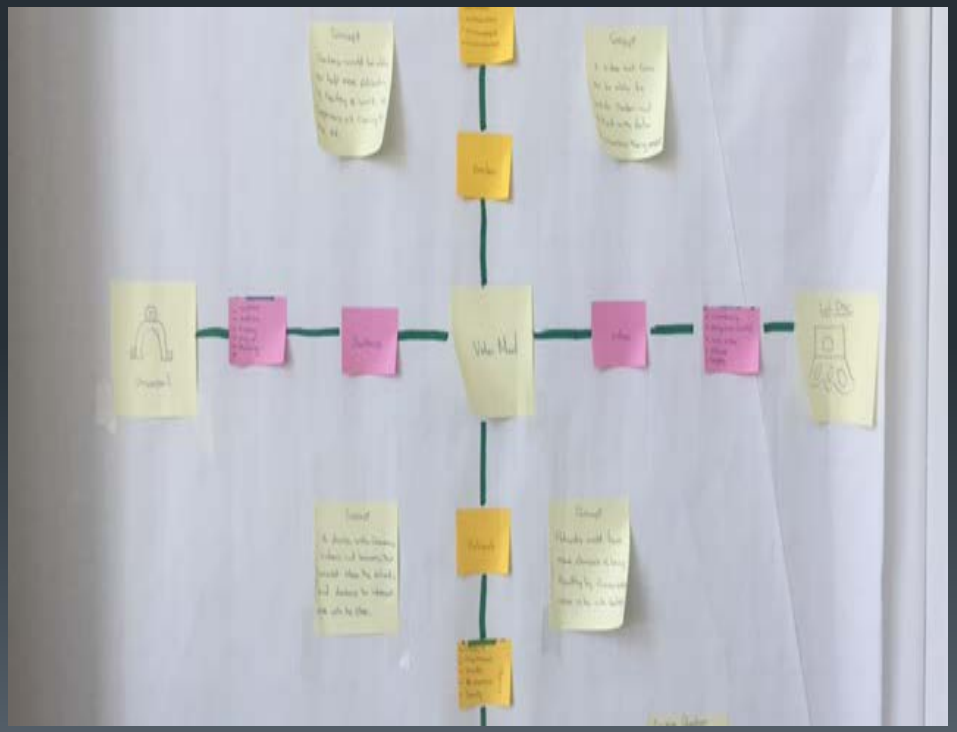
Video-Med

By Juancarlos

My Goals

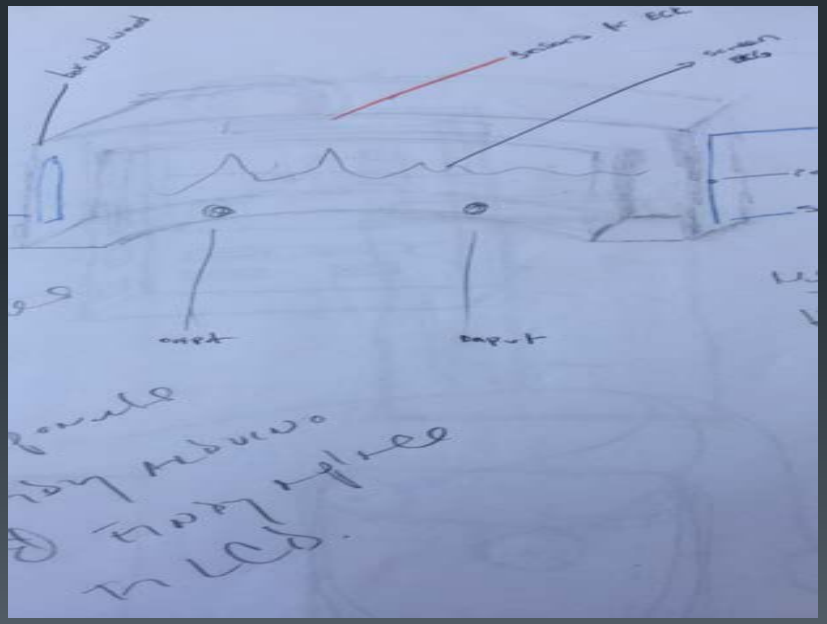
- 1- To Improve patients care outcomes
- 2- Reduce Cost
- 3- Educate Patient and provide experience
- 4- Deliver much more than a video conferencing
- 5- Delivering remote video consulting
- 6- Integrated medical devices
- 7- Easy to use for all patients

Purpose



My purpose is to share via remote patient monitoring in one system with clinician. The device will include automatic video call routing by service with EKG and EMG sensors to process the electrical activity of the heart.

