Freescale Semiconductor
Application Note

Document Number:AN4323
Rev. 1, 04/2013

Freescale Solutions for Electrocardiograph and Heart Rate Monitor Applications

by: Jorge González

Contents

1 Introduction

This application note describes how to use the MED-EKG development board, a highly efficient board that can be connected to the Freescale Tower System to obtain an electrocardiogram signal and measure heart rate.

The demo can be implemented in the TWR-K53N512, TWR-MCF51MM or TWR-S08MM128.

This demo uses the Freescale USB stack to send data to a PC, making it possible to monitor heart activity through a graphical user interface (GUI).

The Tower System and MED-EKG board allow the user to implement heart signal conditioning in three different modes:

- With the internal OPAMPs and TRIAMPs of the microcontrollers
- With the external instrumentation amplifier and operational amplifiers featured in the MED-EKG board
- With a mixture of both

The information in this application note is intended for developers of medical applications, for those in the medical field, such as biomedical engineers or doctors, or for people who want to know more about the operation of heart rate monitors and electrocardiograph devices.

2 Heart rate and electrocardiograph fundamentals

This section provides an introduction to the basic concepts of heart activity in order to explain the development of the electrocardiograph demo.

2.1 Heart: functional description

The heart is located in the middle of the thorax, positioned slightly on the left and surrounded by the lungs. Its function is to pump blood to all parts of the body so that the organs receive oxygen.

The heart can be divided into four chambers: two upper atria (right and left) and two lower ventricles (right and left). Atria receive blood coming, respectively from the tissues or lungs through the venous system, and ventricles eject blood towards the tissues or lungs. Figure 1 shows the cardiac conduction system with the structure just mentioned.

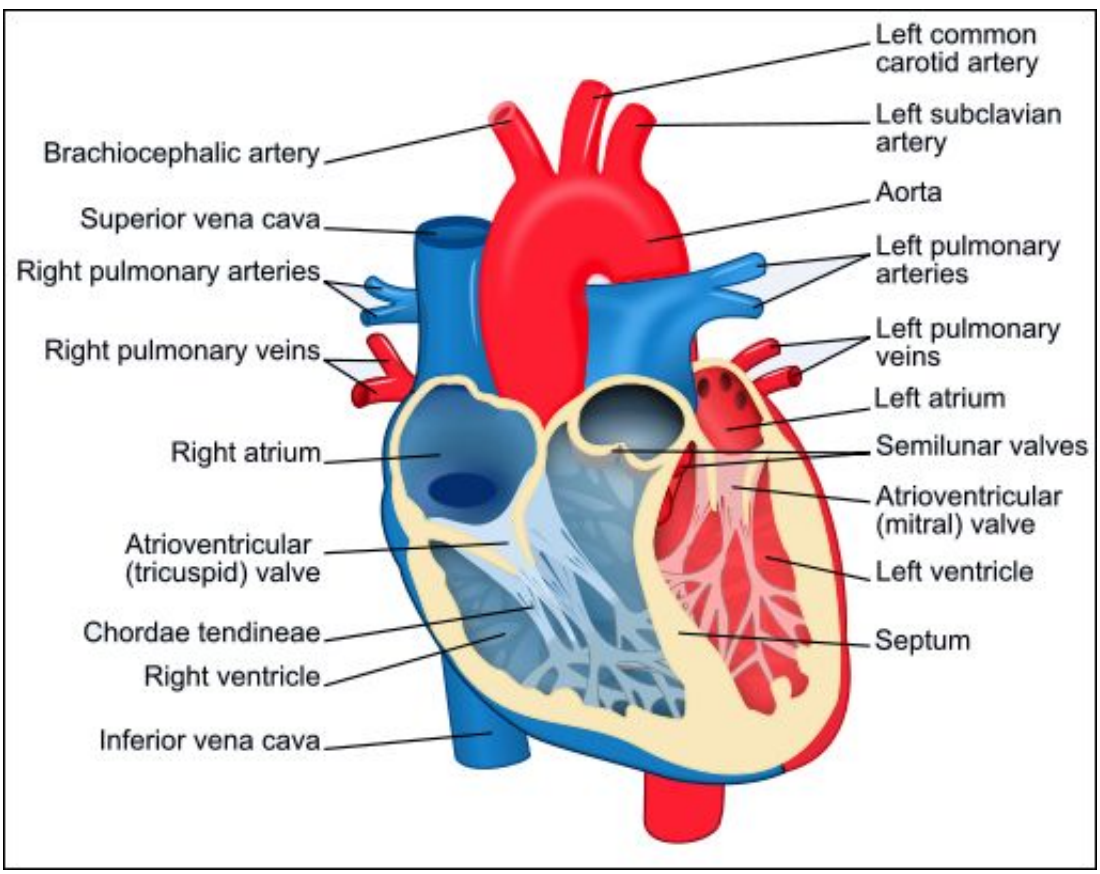


Figure 1. Cardiac conduction system

The cardiac cycle is described next and complemented with Figure 2, which is a scheme of the blood flow in the circulatory system. The cycle consists of three principal stages:

- **Atrial systole:** Atria contract and eject blood to the ventricles in a passive way. Once the blood has been expelled from the atria, atrioventricular valves (located between atria and ventricles) are closed to avoid blood return.
- **Ventricular systole:** The ventricles contract, ejecting blood towards the circulatory system. At this stage, it is necessary to overcome the high pressure of the aortic and pulmonary valves. Ventricles are never completely emptied, retaining some of the blood.
- **Diastole:** At this point, all the parts of the heart are relaxed to allow the arrival of new blood.

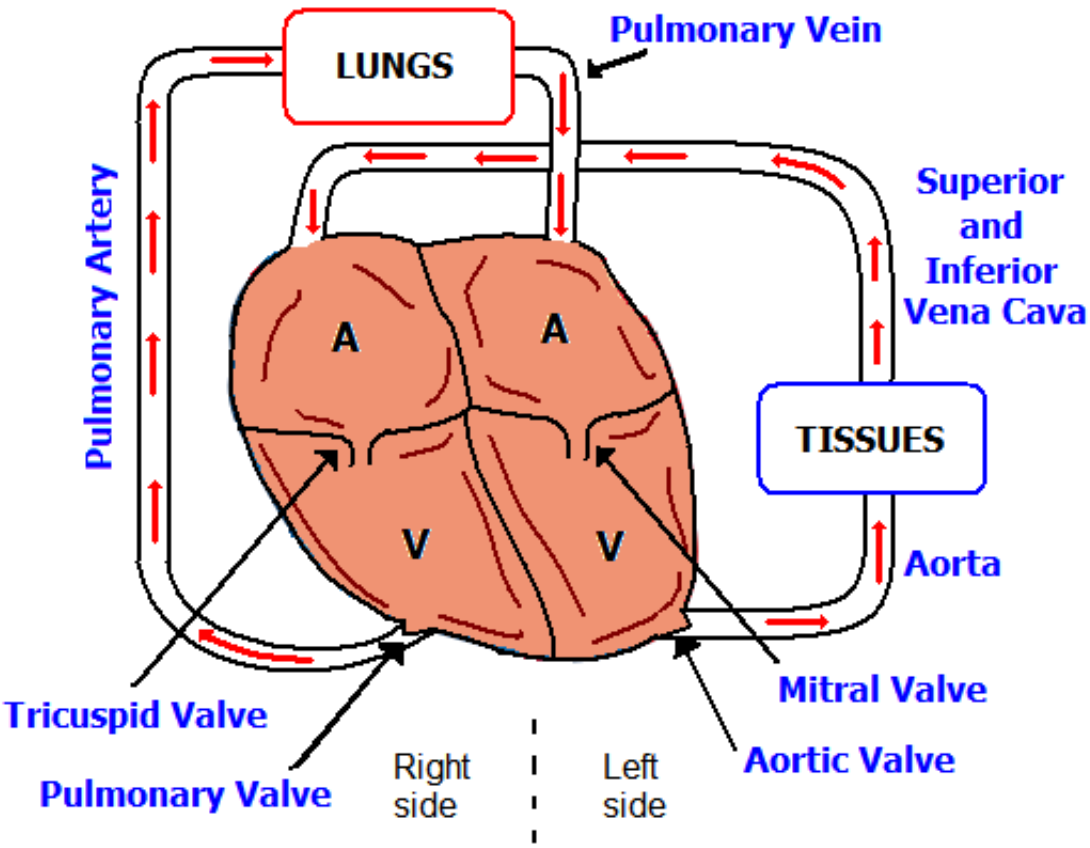


Figure 2. Blood circulation scheme

2.2 Electrical activity of the heart

There is an electrical activity in the heart, related to depolarization and repolarization of myocardial cells, that accompanies the blood flow. The electrical impulse starts in the sinoatrial node and flows first through the atria, reaching the atrioventricular node and generating the atrium contraction. After that, there is a brief pause and the current flows through the His bundle towards the ventricles, generating their contraction. Finally, current arrives to the Purkinje fibers and repolarization of the heart tissue occurs.

The following sequence represents electrical activity during a heartbeat. It is depicted in Figure 3 :

1. Atrium begins to depolarize.
2. Atrium depolarizes.
3. Ventricles begin to depolarize at apex. Atrium repolarizes.
4. Ventricles depolarize.
5. Ventricles begin to repolarize at apex.
6. Ventricles repolarize.

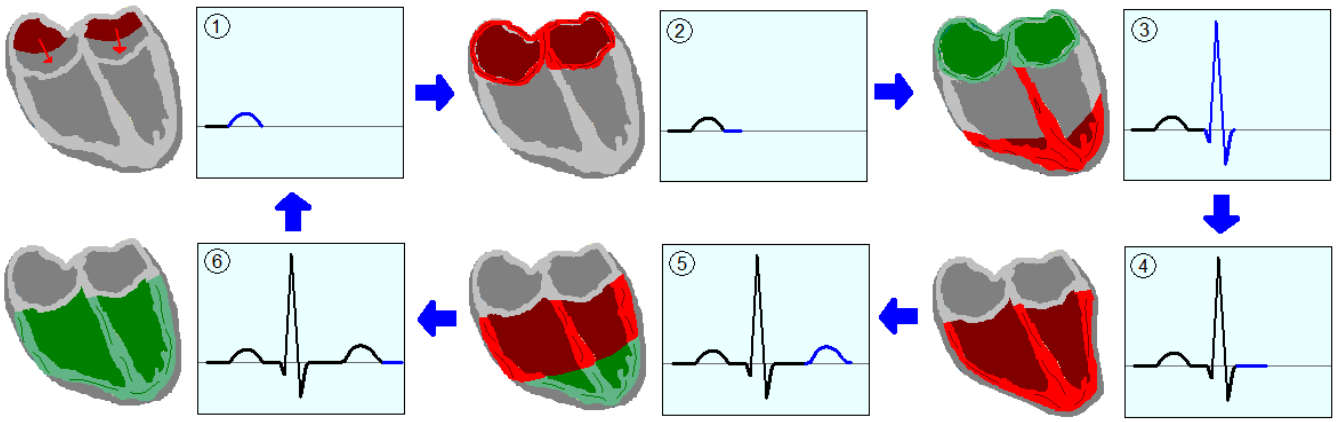


Figure 3. Electrical activity in myocardium

2.3 Heart signals and the QRS complex

Figure 3 represent the voltage signals generated during heartbeat. The sequence of pulses comprise the typical heart signal shown in Figure 4.

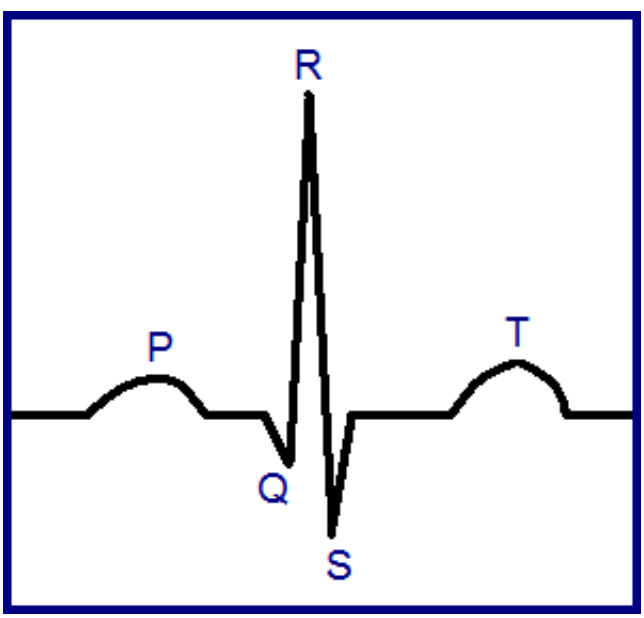


Figure 4. Electrocardiogram signal

Heart muscles generate different voltages in the order of hundreds of microvolts. This signal has very low amplitude and an important amount of electrical noise. The QRS complex is usually the central and largest part of the electrocardiogram signal. The P wave represents the atrium contraction, while the QRS complex and T wave represent the actions of the ventricles. The higher amplitude of the QRS complex compared with the P wave is due to the fact that the ventricles contain more muscle mass than atria. Actually, there is one atrium repolarization wave that resembles an inverse P wave, but it is overlapped by the QRS wave. The electrocardiogram signal is useful in diagnosing cardiac arrhythmias, conduction abnormalities, ventricular hypertrophy, myocardial infarction, electrolyte derangements, and other heart diseases.

2.4 Biological electrical potentials

As mentioned previously, the heart generates potentials that can be expressed as vector quantities, meaning that it is possible to assign a numeric value with its respective sign to those quantities. The heart can be represented as a dipole located in the thorax with a specific polarity at a certain moment, and an inverted polarity the next moment. The potential in a specific moment depends on the amount of charge and the separation between charges.

When we refer to potentials, we are actually talking about potential differences; therefore heart signals must be acquired with a reference point, which is used to determine significant variations from the points whose potentials we want to record. An example of how we can acquire signals is shown in Figure 5. This example uses an operational amplifier, since its principal function is to obtain the potential difference between two signals and amplify it.

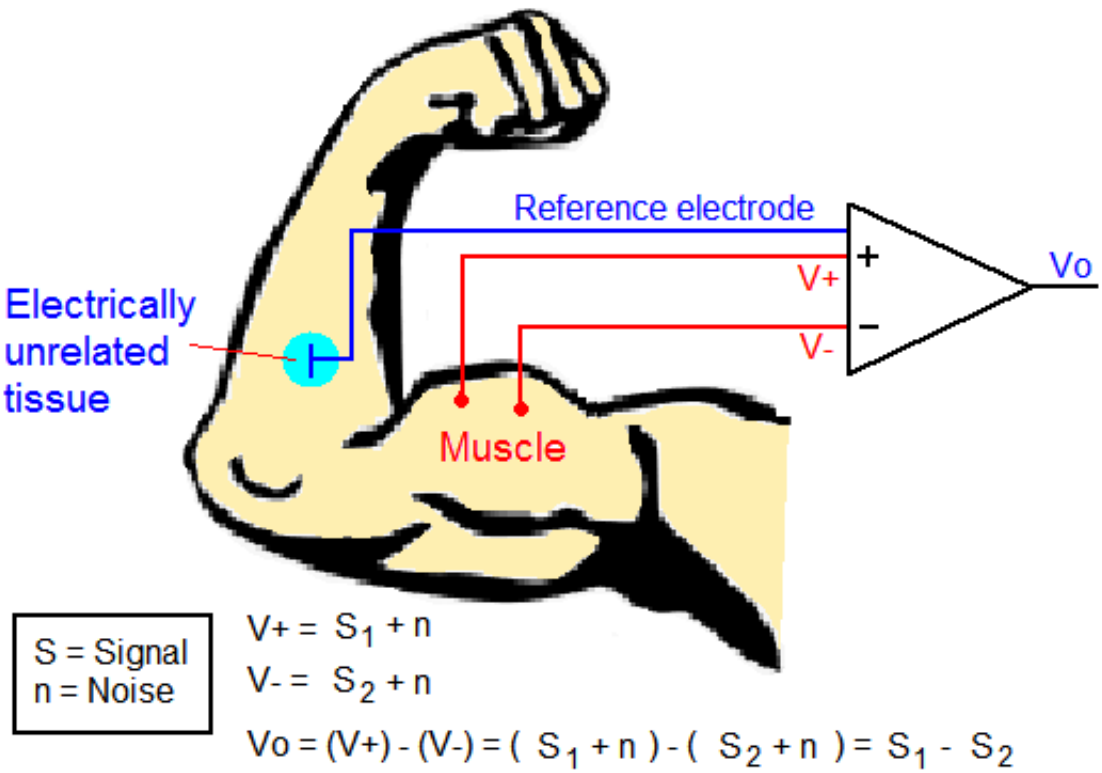


Figure 5. Signal acquisition

A lead is one pair of electrodes or a combination of electrodes. In cardiology, there are three basic leads: at 0° (Lead I), 60° (Lead II) and 120° (Lead III). Figure 6 is a representation of those leads that, together, form an imaginary figure known as Einthoven's Triangle.

