Word Clock

Titolo del progetto: Word Clock
Alunno/a: Gabriele Alessi, Mattia Lazzaroni, Paolo Claudio Weishaupt
Classe: SAM I3
Anno scolastico: 2018/19
Docente responsabile: Adriano Barchi

1 Introduzione

1.1 Informazioni sul progetto

- Allievi: Gabriele Alessi, Mattia Lazzaroni, Paolo Claudio Weishaupt.
  Superiore professionale: Adriano Barchi.
- Scuola d'Arti e Mestieri di Trevano, sezione Informatica, classe 3, modulo 306.
- Data inizio: 13.02.2019
  Data fine: 22.05.2019

1.2 Abstract

A word clock is a clock that shows the time by using spoken language. In this project a box is used where below there is a strip of led and above there is a sheet with the words that together form the time. This strip is programmed in Arduino so that the led that form the words of the current time are turned on. The time is obtained from a time server and set up by a RTC, otherwise it can be configured directly from the box using physical button.
The final product can be considered as a prototype, since the main idea is to expose a large word clock on the roof of the study center.

1.3 Scopo

Lo scopo di questo progetto è quello di creare un prototipo funzionante di un orologio a parole (word clock) per poi in seguito presentarlo a una commissione che deciderà se stanziare i fondi per poterne realizzare una versione più grande montata sulla torre sinistra della scuola.
Un word clock è un orologio che mostra l'orario tramite l'uso del linguaggio parlato. In questo progetto viene usata una scatola dove viene montata una striscia di led con sopra montato un sostegno in cui appoggiare un foglio dove sono scritte le parole. Questa striscia viene programmata in Arduino, in questo modo i led si accendono formando le parole che indicano l'orario. L'orario è ottenuto da un server e viene poi impostato e memorizzato da un RTC, altrimenti è possibile configurarlo direttamente dalla scatola tramite pulsanti fisici.

Analisi

1.4 Analisi del dominio

L'entrata del quarto piano è poco invitante e non ci sono esposti dei progetti realizzati dagli allievi.
Quindi questo prodotto colmerebbe questo spazio vuoto mostrando qualcosa di utile (l'orario) in maniera diversa e interessante. Il contesto in cui il prodotto dovrebbe funzionare è adatto e sarebbe l'ideale metterlo all'inizio del corridoio. Attualmente esiste un piccolo orologio binario nella zona degli elettronici, ma questa soluzione è più chiara e leggibile, quindi non ci sarebbero problemi nel metterlo in mostra.
Gli utenti sono principalmente gli allievi del quarto piano, che arrivando al corridoio verrebbero a conoscenza dell'orario solamente alzando lo sguardo e saprebbero se sono in ritardo per la lezione o meno.
Quindi gli utenti del prodotto non necessitano di particolari competenze (come con l'orologio binario).

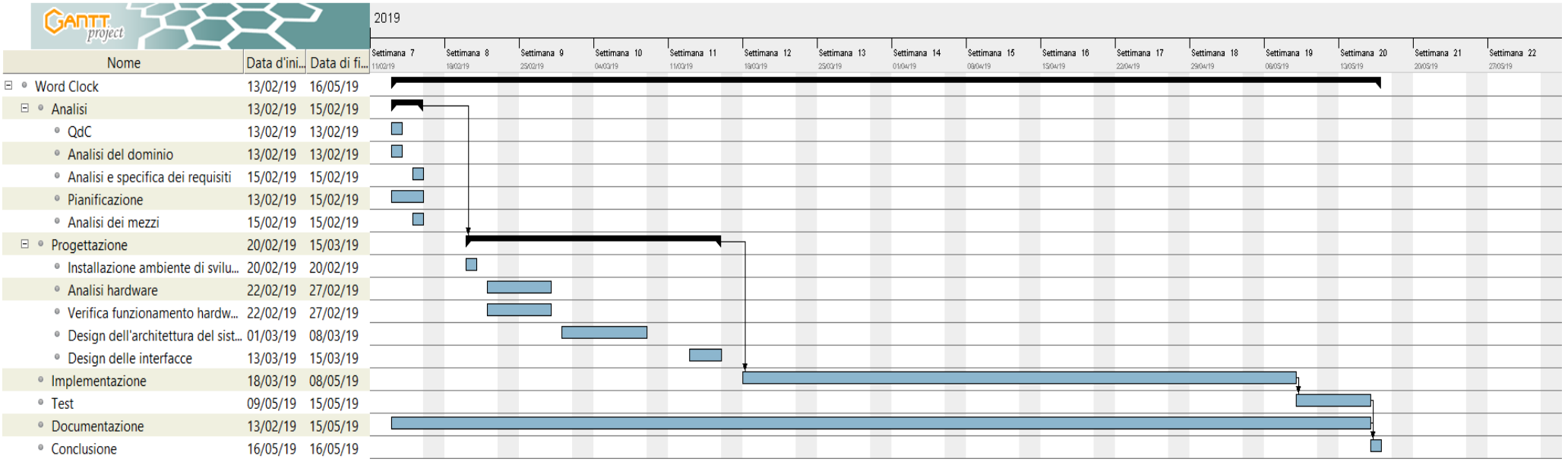
1.5 Analisi e specifica dei requisiti

ID: REQ-001	
Nome	Verifica componenti hardware
Priorità	1
Versione	1.0
Note	È necessario verificare che i componenti hardware (Fishino e led) funzionino correttamente.

