STEM_3

自　學　教　材

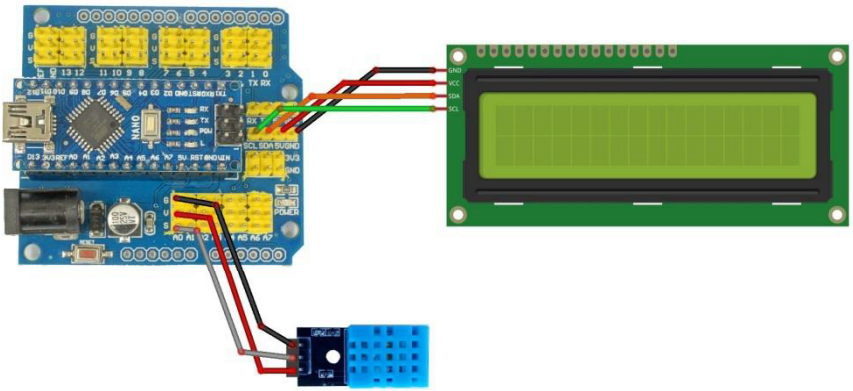
樣本

專案一：智能家居（上）

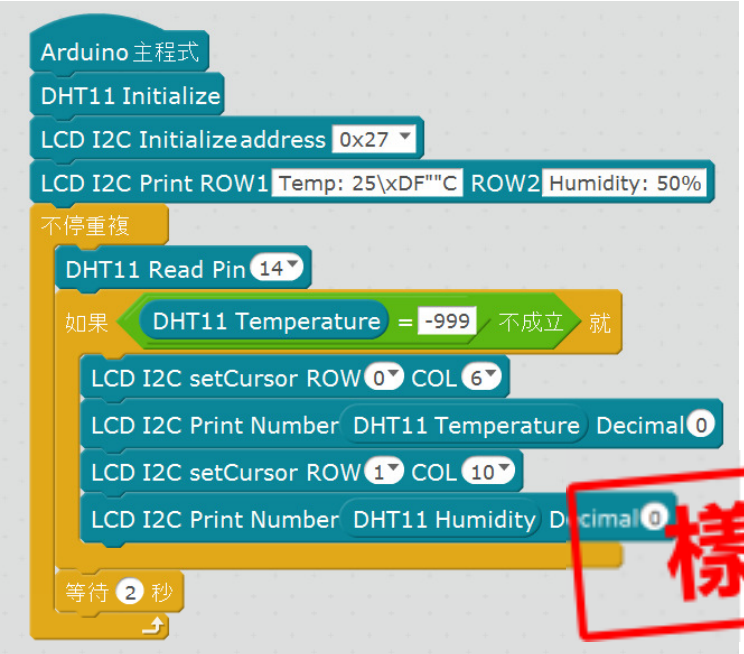
目錄

安裝好 DHT11 擴展後，你可以在 mBlock 的「積木調色板」裡面的「機器人模組」分類下找到一些新積木。

DHT11 感應器模組只需一個針腳（DATA 針腳）就可與 Arduino 主板進行通信。你可以將 DATA 針腳連接到 Arduino 主板的任何數位針腳。但你可能不知道一個秘密，實際上 Arduino 主板的所有模擬輸入針腳也可以用作數位針腳。它們作為數位針腳的號碼順序是 14（A0），15（A1），16（A2）... 21（A7）。



讓我們在連接 DHT11 感應器模組時驗證一下這個秘密吧。當我們用盡了 Arduino 主板上的所有其他數位針腳時，這個技巧就非常有用了。讓我們將 DHT11 感應器模組的 DATA 針腳連接到模擬輸入針腳 A0。在上面的例子中我們使用了一個 1602 I2C LCD 來顯示資料。



在範例中，當使用「DHT11 read Pin X」積木時，我們選擇了 14 作為針腳號。針腳號碼 14 到 21 是我們故意添加到 DHT11 擴展中，用以演示這個秘密技巧的。對於其他積木，你可能需要手動輸入針腳號碼。

