



**Makerspace mobili per promuovere l'educazione dei maker nelle scuole
democratizzare l'istruzione STEAM e lo sviluppo dell'innovazione per tutti gli
studenti**

IO2

Linee guida per gli insegnanti

**Sistema di controllo semaforico con Micro:Bit &
MakeCode**

Livello di difficoltà: medio

Età di riferimento: dai 14 ai 18 anni

Costo: attualmente piuttosto basso

Realizzato da: IIS Leonardo da Vinci

Autori: Renato Gatti & Loris Ranalli



Erasmus+

GRANT NUMBER: 2020-1-PL01-KA201-081698

Dichiarazione

Questo file è stato preparato nell'ambito del progetto STEAM2GO. Laddove sono stati utilizzati altri materiali originali pubblicati e inediti, questi sono stati riconosciuti.

Diritto d'autore

© Copyright 2020 - 2022 il Consorzio [STEAM2GO](#)



Tutti i diritti riservati.



Questo documento è concesso in licenza al pubblico ai sensi di Attribuzione Creative Commons-Non commerciale - Licenza internazionale No Derivatives 4.0.

Esclusione di responsabilità per il finanziamento

Questo progetto è stato finanziato con il sostegno della Commissione Europea. Questo rapporto riflette solo il punto di vista dell'autore e la Commissione non può essere ritenuta responsabile per qualsiasi uso che possa essere fatto delle informazioni in esso contenute.

Linee guida per gli insegnanti

Sistema di controllo semaforico con Micro:Bit

Contenuti

Linee guida per gli insegnanti.....	1
Note introduttive	4
Studio e realizzazione dei diagrammi temporali.....	5
Presentazione della scheda Micro:Bit e del software MakeCode.....	7
Collegamento degli I/O Micro:Bit ai semafori a LED esterni.....	8
La breadboard.....	9
Creiamo i nostri semafori con Micro:Bit	10
Programmazione con Micro:Bit tramite pulsante interno “A”.....	11
Programmare il semaforo con Micro:Bit e i dispositivi di input esterni ..	13
Disegno e realizzazione della planimetria del raccordo a due vie.....	17
Note sui collegamenti di pulsanti e interruttori	21
Note sulle resistenze di pull-up e pull-down.....	22
File di supporto.....	23
Guida Micro:Bit.....	23
Guida MakeCode.....	23

Note introduttive

Informazioni sul progetto

Inizieremo con la programmazione di un prototipo realizzato su singola bread board. Successivamente si potrà realizzare un modello dell'incrocio su carta o cartone che verrà appoggiato su un foglio di polistirolo dove saranno praticati dei fori rettangolari, di adeguate dimensioni, per alloggiare le quattro breadboard da 170 contatti. Queste verranno poi collegate alla breadboard con Micro:Bit, pulsante e interruttore. I vari dispositivi andranno poi collegati ai relativi GPIO della scheda Micro:Bit e, infine, verrà programmato l'intero sistema.

Obiettivi formativi

Gli studenti coinvolti nel progetto avranno la possibilità di:

- Comprendere e implementare diagrammi di temporizzazione
- Comprendere dettagliatamente un circuito elettronico
- Creare circuiti utilizzando Micro:Bit e componenti elettronici
- Creare una planimetria di un impianto realizzando un modello in carta utilizzando materiali di uso comune
- Imparare i fondamenti della programmazione per implementare un modello proposto
- Analizzare e identificare i parametri e le funzioni da controllare in un sistema complesso
- Interfacciare, attraverso i GPIO di Micro:Bit, i segnali di I/O provenienti dai vari dispositivi
- Sviluppare, tramite il software MakeCode, un complesso sistema di controllo
- Testare il prototipo
- Ottimizzare e definire la soluzione di lavoro ottimale condividendola tra i partecipanti

Collegamento ad altre aree tematiche e/o concetti STEM specifici

- Matematica
- Tecnologia
- Elettronica
- Informatica
- Realizzazione di manufatti relativi ad un progetto tecnico-scientifico

Hardware e software di cui potresti aver bisogno

Hardware	Software	Materiale artigianale
- 3-5 x breadboard 170 punti: https://amzn.eu/d/bcFYyKs - 1 scheda Micro:Bit - 1 scheda Micro:Bit BreakOut - 1 pulsante - 1 interruttore SPDT - Jumper wires kit - Wiring cables kit (M/M and M/F) - 2 o 4 x kit singolo semaforo a led preassemblato: https://www.amazon.it/ZHITING-Display-semaforo-Mini-semaforo-elettronici/dp/B0859ZPM47	- Excel - Word - MakeCode	- Alcuni fogli di carta bianca o cartoncino formato A4 (o A3). - Un foglio di polistirolo espanso - Un foglio di polipropilene alveolare - 1 taglierino - 1 forbice - 1 colla - 1 nastro biadesivo - 1 matita - 1 temperamatite - Alcuni pennarelli indelebili di vari colori - Attrezzatura di cancelleria come matite, penne, fogli di carta

Le fasi di lavoro

1. Il docente può scegliere, in base al livello degli studenti, una delle opzioni presentate sia per la parte di programmazione che per la realizzazione del piano dell'intersezione stradale.
2. La fase iniziale sarà l'identificazione del corretto funzionamento di un controllo semaforico. Particolare attenzione dovrà essere prestata alla tempistica dei segnali e alle misure di sicurezza da adottare. Questo può essere fatto attraverso una rapida ricerca su Internet guidata dal docente.
3. Successivamente si dovrà studiare il diagramma temporale dell'impianto, inizialmente nel modo più semplice possibile e successivamente tenendo conto di un tempo di sovrapposizione del semaforo rosso, al fine di garantire che l'incrocio sia libero da eventuali veicoli.
4. Configurazione della scheda Micro:Bit
5. Creare il primo programma operativo utilizzando MakeCode su un PC e testarlo
6. Effettuare tutte le interconnessioni necessarie tra i GPIO della scheda Micro:Bit e i vari componenti esterni
7. Trasferisci il programma dal PC alla scheda Micro:Bit
8. Progettare e realizzare una planimetria dell'intersezione stradale dove verranno posizionate le breadboard e le varie componenti
9. Testare il corretto funzionamento del sistema

Studio e realizzazione dei diagrammi temporali

Il primo passo per realizzare il progetto è analizzare e creare un diagramma con le tempistiche dei vari segnali di ogni semaforo. Per una corretta comprensione di ogni passaggio, è necessario partire dal diagramma temporale più semplice. Una volta completato, i pro ei contro vengono portati all'attenzione degli studenti.

I diagrammi temporali possono essere realizzati su carta a quadretti oppure utilizzando un foglio di calcolo o un qualsiasi programma di grafica.

a) Diagramma di temporizzazione più semplice senza sovrapposizione di luci rosse

La sequenza di ogni semaforo sarà, nell'esempio riportato in Fig.1:

- Verde: 4 sec.
- Giallo: 1 sec.
- Rosso: 5 sec. (il tempo del segnale rosso deve essere uguale alla somma del verde e del giallo del semaforo opposto)

In questo caso si assume che si accendano contemporaneamente il semaforo verde del semaforo [1], che chiameremo G1, e il semaforo rosso del semaforo opposto [2], che chiameremo R2. Trascorso invece il tempo programmato, si accendono contemporaneamente la luce verde del semaforo [2], che chiameremo G2, e la luce rossa del semaforo opposto [1], che chiameremo R1. Fare riferimento alla **Figura 1**

b) Diagramma di temporizzazione con sovrapposizione di luci rosse

In questo caso abbiamo considerato la necessità di concedere un po' di tempo alla sovrapposizione delle due luci rosse prima di attivare il verde. Questo per essere sicuri che, una volta acceso il verde, non ci siano più veicoli nell'incrocio.

Questo è il principale criterio di sicurezza, richiesto dalle norme stradali e dal codice della strada.

