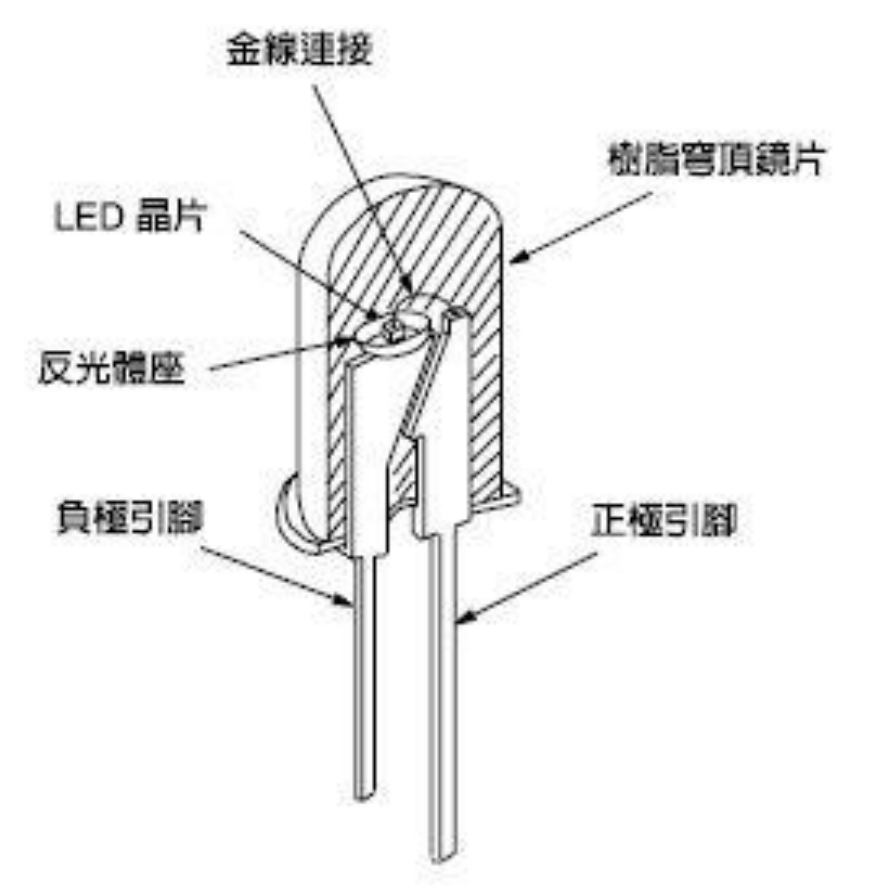
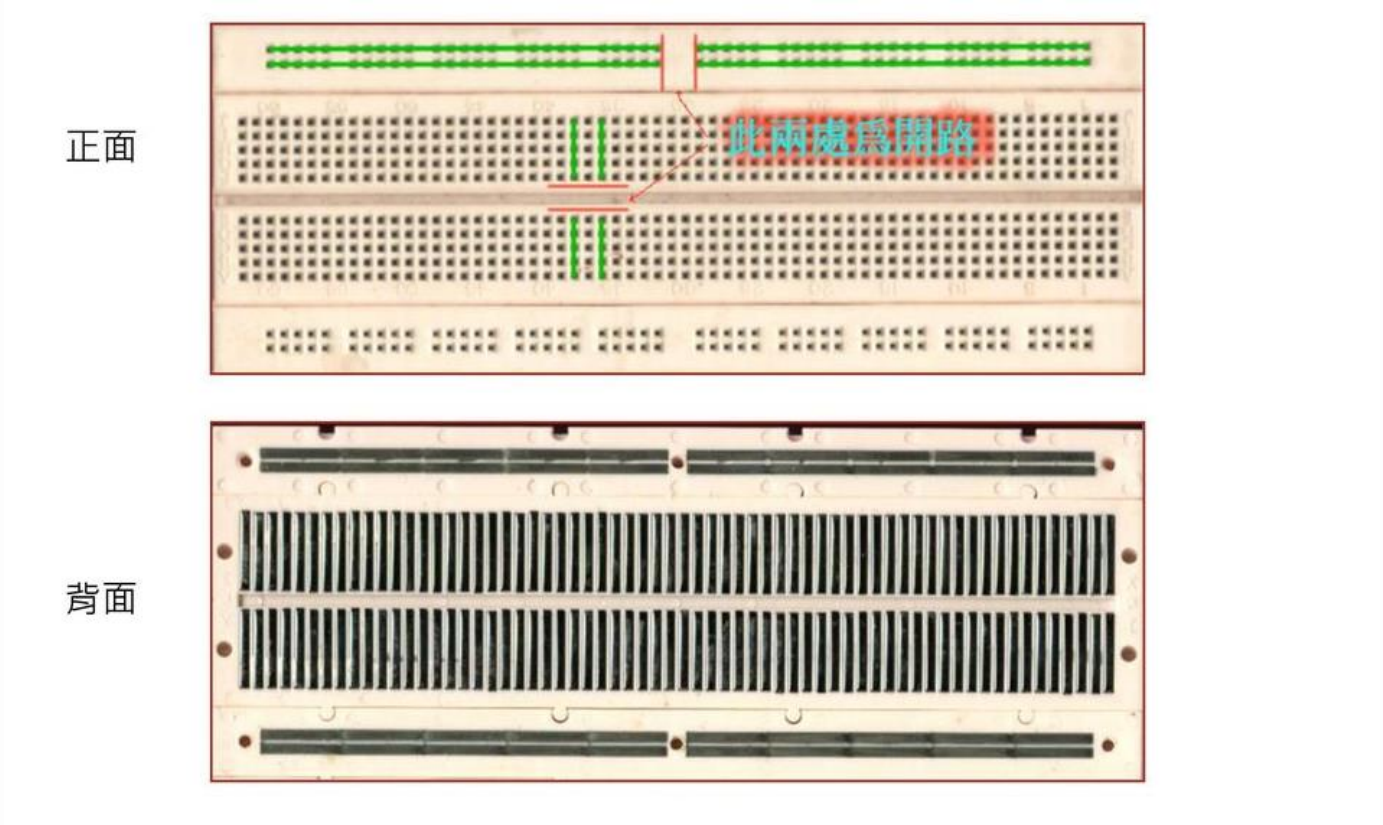
# Arduino實作四：摩斯電碼與捲簾快門效應