ID: REQ-002	
Nome	Implementare il Word Clock
Priorità	1
Versione	1.0
Note	È necessario implementare il Word Clock in modo che mostri l'orario sotto forma di linguaggio parlato.
Sotto requisiti	
001	Scrivere il testo con le differenze di 5 minuti
002	Mostrare i secondi tramite l'utilizzo di 12 led, quindi 5 secondi per ogni pallino acceso.
003	Si dovrà poter controllare e impostare l'orario e le impostazione tramite il modello fisico.
004	Cambiare i colori dei led in base all'orario.

1.6 Pianificazione

La pianificazione del progetto è stata effettuata mediante la realizzazione di un diagramma di Gantt.



Nome	Data d'ini...	Data di fi...
Word Clock	13/02/19	16/05/19
Analisi	13/02/19	15/02/19
QdC	13/02/19	13/02/19
Analisi del dominio	13/02/19	13/02/19
Analisi e specifica dei requisiti	15/02/19	15/02/19
Pianificazione	13/02/19	15/02/19
Analisi dei mezzi	15/02/19	15/02/19
Progettazione	20/02/19	15/03/19
Installazione ambiente di svilu...	20/02/19	20/02/19
Analisi hardware	22/02/19	27/02/19
Verifica funzionamento hardw...	22/02/19	27/02/19
Design dell'architettura del sist...	01/03/19	08/03/19
Design delle interfacce	13/03/19	15/03/19
Implementazione	18/03/19	08/05/19
Test	09/05/19	15/05/19
Documentazione	13/02/19	15/05/19
Conclusione	16/05/19	16/05/19

1.7 Analisi dei mezzi

1.7.1 Software

I software utilizzati per la realizzazione di questo progetto sono:

- Microsoft Word 2016
- Microsoft Power Point 2016
- Arduino IDE 1.8.7
- GanttProject 2.8.9 Pilsen
- Google Chrome 71.0.3578.98
- Atom 1.34.0
- Visual Studio Code 1.34.0
- FishinoFlasher 5.0.0

1.7.2 Hardware

- Computer Mattia Lazzaroni:
  - Modello: Acer Aspire E 15
  - Processore: Intel Core i7 7500u
  - RAM: 16GB LPDDR4 2133 MHz
  - GPU: NVIDIA 940MX
  - SSD: 256 GB
- Computer Gabriele Alessi:
  - Modello: HP ENVY Notebook
  - Processore: Intel Core i7
  - RAM: 16GB
- Computer Paolo Weishaupt:
  - Modello: Huawei MateBook X Pro
  - Processore: Intel Core i7-8550U
  - RAM: 8GB
  - Display 13.9” 3000x2000px
  - SSD: 512 GB
- Fishino UNO REV2:
  - Compatibile al 100% con Arduino UNO
  - Modulo WiFi integrato
  - Slot per schede MicroSD integrato
  - Modulo RTC integrato
  - Connettore sfalsato per facilitare l’uso con breadboards
  - Sezione di alimentazione a 3.3V potenziata
  - Compatibilità con le schede millefori
  - Schemi elettrici, file Eagle e pinout della scheda
- Adafruit NeoPixel
  - 195 led RGB
- Arduino Mega 2560
- Componenti:
  - Resistenza 500Ω
  - Condensatore 1000µF
  - Alimentatore di supporto

2 Progettazione

In questo capitolo viene spiegato come è stato ideato il prodotto e come è stata preparata la fase di implementazione. Inizialmente è stato organizzato l'ambiente di sviluppo (software, striscia di led, resistenze). Poi sono stati analizzati i componenti del progetto e che funzionassero correttamente provando a eseguire qualche programma di test cambiando i parametri.
Successivamente è stata fatta un'idea della struttura del sistema, definendo i file e la struttura delle cartelle. Infine si è pensato a come il prodotto si sarebbe effettivamente presentato, procurandoci il materiale necessario per mostrare le parole (fogli, protezioni, componenti elettronici e di supporto).

2.1 Installazione ambiente di sviluppo

Prima di iniziare a lavorare è stato necessario impostare un ambiente comune ai membri del team e definire da dove cominciare. Quindi si è deciso di studiare i moduli da sviluppare così da identificare i vari componenti che sarebbero stati utili per la realizzazione del prodotto.

