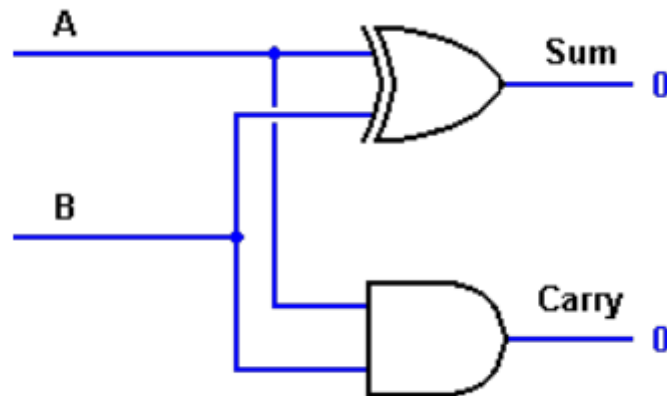


# Assignment 2: High Level Synthesize

105062600 Yi-cheng,Chao 05:37, June 6, 2017

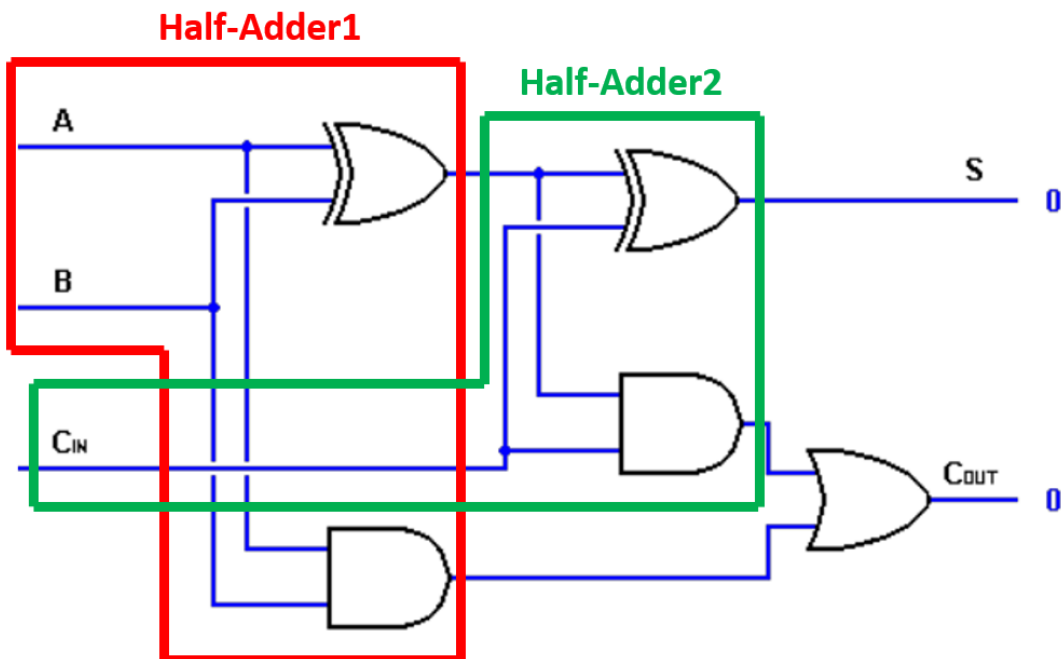
---

## Design1 Schematic: Half-Adder



基本上就是很基本的把 input/output ports 宣告好,並且用 member function 描述出 XOR/AND gate 的行為便可以完成。

## Design1 Schematic: Full-Adder



Full-Adder 的實作則是使用兩個 Half-Adder 連接並且再加上 Cout 端的 OR gate 行為便可以完成。

## Design1 Testbench

```
while(round--){  
  
    for(counter = 0; counter < 9; counter++){  
        t_F_Cin = counter[0].to_bool();  
        t_F_B = counter[1].to_bool();  
        t_F_A = counter[2].to_bool();  
        sc_start(sc_time(10, SC_NS));  
        cout << "[Full Adder] At time " << sc_time_stamp() << "::";  
        cout << "(A, B, Cin): ";  
        cout << t_F_A << t_F_B << t_F_Cin;  
        cout << " (Cout, Sum): ";  
        cout << t_F_Cout << t_F_S << endl;  
    }  
}  
cout << "Info: End of full adder simulation...\n";  
round = 4;  
cout << "Info: Begin half adder simulation...\n";  
while(round--){  
    for(counter = 0; counter < 5; counter++){  
        t_H_B = counter[0].to_bool();  
        t_H_A = counter[1].to_bool();  
        sc_start(sc_time(10, SC_NS));  
        cout << "[Half Adder] At time " << sc_time_stamp() << "::";  
        cout << "(A, B): ";  
        cout << t_H_A << t_H_B;  
        cout << " (Carry, Sum): ";  
        cout << t_H_Carry << t_H_Sum << endl;  
    }  
}  
cout << "Info: End of half adder simulation...\n";
```

