




Tecnica
Mente
Dall'aula all'azienda.

Lasertag
Mazzaro Roberto

Adecco

Obiettivo

Riprodurre il gioco del lasertag

Cos'è il lasertag?

- Gioco di squadra
- Basato sulla simulazione militare
- Usato nell'addestramento militare

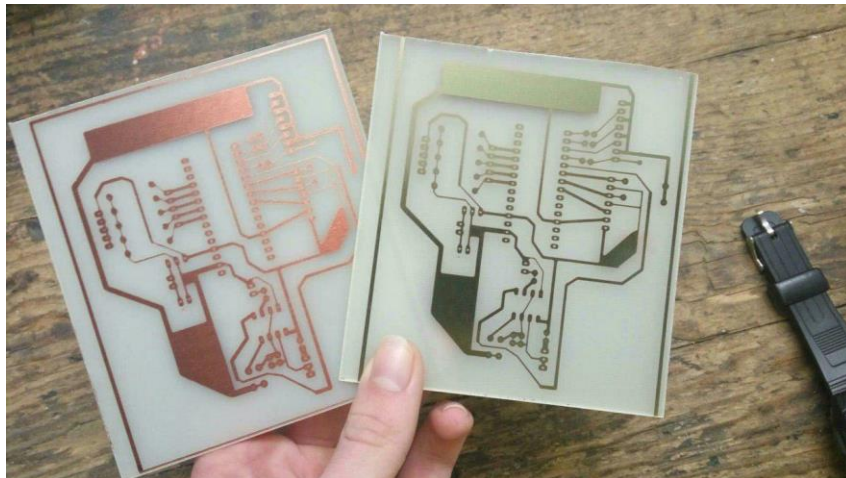
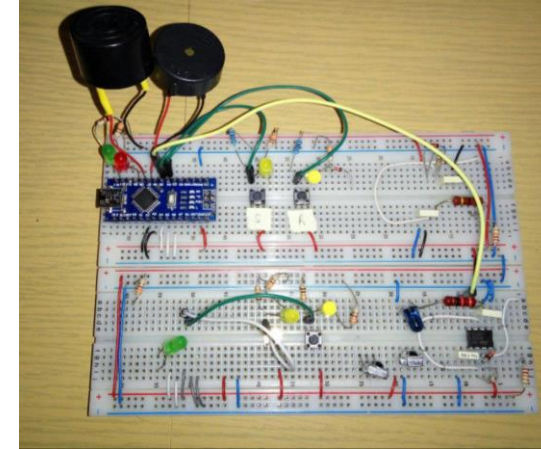
Come funziona?

- Si è muniti di un kit che può sparare e ricevere i colpi
- Bisogna colpire l'avversario fino ad eliminarlo



