探討彈跳現象-以光遮斷感測器製作風速感測器為例

Exploring the Bouncing Phenomenon - A Case Study of Fabricating a Wind Speed Sensor Using an Optical Interrupter Sensor

郭哲源[1] 周佳弘[2]
GUO,JHE-YUAN[1] JHOU,JIA-HONG[2]
[1]國立台南大學電機系
[1]Department of Electrical Engineering, National University of Tainan
[1]Email：home7367962@gmail.com
[2]義守大學電子系 博士
[2]I-SHOU UNIVERSITY Department of Electronic Engineering
[2]Email: topjkpos@gmail.com

摘要

本文旨在研究製作風速感測器遇到的問題，光遮斷感測器的彈跳現象。因為開關的機械結構問題，在開與關後都會繼續產生震盪，此為彈跳現象。本文研究透過硬體或軟體來解彈跳，使感測器的回傳數值可以更加精確，並在最後使用軟體解彈跳，程式編碼的方式，運用延遲來迴避彈跳現象的發生，使讀數更精準。

關鍵字：彈跳現象；接點彈跳

Abstract

This paper aims to investigate the issues encountered in the fabrication of a wind speed sensor, specifically the bouncing phenomenon of the optical interrupter sensor. Due to the mechanical structure of the switch, oscillations continue to occur after opening and closing, which is known as the bouncing phenomenon. This paper explores the use of hardware or software to debounce, making the return values of the sensor more accurate. Finally, software debouncing is used, employing delay in the coding process to avoid the occurrence of the bouncing phenomenon, resulting in more precise readings.

Keywords：switch bounce;bounce

壹、 前言

在現代科技的推動下，感測器的應用已經滲透到我們生活的各個角落，從環境監測到教育實驗，其影響力不容忽視。然而，感測器的精確度和可靠性是實現其潛力的關鍵。本論文將探討在製作風速感測器時遇到的一個具有挑戰性的問題：光遮斷感測器的彈跳現象。

彈跳現象是一種常見的電子問題，它會導致感測器的讀數不準確，進而影響整個系統的性能。在我們的風速感測器實驗中，我們發現光遮斷感測器的讀數並未達到預期，經過一系列的研究和實驗，我們確認了問題的根源在於彈跳現象。

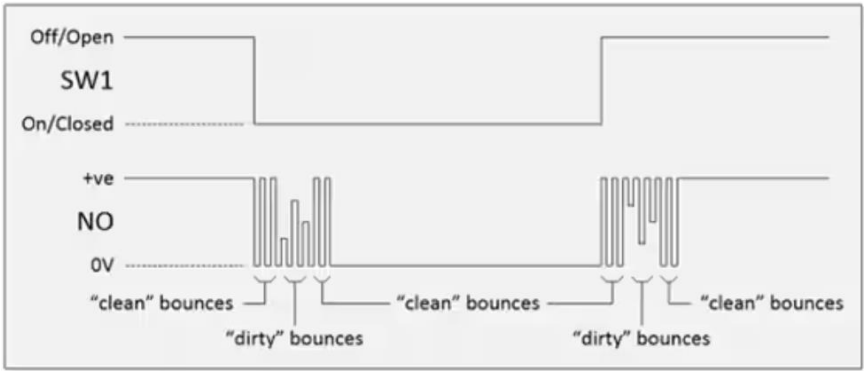
本論文的目標是深入研究彈跳現象，並尋找有效的解決策略。我們將首先介紹彈跳現象的原理，然後分析它如何影響光遮斷感測器的性能。接著，我們將提出並驗證幾種可能的解決方案，並評估它們在實際應用中的效果。

我們希望這項研究能夠對感測器的設計和製作提供有用的見解，並對教育實驗有所貢獻。我們相信，透過對彈跳現象的深入理解和有效的解決策略，我們可以製作出更精確、更可靠的風速感測器。

貳、 文獻探討

一、 彈跳現象的問題

彈跳現象，也稱為接點彈跳，是一種常見的電子問題，尤其在使用機械式開關時。當開關在被按下或釋放的瞬間，其接觸點因屬於機械結構的接觸，因此通電與斷電的瞬間常會因力道或金屬表面的接觸瞬間，有一段電位不穩定的狀態存在(圖一)，這就是所謂的彈跳現象。[1]



圖一　　彈跳現象[2]