每次執行「DHT11 read Pin X」積木時，溫度和濕度讀數將被儲存到「DHT11 Temperature」積木和「DHT11 Humidity」積木中。你可以把它們看作變數，並且總是儲存著上一次的讀數。如果你需要最新的讀數，在使用「DHT11 Temperature」積木和「DHT11 Humidity」積木之前，你必須先執行「DHT11 read Pin X」積木。

如果過於頻繁地讀取 DHT11 感應器模組，讀數會傳回-999。根據製造商的建議，重複讀取的間隔時間最短為 2 秒。為了防止 LCD 顯示-999，我們可以添加一個「如果...就」積木來檢查傳回值，然後再顯示它們。

在 1602 LCD 上顯示度數符號 (°) 有點棘手。原因是度數符號並不是 128 個標準 ASCII 字符的一部分。要顯示度數符號，需要使用一個特殊的語法「\ xDF “” 」，就像上面的例子一樣。

在上面的例子中，我們使用的另一個特殊技巧是在程式開始時將所有需要的字符都打印到顯示屏上。然後在「不停重複」積木之內，我們只打印有需要改變的字符。這樣我們的程式可以更簡潔，LCD 上的閃爍也會比較少。

請特別留意，本範例中不能將 DHT11 感應器模組插入 A4 和 A5 針腳。這是因為 A4 和 A5 針腳已經被用作 I2C 接口的 SDA 和 SLC 針腳，以控制 1602 LCD。

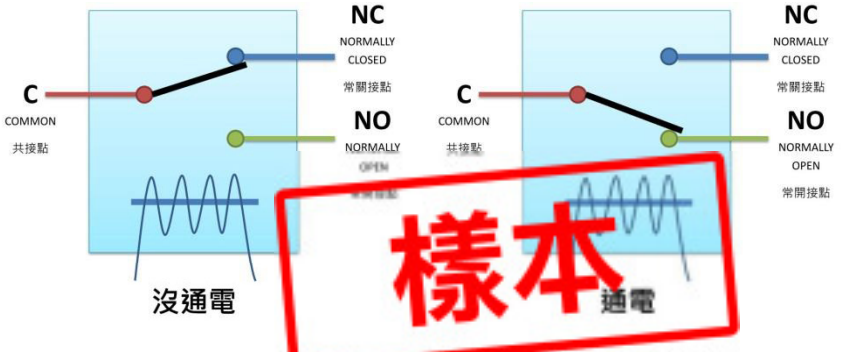
[資源檔案：DHT11.zip，using-DHT11.sb2]

5. 作業 1

1. 在 Arduino 主板上，有一顆名為「L」的 LED，它是接到 13 號針腳上的。試寫一個程式，只有當溫度高於 30°C 時才亮著「L」LED。
2. 試編寫一個程式，它不只顯示當前溫度，還會顯示由程式開始到當時的最高和最低溫度。

6. 繼電器

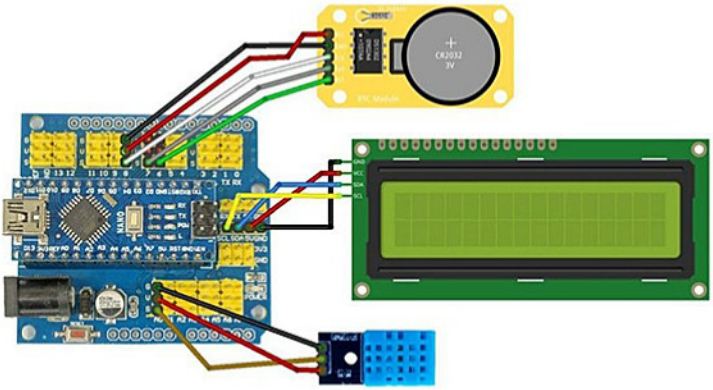
繼電器，是一種電子控制的開關，它是用較小的電流去控制較大電流的一種「自動開關」。繼電器大多使用電磁鐵，但也有使用其他工作原理的。