Ora la sequenza di ogni semaforo sarà, nell'esempio riportato in Fig.2:

- Verde: 4 sec.
- Giallo: 1 sec.
- Rosso: 7 sec. (il tempo del segnale rosso deve essere uguale alla somma del verde e del giallo del semaforo opposto più il tempo di sovrapposizione che abbiamo assunto in 1 sec per lato). Fare riferimento alla **Figura 2**

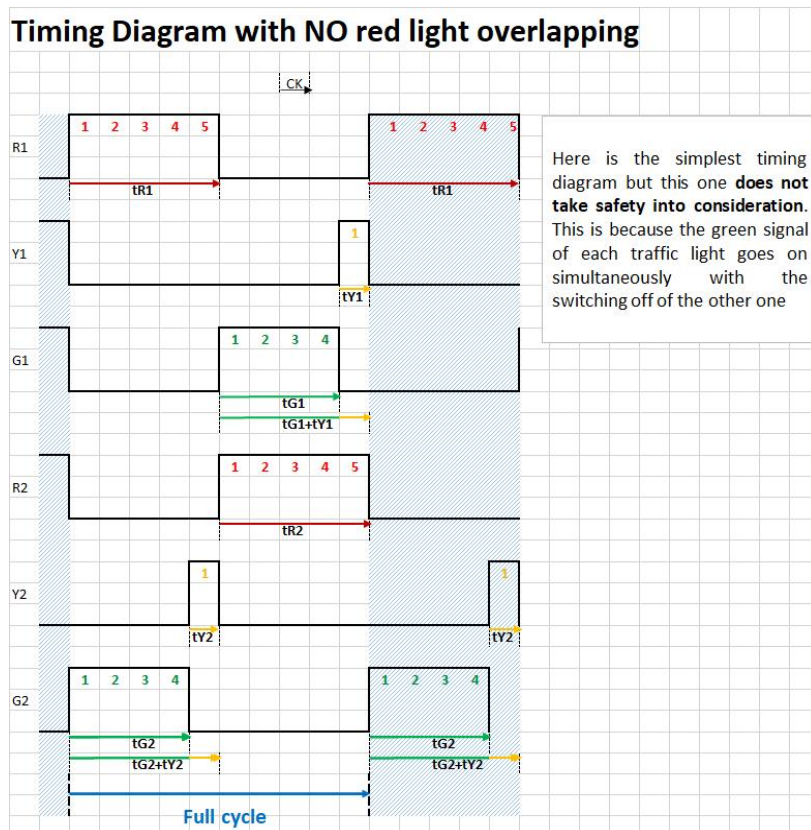


Figura 1

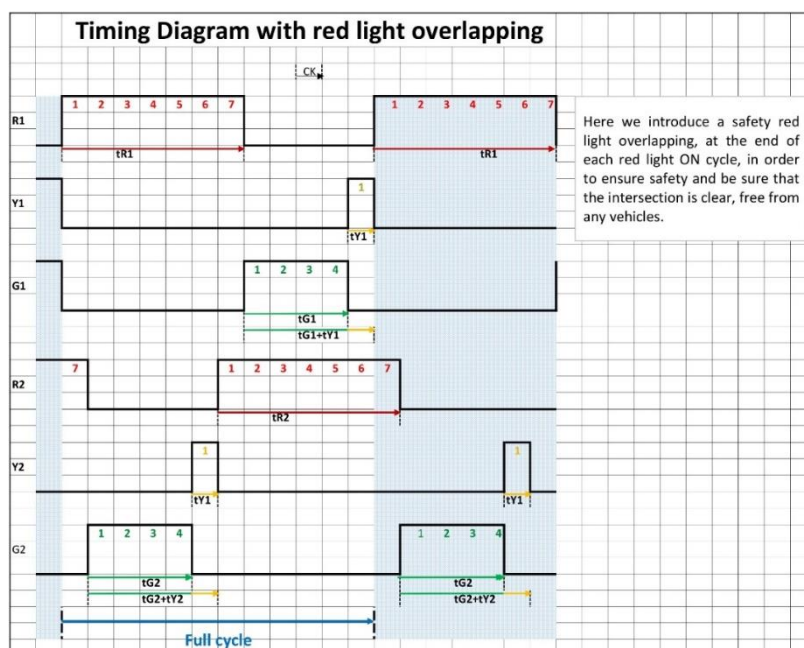


Figura 2

L'insegnante può intervenire sollevando questioni di sicurezza, ad esempio:

- È necessaria una sovrapposizione di luci rosse? Se è così, perché?
- Per quanto tempo le luci rosse dovrebbero sovrapporsi se necessario?
- In base a quali parametri calcolereste il tempo di sovrapposizione dei semafori rossi?
- Nel caso di un semaforo pedonale, come regolereste i vari orari del semaforo?
- Ecc.

Cliccando sui link sottostanti è possibile scaricare i Diagrammi di Temporizzazione in vari formati:

Link ai Diagrammi temporali realizzati in formato MS Excel:

- [Diagramma di temporizzazione più semplice senza sovrapposizione di luci rosse](#)
- [Diagramma di temporizzazione con sovrapposizione di luci rosse](#)

Link ai diagrammi temporali in formato PDF:

- [Diagramma di temporizzazione più semplice senza sovrapposizione di luci rosse](#)
- [Diagramma di temporizzazione con sovrapposizione di luci rosse](#)

Link ai diagrammi temporali in formato JPG:

- [Diagramma di temporizzazione più semplice senza sovrapposizione di luci rosse](#)
- [Diagramma di temporizzazione con sovrapposizione di luci rosse](#)

Presentazione della scheda Micro:Bit e del software MakeCode

Micro:bit è un computer portatile, programmabile dall'utente che contiene:

- 25 LED, programmabili individualmente
- 2 pulsanti programmabili etichettati A e B
- Sensori di temperatura e luce
- Sensori di movimento (accelerometro e bussola)
- Comunicazione wireless via radio e Bluetooth
- connettore con alcune interfacce GPIO per il collegamento di dispositivi esterni (tramite una scheda breakout)

Qui la documentazione di Micro:Bit: <https://microbit.org/get-started/first-steps/introduction/>
<https://microbit.org/get-started/user-guide/overview/>

È possibile programmare la scheda Micro:bit attraverso un'applicazione online, MakeCode, fornita da Microsoft, che troviamo qui:

<https://makecode.microbit.org> oppure si può scaricare un'applicazione offline qui:

<https://makecode.microbit.org/offline-app>

La offline app funziona allo stesso modo dell'applicazione Web nel browser, ma è un'applicazione autonoma, che funzionerà anche quando la connessione a Internet sia limitata o non disponibile.

Il layout della scheda e il pinout della scheda Micro:Bit sono mostrati di seguito **Figure 3-4**

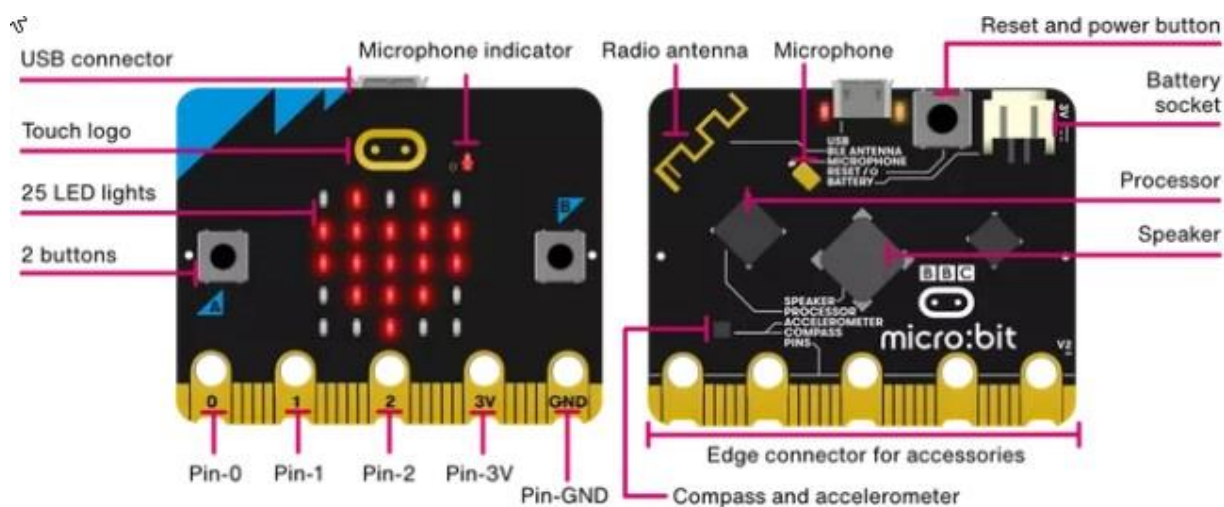
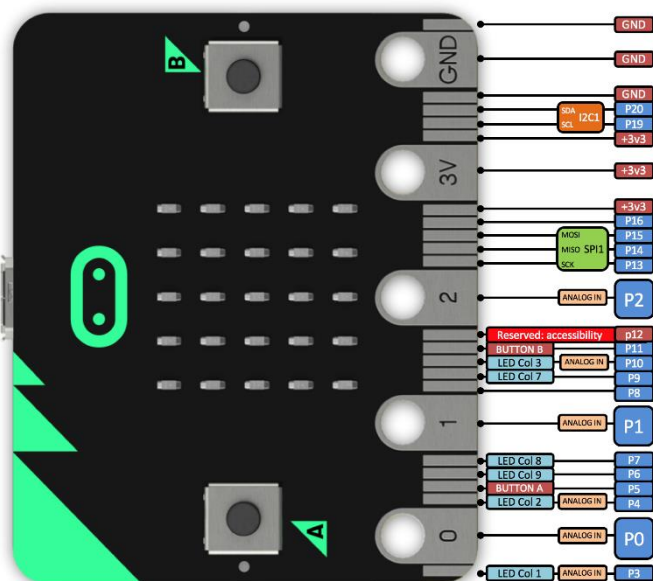


