## 实验一 运算部件实验：加法器

1. 实验目的

1. 熟悉用Quartus II进行逻辑电路设计的方法。

2. 通过8位加法器的设计，掌握行波进位加法器的基本原理。

1. 实验原理

计算机能够进行各种信息处理，其中最常用的是各种算术运算，而加法是最基本的算术运算。下面绕加法器的设计进行讨论。

* 1. 半加器

所谓半加器是指对两个二进制1 位数相加，而不考虑低位进位的加法器。

1. 其逻辑符号如图1.1 所示，其中，Ai 和Bi 分别为两个二进制数的i 位值；Si 为Ai 和Bi 相加之和，叫做本位和；Ci+1 是本位向高位的进位。

  
图1.1 半加器逻辑符号

1. 真值表如表1.1 所示。

|  |  |  |  |
| --- | --- | --- | --- |
| 输入 | | 输出 | |
| Ai | Bi | Si | Ci+1 |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |



图1.2 半加器的电路图

1. 逻辑表达式。

由真值表可写出半加器的逻辑表达式：

Si = Ai Bi + AiBi = Ai ⊕ Bi

Ci+1 = AiBi

由逻辑表达式可得出半加器的电路图，如图1.2 所示。

* 1. 全加器



图1.3 全加器逻辑符号

能对两个1 位二进制数相加并考虑低位来的进位（即相当于3 个1 位二进制数的相加）得到“和”及“进位”的逻辑电路，称之为全加器，它的框图如图1.3 所示，其中Ai 和Bi 分别为两个1 位二进制数的输入；Ci-1 为低位来的进位输入；Si 和Ci 分别为相加后形成的“和”及向高位的“进位”输出。

全加器的真值表如表1.2 所示：

表1.2 全加器的真值表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 输入 | | | 输出 | |
| Ai | Bi | Ci | Ci+1 | Si |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

根据全加器的真值表，可得出全加器的逻辑表达式为

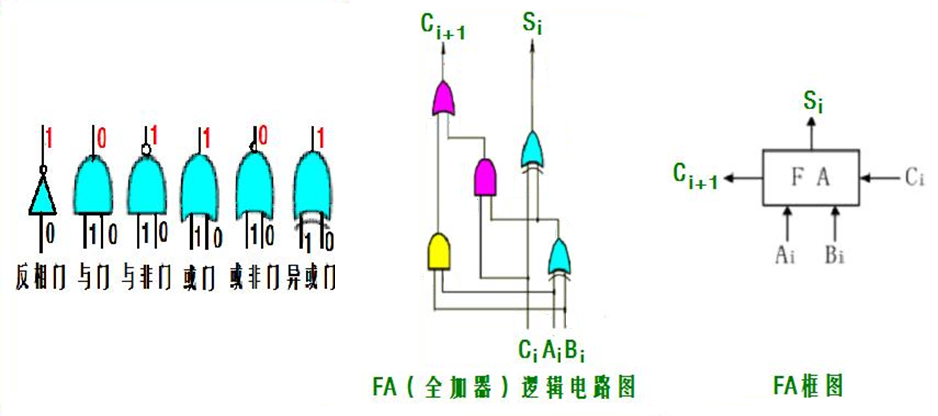
Si = Ai Bi Ci-1 + Ai Bi Ci-1 − + Ai Bi Ci-1 + Ai Bi Ci-1

Ci = AiBi + AiCi-1 + BiCi-1

全加器的逻辑电路图如图1.4 所示。



图1.4 （a） 用或非门实现全加器的逻辑图



1

&

≥1

=1

图1.4 （b） 一位全加器逻辑电路图

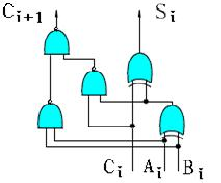


图1.4 （c）用与非门和异或门实现的一位全加器逻辑电路图

* 1. 多位加法器

如果有两个n位二进制数相加，就需n位全加器，这样构成的逻辑电路称为多位并行加法器。按照进位方式的不同，并行加法器分为行波进位加法器和先行进位加法器两种。

1）行波进位加法器

行波进位加法器的逻辑框图如图1.5所示。这种加法器的构成比较简单，只要把n位全加器串联起来，低位全加器的进位输出连到相邻的高位全加器的进位输入。

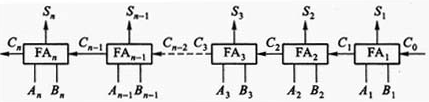


图1.5 行波进位加法器

由图1.5可见，这样构成的加法器，尽管各位相加是并行的，但其进位信号是由低位向高位逐级传递的，好像行波一样。这样，要形成高位的和，要等低位的进位形成后才能确定。正如做手算加法那样，要从低位逐级求出进位，最后才求得高位的和及进位。因此，这种加法器，由于进位是串行的，形成进位的速度很慢，加法器的速度主要受进位传递时间的限制。

