文章编号：1672-2892(2011)04-0507-04

一种简易 MCU 的加法器设计方法

张 恒[1]，江 猛[2]

(1.苏州经贸职业技术学院 机电系，江苏 苏州 215009；2.苏州华芯微电子有限公司，江苏 苏州 215011)

摘 要：算术逻辑部件(ALU)是整个微控制单元(MCU)运算的核心，相当于人类的大脑。ALU 的运算性能直接影响整个 MCU 运行的效率。一般简易 MCU 的内核只需进行加、减、逻辑运算等，不涉及到乘除，针对此特点设计了一种简易的加法电路，并在 FPGA 下进行仿真验证，仿真结果达到了设计要求，该 ALU 部分能根据不同的使能信号实现加、减、逻辑与或非以及数据传输功能。

关键词：微控制单元；算术逻辑部件；全加器；进位电路

中图分类号：TN402 **文献标识码：**A

A simplified method of the MCU adder design

ZHANG Heng[1]，JIANG Meng[2]

(1.Electronic Department，Suzhou Institute of Trade & Commerce，Suzhou Jiangsu 215009，China；
2.HuaXin Micro-electronic Co.,Ltd，Suzhou Jiangsu 215011，China)

Abstract：Arithmetic Logic Unit(ALU) is the core of the Micro Control Unit(MCU) operation, equivalent to the human brain. The operation performance of ALU directly affects the efficiency of the entire running of MCU. General simple MCU cores need only add, subtract and logical operations, not involving multiplication and division. Aiming to this feature, a simple adder circuit is designed. The simulation verification under FPGA is performed, whose results meet the design requirements. The ALU part in the design can achieve the functions of addition, subtraction, logic and-or-not and data transferring according to the enabling signals.

Key words：Micro Control Unit；Arithmetic Logic Unit；full adder；carry out circuit

加法器常用的有串行加法器和超前进位链加法器。串行加法器进位 C_{in} 逐位向上推进，因此速度受门延迟限制，位数越多，速度越慢[1]。一般 8 位加法器工作频率在 50 MHz 以下，但成本较小。超前进位链加法器没有这种限制，但它的电路较多，成本较高，简易 MCU 的 ALU 加法器一般选用串行全加法器[2]。

1 全加器的工作原理

设 A,B 为要相加的数，C_{in} 为进位输入，Sum 为和，C_o 是进位输出。如果要实现多位加法，可以进行级联，就是串起来使用，比如 8 位+8 位，就需要 8 个全加器，这种级联串行结构速度慢[3]。如果要并行快速相加，可以用超前进位加法，以 1 位全加器为例，其真值表如表 1 所示。

表 1 全加器的真值表

Table1 Truth table of full adder

A	B	C_{in}	Sum	C_o
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	1	0
1	1	1	1	1

减法也可以用加法器来实现，实现减法的方法：把被减数取反，然后再加 1，即加上被减数的补码。如，$A-B$ 可以表示为：$A-B=A+\bar{B}+1$。

综合全加器的真值表，可以得出全加器的逻辑表达式为：

$$\begin{cases} Sum = A \oplus B \oplus C_{in} \\ C_o = AB + (A+B)C_{in} \end{cases} \tag{1}$$

为便于实现 MCU 的 ALU 功能，通过变换 Sum 公式，可以得出式(2)。

$$Sum = (A+B) \oplus (AB) \oplus C_{in} \tag{2}$$

收稿日期：2010-08-31；修回日期：2011-01-13

从式(2)可以看出，全加器通过一定的变换，可以实现与或非等逻辑运算。根据式(2)，如果在每 1 项前面加一控制信号，就可以得到需要的算术、逻辑运算，如式(3)。

$$Sum = En1\&(A+B)\oplus En2\&(AB)\oplus En3\&C_{\text{in}} \tag{3}$$

对式(3)进行分析：

1) 当需要算术运算时，让 $En1=En2=En3=1$ 就得到全加器公式(2)；

2) 当需要“异或”逻辑时，让 $En1=1,En2=1,En3=0$，就得到式(4)：

$$Sum = (A+B)\oplus(AB)=A\oplus B \tag{4}$$

3) 当需要“或”逻辑时，让 $En1=1,En2=0,En3=0$，就得到式(5)：

$$Sum=A+B \tag{5}$$

4) 当需要“与”逻辑时，让 $En1=0,En2=1,En3=0$，就得到式(6)：

$$Sum=A\&B \tag{6}$$

5) 当不需要上述运算，只需传输时，让 $En1=0,En2=1,En3=0$，A,B 其中 1 个信号为 1，就可得到：$Sum=A$ 或 $Sum=B$；也可以让 $En1=1,En2=0,En3=0$，A,B 其中 1 个信号为 0，就可得到：$Sum=A$ 或 $Sum=B$。