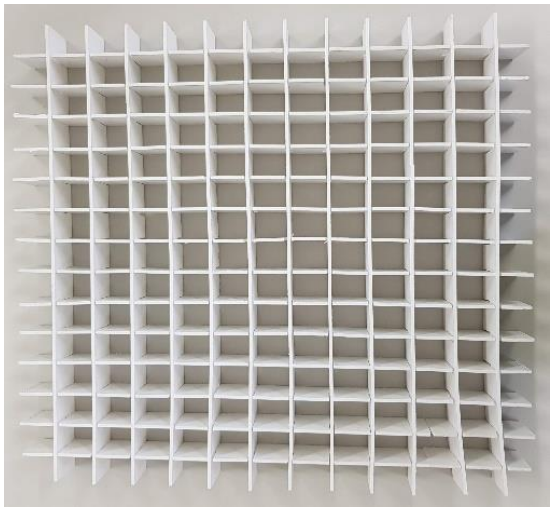
2.1.1 Librerie

- Librerie Adafruit
  - Adafruit NeoPixel
    Libreria che presenta funzioni basate per il controllo di strisce di led RGB prodotte da Adafruit. Nella libreria ci sono funzionalità che sono necessarie per lo sviluppo del progetto, come ad esempio l'accensione di un led di un certo colore.
    Link: https://github.com/adafruit/Adafruit_NeoPixel
  - RTClib
    Gestione di un dispositivo RTC in un ambiente di sviluppo Arduino. La libreria presenta metodi utili per ottenere l'orario corrente e tutti i suoi componenti (anno, mese, giorno, ore, minuti, secondi).
    Link: https://github.com/adafruit/rtclib
- Librerie Arduino e Fishino
  - WiFi
    Grazie a questa libreria, il dispositivo sarà in grado di connettersi a internet. Ciò serve per potersi connettere a un server e ricevere l'orario via wireless.
    Link: https://www.arduino.cc/en/Reference/WiFi
  - Fishino
    Se si usa un Fishino è necessario scaricare la libreria dal sito ufficiale per integrarla in Arduino IDE. Questa libreria è essenziale per il funzionamento del dispositivo e integrare i vari moduli come WiFi e RTC.
    Link: https://www.fishino.it/download-libraries-it.html

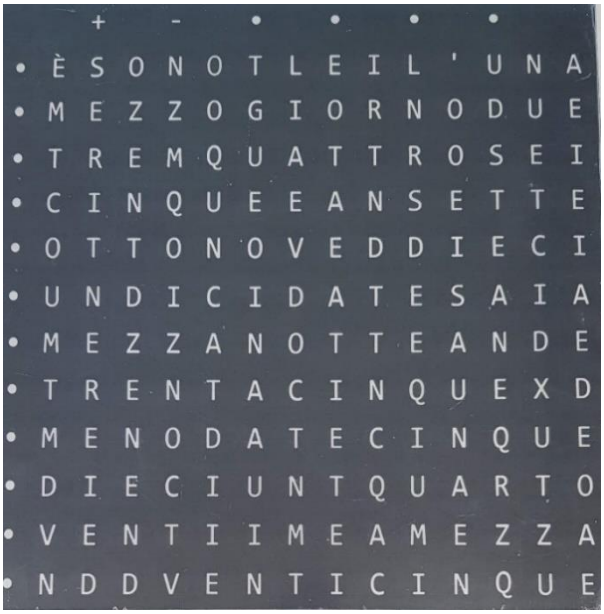
2.1.2 Scatola del word clock

Il “corpo” del prodotto comprende una scatola di legno dove viene piazzata la striscia di led e sopra di essa viene messo un sostegno che separa ogni led dagli altri.

All’interno della scatola principale viene posizionata una base su cui mettere la striscia di led in modo che faccia “avanti e indietro”. Quindi non è semplice identificare un singolo led (che dalla libreria viene riconosciuto tramite un numero incrementale) e per chiarire meglio il sistema si potrebbe creare una matrice che distingue i led in file anziché lungo la striscia. I led sono in totale 195 (13 x 15) e sono tutti RGB.



Sopra ai led va messo un sostegno che funge anche da separatore, in modo che quando viene acceso un led la luce non vada a espandersi accendendo lettere indesiderate.



Infine sopra ai led viene messo il foglio che mostra le parole che formano l'orario. Esso è formato da lettere che formano numeri, congiunzioni e verbi e simboli che rappresentano minuti e secondi.
Ecco i casi che mostrano determinate parole:

- Ore
  - "È"
    - "L'una"
    - "Mezzogiorno"
    - "Mezzanotte"
  - "Sono le"
    - "Due"
    - "Tre"
    - "Quattro"
    - "Cinque"
    - "Sei"
    - "Sette"
    - "Otto"
    - "Nove"
    - "Dieci"
    - "Undici"
- Minuti
  - "E"
    - "Cinque"
    - "Dieci"
    - "Un quarto"
    - "Venti"
    - "Venticinque"
    - "Mezza"
    - "Trentacinque"
  - "Meno"
    - "Cinque"
    - "Dieci"
    - "Un quarto"
    - "Venti"
    - "Venticinque"
  - "+" e "-" si aggiungono o sottraggono da uno a quattro minuti quando non sono multipli di 5.
- Secondi
  - Viene acceso un pallino ogni 5 secondi.

2.2 Analisi e verifica funzionamento componenti

2.2.1 Striscia di led

Per poter sviluppare il prodotto è stato inizialmente necessario verificare che la striscia di led funzionasse correttamente. Per fare ciò abbiamo usato un Arduino Mega 2560 e abbiamo caricato il programma *strandtest* della libreria di Adafruit. Però non siamo stati attenti e abbiamo avviato il test senza seguire le istruzioni del programma e della guida, le quali affermano che è necessario disporre di una resistenza sul pin dei dati e di un condensatore per avere una situazione sicura in caso di problemi con la corrente.
Quindi dopo alcuni test e cambiando dei parametri siamo giunti alla conclusione che i led funzionassero tutti regolarmente, tuttavia ci sono dei problemi quando si accendono tanti diodi con un colore luminoso (bianco), infatti il colore tende a spegnersi e il sistema crolla forzando un riavvio. Ma tutto ciò si può risolvere con l'aiuto di un alimentatore in modo da dare più corrente all'inizio e alla fine della striscia.

```
void colorWipe(uint32_t c, uint8_t wait) {
  for (uint16_t i = 0; i < strip.numPixels(); i++) {
    strip.setPixelColor(i, c);
    strip.show();
    delay(wait);
  }
}
```

Questo è un metodo che accende tutti i led uno dopo l'altro. È stato utile per capire dopo quanti led accesi di bianco il sistema sarebbe saltato. Inoltre è stato scoperto anche che per definire il colore di un led bisogna usare il tipo di variabile *uint32_t*, altrimenti alcuni colori non verrebbero indicati in modo giusto.