# 完成器材組裝

# LED構造

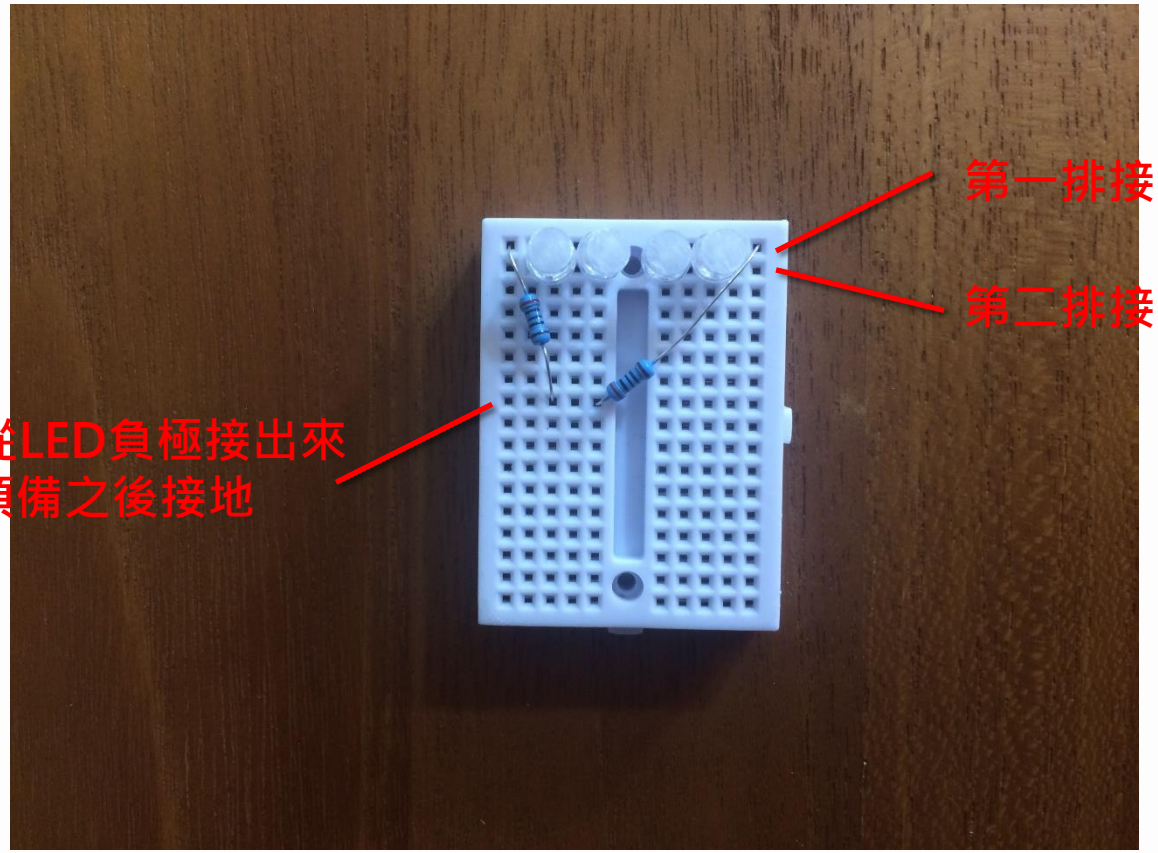


# 麵包板構造

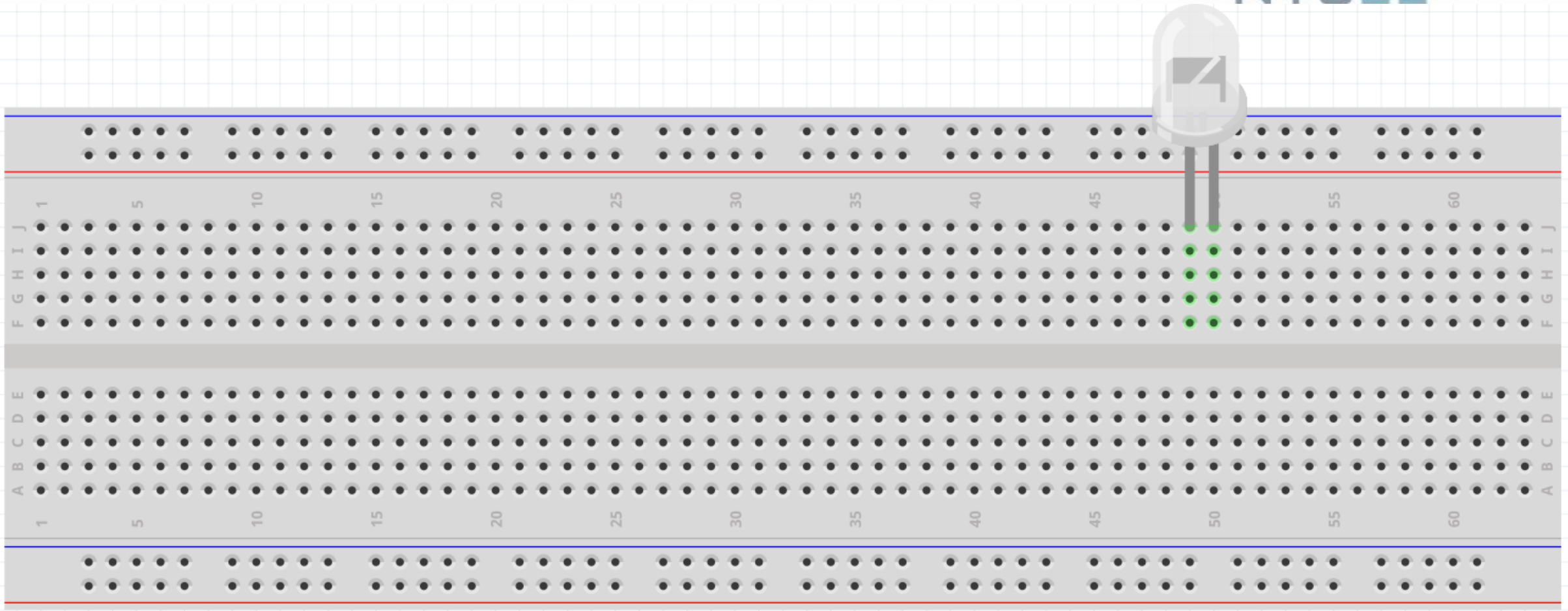


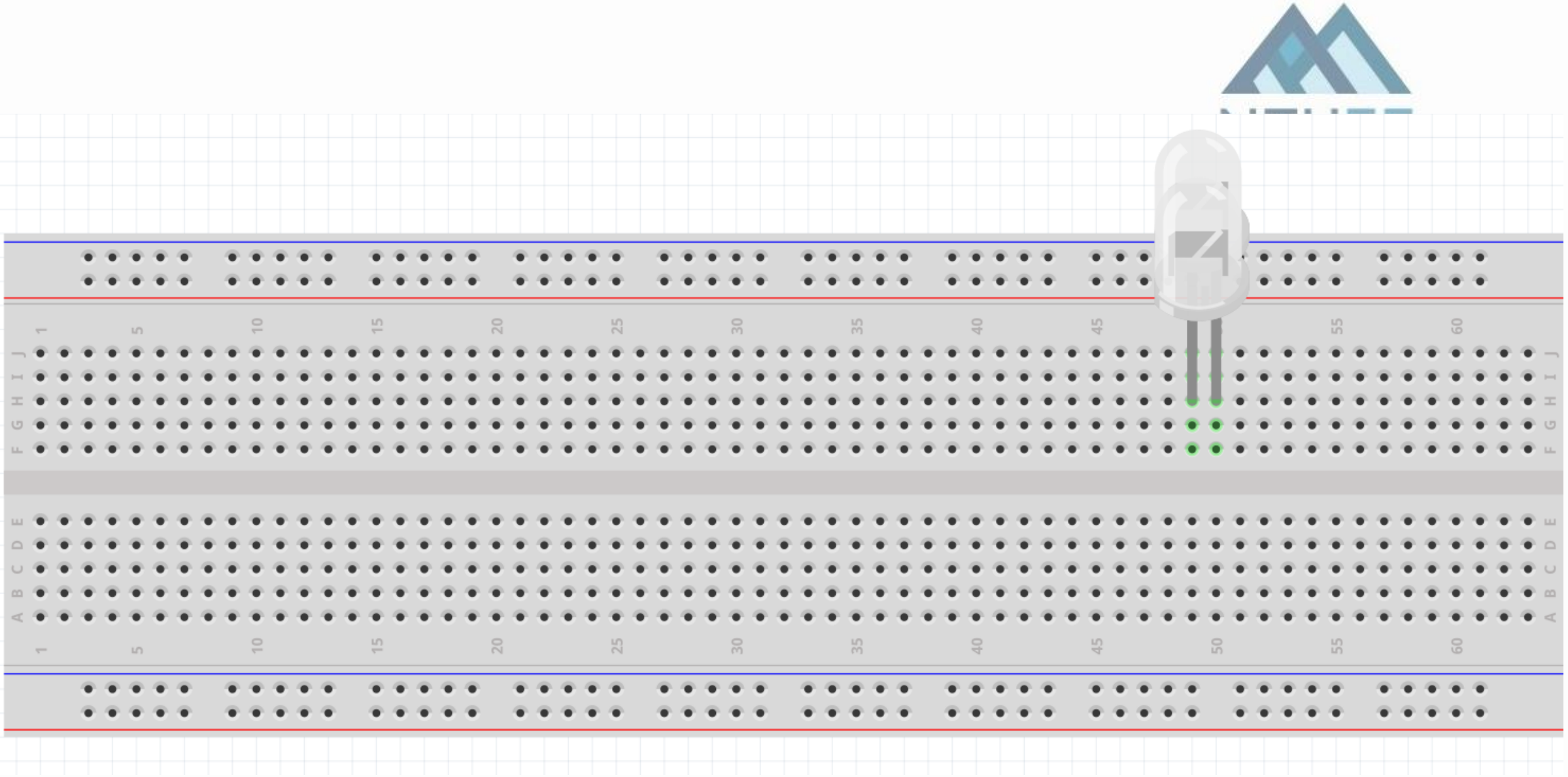
# 外殼組裝

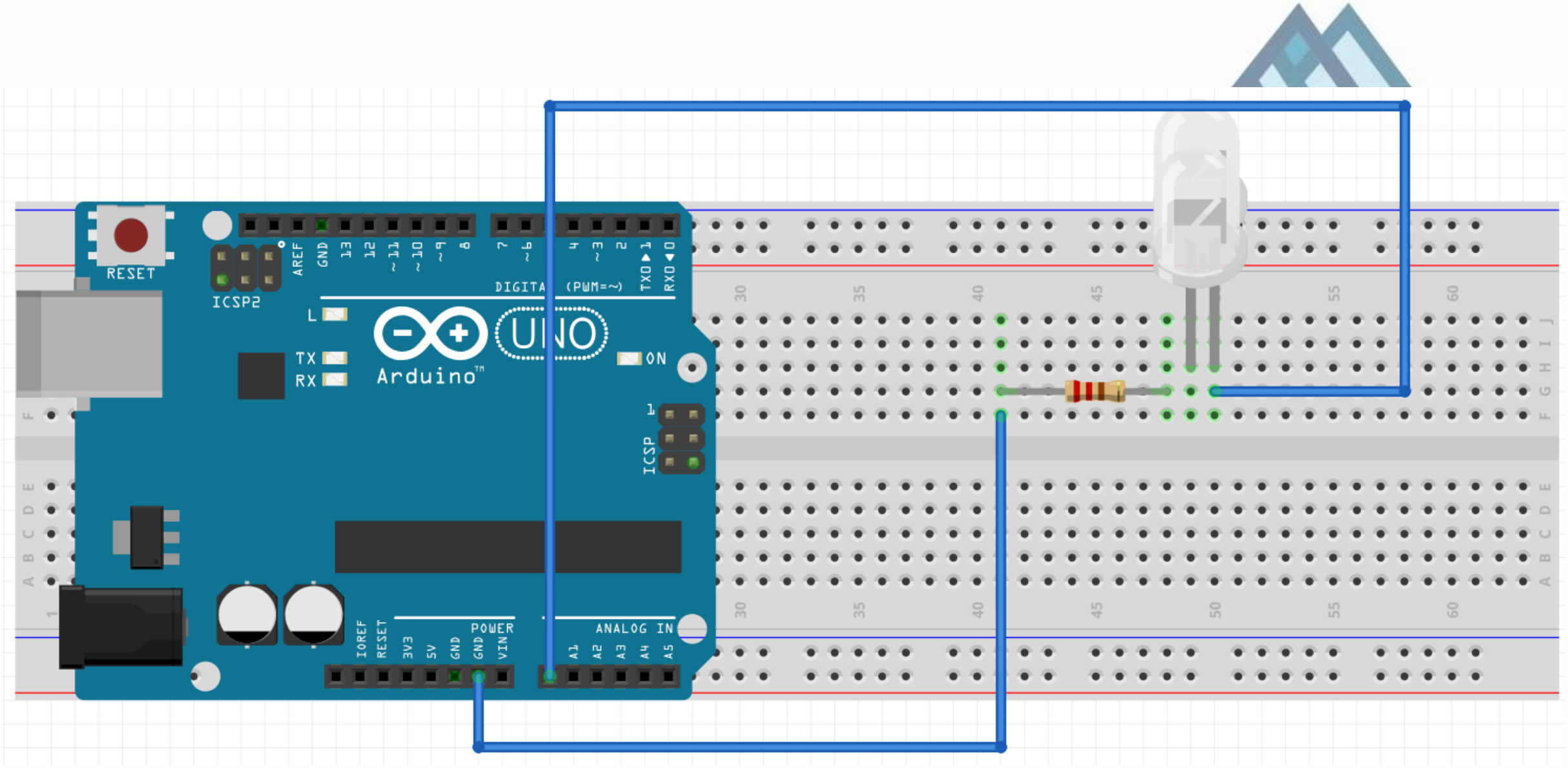
