SMARTWATCH

OBIETTIVO:

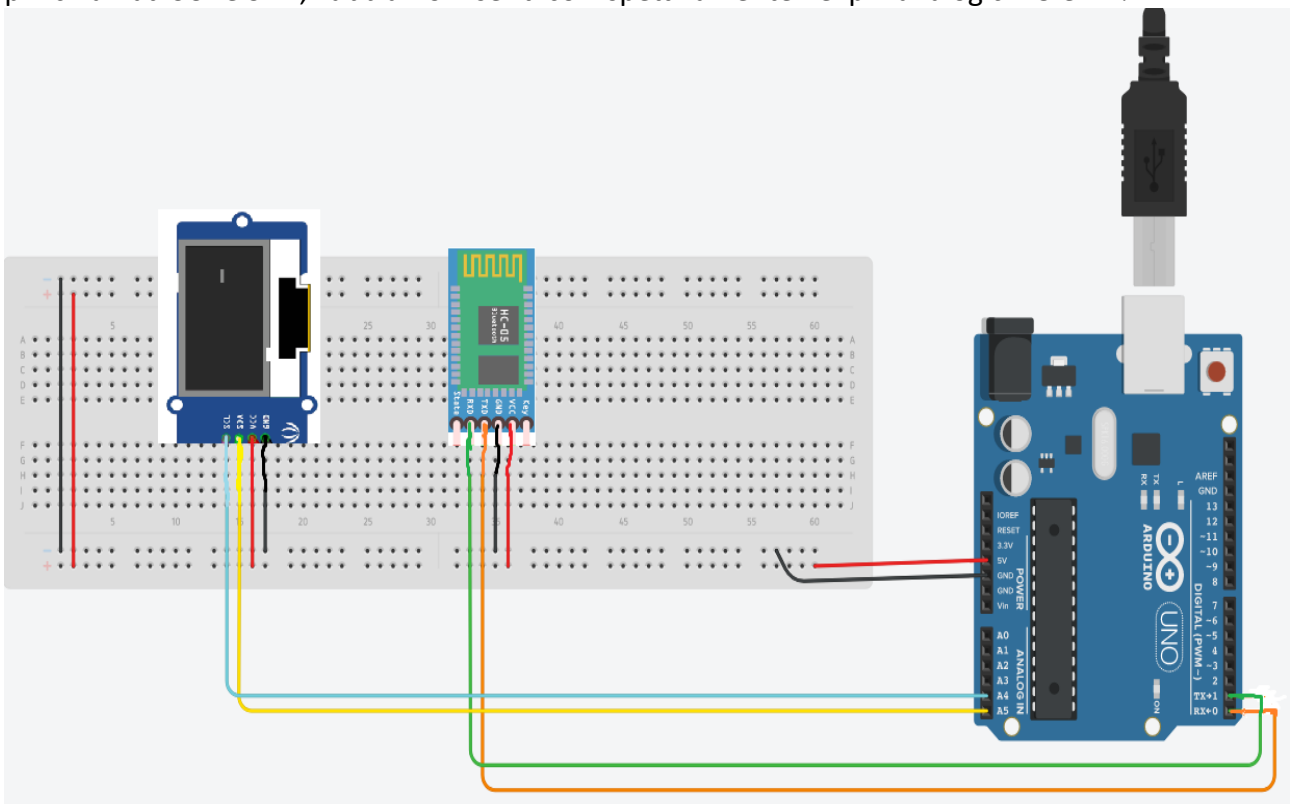
Il nostro scopo è quello di creare uno smartwatch con arduino. Faremo in modo di far comunicare arduino con un telefono android per poter visualizzare i parametri principali del telefono che sono:il dbd(lo stato della rete),l'orario analogico e digitale e la percentuale rimanente della batteria.Sul nostro orologio, quindi, si susseguiranno tre schermate con le rappresentazioni delle caratteristiche del telefno elencate sopra.

