ARDUINO SHIELD EKG/EMG/ECG/EEG

Description:

This is an EKG/EMG shield which allows Arduino like boards to capture Electrocardiography Electromiography signals. The shield opens new possibilities to experiment with bio feedback. You can monitor your heartbeat and log your pulse, recognize gestures by monitoring and analize the muscule activity.

Features:

- Stackable headers up to 6 channels may be stacked and wired to A0-A6 analogue inputs.
- Calibration signal generation by D4/D9 digital output.
- Precise Trimmer potentiometer for calibration.
- Input connector for normal or Active electrodes.
- Works with both 3.3V and 5V Arduino boards.

Hardware:

- SHIELD-EKG-EMG schematic in PDF format released under Creative Commons Attribution-Share Alike 3.0 United States License.
- SHIELD-EKG-EMG schematic and board in Eagle format released under Creative Commons Attribution-Share Alike 3.0 United States License.

Software:

- Electric guru monitoring software.
- Arduino example for EKG capture and interface to Electric Guru for OLIMEXINO-328/Arduino boards
- Maple example for EKG capture and interface to Electric Guru for OLIMEXINO-STM32.
- Pinguino example for EKG capture and interface to Electric Guru for PIC32-PINGUINO/OTG/MX220.

SECTION 2 SETTING UP THE SHIELD-EKG-EMG

This section helps you set up the SHIELD-EKG-EMG development board for the first time.

Please consider first the electrostatic warning to avoid damaging the board, then discover the hardware and software required to operate the board.

The procedure to power up the board is given, and a description of the default board behavior is detailed.

2.1 Electrostatic Warning

The SHIELD-EKG-EMG development board is shipped in a protective anti-static package. The board must not be exposed to high electrostatic potentials. A grounding strap or similar protective device should be worn when handling the board. Avoid touching the component pins or any other metallic element.

2.2 Requirements

In order to set up the SHIELD-EKG-EMG prototype board, the following items are required:

- SHIELD-EKG-EMG itself
- ARDUINO compatible board* (e.g. OLIMEXINO-328, OLIMEXINO-STM32, PIC32-PINGUINO)
- Electrode cable**

*The pinout for the board strictly follows the DUINO extension specification. Best choice for a board would be OLIMEXINO-328 or any DUINO board which utilizes the ATmega328 since we have working and tested examples for those MCUs, configuring them for ATmega128 would require tweaking of the code.

**The cable features three electrodes – two data electrodes (1 channel) and DLR electrode (feedback). If you use more than one SHIELD-EKG-EMG you can use cables without DLR for every shield after the first.

IMPORTANT NOTE: The electrode cables for SHIELD-EKG-EMG and MOD-EEG-SMT are different and not compatible!

There are two electrode cables that we have tested with the board. They are named SHIELD-EKG-EMG-PA and SHIELD-EKG-EMG-PRO. The first one is considered open-hardware and its schematics might be used as a reference if you wish to make the cable yourself.

The SHIELD-EKG-EMG-PRO works with different set of attachment cups that makes it easier to measure EMG signals at hard-to-reach spots and also allow contact materials replacement.

Links to the web-pages of the electrodes:
- https://www.olimex.com/Products/Duino/Shields/SHIELD-EKG-EMG-PA/open-source-

hardware

- https://www.olimex.com/Products/Duino/Shields/SHIELD-EKG-EMG-PRO/

Different pads for SHIELD-EKG-EMG-PRO electrode:

- https://www.olimex.com/Products/Modules/Biofeedback/ECG-GEL-ELECTRODE/
- https://www.olimex.com/Products/Duino/Shields/ECG-CLIP/
- https://www.olimex.com/Products/Duino/Shields/ECG-SCUP/

In case you want to build a working SHIELD-EKG-EMG system without owning a DUNIO board the best choice would be our board OLIMEXINO-328. This is the web page for the board: https://www.olimex.com/Products/Duino/AVR/OLIMEXINO-328/.

