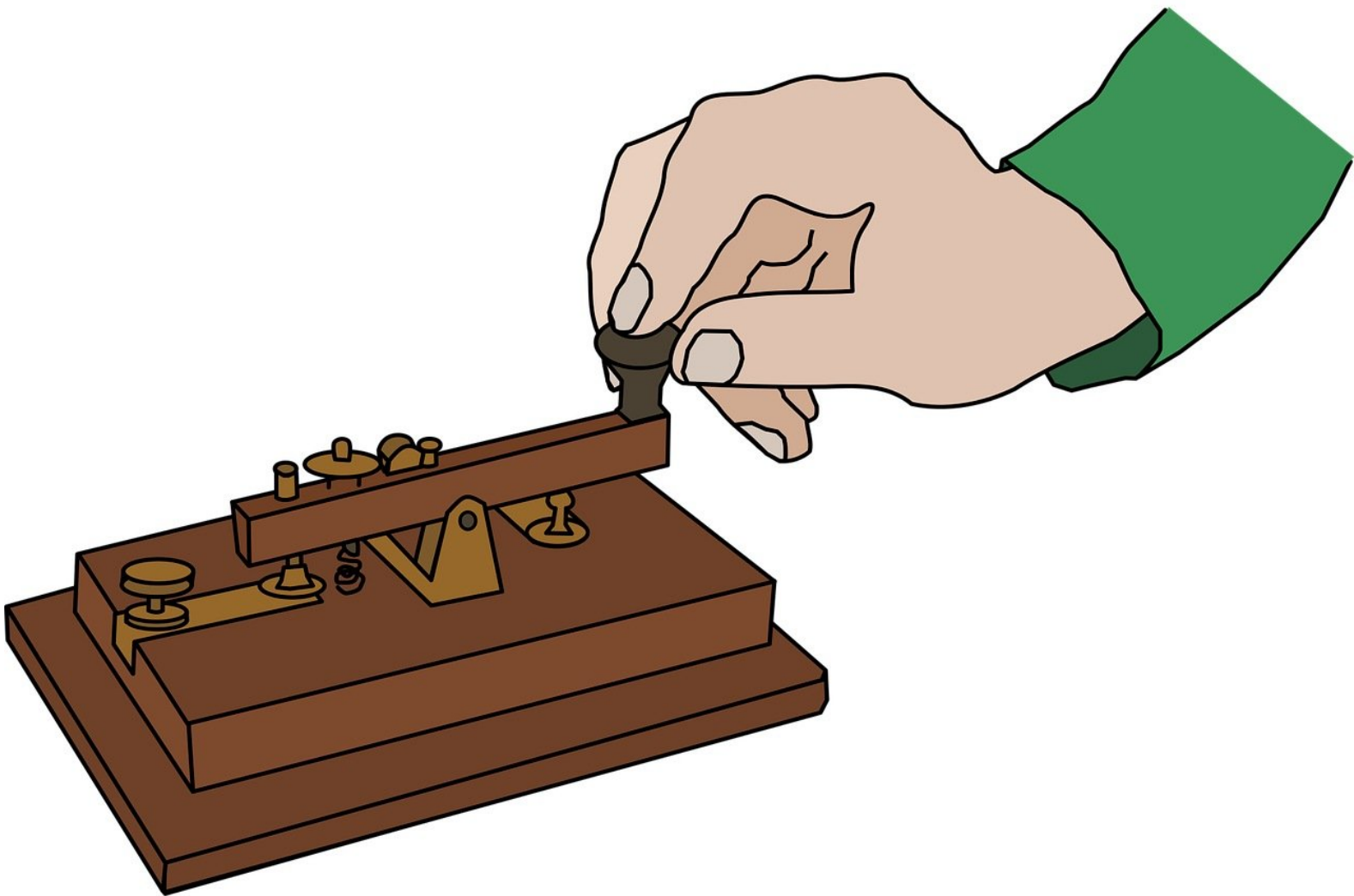


C - Morse Code Machine

Using a switch input and an if statement, make your own Morse Code machine to transmit any letter you like!