當我們想用 Arduino 主板去控制較高電流又或者較高電壓的裝置時，我們便需要使用繼電器。當繼電器沒有通電時，電磁鐵沒有吸力，可動電樞會停靠在常關接點(NC)，令到共接點(C)和常關接點(NC)接通。當繼電器通電時，電磁鐵產生吸力，令可動電樞停靠在常開接點(NO)，令到共接點(C)和常開接點(NO)接通。

通常，高壓裝置的電路會連接到 NO 針腳和 C 針腳，即是平時電路是處於斷開的狀態。當我們改變 Arduino 那邊的信號時，電磁鐵便被激活，然後電路閉合，從而啟動高壓裝置。

問題 2：



- 1602 LCD 模組連接到 I2C 介面。
- DS1302 RTC 模組連接到數位針腳 6，7 和 8。
- DHT11 感應器模組就連接到數位針腳 14 (即是 A0)。

建議答案

- 程式應該很好理解。我們先從 DS1302 模組取得日期時間，再從 DHT11 模組取得溫度濕度，然後把資料顯示在 LCD 上。
- 上例中我們在每個循環都等待 10 秒，但任何合理的等待時間皆可接受。

[project-1-assignment-3-answer-2.sb2]

STEM_3

Self-learning Guide

SMART HOME - PART 2

Contents

Actually there is a better solution. There is no need to update the LCD unless there is a change in the sensor status. We only need to update the LCD when the signal changes from HIGH to LOW or from LOW to HIGH. This is where we can use a technique called **state change detection** or **edge detection**.

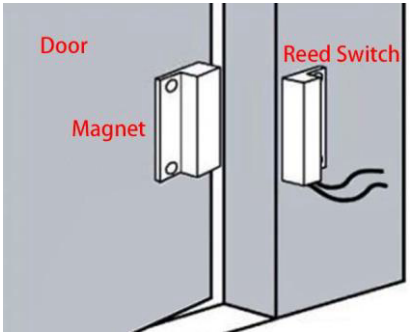
In the second program, we introduce two variables, **doorState** and **doorLastState**, to record the current state and last state of the sensor. And then we compare the current state and the last state in each cycle of the forever loop, the scripts would be executed only when the current state and last state are different.

Let us upload the second program to our Arduino board. You can see that the LCD stops flickering already. The second program saves the Arduino board a lot of unnecessary works. State change detection is a good technique and it can be used with most digital inputs of sensors and buttons.

[Resources file: reed-switch-1.sb2, reed-switch-2.sb2]

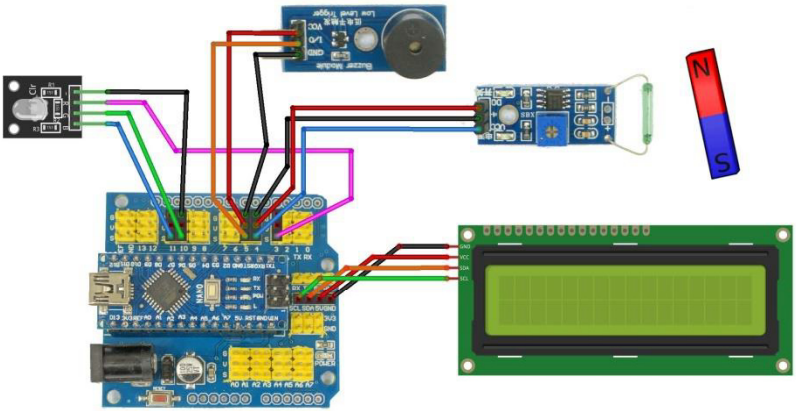
4. Door Alarm

One major application of the reed switches is door and window alarms. Many smart home systems also handle the home security. Let us simulate a door alarm system.

