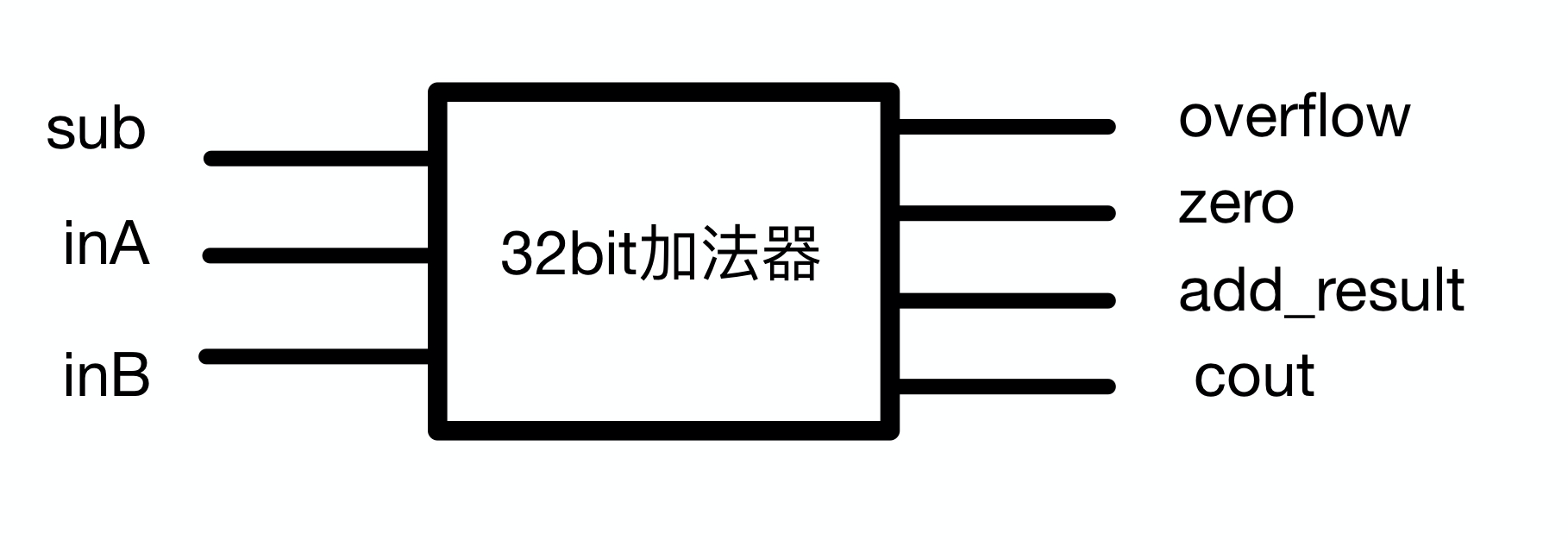
32位超前进位加法器

1. 功能特性

（1）实现有符号、无符号数的加减法。

（2）具有溢出标志（有符号数），进/借位标志（无符号数），零标志位

2.输入输出端口说明

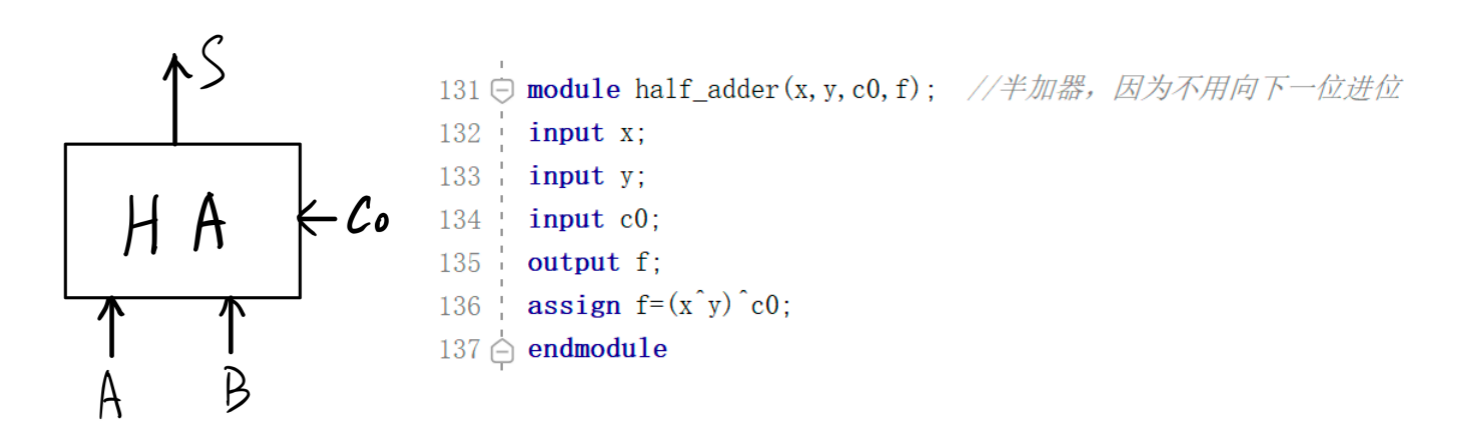


|  |  |  |
| --- | --- | --- |
| 名称 | 类型 | 描述 |
| inA | 输入/wire | 32bit/操作数0 |
| inB | 输入/wire | 32bit/操作数1 |
| sub | 输入/wire | 1bit/减法标志位 |
| overflow | 输出/wire | 1bit/溢出标志位 |
| zero | 输出/wire | 1bit/置零标志位 |
| cout | 输出/reg | 1bit/进位标志位 |
| add\_result | 输出/wire | 32bit/运算结果 |