Fasi di progetto

- Capire come far comunicare due Arduino tramite la funzione PulseIn()
- Capire come inserire la comunicazione infrarosso
- Montare un prototipo su breadboard
- Scrivere le singole funzioni utilizzate nel software
- Disegnare, sviluppare e montare il PCB
- Mettere a punto il programma generale
- Disegnare la struttura per stampa in 3D
- Realizzare la struttura



```
Inserta completo | Arduino 1.8.5
File Modifica Sketch Strumenti Aiuto

Inserta completo | Segnala problema | Verifica | Spazio

int ledPin = 12; //LED di segnalazione, si accende quando si muove o quando si è senza colpi
int ledPin2 = 11; //LED di segnalazione si accende quando si apre per simulare la luce rossa che esce dalla canna di un fucile quando si apre
int buzzer = 8; //buzzer di segnalazione si accende quando si apre
int buzzer2 = 10; //buzzer di segnalazione, si accende quando si è colpiti o a sistema arrestato
int buzzer3 = 9; //segnale che va a pilotare il reset di un MPU la cui uscita è collegata ad un led ir
int buzzer4 = 7; //pulsante di spazio
int buzzer5 = 6; //pulsante di ricarica
int buzzer6 = 5; //pulsante infrarosso che riceve il colpo

//Ingressi impostazione arma e codice
int a0 = 0; //1 bit e impostano la squadra
int a1 = 1; //1 bit e impostano l'arma
int a2 = 2; //1 bit e impostano l'arma
int a3 = 3; //1 bit e impostano l'arma
int a4 = 4; //1 bit e impostano l'arma

//variabili impostazione arma e codice
int data[5] = {0,0,0,0,0}; //variabile che contiene il codice

//variabili che impostano l'arma
int vita = 0; //variabile che indica i punti vita, si decrementa ogni volta che si viene colpiti, aumenta se si viene curati
int colpi = 0; //variabile che indica il numero di colpi per caricatore, decrementa ogni volta che si apre un colpo
int caricatori = 0; //variabile che indica il numero di caricatori, decrementa ogni volta che si ricarica
int squadra = 0; //variabile che indica la squadra della propria arma
int arma = 0; //variabile che indica il tipo della propria arma

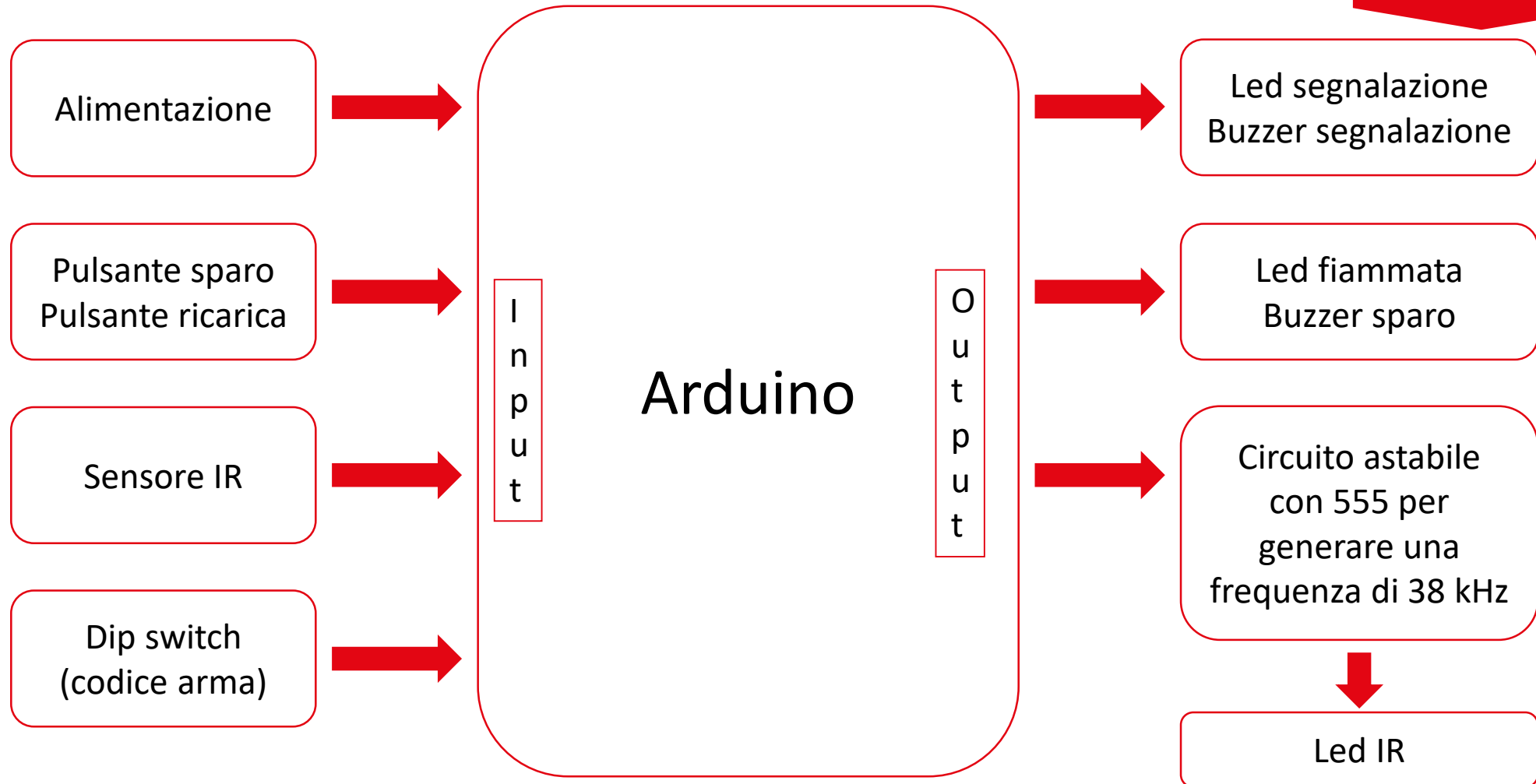
int squadra2 = 0;
int arma2 = 0;

//variabili di supporto per la conversione in binario del codice arma squadra
int base2 = 2;
int base22 = 1;

volatile int irq = 0; //variabili di supporto usate nella ISR sincrona
volatile int irq2 = 0;
volatile int irq3 = 0;

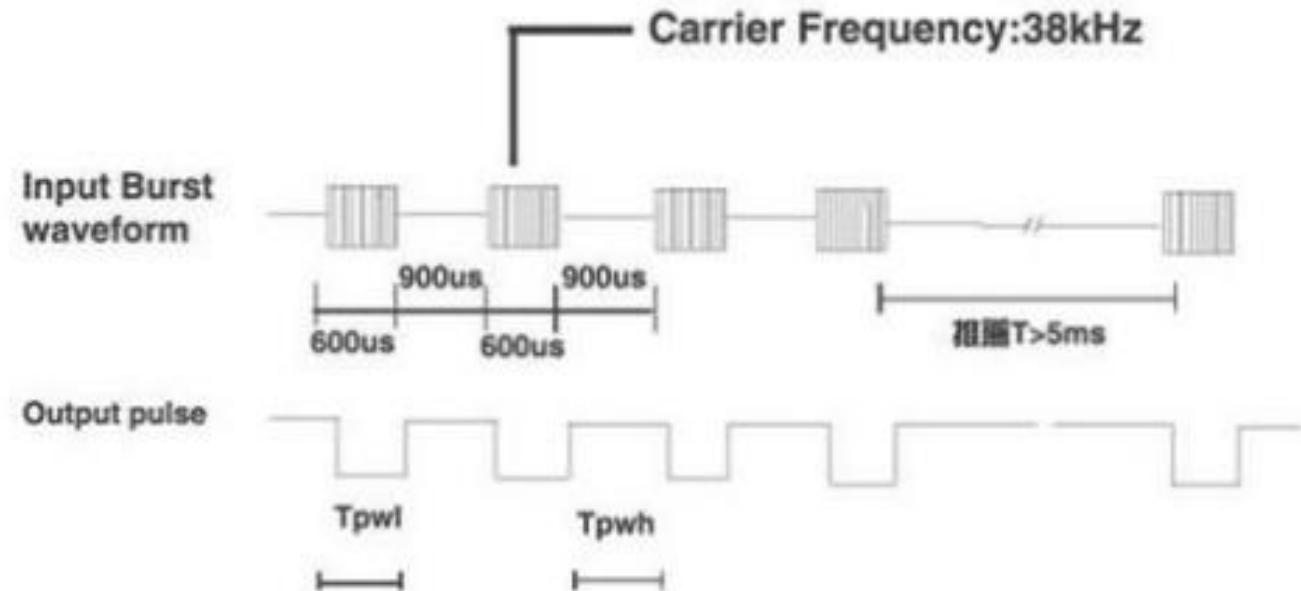
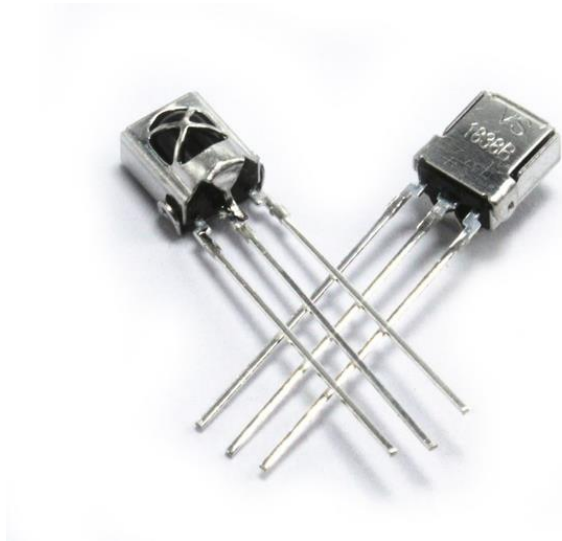
//variabili per codice in uscita
int bitStart = 1000; //variabile che indica il valore del bit di start
int bit1 = 0; //variabile che indica quando voglio spedire un bit 1
int bit0 = 0; //variabile che indica quando voglio spedire un bit 0
int bit2 = 0; //variabile che indica il valore di stato tra un bit e l'altro
```