2.3 Powering up the board

The SHIELD-EKG-EMG board is powered by the host board it is mounted on. There is the option to be powered either by 3.3V or 5.0V host board (configured easily by a jumper).

On powering the board PWR LED must become RED.

2.4 Arduino/Maple/Pinguino note

What is Arduino?

Arduino is an open-source electronics prototyping platform, designed to make the process of using electronics in multidisciplinary projects more accessible. The hardware consists of a simple open hardware design for the Arduino board with an Atmel AVR processor and on-board I/O support. The software consists of a standard programming language and the boot loader that runs on the board.

Arduino hardware is programmed using a Wiring-based language (syntax + libraries), similar to C++ with some simplifications and modifications, and a Processing-based Integrated Development Environment (IDE).

The project began in Ivrea, Italy in 2005 aiming to make a device for controlling student-built interaction design projects less expensively than other prototyping systems available at the time. As of February 2010 more than 120,000 Arduino boards had been shipped. Founders Massimo Banzi and David Cuartielles named the project after a local bar named Arduino. The name is an Italian masculine first name, meaning "strong friend". The English pronunciation is "Hardwin", a namesake of Arduino of Ivrea.

More information could be found at the creators web page http://arduino.cc/ and in the Arduino Wiki http://en.wikipedia.org/wiki/Arduino.

To make the story short – Arduino is easy for beginners who lack Electronics knowledge, but also does not restrict professionals as they can program it in C++ or mix of Arduino/C++ language.
There are thousands of projects which makes it easy to startup as there is barely no field

where Arduino enthusiasts to have not been already.

Arduino has inspired two other major derivatives – MAPLE and PINGUINO. Based on 8-bit AVR technology the computational power of Arduino boards is modest, this is why a team from MIT developed the MAPLE project which is based on ARM7 STM32F103RBT6 microcontroller. The board have same friendly IDE as Arduino and offers the same capabilities as hardware and software but runs the Arduino code much faster. The Maple project can be found at http://leaflabs.com

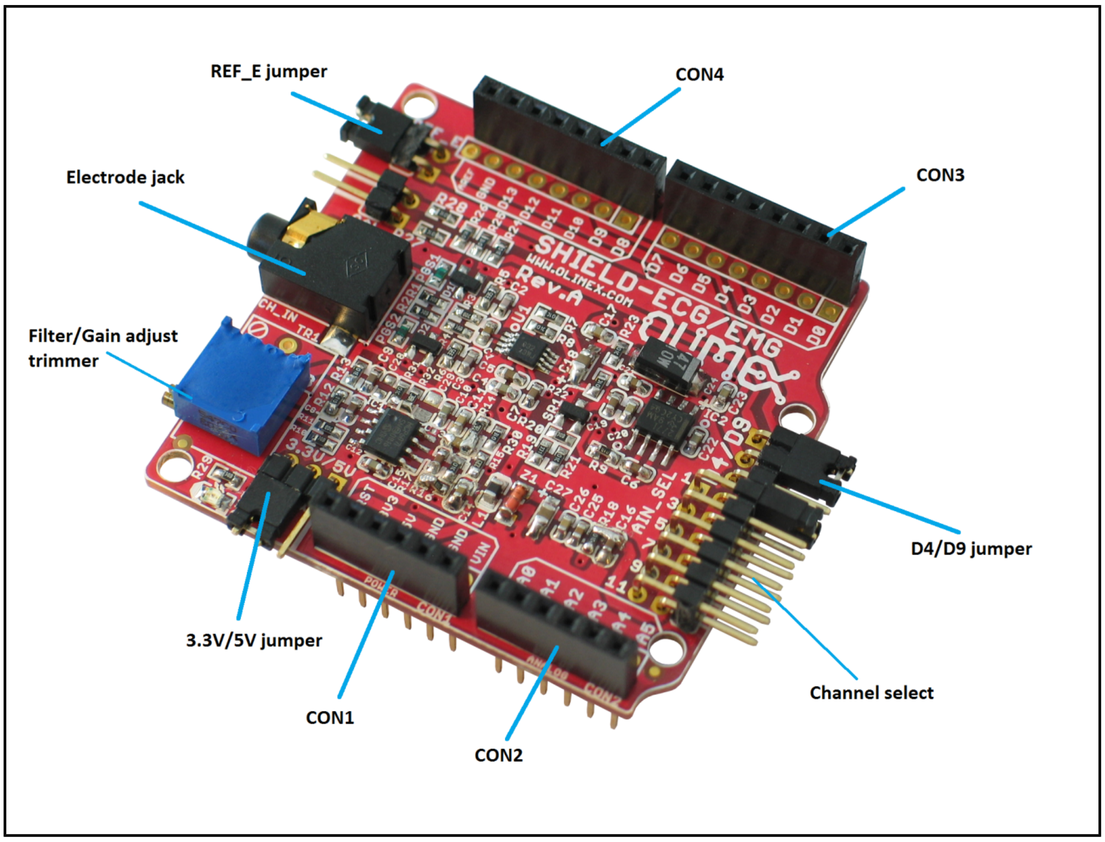
In parallel with Arduino another project was started called PINGUINO. This project chose its first implementation to be with PIC microcontrollers, as AVRs were hard to find in some parts of the world like South America so it is likely to see lot of PINGUINO developers are from that part of the world. PINGUINO project founders decided to go with Python instead Java for processing language. For the moment PINGUINO is much more flexible than Arduino as it is not limited to 8bit microcontrollers. Currently the IDE, which has GCC in background, can support 8-bit PIC microcontrollers, 32bit PIC32 (MIPS) microcontrollers and ARM7/CORTEXM3 microcontrollers which makes PINGUINO very flexible because once you make your project you can migrate easily through different hardware platforms and not being bound to a single microcontroller manufacturer. The PINGUINO project can be found at: http://www.pinguino.cc.