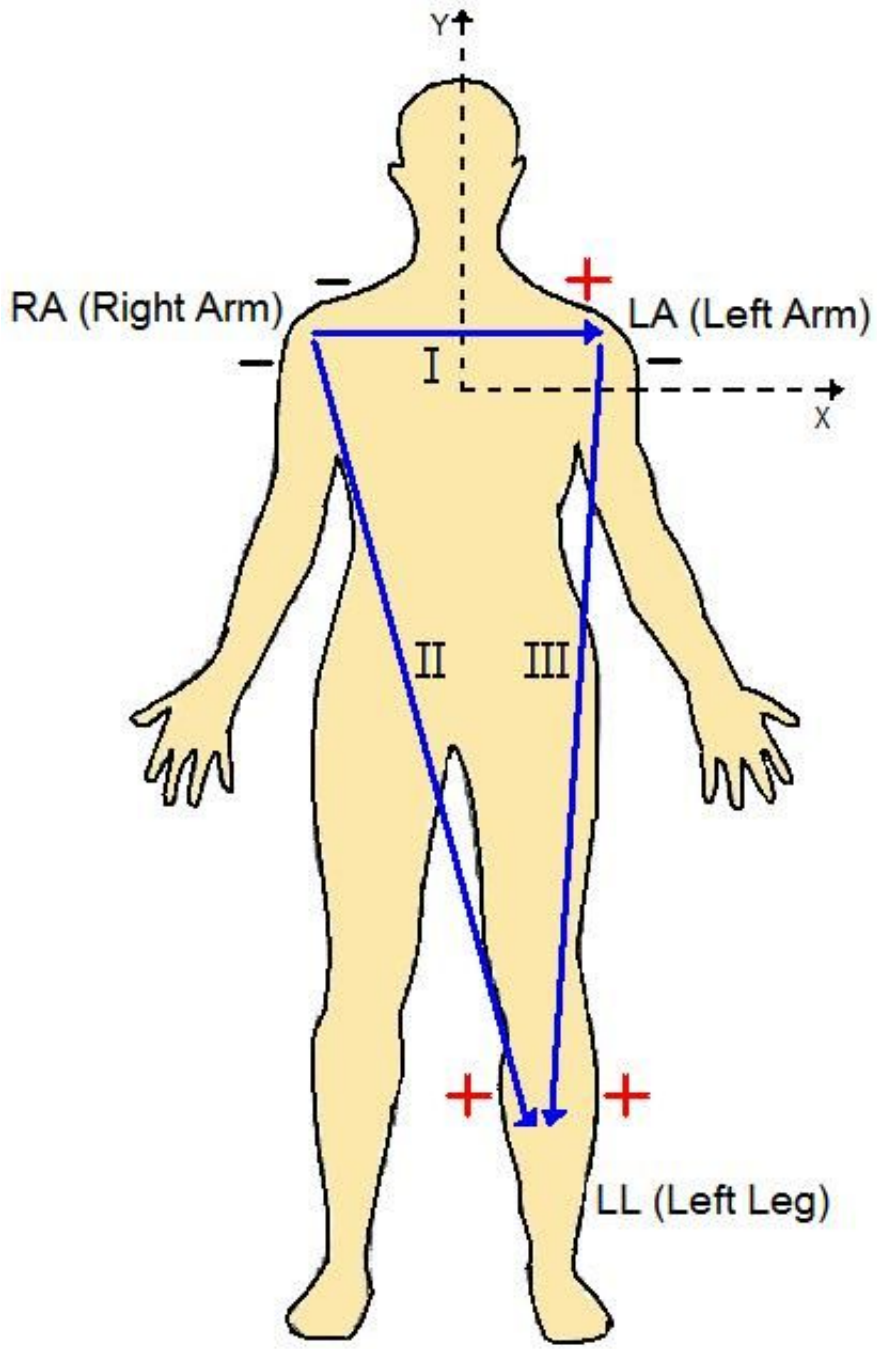


Figure 6. Einthoven's Triangle

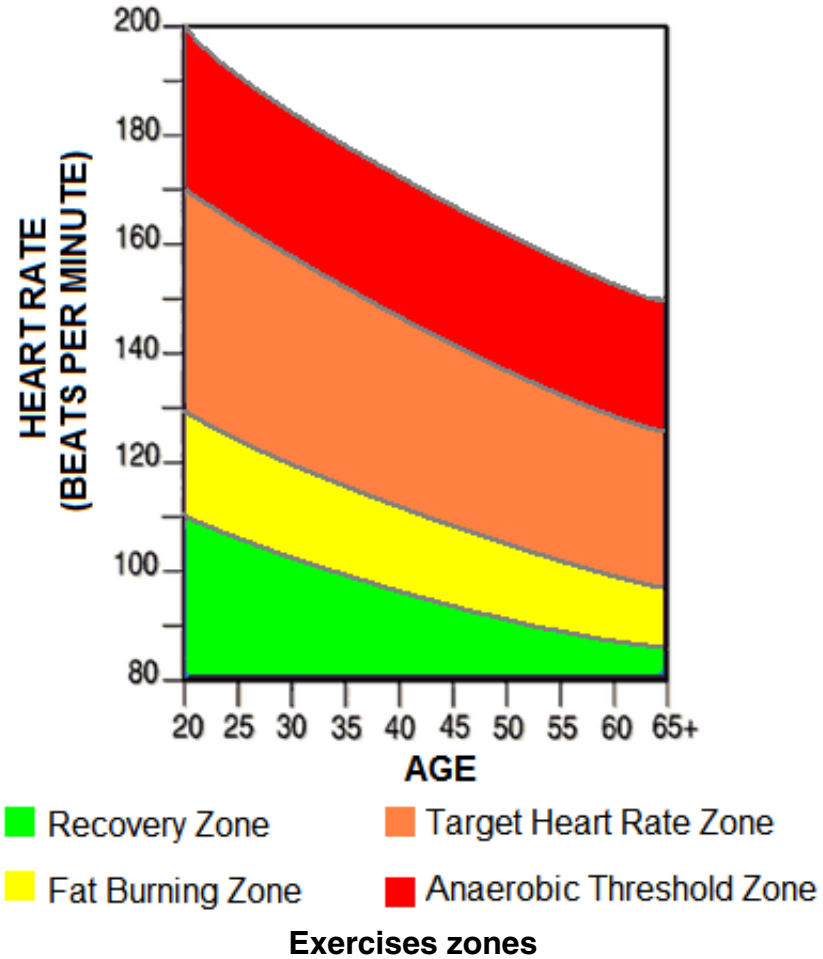
The three electrodes for measuring the electrical signals are placed on the limbs; one on the left arm (LA), one on the right arm (RA), and the last on the left leg (LL). The reason for connecting electrodes that way is to obtain the bio-potentials coming from the chest, more specifically from the heart.

2.5 Heart rate ranges

Heart rate varies depending on several factors such as age, gender, metabolism, genetics, physical activities, temperature, and so on. One of the most common situations in which heart rate is measured is during exercise.

Figure 7 correlates average heart rate with physical condition and age.

Figure 7.



3 Hardware description

This section describes how to implement an EKG system, acquire signals, use instrumentation amplifiers, filtering, and the digital signal controller.

When it comes to measuring heart rate or displaying an electrocardiogram, the first consideration is the acquisition of an accurate, noiseless signal with enough amplitude to be quantified. The Kinetis K53, ColdFire MCF51MM, and S08MM128 offer many advantages in this regard.

The features of Kinetis K53 are listed below.

- Ultra low-power operation
- 2 operational amplifiers (OPAMP)
- 2 transimpedance amplifiers (TRIAMP)
- 2 × 12-bit DAC
- 2 × 16-bit SAR ADC, up to 31 channels with programmable gain amplifiers (PGA)
- Inter-integrated circuit (I^2C)
- USB connectivity
- ARM® Cortex™-M4 core with digital signal processor (DSP) instructions

The MCF51MM and S08MM128 have the following features.

- Ultra low-power operation
- 2 operational amplifiers (OPAMP)
- 2 trans-impedance amplifiers (TRIAMP)
- 16-bit SAR analog to digital converter (ADC), 4 differential channels and up to 12 external single-ended channels.
- 12-bit digital-to-analog converter (DAC)
- Inter-integrated circuit (I^2C)
- Universal serial bus (USB) connectivity
- Multiply-accumulate unit (MAC only in MCF51MM)
- ColdFire V1 and HCS08 cores, respectively

In addition, the MED-EKG board includes the MC56F8006, a 16-bit digital signal controller (DSC) which performs the digital filtering of the signal. The main features of the MC56F8006 DSC are:

- 3 analog comparators (ACMP)
- 2 x 12-bit ADC
- 6 PWM outputs
- I^2C
- JTAG-ONCE for programming

The DSC communicates with the microcontroller through the I^2C bus.

3.1 EKG system implementation

Figure 8 is a general block diagram of the MED-EKG system.

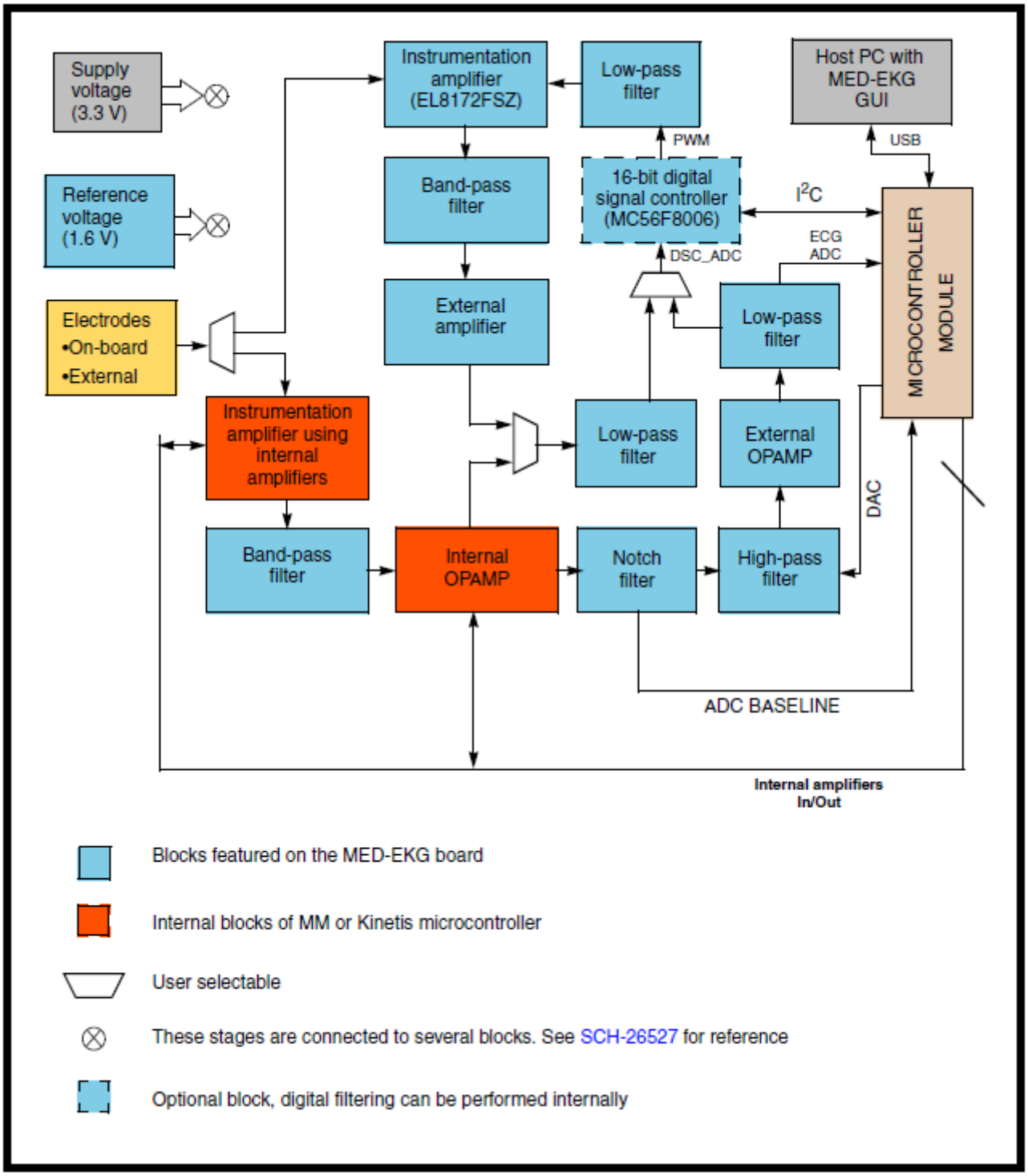


Figure 8. MED-EKG system block diagram

3.2 Signal acquisition

ECG signal can be acquired either by using the on-board fingertip electrodes or connecting external electrodes. Figure 9 shows both the options. The fingertip electrodes are marked with the number 1 and the external electrodes connector is marked with the number 2.

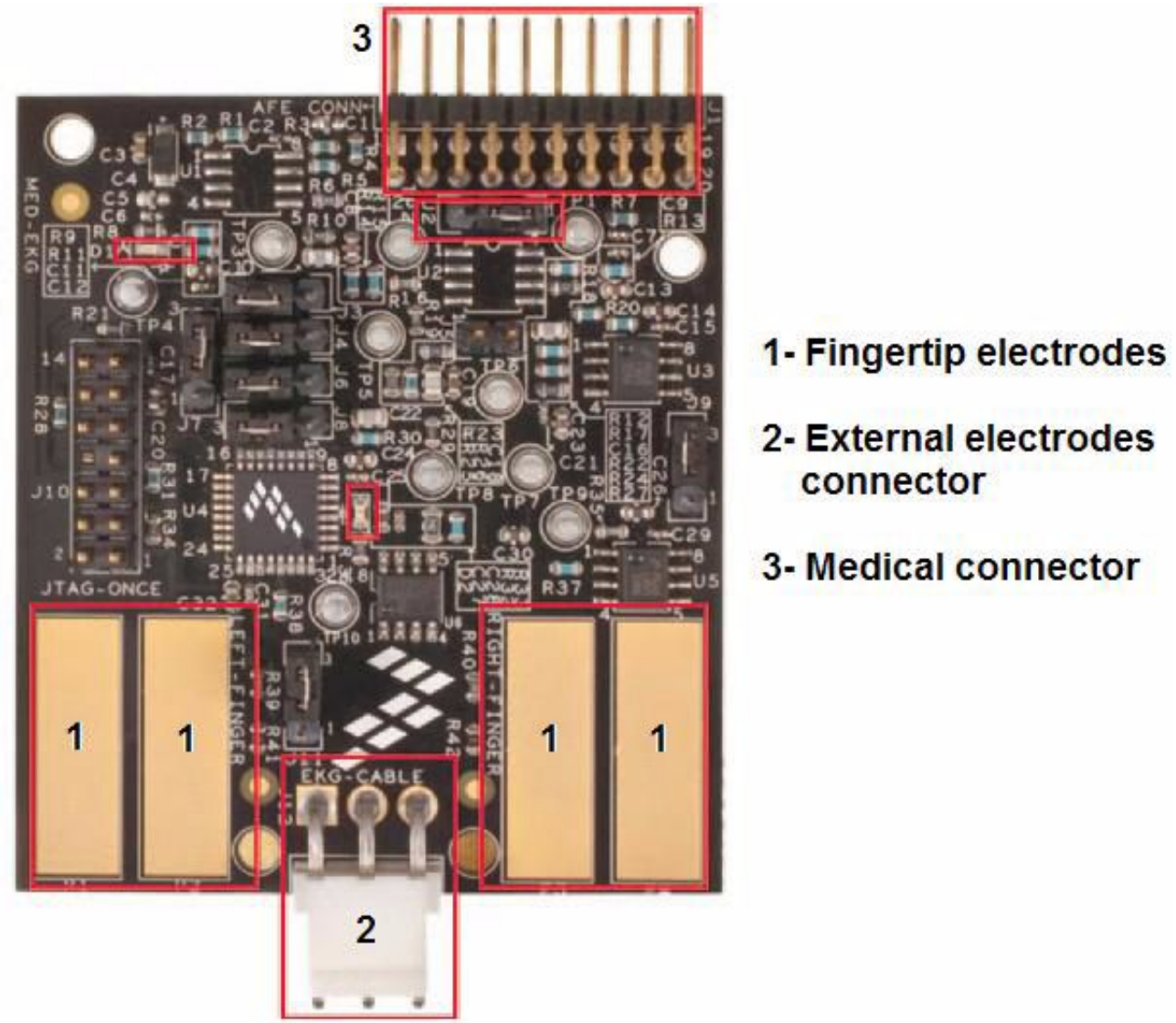


Figure 9. MED-EKG development board connectors

3.3 Instrumentation amplifier

An instrumentation amplifier consists of three operational amplifiers configured to obtain a high gain and high common-mode rejection ratio (CMRR). Instrumentation Amplifiers are widely used in biopotentials acquisition.

