simple enough for beginners to learn with. The platform is equipped with numerous sensors to detect biometric data such as a user's heart rate or step count, wireless communication, and a companion mobile and web application to track user metrics.

The watch firmware is fully customizable, allowing developers to build onto our custom algorithms, build their own, or add features such as GPS, activity tracking, or Wi-Fi.

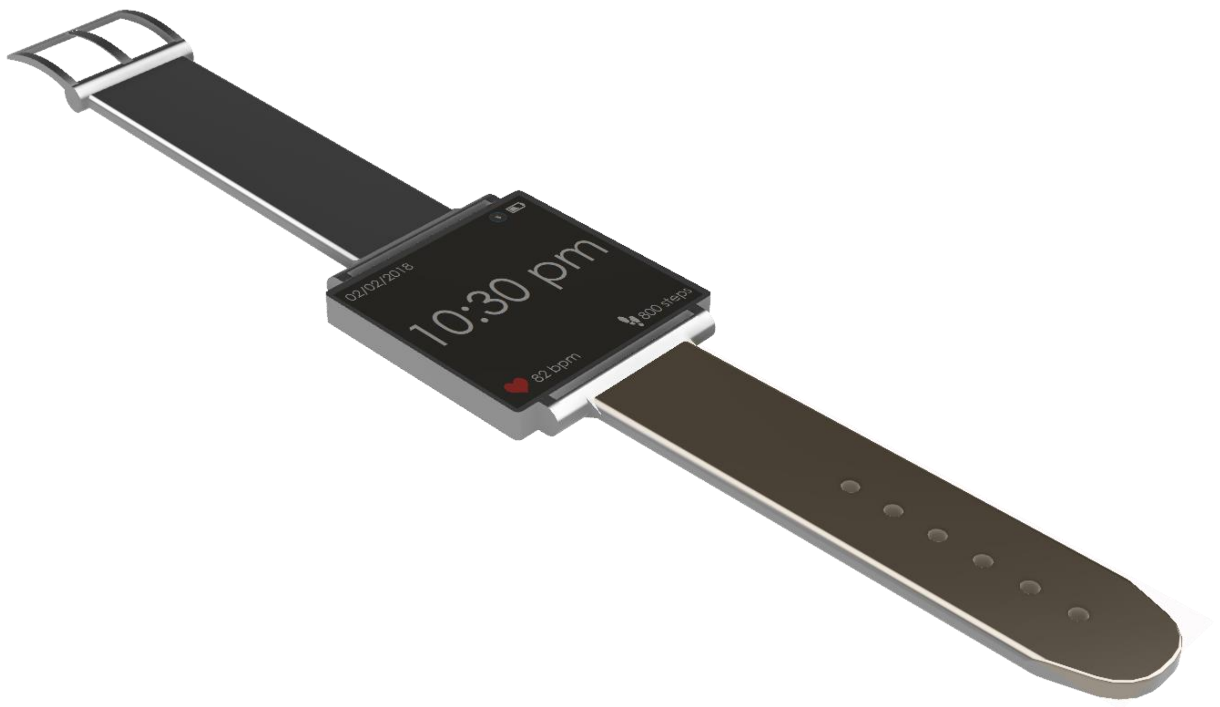

Figure 1: The Esperto Watch

2 System Overview

2.1 Feature Overview

The Esperto Watch supports the following features:

- Time tracking, heart rate detection, step detection, SpO2 calculation
- Firmware upgrades / debugging over micro USB (also available over SWD)
- Battery charging
- FRAM user data storage (non-volatile)
- Bluetooth Low Energy communication for use with a mobile application
- Data synchronization between device and mobile application (heart rate, step count, SpO2)
- Mobile notifications (texts, calls)
- Displaying time, device status, mobile notifications, heart rate, and number of steps taken

2.2 Block Diagram

These features are supported by implementing specialized sensors in the hardware and developing algorithms to obtain data from these sensors, filter the data, and convert the raw data into user metrics. A general system block diagram can be seen in **Figure 2.**