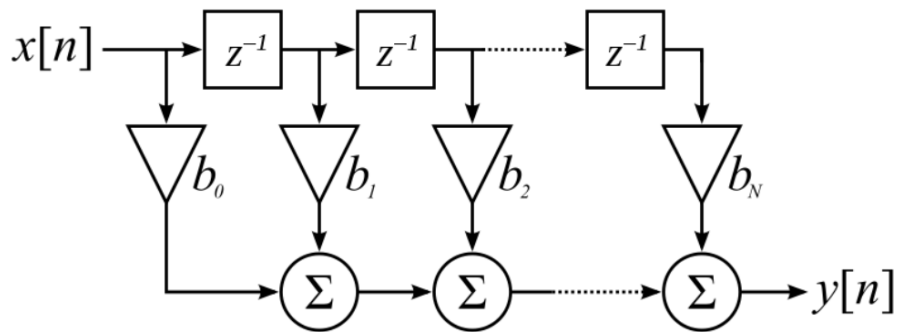
使用討論區討論的 test pattern generate 方法，使用兩輪的 for loop 分別對 Full-Adder 以及 Half-Adder 做 pattern injection。

## Design1 Simulation Result

模擬結果以 Half-Adder 模擬為例，如下圖所示：

```
Info: Begin half adder simulation...  
[Half Adder] At time 370 ns::(A, B): 00 (Carry, Sum): 00  
[Half Adder] At time 380 ns::(A, B): 01 (Carry, Sum): 01  
[Half Adder] At time 390 ns::(A, B): 10 (Carry, Sum): 01  
[Half Adder] At time 400 ns::(A, B): 11 (Carry, Sum): 10  
[Half Adder] At time 410 ns::(A, B): 00 (Carry, Sum): 00  
[Half Adder] At time 420 ns::(A, B): 00 (Carry, Sum): 00  
[Half Adder] At time 430 ns::(A, B): 01 (Carry, Sum): 01  
[Half Adder] At time 440 ns::(A, B): 10 (Carry, Sum): 01  
[Half Adder] At time 450 ns::(A, B): 11 (Carry, Sum): 10  
[Half Adder] At time 460 ns::(A, B): 00 (Carry, Sum): 00  
[Half Adder] At time 470 ns::(A, B): 00 (Carry, Sum): 00  
[Half Adder] At time 480 ns::(A, B): 01 (Carry, Sum): 01  
[Half Adder] At time 490 ns::(A, B): 10 (Carry, Sum): 01  
[Half Adder] At time 500 ns::(A, B): 11 (Carry, Sum): 10  
[Half Adder] At time 510 ns::(A, B): 00 (Carry, Sum): 00  
[Half Adder] At time 520 ns::(A, B): 00 (Carry, Sum): 00  
[Half Adder] At time 530 ns::(A, B): 01 (Carry, Sum): 01  
[Half Adder] At time 540 ns::(A, B): 10 (Carry, Sum): 01  
[Half Adder] At time 550 ns::(A, B): 11 (Carry, Sum): 10  
[Half Adder] At time 560 ns::(A, B): 00 (Carry, Sum): 00  
Info: End of half adder simulation...
```

## Design2 Schematic: 16<sup>th</sup>-order FIR filter



只要使用一個 MA\_filter module，使用 16 個 32 bits Register 便可以完成緩衝區的實作，因為 spec 中直接使用 0x0000F0F 來進行運算，因此最後再結果部分 Y 值要除以  $2^{16}$  來進行還原。

## Design2 Testbench

因為有提供 test pattern，因此我在模擬部分多設計一個 parser 來進行 input file 的讀檔，再把資料送進 MA filter 內，程式碼如下所示：

```
void parser::prc_parser() {
    unsigned int z;
    if(inFile >> hex >> z){
        // cout << "parser File success\n";
    }
    else{
        cout << "Warning: End of File \n";
        z = 0;
    }
    x = z;
    output.write(x);
    y = input.read();
    cout << "At time "<< sc_time_stamp() << " ::X: " << x << " Y: " << (y >> 16) << endl;
    return;
}
```

