

## 1. 背景

這學期的計算機組織為先前數位邏輯設計的課程內容之延伸，計組課程以 MIPS 處理器的運作模式、電路設計架構為主要方向，同時帶入了以硬體描述語言 Verilog 來實作邏輯電路設計的教學內容。

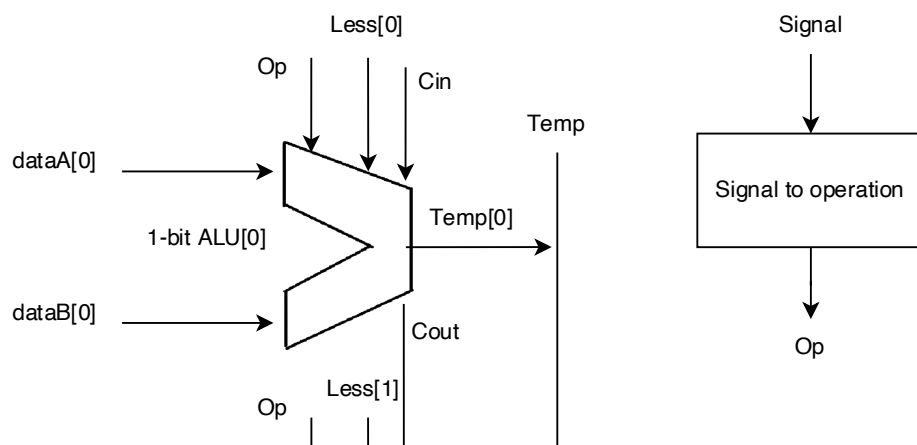
本次作業的目的為實作一個具有加減法、乘法、邏輯運算跟移位功能，且可以選擇功能的算數邏輯器(ALU)。結合先前數位電路的訊號輸入/輸出與邏輯閘等概念，依照本次作業要求編寫出數種功能不一樣的組合邏輯與時序邏輯之電路元件，再將這些元件依照彼此之間的輸入/輸出之關係進行拼接，合而為一，形成功能完整的 ALU。