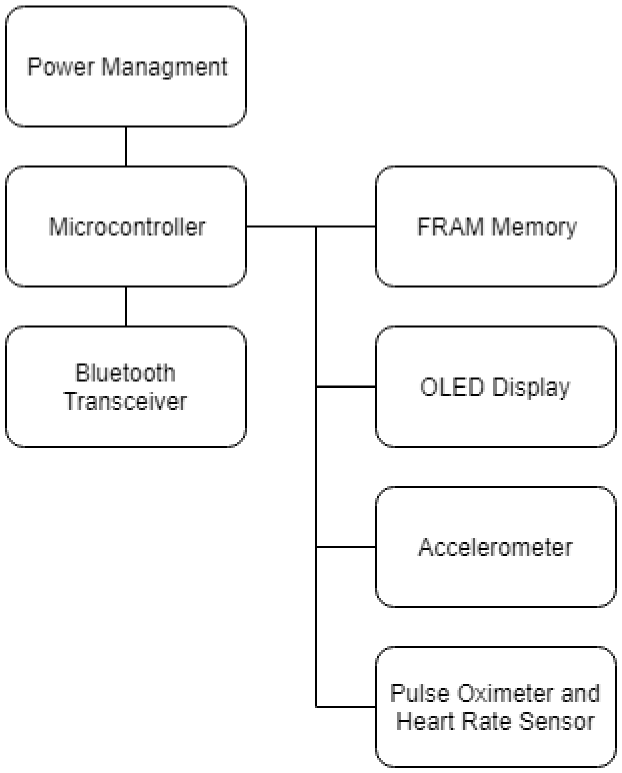

Figure 2: Esperto Watch General System Diagram

3 Main Application

3.1 Overview

The main application is run on a ATSAMD21G18 chosen for its low-cost, low power consumption and wide support. The SAMD21 is a ARM Cortex M0+ flash-based controller running at a CPU speed of 48 MHz. It has the following features:

- 256 KB of embedded flash and 32 KB of SRAM
- Low power consumption at less than 70 uA/MHz
- 6 serial communication modules (SERCOM) supporting interfaces including SPI, USART, I2C
- Embedded full speed USB device and host capable of up to 12Mbps throughput
- 14 12-bit, 350ksps ADC channels and 1 10-bit DAC channel
- Internal 32-bit Real Time Clock (RTC) with calendar functionality
- 256 channel Peripheral Touch Controller (PTC)
- 12 Direct Memory Access controllers (DMAC)
- CRC-32 generator
- Watchdog Timer (WDT)

The microcontrollers firmware can be upgraded over USB or SWD. However, for USB upgrades, a custom bootloader must be flashed over the SWD interface. The application is supported by ASF (Atmel's Software Framework) which provides HAL support for the NVIC controller, timers, pin multiplexing, WDT, and other core system features.

At bootup, the Esperto Watch application is responsible for initializing all of the communication interfaces including USB, I2C, and SPI, peripherals including the accelerometer, heart rate sensor, FRAM, and the Bluetooth transceiver.

Accelerometer setup includes resetting certain registers, setting interrupt levels, enabling the accelerometer and gyroscope, and enabling certain features related to the pedometer and the LPF (Low Pass Filter). Setting up the heart rate sensor involves setting the speed of the I2C interface (100 KHz) and enabling the sensor while initializing the FRAM involve confirming manufacturing and configuration settings. Finally, setup is complete by initializing the Bluetooth transceiver. This step involves initializing the HCI or Host Controller Interface, resetting the module, setting its address, initializing the GAP (Generic Access Profile) and BLE stack, starting the UART service, and the modules output power.

Other peripherals such as the RTC and the display are also enabled and configured during this setup.

During operation, the application is responsible for processing incoming BLE packets, updating the display, writing data to the FRAM, managing power, performing step and heart rate detection, calculating the users SpO2, and sending out data packets to the paired mobile device. The application uses a 1Hz timer interrupt to track when to write to the display, write to memory , send data to the mobile application, clear notifications sent from the mobile application, and manage its power settings.

3.2 Heart Rate Detection

A user's heart rate and SpO2 or blood oxygen saturation is calculated using sensor readings from the MAX30102 High-Sensitivity Pulse Oximeter and Heart-Rate Sensor.

3.2.1 Algorithm

Heart rate is detected by performing a peak detection on infrared light data obtained from the sensor. The sensor emits infrared light onto the user's skin where some is reflected based on the concentration of hemoglobin in the blood which will vary during the different stages of heart contraction. The amount of ambient infrared light reflected is detected by a photodiode and then filtered by the MAX30102. A typical infrared light waveform during multiple heart contractions can be seen in **Figure 3.**

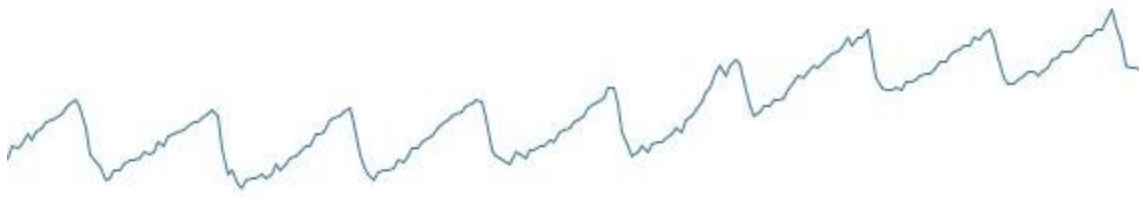

Figure 3: Sample Infrared Waveform During a Heart Beat

When a peak is detected, the time is recorded in milliseconds. The instantaneous heart rate can be calculated by determining the time between two adjacent peaks. A circular FIFO buffer is implemented to store the latest instantaneous heart rates. To determine an average heart rate, all the values are summed and then divided by the size of the buffer. This is the value written to the display and memory and sent to the mobile application when connected. Using this buffer allows for a running average to be calculated and reduces the impact caused by accidental heart rate calculation outliers. To reduce the number of invalid peaks in the infrared data, the data is first passed through two filters described in the following section.