MED-EKG board provides two options for implementing the instrumentation amplifier. Those options are described below.

3.3.1 Instrumentation amplifier using internal OPAMPs

It is possible to use the internal OPAMPs and TRIAMPs of the Kinetis and MM microcontrollers to implement an instrumentation amplifier. The connection diagram is shown in Figure 10.

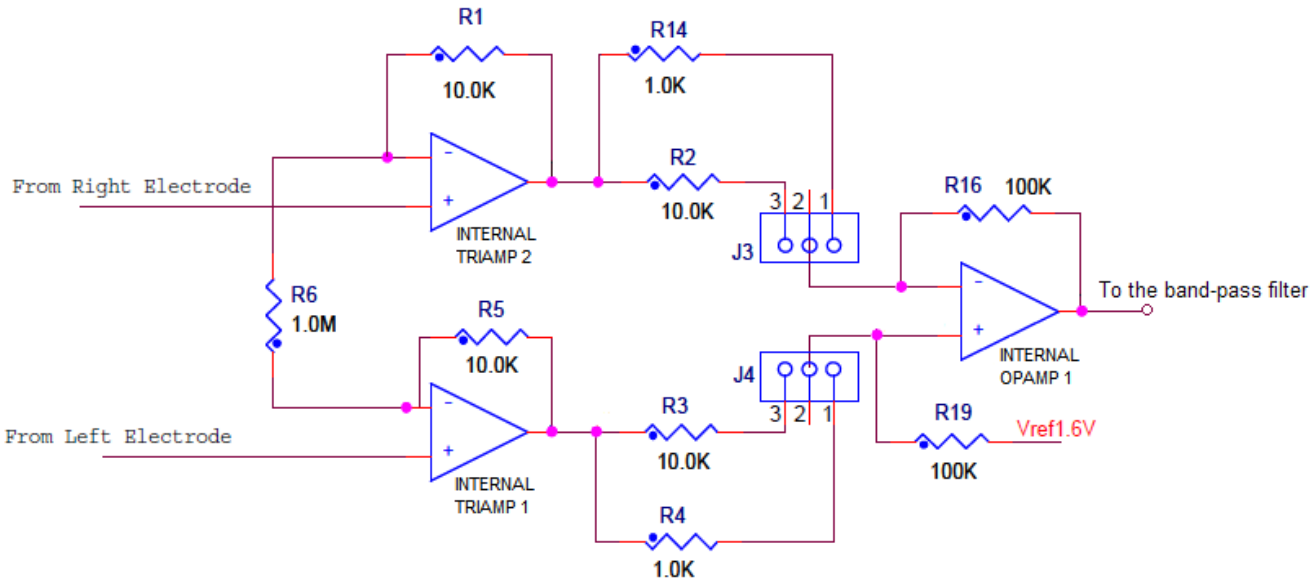


Figure 10. Instrumentation amplifier with internal OPAMPs and TRIAMPs

The gain is configured with jumpers J4 and J3 of the MED-EKG board, and it has to be the same in both cases. The equivalent gain of the amplifier is as detailed below:

Connections (J3 and J4)	Gain
1–2	$G = \frac{100}{1} = 100$
2–3	$G = \frac{100}{1} = 10$

The output of the amplifier is connected to a band-pass filter.

3.3.2 External instrumentation amplifier (EL8172FSZ)

The EL8172 is an instrumentation amplifier integrated circuit (IC) whose gain is configurable through external resistors. Figure 11 shows the use of this IC in the MED-EKG board:

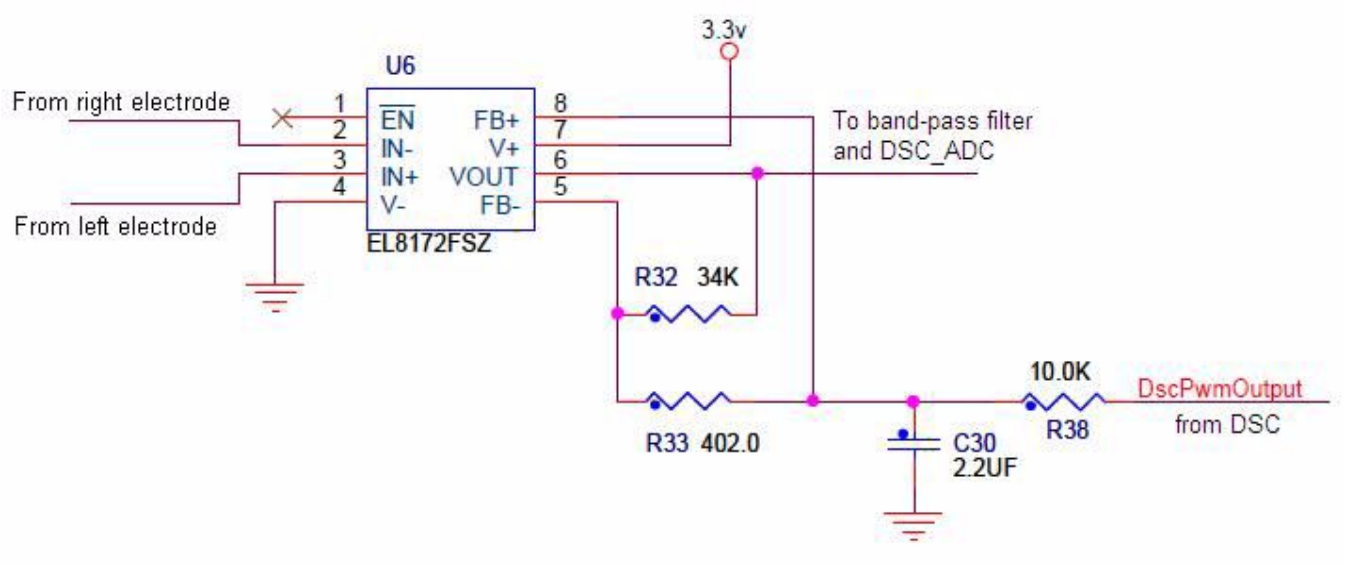


Figure 11. Instrumentation amplifier with IC EL8172

The gain of the amplifier is determined with R33 and R32. The calculation is shown below:

$$G = \left(\frac{R32}{R33}\right) + 1 = \left(\frac{34K}{402}\right) + 1 = 85.57$$

The digital signal controller generates a PWM signal that functions as an offset generator, with the intention of maintaining the EKG signal variations with a base level, giving stability to the signal shown in the GUI and resulting in a more precise heart rate measurement.

The output of this amplifier is sent to the respective band-pass filter, and such output is also used by the DSC as a reference to generate the adequate PWM feedback signal.

3.4 Filtering

Although instrumentation amplifiers have a high CMRR, it is still necessary to filter the signal. Some causes for noise are the line noise, muscle contractions, respiration, electromagnetic interference, and electromagnetic emissions from electronic components. A band-pass filter helps in attenuating those sources of noise. The low limit rejects signals caused by respiration and muscle contractions. The high limit avoids the pass of electromagnetic high-frequency interferences.

As can be seen in the system's block diagram, there is one band-pass filter for the internally implemented instrumentation amplifier and another for the external. Each one has different cutoff frequency; these cutoff frequencies are described in the following section.

3.4.1 Band-pass filter (0.5 Hz–250 Hz)

The cutoff frequencies for the first band-pass filter, the one located after the internal instrumentation amplifier configuration are 0.5 and 250 Hz, correspondingly. The values for capacitors and resistors and the respective process to obtain the cutoff frequencies are described below:

- High limit (153 Hz):C = 22 nF R = 47 kΩ

$$\int = \frac{1}{2\pi RC} = \frac{1}{2\pi(29.4\Omega)(22\text{nF})} = 246.06\text{Hz}$$

- Low limit (0.5 Hz): C = 0.33 F R = 1 MΩ

$$\int = \frac{1}{2\pi RC} = \frac{1}{2\pi(47\Omega)(22\text{nF})} = 153.9\text{Hz}$$

Filter implementation is shown in Figure 12

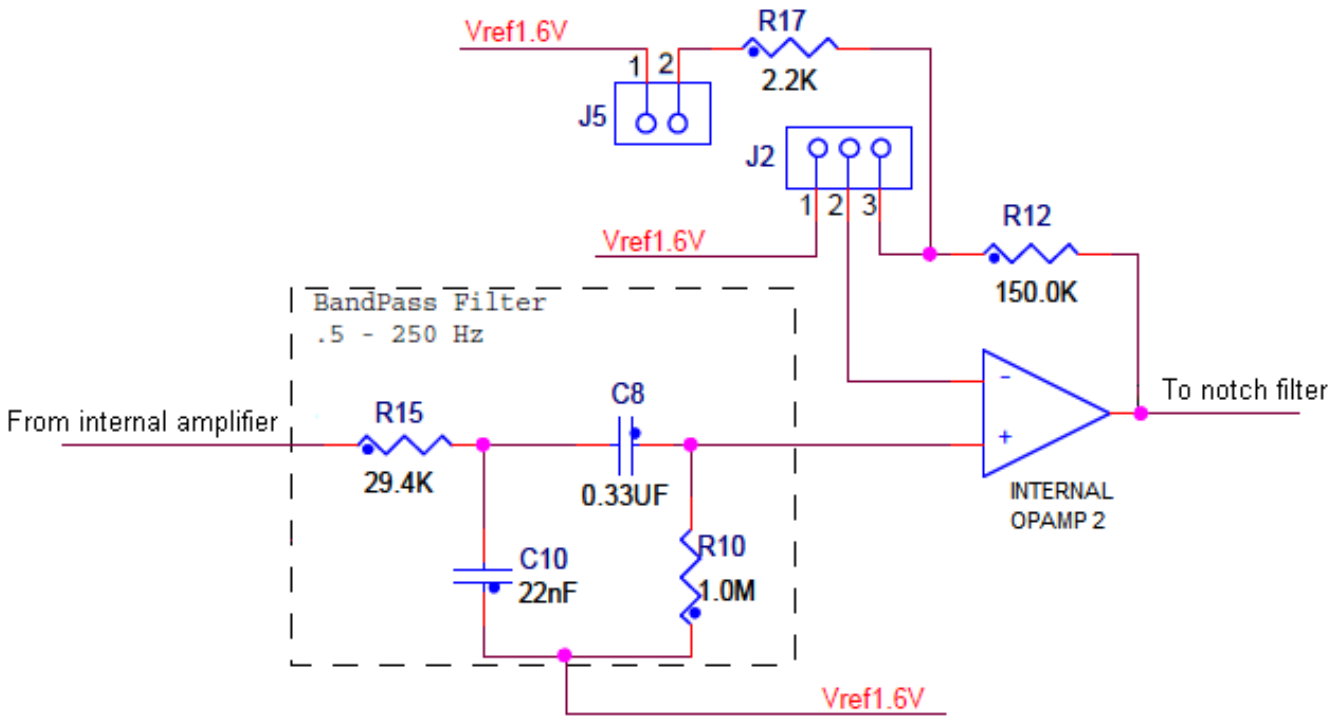


Figure 12. Band-pass filter from 0.5 Hz to 250 Hz

After band-pass filtering, an extra internal operational amplifier adjusts the signal to better levels to be sent to the microcontroller's ADC. Developers may configure the MED-EKG board's J2 and J5 jumpers to select one of the following options:

J2	J5	Operation
1–2	—	Negative input of the amplifier is connected directly to Vref (1.6 V) and gain is configured by software.
2–3	1–2	Amplifies the signal with a gain of 69.
2–3	Open	Gain of the signal = 1. Amplifier acts like a buffer.

3.4.2 Band-pass filter (0.5 Hz – 153 Hz)

The limit frequencies for the first band-pass filter, the one located after the internal instrumentation amplifier configuration are 0.5 and 250 Hz, correspondingly. The values for capacitors and resistors and the correspondent process to obtain the cutoff frequencies are described below:

High limit (250 Hz):C = 22 nF R = 29.4 kΩ

$$f = \frac{1}{2\pi RC} = \frac{1}{2\pi(47\Omega)(22\text{nF})} = 153.0\text{Hz}$$

As can be seen in Figure 13, there is another external OPAMP and a low-pass filter that condition the signal with more appropriate levels to connect to the DSC. The low-pass filter values have already been used in a previous filter from Band-pass filter (0.5 Hz–250 Hz), and the gain of the amplifier is simply calculated with the following equation:

$$G = \frac{R35}{R37} = +1 = \frac{100}{5.11} + 1 = 20.56$$

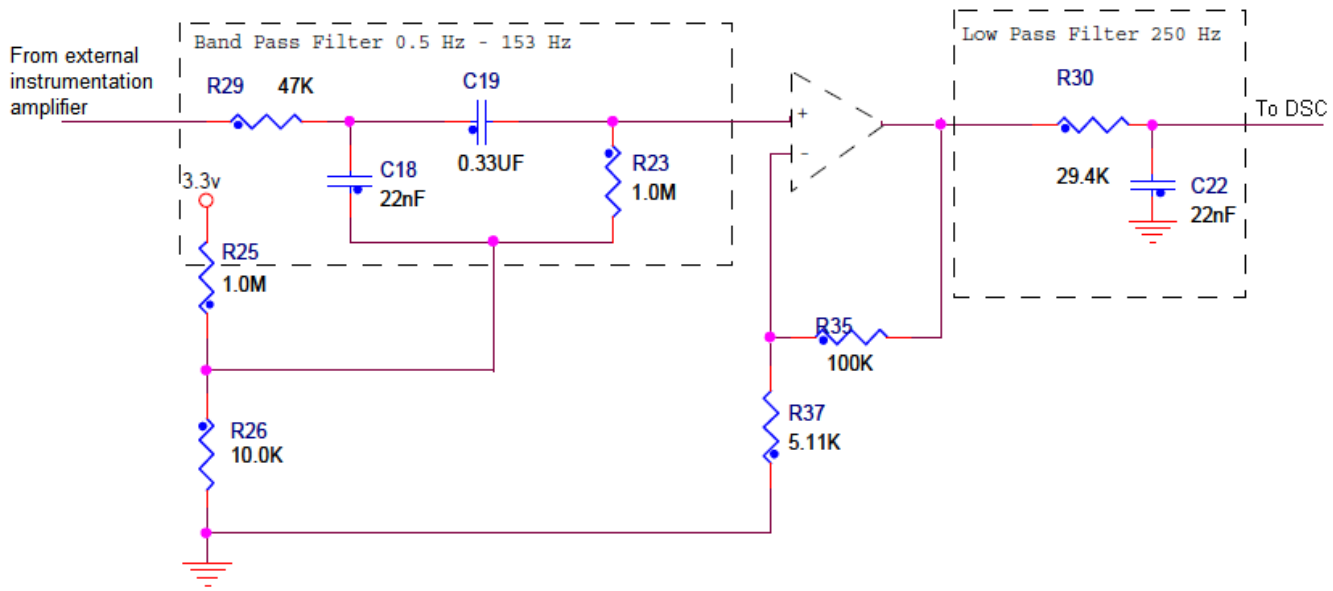


Figure 13. Band-pass filter from 0.5 to 153 Hz

Resistors R25 and R26 help to maintain the offset of the band-pass filter's input signal, since negative values of the signal must not get lost. The output of all these blocks is finally sent to the DSC.