## Design2 Simulation Result

```
At time 860 ns::X: 0 Y: 15420
At time 880 ns::X: 0 Y: 11565
At time 900 ns::X: 0 Y: 7710
At time 920 ns::X: 4294967295 Y: 3855
At time 940 ns::X: 4294967295 Y: 0
At time 960 ns::X: 4294967295 Y: 65535
At time 980 ns::X: 4294967295 Y: 65535
At time 1 us::X: 4294967295 Y: 65535
At time 1020 ns::X: 4294967295 Y: 65535
At time 1040 ns::X: 4294967295 Y: 65535
At time 1060 ns::X: 4294967295 Y: 65535
At time 1080 ns::X: 4294967295 Y: 65535
At time 1100 ns::X: 4294967295 Y: 65535
At time 1120 ns::X: 4294967295 Y: 65535
At time 1140 ns::X: 4294967295 Y: 65535
At time 1160 ns::X: 4294967295 Y: 65535
At time 1180 ns::X: 4294967295 Y: 65535
At time 1200 ns::X: 4294967295 Y: 65535
At time 1220 ns::X: 4294967295 Y: 65535
At time 1240 ns::X: 4294967295 Y: 65535
Warning: End of File
At time 1260 ns::X: 0 Y: 65535
```

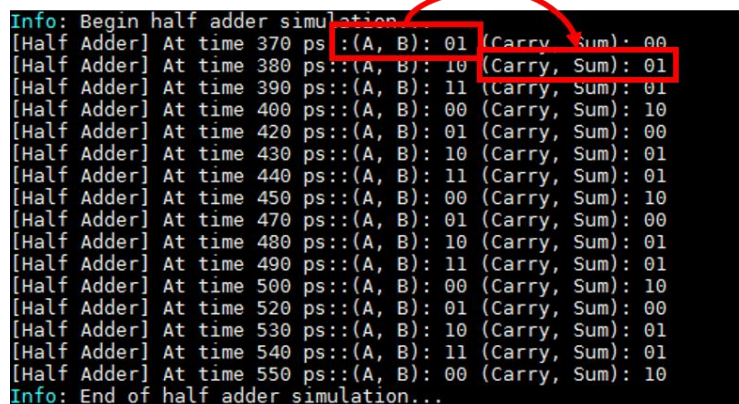
## Difficulty Encounter & Experience

原本 Design1 有使用 monitor 的 module 來進行接收數據並且顯示，如下所示：

```
// full_monitor.h
#include "systemc.h"
#ifdef _FULL_MONITOR_H
#define _FULL_MONITOR_H
SC_MODULE(full_monitor){
    sc_in<bool> t_F_A, t_F_B, t_F_Cin, t_F_S, t_F_Cout; // full adder signal
    void prc_monitor(){
        cout << "[Full Adder] At time " << sc_time_stamp() << " :: ";
        cout << "(A, B, Cin): ";
        cout << t_F_A << t_F_B << t_F_Cin;
        cout << " (Cout, Sum): ";
        cout << t_F_Cout << t_F_S << endl;
    }
    // Constructor
    SC_CTOR(full_monitor){
        SC_METHOD(prc_monitor);
        sensitive << t_F_A << t_F_B << t_F_Cin;
    }
};
```

但發現在 Simulation 上都會有延遲一個 clock 的情況發生，如下圖所示：

**Input port與output port會有訊號延遲一個cycle對上的情形發生**



```
Info: Begin half adder simulation...
[Half Adder] At time 370 ps: (A, B): 01 (Carry, Sum): 00
[Half Adder] At time 380 ps: (A, B): 10 (Carry, Sum): 01
[Half Adder] At time 390 ps: (A, B): 11 (Carry, Sum): 01
[Half Adder] At time 400 ps: (A, B): 00 (Carry, Sum): 10
[Half Adder] At time 420 ps: (A, B): 01 (Carry, Sum): 00
[Half Adder] At time 430 ps: (A, B): 10 (Carry, Sum): 01
[Half Adder] At time 440 ps: (A, B): 11 (Carry, Sum): 01
[Half Adder] At time 450 ps: (A, B): 00 (Carry, Sum): 10
[Half Adder] At time 470 ps: (A, B): 01 (Carry, Sum): 00
[Half Adder] At time 480 ps: (A, B): 10 (Carry, Sum): 01
[Half Adder] At time 490 ps: (A, B): 11 (Carry, Sum): 01
[Half Adder] At time 500 ps: (A, B): 00 (Carry, Sum): 10
[Half Adder] At time 520 ps: (A, B): 01 (Carry, Sum): 00
[Half Adder] At time 530 ps: (A, B): 10 (Carry, Sum): 01
[Half Adder] At time 540 ps: (A, B): 11 (Carry, Sum): 01
[Half Adder] At time 550 ps: (A, B): 00 (Carry, Sum): 10
Info: End of half adder simulation...
```

但由於只是單純的接上訊號顯示，也因為礙於時間便沒有進行 Debug 來修正，最後直接在 main.cpp 上來進行顯示，也算是 systemC 強大的地方—可以使用不像是硬體語言的模組化區塊而是使用比較偏軟體的寫法來進行表現。

此外在討論區討論到的 SC\_MODULE\_EXPORT 問題也多虧同學的即早提問，讓我在 CTOS 合成中能立刻解決。



Maybe it can help:

Add following lines to FA.cpp

```
#ifdef __CTOS__
SC_MODULE_EXPORT(FA);
#endif
```