3.2.2 Filtering

The application uses two digital filters to remove unwanted high frequency data and simplify the peak detection algorithm mentioned in the previous section.

The first filter is a Single-Pole Recursive Low Pass Filter which uses a moving average to smooth out the infrared data obtained from the sensor. This is an example of an IIR or Infinite Impulse Response filter as its impulse response will always be non-zero. This moving average filter was implemented with the following recursive coefficients: a_0 equalling 1/16 and b_1 equalling 15/16. For a single-pole filter, the coefficients must follow the following relationship:

$$\begin{aligned} a_0 &= 1-x \\ b_1 &= x \end{aligned}$$

where *x* is the amount of decay exhibited by the filter. a_0 is multiplied by the incoming signal and b_0 is multiplied by the running average as seen in the following formula:

$$x[n] = b_1 x[n-1] + a_0 u[n]$$

Where $x[n]$ is the output of the filter and $u[n]$ is the infrared input to the system.

This filter will separate an input into high and low frequency components as seen in **Figure 4.**

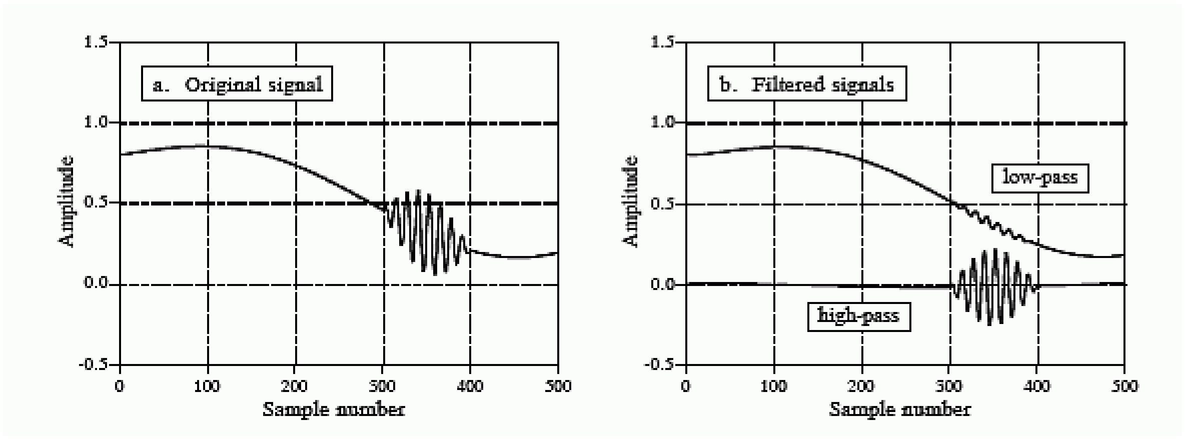

Figure 4: Input and Output of a Typical Recursive Filter

This filter was implemented with fixed point arithmetic for multiplying and dividing the inputs and outputs instead of typical long division and multiplication as the SAMD21 does not have an internal FPU or Floating Point Unit. This decision results in less clock cycles and drastically improves performance of the filter.

The output of this recursive filter is passed into the next filter digital filter, a 12th order Low-Pass Finite Impulse Response (FIR) Filter. This is an FIR filter as its impulse response is of finite length and converges to 0 over time. This filter uses a circular FIFO buffer to store the latest outputs of the previous filter. The output of this filter is the weighted sum of its *N* latest inputs. The filter may be described using the following formula, where b_i is the weight of the *i*th sample, $x[n-i]$ is one of the samples, and $y[n]$is the output of the filter:

$$\begin{aligned} y[n] &= b_0 x[n] + b_1 x[n-1] + \cdots + b_N x[n-N] \\ &= \sum_{i=0}^{N} b_i \cdot x[n-i], \end{aligned}$$

A general diagram of an FIR filter can be seen in **Figure 5**.