Visualize Concept

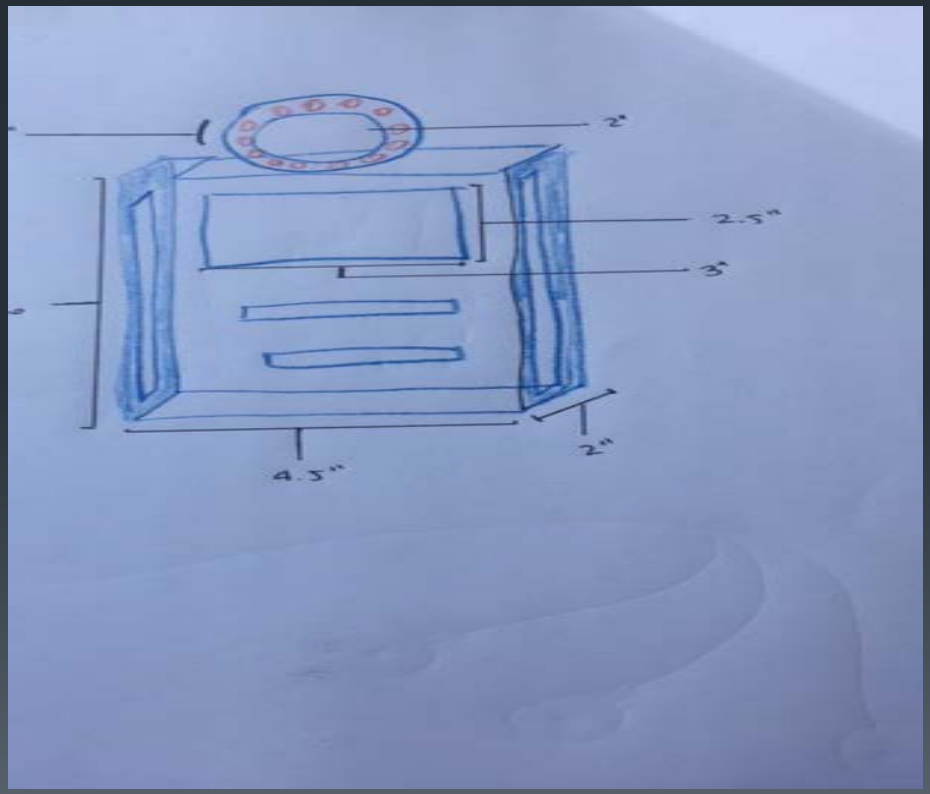


This is my original concept a screen in the center of the device with all the input sensors on the outside

Form

This was my form and idea behind of the video-med

Dimensions



All the dimensions are in inches. I started with the height 6” , depth 2” width 4.5”, screen width 3” and a height 2.5”

Video-Med

- Consist of the followings:
- Capacitive Touch Sensors as input
- An Actuator as output
- An Arduino
- An Olimexino 328
- Communication Network Protocol