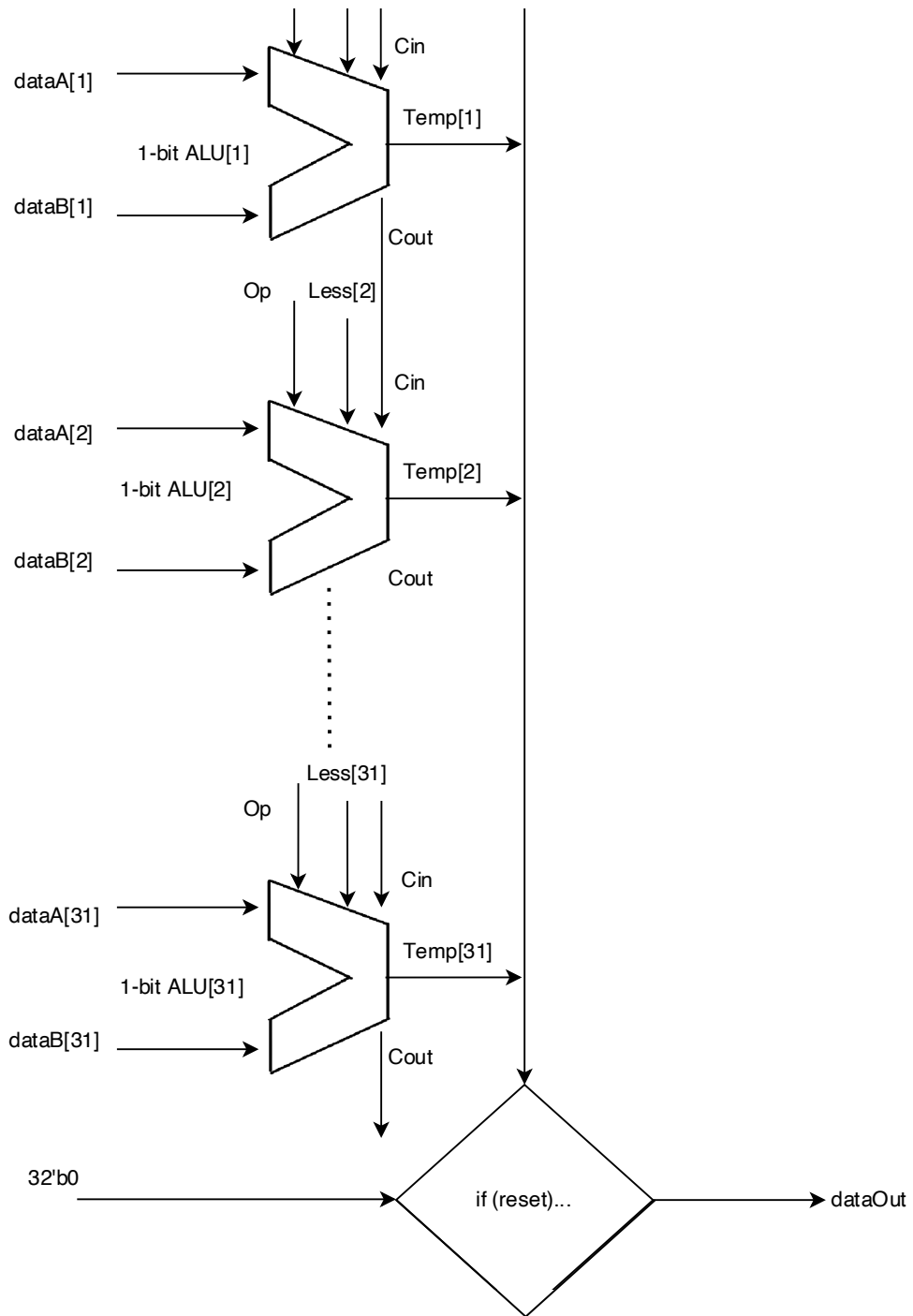
## 2. 方法

設計重點說明：

- ALU 的全加法器須確實寫出，不可直接用+，架構也應為 32 個 1-Bit ALU 連接起來，資料要單個位元分開處理。
- 以時脈週期的累加去做乘法器的計算。
- 160 個多工器並以 dataB 的五個位元做判斷的移位器。
- 將 reg 改 wire 解決不能使用 always@ ( ) 的狀況。
- Mux 在時脈下沿觸發，以避免值延後輸出。