Figura 3

Figura 4



Collegamento degli I/O Micro:Bit ai semafori a LED esterni

Il passaggio successivo consiste nel collegare gli I/O Micro:Bit alle breadboard. Per fare ciò è necessario utilizzare la scheda breakout per Micro:Bit e i cavi jumper, inclusi nel Mobile Maker Space.

Gli I/O della scheda breakout Micro:Bit sono mostrati nelle seguenti **Figure 5-6-7**

Figura 5: come inserire correttamente la scheda Micro:Bit nella scheda BreakOut

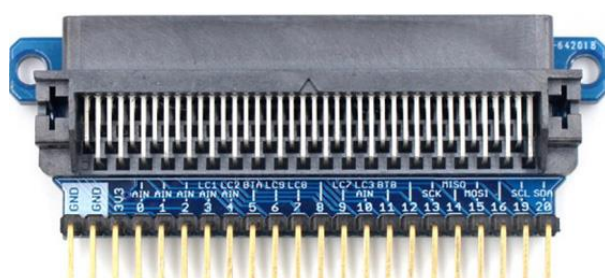
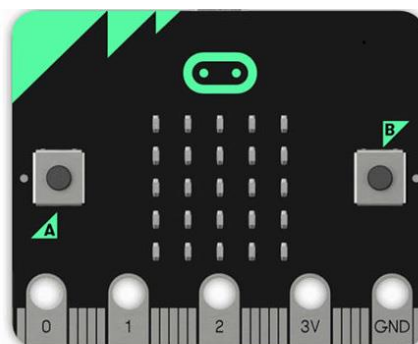


Figura 6

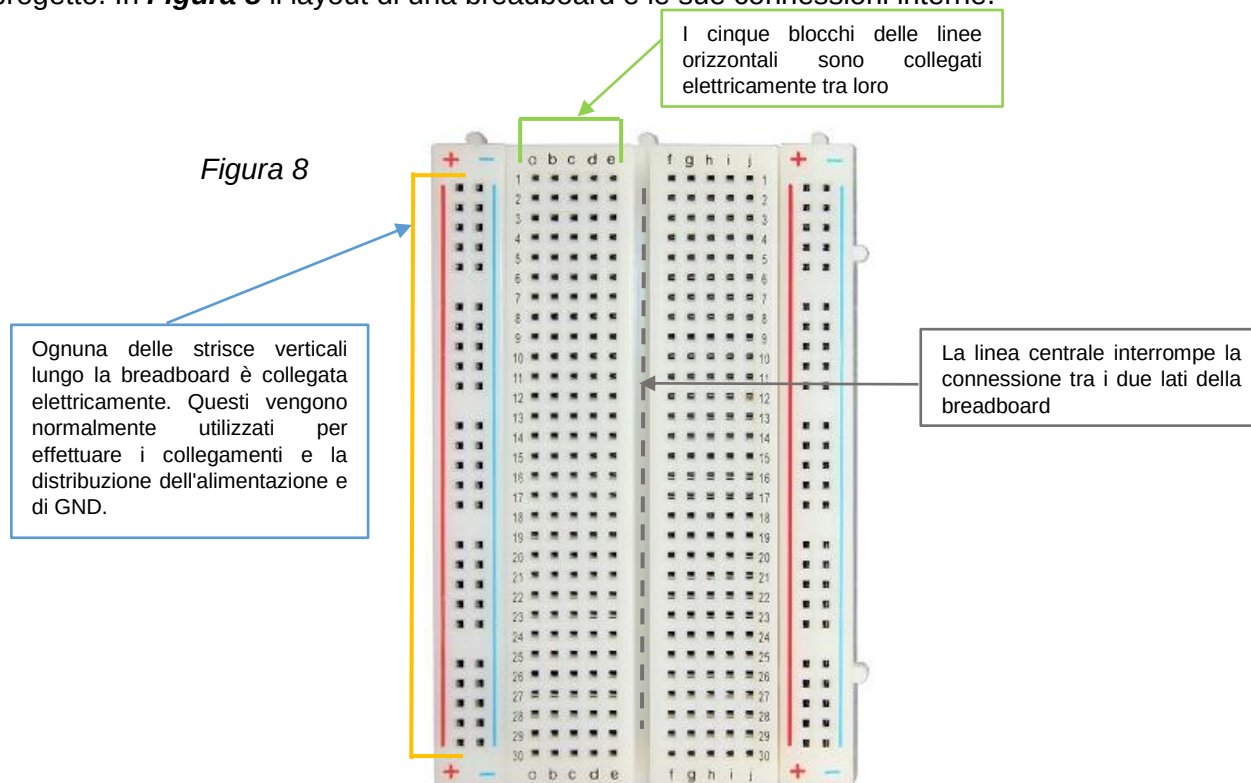


Figura 7

La breadboard

Breadboard è un modo per realizzare dispositivi elettronici senza dover utilizzare un saldatore. I componenti vengono inseriti nelle prese (fori) presenti sulla breadboard e quindi vengono collegati utilizzando fili "jumper" esterni per effettuare le connessioni tra i vari componenti.

Le breadboard sono fatte per effettuare prototipi o esperimenti veloci. Non sono adatte per mantenere in funzione i circuiti per molto tempo. Le connessioni, non essendo permanenti, sono facilmente removibili nel caso si commetta un errore o semplicemente iniziare da capo un nuovo progetto. In **Figura 8** il layout di una breadboard e le sue connessioni interne.



In **Figura 9** un esempio di come collegare un LED con una batteria e un pulsante

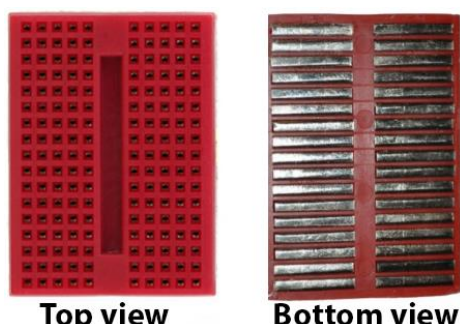
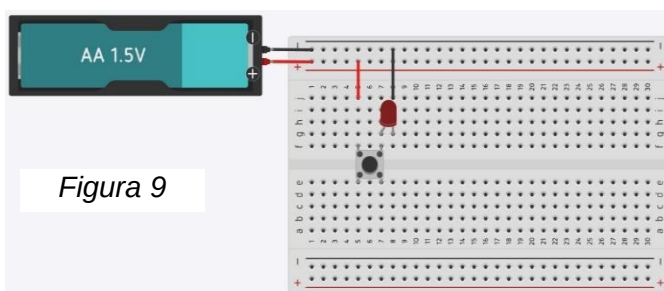


Figura 10

In **Figura 10** un altro tipo di breadboard, più semplice e più piccolo, che non ha le strisce verticali per i collegamenti e la distribuzione dell'alimentazione e GND