2.2.2 RTC

Il Real Time Clock è un modulo integrato nel Fishino che permette di salvare e aggiornare le informazioni riguardanti data e ora.

```
#include "RTClib.h"

void setup()
{
  // Impostazione del RTC
  if (!rtc.begin())
  {
    Serial.println("Impossibile trovare RTC");
    while (1)
      ;
  }
  // Verifica funzionamento RTC
  // Inserisce l'orario del computer durante la compilazione
  else if (!rtc.isrunning())
  {
    Serial.println("RTC non è in funzione!");
  }
  else
  {
    // Se vuoi un orario personalizzato, togli il commento alla riga successiva
    // l'orario: ANNO, MESE, GIORNI, ORA, MINUTI, SECONDI
    //rtc.adjust(DateTime(2014, 1, 12, 0, 59, 40));
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
  }
}
```

Viene importata la libreria e nel metodo di setup viene impostato il RTC controllando che funzioni. Successivamente si può impostare il dispositivo tramite un orario personalizzato oppure prendendolo dal computer.

```
void loop()
{
  DateTime now = rtc.now(); //creo istanza ora/data
  int hour = now.hour();
  int minute = now.minute();
  int second = now.second();
  Serial.print(now.year(), DEC); //stampo anno in decimale
  Serial.print('/');
  Serial.print(now.month(), DEC); //stampo mese in decimale
  Serial.print('/');
  Serial.print(now.day(), DEC); //stampo giorno in decimale
  Serial.print(" (");
  Serial.print(now.hour(), DEC); //stampo ora in decimale
  Serial.print(':');
  Serial.print(now.minute(), DEC); //stampo minuto in decimale
  Serial.print(':');
  Serial.print(now.second(), DEC); //stampo secondi in decimale
  Serial.print(") ");
  Serial.println();
}
```

Per testare il modulo si definisce un nuovo DateTime chiamando il metodo now(). Da ciò si può ricavare anno, mese, giorno, ora, minuto e secondo. Queste informazioni vengono stampate sul monitor seriale.

2.3 Design dell'architettura del sistema

La seguente è la struttura delle cartelle del progetto:

- Analisi
  - Domande_QDC
  - Pianificazione
- Diari
  - img
  - 2019_XX_XX_Diario_Word_Clock
- Documentazione
  - img
  - adafruit-neopixel-uberguide
  - Appunti
  - Documentazione_Word_Clock
- Presentazione
  - Presentazione_Word_Clock
- Progettazione
  - Matrice
- src
  - Arduino
    - buttontest
    - pixeltest
    - strandtest
  - Fishino
    - RTClcd
    - RTClcd2
    - RTCneopixel
      - FishinoUdpNtpClient
      - FishinoUdpNtpClientBuffer
      - FishinoUdpNtpClientProtothread
  - WordClock
- WordClock_web
  - css
  - js
  - index
  - indexDE
  - indexIT

Il codice sorgente principale si trova sotto "src/WordClock" e il resto viene usato come test prima di essere integrato.

3 Implementazione

L'implementazione del prodotto è composta semplicemente da un file Arduino che contiene tutti i moduli necessari per il funzionamento del word clock. Tuttavia, durante questo capitolo sono stati creati altri piccoli progetti utili per capire il meccanismo di alcuni moduli.

3.1 Definizione dei parametri

Inizialmente vanno definiti i componenti del sistema: il RTC e la striscia di led, dove si determina il numero di led, il pin dei dati e un paio di informazioni sulle proprietà dei led.

```
#define PIN 6
RTC_DS1307 rtc;
Adafruit_NeoPixel strip = Adafruit_NeoPixel(195, PIN, NEO_GRB + NEO_KHZ800);
```

È stato creato un array bidimensionale di interi per rappresentare la matrice di led. Questo è dovuto al fatto che abbiamo deciso di non mettere lo 0 in alto a sinistra, bensì in alto a destra, per poi continuare verso il basso e da destra verso sinistra. I numeri rappresentano l'ordine di accensione dei led: più il numero dell'array è basso, prima si accenderà. Inoltre creando questo array è più semplice definire ogni singolo led ed è più facile implementare dei cicli per controllarli.

```
const int pixels[13][15] = {
  {182, 181, 156, 155, 130, 129, 104, 103, 78, 77, 52, 51, 26, 25, 0},
  {183, 180, 157, 154, 131, 128, 105, 102, 79, 76, 53, 50, 27, 24, 1},
  {184, 179, 158, 153, 132, 127, 106, 101, 80, 75, 54, 49, 28, 23, 2},
  {185, 178, 159, 152, 133, 126, 107, 100, 81, 74, 55, 48, 29, 22, 3},
  {186, 177, 160, 151, 134, 125, 108, 99, 82, 73, 56, 47, 30, 21, 4},
  {187, 176, 161, 150, 135, 124, 109, 98, 83, 72, 57, 46, 31, 20, 5},
  {188, 175, 162, 149, 136, 123, 110, 97, 84, 71, 58, 45, 32, 19, 6},
  {189, 174, 163, 148, 137, 122, 111, 96, 85, 70, 59, 44, 33, 18, 7},
  {190, 173, 164, 147, 138, 121, 112, 95, 86, 69, 60, 43, 34, 17, 8},
  {191, 172, 165, 146, 139, 120, 113, 94, 87, 68, 61, 42, 35, 16, 9},
  {192, 171, 166, 145, 140, 119, 114, 93, 88, 67, 62, 41, 36, 15, 10},
  {193, 170, 167, 144, 141, 118, 115, 92, 89, 66, 63, 40, 37, 14, 11},
  {194, 169, 168, 143, 142, 117, 116, 91, 90, 65, 64, 39, 38, 13, 12}
};
```