SECTION 3 SHIELD-EKG-EMG BOARD DESCRIPTION

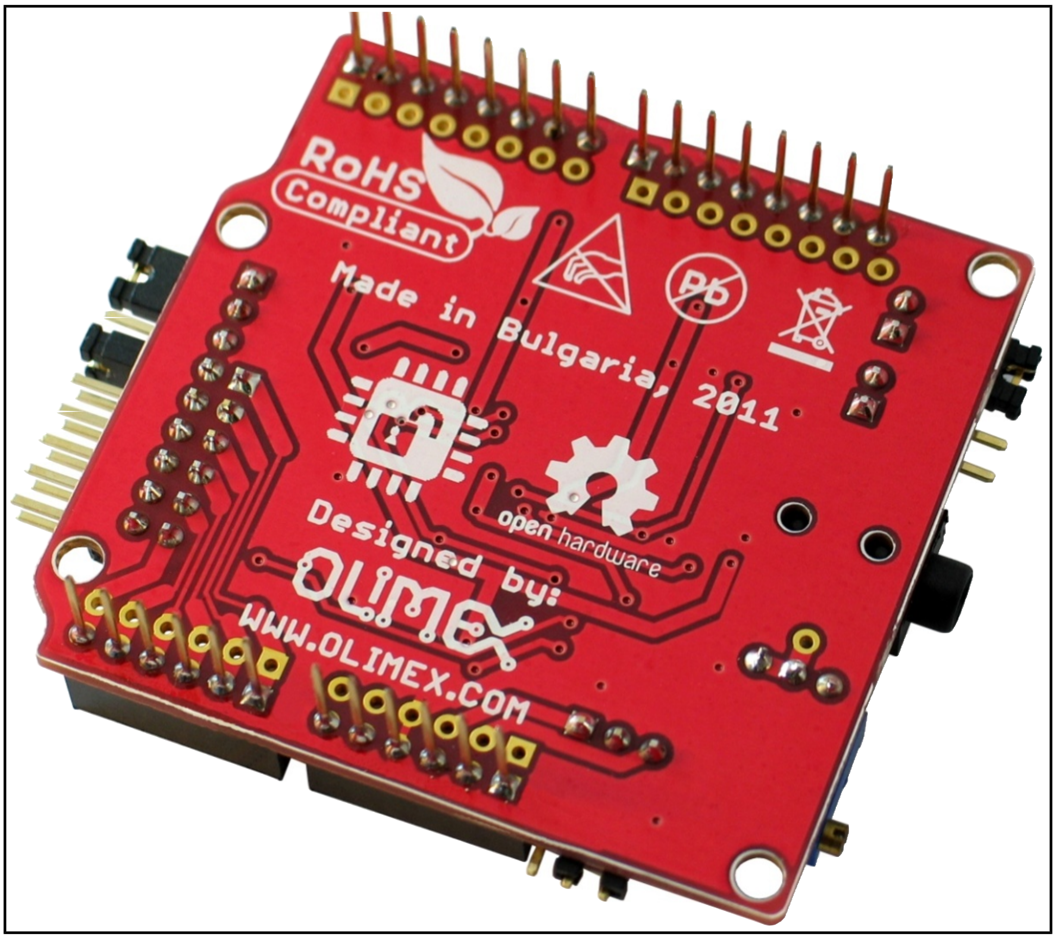
Here you get acquainted with the main parts of the board. Note the names used on the board differ from the names used to describe them. For the actual names check the SHIELD-EKG-EMG board itself.