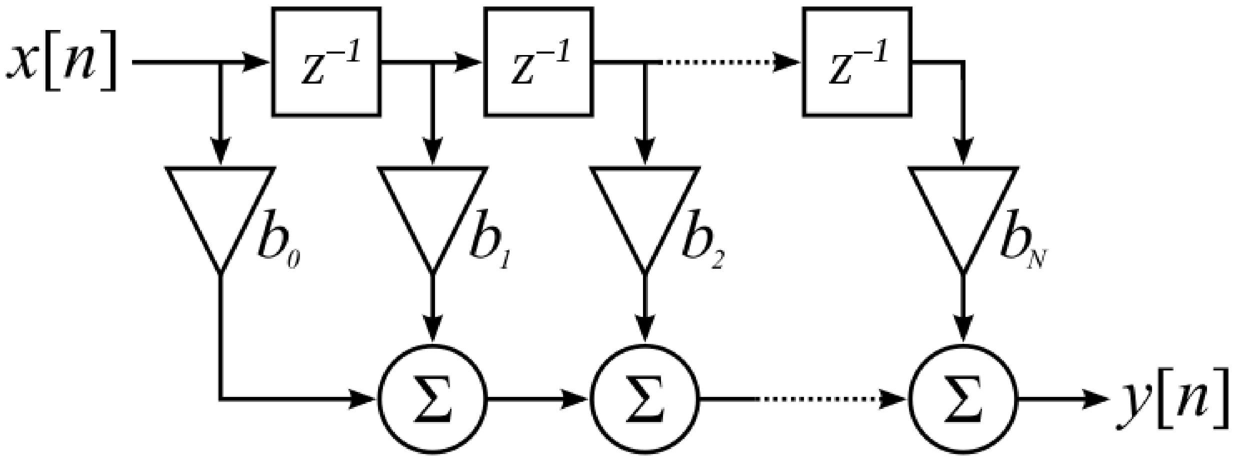

Figure 5: General FIR Filter

In the case of this application, the following filter parameters were chosen:

- Filter order (number of distinct coefficients): 12
- Number of taps/samples used (N): 23

The FIR coefficients are based on a normal distribution curve and are generated from a filter design program based on the intended cut-off frequencies of the filter. Plotting the coefficients results in the curve seen in **Figure 6** where the first signal tap**,** $x[n]$, is multiplied by the first coefficient in the series and the last, $x[n-N]$, is multiplied by the last coefficient. The multiplicands are then all summed to produce the output of the filter.

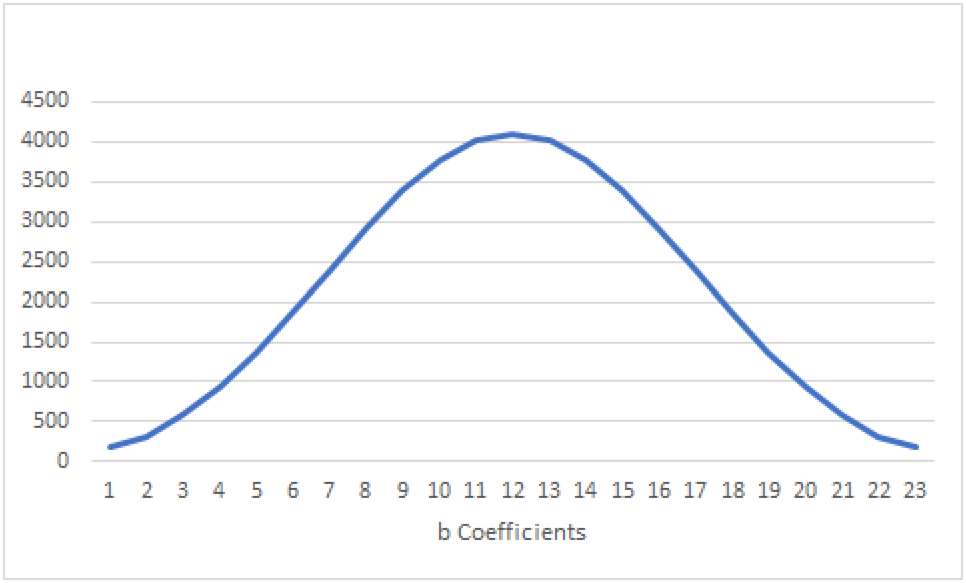

Figure 6: FIR Coefficients

The coefficients may be used to represent the transfer function of the filter (the Z transform of the impulse response). The transfer function can be written as:

$$H(z) = 172 + 321\,z^{-1} + 579\,z^{-2} + 927\,z^{-3} + 1360\,z^{-4} + 1858\,z^{-5} + 2390\,z^{-6} + 2916\,z^{-7} + 3391\,z^{-8} + 3768\,z^{-9} + 4012\,z^{-10} + 4096\,z^{-11}$$

or as:

$$H(z) = (172\,z^{11} + 321\,z^{10} + 579\,z^{9} + 927\,z^{8} + 1360\,z^{7} + 1858\,z^{6} + 2390\,z^{5} + 2916\,z^{4} + 3391\,z^{3} + 3768\,z^{2} + 4012\,z + 4096) / z^{11}$$

The magnitude of this transfer function can be plotted as a surface where the x-axis is the real component and the y-axis is the imaginary component of the transfer function as seen in **Figure 7**. The green curve is the image of the unit circle, e^{jw}, which is the same curve as seen in **Figure 6.** Furthermore, there are 12 downward asymptotes which represent the zeros of the transfer function, *H(z),* which shape the green curve and determine the frequency response of the system.