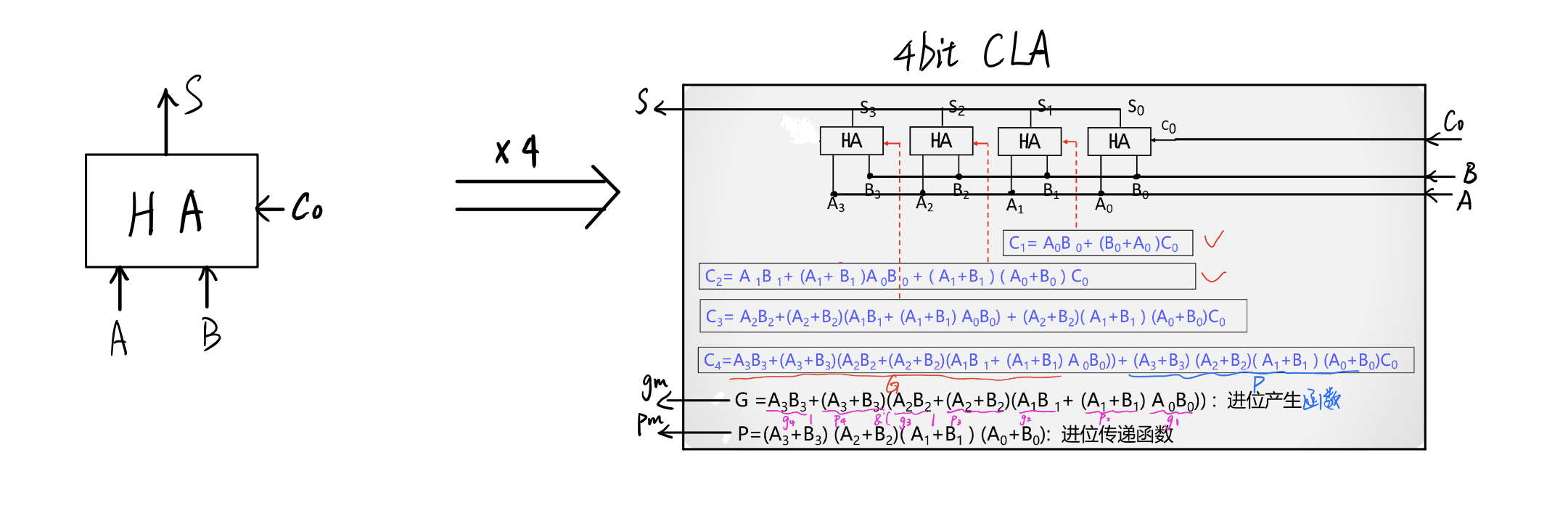
1. 设计思路

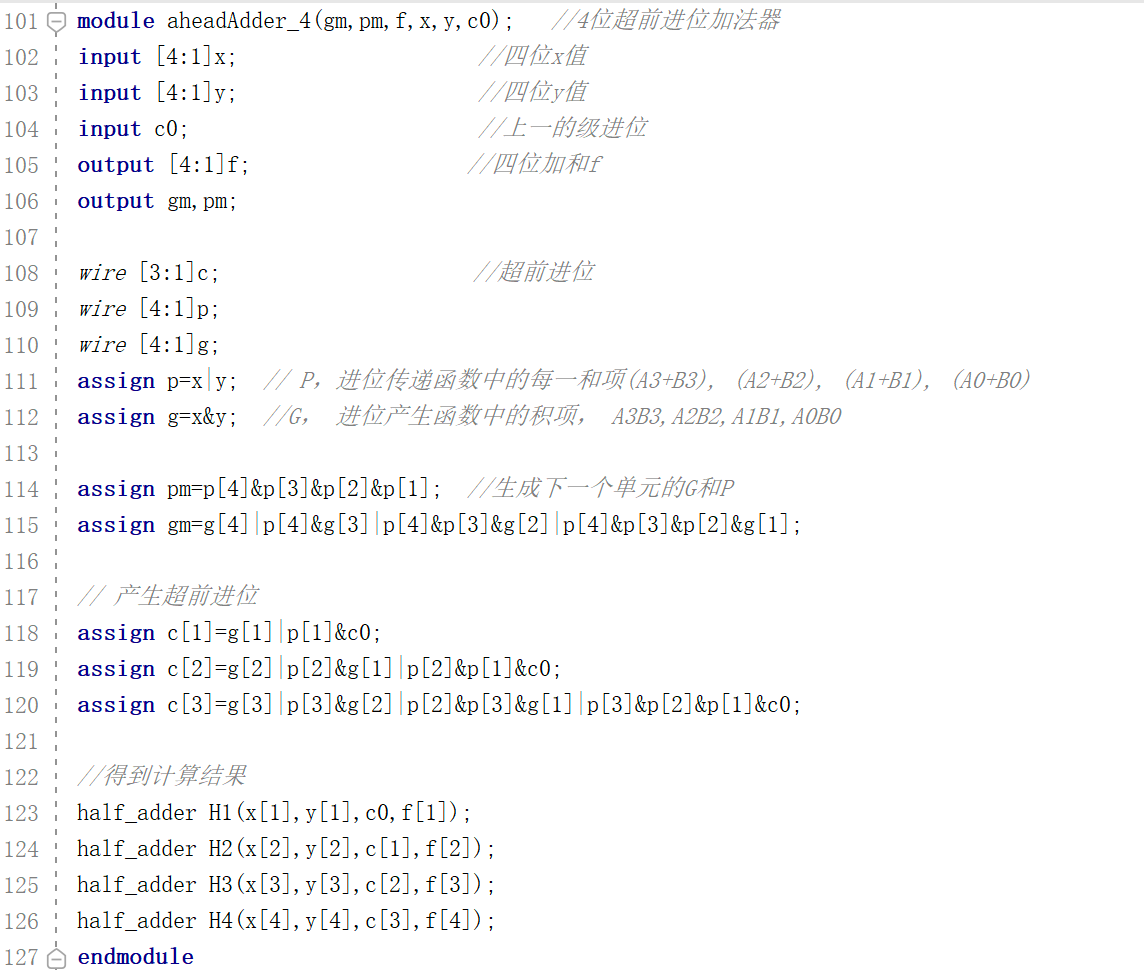
我们采取了自底向上的思路：