Per il corretto montaggio del pulsante e dell'interruttore è necessario verificare il funzionamento dei pin utilizzando il tester fornito con il Mobile Maker Space. Vedere *) Note su pulsanti e interruttori alla fine di questo documento

Convien realizzare un foglio di calcolo con le interconnessioni tra gli I/O Micro:Bit e i semafori sulle breadboard. Gli esempi sono in **Figura 11 - 12**

Micro:bit		Led Traffic Lights	
PIN	Traffic Light n.	Colour	Pin
16	1	Green 1	G
15	1	Yellow 1	Y
14	1	Red 1	R
GND	1	---	GND
13	2	Green 2	G
9	2	Yellow 2	Y
8	2	Red 2	R
14	2	---	GND

Figura 11

Micro:bit	Input device
PIN	Type
0	Start Push-button
2	Maintenance switch
3V3	+Vcc

Figura 12

Cliccando sui link sottostanti è possibile scaricare le tabelle di collegamento in vari formati:

Link alla Tabella dei collegamenti in formato MS Excel:

- [Tabella di connessione Semaforo 2 luci.xlsx](#)
- [Tabella di connessione Semaforo 4 luci - 2 x strada.xlsx](#)

Link alla Tabella dei collegamenti in formato PDF:

- [Tabella di connessione Semaforo 2 luci.pdf](#)
- [Tabella di connessione Semaforo 4 luci - 2 x strada.pdf](#)

Link alla Tabella dei collegamenti in formato JPG:

- [Tabella di connessione Semaforo 2 luci.jpg](#)
- [Tabella di connessione Semaforo 4 luci - 2 x strada.jpg](#)

Creiamo i nostri semafori con Micro:Bit

Il sistema può essere implementato in due modi diversi:

- il più veloce è costruire un prototipo con un'unica breadboard su cui assemblare tutti i componenti. Questo sistema non prevede la realizzazione dei disegni di tracciato stradale e la realizzazione della piattaforma su cui posizionare i vari componenti (**Figura 13**)
- realizzando del disegno di tracciato stradale e della piattaforma su cui posizionare i vari componenti. Questo sistema richiede più tempo perché comporta anche una parte di realizzazione grafica e la costruzione del pannello su cui inserire i vari elementi (**Figura 14 e 15**)

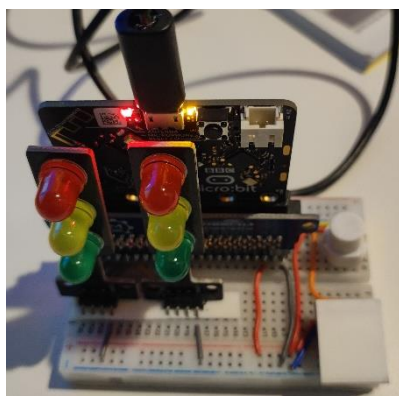


Figura 13

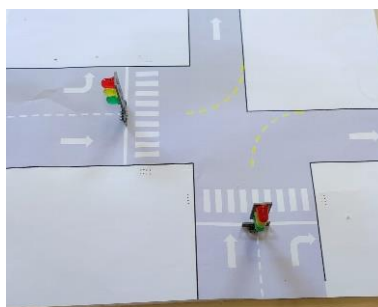


Figura 14

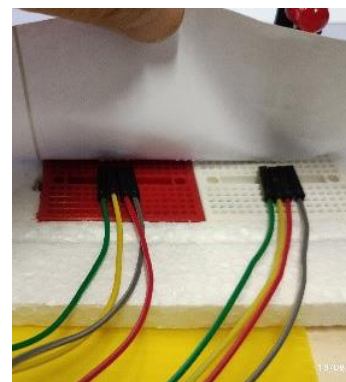


Figura 15

Adesso possiamo effettuare tutti i collegamenti tra la scheda Micro:Bit e la/le breadboard seguendo il pinout indicato in **Figura 11**.

Una volta completate le connessioni possiamo iniziare a programmare la nostra scheda Micro:Bit.

Abbiamo sviluppato due coppie di progetti in ordine progressivo di difficoltà:

- i progetti 1) e 2), i più semplici, **non** utilizzano il pulsante e l'interruttore esterni ma verrà usato il pulsante (denominato A sul retro della scheda) integrato nella scheda Micro:Bit per comandare lo "Start" del sistema
- i progetti 3) e 4), un po' più complessi, utilizzano il pulsante esterno e l'interruttore per comandare le funzioni "Start" e "Manutenzione"

Programmazione con Micro:Bit tramite pulsante interno “A”.

IMPORTANTE: i tempi in MakeCode sono sempre espressi in microsecondi.

Di seguito vengono mostrati i primi due progetti che non utilizzano pulsanti esterni:

- 1) TL-NO-overlap-NO-buttons
- 2) TL-overlap-NO-buttons

1) TL-NO-overlap-NO-buttons

Prestare attenzione alle seguenti indicazioni:

- all'inizio di ogni sequenza è importante impostare tutte le uscite GPIO a "0", in modo che tutti i LED siano spenti all'avvio del programma
- ogni volta che si imposta un GPIO a "1" e si programma il tempo in cui quel led rimarrà acceso, dovremo anche spegnerlo riportandolo a "0" prima di accendere il successivo e viceversa.

Alla pressione del pulsante "A" inizia la sequenza facendo lampeggiare entrambe le luci gialle per 2 secondi.

Quindi il programma entra nel ciclo di lavoro infinito.

In questo programma, i tempi, usati come esempio, sono i seguenti:

ROSSO = 7 secondi

GIALLO = 2 secondi

VERDE = 5 secondi

Qui puoi scaricare il programma MakeCode:

[Programma MakeCode: TL-NO-overlap-NO-buttons](#)

La **Figura 16** mostra il codice del programma TL-NO-overlap-NO-buttons.

La **Figura 17** mostra il codice del programma TL-overlap-NO-buttons.

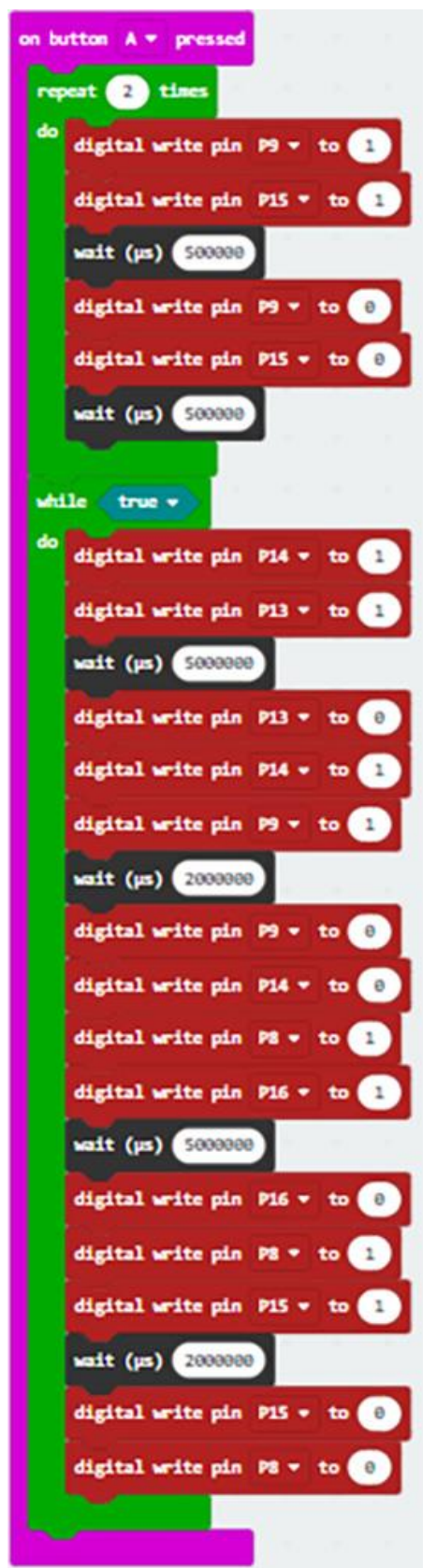


Figura 16

Figura 17

1)

2)

3)

4)

5)

6)

7)



2. TL-overlap-NO-buttons

Il programma è simile a quello già fatto (TL-NO-overlap-NO-buttons).