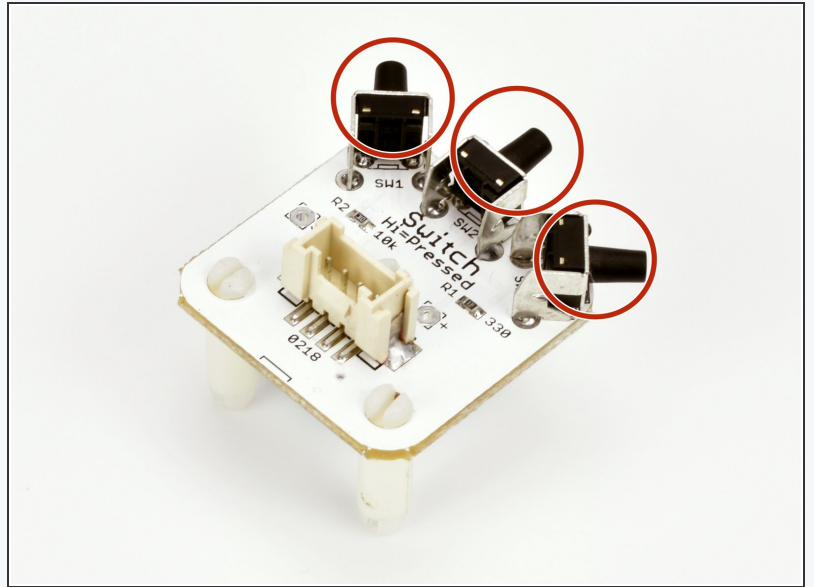
INTRODUCTION

Using a switch input and an if statement, make your own Morse Code machine to transmit any letter you like!

Step 1

C - Morse Code Machine

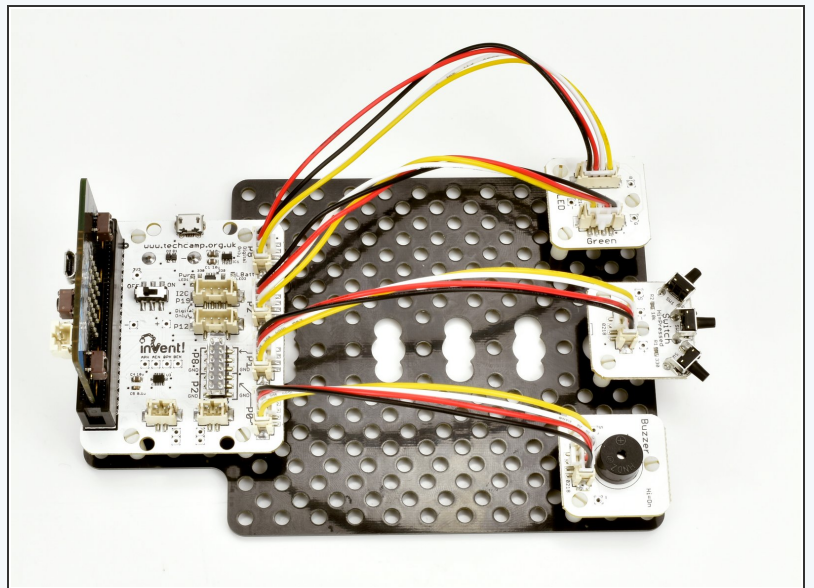
- So far, we have only used **outputs** - things that the robot can change to **1** or **0**.
- **Inputs** work in a similar, but opposite way - they can send a 1 or 0 signal **back to the robot**!
- Our program then needs to **decide** what to do, depending on whether the signal is 1 or 0.
- The **switch module** is a great example of an input - when one of the switches is pressed, the pin it is connected to will change to **1**.