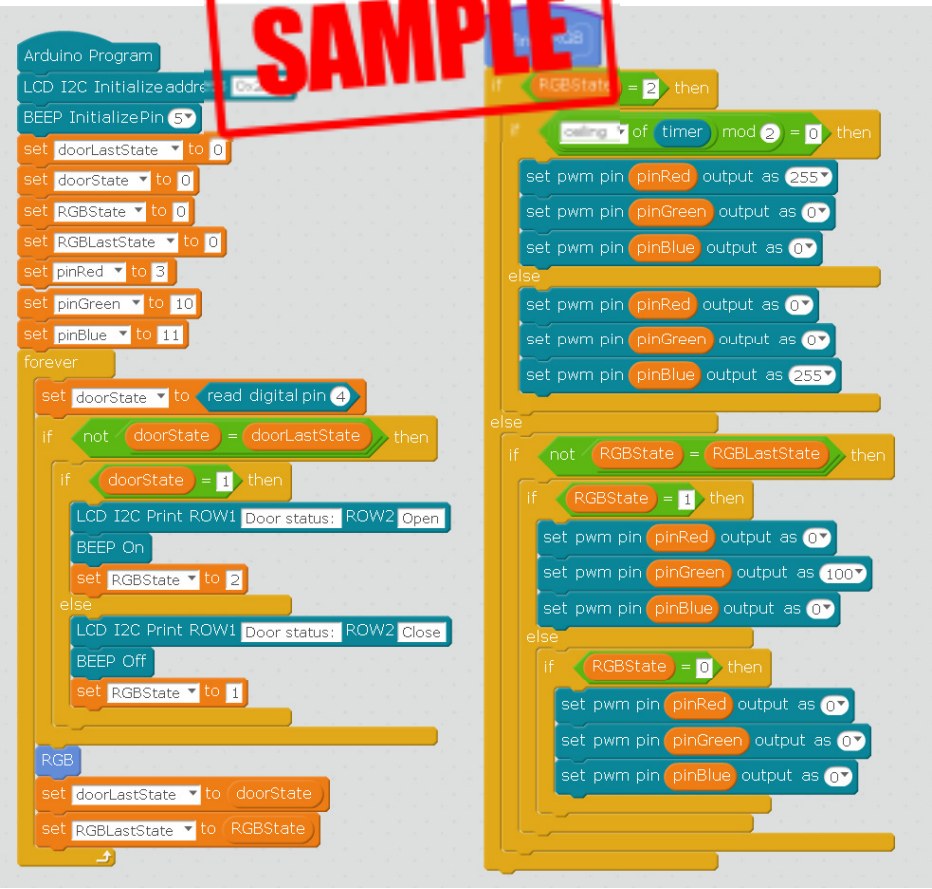


A door alarm system would usually consist of a reed switch, a sounding device and a lighting device. Normally, high decibel buzzers and special flashing signal lights would be used, but they are not very pleasant for practices at classroom or home.

We would be using the passive buzzer module as the sounding device and a full color RGB LED module as the lighting device. The RGB LED module consists of a 5mm RGB LED and three 150Ω limiting resistors to prevent burnout. Adjusting the PWM signal on each color pin will result on different colors.



The reed switch is connected to digital pin 4 of Arduino board. The 1602 LCD is connected to the I2C interface. The passive buzzer module is connected to digital pin 5 of Arduino board. The R (red) pin, G (green), B (blue) pin of the RGB LED module are connected to digital pin 3, 10, 11 respectively. Do notice that not all digital pins have PWM. The PWM function only works on pins 3, 5, 6, 9, 10, and 11 of Arduino NANO boards.



The program is very similar to the second example in the last section. The only differences are we would turn on the buzzer and show a red-blue warning signal when the door is open.

We use a variable **RGBState** to determine different color pattern. If **RGBState** is equal to 0, we turn off the LED. If **RGBState** is equal to 1, we change the color of the LED to green. If **RGBState** is equal to 2, the color of the LED would toggle between red and blue. The **Timer** block is used to determine the color of the LED when **RGBState** is equal to 2.

We do NOT call the custom block **RGB** when we want to change the color of the LED. Instead, we set the desire value to the **RGBState** variable. The custom block **RGB** is called in every cycle of the forever block and it would change the color of the LED according to the value of the **RGBState** variable.

The custom block **RGB** is a typical example of non-blocking timed operation. It can change the color of the LED at a fixed rate without sacrifice the responsiveness of the reed switch module.