Quando si preme il pulsante "A" inizia la sequenza, facendo lampeggiare due volte entrambe le luci gialle per 0,5 secondi. La differenza è che i tempi dovranno tenere conto della sovrapposizione delle luci rosse. Tab. 1 spiega le funzioni presenti nel programma in **Figura 17**.

Nel nostro caso abbiamo impostato:

1)ROSSO_1 (P14) e VERDE_2 (P13) = "ON" per 5 secondi	Tab. 1
Quindi	
2)VERDE_2 (P13) = "OFF" e {ROSSO_1 (P14) e GIALLO_2 (P9)} "ON" per 2 secondi	
Quindi	
3)GIALLO_2 (P9) = "OFF" e ROSSO_2 (P8) = "ON" per 2 secondi <i>(qui le luci rosse per 2 secondi si sovrappongono)</i>	
Quindi	
4)ROSSO_1 (P14) = "OFF" e {ROSSO_2 (P8) e VERDE_1 (P16)} = "ON" per 5 secondi	
Quindi	
5)VERDE_1 (P16) = "OFF" e {ROSSO_2 (P8) e GIALLO_1 (P15)} = "ON" per 2 secondi	
Quindi	
6)GIALLO_1 (P15) = "OFF" e ROSSO_1 (P14) = "ON" per 2 secondi <i>(qui le luci rosse per 2 secondi si sovrappongono)</i>	
Quindi	
7)ROSSO_1 (P14) e ROSSO_2 (P8) = "OFF"	
Quindi	
Il ciclo riparte dal punto 1)	

Qui si può scaricare il programma MakeCode:

[Programma MakeCode: TL-overlap-NO-buttons](#)

Programmare il semaforo con Micro:Bit e i dispositivi di input esterni

Di seguito vengono mostrati due progetti che utilizzano pulsanti esterni per attivare le funzioni "Start" e "Maintenance":

- 3) TL-NO-overlap-buttons
- 4) TL-overlap- buttons

Qui utilizziamo due dispositivi di input esterni, un pulsante (Start) per Avviare la sequenza e un interruttore (Maintenance) per entrare in modalità "Manutenzione". In modalità "Manutenzione" facciamo lampeggiare solo le luci gialle ogni 0,20 secondi (200.000 microsecondi).

È necessario creare le tre funzioni:

- 1) funzione principale "forever" che parte non appena accendiamo il dispositivo (**Figura 18**)

2) funzione "maintenance" che avvia la sequenza di manutenzione (le luci gialle lampeggiano alternativamente ogni 200 millisecondi) **Figura 19**

3) funzione "on" che avvia la normale sequenza operativa **Figura 20**

Le prime due funzioni, 1) "forever" e 2) "maintenance", rimarranno invariate sia che il programma venga realizzato con o senza sovrapposizione delle luci rosse.

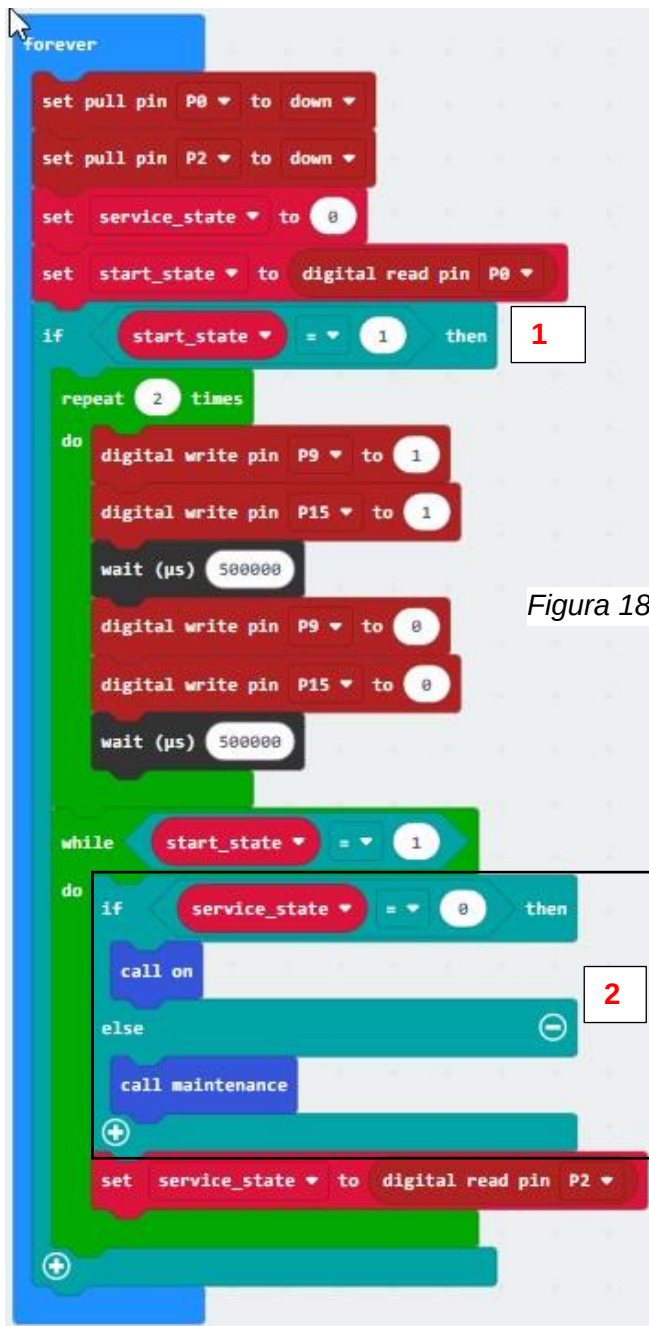


Figura 18

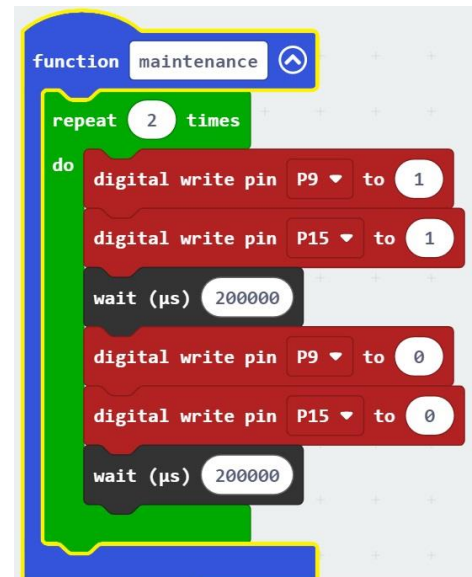


Figura 19

If sequence:

Qui viene letto il valore del PIN P2 (interruttore "maintenance" e viene attribuito alla variabile "service_state".

Quando la macchina viene alimentata, i pin digitali P0 (Pulsante di avvio) e P2 (Interruttore di manutenzione) vengono messi in modalità pull-down (**Figura 18**)

Quindi vengono impostate le due variabili:

- service_state: è inizializzato a 0 (OFF)

- **start_state**: a questa variabile viene assegnato il valore letto dal **digital_pin_1**, che rappresenta lo stato del P0 - Pulsante "Start" (nota **1** in Figura 18).

Quindi se si preme il P0 (Pulsante di Start) la sequenza avrà inizio, altrimenti rimane in attesa. Una volta avviata la sequenza, le spie gialle lampeggiano due volte per 0,5 secondi ("repeat/do") e poi si entra nella condizione "while/do". Qui troviamo il blocco logico "if/else" che controlla il valore della variabile "service_state", che dipende dallo stato "Maintenance_switch": (nota **2** in Figura 18).

- se "Maintenance_switch" è OFF (0) viene richiamata la funzione "on" e il sistema entra in funzionamento normale
- else "Maintenance_switch" è ON (1) viene richiamata la funzione "maintenance" e il sistema entra in modalità manutenzione, facendo lampeggiare "n" volte le spie gialle (nel nostro caso 2 volte) prima di ricontrollare la funzione else "Maintenance_switch"
- una volta che l'interruttore di manutenzione viene nuovamente spento, la sequenza di controllo riporta il programma alla funzione "on".