For example: BUTTON (seen on the op view below) is named BUT; RESET is named RST; etc

3.1 Layout (Top view):



3.2 Layout (Bottom view):



SECTION 4 INSTALLATION EXAMPLE

This is a step by step example of installing SHIELD-EKG-EMG on OLIMEXINO-328 using Windows. You can refer to the tips keeping in mind that the example utilizes a board with ATmega328 MCU.

4.1 SHIELD-EKG-EMG and OLIMEXINO-328

In this example we use OLIMEXINO-328; SHIELD-EKG-EMG; USB – USB mini cable; Arduino 1.0 IDE; two external libraries for the IDE (the latest versions of TimerOnev9 and FlexiTimer2); the latest FTDI VCP drivers (2.08.14), demo code provided by us that can be downloaded from the web site and free monitoring software Electric Guru.

0. **IMPORTANT!** Before starting you have to prepare the OLIMEXINO-328 board by removing its wire between the R6 pads (R6 is not mounted). There is increased power consumption and a chance of electrical failure if you omit to do so. This is only for the OLIMEXINO-328 board! Don't do the modification if you lack basic electronics skills, because it is possible to damage the board.

If the host board provides reference voltage on the AREF pin then open REF_E jumper. We don't want two supplies to provide power to the same line – the 3V_REF_ADC one.

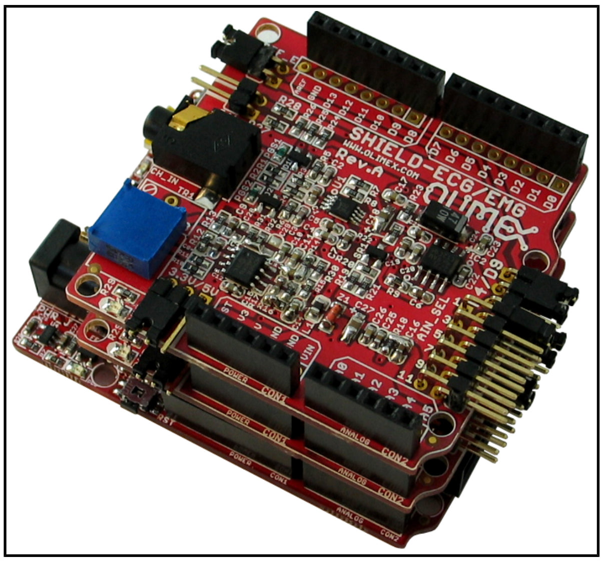
1. Download and extract the Arduino 1.1 IDE package from the Arduino web site: http://www.arduino.cc/
2. Download and place the two timer libraries required (TimerOne, FlexiTimer2) in \arduino-1.0\libraries by placing each of them in properly named folder (check the other libraries for reference)
http://arduino.cc/playground/Code/Timer1
http://www.pjrc.com/teensy/td_libs_MsTimer2.html
3. Download the demo project from product's web page:
https://www.olimex.com/Products/Duino/Shields/SHIELD-EKG-EMG/
4. Download the FTDI VCP drivers: http://www.ftdichip.com/Drivers/VCP.htm