Sono state dichiarate tutte le costanti che rappresentano i colori principali (in RGB) da attribuire ai led.

```
const uint32_t red = strip.Color(255, 0, 0);
const uint32_t green = strip.Color(0, 255, 0);
const uint32_t blue = strip.Color(0, 0, 255);
const uint32_t white = strip.Color(255, 255, 255);
const uint32_t black = strip.Color(0, 0, 0);
const uint32_t orange = strip.Color(255, 165, 0);
```

Un passo importante per il funzionamento del word clock è l'impostazione della connessione a internet. Ciò consente la ricezione di pacchetti che contengono le informazioni contenenti l'orario.

```
#ifndef __MY_NETWORK_H
#define MY_SSID "<NomeWiFi>"
#define MY_PASS "<Password>"
#define GATEWAY xxx, xxx, xxx, xxx
#define NETMASK xxx, xxx, xxx, xxx
#endif
#ifdef IPADDR
IPAddress ip(IPADDR);
#ifdef GATEWAY
IPAddress gw(GATEWAY);
#else
IPAddress gw(ip[0], ip[1], ip[2], 1);
#endif
#ifdef NETMASK
IPAddress nm(NETMASK);
#else
IPAddress nm(xxx, xxx, xxx, xxx);
#endif
#endif
```

Quindi vanno definite le informazioni riguardanti il server NTP e i pacchetti UPD con i dati sull'orario. Si definiscono la porta, l'indirizzo del server, la dimensione e la ricezione dei messaggi.

```
unsigned int localPort = 2390;
IPAddress timeServer(129, 6, 15, 28);
const int NTP_PACKET_SIZE = 48;
byte packetBuffer[NTP_PACKET_SIZE];
FishinoUDP Udp;
```

3.2 Setup()

Prima di parlare del metodo di setup, ci sono due metodi “di supporto” da spiegare.
printWifiStatus(), che stampa le informazioni sulla connessione WiFi e l’intensità del segnale.

```
void printWifiStatus()
{
  Serial.print("SSID: ");
  Serial.println(Fishino.SSID());
  IPAddress ip = Fishino.localIP();
  Serial << F("IP Address: ");
  Serial.println(ip);
  long rssi = Fishino.RSSI();
  Serial << F("signal strength (RSSI):");
  Serial.print(rssi);
  Serial << F(" dBm\n");
}
```

Poi c’è un metodo che viene utilizzato per inviare una richiesta al server NTP passato e inizializza il pacchetto contenente le informazioni riguardanti l’orario.

```
void sendNTPpacket(IPAddress &address, unsigned long waitTime)
{
  // Invio pacchetto al server NTP
  Serial << F("Invio richiesta UDP...");
  // Azzera il buffer di ricezione NTP
  memset(packetBuffer, 0, NTP_PACKET_SIZE);
  // Inizializza i valori da inviare al server NTP
  // (vedere URL del server per dettagli sul formato pacchetto)
  packetBuffer[0] = 0b11100011;
  packetBuffer[1] = 0;
  packetBuffer[2] = 6;
  packetBuffer[3] = 0xEC;
  packetBuffer[12] = 49;
  packetBuffer[13] = 0x4E;
  packetBuffer[14] = 49;
  packetBuffer[15] = 52;
  /* Tutti i campi del paccketto NTP sono stati impostati
  è quindi possibile inviare il pacchetto di richiesta di data/ora */
  /* Invia la richiesta NTP alla porta 123
  beginPacket() apre solo la connessione */
  Udp.beginPacket(address, 123);
  // Riempie il buffer di invio UDP con i dati del pacchetto
  Udp.write(packetBuffer, NTP_PACKET_SIZE);
  // Termina e invia il pacchetto
  Udp.endPacket();
  Serial << "OK\n";
  // Attesa per vedere se una risposta è disponibile
  delay(waitTime);
}
```

Nel metodo di setup, viene inizialmente impostata la connessione WiFi riavviandola e cercando continuamente di connettersi finché non si trova un collegamento.

```
while (!Fishino.reset())
  Serial << F("Fishino RESET FAILED, RETRYING...\n");
Serial << F("Fishino WiFi RESET OK\n");
Fishino.setMode(STATION_MODE);
Serial << F("Connecting to AP...");
while (!Fishino.begin(MY_SSID, MY_PASS))
{
  Serial << ".";
  delay(2000);
}
Serial << "OK\n";
```

Poi viene impostata l'indirizzo e se va utilizzato il DHCP se non è stato definito un indirizzo statico.

```
#ifdef IPADDR
  Fishino.config(ip, gw, nm);
#else
  Fishino.staStartDHCP();
#endif
```

Con le seguenti operazioni il Fishino aspetta un indirizzo e le azioni di connessione a internet finiscono dopo la stampa a terminale delle informazioni sul WiFi.

```
Serial << F("Waiting for IP...");
while (Fishino.status() != STATION_GOT_IP)
{
  Serial << ".";
  delay(500);
}
Serial << "OK\n";
printWifiStatus();
```

Successivamente va instaurata una connessione con il server.

```
Serial << F("Starting connection to server...\n");
Udp.begin(localPort);
```

Dopo va avviata e accesa la striscia di led e va impostata la luminosità.

```
strip.begin();
strip.setBrightness(255);
strip.show();
```