將平頭白光LED兩兩一組串接在麵包板上，共四顆，並接上電阻。

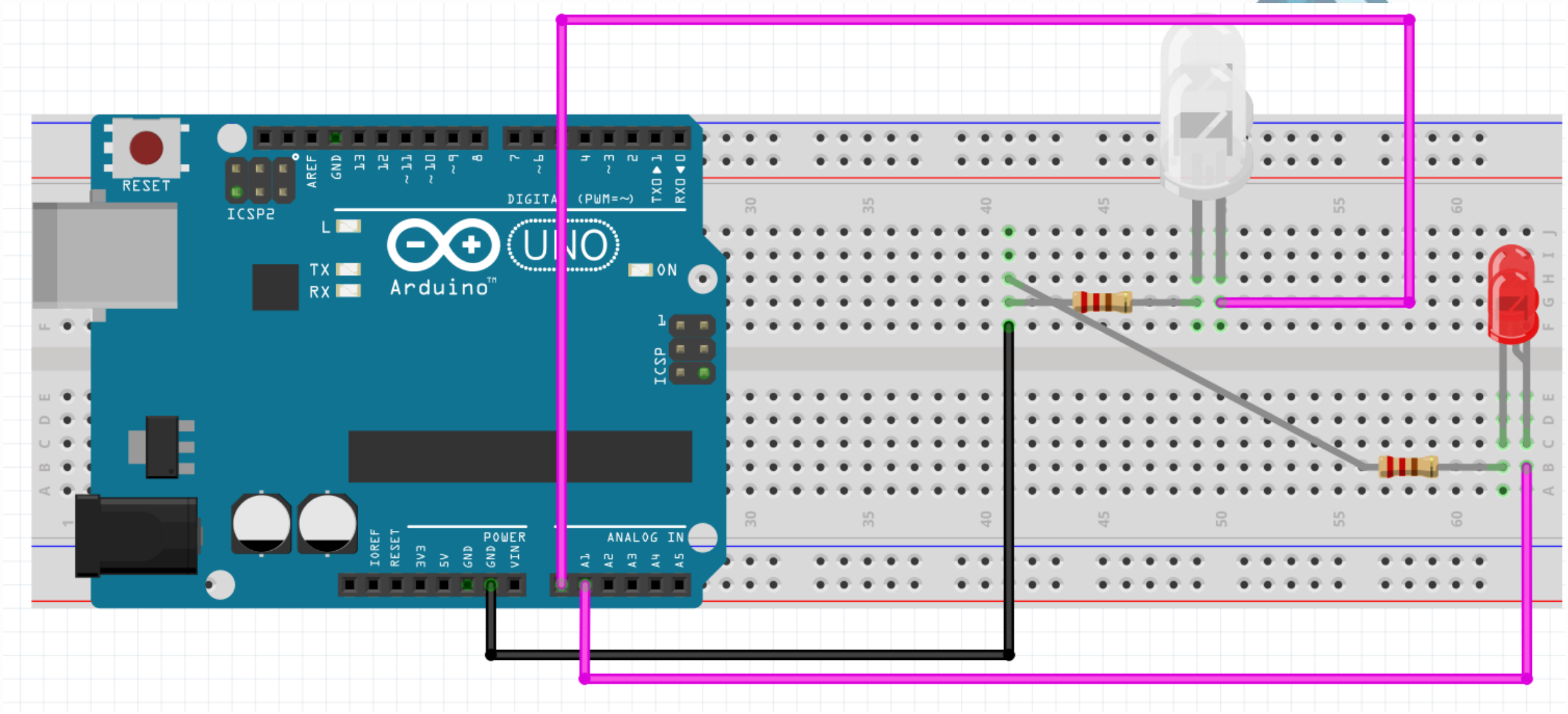


NTUEE




A B C D E F G H I J
1 5 10 15 20 25 30 35 40 45 50 55 60


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
4
~3
2
TX→1
RX←0
DIGITAL (PWM=~)
L
TX
RX
Arduino™
UNO
ON
ICSP
POWER
ANALOG IN
IOREF
RESET
3V3
5V
GND
GND
VIN
A1
A2
A3
A4
A5


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM=~)
L
TX
RX
Arduino™
UNO
ON
ICSP
POWER
ANALOG IN
IOREF
RESET
3V3
5V
GND
GND
VIN
A0
A1
A2
A3
A4
A5

# 範例一: 測試LED是否正常運作

是否有一閃一閃的?

# 摩斯電碼

摩斯電碼（英語：Morse code）是一種時通時斷的訊號代碼，通過不同的排列順序來表達不同的英文字母、數字和標點符號。是由美國人艾爾菲德·維爾（Alfred Lewis Vail）與薩繆爾·摩斯（Samuel Finley Breese Morse）在1836年發明。