Schema a blocchi di un kit



Sensore IR

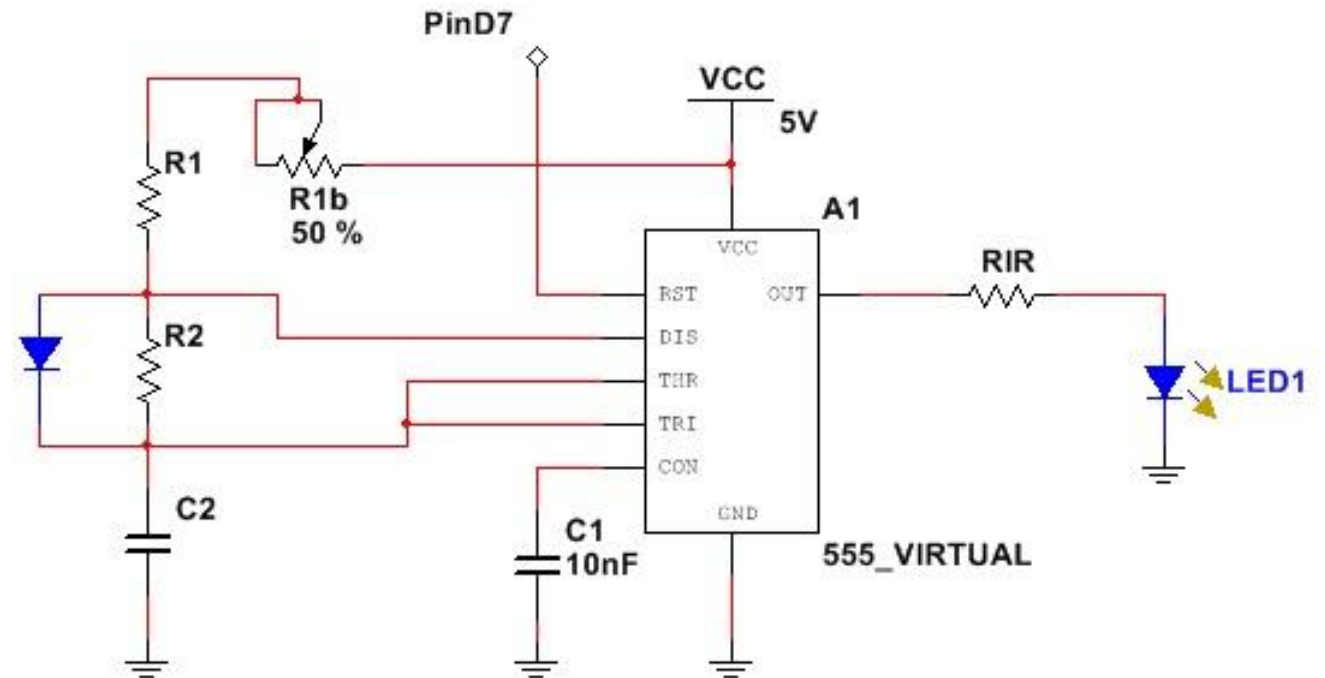
- Modello: TL1838
- Frequenza di lavoro: 38kHz
- Durata minima impulso 600μs