ALU 詳細架構：





### 3. 結果

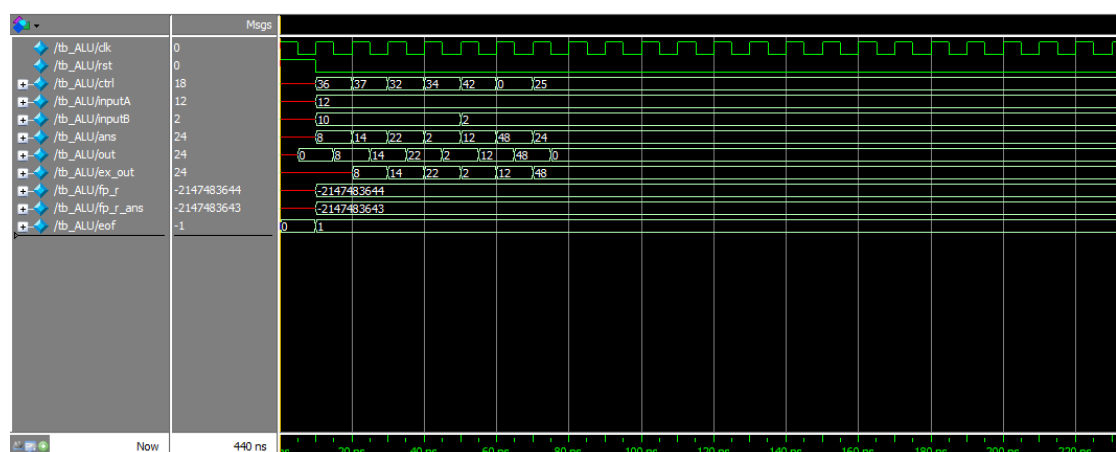
模擬結果：

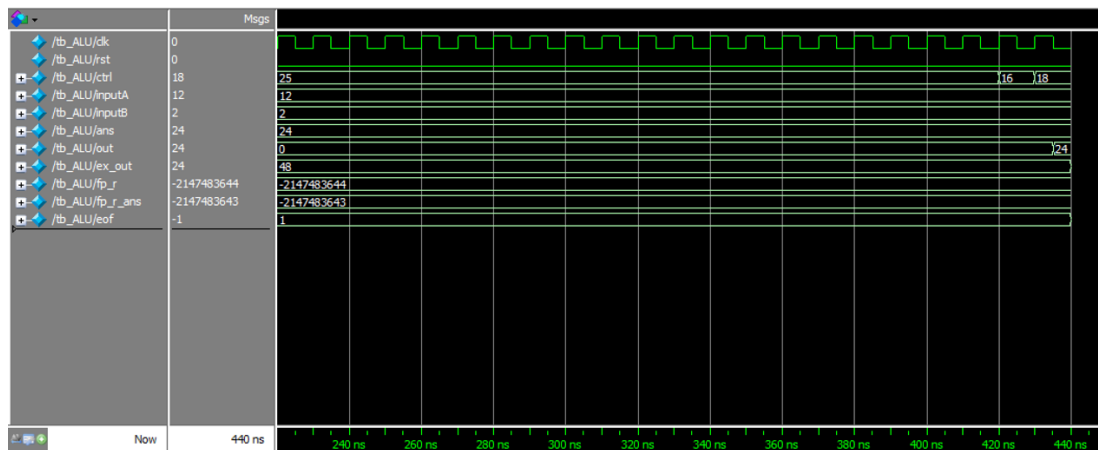
```

add wave -position insertpoint sim:/tb_ALU/*
VSIM 44> run -all
# Start
#
#           1: Input: AND(36)           12           10
#           2: Correct: Your answer is:      8,
#               Correct answer is:                8
#
#           2: Input: OR(37)            12           10
#           3: Correct: Your answer is:    14,
#               Correct answer is:                14
#
#           3: Input: ADD(32)            12           10
#           4: Correct: Your answer is:    22,
#               Correct answer is:                22
#
#           4: Input: SUB(34)            12           10
#           5: Correct: Your answer is:      2,
#               Correct answer is:                2
#
#           5: Input: SLT(42)            12           2
#           6: Correct: Your answer is:    12,
#               Correct answer is:                12
#
#           6: Input: SLL( 0)            12           2
#           7: Correct: Your answer is:    48,
#               Correct answer is:                48
#
#           7: Input: MULTU(25)          12           2
#           40: Mul End
#
#           44: Correct: Your answer is:    24,
#               Correct answer is:                24
#
# Simulation End
#
# Break in Module tb_ALU at //Mac/Home/Desktop/CO midterm/alu/tb_ALU.v line 68

```

Waveform :





#### 4. 討論

設計上的挑戰：

作業上對 Verilog 程式的編寫方式有特別規範，因此會需要將老師原本提供的範本給大幅修改，寫出的成品在功能上會與範例原版大同小異，但實際上架構相差甚遠，為了思考出新的編寫方式、架構排版，就會耗費而外的心力來完成。

舉例來說，Module ALU 原版中的架構是 behavior modeling，而且 A 與 B 直接使用運算元來求取相對應的輸出值；作業則需要運用 gate level 與 dataflow modeling，電路的結構與設計上有更多細節，資料傳輸、運算的過程也會步驟更多。

程式驗證&修正：

除了運用編譯器做為第一關的程式碼錯誤檢查，也檢查模擬的執行結果。檢視波型圖與 transcript 上的執行輸出訊息，以此評估是否有其他編譯器無法指出的邏輯錯誤，或是元件數值沒有正常傳輸到另一端的錯誤。

#### 5. 結論

此次作業總共使用了 10 種 module，每個元件該具有的功能必須都齊全，除此之外題目、講義沒提到的部分也不該加，規範之所以限制編寫架構與方式，是為了能強調移位器、ALU 等模塊的基礎組成元件、結構設計，另外也可以訓練我們以不同的電路描述方式，實作出功能相同的處理器元件。