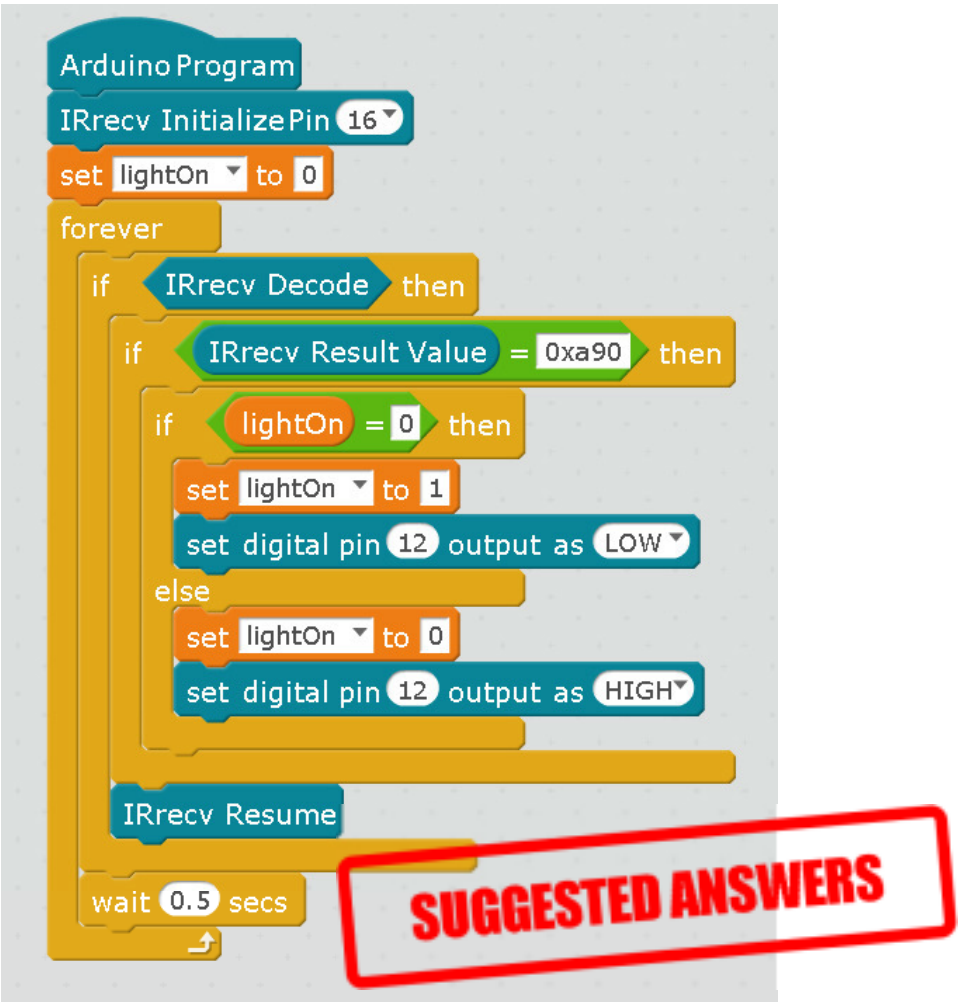
[Resources file: beep.zip, door-alarm.sb2]

5. Assignment 1

1. The position, distance, and orientation of the magnet all play a role in determining how the switch activates. Students should experiment with magnet and reed switch and describe your findings.
2. The color of the LED would change in every one second in the example of section 4. Change this rate to once every 2 second and once every 0.5 second.

6. LDR Sensor Module

LDR (light-dependent resistor) sensor module is used to detect the intensity of light.



- We are using the same Sony TV remote as in Section 10. The POWER button of the TV remote is used in this example. The hexadecimal value of the IR signal is A90.
- We use the same POWER button to turn on and turn off the USB light. Students can use separate buttons.
- To express a value in hexadecimal, we need to add the notation 0x before the value, so A90 become 0xA90.
- Students can also use the decimal value of the IR signal which is 2704 in the compare block, they are just the same.
- Do not forget to use the **IRrecv Resume** block after each read.

[project-2-assignment-4-answer-1.sb2]