3.4.3 Notch filter (60 Hz)

The notch filter is designed to remove the 60 Hz signals. The reason to include this filter is to eliminate any noise related to an electrical power source, since electrical installations in America have a 60 Hz frequency. Figure 14 shows the distribution of the components that form the notch filter. The diagram also includes two more filters and an operational amplifier. The filters are high-pass and low-pass, but with the same cut frequencies of the band-pass filter. The amplifier provides a gain of 3.3.

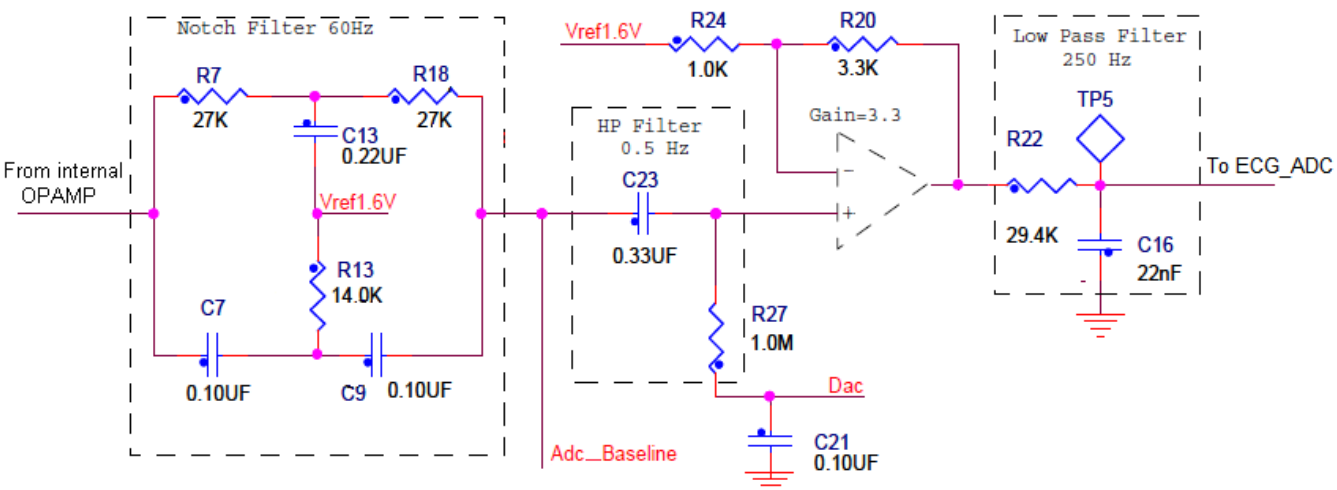


Figure 14. Notch filter

The microcontroller uses the signal from the notch filter as a reference upon which to set a baseline level. The microcontroller does this by sending a feedback signal from the output of the internal DAC to the high-pass filter. Finally, the signal resulting from all these blocks is sent to the microcontroller for sampling and measurement.

3.5 Digital signal controller (DSC)

Once the DSC receives the signal through one of its ADCs, it takes samples of the signal and makes conversions from the analog values to digital values. These values are used to perform a digital filtering that sends an improved signal to the microcontroller through the I2C bus. The DSC also calculates heart rate by itself and sends the value to the MCU.

As mentioned before, the DSC also gives back a PWM signal to the circuit to establish a baseline on the electrocardiogram.

An advantage of this DSC is that, despite the fact that it has been factory programmed for digital filtering, it is possible to reprogram it with the external JTAG-ONCE interface, using the connector embedded in the MED-EKG board; this way, users can program their own applications.

The use of the DSC is optional. The ARM Cortex-M4 (Kinetis K50) and ColdFire V1 (MCF51MM) cores can execute multiply-accumulate (MAC) DSP instructions to perform the digital filtering internally. The HCS08 core is capable of filtering the signal even though it doesn't have the MAC feature, but with a loss in performance. Besides, the peripherals in the microcontroller modules can obtain the different signals and achieve both the conversion to digital and the computation of heart rate, as well as the offset control.

4 Software architecture

This section describes the software functions most relevant for obtaining a valid heart rate and maintaining the signal in a stable operation baseline. It also explains some of the routines related to the ECG demo.

NOTE

The respective software for TWR-K53N512, TWR-MCF51MM, and TWR-S08MM128 is available on **freescale.com** along with this application note.

4.1 Freescale USB stack

The microcontroller communicates with the host PC through a USB interface. Freescale offers a solution to implement medical applications and easily connect medical devices to the PC: Freescale's USB stack with personal healthcare device class (PHDC), which is a portable and easy-to-use source code designed for many applications that require interconnection with a host computer. The USB stack makes it possible for developers to focus only on their applications without having to worry about USB protocol. In this case, the USB device acts like a communication device class (CDC).

The graphical user interface (GUI) installed in the host computer controls the actions of the MED-EKG demo with a series of data packet transactions. Basically, there are three types of packets:

- REQ packet: The host sends a REQUEST packet either to start or to stop a measurement.
- CFM packet: After the device receives a REQ packet, a CONFIRMATION packet is sent to the host in order to notify that the command is valid or that there was an error.
- IND packet: Once the buffer of the ECG with the data to be sent is full, the microcontroller sends an INDICATION packet and the data as well.

Figure 15 is a diagram showing the basic functioning of the USB protocol, and a more detailed explanation can be found in Communication protocol, at the end of this document.

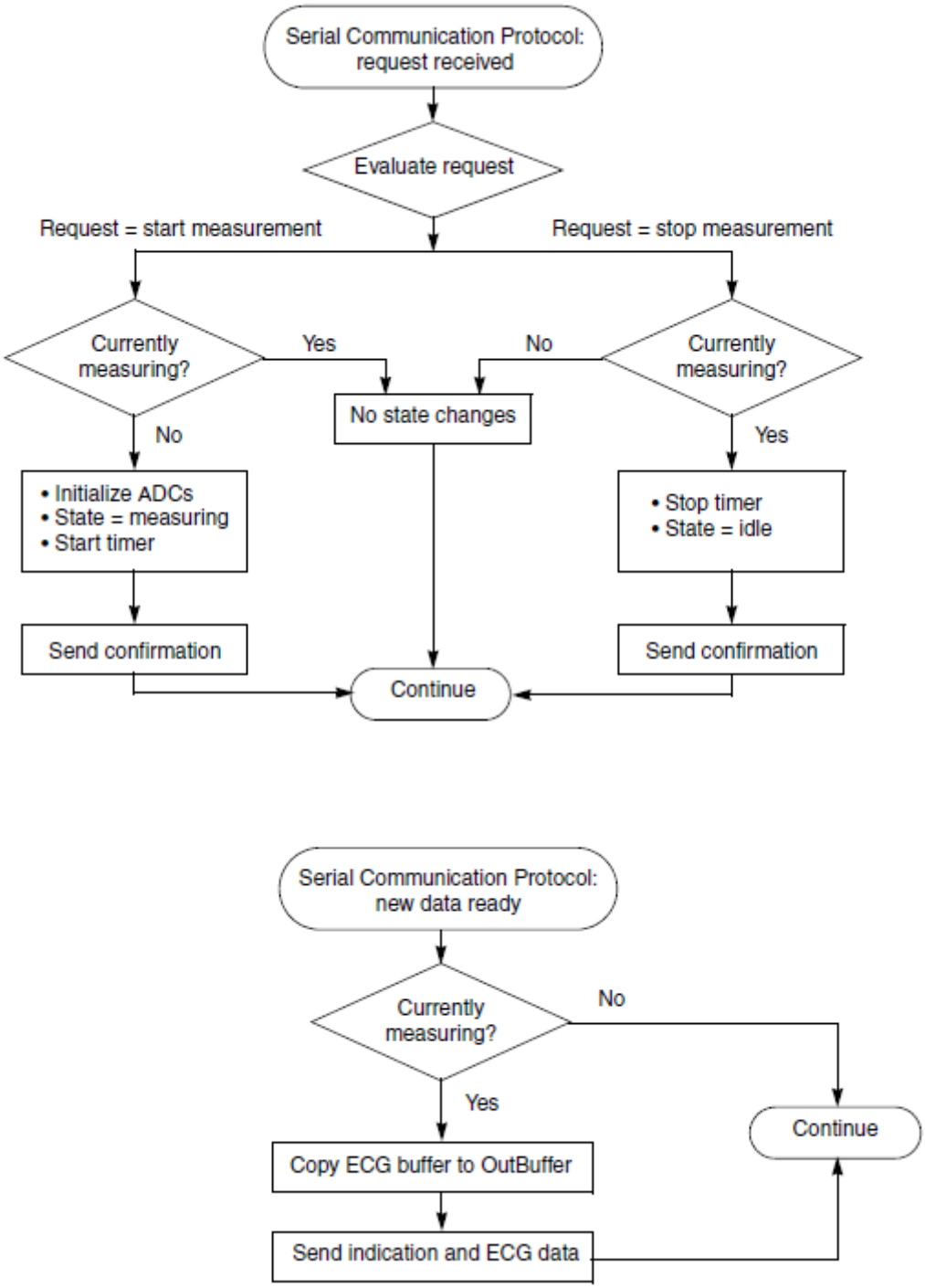


Figure 15. Communication with host PC

4.2 State machines

The software for the MED-EKG includes the functionality of state machines. This type of execution emulates parallelism and ensures that the process follows a well-defined sequence of steps, or "states." It is this functionality that allows the device to send data packets, for example, while measurement is underway. Figure 16 presents the concept of state machines in a graphical format.

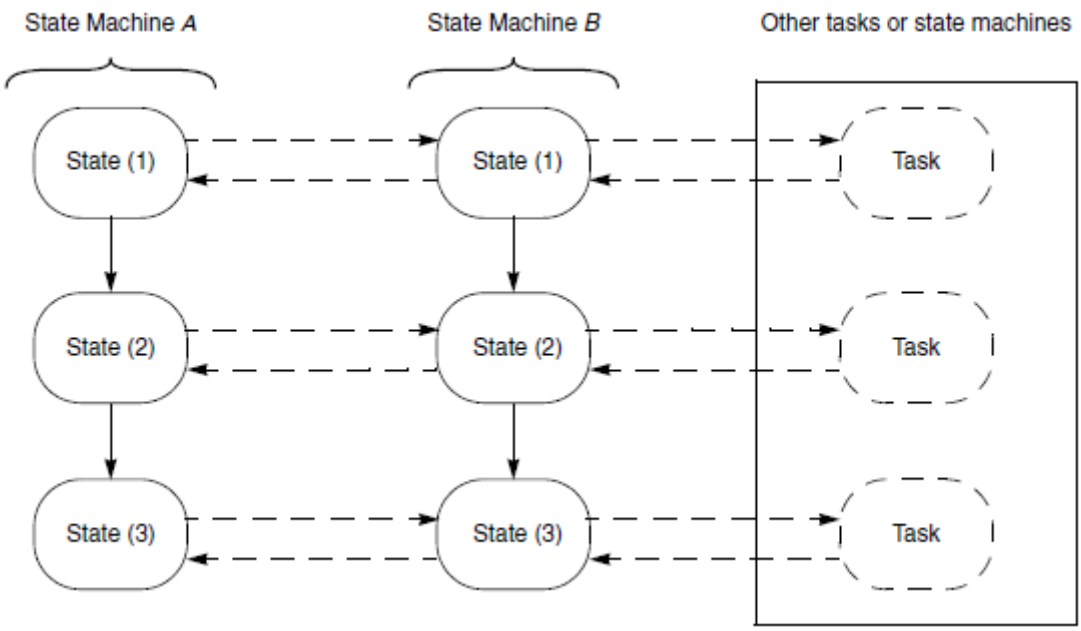


Figure 16. State machine outline

4.3 Timer configuration and functioning

The MED-EKG demo uses the timer module to generate interrupts each millisecond. One sample of the electrocardiogram signal is taken every 2 milliseconds, meaning that two periods of the timer have to pass between successive samples.

Figure 17 contains a flow chart explaining the basic functioning of the timer.

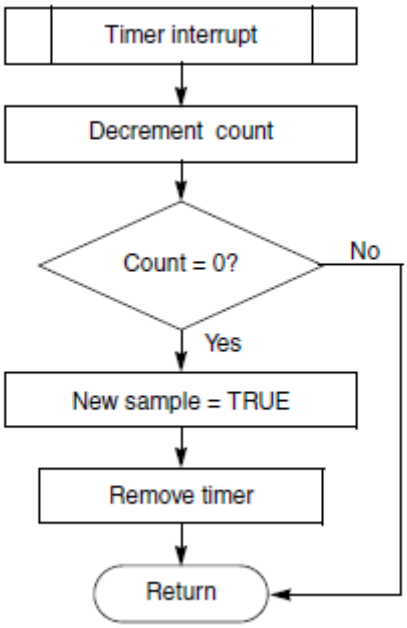


Figure 17. Timer events flow chart

With each interrupt generated by the timer, the interrupt routine verifies whether or not the count is over in order to indicate that a new sample of the ECG signal can be taken. When the count is zero, the timer is deactivated until the software starts it again.

4.4 ECG initialization

The following sequence is used to initialize the peripherals of the microcontroller used in the MED-EKG demo. The flow chart for this stage is shown in Figure 18.

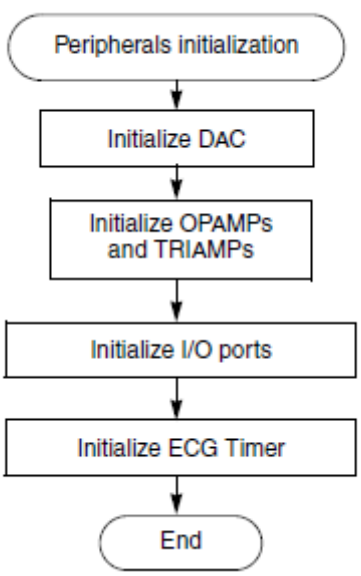


Figure 18. ECG initialization flow chart

4.5 Measurement and conditioning

The software described in this section handles the ECG signal data and calculates the heart rate as well. All this information is sent, afterwards, to a host device using the Freescale USB stack. This block also controls the offset voltage provided by the microcontroller's DAC—which sets a base level for the ECG signal—and the gain for one of the internal OPAMPs.

4.5.1 State measuring

The principal function performed by the system consists of the process depicted in Figure 19. It is one of the possible states in the state machine and its name is "StateMeasuring." This function is called after a request from the host device to start ECG measurement. As the flow chart in Figure 19 shows, execution departs from the function through the blocks labeled "Return." The purpose of this is to perform some tasks related to USB communication while waiting for another lapse to sample the signal. It is a periodic routine; therefore, the system always comes back to it while the host device does not send a request for abortion.

The basic algorithm consists of the following:

- While the current state is "State Measuring," the software waits for the timer interrupt routine to indicate when a new sample of the signal must be taken.
- Once the time for a new sample has passed, the microcontroller performs two analog-to-digital conversions; one of them is merely to adjust the baseline and the other is the ECG signal itself.
- The software calls a routine that handles the baseline adjustment. That routine is "PerformControlAlgorithm."
- The successive samples are compared, in order to detect when a big variation occurs, by checking the difference between samples against a predefined threshold.
- Heart rate is calculated with each new pulse detected in the base of a counter incremented with each sample taken between successive peaks. This value is called Real Time Heart Rate because it is obtained immediately after the occurrence of a new pulse.
- The function "CalculateHeartRateMedian" is called to obtain a more reliable heart rate—one that considers not only the last value but the previous ones as well.
- The heart rate calculated in the last step is stored in a buffer. When the buffer is full, the data is sent to the PC via the communication protocol.

