DOI:10.13382/j.jemi.2005.01.010
第19卷 第1期 电子测量与仪器学报 Vol. 19 No. 1
JOURNAL OF ELECTRONIC
2005年2月 MEASUREMENT AND INSTRUMENT

自适应滤波器的FPGA实现

高清运 李学初

(南开大学微电子科学系,天津 300071)

摘 要:本文采用自上而下的设计思想,用FPGA实现了自适应滤波器。自适应滤波器选择FIR滤波器结构,采用了改进的LMS算法,从而使得在同样的硬件资源下滤波的速度可提高一倍;本论文用VHDL编写代码,用MaxplusII进行编译、综合和仿真,FPGA器件选用ALTERA公司的FLEX10K系列3万门的芯片,综合结果显示,所设计的自适应滤波器使用FLEX10K30的70%的资源;对综合后的网表进行了仿真,当时钟周期取40ns时,滤波器仍能够很好地消除噪声。

关键词:自适应算法;滤波器;FPGA

Implementation of Adaptive Filter with FPGA

Gao Qingyun LI Xuechu

(Department of Microelectronic Science, Nankai University, Tianjin, 300071, China)

Abstract: An adaptive filter is realized with FPGA using top-down design idea in this paper. A FIR(finite impulse response) filter structure was used for it. Since an improved LMS (least mean square) algorithm was adopted, the operation speed of the proposed adaptive filter was twice of that of a standard LMS algorithm FIR filter. The adaptive filter was coded with VHDL, compiled and synthesized with ALTERA MaxplusII. A FLEX10K30 was selected for synthesizing, and the proposed adaptive filter occupied 70% resources of this chip. The proposed adaptive filter was simulated with MaxplusII, when the period of clock was 40ns, the filter could operate very well. The simulation results verified the correctness of the design.

Keywords: adaptive algorithm; filter; FPGA.

1 引 言

在数字信号处理中,滤波器是一个重要的单元。自适应滤波器的参数可以调整以满足被控制对象的未知和时变的要求,因此,在未知统计量环境中进行信号滤波时,自适应滤波器较传统的固定系数滤波器具有更高的性能,自适应滤波器已广泛应用于通信、控制、雷达等诸多方面[1]。

本文选择FIR滤波器结构,采用改进的LMS算法,从而使得在同样的硬件资源下滤波的速度可提高一倍。选用ALTERA公司的FLEX10K30器件,用MaxplusII进行了仿真验证,结果表明,自适应滤波器能够很好的消除噪声。

2 自适应算法的改进

自适应算法有很多种,如果用FPGA来实现,理想的算法是最小均方差(LMS)算法,其输出信号$y(n)$、输出误差$e(n)$的计算公式为[2]

$$y(n) = W(n)X^T(n) \tag{1}$$

$$e(n) = d(n) - y(n) = d(n) - W(n)X^T(n-1) \tag{2}$$

式中$X(n)$表示第n时刻输入信号向量,$X(n)=[x(n), x(n-1)\cdots\cdots x(n-M+1)]$,$M$为滤波器的阶数$d(n)$表示第$n$时刻的输入期待响应,$y(n)$、$e(n)$分别表示第$n$时刻的输出信号与输出误差,$W(n)$表示$n$时刻权系数向量,$W(n)=[w(n,0),$

本项目为深圳华为公司科技基金资助项目。
本文于2003年3月收到。高清运:博士,副教授;李学初:硕士研究生。

$w(n,1)\cdots\cdots w(n,M-1)]$。对于自适应滤波器来说,权系数是不断更新的,权系数的更新使用

$$W(n+1)=W(n)+\mu[-\nabla(n)] \tag{3}$$

式中 μ 表示收敛因子,自适应滤波器收敛的条件是

$$0<\mu<\frac{1}{2\lambda_{\max}} \tag{4}$$

其中 $\lambda_{\max}$是输入信号的自相关矩阵 $\boldsymbol{R}_{XX}$的最大特征值; $\nabla(n)$表示 n 时刻的均方误差梯度,它的精确计算十分困难,通常使用一种非常有效的近似[1]