La terza funzione 3) "on" per programma TL-NO-overlap-buttons



Figura 20

Nella **Figura 20** è illustrata la funzione "on" senza sovrapposizione delle luci rosse.

È esattamente lo stesso che abbiamo visto nel primo esercizio TL-NO-overlap-NO-buttons. È la parte contenuta nel ciclo "while true" "do" nella Figura 16

[Clicca qui per vedere il video del prototipo funzionante](#)

Qui è possibile scaricare i programmi MakeCode che utilizzano i pulsanti esterni:

[Programma MakeCode: TL-NO-overlap-buttons](#)

[Programma MakeCode: TL-overlap- buttons](#)

4. Funzione "on" TL-overlap-buttons (vedi Foglio di lavoro punto 3-2 pag. 5)

Sulla base di quanto fin qui esposto, lo sviluppo di questa funzione potrebbe essere un valido esercizio da proporre agli studenti, come indicato nella Foglio di lavoro punto 3-2 pag. 5.

Le due funzioni 1) “forever” (**Figura 18**) e 2) “maintenance” (**Figura 19**), rimangono invariate sia che il programma sia realizzato con o senza sovrapposizione dei semafori rossi. Fare riferimento alle funzioni già spiegate a pagina 14.

Questa funzione è esattamente la stessa che abbiamo visto nel primo esercizio TL-overlap-NO-buttons. È la parte contenuta nel ciclo “while true” “do” nella **Figura 17**.

La funzione “on” del programma TL-overlap-buttons è mostrata nella **Figura 21**.



Figura 21 funzione “on” del programma TL-overlap-buttons

Terminata la programmazione si può passare alla fase di test, utilizzando il prototipo con un'unica breadboard (**Figura 13**) oppure realizzare i disegni del tracciato stradale e la costruzione della piattaforma (**Figure 14 - 15**) come indicato di seguito.

Disegno e realizzazione della planimetria del raccordo a due vie

La planimetria dell'incrocio semaforico può essere realizzata con un foglio di carta delle dimensioni, come nel nostro caso, 28x34 cm., oppure utilizzando un semplice foglio A4, come indicato nelle **Figure 22 - 23 - 24**

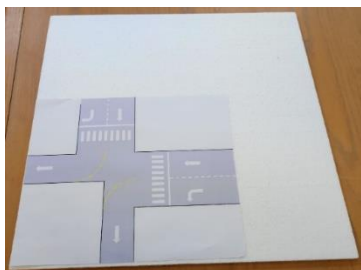


Figura 22

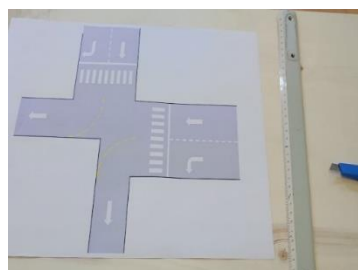


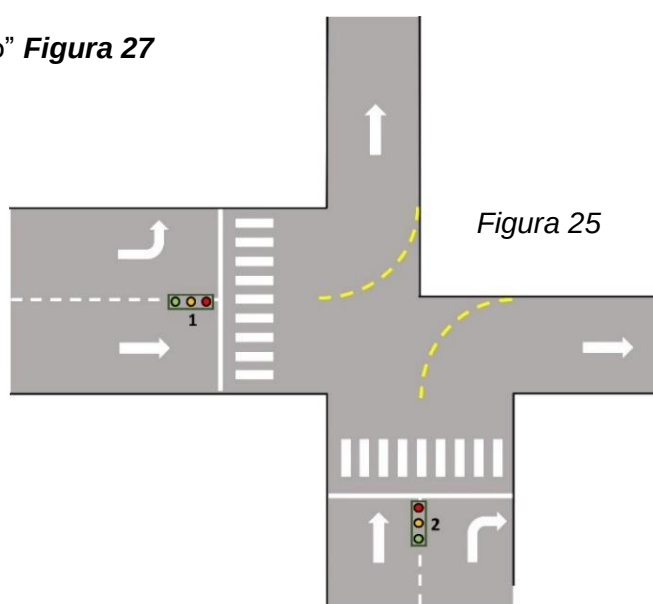
Figura 23



Figura 24

Il progetto può essere realizzato a mano con pennarelli o matite colorate oppure utilizzando un programma di disegno e stampandolo; nel nostro caso abbiamo utilizzato MS Word. Di seguito sono mostrati tre diversi layout:

- “Draft Layout_2 luci” **Figura 25**
- “Draft Layout_4 luci” **Figura 26**
- “Layout a semaforo per il montaggio” **Figura 27**



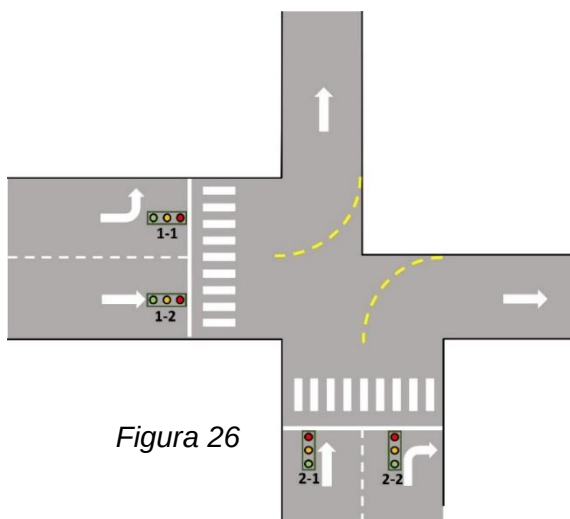


Figura 26

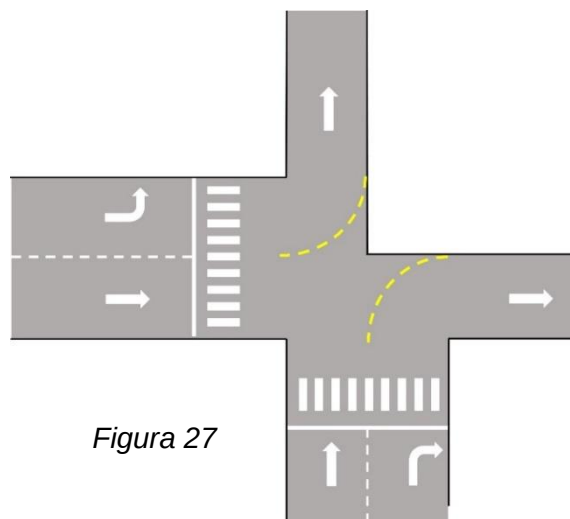


Figura 27

Cliccando sui link sottostanti è possibile scaricare il Layout in vari formati:

Link al Layout realizzato con MS Word:

- [Bozza Layout 2 luci](#)
- [Bozza Layout 4 luci](#)
- [Disposizione del semaforo per il montaggio](#)

Link al Layout realizzato in formato PDF:

- [Bozza Layout 2 luci](#)
- [Bozza Layout 4 luci](#)
- [Disposizione del semaforo per il montaggio](#)

Link al Layout realizzato in formato JPG:

- [Bozza Layout 2 luci](#)
- [Bozza Layout 4 luci](#)
- [Disposizione del semaforo per il montaggio](#)

Dopo aver creato il layout cartaceo sarà necessario realizzare una base in polistirolo alta 1 cm dove andranno inserite le breadboard. Il foglio di polistirolo avrà le dimensioni del foglio di carta A4 usato per creare il layout grafico o leggermente più grande (**Figura 28 - 29**)



Figura 28



Figura 29

Il layout e i fogli di polistirolo tagliati a misura sono mostrati in **Figura 30**

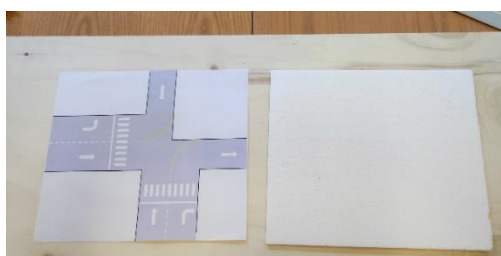


Figura 30

Il prossimo passo sarà segnare la posizione esatta delle breadboard su un disegno dell'intersezione utilizzando un layout creato in precedenza. **Figura 31 - 32**