2 Sum 的电路设计

通过对全加器的原理分析，得到全加器的 Sum 逻辑方程，由逻辑方程(3)得到如图 1 所示的 Sum 电路结构。

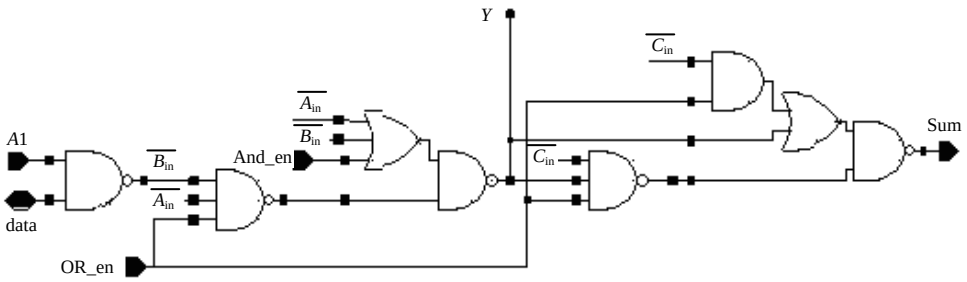

Fig.1 Sum structure
图 1 Sum 结构

通过该电路，可以实现 ALU 的算术、逻辑、传输等功能，所有数据总线进来的数据都取反。

其电路工作情况分析如下：

1) 当 OR_en=1,And_en=0 时，由图 1 的电路逻辑工作过程得到 Y 的逻辑表达式为：

$$Y = \overline{\overline{\overline{A_{\text{in}}B_{\text{in}}}}(\overline{A_{\text{in}}}+\overline{B_{\text{in}}})} = \overline{\overline{\overline{A_{\text{in}}B_{\text{in}}}}}+\overline{(\overline{A_{\text{in}}}+\overline{B_{\text{in}}})} = \overline{A_{\text{in}}\oplus B_{\text{in}}} \tag{7}$$

Sum 反的逻辑表达式为：

$$\overline{Sum} = \overline{Y\overline{C_{\text{in}}}}(Y+\overline{C_{\text{in}}}) = (\overline{Y}+C_{\text{in}})(Y+\overline{C_{\text{in}}}) = A_{\text{in}}\oplus B_{\text{in}}\oplus C_{\text{in}} \tag{8}$$

如果 Sum 非再取反一次，就得到 Sum 和的结果，符合全加器的表达式。

2) 当 OR_en=1,And_en=0 时，由图 1 的电路逻辑工作过程可知，如果 $C_{\text{in}}=0$，则得式(9)：

$$\overline{Sum} = A_{\text{in}}\oplus B_{\text{in}} \tag{9}$$

如果 Sum 非再取反一次，则就会得到“异或”的结果。

3) 当 OR_en=1,And_en=1 时，由图 1 的电路逻辑工作过程得到 Y 的逻辑表达式为：

$$Y = \overline{\overline{\overline{A_{\text{in}}B_{\text{in}}}}} = \overline{A_{\text{in}}+B_{\text{in}}} \tag{10}$$

Sum 反的逻辑表达式为：

$$Sum = \overline{Y\overline{C_{\text{in}}}}(Y+\overline{C_{\text{in}}}) = (\overline{Y}+C_{\text{in}})(Y+\overline{C_{\text{in}}}) = \overline{Y}\,\overline{C_{\text{in}}}+YC_{\text{in}} = Y\odot C_{\text{in}} \tag{11}$$

如果此时 $C_{\text{in}}=0$，则 $\overline{Sum}=\overline{Y}=A_{in}+B_{in}$。如果 Sum 非再取反 1 次，则得到“或”的结果。

4) 当 OR_en=0,And_en=0 时，由图 1 的电路逻辑工作过程得到 Y 的逻辑表达式为：

$$Y = \overline{\overline{A_{\text{in}}}+\overline{B_{\text{in}}}} \tag{12}$$

Sum 反的逻辑表达式为：

$$\overline{Sum} = Y = \overline{\overline{A_{in}} + \overline{B_{in}}} = A_{in}B_{in} \tag{13}$$

如果 Sum 非再取反 1 次，则得到“与”的结果。

通过以上逻辑功能分析，可知该结构的 Sum 电路基本上可以实现简易 MCU 的加法器内核逻辑功能。

3　$C_{out}(C_o)$“进位”电路设计

通过对全加器的原理分析，得到全加器的 C_{out} 逻辑方程，由逻辑方程(1)中的 C_o 逻辑表达式[4]，得出如图 2 所示的全加器的进位电路 C_o 的电路结构。