Hardware trasmissione

Il led IR va pilotato con una frequenza di 38kHz

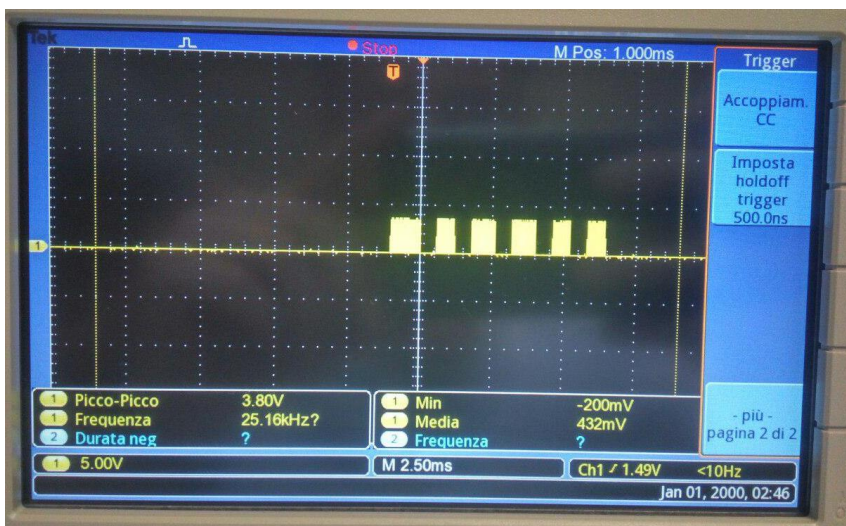
Siccome i timer di Arduino erano già occupati ho utilizzato un 555 in configurazione astabile