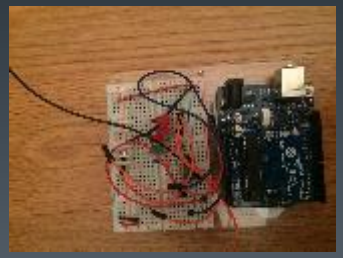
Materials

Shield EKG

Shield EMG

Electrode Monitoring
medi frace foam

3.2 TFT LCD Touch Screen



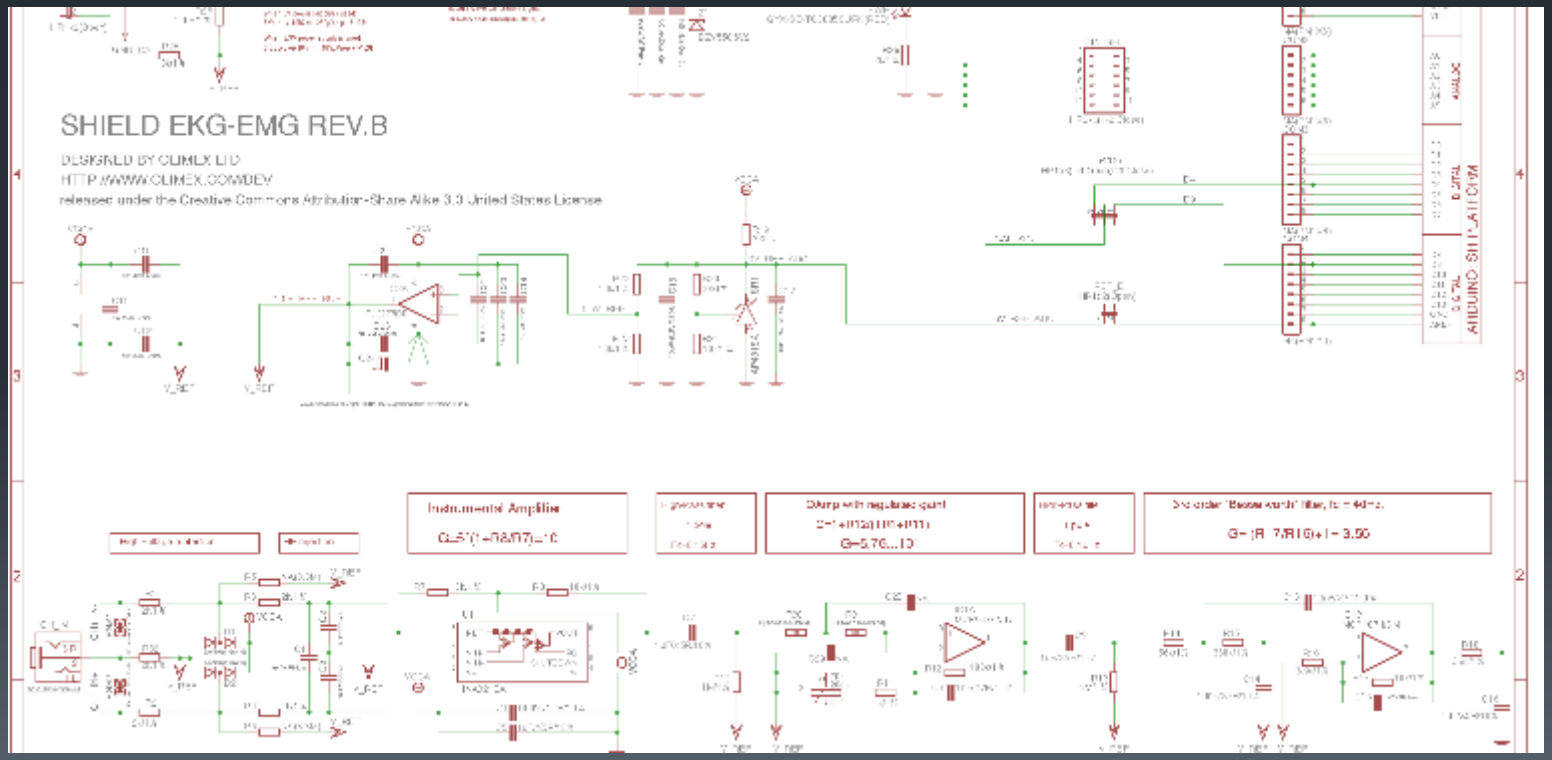
Arduino uno

Olimex Board

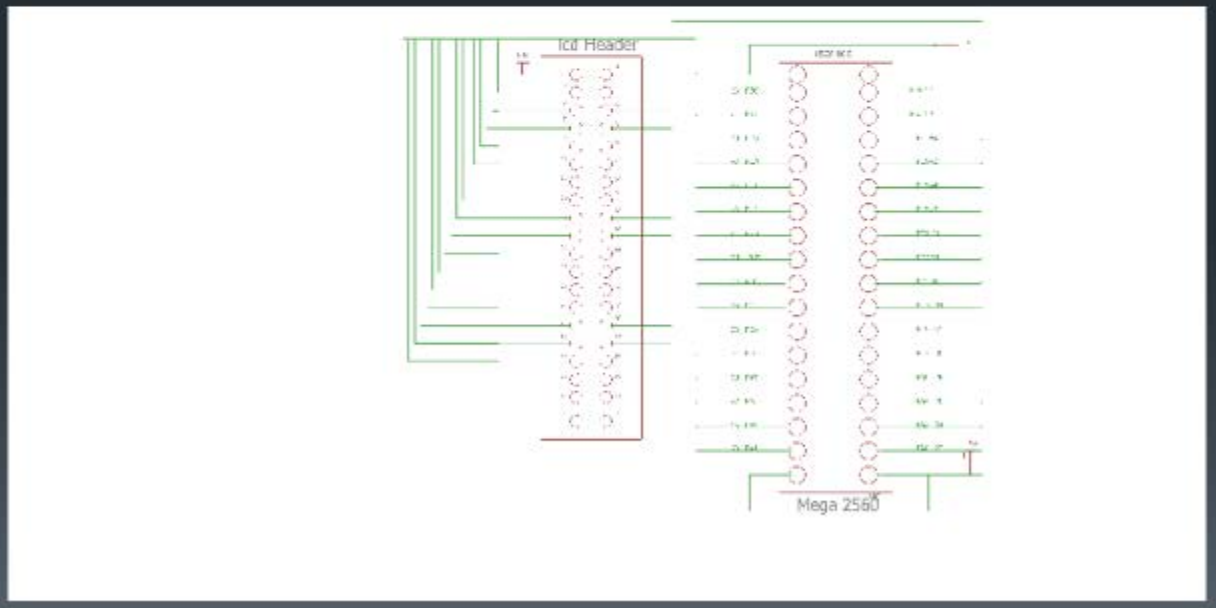
Modeling 3D

Prototype

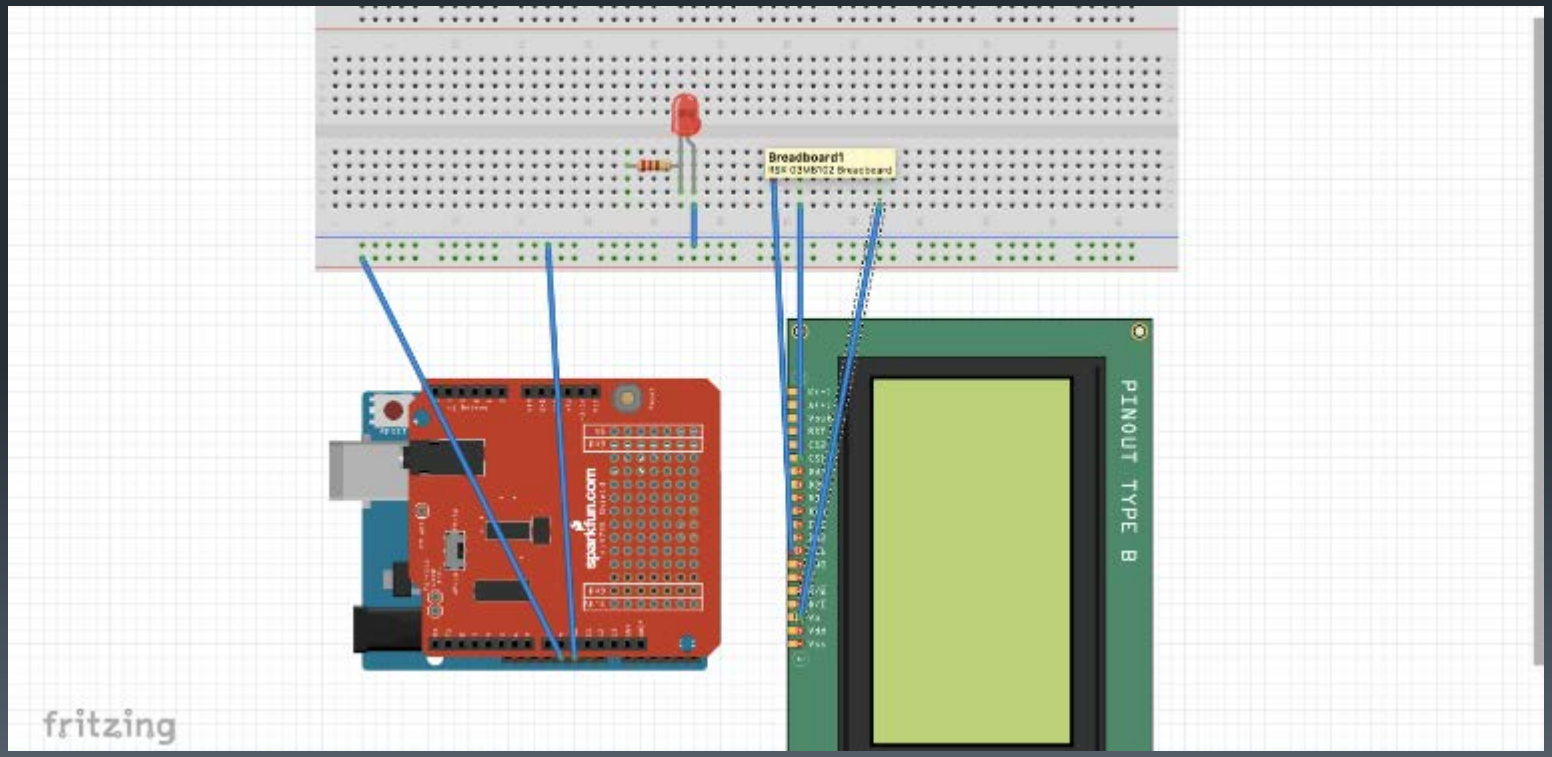
Schematic for Olimexino 328



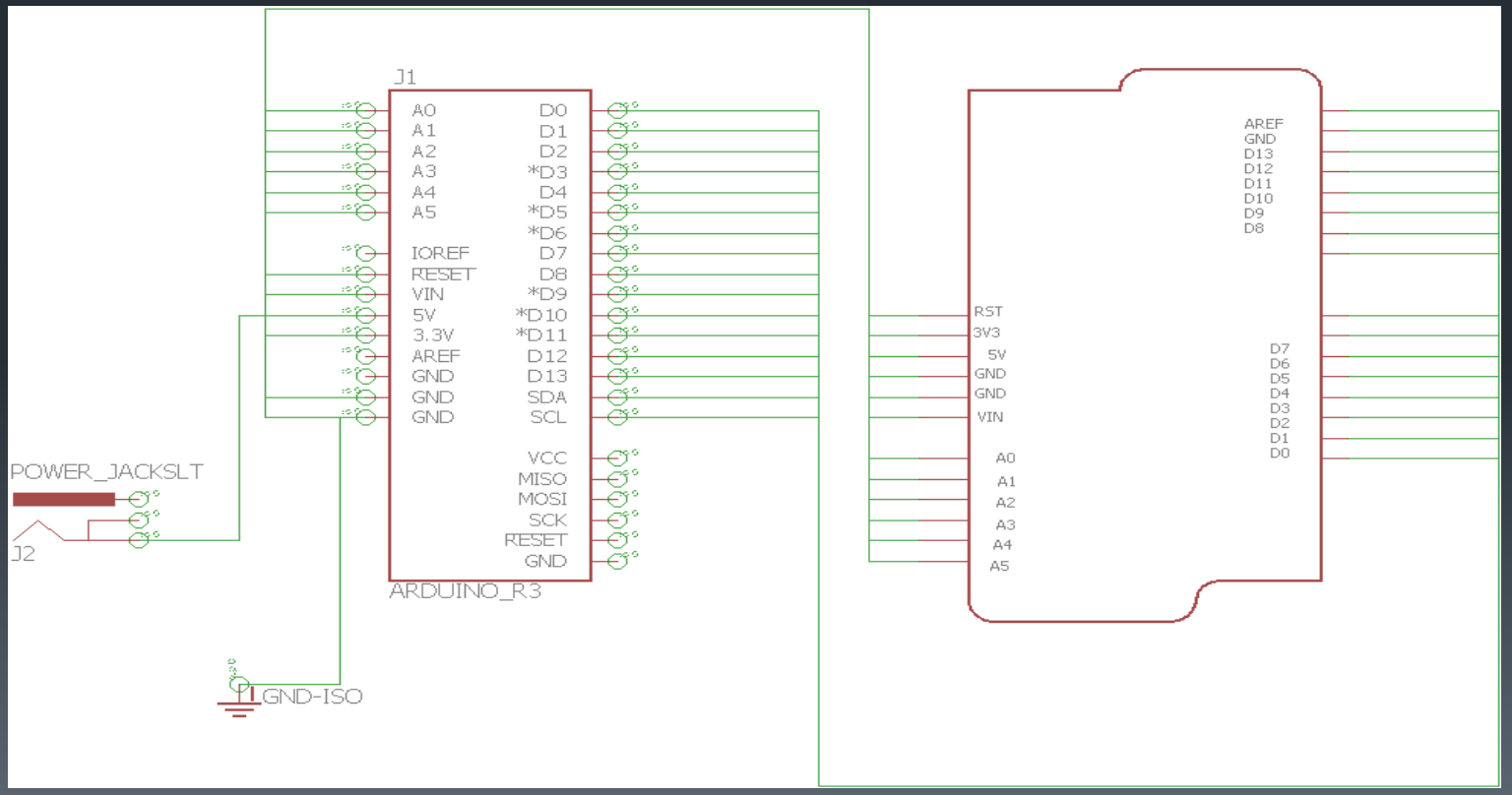
LCD and Arduino Schematic



Breadboard + arduino + Olimexino328 and a 3.2 LCD screen



Schematic for Olimex with Arduino



Code for Olimex

ShieldEkgEmgCode.ino

```
#include <compat/deprecated.h>
#include <FlexiTimer2.h>
//http://www.arduino.cc/playground/Main/FlexiTimer2

// All definitions
#define NUMCHANNELS 6
#define HEADERLEN 4
#define PACKETLEN (NUMCHANNELS * 2 + HEADERLEN + 1)
#define SAMPFREQ 256                    // ADC sampling rate 256
#define TIMER2VAL (1024/(SAMPFREQ))     // Set 256Hz sampling frequency
#define LED1  13
#define CAL_SIG 9

// Global constants and variables
volatile unsigned char TXBuf[PACKETLEN];  //The transmission packet
volatile unsigned char TXIndex;           //Next byte to write in the transmission packet.
volatile unsigned char CurrentCh;         //Current channel being sampled.
volatile unsigned char counter = 0;     //Additional divider used to generate CAL_SIG
volatile unsigned int ADC_Value = 0;                            //ADC current value

//~~~~~~~~~~
// Functions
//~~~~~~~~~~

/*****************************************************/