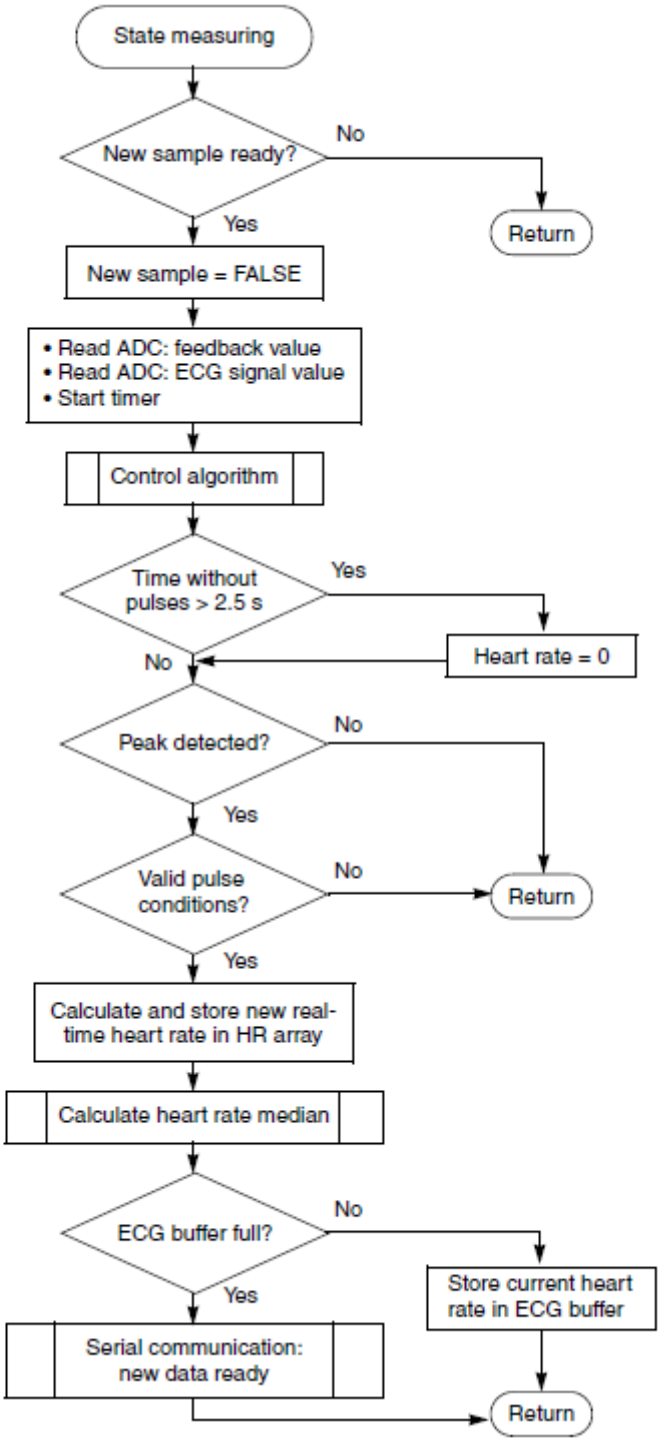


Figure 19. Measurement flow chart

4.5.2 Baseline acquisition

In order to provide a feedback signal and set a base level so the MED-EKG demo signal is stable on the screen of the GUI, the microcontroller first has to obtain the current approximate offset of the signal. To do this, the software performs an averaging algorithm that takes a certain number of samples (in this case 15), orders them, and obtains the median of the data.

Afterwards, the microcontroller decides between keeping the signal on the same level or modifying the voltage delivered to the MED-EKG board through the DAC output. Such process correspond to the function “PerformControlAlgorithm” and can be observed in Figure 20.

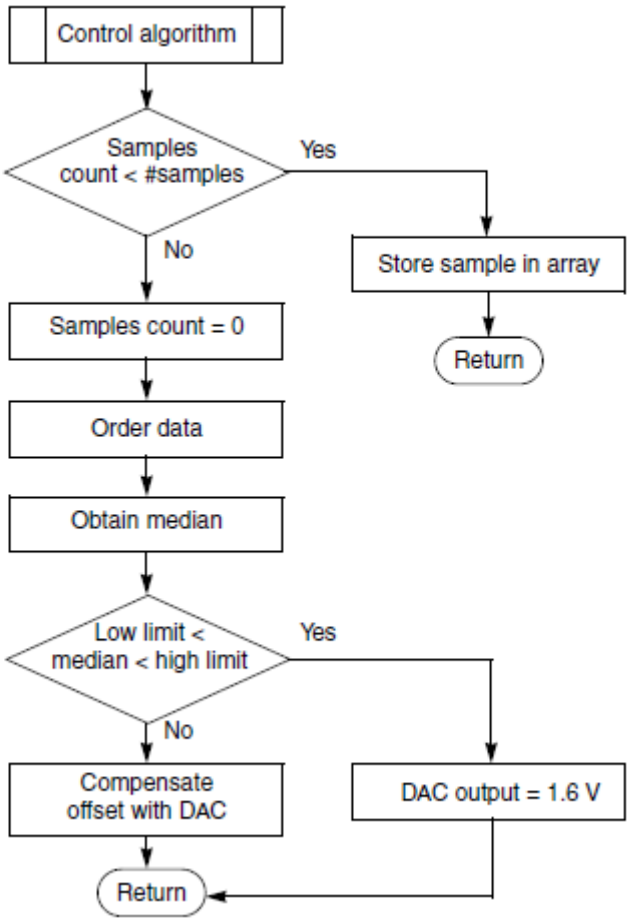


Figure 20. Baseline control flowchart

Below is a brief explanation of the algorithm:

- Each sample of the baseline signal is stored in an array.
- Once the determined quantity of samples have been stored in the array, the software orders the values within the same array in ascending order and obtains the middle, or median, value.
- The median of the values is compared with several ranges to determine whether or not it is necessary to change the feedback voltage and how much to change it.

4.5.3 Heart rate

It was already mentioned in State measuring, that heart rate is calculated with each sample of the ECG signal. That value corresponds to a real time heart rate, but it is not reliable because eventually some pulses take less or more time to occur, and the measurement could change drastically. For this reason, several successive heart rate indicators are averaged to obtain a more approximate value. The algorithm is similar to the one used to obtain the baseline, but in this case the heart rate median is updated every 4 peaks, although the last 18 pulses are considered. The algorithm corresponds to the software routine called “CalculateHeartRateMedian,” whose flow chart is in the Figure 21.

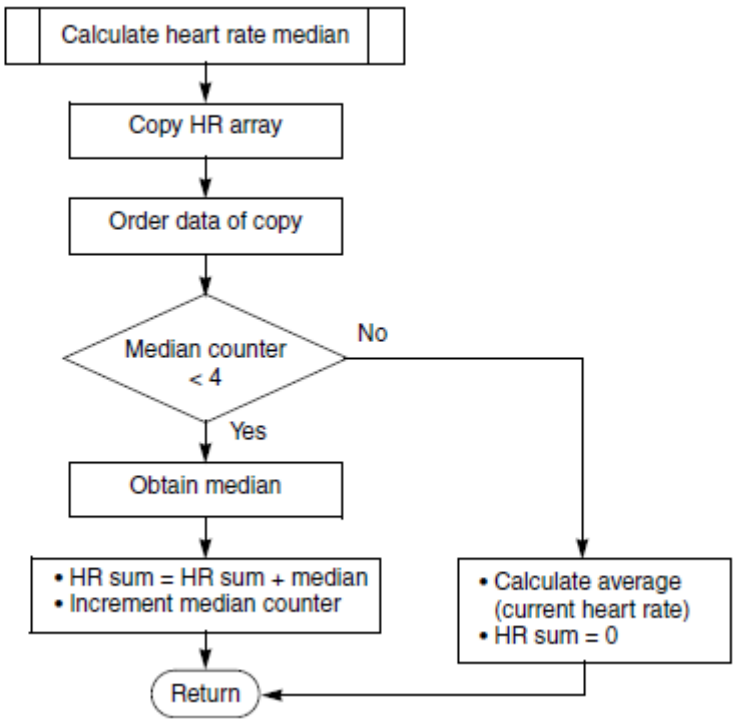


Figure 21. Heart rate flowchart

4.5.4 Gain control

The gain control algorithm allows the user to modify the amplitude of the signal shown on the GUI by pressing a pair of buttons featured in the tower board for each microcontroller. This action consists of changing the gain for one of the internal OPAMPs of the microcontroller. With these buttons, it is possible either to increase or decrease the gain of the operational amplifier so that the ECG signal is at a more convenient level. Software dedicated to this function consists simply in polling the signals coming from the buttons and changing the gain of the OPAMP with the appropriate register.

The following flowchart (Figure 22) depicts the gain control.

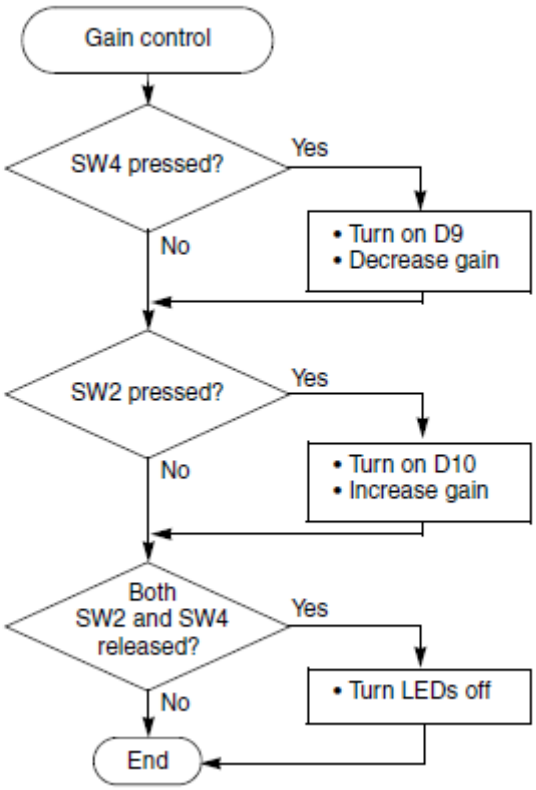


Figure 22. Gain control flowchart

The microcontroller is continuously checking for the state of the buttons, since it is part of an infinite loop, so the gain is updated at any time when the user presses one of the buttons.

4.5.5 Automatic gain algorithm

Automatic gain control (AGC) is an alternative to the buttons mentioned in Gain control. The algorithm consists of checking the amplitude of the ECG signal constantly to change the gain of the internal amplifier in real time if necessary. This offers a convenient alternative to pushing a button to maintain the signal between appropriate levels.

The basic algorithm consists of the following:

- There is a counter in the software that is incremented each time a sample of the ECG signal is taken. This counter serves to maintain a register of the time elapsed.
- If we consider a minimum heart rate of 40 bpm, a pulse should occur at least every 1.5 seconds. A window time of 1.2 seconds (50 bpm) is acceptable to detect a heartbeat since heart rate is generally higher than 40 bpm.
- Within the time window, we look for the lowest and highest sample of the ECG signal by constantly updating two registers that contain the current minimum and maximum values. Once the time window is over, the difference between those values determines the maximum peak-to-peak amplitude of the signal during that period.
- The software checks if the amplitude value is between the proper ranges; if it is not, the gain of the internal amplifier is changed either to increase or decrease it.

Figure 23 is a graphical representation of this algorithm.

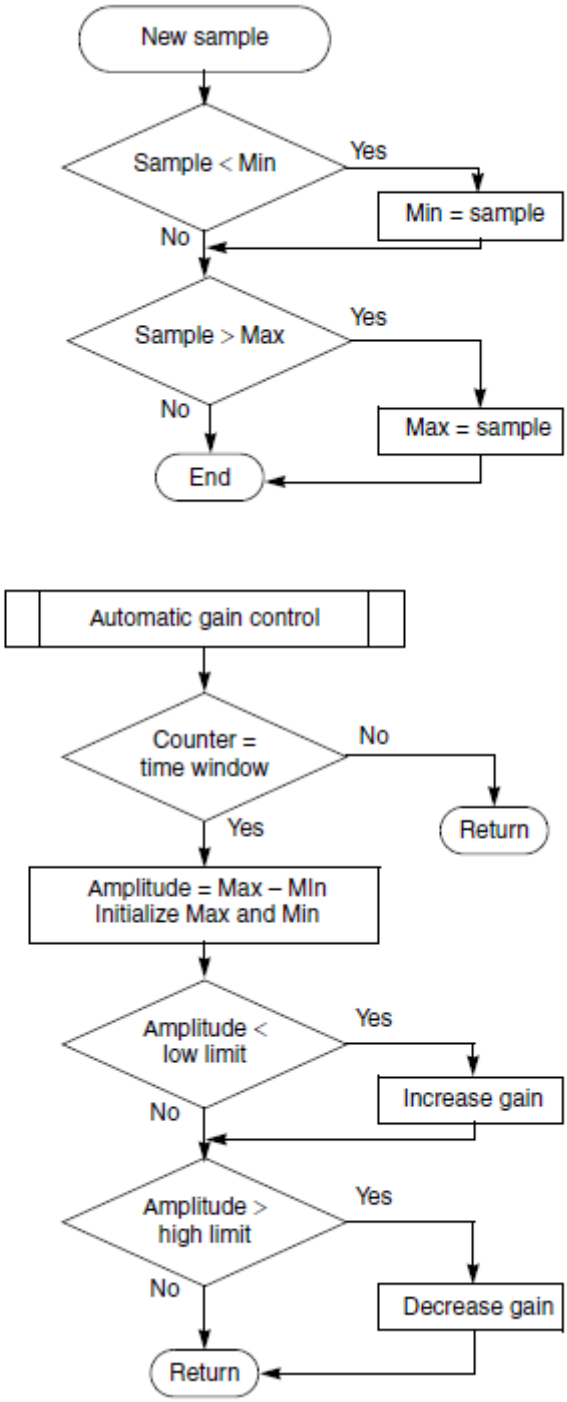


Figure 23. Automatic gain control flow chart

5 Getting started and running the MED-EKG demo

This section explains how to use the demo and the graphical user interface (GUI).

The GUI communicates with the microcontroller via USB port, using the Freescale USB stack, which performs an algorithm that creates a virtual serial port to send and receive data. For further reference and information about Freescale USB stack with PHDC please visit **freescale.com**.

5.1 Hardware settings and connections

The contents of this section are intended as a guide in the implementation and operation of the MED-EKG demo. Instructions are provided for both the Kinetis K50 and Flexis MM tower modules.

5.1.1 MED-EKG demo with Kinetis TWR-K53N512

The following steps have to be completed in order to run the MED-EKG demo in the TWR-K53N512:

1. Download the MEDICAL GUI from **freescale.com**. Follow the instructions included in the installer's ZIP file for GUI installation.
2. Download the IAR Embedded Workbench® from the IAR Systems® website (**iar.com**). The Kinetis demo software was developed using this platform. Install it on your computer.
3. Assemble the tower system with the elevators, TWR-SER, TWR-K53, and the MED-EKG analog front end. Be sure to match the primary and secondary sides of the microcontroller board with the respective elevators. (Refer to the jumper setting file to configure the necessary jumpers).