5. Set the jumpers of SHIELD-EKG-EMG in the following way:

REF_E – closed
3.3V/5V – 5V position
D4/D9 – D9 position
ANI_SEL – 1 position (channel)

6. Connect the shield to the board

7.Connect OLIMEXINO-328 to the USB



8. Install the VCP FTDI drivers by going in *Device Manager*; right-clicking over the

unrecognized device and choosing *Update Driver* and then pointing to the folder where you downloaded and extracted the FTDI VCP driver.
Here it is advisable to go to *Device Manager* and from advanced settings of our recognized USB Serial Port (COMx) device to set x to a free port between 1 and 4 (because the monitoring software in this example can read only from COM ports 1 to 4).

9. Start Arduino IDE and open the provided by Olimex project ShieldEkgEmgDemo.pde

10. Set Tools -> Board -> Arduino Duemilanove w/ ATmega328
Set Tools -> Serial port -> the COM we configured our board at

11. Click Upload (→)

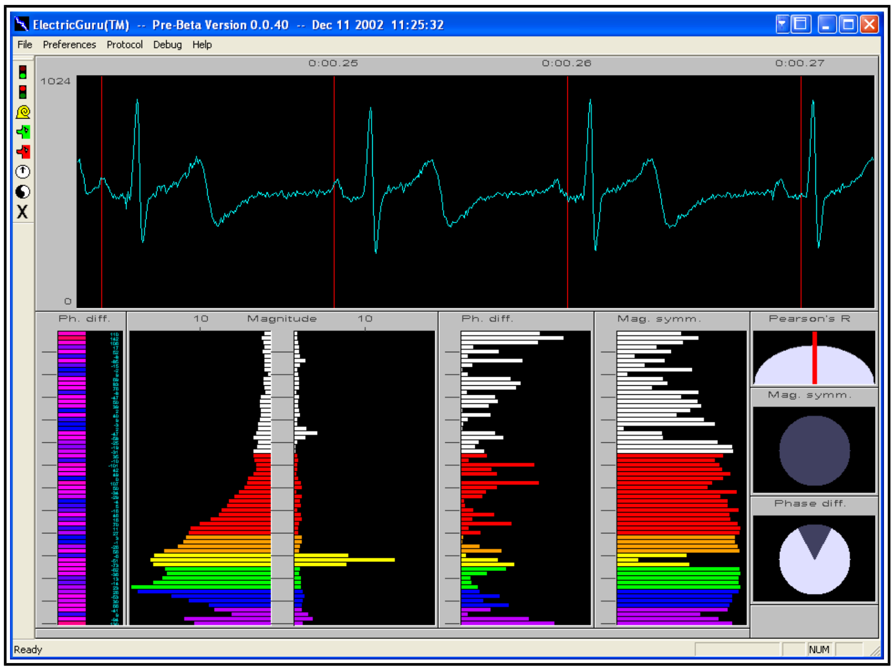
12. Download, install and start the software from this page:
https://www.olimex.com/Products/EEG/OpenEEG/EEG-SMT/resources/ElecGuru40.zip

13. It is advisable to adjust the settings in your ElecGuru program in Preferences-> Trace (waveform)… (depending how many channels/shields you use)

14. Choose the COM port your OLIMEXINO is connected to from Preferences -> Serial port… (Remember you have to set it to COM 1 to 4)

15. You connect the electrodes to your right arm, left arm and the DLR to your right leg

16. Start monitoring, adjust the settings until you receive an image like the one shown on the picture



SECTION 5 HARDWARE

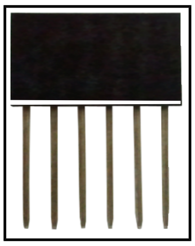
You can get a good view of the hardware observing the board. All pins, connectors and jumpers are named individually.

