**期末專案 – 數位邏輯電路模擬器(Logic Simulator)**

1. **前言**

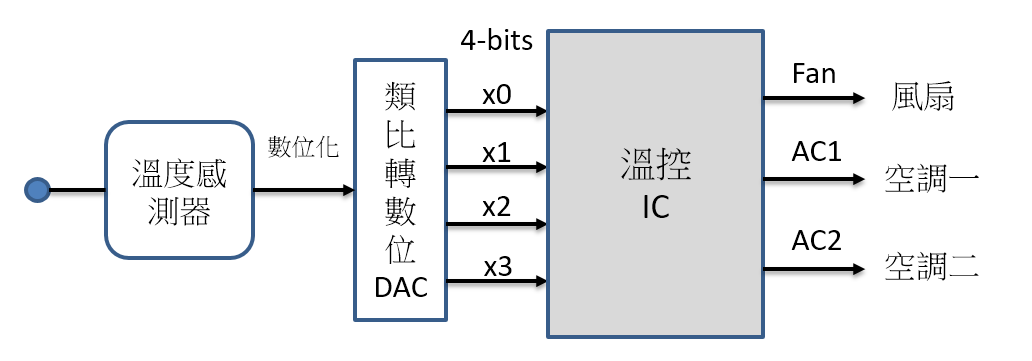
IC設計大致可分為類比電路設計與數位電路設計二種類型，當客戶指定規格(specification)後，便可交由設計工程師來設計。但由於IC製作成本高昂，在量產之前，需經嚴格的驗證，以確定設計結果(電路)是否符合客戶定的規格。

* 範例: 室溫控制IC設計

<規格表>

|  |  |  |  |
| --- | --- | --- | --- |
| 氣溫 | (天花板)風扇 | 空調1號機 | 空調2號機 |
| <=25 | X | X | X |
| 26~28 | V | X | X |
| 29~33 | V | V | X |
| 34~ | V | V | V |

硬體方塊圖



溫控IC之數位邏輯電路設計

1. 使用4-bits(x1,x2,x3,x4)表示溫度:

0000: <=20度, 0001: 21度, …., 0110: 26度, …, 1001:29度, …, 1110:34度, 1111: >=35度

1. 建立真值表(truth table)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 溫度 | X0 | X1 | X2 | X3 | Fan | AC1 | AC2 |
| 20 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 21 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 22 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 23 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 24 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 25 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 26 | 0 | 1 | 1 | 0 | 1 | 0 | 0 |
| 27 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| 28 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 29 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 30 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |
| 31 | 1 | 0 | 1 | 1 | 1 | 1 | 0 |
| 32 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 33 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| 34 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 35 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

1. 利用真值表產生數位邏輯電路

- 人工設計或自動合成

|  |  |  |
| --- | --- | --- |
| 邏輯運算式 | 邏輯電路文字描述 | 邏輯電路圖 |
| FAN=x0+x1\*x2  AC1=x0\*(x1+x3)  AC2=x0\*x1\*x2 | g4=AND(g1,g2)  g5=OR(g1,g3)  g6=OR(g0,g4)  g7=AND(g0,g5)  g8=AND(g0,g4) |  |

1. 邏輯驗證 (Logic verification): 重要!

- 電路的行為是否與規格相同 🡪 使用數位電路模擬(Logic Simulation)

|  |  |  |
| --- | --- | --- |
| 測試資料(g0,g1,g2,g3) | 邏輯電路圖 | 輸出(g6,g7,g8) |
| 0000  0001  0010  0011  …  1111 |  | 0000 -> 000  0001 -> 000  0010 -> 000  0011 -> 000  …  1111 -> 111 |

1. **數位電路模擬器(Logic Simulator)**

* 是一種軟體，根據指定的輸入，在邏輯電路中進行布林運算，並產生輸出值，用途為驗證邏輯電路的正確性。

1. 數位邏輯電路的表示方式

<邏輯閘的種類> AND, NAND, OR, NOR, XOR, XNOR, BUFF

- 如g5 = nand(g1,g2,g3) 🡪 劃出電路圖

<輸入>: 可被視為一種沒有運算過程的邏輯閘，直接將值傳導過去

INPUT(g0)

INPUT(g1)

….