Misure con oscilloscopio

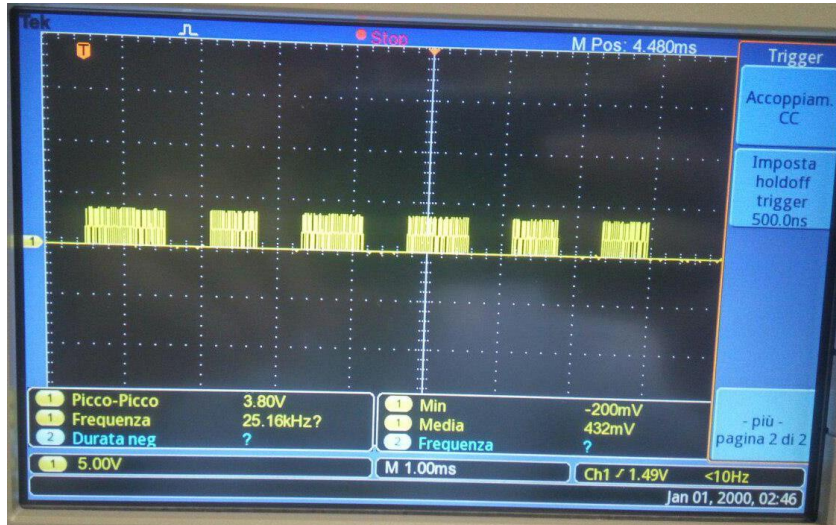


Segnale in uscita da Arduino che pilota il reset del 555

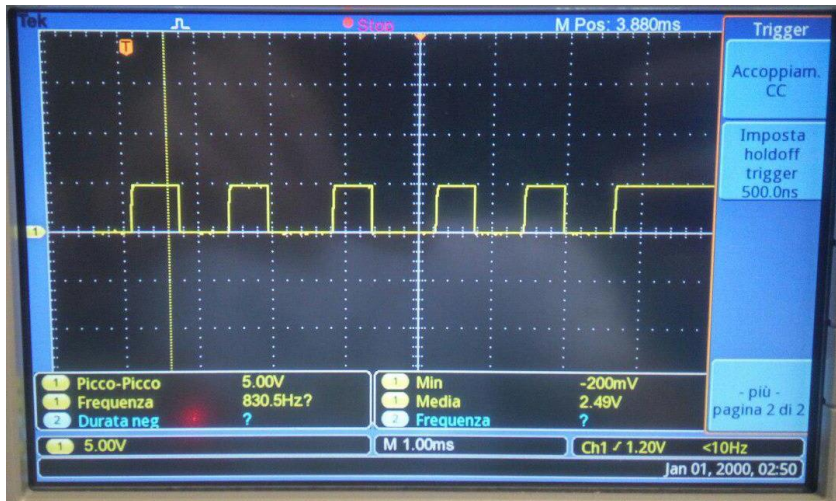


Uscita del 555, che viene trasmessa dal led IR

Misure con oscilloscopio



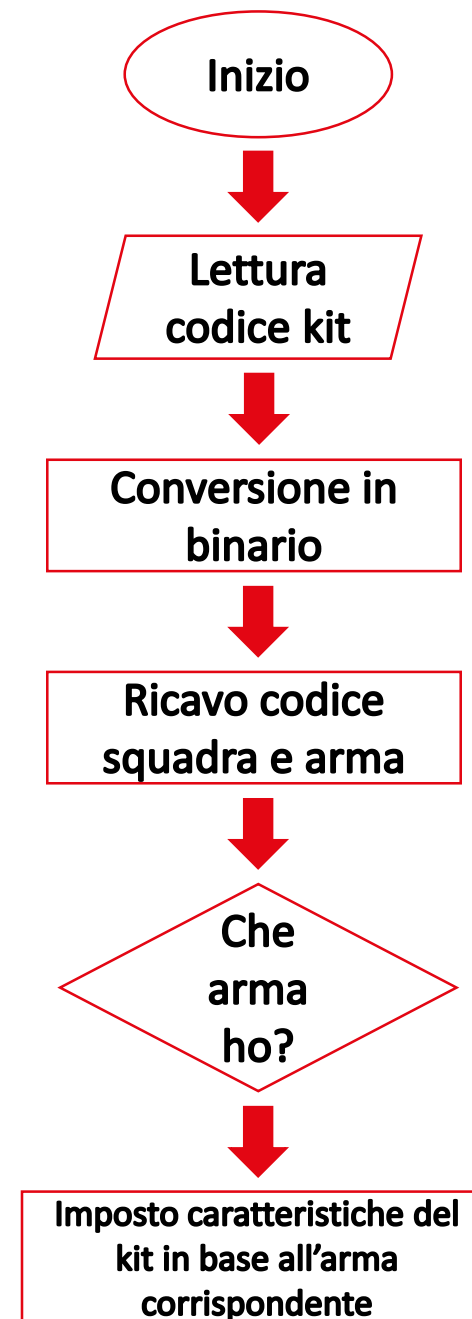
Segnale trasmesso dal
led IR



Segnale in uscita
dal sensore IR

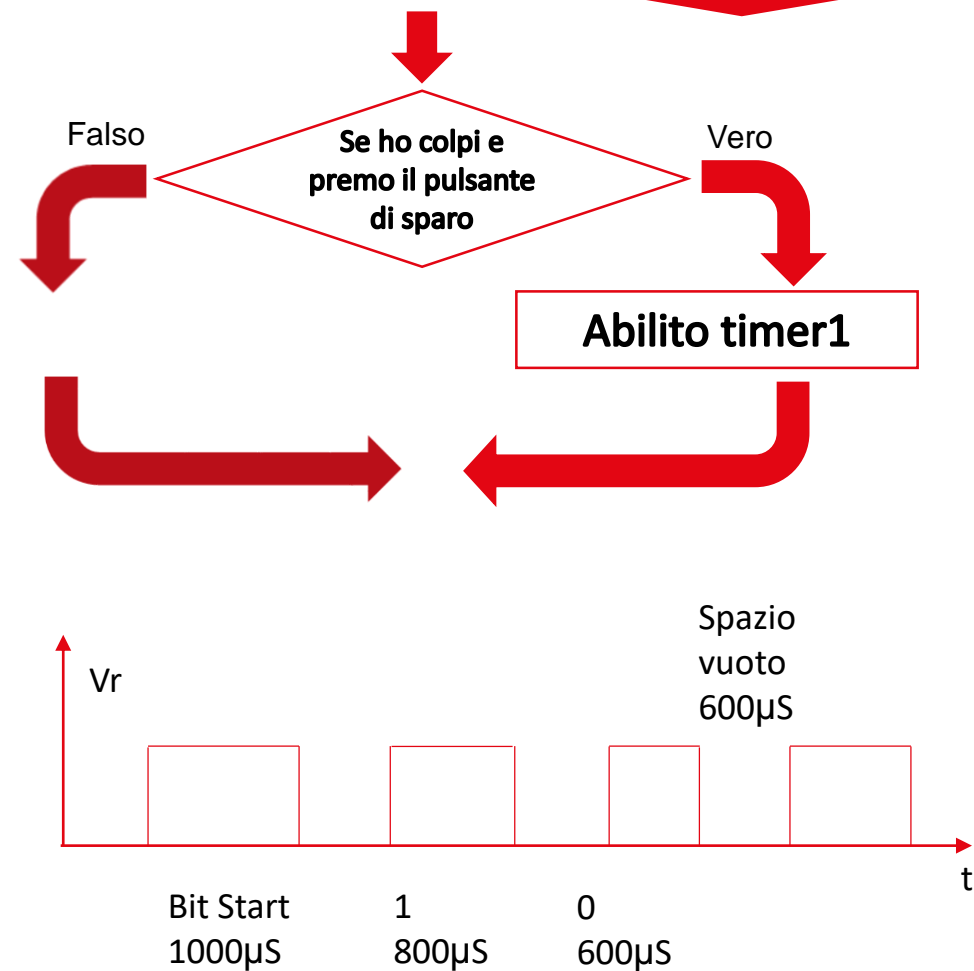
Acquisizione codice arma

- Eseguita nel void setup
- Lettura di 5 piedini analogici (per non occupare quelli digitali)
- Conversione dei valori analogici in valori digitali
- Estrazione codice squadra e arma
- Impostazione caratteristiche del kit a seconda del codice dell'arma