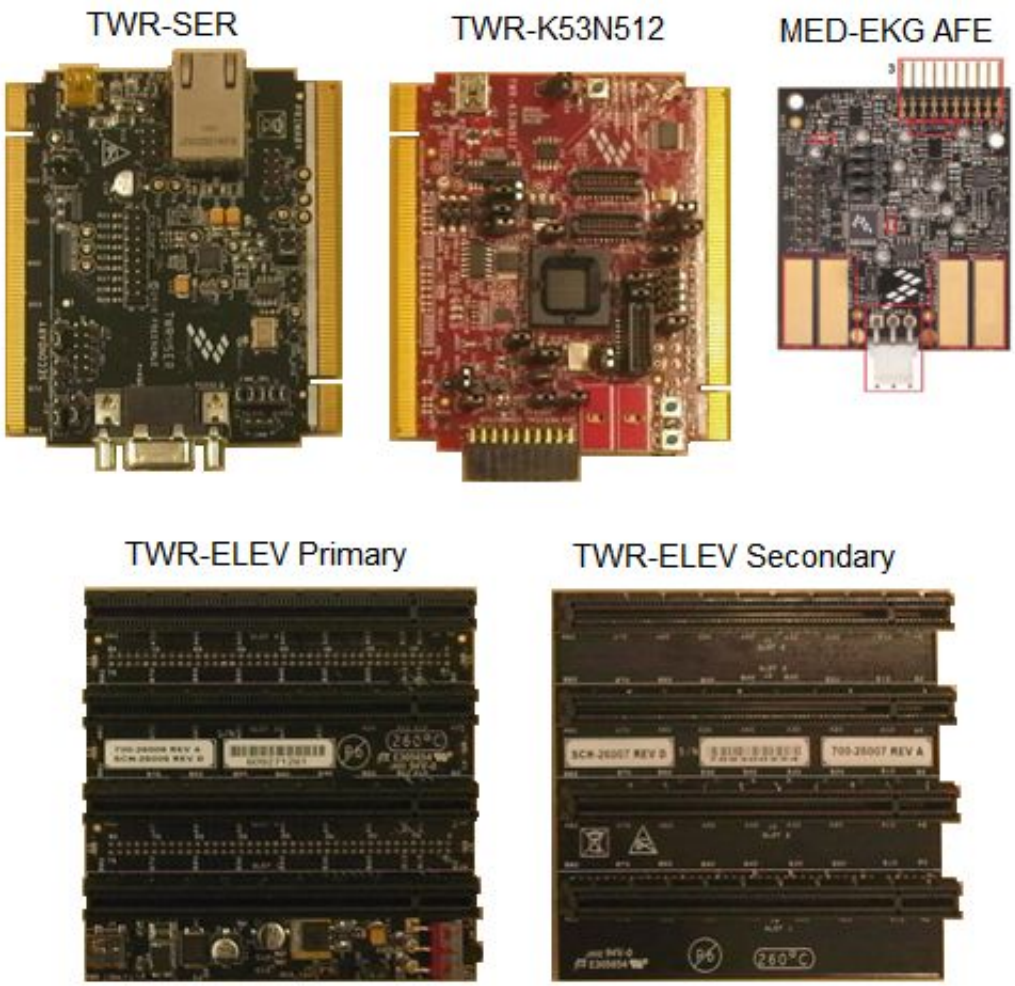


Figure 24. Parts of the tower to be assembled

4. Download and unzip AN4323.zip file. Open the IAR system and search for the project in the software folder. The name is USB_CDC.eww.
5. Connect the microcontroller board to the computer using a USB cable. The computer must recognize the Open Source BDM debug port. Install the driver for the OSBDM in the IAR folder.
6. Verify that the selected debugger is “PE micro” in the project options panel (Project/Options/Debugger) as shown in Figure 25.

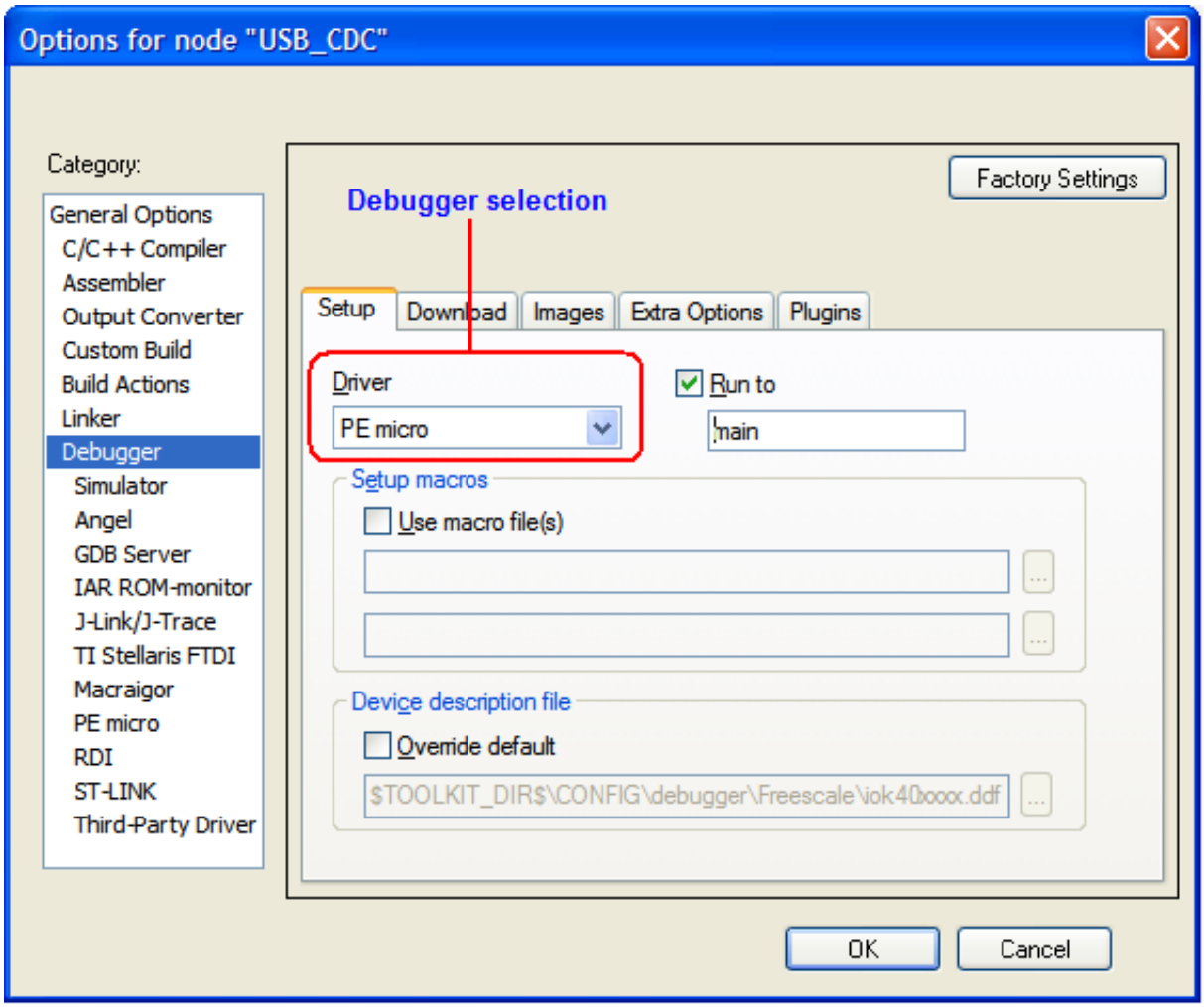


Figure 25. Selecting appropriate debugger

7. To load the microcontroller with the project, click on the "Download and debug" icon. The IAR screen of Figure 26 shows the location of the icon.

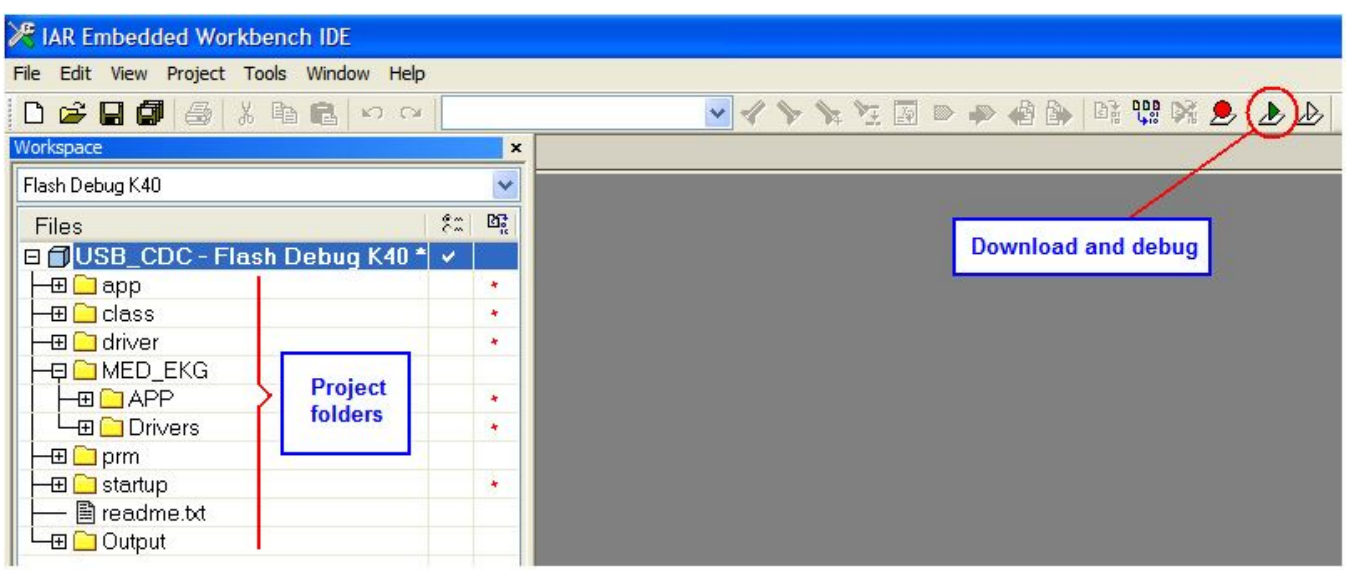


Figure 26. IAR screen

8. After programming the Kinetis K53, refer to Graphical user interface for the MED-EKG, for instructions on how to connect the electrodes and use the graphical interface.

5.1.2 MED-EKG demo with Flexis MM

The steps for the TWR-MCF51MM and the TWR-S08MM128 are listed below:

1. Download the MEDICAL GUI from **freescale.com**. Follow the instructions included in the installer's ZIP file for GUI installation.
2. Ensure that the host computer has CodeWarrior v6.3 and the latest service pack for MCF51MM256 (if necessary). Otherwise, download and install them.
3. Connect the modules of the tower system (elevators, TWR-SER, TWR-MCF51MM256 or TWR-S08MM128, and the MED-EKG board). The primary and secondary elevators need to match the corresponding sides of the microcontroller board.

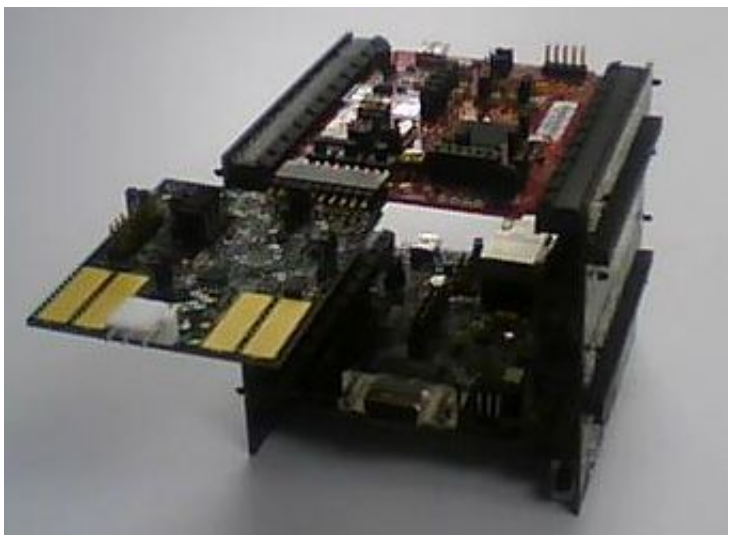


Figure 27. Tower system assembled for the MED-EKG demo

4. Using the USB ports, connect the serial and microcontroller modules from the tower system to the host computer. The first time you connect the tower to the PC, it will ask for the driver of the Open Source BDM. You can either choose the option “Install the software automatically” or specify the path to the driver, which is, by default: C:\Program Files \Freescale\CodeWarrior for Microcontrollers V6.3\Drivers\Osbdm-jm60.

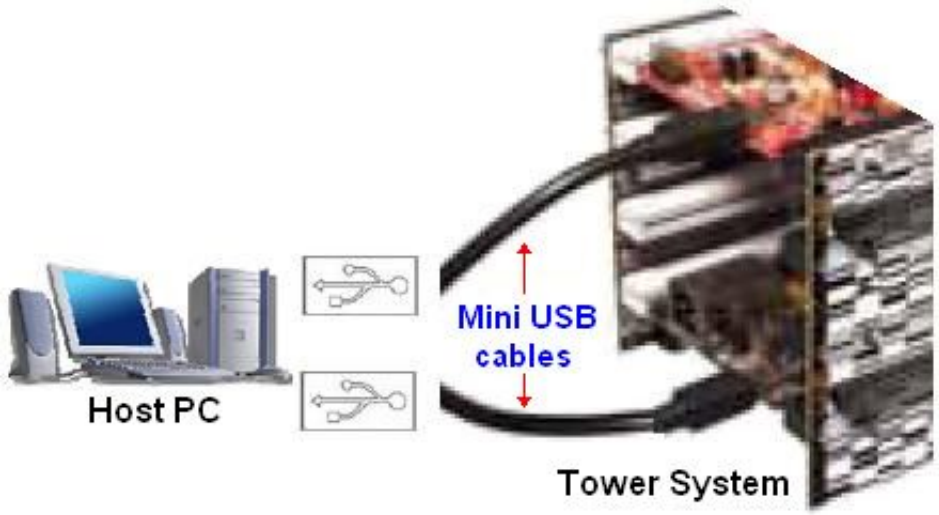


Figure 28. USB connections

5. Open CodeWarrior v6.3 and program the microcontroller with the “ECG for MCF51MM” or “ECG for S08MM” project, included in the compressed folder AN4323.zip. Figure 29 shows an example of how to do this if you are using the TWR-MCF51MM. For the TWR-S08MM128, the target must be “HCS08 FSL Open Source BDM.” Those options are placed on the left side of the CodeWarrior screen.

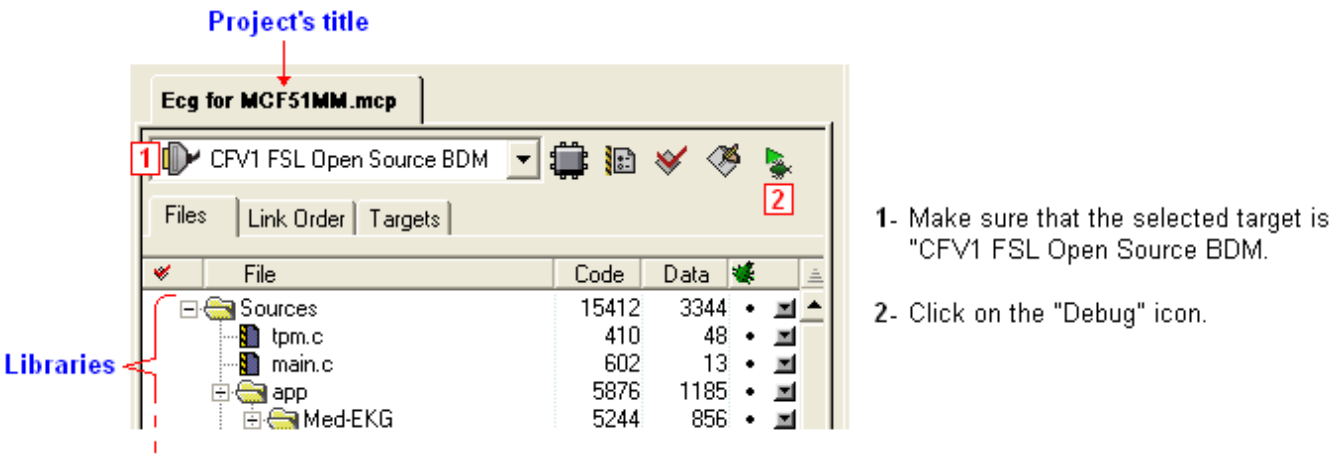


Figure 29. CodeWarrior programming interface

6. When the microcontroller has been programmed, refer to Graphical user interface for the MED-EKG, for instructions about the connection of the electrodes and the graphical user interface.

5.2 Graphical user interface for the MED-EKG

The graphical user interface shows the EKG results and the corresponding graph generated when performing a test. To run a test using the GUI, follow the steps explained below:

Follow the steps below to perform ECG measurement:

1. After the microcontroller is programmed and the TWR-SER module is connected to the computer via USB as shown in Figure 30, the next step is to connect the electrodes. Figure 31 and Figure 32 are an example of the correct way to do this.