<輸出>: 指定那些邏輯閘的值就是整個電路的”輸出”，這些gate不做為其他邏輯閘的輸入

OUTPUT(g6)

OUTPUT(g7)

OUTPUT(g8)

|  |
| --- |
| 範例: 電路編號c17，檔名c17.bench.txt (畫出對應的電路圖) |
| # c17 iscas example (to test conversion program only)  # ---------------------------------------------------  #  # total number of lines in the netlist .............. 17  # simplistically reduced equivalent fault set size = 22  # lines from primary input gates ....... 5  # lines from primary output gates ....... 2  # lines from interior gate outputs ...... 4  # lines from \*\* 3 \*\* fanout stems ... 6  # avg\_fanin = 2.00, max\_fanin = 2  # avg\_fanout = 2.00, max\_fanout = 2  INPUT(G1gat)  INPUT(G2gat)  INPUT(G3gat)  INPUT(G6gat)  INPUT(G7gat)  OUTPUT(G22gat)  OUTPUT(G23gat)  G10gat = nand(G1gat, G3gat)  G11gat = nand(G3gat, G6gat)  G16gat = nand(G2gat, G11gat)  G19gat = nand(G11gat, G7gat)  G22gat = nand(G10gat, G16gat)  G23gat = nand(G16gat, G19gat) |

[注意]: 電路中邏輯閘的編號，不見得能被用來做為判斷訊號傳遞的先後次序。

1. 產生測試資料

<測試檔案(輸入)格式> c17有5個輸入(G1gate,G2gat,G3gat, G6gat,G7gat)

|  |  |
| --- | --- |
| 完整測試檔案(c17\_all\_ip.txt) | 隨機測試檔案(c17\_4\_ip.txt)只抽選某些輸入) |
| 00000  00001  00010  ….  11111 | 00010  10101  11010  00101 |

Q: 為何使用隨機測試檔? 考量大型電路，如c432有36個輸入…

1. 以測試檔案為輸入，進行軟體模擬(使用Logic Simulator)，產生輸出。

<輸出檔案格式>: c17有5個輸入(G1gate,G2gat,G3gat, G6gat,G7gat)、2個輸出(G22gat, G23.gat)

|  |
| --- |
| c17\_all\_ip.txt 🡪 c17\_all\_op.txt |
| 00000 00  00001 10  00010 01  ….  11111 10 |

1. 驗證正確性: 與正確答案比對 (注意: 以下輸出為舉例用，並非正確答案)

|  |  |
| --- | --- |
| c17\_all\_op.txt(跑出來的結果) | c17\_all\_ans.txt(答案) |
| 00000 00  00001 10  00010 01  ….  11111 10 | 00000 00  00001 10  00010 11  ….  11111 10 |

Q: 若有誤，可能原因為?

1. 驗收: 預定最後一週上課時間驗收
2. 使用ISCAS85電路集(benchmark)進行測試:

- C17, C432, C880, C1355, C1980, C2670, C3540, C5315, C6288, C7752

- 電路將公布在iClass。

1. 除C17使用完整輸入(c17\_all\_ip.txt)進行測試外，其餘電路均使用隨機產生的輸入資料

- 如C432\_1K\_ip.txt, C432\_10K\_ip.txt, C432\_50K\_ip.txt… (老師提供)

- 如C432\_1K\_ans.txt, C432\_1K\_ans.txt, C432\_50K\_ans.txt… (老師提供)

驗收重點

1. 正確性
2. 執行速度
3. 可擴展性(Scalability):

- 能執行很大的電路，不會很快資源不足(如RAM)。

- 執行時間不會因電路變大，而快速大幅增加。

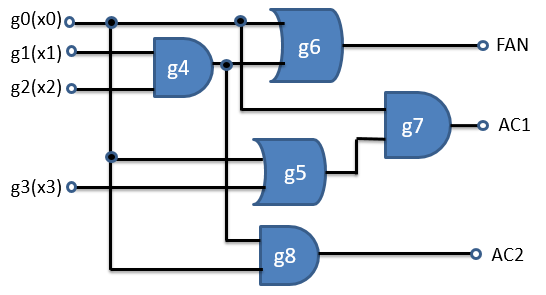
1. **軟體設計**
2. 資料結構設計: 讀取電路描述檔後，如何儲存，以備後續的運算?

|  |
| --- |
| c17.bench.txt |
| INPUT(G1gat)  INPUT(G2gat)  INPUT(G3gat)  INPUT(G6gat)  INPUT(G7gat)  OUTPUT(G22gat)  OUTPUT(G23gat)  G10gat = nand(G1gat, G3gat)  G11gat = nand(G3gat, G6gat)  G16gat = nand(G2gat, G11gat)  G19gat = nand(G11gat, G7gat)  G22gat = nand(G10gat, G16gat)  G23gat = nand(G16gat, G19gat) |

1. 演算法設計
2. 如何根據輸入，產生電路的正確輸出? 先從單一logic gate開始

AND: OR: NAND: NOR: XOR:

1. 如何加速運算? (0000), (0001), (0010), (0011) …



1. 如何節省記憶體