/*  Function name: Toggle_LED1                       */
/*  Parameters                                       */
/*    Input   :  No                                  */
/*    Output  :  No                                  */
/*    Action: Switches-over LED1.                    */
/*****************************************************/
void Toggle_LED1(void){

 if((digitalRead(LED1))==HIGH){ digitalWrite(LED1,LOW); }
 else{ digitalWrite(LED1,HIGH); }

}
```

```
void toggle_GAL_SIG(void){

 if(digitalRead(CAL_SIG) == HIGH){ digitalWrite(CAL_SIG, LOW); }
 else{ digitalWrite(CAL_SIG, HIGH); }

}


/*************************************************************/
/*  Function name: setup                                     */
/*  Parameters                                               */
/*    Input   : No                                           */
/*    Output  : No                                           */
/*    Action: Initializes all peripherals                    */
/*************************************************************/
void setup() {

 noInterrupts();  // Disable all interrupts before initialization

 // LED1
 pinMode(LED1, OUTPUT);  //Setup LED1 direction
 digitalWrite(LED1,LOW); //Setup LED1 state
 pinMode(CAL_SIG, OUTPUT);

 //Write packet header and footer
 TXBuf[0] = 0xa5;    //Sync 0
 TXBuf[1] = 0x5a;    //Sync 1
 TXBuf[2] = 2;       //Protocol version
 TXBuf[3] = 0;       //Packet counter