1. 我们先通过门级实现了半加器（因为每一位的进位会先行产生，所以每一位的加法不需要向下一级进位，用半加器就可以了）

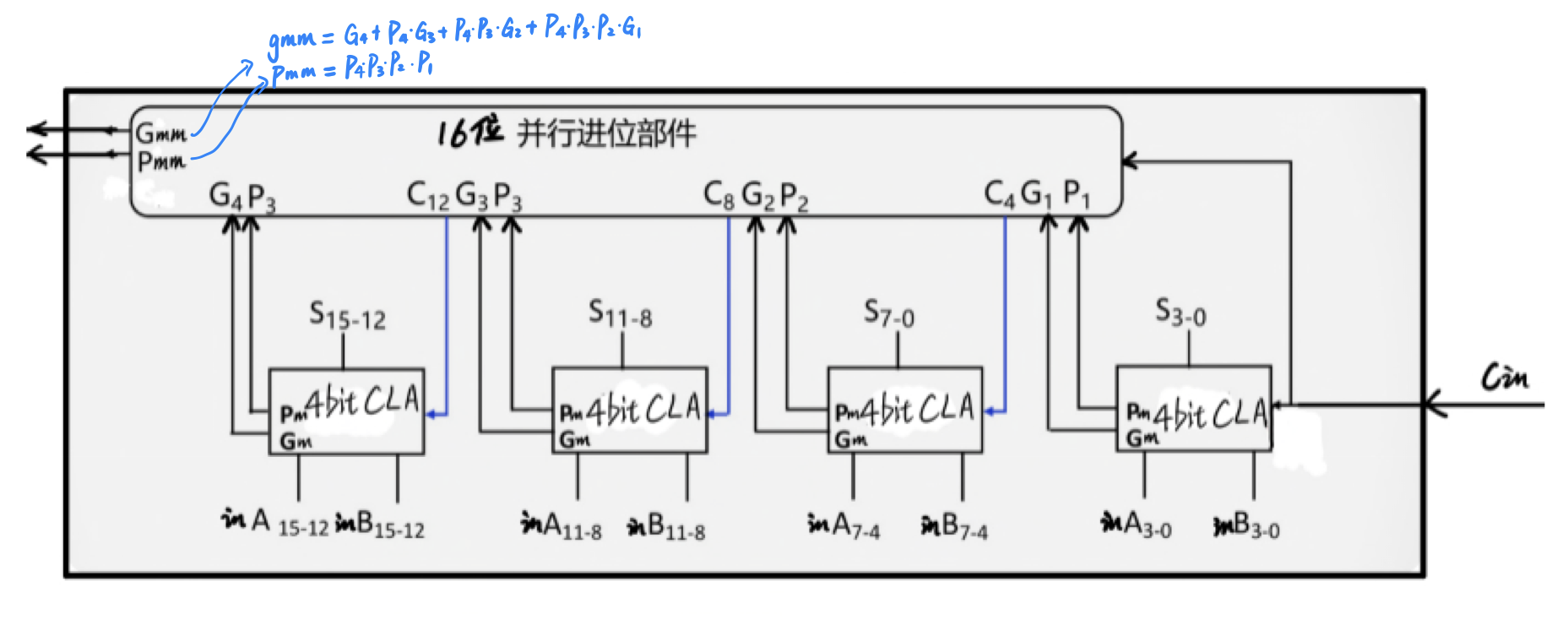