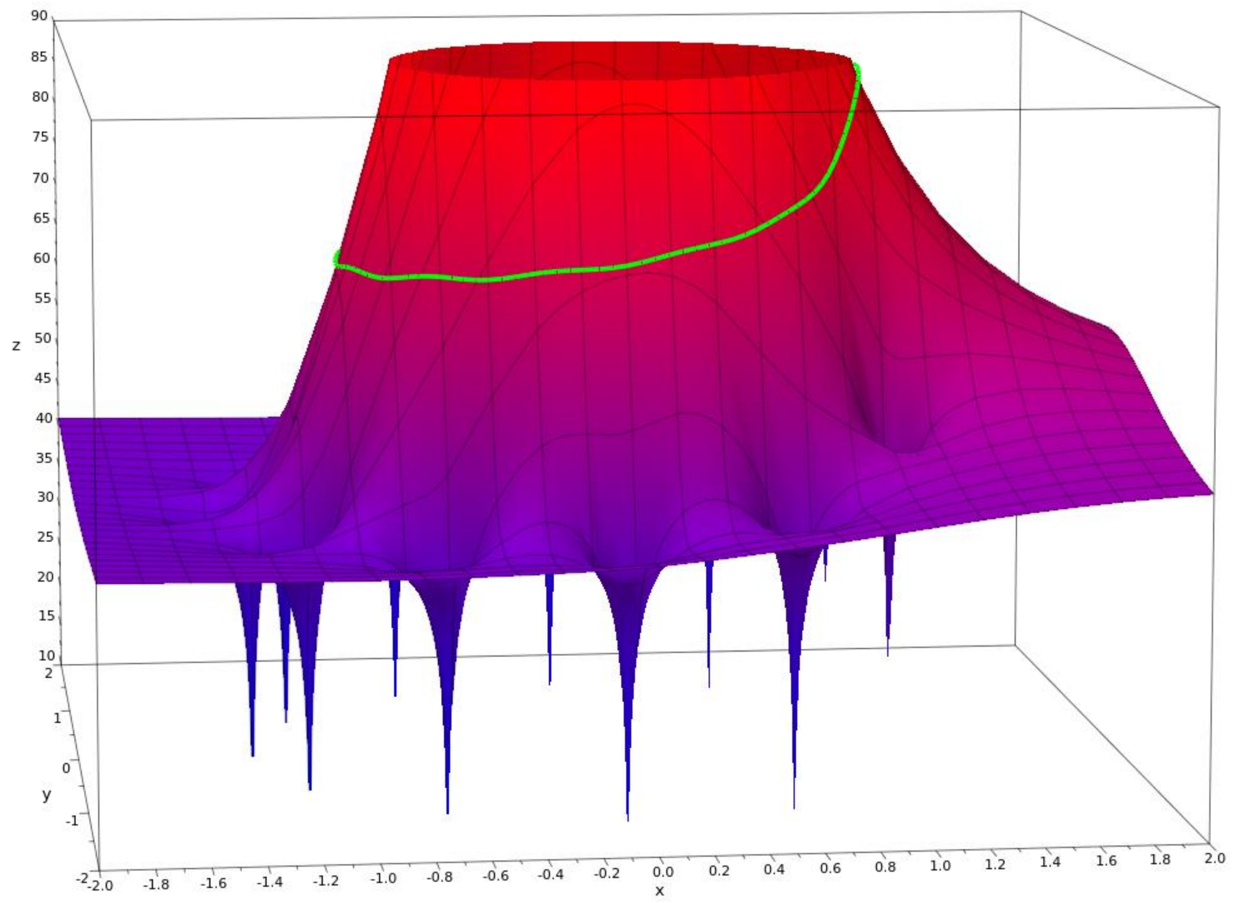

Figure 7: 3D Plot of H(z)

The output of this filter is then used in the peak detection process mentioned above.

3.3 SpO2 Calculation

SpO2 is calculated through a process called pulse oximetry. This process is used to determine a user's blood oxygen saturation, which is based on the concentration of hemoglobin, a protein bound to oxygen, and deoxyhemoglobin, hemoglobin without oxygen. SpO2 can be used to diagnose whether a user is suffering from respiratory diseases and whether their body is receiving supplementary oxygen.

3.3.1 Algorithm

To calculate SpO2, infrared and red light reflection values are obtained from the MAX30102 pulse oximeter. Two sources are used as oxygenated ($Hb0_2$) and deoxygenated (Hb) hemoglobin absorb different wavelengths. Hb has a high absorption rate at 660 nm wavelength (red light) while $Hb0_2$ absorbs light pertaining to wavelengths of 940 nm (infrared). **Figure 8** outlines the light absorption of the two proteins.