 TXBuf[4] = 0x02;    //CH1 High Byte
 TXBuf[5] = 0x00;    //CH1 Low Byte
 TXBuf[6] = 0x02;    //CH2 High Byte
 TXBuf[7] = 0x00;    //CH2 Low Byte
 TXBuf[8] = 0x02;    //CH3 High Byte
 TXBuf[9] = 0x00;    //CH3 Low Byte
 TXBuf[10] = 0x02;   //CH4 High Byte
 TXBuf[11] = 0x00;   //CH4 Low Byte
 TXBuf[12] = 0x02;   //CH5 High Byte
 TXBuf[13] = 0x00;   //CH5 Low Byte
 TXBuf[14] = 0x02;   //CH6 High Byte
 TXBuf[15] = 0x00;   //CH6 Low Byte
 TXBuf[2 * NUMCHANNELS + HEADERLEN] =  0x01;                    // Switches state

 // Timer2
 // Timer2 is used to setup the analag channels sampling frequency and packet update.
 // Whenever interrupt occures, the current read packet is sent to the PC
 // In addition the CAL_SIG is generated as well, so Timer1 is not required in this case!
 FlexiTimer2::set(TIMER2VAL, Timer2_Overflow_ISR);
 FlexiTimer2::start();

 // Serial Port
 Serial.begin(57600);
 //Set speed to 57600 bps

 // MCU sleep mode = idle.
 //outb(MCUCR,(inp(MCUCR) | (1<<SE)) & (~(1<<SM0) | ~(1<<SM1) | ~(1<<SM2)));

 interrupts();  // Enable all interrupts after initialization has been completed
}

/*************************************************************/
/*  Function name: Timer2_Overflow_ISR                       */
/*  Parameters                                               */
/*    Input   : No                                           */
/*    Output  : No                                           */
/*    Action: Determines ADC sampling frequency.             */