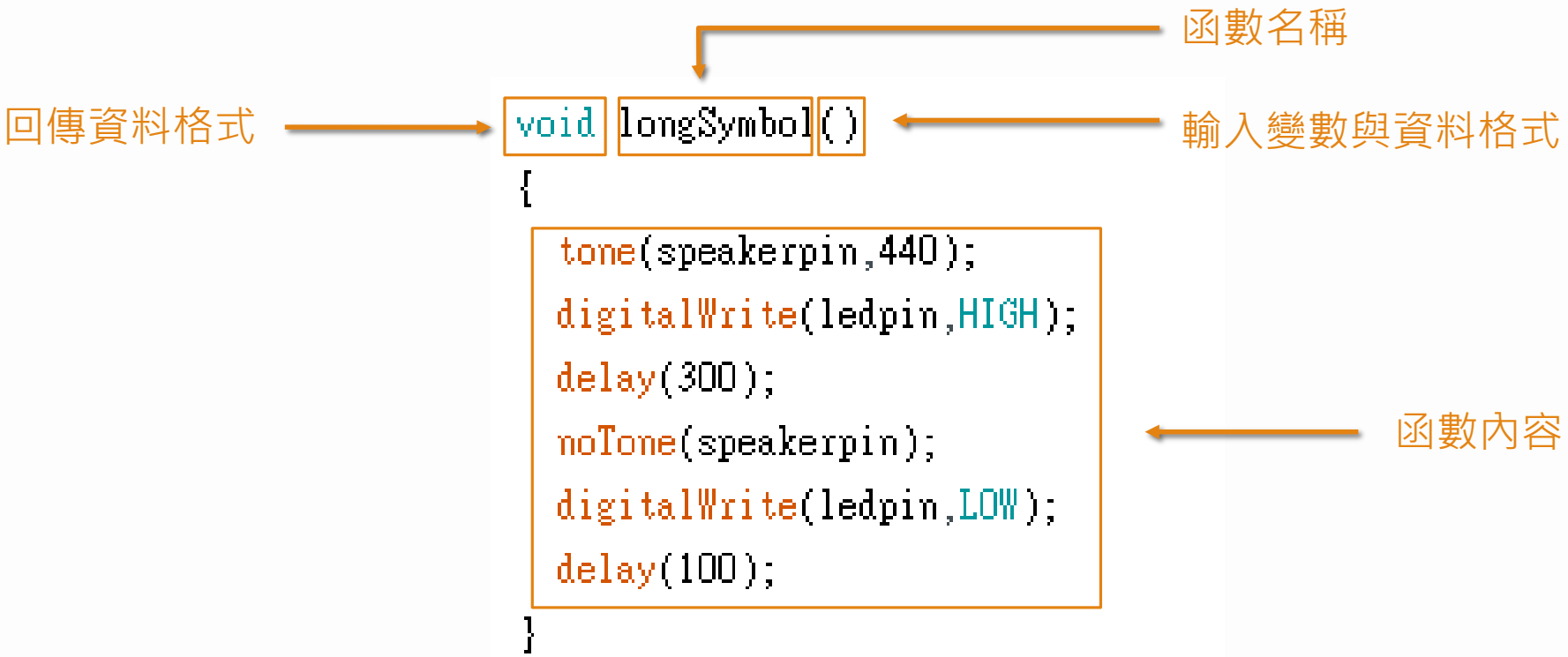
# 摩斯電碼

有五種代碼:

1. 點( . )
2. 劃( - )
3. 字元(點與劃)間的短暫停頓
4. 字母(a~z, 0~9等等)間的停頓
5. 詞彙間的停頓

以上五種代碼的時間長度分別為1、3、1、3、7個單位時間長。

# 定義函數


函數名稱
回傳資料格式
輸入變數與資料格式
函數內容
void longSymbol()
{
tone(speakerpin,440);
digitalWrite(ledpin,HIGH);
delay(300);
noTone(speakerpin);
digitalWrite(ledpin,LOW);
delay(100);
}

# 範例二: 鍵入數字自動輸出摩斯電碼

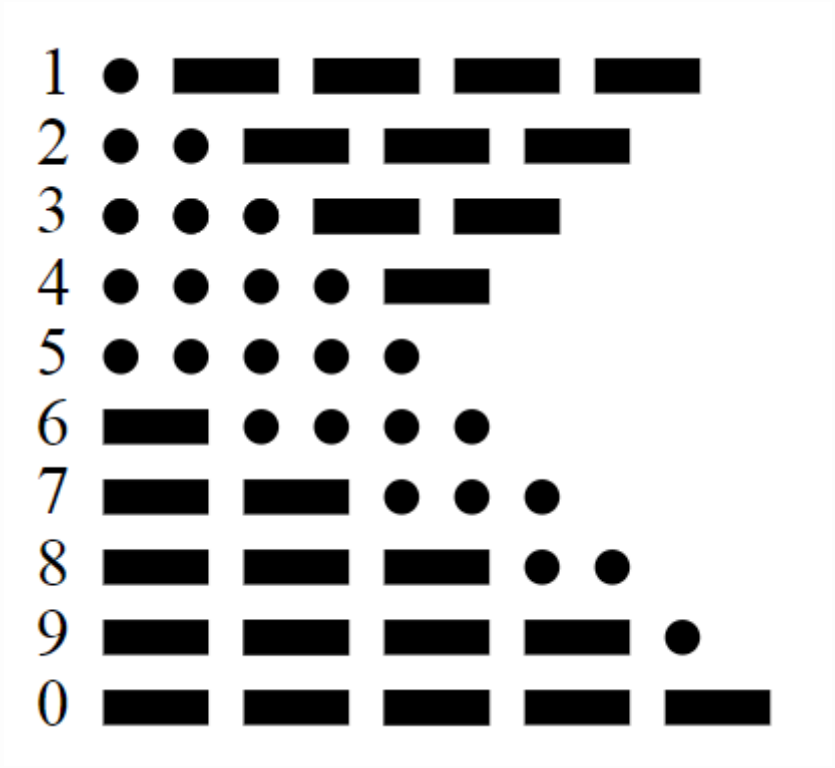
1. 定義「點」與「劃」的函數(可以參照前一頁)，發出訊號時同時亮LED燈和發出聲音。