Funzione di sparo

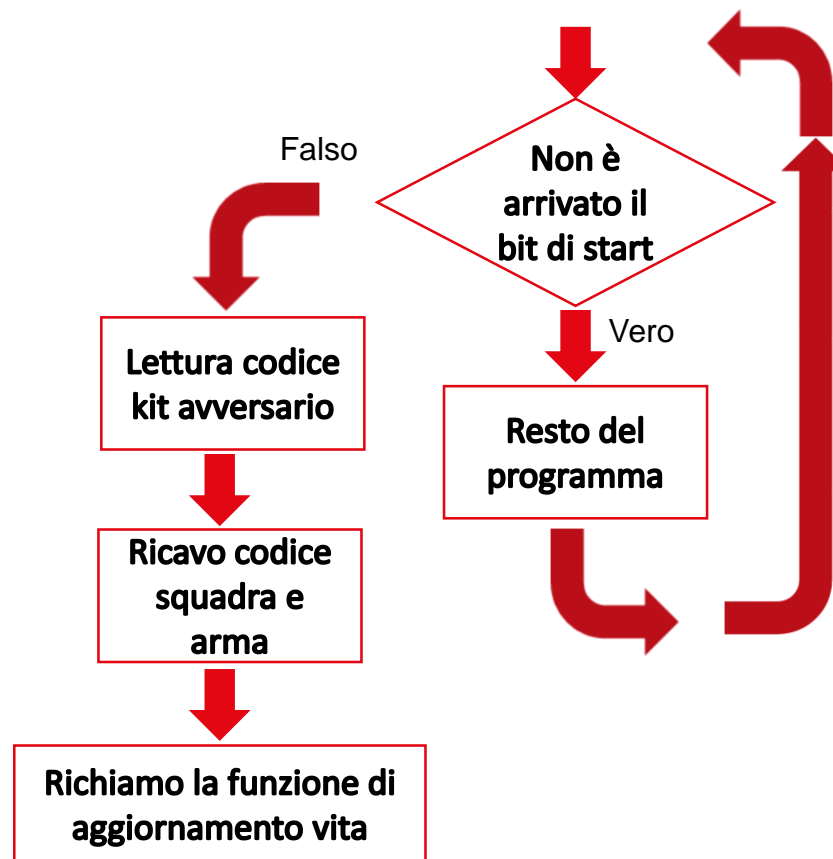
- Gestita con timer1, la frequenza dipende dall'arma
- Bit di start, si spedisce la frequenza di 38kHz per 1000μS
- Bit a 1, si spedisce la frequenza di 38kHz per 800μS
- Bit a 0, si spedisce la frequenza di 38kHz per 600μS
- Per separare un bit dall'altro non si spedisce niente per 600μS



Funzione di ricezione

Si basa sulla funzione pulseIn()

- Restituisce la durata dello stato alto/basso a partire dal fronte stabilito
- Utilizzata con timeout, se entro il tempo prestabilito non arriva il fronte desiderato la funzione restituisce come risultato 0; è necessario usare i timeout per non bloccare il resto del programma