Infine va configurato il RTC verificando che sia connesso e che sia in funzione, altrimenti si imposta la data e ora corrente.

```
if (!rtc.begin())
{
  Serial.println("Impossibile trovare RTC");
  while (1)
    ;
}
if (!rtc.isrunning())
{
  Serial.println("RTC non è in funzione!");
}
rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
```

3.3 getPacket()

Metodo che legge il pacchetto ricevuto dal NTP e lo formatta in modo da ricavarne le informazioni sull'orario. In questo caso si tratta dei secondi passati dal primo gennaio 1900.

```
unsigned long getPacket()
{
  Serial << F("Pacchetto ricevuto\n");
  // Stampa IP e porta remoti per mostrare la provenienza del pacchetto
  IPAddress remoteIp = Udp.remoteIP();
  uint32_t remotePort = Udp.remotePort();
  Serial << F("Remote IP   : ") << remoteIp << "\n";
  Serial << F("Remote port : ") << remotePort << "\n";
  // Abbiamo ricevuto un pacchetto, leggiamo i dati e inseriamoli in un buffer
  Udp.read(packetBuffer, NTP_PACKET_SIZE);
  // Il timestamp inizia dal byte 40 del pacchetto ricevuto, e consiste in 4 bytes
  // o due words, long. Innanzitutto estraiamo le due words
  unsigned long highWord = word(packetBuffer[40], packetBuffer[41]);
  unsigned long lowWord = word(packetBuffer[42], packetBuffer[43]);
  // Combiniamo i 4 bytes (o 2 words) in un long integer
  // che è il tempo NTP (secondi dal primo Gennaio 1900)
  unsigned long secsSince1900 = highWord << 16 | lowWord;
  Serial << F("Seconds since Jan 1 1900 = ") << secsSince1900 << "\n";
  return secsSince1900;
}
```

3.4 pixelOn()

È un metodo molto semplice ma importante per il funzionamento del sistema: tramite il passaggio di due parametri (indice del led e colore), questo metodo permette di accendere un led del word clock.

```
void pixelOn(int pixel, uint32_t color)
{
  strip.setPixelColor(pixel, color);
  strip.show();
}
```

3.5 generateWord()

Metodo che accende una parola del word clock con la definizione della riga, l'indice iniziale, l'indice finale e il colore dei led.

```
void generateWord(int row, int start, int end, uint32_t color)
{
  for (int i = start; i <= end; i++)
  {
    pixelOn(pixels[row][i], color);
  }
}
```

L'implementazione in questo caso risulta molto semplice grazie alla creazione della matrice che identifica ogni led. Infatti si può accendere una fila di led di una certa riga con un intervallo passato (start e end).

3.6 Generazione parole

I metodi di questo capitolo hanno il compito di richiamare generateWord() e accendere sul word clock una certa parola. In questo modo il codice sorgente risulta più ordinato e comprensibile.

```
void pausa(uint32_t color)
{
  generateWord(0, 8, 12, color);
}
```

In questo esempio si chiama il metodo per scrivere la parola “pausa”, infatti la parola si trova nella prima riga e parte dalla nona colonna e finisce nella tredicesima colonna.

```
void sonoLe(uint32_t color)
{
  generateWord(1, 2, 5, color);
  generateWord(1, 7, 8, color);
}
```

```
void unQuarto(uint32_t color)
{
  generateWord(10, 6, 7, color);
  generateWord(10, 9, 14, color);
}
```

Questi sono altri esempi dell’implementazione di questo tipo di metodi. Quindi per scrivere una parola si usano questi metodi che sono tutti molto simili tra loro.

3.7 printBreak()

In questo metodo vengono passati i parametri contenenti ora e minuto e da ciò si stampa la parola “pausa” in base al colore che indica a che punto è la pausa.

```
void printBreak(int hour, int minute)
{
  uint32_t color = black;
  // Pausa OK (9:50 - 10:01 / 14:45 - 14:56)
  if (
      (hour == 9 && minute >= 50) ||
      (hour == 10 && minute >= 0 && minute < 2) ||
      (hour == 14 && minute >= 45 && minute < 57))
  {
    color = green;
  }
  // Pausa sta per finire (10:02 - 10:05 / 14:57 - 15:00)
  else if (
      (hour == 10 && minute >= 2 && minute < 5) ||
      (hour == 14 && minute >= 57 && minute <= 59))
  {
    color = red;
  }
  // Pausa non c'è, led spenti
  else
  {
    color = black;
  }
  // Imposto la parola
  pausa(color);
}
```

3.8 generateSeconds()

Questo metodo è utile per stampare i secondi. Per fare questo utilizza il parametro passato che indica quanti pallini vanno accesi. Poi si usa un semplice ciclo che accende i pallini necessari.

```
void generateSeconds(int length, uint32_t color)
{
  for (int i = 1; i <= length; i++)
  {
    generateWord(i, 0, 0, color);
  }
}
```

3.9 printSecond()

Questo è l'ultimo metodo riguardante i secondi. Ad esso viene passato il secondo corrente e chiama generateSeconds() tramite uno switch che passa i casi dei secondi da 0 a 59.

```
void printSecond(int second)
{
  switch (second)
  {
  case 0:
    generateSeconds(12, black);
    break;

  case 1 ... 4:
    generateSeconds(1, white);
    break;

  case 5 ... 9:
    generateSeconds(2, white);
    break;

  case 10 ... 14:
    generateSeconds(3, white);
    break;
```

3.10 printTime()