2. 可以隨意定義單位時間間隔，以前頁函數為例，單位時間間隔為100毫秒。

3. 按鍵板的部分和實作一類似，在使用者按下0~9的按鍵時發出對應的摩斯電碼即可。

4. 0~9的摩斯電碼對照表參見下一頁。

# 摩斯電碼對照表



# 摩斯電碼通訊實驗

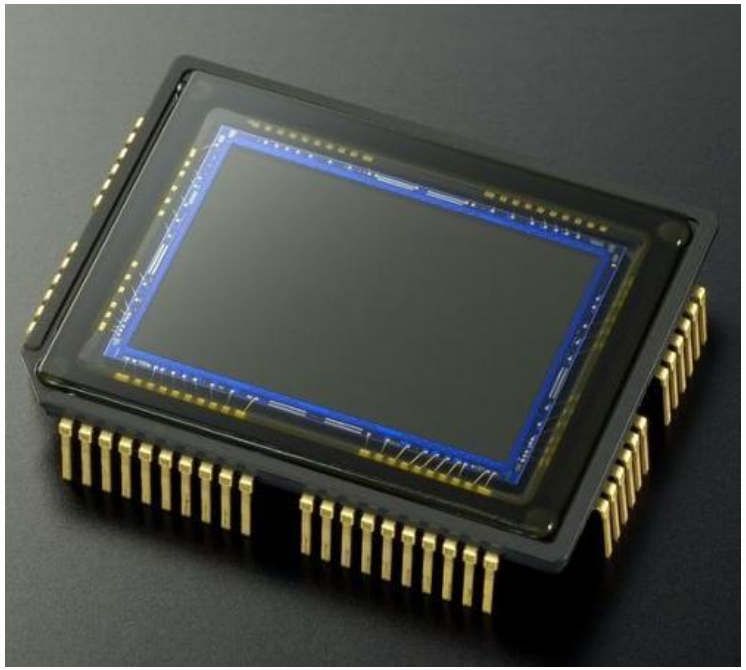
兩人一組試著互相使用摩斯電碼傳數字串

是否能夠清楚的接收正確資訊?

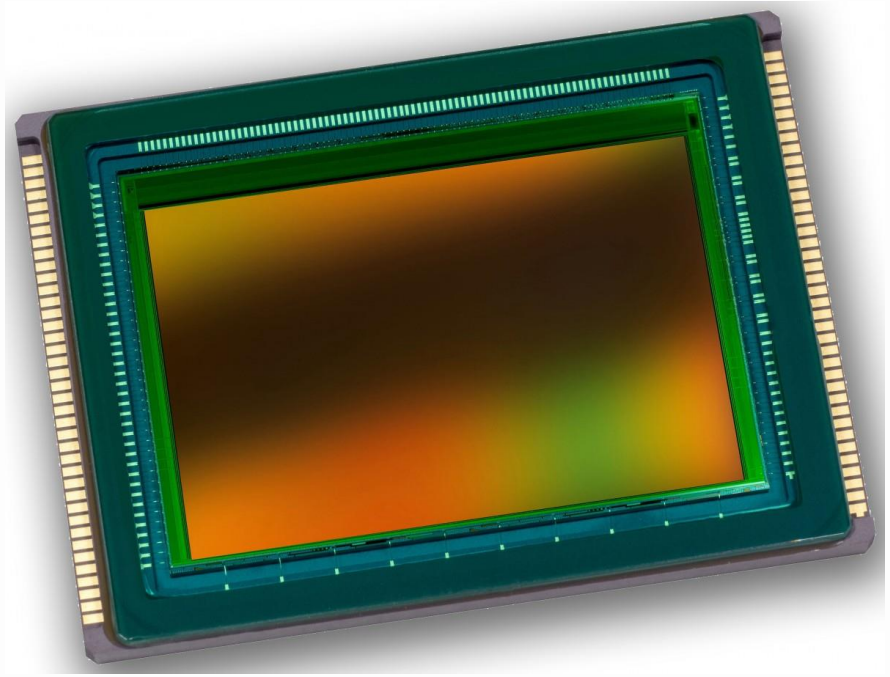
# 捲簾快門效應

捲簾快門效應(Rolling shutter effect)為CMOS感光元件獨有的現象，原理是當影像訊號輸出給主機時，影像中的每一行都有獨立的放大器。這些放大器輸出訊號有微小的時間差，導致在拍攝快速移動的物體時會得到變形的影像