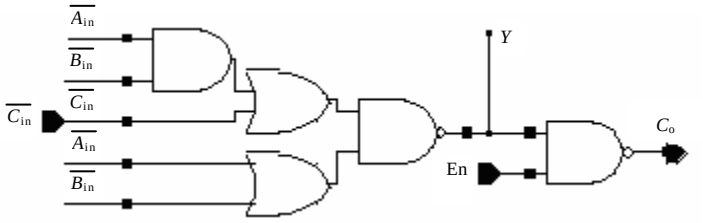

Fig.2 C_{out} carry circuit
图 2 C_{out} 进位电路

由电路图 2 的逻辑工作过程得到 Y 的逻辑表达为：

$$Y = \overline{(\overline{A_{in}}\,\overline{B_{in}} + \overline{C_{in}})(\overline{A_{in}} + \overline{B_{in}})} = \overline{(\overline{A_{in}}\,\overline{B_{in}} + \overline{C_{in}})} + \overline{(\overline{A_{in}} + \overline{B_{in}})} = \overline{\overline{A_{in}}\,\overline{B_{in}}}\,\overline{C_{in}} + A_{in}B_{in} = (A_{in} + B_{in})C_{in} + A_{in}B_{in} \tag{14}$$

由此可见 Y 值符合全加器的进位式(15)。

$$C_{out} = \overline{Y \,\&\, En} \tag{15}$$

当 En=1 时，$C_{out} = \overline{Y}$，此时 C_{out} 为 Sum 表达式中的 $\overline{C_{in}}$。当 En=0 时，C_{out}=1，此时 Sum 电路执行逻辑运算。

结合图 1 的 Sum 电路结构和图 2 的 C_{out} 进位电路结构，就完成了 1 位的全加器，对此电路进行扩展成 8 位加 8 位，就需要 8 个全加器，这种级联串行结构速度慢，一般 8 位加法器工作频率在 50 MHz 以下，但成本较小，可作为简易 MCU 的 ALU 的内核电路[5]。

4　仿真验证

通过前期的加法电路结构设计，整个适合 MCU 的加法器电路设计已基本完成。对整个全加器的整体电路进行仿真，再通过 FPGA 进一步来验证时序是否正确，理论波形能否正常输出。通过 FPGA 验证，能够提高流片的成功率。

FPGA 验证需要特定的软件工具来进行，一般选择自己比较熟悉的工具，如 Altera 公司的 MaxplusII。FPGA 为 Altera 公司的 EP1K100QC208-3，其资源有 100 000 门，足够用[6]。FPGA 验证结果如图 3 所示。

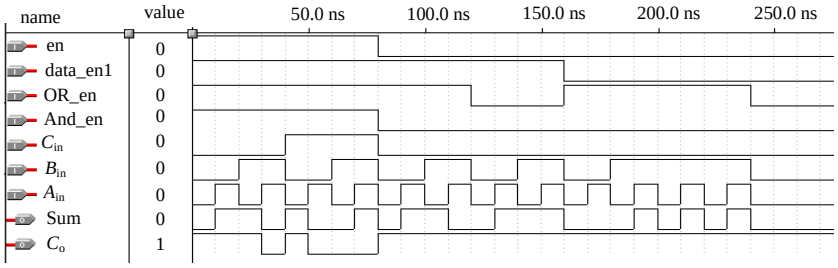

Fig.3 FPGA simulation verification of one-bit full adder
图 3　一位全加器的 FPGA 仿真验证

仿真讨论如下：

1) 当需要算术运算时，让 data_en1=OR_en2=And_en3=1，其仿真结果表明实现了算术运算；

2) 当需要“异或”逻辑时，让 data_en1=1,OR_en2=1,And_en3=0，其仿真结果表明实现了算术运算；

3) 当需要“或”逻辑时，让 data_en1=1,OR_en2=0,And_en3=0，其仿真结果表明实现了算术运算；

4) 当需要“与”逻辑时，让 data_en1=0,OR_en2=1,And_en3=0，其仿真结果表明实现了算术运算；

5) 当不需要上述运算，只需在传输时，让 En1=0,En2=1,En3=0，A_{in},B_{in} 其中 1 个信号为 1，就可得到：Sum=B_{in}；仿真结果表现为数据传输。

通过 8 个一位全加器进行串行连接，即前一位全加器的进位作为后一位全加器的 C_{in}，从而实现八位全加器的逻辑和算术运算。

(下转第514页)