$$\nabla(n)\approx -2e(n)X(n) \tag{5}$$

此时,权系数更新可以表示为

$$W(n+1)=W(n)+2\mu e(n)X(n) \tag{6}$$

利用式(6)权系数可以非常方便的更新,但是有一个问题:对当前的权系数进行更新,必须知道当前时刻的误差信号 $e(n)$,显然必须在输出信号 $y(n)$与误差信号 $e(n)$计算完毕以后才能进行权系数的更新,换言之,权系数的更新与滤波($y(n)$的计算)不能同步进行。后面将看到,我们采用的是串行工作方式,如果权系数的更新与滤波($y(n)$的计算)不能同步进行,滤波器完成一次滤波需要 17 个时钟周期,前 8 个时钟周期进行滤波,后 8 个时钟周期完成对权系数向量的更新,最后一个时钟周期为下一次滤波作预处理。如果能够实现权系数的更新与滤波同步进行,那么在滤波的同时权系数也被更新了,因此,滤波($y(n)$的计算)与权系数的更新只需要 8 个时钟周期,而不是 16 个。这样,自适应滤波器的滤波速度将提高了将近一倍,这是我们所期待的。

要实现这一点,必须对算法进行改进。前面提到,权系数的更新之所以不能与滤波同步进行是由式(6)决定的,如果将式(6)中的 $e(n)$改成 $e(n-1)$后,自适应算法还能成立,权系数更新与 $y(n)$的计算当然就能同步进行了。有鉴于此,我们根据

$$W(n+1)=W(n)+2\mu e(n-1)X(n-1) \tag{7}$$

设计电路[3],式中,$2\mu e(n-1)X(n-1)$为 $n-1$ 时刻均方误差梯度的近似形式。

3 自适应数字滤波器的结构和工作原理

3.1 自适应数字滤波器的结构

本文设计了一个 8 阶自适应滤波器,输入信号 $x(n)$、输入期待响应 $d(n)$、输出信号 $y(n)$与输出误差 $e(n)$均为 8 位数据,权系数向量 $W(n)$中的每个元素都为 12 位数据,其中低 9 位为小数位,最高位为符号位,图 1 给出了滤波器的结构框架图。

图中,A1、B、C1、D1、E1、G 用于计算输出信号 $y(n)$,由于在运算过程中,累加器的内容不断变化,等运算完毕以后(也就是运算进行 8 个时钟周期以后),才等于输出信号 $y(n)$。要保存 $y(n)$的值,必须提供一个寄存器,它就是图 1 中的寄存器 G。F 用于产生输出误差 $e(n)$,并将误差锁存在寄存器 G2 中,以供更新权系数之用;A1、A2、B、C2、D2、E2 用于更新权系数,更新过程中所使用的误差为上一次计算所产生的并保存在寄存器 G2 中的误差 $e(n-1)$。

需要指出的是,更新权系数时要用到 $2\mu X(n-1)$ $e(n-1)$

从图 1 可以看出,$X(n-1)$与 $e(n-1)$相乘用的是乘法器,但是二者的乘积与 2μ 相乘,是用一个数据选择器来实现。这是因为 2μ 为收敛因子的两倍,通常数值远小于 1,只要满足收敛条件,它的值大一点或小一点对滤波效果影响并不明显,因此可以考虑它只取诸如以下的一系列分立的值

$$\frac{1}{2}\quad\frac{1}{4}\quad\frac{1}{8}\quad\frac{1}{16}\cdots\cdots\frac{1}{2^n}$$

这样就可以通过移位运算来代替乘法运算,从而大大减少电路所耗的硬件资源;不仅如此,由于这样实现起来电路的延时远小于一个乘法器的延时,用数据选择器 C2 代替乘法器可以很大程度上提高滤波器的最高采样频率。

3.2 滤波器的工作原理

3.2.1 运算周期与时钟周期

在介绍滤波器的工作原理之前,预先定义运算周期与时钟周期这两个概念:

运算周期:从滤波器开始运算到滤波完毕需要一段时间,这段时间我们将其定义为一个运算周期。

时钟周期:定义图 1 中 CLOCK 的周期为时钟周期,滤波器内部的移位寄存器使用的时钟就是 CLOCK。前面已提到,完成一次对 $y(n)$的计算需要 8 个时钟周期,定义一个运算周期内的第一个时钟周期为 T1,以此类推,第 8 个时钟周期为 T8;第 8 个时钟周期过后,还要完成对误差 $e(n)$的计算,同时为下一个运算周期做准备,定义第 9 个时钟周期为 T9,尔后,滤波器进入等待状态。