# 感光元件—CCD or CMOS



CCD



CMOS

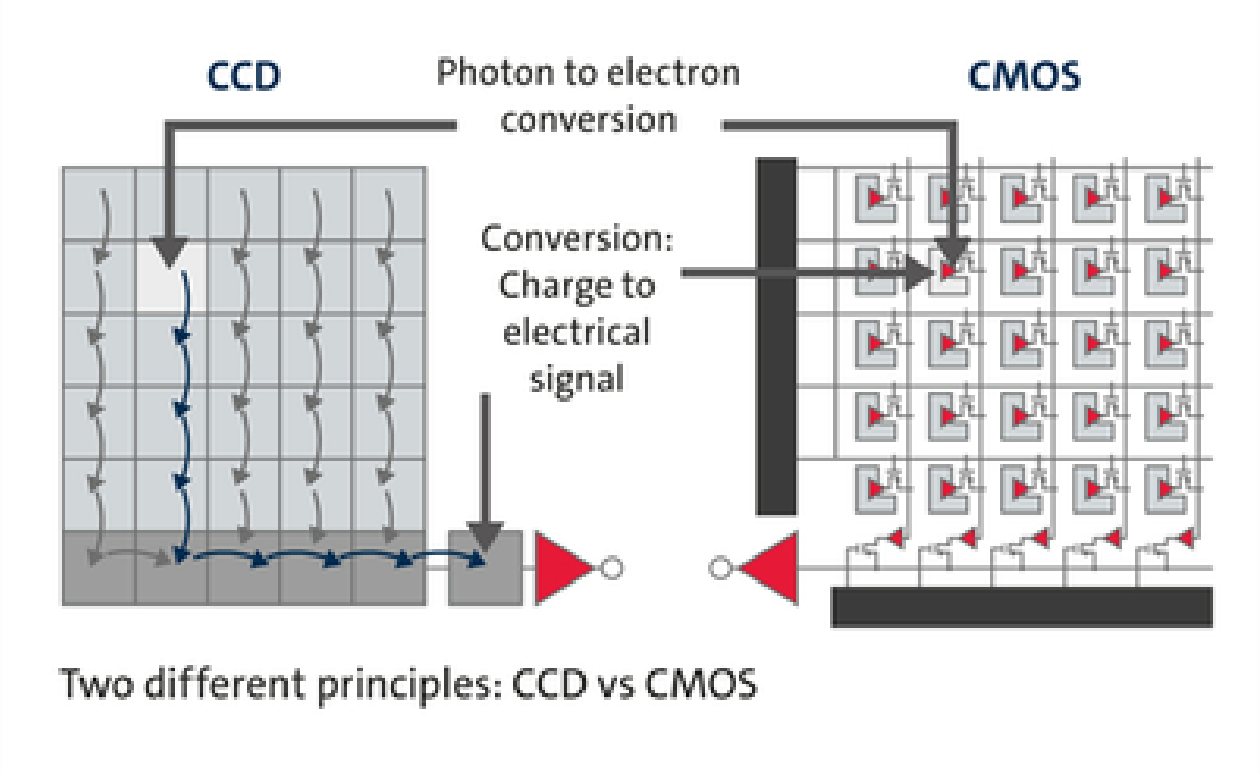
# 感光元件—CCD or CMOS

CCD(Charge-Coupled Device): 透過感光二極體接收來自鏡頭光刺激並轉成電荷，再用電容陣列暫存接收到的電荷。呈像完畢再將整個畫面的電荷放大並轉成數位訊號輸出。

CMOS(Complementary Metal–Oxide–Semiconductor): 一樣使用感光二極體，但CMOS運用大量生產的MOS製程使每個畫素都有獨立的放大器，可以直接將每個畫素的電荷轉成數位訊號輸出。比起CCD，這樣的做法可能造成訊號放大比例不均勻，但相對的影像輸出的速度也比CCD快得多。

近年來CMOS由於製程便宜、輸出速度快、技術不斷進步等原因，在數位相機的市場逐年取得領先地位，市占率也越來越高，現今的手機相機和數位相機感光元件幾乎都是使用CMOS。

# 感光元件—CCD or CMOS




Two different principles: CCD vs CMOS

# 感光元件—CCD or CMOS

| 項目 | CCD | CMOS |
| --- | --- | --- |
| 對雜訊的抗性 | 較好 | 易受雜訊影響 |
| 感光能力 | 較敏感 | 較遲鈍 |
| 功耗量 | 較高 | 較低 |
| 製程難度 | 需要客製化的製程 | 製程難度較低 |
| 生產成本 | 較高 | 較低 |
| 動態的廣度 | 較廣 | 較窄，會有色調失真的情形 |
| 反應速度 | 較慢 | 較快 |

# 捲簾快門效應

# 範例影片

吉他弦的震動:

https://www.youtube.com/watch?v=jcOKTTnOIV8

旋轉中的風扇葉片:

https://www.youtube.com/watch?v=EaB9EHeDLSk

# 範例三: 閃燈實作(二)

如同上一堂課作的基本閃燈實作練習，寫一個簡單的程式讓LED燈穩定的在亮與暗間切換。但這次使用delayMicroseconds()將亮/暗的間隔設定為200微秒，然後用以下兩種方式觀察:

1. 肉眼觀察
2. 用手機相機對著LED拍攝

2vs8判斷???

# Bonus: 一鍵式摩斯電碼

實際上用來發送摩斯電碼的裝置只需要一個按鍵來發送「點」與「劃」的信號。試著僅使用按鍵板上的一個按鍵，透過長按與短按來打出摩斯電碼，並將打出來的內容顯示在監控序列埠上，並把摩斯電碼轉換成英文和數字。

提示1: 可以使用millis()函數來計算按壓的時間長度。

提示2: getKey()函數無法獲得連續按壓的資訊，可以嘗試getState()獲得按鍵狀態。

提示3: 使用KeypadEvent在loop以外隨時偵測按鍵狀態。