COMPONENTI HW/SW:

I componenti base del progetto sono ovviamente arduino, breadboard e dei fili resi disponibili dalla scuola.
Per la comunicazione tra telefono e arduino abbiamo comprato un modulo bluetooth HC-05 a 6 pin che supportasse i 5 volt così da non avere problemi con l'alimentazione ed infine abbbiamo comprato il display oled SSD1306 a 4 pin su cui visualizzare i vari parametri del telefono.
Per lo sviluppo del programma abbiamo utilizzato Arduino IDE e abbiamo scaricato dal sito della scuola l'applicazione (arduBTH) adatta per il telefono per avviare la trasmissione dei dati.
I componenti non forniti dalla scuola sono 2: il modulo bluetooth e il display oled che sono costati rispettivamente 7,50€ e 7€.

MONTAGGIO HW:

Abbiamo collegato il modulo bluetooth a massa e all'alimentazione e i due piedini per la trasmissione e la ricezione li abbiamo inseriti corrispettivamente nel piedino 0 e 1 di arduino.
Per quanto riguarda il display,lo abbiamo connesso ovviamente a massa e all'alimentazione e i due pin chiamati SCL e SDA,li abbiamo inseriti corrispettivamente nei pin analogici A5 e A4.



COSTRUZIONE SW:

Si inizia con l'inclusione di tutte le librerie.

```
//libreria che serve per far comunicare più dispositivi
#include <SPI.h>
//libreria che serve per utilizzare i pin SDA e SCL come mezzi di comunicazione
#include <Wire.h>
//libreria che ci permette di utilizzare il nostro display
#include <Adafruit_GFX.h>
#include "Adafruit_SSD1306.h"
//definiamo il pin di reset
#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);
```

Abbiamo poi definito alcune macro e variabili.

```
// il numero di NUMFLAKES nell'esempio dell'animazione
#define NUMFLAKES 10
//indicizziamo nella matrice delle icone
#define XPOS 0
#define YPOS 1
#define DELTAY 2
//definiamo la grafica del display con le sue misure in pixel
#define LOGO16_GLCD_HEIGHT 16
#define LOGO16_GLCD_WIDTH 16
//definiamo le variabii che ci serviranno più avanti nel programma
String str = "";
byte h = 0;
byte m = 0;
byte S = 0;
String dmy, stime, network, battery, inNumber, s;
byte centerX = 24;
byte centerY = 39;
byte Radius = 24;
double RAD = 3.141592 / 180;
double LR = 89.99;
```

Abbiamo poi scritto la funzione per l'orologio analogico.

```
//Inizializzazione della funzione dell'orologio analogico
void showTimeAnalog (int center_x, int center_y, double pl1, double pl2, double pl3)
{
  double x1, x2, y1, y2;
  //operazioni che servono per disegnare il contorno e le lancette del nostro orologio
  x1 = center_x + (Radius * pl1) * cos ((6 * pl3 + LR) * RAD);
  y1 = center_y + (Radius * pl1) * sin ((6 * pl3 + LR) * RAD);
  x2 = center_x + (Radius * pl2) * cos ((6 * pl3 - LR) * RAD );
  y2 = center_y + (Radius * pl2) * sin ((6 * pl3 - LR) * RAD );
  //istruzione che disegna una linea bianca nei punti calcolati precedentemente
  display.drawLine ((int) x1, (int) y1, (int) x2, (int) y2, WHITE);
}
```

Quella per l'orologio digitale

```
//inizializzazione della funzione dell'orario
void digitalClock ()
{
  //impostiamo la grandezza di quello che verrà scitto sul display
  display.setTextSize (1);
  //impostiamo il colore di quello che deve essere visualizzato
  display.setTextColor (WHITE);
  //impostiamo i pixel da dove poter scrivere
  display.setCursor (60, 20);
  //scrive sul dispaly il valore di dmy
  display.println (dmy);
  //impostiamo la grandezza di quello che verrà scritto sul display
  display.setTextSize (1);
  //impostiamo i pixel da dove poter scrivere
  display.setCursor (60, 30);
  //scrive sul display il cntenuto della variabile stime
  display.println (stime);
  //conclude il gruppo di azioni
  display.display ();
  //ritarda di 3 secondi
  delay (3000);
}
```