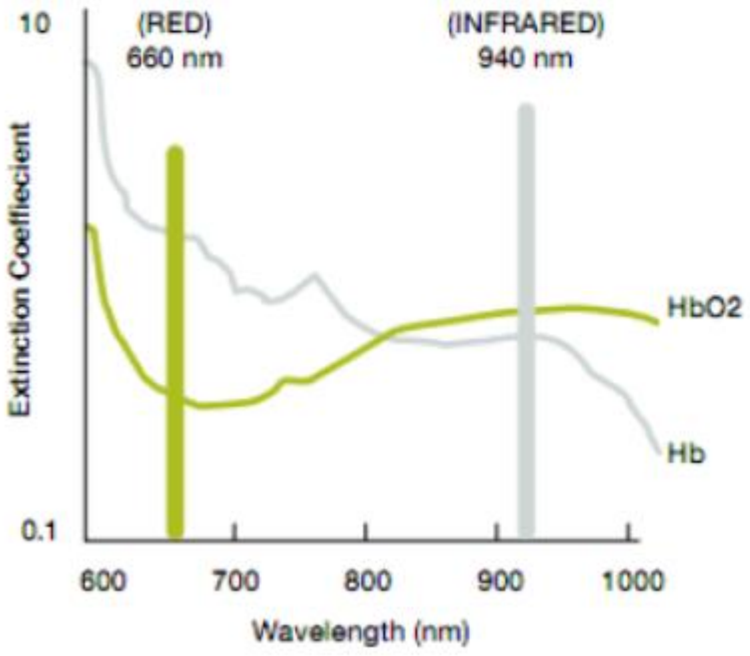

Figure 8: Light Absorption of Hemoglobin and Deoxyhemoglobin

A photodetector on the MAX30102 obtains analog light reflection values from the user. The reflected light is split into a DC and AC component. The DC component represents the constant light absorption by the user's tissue, venous blood, and non-pulsatile arterial blood where the AC component represents the amount of light reflected by the pulsatile arterial blood. A DC and AC component exist for each of the light wavelengths. To calculate SpO2, an absorption ratio is calculated using the reflected DC and AC components of each of the wavelengths as seen in the following formula:

$$absorption\ ratio = \frac{AC_{660}/DC_{660}}{AC_{940}/DC_{940}}$$

The ratio is used to index a Look Up Table (LUT) of empirical data which maps ratios to SpO2 values. The data is based on experimental data obtained from healthy patients.

3.3.2 Filtering and Initial Processing

To accurately determine the users blood oxygen concentration or SpO2, the infrared and red data values are put through various steps. The first step is subtracting the DC component from the signal. Once this is done, the signal is put through an IIR filter which results in a smoothened-out curve and then inverted. The signal is inverted is it allows detection of signal valleys using a peak detector. Detection of the valleys is required as infrared reflection values drop right after the heart beats and since HbO_2 absorbs infrared light, a minimum amount of light is reflected. A peak detector is then used, and the application stores the timestamp of each of the valleys.

3.3.3 Determining Absorption Coefficient

Once the filtering process is complete, the AC values of each of the wavelengths need to be calculated at the peaks of each of the signals. This is done by finding the time stamp difference and height difference between two adjacent valleys. These values are then used in addition to the time stamp difference from the current valley to the peak in the middle to perform linear interpolation and find the DC component of the signal during the next peak. The AC component can be calculated by subtracting the DC component from the raw value. **Figure 9** demonstrates this process on a sample waveform. Using these values, the absorption ratio may now be calculated and SpO2 can be determined as per **Section 3.3.1.**

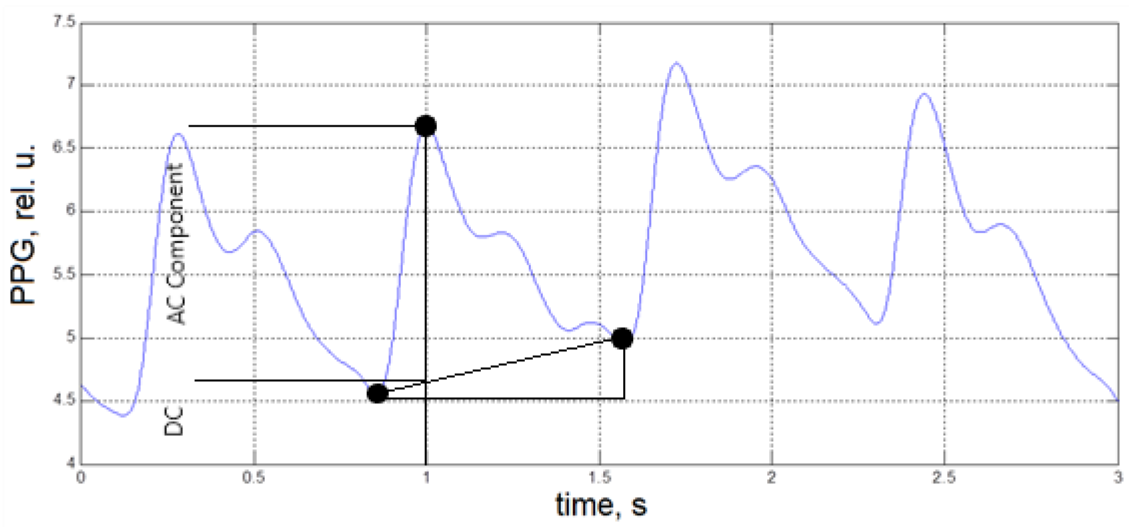

Figure 9: Linear Interpolation of Infrared Curve

3.4 Step Detection

Acceleration and gyration values are retrieved and calculated from the MPU-9250, a 9-axis device containing an accelerometer, gyroscope, and magnetometer. Raw values are obtained from a FIFO buffer provided by the device over I2C. The sensor consists of a DMP (Digital Motion Processor) which performs some initial data filtering and implements features such as tap detection, a pedometer, and orientation determination.