Figure 30. Connecting the USB cable

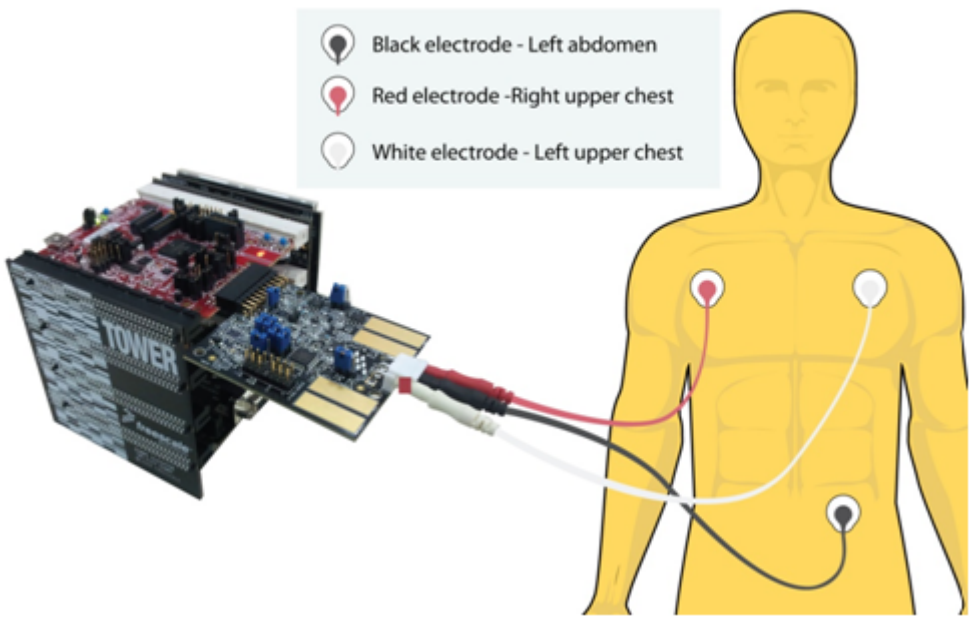


Figure 31. Electrode connection

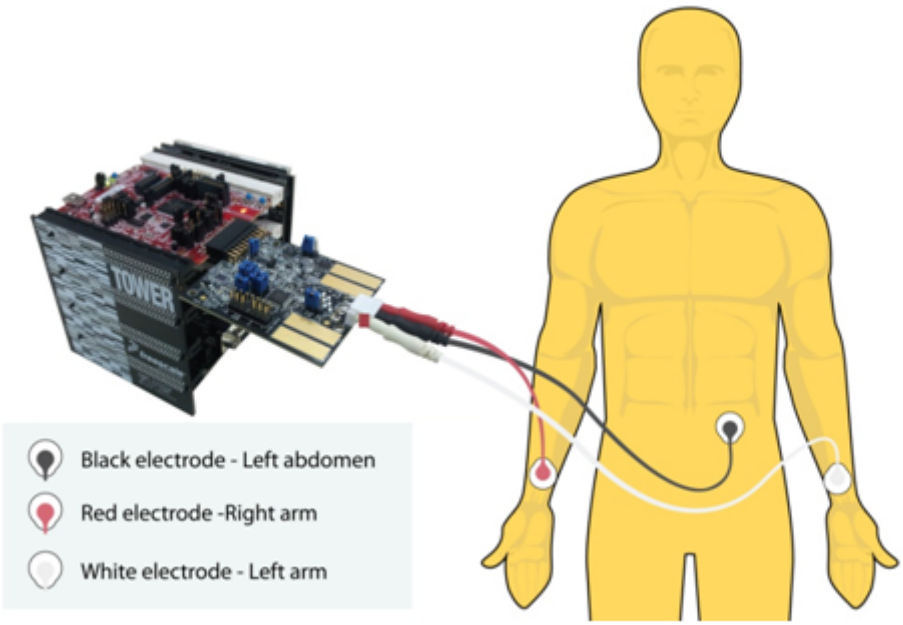


Figure 32. Electrode connection

2. Press the Reset button on the microcontroller board. When the demo is used for the first time, it is necessary to install the driver for the USB CDC Virtual COM. The default path for the driver is: *c:\Freescale\Medical GUI\Drivers*. Select x32 for 32-bit windows versions and x64 for 64-bit windows versions
3. Open the Windows Device Manager and search for the port assigned to the USB CDC virtual com. This information is requested by the GUI.
4. Open the graphical user interface (GUI). It rwill ask for a port number, obtained in step 3. Select the correct port and click OK. This is shown in Figure 33.

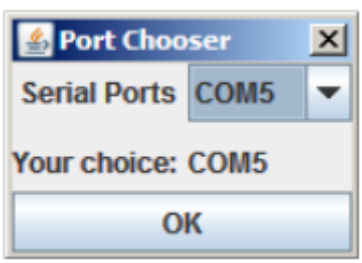


Figure 33. Selecting port

5. The main screen of the GUI (See Figure 34) appears. Make sure that Caps Lock key is not activated on your keyboard and press Shift + D. This starts the GUI doctor mode.

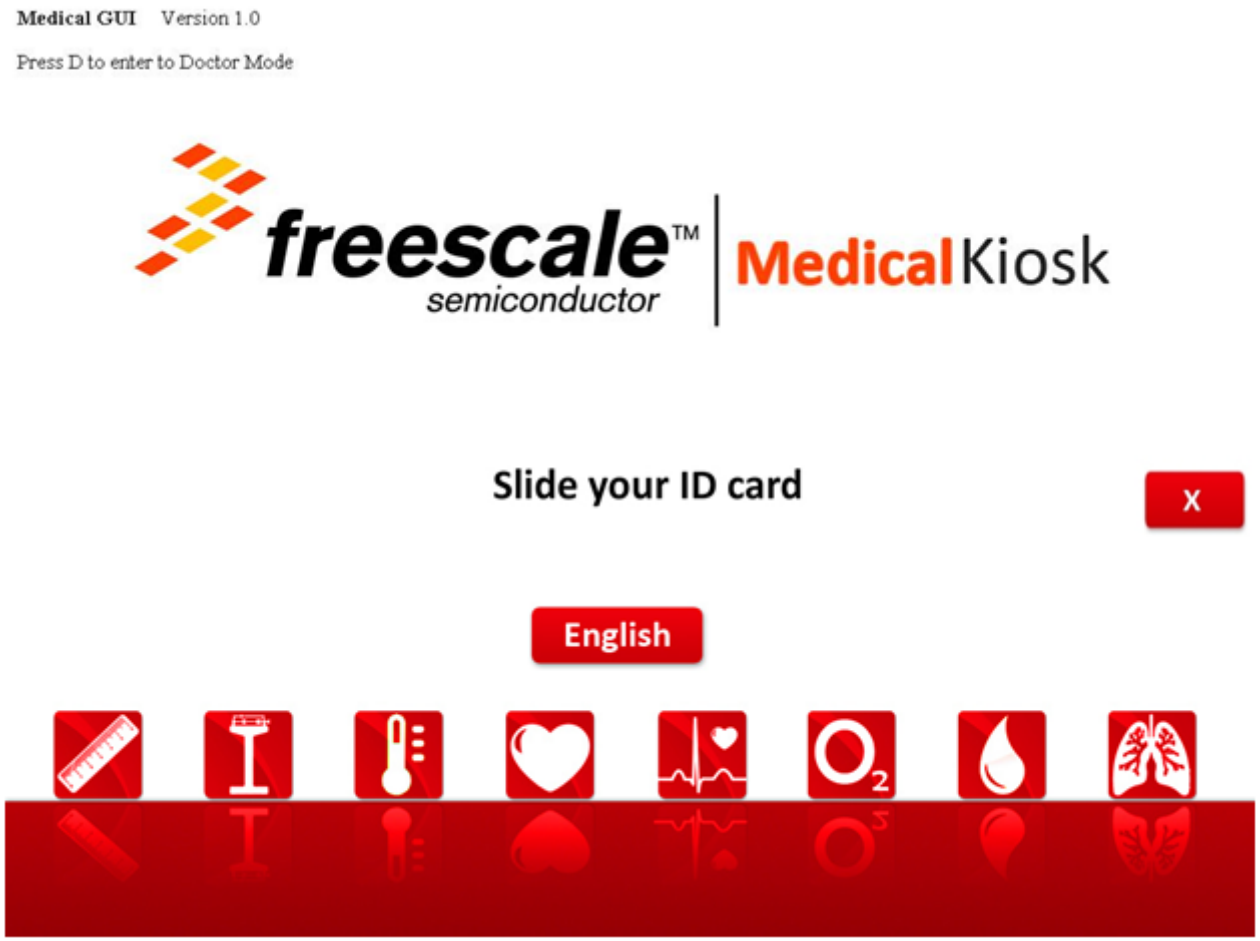


Figure 34. GUI main screen

6. In the doctor mode screen shown in Figure 35, select the ECG option.

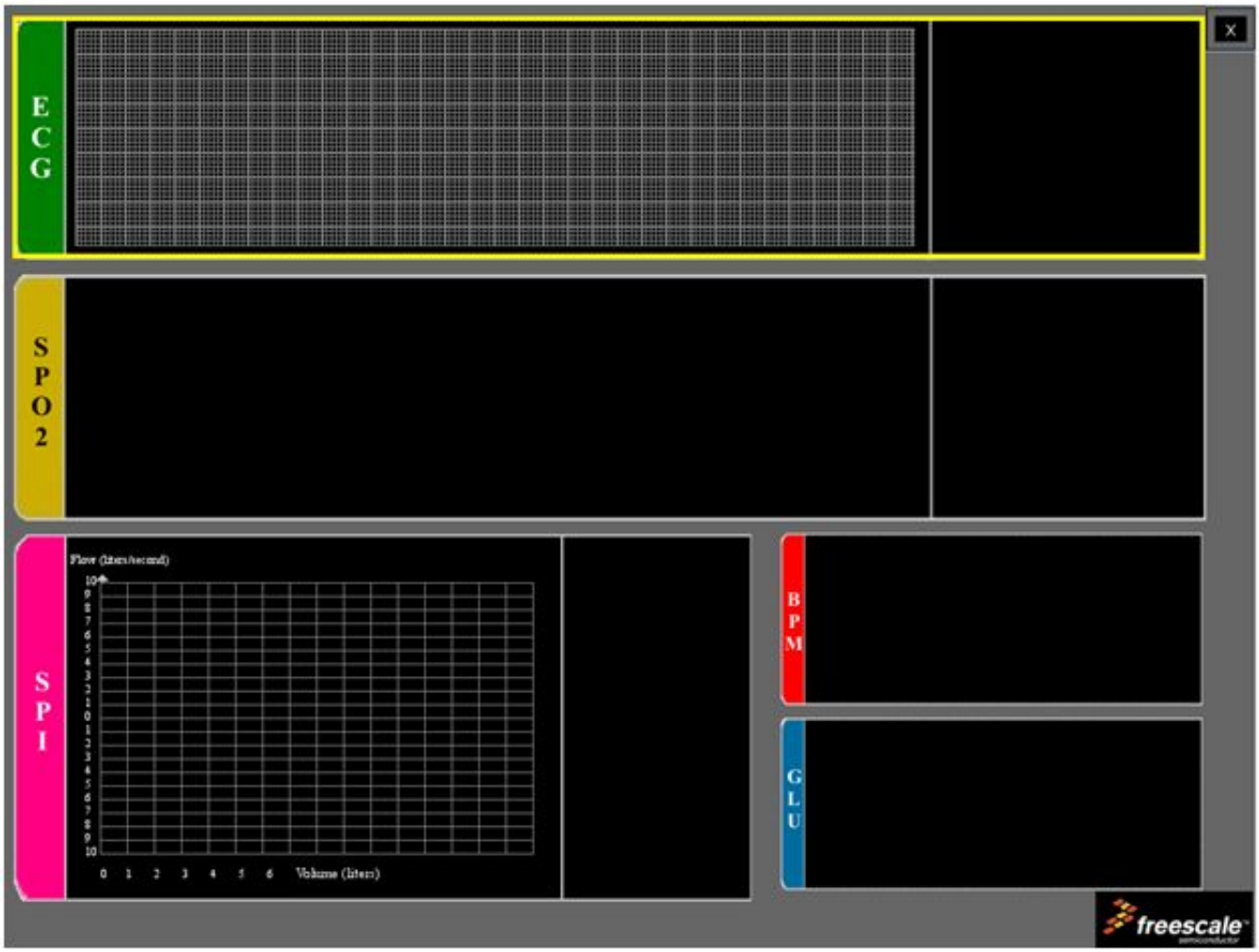


Figure 35. Doctor mode

7. Click the ECG area and the ECG signal will start being displayed. Wait for the signal to stabilize. You can see the electrocardiogram signal and the heart rate as well.
8. The ECG parameters and the graph are shown on the GUI. Figure 36 is an example of the results of a ECG test.

9

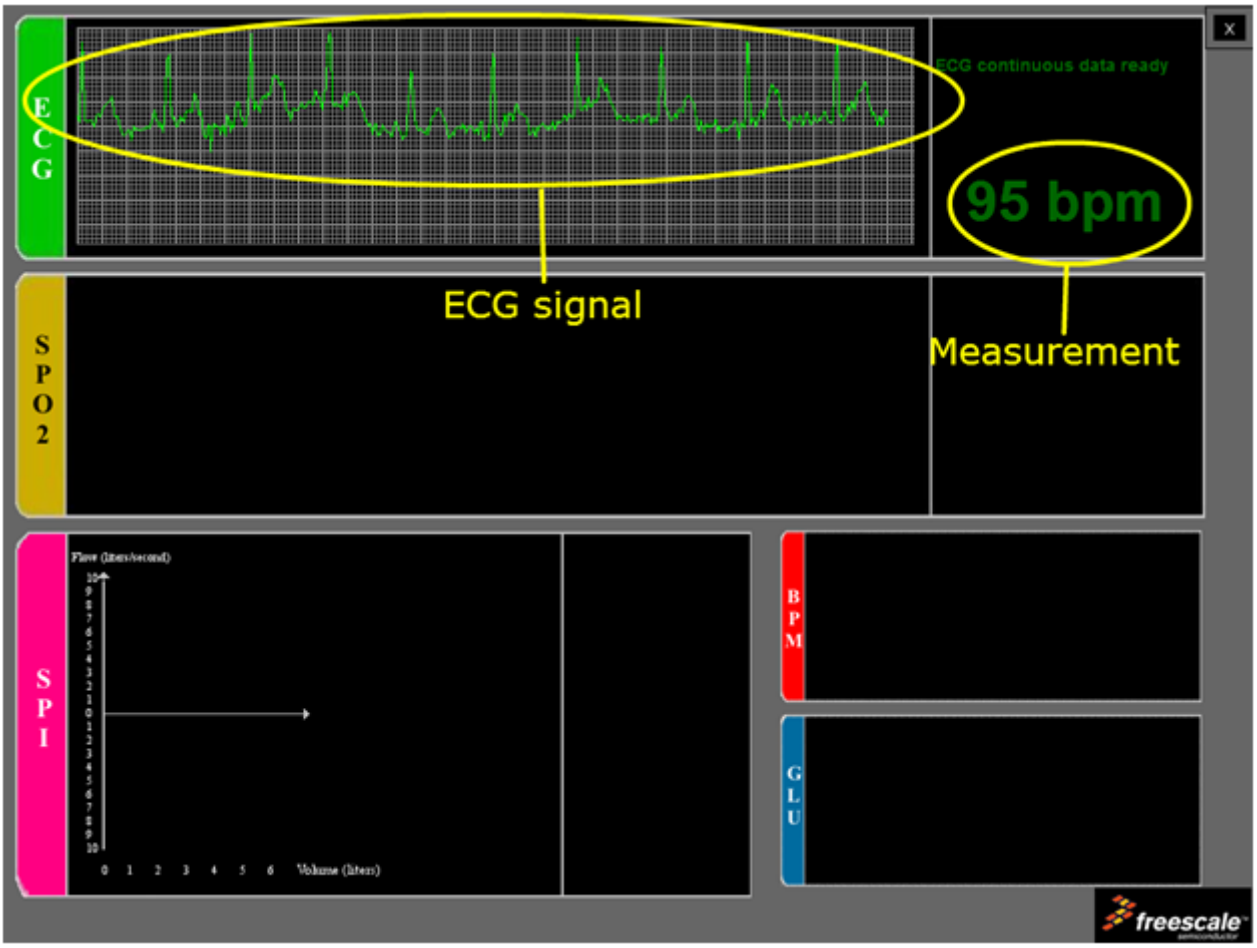


Figure 36. MED-EKG GUI results

6 Reference documents

In addition to this application note, the following documents are available at **freescale.com**:

- Reference manuals and data sheets for Kinetis K53 (K53P144M100SF2RM, K53P144M100SF2), MCF51MM256 (MCF51MM256RM, MCF51MM256), and MC9S08MM128 (MC9S08MM128RM, MC9S08MM128).
- MED-EKG Board User Manual (MED-EKGUG)
- Schematics of the Tower Modules and MED-EKG board
- Quick Start Guides
- USB Stack with PHDC User Guide (MEDUSBUG)

Appendix A Comparison between Freescale microcontrollers

The demo described in this document has been implemented with three microcontrollers: the MCF51MM256, the MC9S08MM128, and the Kinetis K50. The following table is a brief comparative analysis of those MCUs.