這種現象會導致感測器的讀數不準確，進而影響整個系統的性能。例如，若數位電路速度夠快，可偵測多次彈跳並加以反應，就會有嚴重的後果。因此，工程師的任務就是避免或減輕彈跳的影響，或是將開關「解彈跳」。

二、 如何解彈跳

(一)硬體解彈跳

1960 與 1970 年代左右，多使用硬體解彈跳的方式，透過多種硬體技術達成，從較為簡易的延遲電路搭配 SPST 開關，到複雜的設定，如：重製閘門功能。

而近年會使用硬體解彈跳的情況多半是受硬體限制，像是某些處理器較為小型，其中可能受低效能、記憶體容量過少的問題，不能負載大量的程式編碼，才會考慮使用這種方式解彈跳。近年則是多使用下面介紹的軟體解彈跳」。

(二)軟體解彈跳

這種方法的成本較低，是近期主要的解彈跳方式，在程式碼上下功夫來消除彈跳現象。軟體解彈跳的方法有很多種，例如可以設定一個延遲，利用時間延遲來判斷高低電位是否變化，只有當開關狀態在一段時間內保持不變時，才認為開關狀態已經改變。

參、研究實施與設計

一、 軟體解彈跳

前言中有提到，彈跳現象是在製作風速感測器時遇到的問題，硬體結構的設計中，風會帶動風車轉動，進而使同一個軸體上的紙片遮擋光遮斷感測器，即可計算風車的轉速(圈數)。在最初的程式設計中(圖二)，只有單純偵測數位訊號的改變來增加圈數的變數，但偵測到的圈數與實際的不符，經實驗後發現是彈跳現象的問題，於是我在數位訊號做改變時增加 1 毫秒的延遲(圖三)[3]，來抵抗彈跳現象的發生，經測試後實際的圈數與偵測值相等。

```
// 宣告變數 round
int round = 0;

void setup() {
  // 設定數位腳位 7 為輸入模式
  pinMode(7, INPUT);
  // 啟用串列通訊
  Serial.begin(9600);
}

void loop() {
  // 讀取數位腳位 7 的狀態
  int digitalValue = digitalRead(7);

  // 如果數位腳位 7 的狀態為 LOW (等於 0)
  if (digitalValue == LOW) {
    // 變數 round 加 1
    round++;
    // 顯示變數 round 的值到序列埠 (監視視窗)
    Serial.print("round = ");
    Serial.println(round);
  }

  // 延遲 100 毫秒
  delay(100);
}
```

圖二

```
void loop() {
  // 讀取數位腳位 7 的狀態
  int digitalValue = digitalRead(7);

  // 去彈跳機制
  if (digitalValue != lastState) {
    delay(1); // 等待 1 毫秒
    digitalValue = digitalRead(7);
  }

  // 如果數位腳位 7 的狀態由 LOW 變為 HIGH (從 0 變成 1)
  if (digitalValue == HIGH && lastState == LOW) {A
    // 變數 cc 加 1
    cc++;
    // 顯示變數 cc 的值到序列埠 (監視視窗)
    Serial.print("cc = ");
    Serial.println(cc);
  }

  lastState = digitalValue; // 更新上一次的狀態

  // 延遲 1 毫秒
  delay(1);
}
```

圖三

肆、 結論

本研究通過一系列實驗，成功證明了軟體解彈跳方法在光遮斷感測器的應用中能有效減少彈跳現象，從而提高風速感測器的準確性和可靠性。通過在數位訊號變化時增加 1 毫秒的延遲，我們不僅解決了彈跳現象的問題，也使得風車轉速的偵測更為精確。這項研究不僅對風速感測器的設計和製作有重要的意義，也為其他感測器應用中的彈跳現象提供了一個可行的解決方案。

參考文獻

[1]E. Lemoine, C. Guines, A. Pothier and P. Blondy, "Asymmetrical mechanical design for bouncing suppression in RF-MEMS switches," 2014 44th European Microwave Conference, Rome, Italy, 2014, pp. 1600-1603

[2] Digi-Key Electronics. (2021). How to Implement Hardware Debounce for Switches and Relays When Software Debounce Isn’t Appropriate. Retrieved from https://www.digikey.tw/en/articles/how-to-implement-hardware-debounce-for-switches-and-relays

A. Peschot, C. Poulain, C. Valadares, B. Reig, N. Bonifaci and O. Lesaint, "Evolution of contact bounces in MEMS switches under cycling," 2014 IEEE 60th Holm Conference on Electrical Contacts (Holm), New Orleans, LA, USA, 2014, pp. 1-6