È la funzione principale che utilizza tutti i metodi spiegati precedentemente per stampare l'orario completo sul word clock. Esso necessita di tre parametri: ora, minuto e secondo.

```
void printTime(int hour, int minute, int second)
{
  boolean meno = false;

  //PAUSA
  printBreak(hour, minute);

  //Più o meno
  if (minute < 35)
  {
    meno = false;
  }
  else
  {
    meno = true;
  }
```

Inizialmente viene impostata la pausa e viene deciso tramite una variabile booleana se impostare il "meno".

```
  if (hour == 12)
  {
    if (minute < 35)
    {
      egrave(white);
      mezzogiorno(white);
    }
    else if (minute == 35)
    {
      egrave(white);
      mezzogiorno(white);
      eTrentacinque(white);
    }
    if (minute > 35)
    {
      eTrentacinque(black);
      egrave(white);
      mezzogiorno(black);
      una(white);
    }
  }
```

Poi per impostare l'ora c'è una serie di if che stampa l'ora corrente oppure quella successiva se il minuto è maggiore a 35.

```
if (minute >= 5 && minute <= 35)
{
  e(white);
}
else
{
  e(black);
}
```

Tramite queste condizioni si definisce se stampare la parola “e” o meno. In genere questa lettera viene stampata quando i minuti sono da 5 a 35.

```
int diff = minute - int(minute / 10) * 10;

//Minuti
if (meno == false)
{

  if (minute % 5 != 0)
  {
    piu(white);
    menoSign(black);
  }
  else
  {
    piuMenoEMinuti(black);
  }
```

Per iniziare con la stampa dei minuti, si istanzia una differenza che definisce se è necessario usare i pallini dei minuti quando non sono esattamente multipli di 5 (“+” e “-“). Poi si stampa il più appunto se i minuti non sono multipli di 5, altrimenti si spengono.

```
if (diff != 0 && diff != 5)
{
  if (diff >= 1 && diff <= 4 || diff >= 6 && diff <= 9)
  {
    printMinutePoints(3, white, black);
  }
  if (diff >= 2 && diff <= 4 || diff >= 7 && diff <= 9)
  {
    printMinutePoints(4, white, black);
  }
  if (diff >= 3 && diff <= 4 || diff >= 8 && diff <= 9)
  {
    printMinutePoints(5, white, black);
  }
  if (diff == 4 || diff == 9)
  {
    printAllMinute(white);
  }
}
```

Queste linee di codice definiscono quali sono i punti dei minuti da accendere tramite l'aiuto del metodo printMinutePoints().

```
if (minute >= 5 && minute < 10)
{
  cinqueMinuti(white);
}
else if (minute >= 10 && minute < 15)
{
  dieciMinuti(white);
  cinqueMinuti(black);
}
else if (minute >= 15 && minute < 20)
{
  unQuarto(white);
  dieciMinuti(black);
}
else if (minute >= 20 && minute < 25)
{
  unQuarto(black);
  venti(white);
}
```

In seguito ci sono tutte le condizioni per i minuti restanti, dove vengono accesi oppure spenti i minuti tramite i metodi che generano le parole.

```
printSecond(second);
```

Infine si chiama il metodo printSecond() e fa tutto il lavoro per ciò che riguarda la stampa dei pallini dei secondi.

4 Test

4.1 Protocollo di test

Definire in modo accurato tutti i test che devono essere realizzati per garantire l'adempimento delle richieste formulate nei requisiti. I test fungono da garanzia di qualità del prodotto. Ogni test deve essere ripetibile alle stesse condizioni.

Test Case:	TC-001	**Nome:**	Test hardware
Riferimento:	REQ-001		
Descrizione:	È necessario verificare che la striscia Neopixel funzioni correttamente.		
Prerequisiti:	Avere a disposizione una striscia led neopixel e un Fishino		
Procedura:	1. Collegare la striscia Neopixel al Fishino correttamente 2. Caricare il codice di pixeltest.ino		
Risultati attesi:	La striscia Neopixel dovrebbe accendersi seguendo il pattern che è stato definite nel codice.		

Test Case:	TC-002	**Nome:**	Test hardware
Riferimento:	REQ-001		
Descrizione:	È necessario verificare che il modulo WiFi funzioni		
Prerequisiti:	Avere a disposizione un Fishino		
Procedura:	1. Collegare il Fishino ramite l'apposito cavo al computer. 2. Caricare il codice di FishinoUdpNtpClient.ino 3. Aprire il monitor Seriale		
Risultati attesi:	Fishino dovrebbe connettersi al time server e ritornare l'orario		

Test Case:	TC-003	**Nome:**	Codice Word Clock
Riferimento:	REQ-002		
Descrizione:	Avere un Word Clock funzionante		
Prerequisiti:	Avere a disposizione una striscia led neopixel un Fishino		
Procedura:	1. Collegare la striscia Neopixel al Fishino correttamente 2. Caricare il codice Word-Clock.ino		
Risultati attesi:	Il Fishino dovrebbe collegarsi alla rete WiFi e dovrebbe stampare sulla matrice il giusto or		

