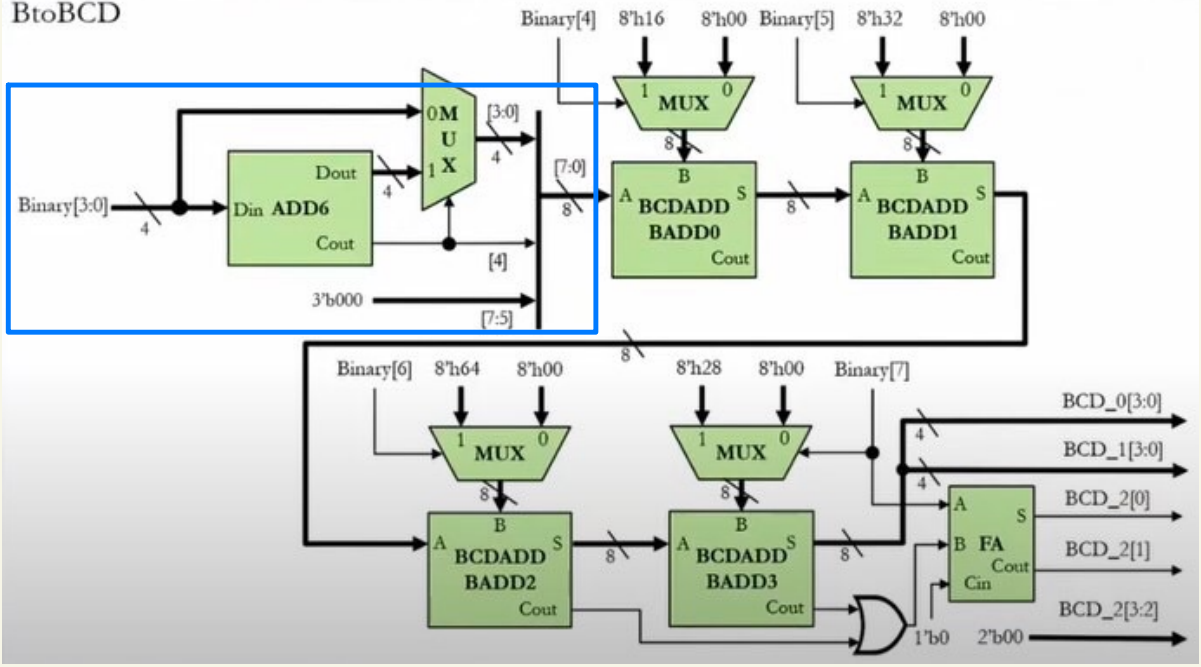
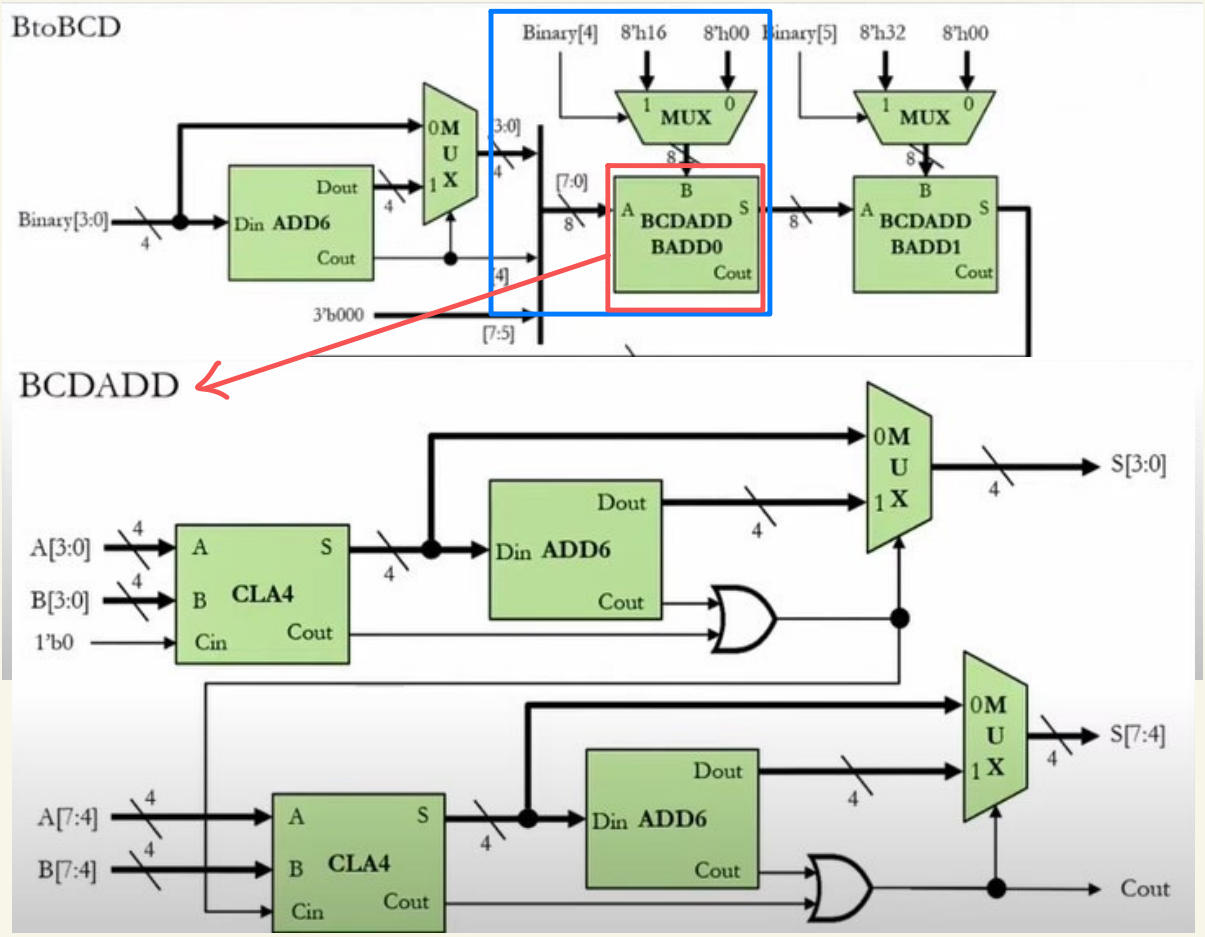
Binary to BCD code