若每级全加器形成进位的延时为2tpd，则在最坏情况下，从FA1的输入到产生高位进位Cn需要时间为2tpd×n。当 n增大时，完成一次加法所需时间也随之增加。

2）先行进位加法器

为了提高加法速度，在逻辑设计上采用所谓先行进位的方法，即每一位的进位根据各位的输入同时预先形成，而不需要等到低位的进位送来后才形成。

先行进位的原理是这样的：根据进位表达式Ci = AiBi + (AiBi)Ci-1可以知道，进位由两部分组成，AiBi表示当Ai=1且Bi=1时，进位Ci=1，它只与本位的输入Ai、Bi有关，与低位来的进位Ci-1无关；(AiBi)Ci-1表示当Ai、Bi中有一个为1时，当低位来进位信号，则Ci = 1。令

Gi = AiBi

Pi = AiBi

则 Ci = Gi + PiCi-1

称Gi为第i位的进位生成项；称Pi为进位传递条件。同时，可以把Si写成

Si = AiBiCi-1 = PiCi-1

由这些公式可以写出各位全加器的表达式，即有

S1 = P1C0

C1 = G1 + P1C0

S2 = P2C1

C2 = G2 + P2C1 = G2 + P2G1 + P2P1C0

S3 = P3C2

C3 = G3 + P3C2 = G3 + P3 G2 + P3 P2G1 + P3P2P1C0

S4 = P4C3

C4 = G4 + P4C3 = G4 + P4G3 + P4P3G2 + P4P3 P2G1 + P4P3P2P1C0

… …

Sn = PnCn-1

Cn = Gn + PnCn-1

= Gn + PnGn-1 + PnPn-1Gn-2 +… + PnPn-1…P2G1 + PnPn-1…P2P1C0

其中，C0为来自外部的进位输入；Gi = AiBi，Pi = AiBi（i = 1,2,…,n）是各位的进位生成项和进位传递条件，由各位的数据输入确定。

由这些表达式，可画出n位先行进位加法器的逻辑图。图3.6为4位先行进位加法器的逻辑图。从图可见，利用先行进位方法，各位进位都只经过三级门延时（约4tpd，假定异或门延时为2tpd），而形成各位和Si需经四级门延时（约6tpd），即先形成进位，再形成和。

显然，进位传递时间的节省是以逻辑电路的复杂性为代价的。随着位数的增加，所需电路元件也迅速增加，而且门电路的扇入和扇出数也会增大。扇入和扇出是反映门电路的输入端数目和输出驱动能力的指标。扇入是指一个门电路所能允许的输入端个数；扇出是指一个

门电路所能驱动的同类门的数目。因此，当位数较多时，为避免门电路的扇入、扇出增大的问题，通常采用折中的办法，即将字长n位分为若干组。例如，一个16位并行加法器，电路可分为4组，每组4位，每组内采用先行进位，组间采用串行进位。

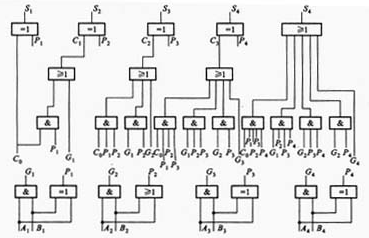
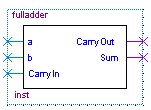


图1.6 4位先行进位加法器

4、全加器实现的8位行波进位加法器



本实验中，用8个全加器实现8位的串行波进位加法器。所用的全加器内部逻辑如全加器的两个基本公式，如图1.7所示。将8个全加器串连起来，也就是说低位全加器的进位输出连到相邻的高位全加器的进位输入，就构成了8位加法器，如图1.8所示，A＋B的和sum以及最后的进位carryout。

图1.7 实验中用到的1位全加器

Cy一

S6

……

B6

A6

FA

S7

C6

C7

B7

A7

FA

S1

C1

B1

A1

FA

S0

C0

FA

A0

B0

图1.8 串行进位加法器

1. 实验内容
   1. 设计一个全加器
   2. 设计一个8位行波进位加法器

四、实验步骤

需要建立两个工程——FA和Add8，FA中创建全加器并进行仿真。正确后保存为模块。

**模块定义方法：**

File——Create/Update——Create Symbol File for Current File （.bsf）

模块的使用：在原理图文件中，和其他系统模块一样使用

CTRL+鼠标滚轮 可以调整原理图、波形图窗口显示比例。

**模块的使用方法：**

新建Project，若用到已建好的模块，需将 bdf和bsf文件同时拷贝到新建工程的文件夹

加减法器波形仿真时，先调整信号顺序**由高到低（A7-A0、B7-B0、S7-S0）**，

选择一组（A3-A0），鼠标右键单击——快捷菜单——Grouping——Group——命名（A）

显示类型也可以此刻更改，也可以在后面更改properties中的Radix

**总线的用法**：input（output）改为A[3..0]，自动拖线产生粗线(总线)，

各细线更改属性分别为A[0] A[1] A[2] A[3] 这样就可以了

如果觉得不好看，可以拖动粗线，让它依次与细线相交，然后松开鼠标即可。

仿真时，请使用功能仿真（functional）

使用时序仿真时最好一次变化一个量，否则会产生毛刺。