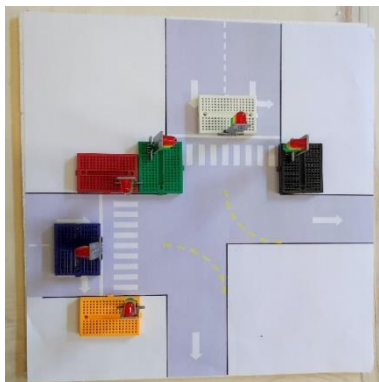


Figura 31

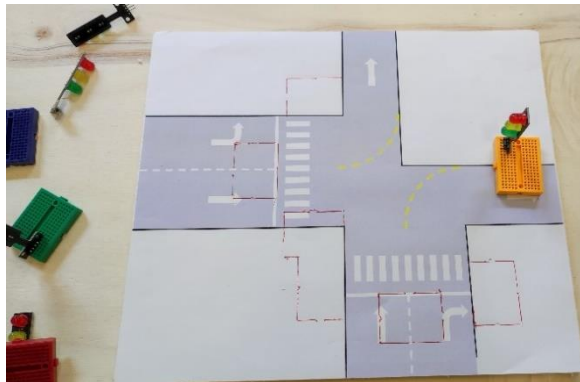


Figura 32

Una volta tracciati sul disegno i contorni esatti delle breadboard, dovremo ritagliare i contorni utilizzando un taglierino **Figura 33**

Fatto ciò, dobbiamo tracciare i contorni sulla lastra di polistirolo usando il disegno che abbiamo appena modellato come riferimento e guida. **Figura 34**

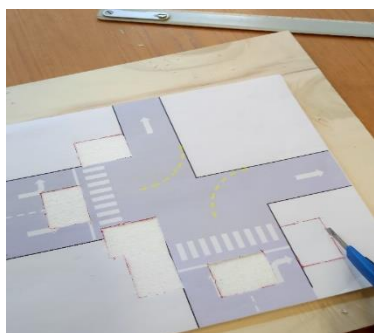


Figura 33

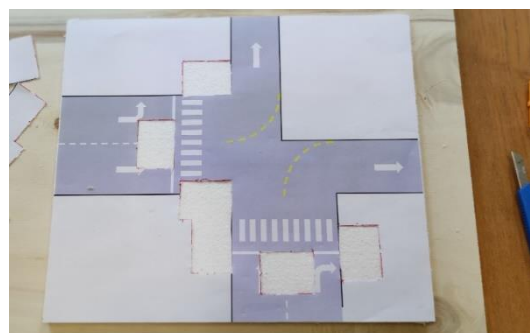


Figura 34

Ora possiamo ritagliare le fessure della breadboard nel foglio di polistirolo. **Figura 35 - 36 - 37**

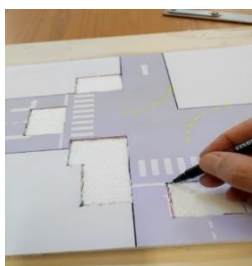


Figura 35

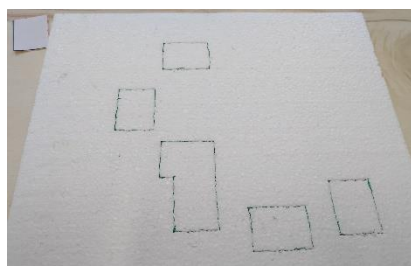


Figura 36

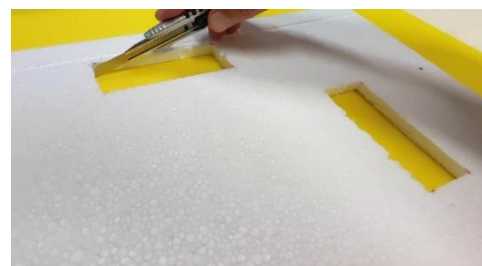


Figura 37

Sarà necessario prestare un po' di attenzione nel sagomare perfettamente le fessure nella lastra di polistirolo, in modo che le breadboard vi si incastrino perfettamente.

Una volta ritagliate le fessure nei fogli di polistirolo, si dovranno inserire le breadboard, ognuna nel proprio alloggiamento, come mostrato in **Figura 38**.

Il risultato di questo passaggio intermedio è mostrato in **Figura 39**.

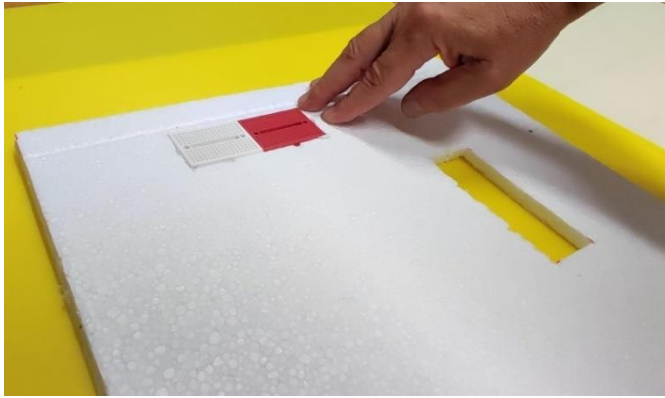


Figura 38

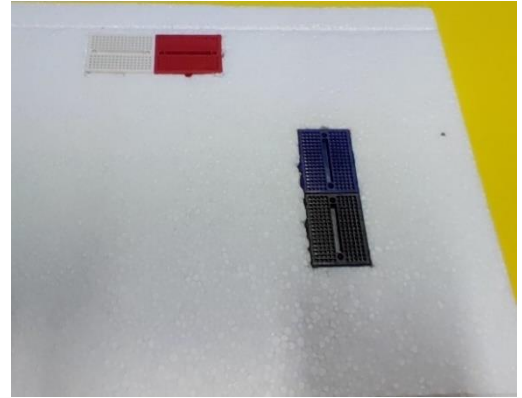


Figura 39

Possiamo ora appoggiare il layout cartaceo precedentemente realizzato sulla base di polistirolo e procedere all'inserimento dei mini semafori a led sulle relative breadboard, seguendo i passaggi mostrati nelle **Figure 40 - 41**

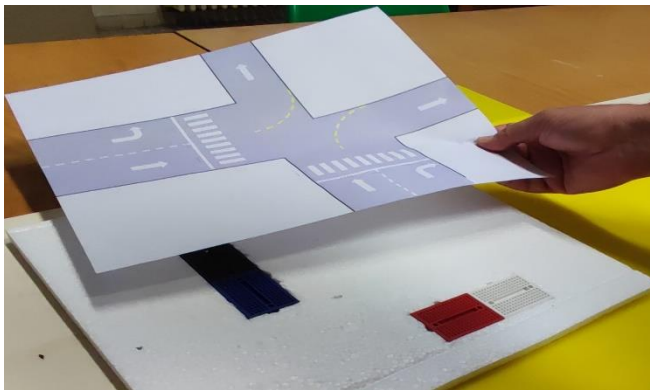


Figura 40

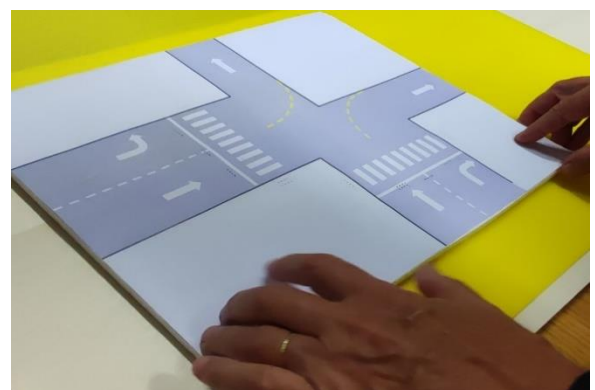


Figura 41

Sotto il layout con i due semafori a LED inseriti nella breadboard **Figura 42**

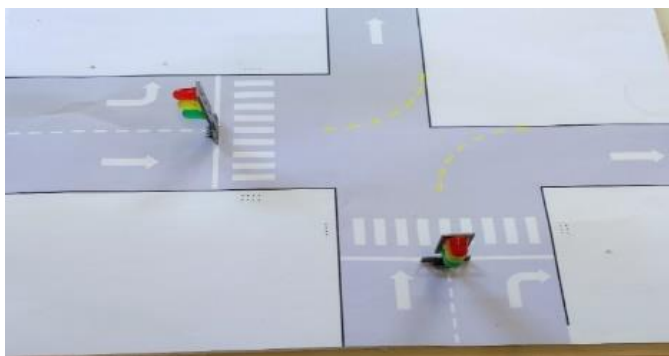


Figura 42

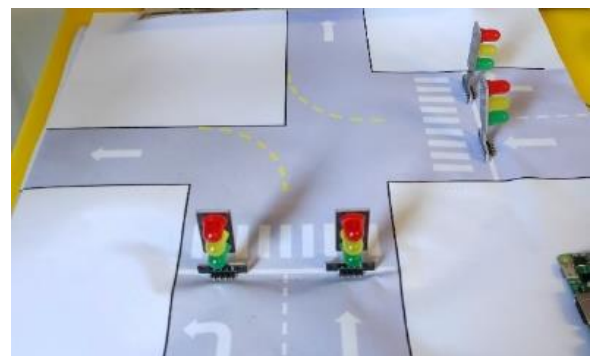


Figura 43

La **Figura 43** mostra un altro esempio dello stesso incrocio ma con due semafori per strada.