Table A-1. Freescale MCU comparison

Characteristic	MC9S08MM128	MCF51MM256	Kinetis K50
Core	HCS08 (8-bit) 48 MHz Core 24 MHz Bus	ColdFire V1 with MAC 8–32 bit compatibility 50.33 MHz Core 25.165 MHz Bus	ARM Cortex-M4 with MAC DSP+SIMD 72 MHz or 100 MHz
Flash / SRAM	128 KB Flash 12 KB RAM	256 KB Flash 32 KB RAM	64–512 KB Flash 64–128 KB RAM
A/D converter	16-bit SAR ADC	256 KB Flash 32 KB RAM	2 × 16-bit SAR ADC
D/A converter	12-bit DAC	12-bit DAC	2 × 12-bit DAC
Timers	2 × 4 channel, TPM with PWM	2 × 4 channel, TPM with PWM	3 × 12 channel Timer
Communication Interfaces	2 × SCI 2 × SPI USB Device I^2C	2 × SCI 2 × SPI USB Device/Host/OTG I^2C MiniFlexBus	6 x SCI 3 × SPI USB OTG Ethernet Secure Digital Host Controller (SDHC) FlexBus
Other	PRACMP 2 × OPAMP 2 × TRIAMP	PRACMP 2 × OPAMP 2 × TRIAMP	3 × ACMP 2 × OPAMP 2 × TRIAMP 2 × Programmable Gain Amplifier (PGA) Random Number Generator (RNG) LCD Driver Touch Sensing Interface

Appendix B Communication protocol

The application communicates with the Graphic User Interface (GUI) in the PC using the Freescale USB Stack with PHDC with the device acting as a CDC (Communication Device Class). The device communicates via a serial interface similar to the RS232 communications standard, but emulating a virtual COM port.

After the device has been connected and a proper driver has been installed, the PC recognizes it as a Virtual COM Port and it is accessible, for example using HyperTerminal. Communication is established using the following parameters.

- Bits per second—115,200
- Data bits—8
- Parity—None
- Stop Bits—1
- Flow Control — None

Communication starts when the host (PC) sends a request packet indicating to the device the action to perform. The device then responds with a confirmation packet indicating to host that the command has been received. At this point, the host must be prepared to receive data packets from the device and show the data received on the GUI. Communication finishes when the host sends a request packet indicating the device to stop. The following block diagram describes the data flow.

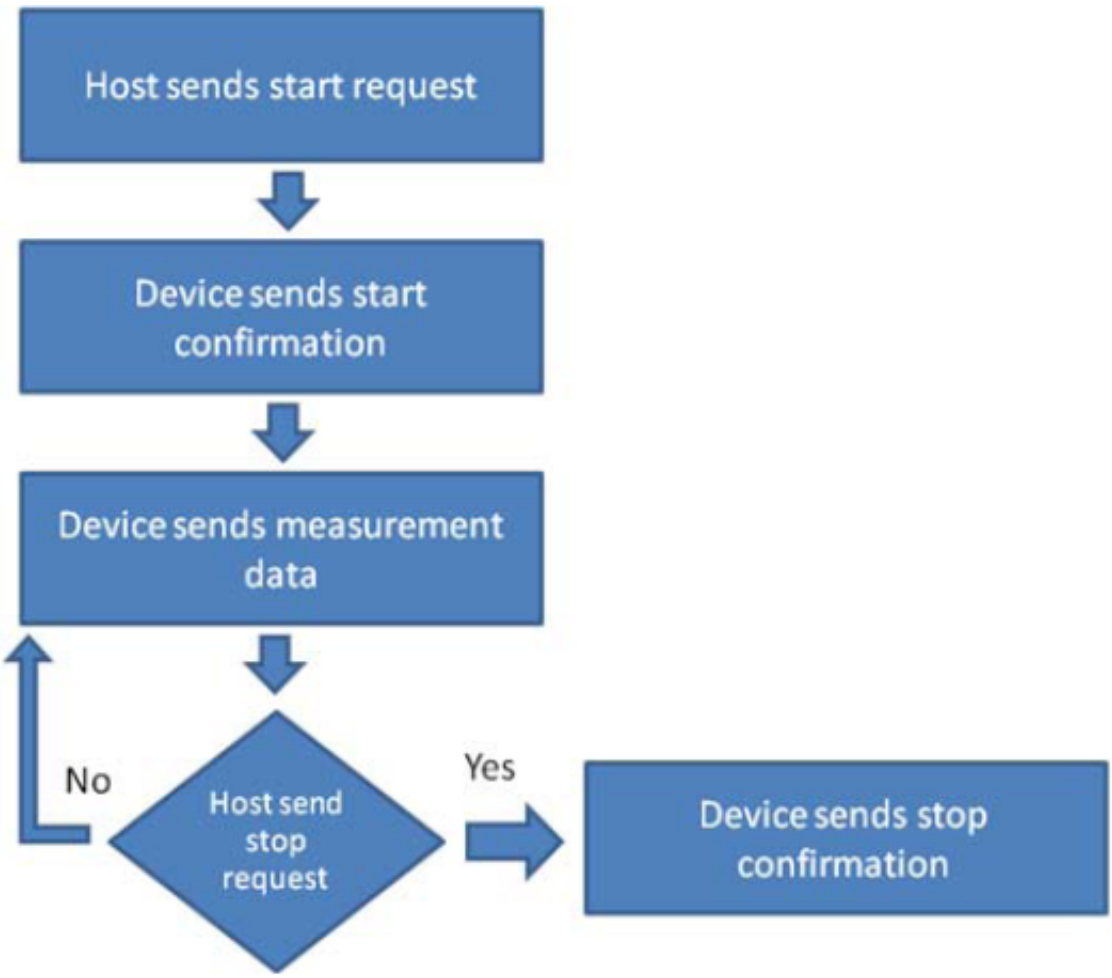


Figure B-1. Communication protocol data flow

Packets sent between host and device have a specific structure. The Packet is divided in four main parts:

- Packet Type
- Command Opcode
- Data length
- Data

The image below shows the packet structure.

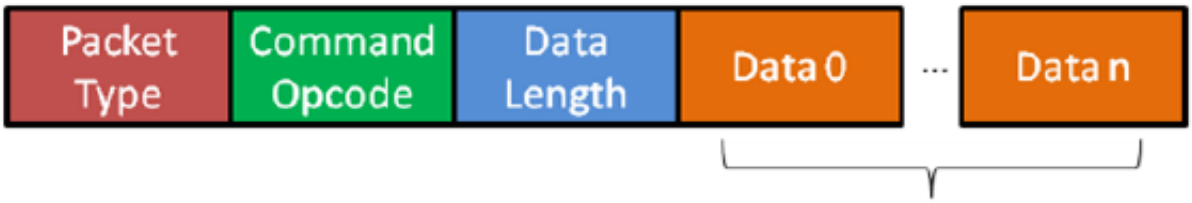


Figure B-2. Packet structure

B.1 Packet type

The Packet Type byte defines the kind of packet to be sent. There are three kinds of packets that can be sent between host and device.

B.1.1 REQ packet

This is a request packet, this kind of packet is used by the host to request to the device to perform some action like a start or stop measurement. A REQ packet is usually composed of 2 bytes, Packet Type and Command Opcode. Data Length and Data Packet bytes are not required.

B.1.2 CFM packet

This is a confirmation packet; this kind of packet is used by the device to confirm to the host that a command has been received, and sends a response indicating if the command is accepted, or if the device is busy.

B.1.3 IND packet

This is an indication packet. This kind of packet is used to indicate to the host that an event has occurred in the device and data needs to be sent. For example, this is used when the device has a new data to be sent to the GUI.

The following table shows the HEX codes for every Packet Type.

Table B-1. HEX codes

Packet type	Hex codes
REQ	0x52
CFM	0x43
IND	0x69

B.2 Command opcode

The Command Opcode byte indicates the action performed for a REQ packet, and the kind of confirmation or indication, in case of CFM and IND packet types. There are different Opcodes for every Packet Type. The following table shows the different Opcodes. See Note[1]

Table B-2. Opcodes

Opcode	REQ	CFM	IND	Opcodes (Hex)
Glucose meter				
GLU_START_MEASUREMENT	X	X		0x00
GLU_ABORT_MEASUREMENT	X	X		0x01
GLU_START_CALIBRATION	X	X		0x02
GLU_BLOOD_DETECTED			X	0x03
GLU_MEASUREMENT_COMPLETE_OK			X	0x04
GLU_CALIBRATION_COMPLETE_OK			X	0x05
Blood Pressure Meter				
BPM_START_MEASUREMENT	X	X		0x06
BPM_ABORT_MEASUREMENT	X	X		0x07
BPM_MEASUREMENT_COMPLETE_OK			X	0x08
BPM_MEASUREMENT_ERROR			X	0x09
BPM_START_LEAK_TEST	X	X		0x0A
BPM_ABORT_LEAK_TEST	X	X		0x0B
BPM_LEAK_TEST_COMPLETE			X	0x0C
BPM_SEND_PRESSURE_VALUE_TO_PC			X	0x28
Electro Cardiograph Opcode				
ECG_HEART_RATE_START_MEASUREMENT	X	X		0x0D
ECG_HEART_RATE_ABORT_MEASUREMENT	X	X		0x0E
ECG_HEART_RATE_MEASUREMENT_COMPLETE_OK			X	0x0F
ECG_HEART_RATE_MEASUREMENT_ERROR			X	0x10
ECG_DIAGNOSTIC_MODE_START_MEASUREMENT	X	X		0x12
ECG_DIAGNOSTIC_MODE_STOP_MEASUREMENT	X	X		0x13
ECG_DIAGNOSTIC_MODE_NEW_DATA_READY			X	0x14
Thermometer				
TMP_READ_TEMEPRATURE	X	X		0x15
Height scale				
HGT_READ_HEIGHT	X	X		0x16
Weight scale				
WGT_READ_WEIGHT	X	X		0x17
Spirometer				
SPR_DIAGNOSTIC_MODE_START_MEASURMENT	X	X		0x0C
SPR_DIAGNOSTIC_MODE_STOP_MEASURMENT	X	X		0x0D
SPR_DIAGNOSTIC_MODE_NEW_DATA_READY			X	0x0E
Pulse oximetry				
SPO2_START_MEASURMENT	X	X		0x21

Table continues on the next page...

1. Software related with this application note does not respond to all of these commands.

Table B-2. Opcodes (continued)

Opcode	REQ	CFM	IND	Opcodes (Hex)
SPO2_ABORT_MEASURMENT	X	X		0x22
SPO2_MEASURMENT_COMPLETE_OK			X	0x23
SPO2_MEASURMENT_ERROR			X	0x24
SPO2_DIAGNOSTIC_MODE_START_MEASURMENT	X	X		0x25
SPO2_DIAGNOSTIC_MODE _STOP_MEASURMENT	X	X		0x26
SPO2_DIAGNOSTIC_MODE_NEW_DATA_READY			X	0x27
				0x21
System commands				
SYS_CHECK_DEVICE_CONNECTION	X	X		0x29
SYS_RESTART_SYSTEM	X			0x2A

B.3 Data length and data string

The data length and data string bytes are the data quantity count and the data itself. The data length byte represents the number of bytes contained into the data string. The data string is the information sent, just the data, therefore the Data Length byte must not count the Packet Type byte, the Command Opcode byte or itself.

B.4 Functional description

Communication starts when the host sends a REQ packet indicating to the device to start a new measurement. The host must send a REQ Packet Type to start transactions (Figure B-3).

Figure B-3. Start packet sent by host

The Start Opcode can be any Opcode related with start a measurement, for example, if we wanted to start the ECG in diagnostic mode, the Data Packet will look like Figure B-4.

Figure B-4. Starting ECG in diagnostic mode

0x52 is the HEX code for a request (REQ) Packet Type, 0x12 corresponds to ECG_DIAGNOSTIC_MODE_START_MEASUREMENT. After sending the REQ packet, a CFM packet must be received indicating the status of the device. The received packet must look like Figure B-5.

Figure B-5. Confirmation packet structure

The error byte indicates the device status. The table below shows possible error codes.

Table B-3. Error codes

Error	HEX code
OK	0x00
BUSY	0x01
INVALID OPCODE	0x02

If the error byte received corresponds to OK, the device starts sending data as soon as a new data packet is ready. If BUSY error is received, the host must try to communicate later. If the error received is INVALID OPCODE, data sent and transmission lines must be checked.

If a CFM packet with an OK error has been received, the device starts sending Information related with the measurement requested. This is performed using indication packets (IND). Indication packet structure is shown in Figure B-6.

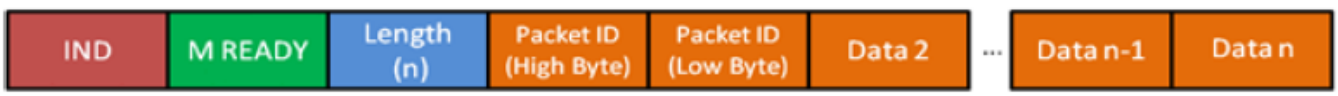


Figure B-6. Indication packet structure

The first byte contains the HEX code for an Indication Packet type. The second byte contains the Opcode for the kind of indication, for example if the device is sending an Indication Packet for ECG_DIAGNOSTIC_MODE_NEW_DATA_READY, the HEX code read in this position is 0x14 because this is the Indication Opcode for a new set of data from the ECG diagnostic mode. The next byte is the Length which indicates the quantity of data sent.

The first couple of bytes after the Length byte are the Packet ID bytes. The Packet ID is a 16-bit data divided in 2 bytes to be sent and contains the number of packets sent. The Packet ID number of a data packet is the Packet ID of the previous packet + 1. For example, if the Packet ID of the previous packet sent was 0x0009, the Packet ID of the next packet must be 0x000A. This allows the GUI to determine if a packet is missing.

The following data bytes are the Data String and contain the information of the measurement requested. The Data quantity is determined by the Data Length byte and data is interpreted depending on the kind of measurement. For example, for the MED-EKG Demo from Data 2 to Data n-1 contains the data graphed. Every point in the graph is represented by a 16-bit signed number, this means that every 2 data bytes in the packet, means it is one point in the graph. The first byte is the most significant part of the long (16-bits) and the second byte is the less significant part. The long is signed using 16-bit complement. The last byte contains the Heart Rate measurement. This byte must be taken as it is, an unsigned char data that contains the number of beats per minute. Figure B-7 shows a typical MED-EKG demo indication packet.

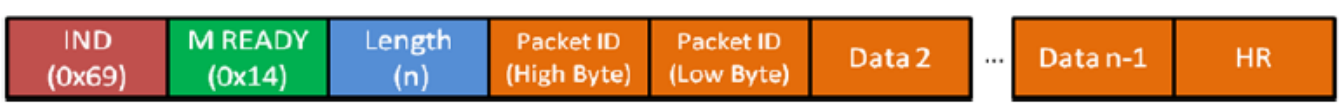


Figure B-7. MED-EKG IND packet

When a Stop request is sent by the host, the device stops sending data and waits for a new Start request command. Figure B-8 shows the Stop Command structure.

Figure B-8. Stop command structure

Immediately after this, the device must acknowledge with a CFM packet shown in Figure B-9.

Figure B-9. Stop CFM packet

The CFM packet for stop does not require an error code, it just must be received. If this packet has not been received, the request has been rejected or not taken and must be sent again to stop the measurements.

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or +1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor China Ltd.
Exchange Building 23F
No. 118 Jianguo Road
Chaoyang District
Beijing 100022
China
+86 10 5879 8000
support.asia@freescale.com

Document Number: AN4323
Rev. 1, 04/2013

freescale