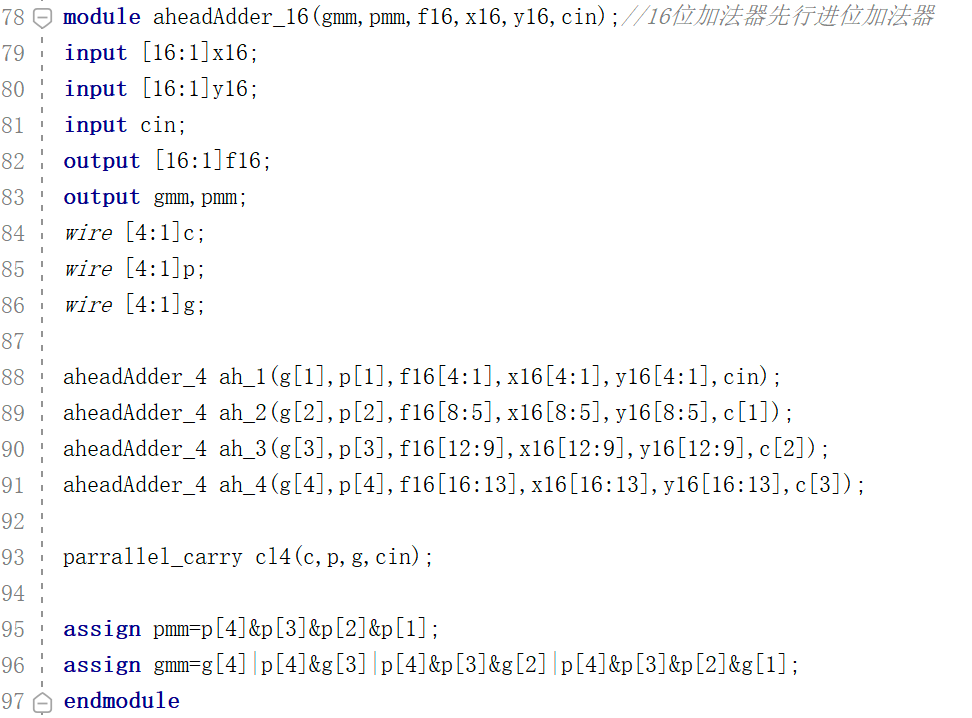
（2）随后，将4个半加器以及超前进位产生函数G和进位传递函数P组合，形成了一个4位超前进位运算器，结构如下：