Un'altra funzione per la batteria numerica e grafica

```
//inizializzo la funzione per la batteria
void Battery ()
{ //istruzione che elimina tutto ciò che è visualizzabile sul display
  display.clearDisplay ();
  //impostiamo la grandezza di quello che verrà scitto sul display
  display.setTextSize (2);
  //impostiamo il colore di quello che deve essere visualizzato
  display.setTextColor (WHITE);
  //impostiamo i pixel da dove poter scrivere
  display.setCursor (20, 0);
  //scrive sul display "Bat:"
  display.print ( "Bat:");
  //scrive sul dispay il valore della variabile battery
  display.print (battery);
  //scrive sul display"%"
  display.print ( "%");
  //forma un rettangolo con i vertici segnati dai pixel di colore bianco
  display.drawRect (14, 20, 80, 40, WHITE);
  //forma un rettangolo con i vertici segnati dai pixel di colore bianco
  display.drawRect (94, 30, 10, 20, WHITE);
  //riempe di bianco l'interno del retangolo a seconda del valore di battery
  display.fillRect (14, 20, (int)(8 * (battery.toInt ()) / 10), 40, WHITE);
  //conclude il gruppo di azioni
  display.display ();
  //ritarda di tre secondi
  delay (3000);
}
```

E l'ultima funzione per il dbd numerico e grafico.

```
//inizializza la funzione del dbd
void Network ()
{ //istruzione che elimina tutto ciò che è visualizzabile sul display
  display.clearDisplay ();
  //serie di istruzioni che disegnano un palo dela luce molto stilizzato.
  display.drawLine (5, 15, 25, 15, WHITE);
  display.drawLine (5, 15, 14, 30, WHITE);
  display.drawLine (25, 15, 17, 30, WHITE);
  display.fillRect (14, 15, 4, 40, WHITE);
```

```
  //impostiamo la grandezza di quello che verrà scitto sul display
  display.setTextSize (2);
  //impostiamo il colore di quello che deve essere visualizzato
  display.setTextColor (WHITE);
  //impostiamo i pixel da dove poter scrivere
  display.setCursor (80, 34);
  //facciamo srivere sul dispay il valore della variabile network
  display.print (network);
  //impostiamo la grandezza di quello che verrà scitto sul display
  display.setTextSize (1);
  //impostiamo i pixel da dove poter scrivere
  display.setCursor (117, 44);
  //conclude il gruppo di azioni
  display.display ();
  //ritarda di tre secondi
  delay (3000);
}
```