/*************************************************************/
void Timer2_Overflow_ISR()
{
  // Toggle LED1 with ADC sampling frequency /2
  Toggle_LED1();

 //Read the 6 ADC inputs and store current values in Packet
 for(CurrentCh=0;CurrentCh<6;CurrentCh++){
   ADC_Value = analogRead(CurrentCh);
   TXBuf[((2*CurrentCh) + HEADERLEN)] = ((unsigned char)((ADC_Value & 0xFF00) >> 8));                    // Write High Byte
   TXBuf[((2*CurrentCh) + HEADERLEN + 1)] = ((unsigned char)(ADC_Value & 0x00FF));                       // Write Low Byte
 }

 // Send Packet
 for(TXIndex=0;TXIndex<17;TXIndex++){
   Serial.write(TXBuf[TXIndex]);
 }

 // Increment the packet counter
 TXBuf[3]++;

 // Generate the CAL_SIGnal
 counter++;                                                                    // increment the devider counter
 if(counter == 12){                 // 250/12/2 = 10.4Hz ->Toggle frequency
   counter = 0;
   toggle_GAL_SIG();                // Generate CAL signal with frequ ~10Hz
  }
}
```

```
/*************************************************/
/*  Function name: loop                          */
/*  Parameters                                   */
/*    Input   :  No                              */
/*    Output  :  No                              */
/*    Action: Puts MCU into sleep mode.          */
/*************************************************/
void loop() {

 __asm__ __volatile__ ("sleep");

}
```

