

Lab 3

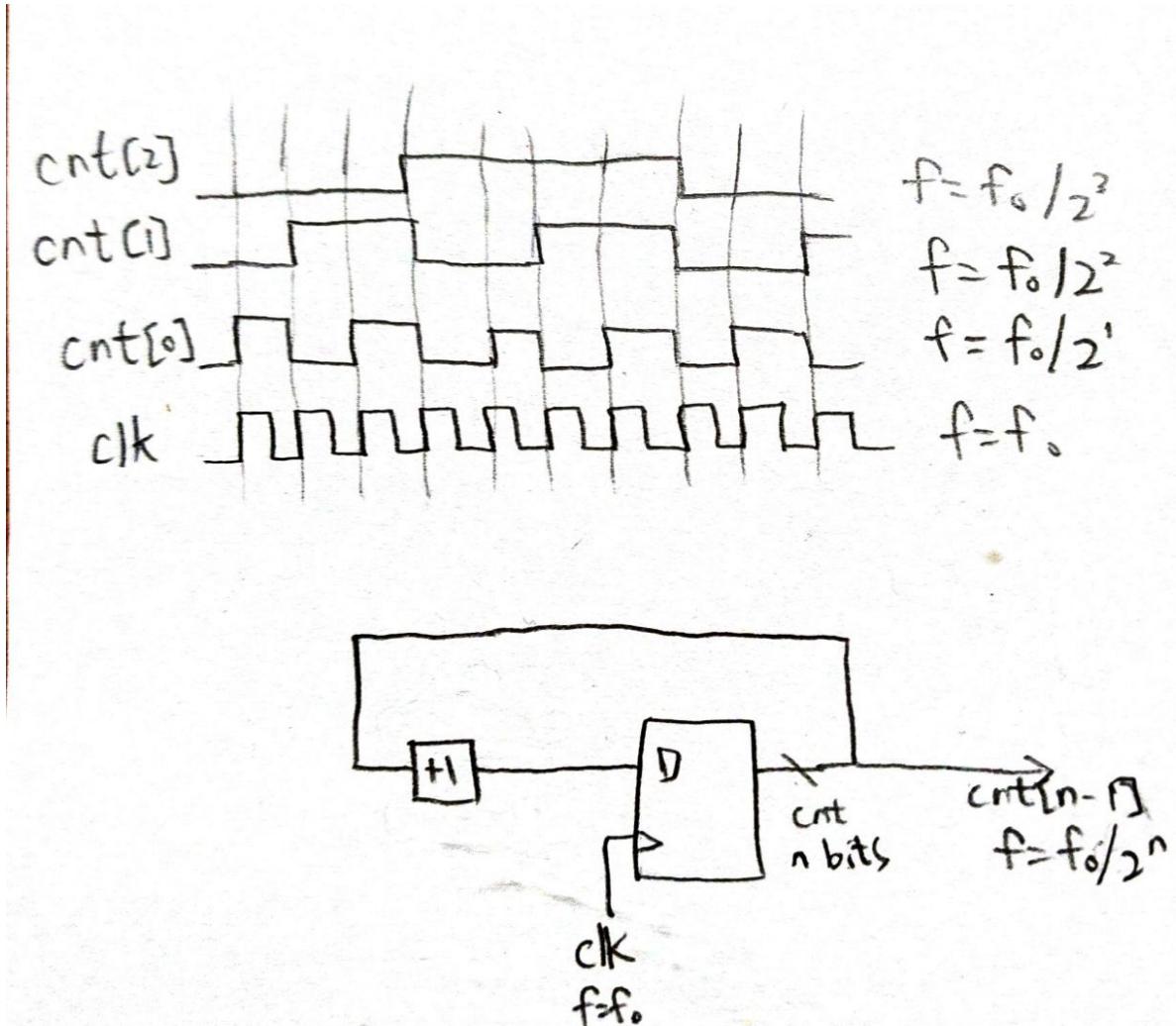
學號: 109062302

姓名: 郭品毅

1. 實作過程

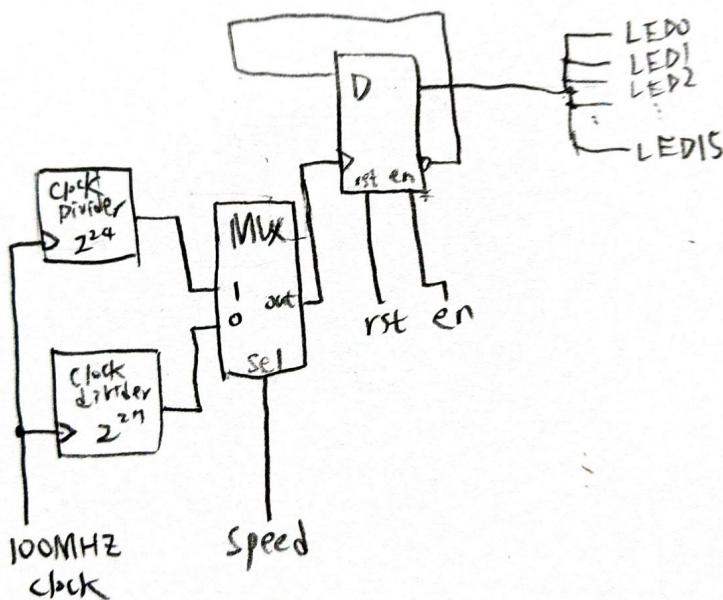
- clock_divider

- 因為是下除 2 的幕次，所以可以使用一個 n bit 的 binary counter, 並取其中的 nth bit (MSB) 就會是下除 2^n 的 clock signal 了