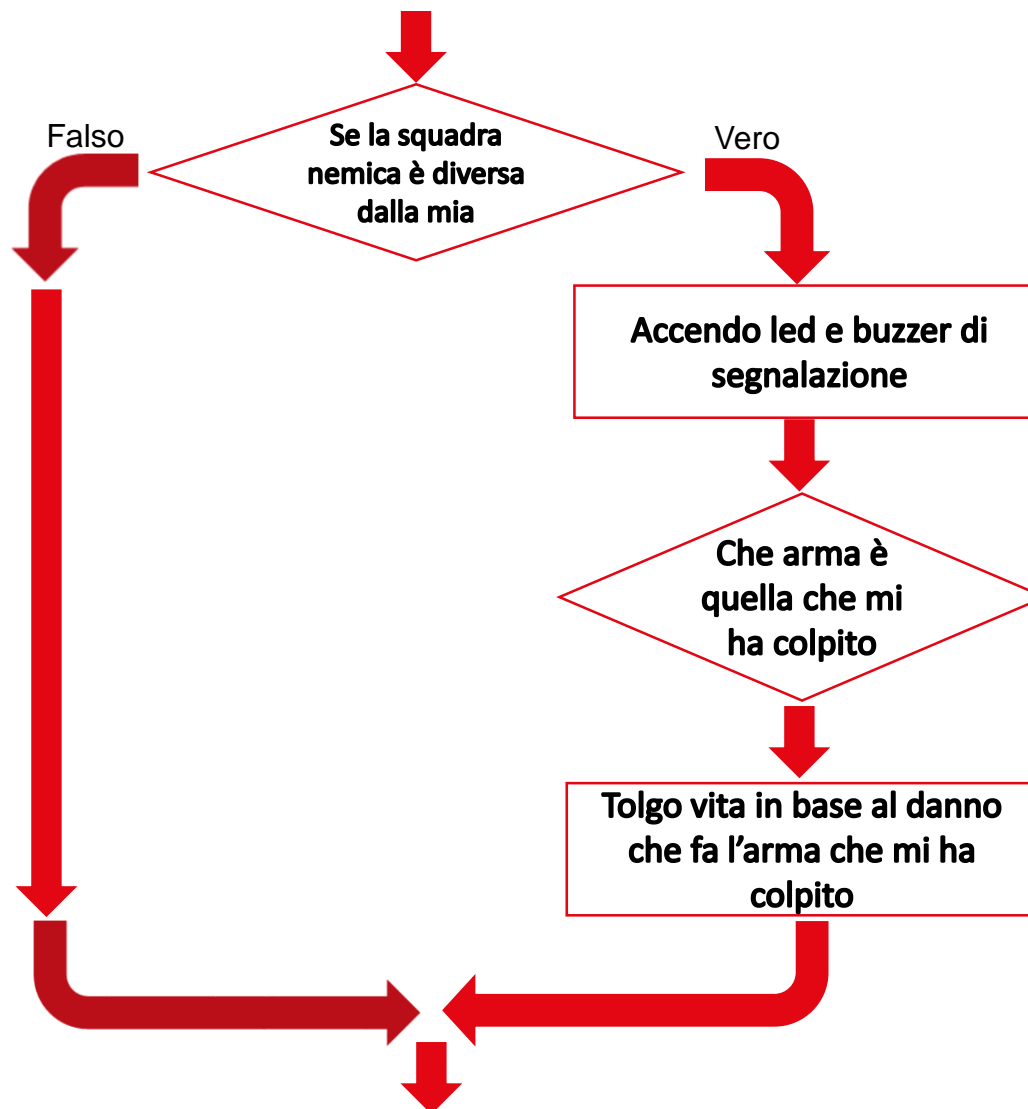


- Per individuare il bit di start c'è un ciclo while che cicla finché non arriva un impulso di almeno 950µs
- Rilevato il bit di start si va a rilevare la durata degli altri 5 impulsi, quelli di codice
- Si convertono i tempi in 0 e 1
- Si ricava il codice della squadra e dell'arma del kit avversario

Funzione aggiornamento vita

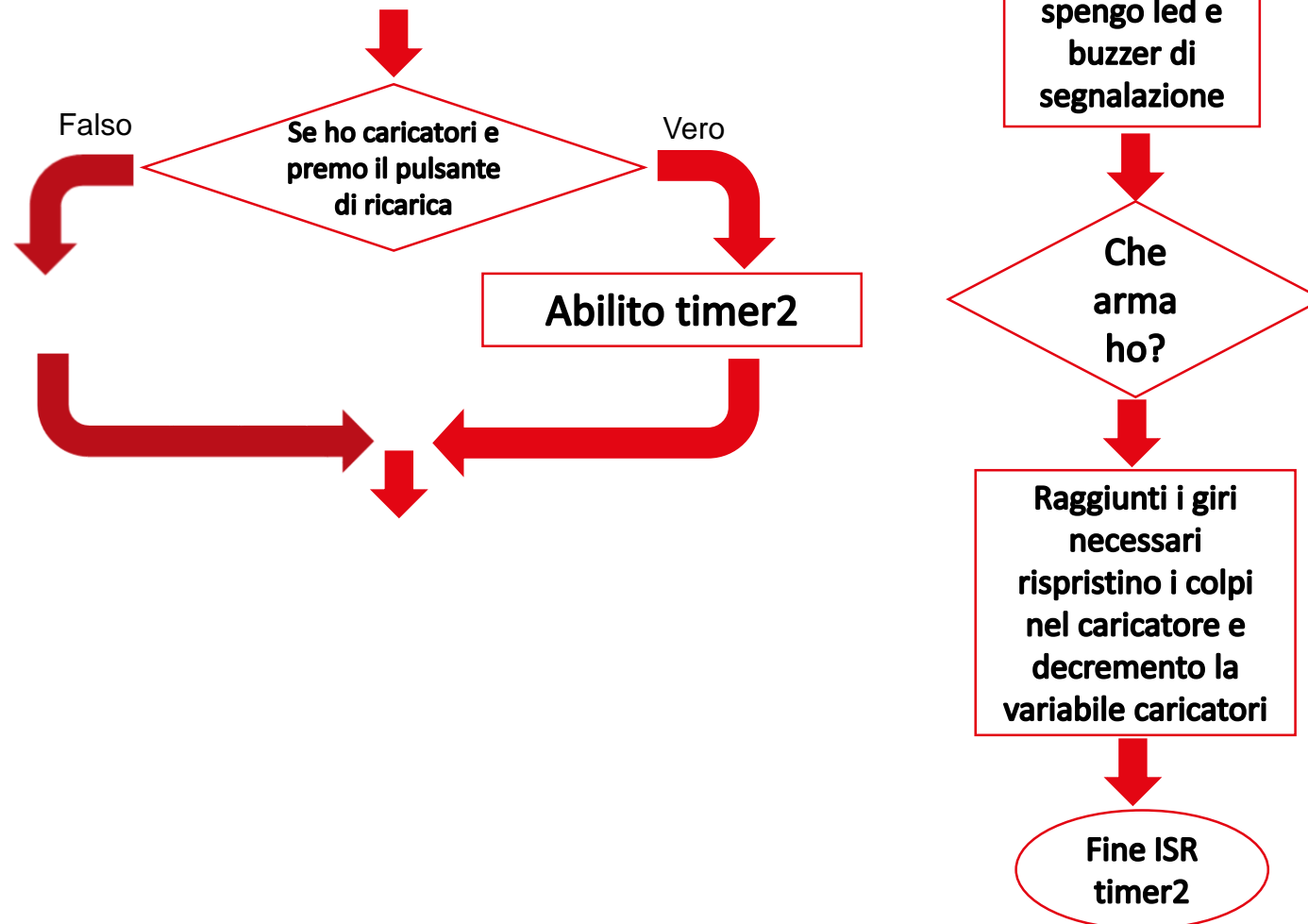
- Avviene una volta ricevuto il codice avversario
- Si controlla che la squadra dei due kit non sia la stessa
- Se la squadra è diversa si decrementa la variabile vita in base al codice arma avversario
- Si accendono il led ed il buzzer di segnalazione