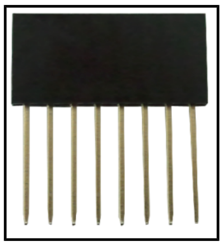
5.1 Arduino shield connector

These connectors follow the ARDUINO specification for shield connection. The shield comes with soldered connectors making it ready for mounting on compatible board with the possibility to have another shield mounted on it.

Pin #	**POWER CON1**	**ANALOG CON2**	**DIGITAL CON3**	**DIGITAL CON4**
1	RST	A0	D0	D8
2	3.3V	A1	D1	D9
3	5V	A2	D2	D10
4	GND	A3	D3	D11
5	GND	A4	D4	D12
6	Vin	A5	D5	D13
7	-	-	D6	GND
8	-	-	D7	AREF

6-pin and 8-pin connectors mounted (CON1, CON2 and CON3, CON4):





5.2 Trimmer TR1

Trimmer TR1 is calibrated during the factory testing. However it may be adjusted for the gain. Use at own risk.

5.3 Jumper description

The names of the jumpers on the board correspond to the bold names used below:

3.3V/5V

This jumper controls the power circuit. Whether powered by 3.3V or 5V board.

Default state is 3.3V.

REF_E

The position of the REF_E jumper depends on the "host" board. If the "host" board provides voltage on the AREF pin of the digital connector REF_E has to be open. If there is no voltage provided on the AREF pin then SHIELG-EKG-EMG's REF_E jumper has to be closed.

If both boards provide reference voltage to the 3V_REF_ADC line (and AREF pin respectively) then there would be a circuit conflict which might damage a component on one of the boards.

You need to ensure only one of the boards provides AREF voltage.

If you use only one shield this jumper has to be closed.
If you have multiple shields the first one should be closed; the respective REF_E jumpers on every other shield above it should be open.

Default state is closed.

AIN_SEL

This jumper is responsible for which channel the current SHIELD-EKG-EMG would utilize. If you have more than one shield one of them should have AIN_SEL in position 1, the second in position 3, etc.

Default state is in position 1.

D4/D9

Controls pin D4/D9. Some processors utilize the default D9 pin so you have to switch to D4. This jumper provides easy option to do so.

Default state is D9.

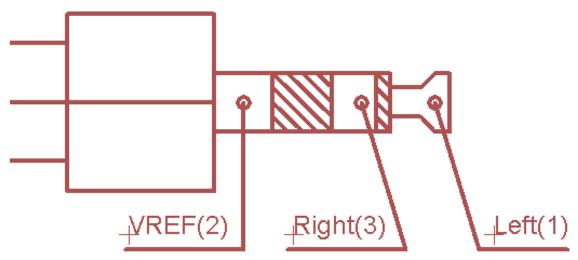
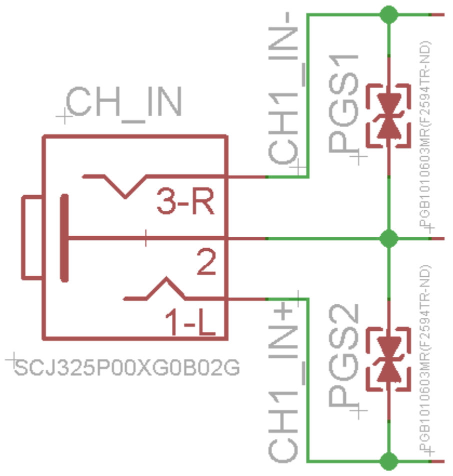
CAL

CAL jumper is used for feedback of the calibration and requires additional cable.