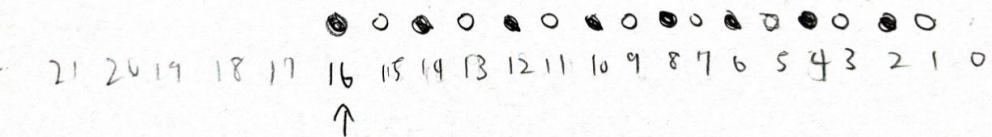


- Lab 3_1

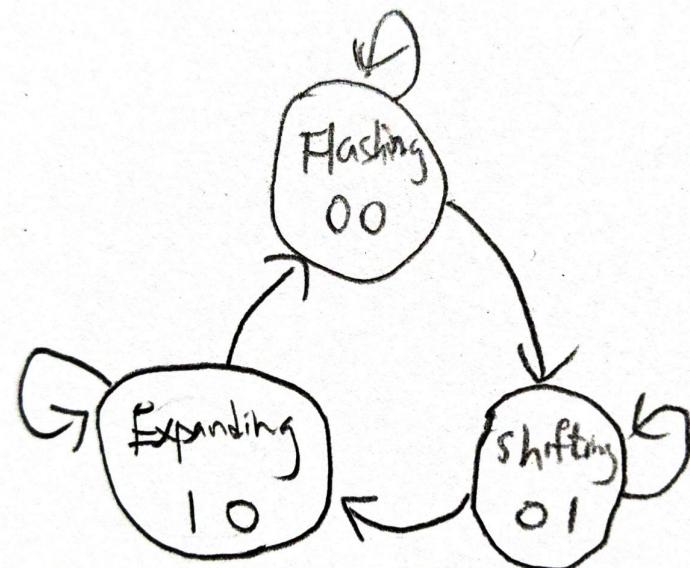
- 使用前面製作的 clock divider, 分別產生下除 2^{24} 與 2^{27} 的 clock signal, 透過 MUX 切換 memory element 使用的 clock, 以讓 LED 切換不同速度的閃爍頻率