La ricezione del colpo non può essere obbiettata perchè viene sancita dall'elettronica



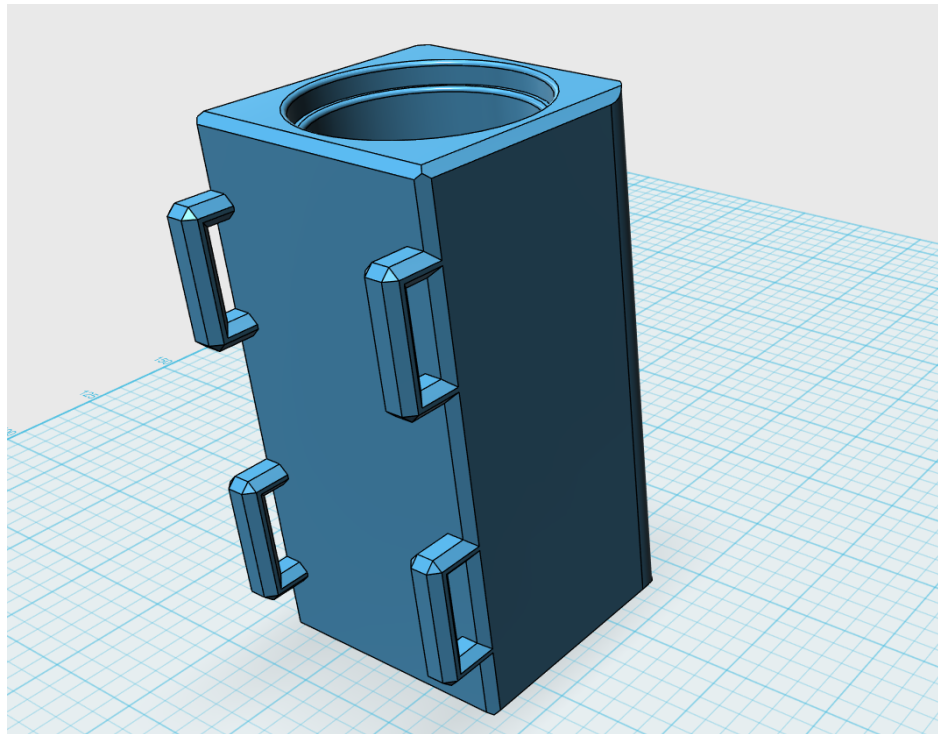
Funzione di ricarica

- Gestita con timer2 impostato ad una frequenza di 2kHz
- Il tempo che si impiega a ricaricare dipende dall'arma e per variare il tempo si fa compiere un numero variabile di cicli al timer2 dato che quest'ultimo è a 8 bit e non si possono ottenere periodi elevati



Struttura esterna

Disegnata con un cad 3D adatto alla creazione di file STL che sono i file richiesti dalle stampanti 3D



Ringraziamenti



Ringrazio inoltre tutti i professori che mi hanno seguito nella realizzazione di questo progetto in particolare i professori Franco Duso e Piergiuseppe Dal Santo

Grazie per l'attenzione