Code for Processing

ekg_processing_code.txt

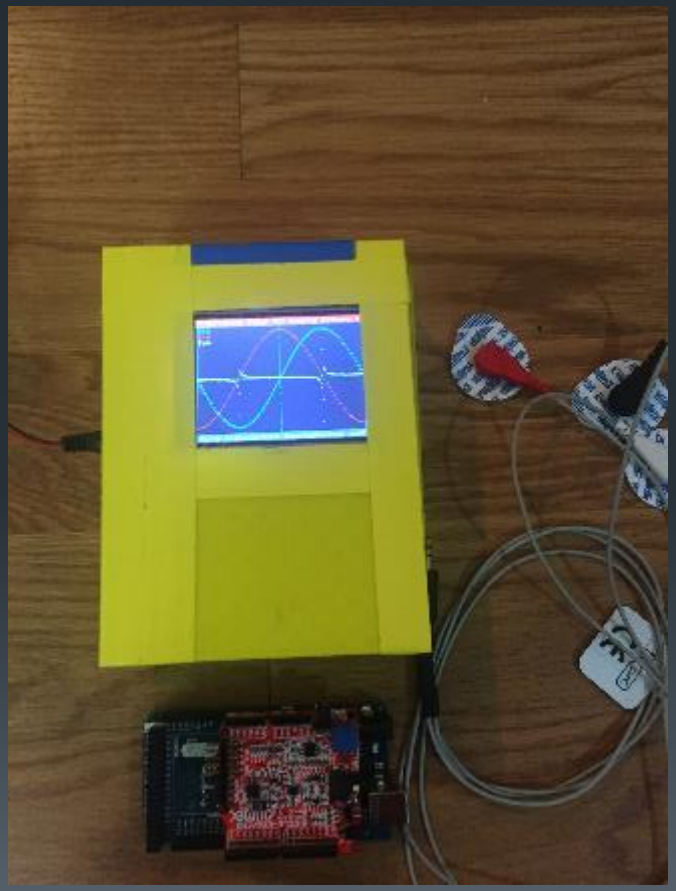
```
import processing.serial.*;

Serial myPort;  // The serial port

int counter = 0;
int inByte = 0;
float x1, y1, x2, y2, y;
float spacing = 1.0;

color RED = color(255, 0, 0);
color GREEN = color(0, 255, 0);
color BLUE = color(12, 16, 255);
color WHITE = color(255, 255, 255);
color BLACK = color(0, 0, 0);

void setup () {
size(800, 400); // window size
 // **** List available serial ports **** //
  String[] ports = Serial.list();
  print("No. ports = ");
  println(ports.length);
  for (int i = 0; i < ports.length; i++) {
    print("[" + i + "]");
  println(ports[i]);
  }
 // **** Enter selected port number here **** //
  String portName = Serial.list()[0];
  print("port selected = ");
  println(portName);
  myPort = new Serial(this, portName, 9600);
  background(BLUE);
}
```

```
void serialEvent (Serial myPort) {
  int inByte = myPort.read();
  println(inByte);
  y = height - inByte;
}

void draw () {
// line color:
  stroke(WHITE);

if (counter > width/spacing) {
    counter = 0;
    background(BLUE);
  }

  if (counter == 0) {
      x1 = 0;
      y1 = y;
  }

  if (counter > 0) {
    x2 = counter*spacing;
    y2 = y;
    line(x1, y1, x2, y2);
    x1 = x2;
    y1 = y2;
   }
  counter++;
}
```

Instructable

- INTRODUCTION:

- Video med uses information Technology and telecommunication to provide clinical health care from a distance. This Technology allows communication between patients and medical personnel and the transmission of health informatics from one site to another.