Nel void setup e nel void loop invece abbiamo completato le due operazioni di trasmissione e ricezione dati e la seguente scrittura dei dati sul display.

```
//inizializza il setup
void setup ()
{ //inizializza con il monitor seriale
  Serial.begin (9600);
  // inizializza con l'addr I2C 0x3D (per 128x64)
  display.begin (SSD1306_SWITCHCAPVCC, 0x3C);
  //istruzione che elimina tutto ciò che è visualizzabile sul display
  display.clearDisplay ();
}
//inizializza il loop
void loop () {
 //adesso inizia una serie di istruzioni che servono per poter trasmettere
 //e ricevere i dati da telefono a computer
  //fintantochè la variabile è maggiore di 0 allora esegue le istruzioni successive
  while (Serial.available ()> 0) {
    //assegnamo l'istruzione Serial.read alla variabile ch
    char ch = Serial.read ();
    str += ch;
    if (ch == '$') {
      char buf[str.length()+1];
      str.toCharArray(buf, sizeof(buf));
 //utilizziamo i token perchè rinchiudono sequenze di caratteri che saranno
//scansionati per permettere la successiva ricezione e trasmissione dei dati
      //con l'istruzione Null si ferma momentaneamente il token
      char* token = NULL;
      //assegnamo la variabile i a 0
      int i=0;
      token = strtok(buf, ";");
      //se il token non è annullato allora esegue le istruzioni successive
       while (token != NULL) {
        if(0==i)
        //memorizza nel token la variabile dmy
          dmy =String(token);
```

```
    else if(1==i){
    //memorizza nel token la variabile stime
      stime=String(token);
    }
    else if(2==i)
    //memorizza nel token la variabile battery
      battery=token;
    else if(3==i)
    //memorizza nel token la variabile network
      network=token;
    //concludiamo il funzionamento dei token
    token = strtok(NULL, ";");
    //aumenta la variabile i
    i++;
  }

  char buf2[stime.length()+1];
  stime.toCharArray(buf2, sizeof(buf2));
  i=0;
  token = strtok(buf2, ":");
     //fintantochè il token non è nnullato allora esegue
    //delle istruzioni che ricevono e fanno trasferire i dati
     while (token != NULL) {
       if(0==i)
         h =String(token).toInt();
       else if(1==i)
         m=String(token).toInt();
       else if(2==i)
         S=String(token).toInt();
       //concludiamo il funzionamento dei token
       token = strtok(NULL, ":");
       //aumenta la variabile i
       i++;
     } //scrive sul monitor il valore di str
     Serial.println(str);
     str="";
   }
  } //istruzioni che fano ripetere la scrittura dei dati delle variabili
  Serial.println(dmy);
  Serial.println(stime);
  Serial.println(battery);
  Serial.println(network);
  //istruzione che elimina tutto ciò che è visualizzabile sul display
  display.clearDisplay ();
  //istruzione che crea il cerchio bianco del nostro orologio
  display.drawCircle (centerX, centerY, Radius, WHITE);
  //serie di istruzioni che servono per dichiarare le funzioni chiamate sopra
  showTimeAnalog (centerX, centerY, 0.1, 0.5, h * 5 + (int) (m * 5/60));
  showTimeAnalog (centerX, centerY, 0.1, 0.78, m);
  digitalClock();
  Battery();
  Network();
}
```

COSA SUCCEDE:

Dopo aver compilato e caricato il programma abbiamo osservato che sul display venivano visualizzati tutti i grafici dei vari parametri ma il valore numerico.
Allora ,accendendo il bluetooth del nostro telefono, abbiamo utilizzato l'applicazione, fornita dal sito della scuola, trasmettendo così i dati al modulo HC-05 e appena la schermata del parametro cambiava, abbiamo visto che erano presenti anche i vaori effettivi dell'orario corrente e dei due stati;quello della batteria e quello dello stato di rete.

CONCLUSIONI:

Durante la fase di lavorazione abbiamo riscontrato 2 problemi:
il primo è stato che il programma non veniva caricato su arduino perchè i pin 0 e 1 erano occupati dal modulo bluetooth e quindi, abbiamo dovuto scollegare il modulo, tranne l'alimentazione e la massa,prima del caricamento su arduino e succesivamente ricollegarlo per poter così caricare correttamente il programma.
Il secondo era un problema con l'applicazione che abbiamo risolto grazie all'aiuto del professore e consisteva nella mancanza di un comando dentro l'applicazione, poi risolto grazie al professore.
Infine dopo aver visto il corretto funzionamento dell'orologio, abbiamo concluso che è stato raggiunto lo scopo prefissato.
Per migliorare il nostro progetto, si potevano implementare altri parametri del telefono come il gestore telefonico o eventuali notifiche del nostro telefono in modo da renderlo sempre più simile ad un vero e proprio smartwatch.

Gruppo n2 Componenti: Bonavita Bresciani Barabino Classe 4AEA A.S.2018-2019