3.4.1 Algorithm

To determine whether a step was taken by the user, gyroscope values obtained from the sensor are used. The magnitude of gyration is calculated to allow for step detection in all 3 axes. See **Figure 10** for a plot of sample gyration magnitude data while walking. The magnitude of the gyration vector can be calculated as:

$$|g| = \sqrt{gx^2 + gy^2 + gz^2}$$

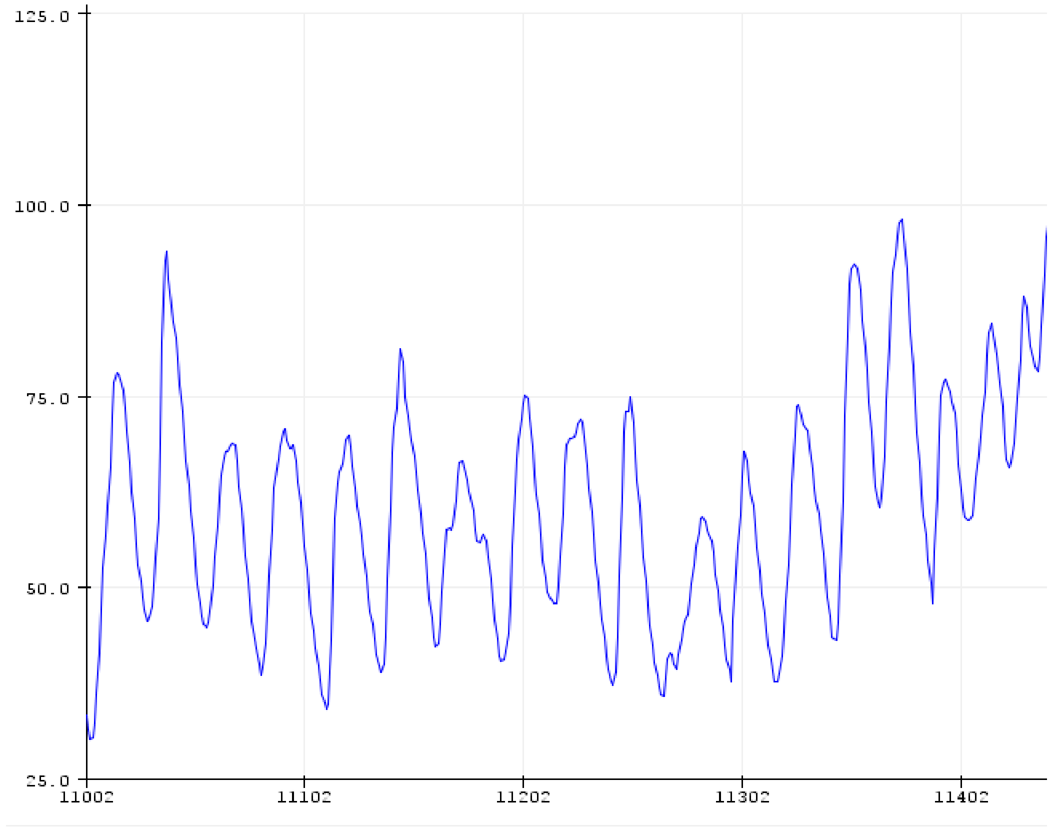

Figure 10: Sample Gyration Magnitude Data

Once calculated, a peak detection algorithm is used to monitor and count peaks in the magnitude of gyration. If the difference between the trough and peak are greater than a predefined threshold, each peak is counted as a step whether it's a user turning, jumping, or running on the spot. To reduce the number of invalid peaks in the gyration data, the gyration magnitude is first passed through a digital filter described in the following section after being calculated and before peak detection takes place.

3.4.2 *Filtering*

Step detection uses a Single-Pole Recursive Low Pass Filter to calculate a moving average of the data and remove any unwanted high frequency gyration data. This step uses the same IIR filter described in **Section 3.2.2** with the same recursive coefficients.

3.5 Memory

The Esperto Watch uses a MB85RC256V FRAM (Ferroelectric Random Access Memory) IC to store user data and device configuration. The IC has the following features:

- I2C communication
- 32KB of non-volatile memory
- Instantaneous read/write (no page or block erases required)
- No wear leveling. Can be written / read to over 10,000,000,000,000 times

The memory Is being used in a 4 byte aligned manner where the first 4 bytes are being used to store the amount of valid data written to the FRAM and the rest of the memory is used to store user data including the step count, heart rate, and SpO2. The format of the data can be seen in **Table 1.**

Byte 1	Byte 2	Byte 3	Byte 4
Hear Rate	Step Count MSB	Step Count LSB	SpO2

Table 1: User Data Format

Where MSB represents Most Significant Byte and LSB is the Least Significant Byte as the step count is store as a 16 bit integer. The watch application stores user data to the FRAM at a fixed frequency whenever not connected to a mobile application.

3.6 Bluetooth Communication

The Esperto Watch uses the SPBTLE-RF module by STMicroelectronics, a low power network processor compliant with Bluetooth Low Energy 4.1. The application uses this module to send user data to the mobile application and receive information such as notifications, date, and time from the mobile application. The SAMD21 communicates to the module over SPI. The application utilizes the BlueNRG layer provided by ST to communicate with the module.