Ora possiamo collegare i dispositivi LED agli I/O della scheda Arduino.

È necessario piegare di 90° i piedini dei ponticelli da inserire nella breadboard, in modo da poterli ricoprire con il disegno planimetrico. Vedi Figure **44 - 45 - 46**



Figura 44



Figura 45

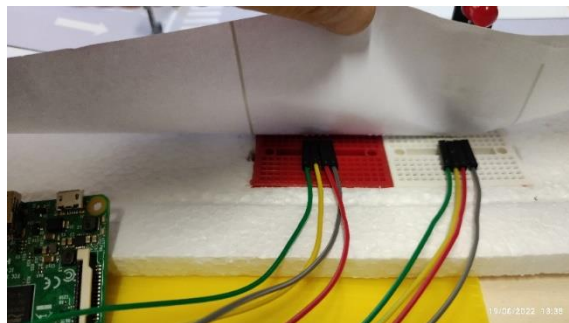


Figura 46

In **Figura 47** un esempio del sistema completo finale.

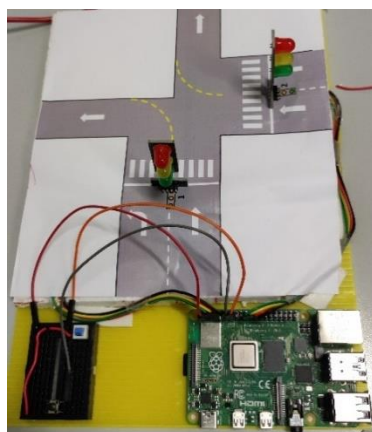


Figura 47

Note sui collegamenti di pulsanti e interruttori

Il PinOut del pulsante utilizzato nel prototipo è mostrato in **Figura 48**.



Figura 48

Si noti che i pin 1-3 (blu nella figura) sono collegati tra loro, così come i pin 2-4 (rossi nella figura). La connessione tra i pin 1-2 è NORMALMENTE APERTA e si chiuderà solo alla pressione del pulsante. Lo stesso vale per i pin 3-4.

Il PinOut dell'interruttore utilizzato nel prototipo è mostrato in **Figura 49**:

Si usino le barre verticali su entrambi i lati come riferimento (su un lato ha solo una barra, l'altro lato ha due barre)

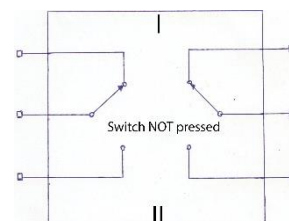


Figura 49

Note sulle resistenze di pull-up e pull-down

I resistori di pull-up (o pull-down) sono resistori a valore fisso, tra l'alimentazione (o massa) e un particolare pin in un circuito logico digitale. Comunemente abbinato agli interruttori, il suo scopo è garantire che la tensione tra massa (GND) e Vcc sia controllata attivamente quando l'interruttore è aperto. Senza il resistore di pull-up lo stato logico dell'interruttore aperto è flottante, quindi potrebbe essere uno '0' o un '1', perché non essendo connesso a un potenziale stabile, si comporta come un'antenna e la tensione ai suoi capi è fluttuante, indeterminata.

Di seguito alcuni esempi di come collegare la resistenza di pull-up (o pull-down).

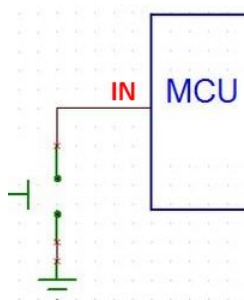


Figure 50
Senza
resistore

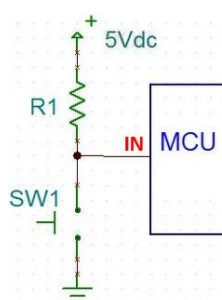


Figure 51
Resistore Pull-Up
Switch non premuto

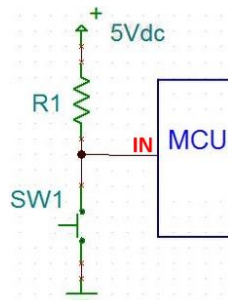


Figure 52
Resistore Pull-Up
Switch premuto

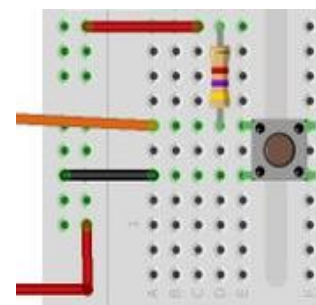


Figure 53
Resistore Pull-Up
sulla breadboard

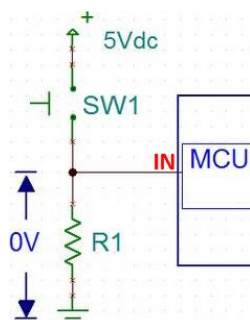


Figure 54
Resistore Pull-Down
Switch non premuto

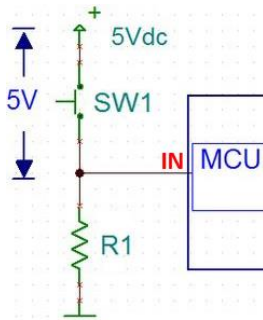


Figure 55
Resistore Pull-Down
Switch premuto

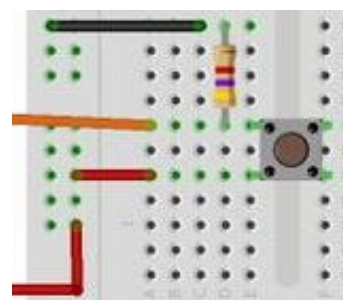


Figure 56
Pull-Down resistor
on the breadboard

Quale dovrebbe essere il valore della resistenza?

Supponiamo di voler limitare la corrente a 1mA. Ad esempio se $V_{cc} = 5V$, utilizzando la legge di Ohm: $R = V/I$ otteniamo $R = 5V/1mA$ (oppure $R = 5000mV/1mA$) quindi $R = 5000\Omega = 5k$. In questo caso possiamo utilizzare una resistenza da 4.7k Ω . Normalmente è possibile utilizzare, in generale, un resistore da 10k Ω , questo richiederà solo 0,5mA

Normalmente in tutti i sistemi a microcontrollore le resistenze di pull-up e pull-down sono già integrate all'interno della scheda stessa. I pin di ingresso possono essere programmati per essere configurati come ingressi pull-up o pull-down con semplici istruzioni.

Si veda la Guida Micro:Bit per ulteriori informazioni.



Esempio di Pull-down:

File di supporto

Foglio di lavoro per gli studenti

[Il foglio di lavoro per lo studente](#)

Programmi

[La cartella con tutti i programmi](#)

Figure

[La cartella con tutte le figure](#)

Video

[La cartella con tutti i video](#)

Guida Micro:Bit

Iniziare con Micro:Bit:

<https://microbit.org/get-started/first-steps/introduction/>

Documentazione Micro:Bit:

<https://microbit.org/get-started/user-guide/overview/>

Guida MakeCode

<https://makecode.microbit.org>

<https://makecode.microbit.org/offline-app>