Import a card with KIC, KID and KIK keys, but not shown with the GUI

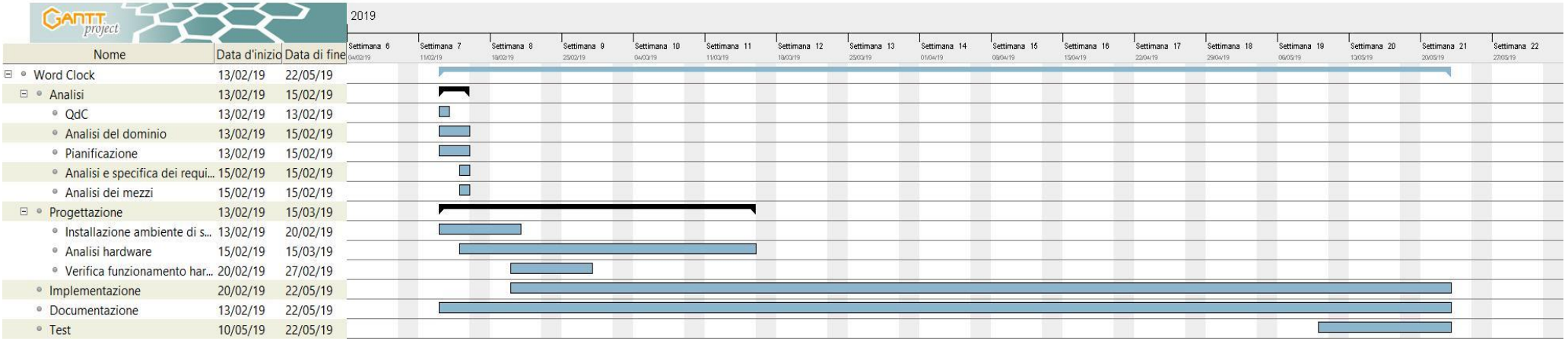
4.2 Risultati test

TC-001	OK	
TC-002	OK	
TC-003	FALLITO	Il sottorequisito 003 non funziona.

4.3 Mancanze/limitazioni conosciute

La grande mancanza del prodotto è il controllo tramite il modello fisico. È successo ciò per colpa della sottovalutazione della situazione a un certo punto del progetto, visto che si pensava di essere messi a buon punto e abbiamo voluto rallentare.

5 Consuntivo



I primi capitoli del progetto sono stati eseguiti in linea con la pianificazione, ma la grande differenza con la pianificazione iniziale sono i tempi dell'implementazione. Infatti è stato speso molto più tempo su ciò perché ci sono state alcune difficoltà con i componenti del prodotto che non funzionavano costantemente (come spiegato nei diari di lavoro).

6 Conclusioni

Per realizzare questo progetto abbiamo speso molte ore, nonostante ciò non crediamo che si sia trattato di una perdita di tempo dato che questo progetto ci ha aiutati a rafforzare le nostre capacità di lavorare in team e di conoscere dei nuovi componenti elettronici a noi prima sconosciuti, come il Fishino. Sicuramente questo progetto non servirà a cambiare il mondo, tuttavia nel suo piccolo può essere utile per coloro che transitano nel quarto piano della SAMT, dando loro la possibilità di sapere velocemente l'ora e di osservare, tramite dei cambi di colore, se sono le pause stanno per finire. Si tratta di un orologio abbastanza basilare da realizzare, tuttavia ci riteniamo soddisfatti del lavoro svolto e delle nozioni acquisite.
Purtroppo non tutti i requisiti sono stati realizzati, infatti non abbiamo avuto tempo di realizzare il sito web che controllasse l'orario. Inoltre, non siamo riusciti ad implementare il controllo dell'orologio tramite bottoni, seppure ci abbiamo provato.

6.1 Sviluppi futuri

Nel futuro, tramite il prototipo che abbiamo creato, si potrebbe realizzare una versione più grande di questo word clock, magari da appendere all'esterno della scuola per consentire a tutti di sapere quando è il momento di rientrare dalla pausa. Inoltre si potrebbe eventualmente realizzare delle versioni con lingue differenti, come l'inglese o il tedesco. Per finire, si potrebbe realizzare ciò che non siamo riusciti a finire nel progetto, ovvero il controllo tramite bottoni e tramite pagina web.

6.2 Considerazioni personali

Da questo progetto abbiamo imparato a gestirci meglio il lavoro, tramite strumenti come Atom.
Inoltre abbiamo compreso il funzionamento del Fishino, delle strisce di led Adafruit NeoPixel e come gestire un vero e proprio orologio.
Purtroppo abbiamo sottovalutato i requisiti, infatti non siamo riusciti a terminare alcune funzionalità del progetto.

7 Bibliografia

7.1 Bibliografia per libri

- Massimo Del Fedele, *Fishino*, Futura Group srl, 2017, 978-88-909529-5-1.

7.2 Sitografia

- https://fishino.it/fishino-uno-it.html, *documentazione fishino uno*,15.02.2019
- https://atom.io/packages/git-plus, *auto fetch github*,15.02.2019
- http://www.fishino.com/caricamento-wireless-degli-sketch-ota.html, *caricamento wireless degli sketch (ota)*, 22.02.2019
- https://www.markdowntutorial.com/lesson/4/, *inserire immagine in MD,* 22.02.2019
- https://aticleworld.com/http-get-and-post-methods-example-in-c/, *richiesta GET in C per connessione time server*, 22.02.2019
- http://www.fishino.com/arduino-ide-packages-it.html, *packages per l'ide di arduino – fishino*, 22.02.19
- https://www.adafruit.com/product/2969, *ricerca informazioni led NeoPixel*, 22.02.2019
- http://www.fishino.com/download-libraries-it.html, *librerie – Fishino*, 22.02.2019
- http://www.fishino.com/download-drivers-it.html, *drivers – Fishino*, 22.02.2019
- https://www.wikiinfo.net/t2800-come-funziona-rtc-del-fishino-codice, come funziona RTC del Fishino, 27.02.2019

8 Allegati

Elenco degli allegati:

- Quaderno dei compiti
- Diari di lavoro