3.6.1 *Mobile Application to Device Communication*

Once the Esperto Watch is connected to the mobile application, the application sends the current date and time according to the mobile device. Whenever the mobile device receives a text or call notification, this information is also sent to the Esperto Watch over BLE. The application firmware knows how to parse incoming packets based on their format:

Time: HH:MM:SS

Date:	DD/MM/YYYY
Call:	C##########
Text:	T##########

3.6.2 Device to Mobile Application Communication

When the watch is connected to the mobile application, it will send user data at a fixed interval. The data will be sent in 4 Byte packets containing the users daily step count, heart rate, and blood oxygen concentration. The format of this data is the same as the data format of the data stored in the FRAM, described in **Section 3.5.**

When the watch connects to the mobile application, the watch will perform a burst transfer and send all the valid data which was stored in the FRAM since the last time the device was connected. The packets will contain up to 20 Bytes of data or 5 instantaneous FRAM writes.

3.7 Display

The display firmware uses the u8g2 library to interface the microcontroller with the SSD1306 display driver over I2C. The display is updated whenever a new message is received from the mobile application or at a frequency of 1 Hz during normal operation. The application uses a 1KB display buffer to depict which pixels are on and off on the 64x128 monochrome display. During normal operation, the display shows the following:

- Time
- Date
- Bluetooth connection status
- Battery status
- Current heart rate
- Daily step count

When a text or call notification is received form the mobile application, the display shows the following:

- Type of notification
- Phone number
- Time
- Bluetooth connection status
- Battery device

3.8 Real Time Clock

The SAMD21 has an internal Real Time Clock running at 32.768 KHz. To keep track of the current time and date more accurately, an external 32.768 KHz crystal is used instead of the oscillator internal to the microcontroller. The RTC is set when incoming date and time data is received over BLE from the mobile application. Furthermore, the RTC also has alarm functionality and can be setup to cause an interrupt after a certain time. This functionality is used in **Section 3.9.**

3.9 Power Management

Numerous power management methods have been implemented to increase the battery lifetime of the Esperto Watch. The current consumption for each of the peripherals and the microcontroller were measured and can be seen in **Table 2.**

Peripheral	Current During Normal Operation (mA)	Current During Shutdown (uA)
SAMD21	10.5	500
MAX30102	2.5	0.7
MPU-9250	3.7	8.4
SPBTLE-RF	0.1	-
MB85RC256V	0.2	27
SSD1306	5.2	200
Other	Approximately 1.9mA	Approximately 1900uA
Total	**24.1 mA**	**2.63 mA**

Table 2: Power Consumption

The SPBTLE-RF already implements a standby/shutdown mode when its not broadcasting or transmitting/receiving data and the FRAM will consume minimal current when a read or write operation is not ongoing thus, they will not have to be shutdown by the application.

4 different device modes have been implemented which will be enabled based on certain circumstances. **Table 3** depicts which devices will be shutdown or turned in during each of these modes of operation.

Peripheral	Normal Operation	Standby	Sleep	Shutdown
SAMD21	ON	ON	ON	OFF
MAX30102	ON	ON	OFF	OFF
MPU-9250	ON	ON	ON	OFF
SSD1306	ON	OFF	OFF	OFF

Table 3: Peripherals Turned on During Different Modes of Operation

3.9.1 Normal Operation Mode

The normal operation mode is enabled when the device is being actively worn and the display is being looked at by the user. The devices state will be toggled between normal operation and standby based on if the user is attempting to look at the display. This action is detected by analyzing the acceleration of the device in the Z direction. If the Z acceleration remains relatively constant and within certain bounds, the display is turned on and the device enters normal operation mode from standby mode.

3.9.2 Standby Mode

Standby mode is entered when the device is being actively used and worn but the user is not attempting to look at the display. This mode is enabled to save on current consumed by the display and the display driver.

3.9.3 Sleep Mode

The device enters sleep mode when a user's presence is not detected. Presence is detected using the heart rate sensor. When the amount of red light reflected is very low, there is nothing obstructing the sensors photodiode and this, a user is not wearing the device. A typical case where the device enters this state is if it is on a desk and not being worn.

When the device is in this state, the heart rate sensor and display are turned off. The processor will wait on a motion interrupt generated by the MPU-9250. Once woken up, the device will enter normal operation mode for a fixed amount of time and check if any presence is detected. If not, the device re-enters sleep mode.

3.9.4 Shutdown Mode

The device enters its shutdown mode when it detects that the battery is low and there is no USB plugged in to charge it. It does this by reading the Analog to Digital Converter (ADC) connected to the battery and USB supply. In this mode, all peripherals are turned off including all interfaces and timers on the SAM D21. The only peripheral which is kept on is the internal 32.768 KHz RTC as it maintains track of the current time and date and is used as an alarm to wake up the processor.

After entering shutdown mode, the processor sets up an alarm to wake it up periodically to check on the battery and charging status. If a valid input is detected, the device proceeds into the normal operation mode. When the device is in this mode, it will appear completely off to the user.

4 Notes