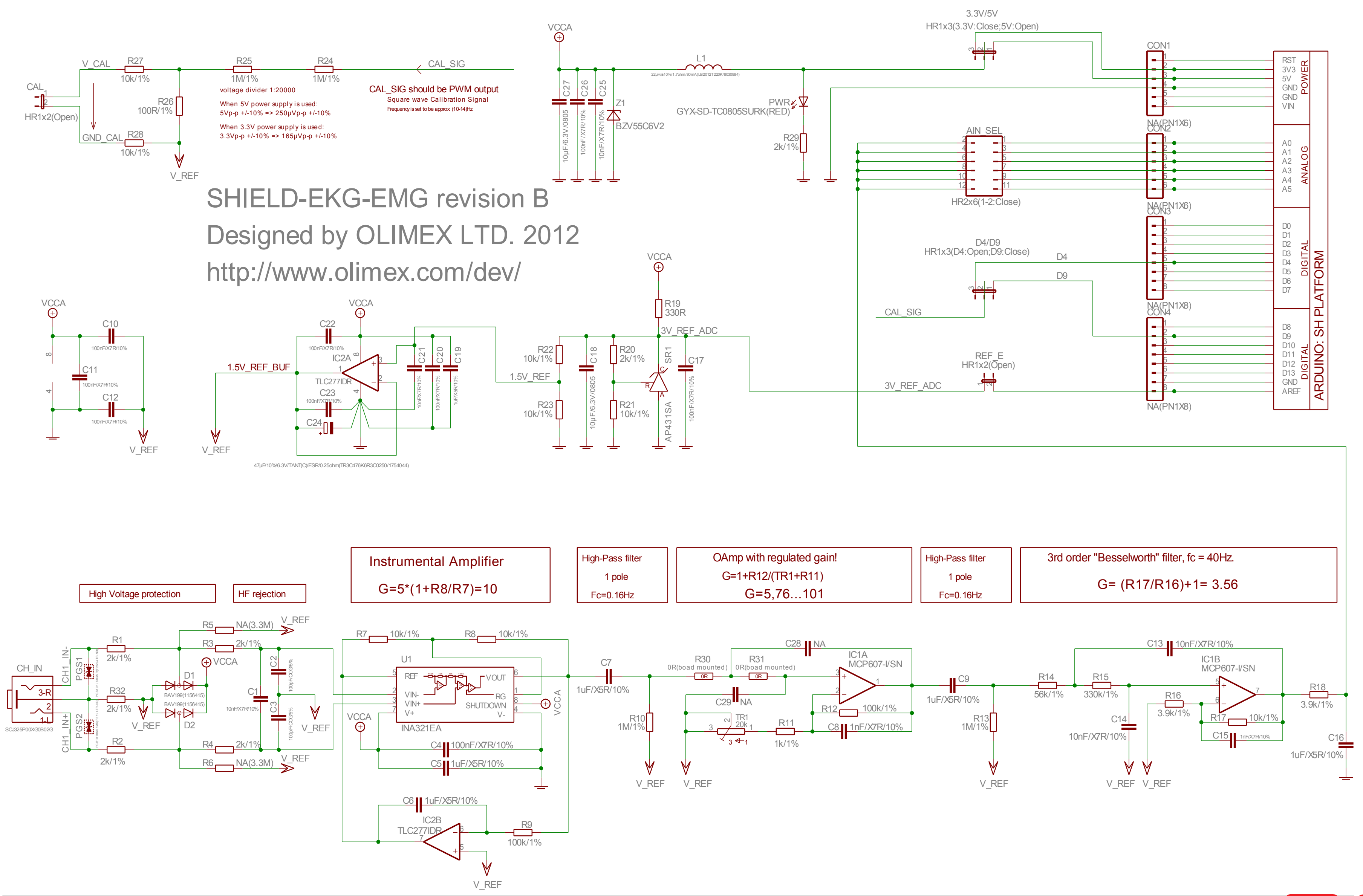
Default state is open.

5.4 Custom electrode connection

To make the passive electrode yourself you need three cables, passive electrode surface and audio jack. The way the signals go for connecting the audio jack is shown below:

	
Electrode cable side	**SHIELD-EKG-EMG side**

Note that connecting active electrode that way will probably lead to electrical shock for SHIELD-EKG-EMG. Only passive electrodes can be used with SHIELD-EKG-EMG.