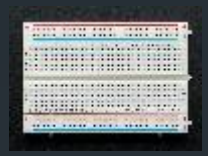


1- Materials

- Arduino Uno

- Olimex Shield Ecg, Ekg

- Breadboard

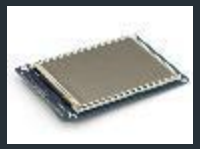


- **ECG-GEL-ELECTRODE**

ECG-PRO-3-WAY-CABLE

SHIELD-EKG-EMG-PA

3.2 sainsmart tft lcd



TFT Shield For Arduino UNO R3

3D Models

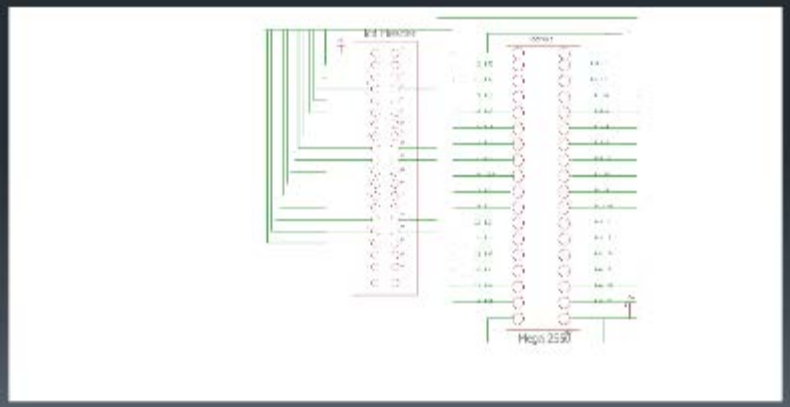
telemedprototype.123dx

telemed print front hole.stl

telemed 1 print.stl

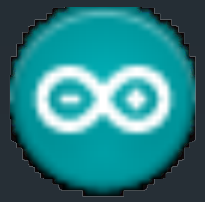
Schematics

LCD and Arduino Schematic



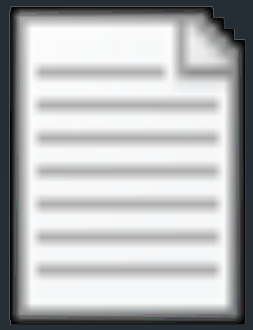
olimexarduino.sch

Arduino Code



sketch_dec06jca.ino

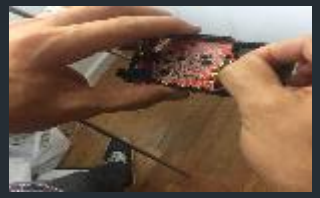
Processing Code



ekg_processing_ino.txt

Step by Step Desing

- 1) My first step was to prepare the Olimex 328 Board by removing its wire between the R6 pads. Failure to perform this step could result in electrical failure from increase power consumption.



- 2) I downloaded the IDE package from the Arduino website.

- 3) I downloaded 2 libraries required to run Olimex (Timer one, Timer2)

- 4) Next I arranged the EKG shield jumpers as follow:

- REF_E – closed
- 3.3V/5V – 5V position
- D4/D9 – D9 position
- ANI_SEL – 1 position

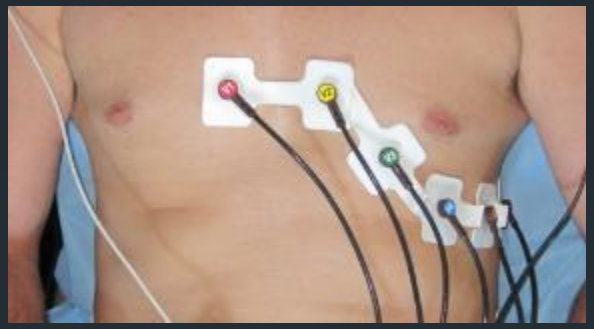
- 5) I connected the Shield to The Arduino Board.

- 6) I then connected the Arduino Olimex board to the USB

- 7) I then Downloaded the Code Shield EkgrmgCode. I compiled the code and I rang it.

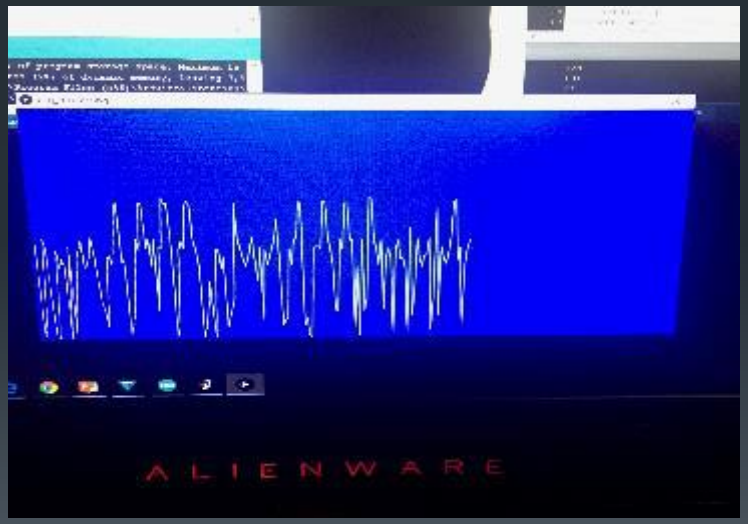
- 8) I then downloaded the code for EKG Processing

- 9) I Then attached the Ekg gel pads to “the patient”



- 10) The Ekg Processing code was compile and rang.

- 11) The screen Opened and the Electro Cardiogram (EKG) was displayed.



Difficulties encountered

- 1) In setting up my initial LCD screen, I learned that the 3.2 LCD screen was compatible with Arduino mega and not Arduino Uno
- 2) My LCD screen no longer supported myGLCD library. I then researched and found an alternate library Open.myGLCD
- 3)Manipulating the code for showing the EKG on the LCD screen did not work. This need more time to solve with additional trial and error.

What I learned

What I learned is how to manipulate LCD screens, how to trouble shoot, including testing and debugging. The process was fascinating, challenging but very rewarding. It was valuable experience to go from concept inspiration (Ideation) to prototyping to 3D Modeling, Designing schematics and finally coding.

What is still Outstanding

While I was able to show a graph on the LCD screen, I was unable to debug the code to show the actual EKG on the LCD Screen.