Step 2

Setup the Switch

- Assemble your robot like the picture. The connections should be:
- Buzzer - **P0**
- Switch - **P1**
- Green LED - **P2**
- Red LED - **P8**



Step 3

read_digital

- To use an input, we need to **read** it to see whether it is 1 or 0.
- We can use the **read_digital** function for this - to use it we just need to write **pinnumber.read_digital()**
- In our case, this is **pin1**!
- **read_digital** will return **1** if the switch is pressed, and **0** if it isn't.

```
pin1.read_digital()
```

Step 4

If Statements

- To use an input, the robot needs to know how to **make a decision**.
 - We can do this with an **If statement**!
 - An If statement has **two parts**:
 - **Condition** - this is a **test**, and it goes after the if, followed by a **colon**. It will usually test if something is **equal** to something else - here we are seeing if **P1** is **1**.
 - **Conclusion** - this is some **indented code** after the condition, that is only run if the condition is **True**.
- i* To check if something is equal to something else in Python, we need to use **two equals signs (==)**.

```
if pin1.read_digital() == 1:  
    # Do something if condition is true
```

Step 5

Test the if

- Make the program in the picture - it tests **if the switch is pressed**, and then turns **on** the **red LED** if it is.
- **Test it out** - does it do what you expect?


```
9   p[6]=(p[5]+p[3])/2;p[7]=(p[4]+p[2])
10   while(p0()>p[7]and p1()>p[6]):d(0 i
11   def digital_read_line(s): return 1 if (
12   def analog_read_line(s): v=p0()if s==0
13   # Invent! Code End
14   # Start your code below here!
15
16   if pin1.read_digital() == 1:
17       pin8.write_digital(1)
18
19
20
21
22
23
24
```

Step 6

If in the Loop

- You may have found the LED only turns on if you are **holding the switch** when the program starts.
- This is because the switch is only checked **once** - then the program is done!
- **Put the if statement inside a while True: loop**, and test it out again - this way, the switch is checked every time the loop is run.
- The LED should now turn on when you **press the switch**!

```
9   p[6]=(p[5]+p[3])/2;p[7]=(p[4]+p[2])
10   while(p0()>p[7]and p1()>p[6]):d(0 i
11   def digital_read_line(s): return 1 if (
12   def analog_read_line(s): v=p0()if s==0
13   # Invent! Code End
14   # Start your code below here!
15
16   while True:
17       if pin1.read_digital() == 1:
18           pin8.write_digital(1)
19
20
21
22
23
24
```



Step 7

Turn off the LED

- Let's add some more code so the LED **turns off** when we let go of the switch!
- **Add another if statement** into the loop, that checks if the switch is **0** and then **turns off the red LED**.
- Your LED should now be **controlled by the switch!**



Step 8

Else

- Often, we want to do **one thing** if something is 1, and **something else** if it is 0, like turn on/off the LED.
- We can use **two** if statements, but there is a shorter way - using an **else**.
- **Replace** your second if statement with an else, to make an if/else statement:
 - If the condition is **true**, the code inside the **if statement** is run
 - If the condition is **false**, the code inside the **else statement** is run.
- Your LED should still be controlled by the switch, but the program is **simpler!**

```
11 def digital_read_line(s): return 1 if (
12 def analog_read_line(s): v=p0()if s==0
13 # Invent! Code End
14 # Start your code below here!
15
16 while True:
17     if pin1.read_digital() == 1:
18         pin8.write_digital(1)
19     else:
20         pin8.write_digital(0)
21
22
23
24
25
26
```

A red arrow points from the left margin to the 'else:' line in the code block.

Step 9

Your Own Morse Code

Machine

- You may have noticed we have actually made our own **Morse Code machine** already!
- By holding down the switch for a long time you can send a **dash**, and a short press would send a **dot**.
- For this challenge, add some more code so that:
 - When the switch **is** pressed, the green LED **and** buzzer are on
 - When the switch **isn't** pressed, only the **red LED** is on.



Step 10

Decode Morse Code

from a friend

- Time for a harder challenge!
- Using the Morse Code video from the first step, can you send a **secret message** to your neighbour?
- Get them to decode it and see what they come up with - **no speaking allowed!**



Step 11

Your Name in Morse

Code

- A super hard challenge now - can you write a program that sends Morse Code for your initials (or even your whole name) **automatically, but only when you press the switch?**
- If you're feeling really clever, try and **use some loops** to reduce the length of your program if you need to send lots of dots or dashes **in a row.**