SHIELD-EKG-EMG revision B

Designed by OLIMEX LTD. 2012

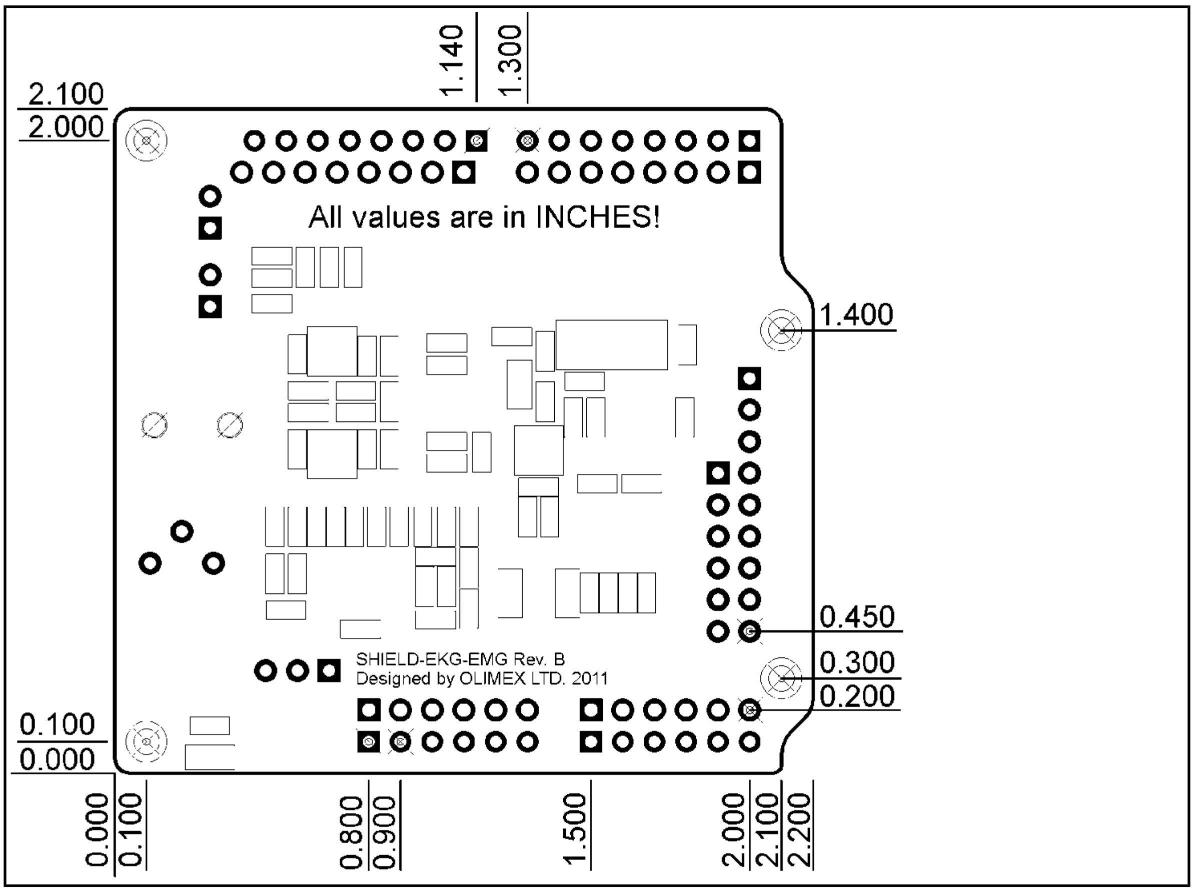
http://www.olimex.com/dev/

Electronics Katrangi Trading

6.2 Physical dimensions

As you can see below the dimensions follow the classic Arduino shield pin specification making it compatible with all Olimex Duino boards.

Note: All dimensions are in inches!



1.140
1.300
2.100
2.000
All values are in INCHES!
1.400
0.450
0.300
0.200
SHIELD-EKG-EMG Rev. B
Designed by OLIMEX LTD. 2011
0.100
0.000
0.000
0.100
0.800
0.900
1.500
2.000
2.100
2.200