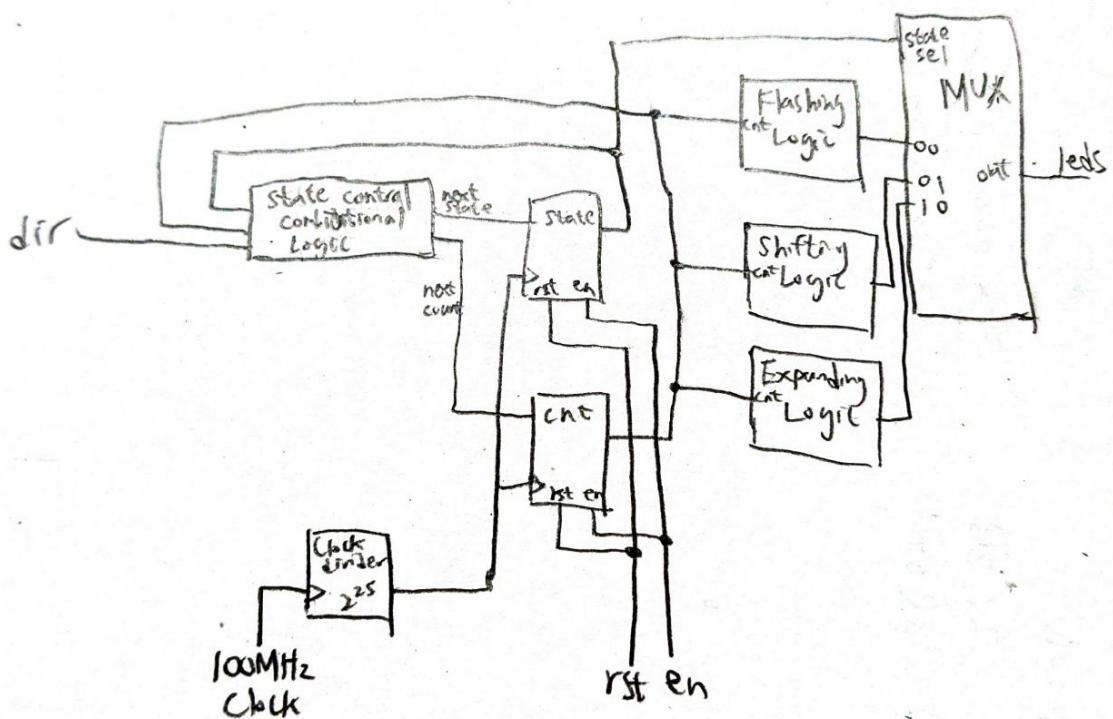
-
- Lab 3_2
 - 使用 Finite State Machine
 - 三個模式個別是一個 state, 每個模式達到一定條件之後就會換到下一個 state / 模式
 - 使用一個 cnt 的 flip flop, 記錄目前在模式中的哪一個狀態
 - 使用目前的 state 和 cnt 計算目前 led 該有的樣子
 - 在 Flashing Mode, cnt 從 0 開始往上加, 偶數亮, 奇數暗, cnt 到 11 時進入 Shifting Mode
 - 在 Shifting Mode, cnt 一開始設為 16, 表示預設的狀態, cnt 減小時, 表示往右 shift, 當 cnt 為 0 時, 表示全部的燈都 shift 出右側了; cnt 減增加時, 表示往左 shift, 當 cnt 為 31 時, 表示全部的燈都 shift 出左側了



-
- 當 $cnt \leq 0$ 或 ≥ 31 時, 表示 led 全部暗, 即進入 Expanding Mode
- 在 Expanding Mode, 一開始 cnt 設為 1, 表示中間的兩個燈亮。cnt 增加時, 燈就往外擴展, 擴展到全亮時, 即回到 Flashing Mode
- 每個模式都有一個 combinational logic 輸入 cnt 計算 led 的樣子
- 在輸出時透過 MUX 切換 led 接上與目前 state 相對應的 combinational logic



o



o

- Lab 3_3

- 因為還有其他事情要忙，所以沒有實作 QQ
- 我想會遇到問題的地方應該是因為一個 signal 不能有兩個 driver，所以不能直接拿兩個不同速度的 clock 控制 LED
- 所以需要拿兩個 clock 的最小公倍數來 trigger，因為兩個 clock 都是 2 的冪次，所以拿快的那個 clock 來當 trigger
- 但是我們還需要將慢的 clock 處理成 pulse，不然會認為慢的 clock trigger 好幾次
- 可以用 CT Verilog Series 教的 level to pulse converter

2. 學到的東西與遇到的困難

- 學會實際將我們設計的電路燒進板子，實現真正的硬體電路
- 瞭解模擬和實際 synthesis 出來的電路有很大的區別
 - 很多可以模擬的邏輯都不能變成電路
- 學會製作 clock divider
- 學會將速度快的振盪器除頻到肉眼看得到 led 閃爍
- 學會用 led 來 debug
- 學會使用 generate 語法來 assign 多個 wire
- 一開始常常 multiple drivers 的問題，之後慢慢學會解決這種問題
 - 只要用硬體的觀念來寫，就幾乎不會出現這種問題了
- 發現如果修改了 source，不用慢慢等 Run Synthesis -> Run Implementation -> Generate Bitstream，可以直接按 Generate Bitstream，會自動做完前面的事情，這大概總共省了我好幾個小時的時間吧

3. 想對老師或助教說的話

- 謝謝老師和助教～
- 希望有蒐集到好笑的笑話可以在課堂上分享 :)