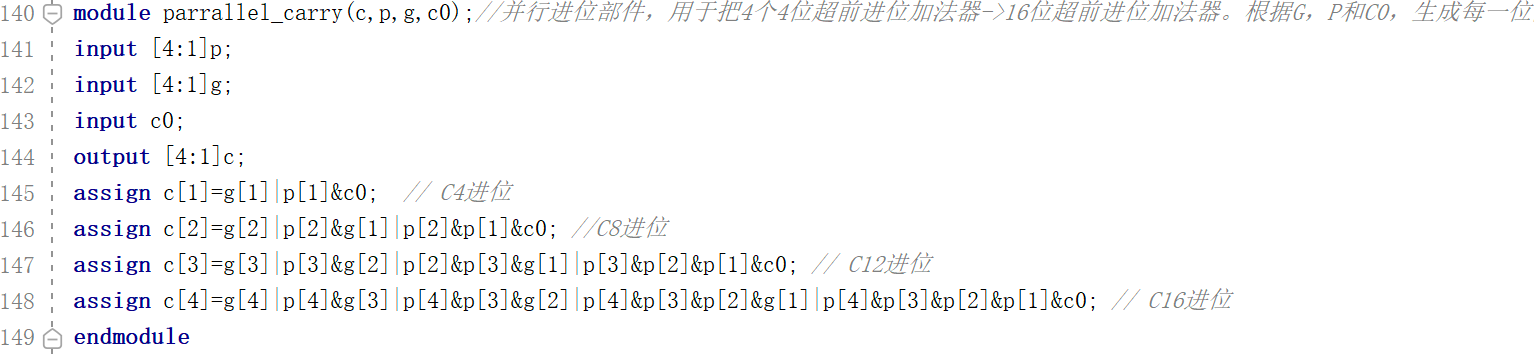


（3）随后，将4个超前进位运算器和16位并行进位部件组合，构造了16位并行运算单元，结构如下：

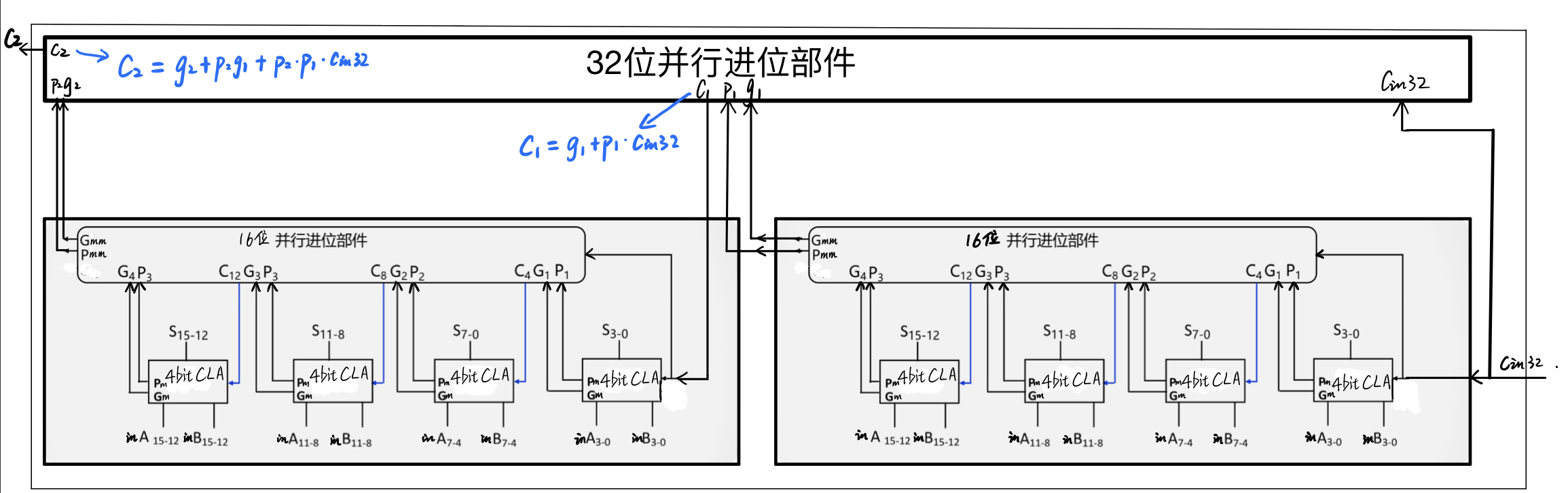


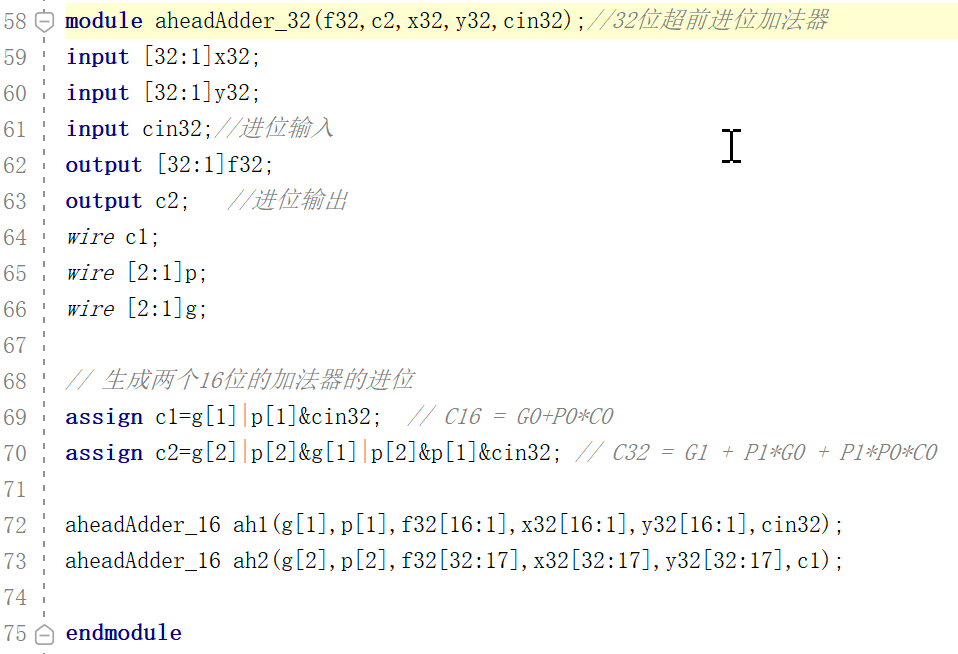


16位并行进位部件：



1. 将两个16位的超前进位加法器和32位并行进位部件组合，构成32位超前进位加法器，结构如下：

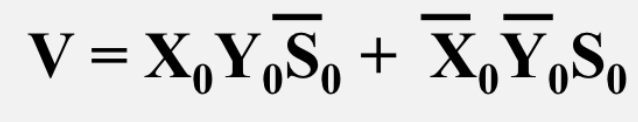




1. 最后，我们在32位超前进位加法电路的基础上，加入了求补运算，判断零标志，溢出标志，进/借位标志的语句，最终形成了符合需求的32位加法运算模块：

1>求补运算实现：当sub为1时做减法，需要求出减数的补码，即取反再加上1（sub输入最低位进位）

2>零标志：将结果的每一位相与，最后取反。

3>溢出标志：对于有符号运算，使用判断是否溢出，X0Y0分别是两个输入的最低位，S0是计算结果的最低位。

4>进/借位标志：对于无符号运算，做加法（sub=0）时，加法器的进位即为进位标志；做减法（sub=1）时，加法的进位取反即为借位标志。