此練習是 8-bit binary number convert to BCD code，∴分成 Binary [3:0]（個位數）和 Binary [7:4]（十位數）處理，∵8-bit 最大值為255，∴還要處理百位數，

最後BCD結果分成 BCD_0，BCD_1，BCD_2 分別代表個，十，百位數值



先處理個位數(Binary[3:0])，因為BCD code只能表示0~9，但個位數可能會有A~F，所以要檢查，若個位數+6後有進位(Cout==1)，則個位數結果取+6過後的值，若沒有進位(Cout==0)，則用原本的值，再把個位數擴充成8-bit，以便後續處理十位數。



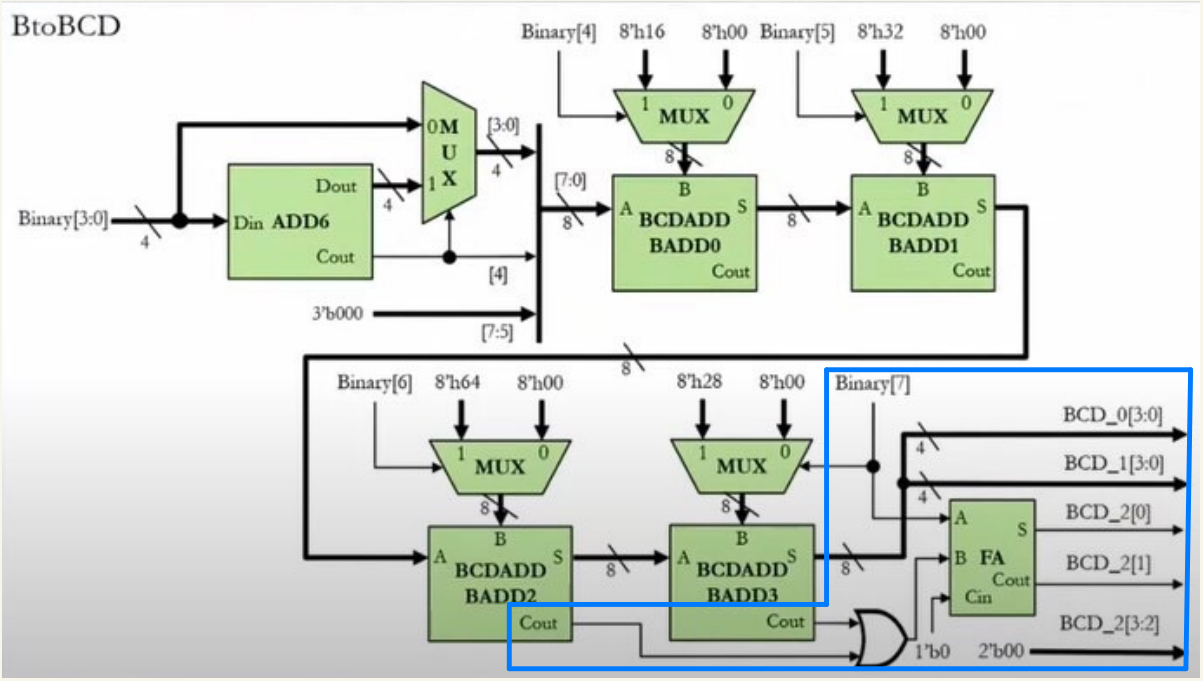
再來將十位數(Binary[7:4])考慮進來，若要計算十位數的話，需要把Binary[7:4]的值加總，ex: Binary[7:4] = 1001，所以是2^4 + 2^7 = 16 + 128 = 144。

而MUX上方的值就是在看Binary各個bit是否為1，assign相對應的值做加總的動作。

加總的邏輯在BCDADD，我們將個位數與十位數分別加總，加總的過程中我們也需要檢查是否需要+6，以維持BCD code的正確性。

將CLA4與ADD6的Cout做OR是因為，不管哪個Cout==1，都需要+6。

(註：要知道如果兩個Cout同時為1，則最後Cout會為0，而選擇沒有+6的結果，所以像處理個位數一樣只看ADD6的Cout是不行的。)



最後處理百位數，加總過程中，十位數的Cout就是百位數的值，而產生百位數的情況有兩種：

(1) 加總的過程，十位數進位(BCDADD的Cout==1)。

(2) Binary[7]為1，值為128。

而十位數進位只會發生在第三、四個BCDADD，所以將它們兩個的Cout做OR，再來把(1)和(2)的結果給FA加總，就能知道百位數的值為何。