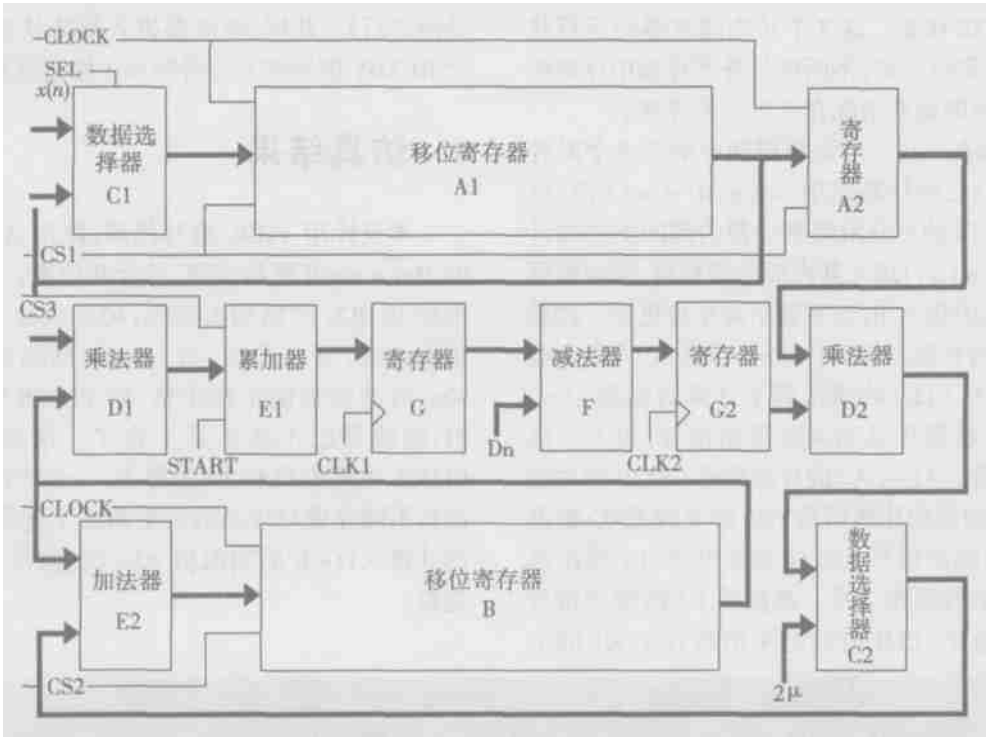

图 1　自适应滤波器的电路结构图

3.2.2 工作原理

图 1 中，移位寄存器 A1 与寄存器 A2 用于存放输入数据 $X(n)$，移位寄存器 B 中存放的是权系数。当采样的输入信号 $x(n)$准备好以后，应发出一个准备好的信号 READY 给滤波电路的控制器，它可以开始滤波了。下面将介绍滤波器从 READY 信号有效到滤波完毕这一过程中控制信号的值以及各寄存器的状态。

(1)READY 上升沿出现到 T1 状态来临之前。这一时间段，外部采样好的数据 $x(n)$还没有被送入到移位器存器 A1 中，因此 A1 中存放的数据由左至右依次为 $x(n-1)$、$x(n-2)$、$x(n-3)$、$x(n-4)$、$x(n-5)$、$x(n-6)$、$x(n-7)$与 $x(n-8)$；寄存器 A2 中存放的数据时 $x(n-9)$；移位寄存器 B 中存放的是 $w(n,0)$、$w(n,1)$、$w(n,2)$、$w(n,3)$、$w(n,4)$、$w(n,5)$、$w(n,6)$与 $w(n,7)$；寄存器 G 中存放的是 $y(n-1)$；G2 中存放的是 $e(n-1)$。移位寄存器 A1 与寄存器 A2 的片选信号 CS1 从高电平变为低电平(低电平有效)；数据选选择器 C1 的控制信号 SEL 维持低电平，C1 选择的是外部送进来的采样信号 $x(n)$；累加器 E1 的清零信号在 T1 上升沿到来前把 E1 清零；移位寄存器 B 的片选信号 CS2 与累加器 E1 的片选信号 CS3 此时都为无效。这样，一旦 T1 到来，外部送来的采样信号 $x(n)$被送进移位寄存器 A1 的最左端，与此同时，A1 中的数据均自左向右移一位，$x(n-8)$被这个上升沿送入寄存器 A2。滤波器中的其他寄存器此时都不发生任何动作。

(2)T1 状态。这期间各寄存器的状态不难由 T1 的上升沿动作推测出来，移位寄存器 A1 中存放的数据由左至右依次为 $x(n)$、$x(n-1)$、$x(n-2)$、$x(n-3)$、$x(n-4)$、$x(n-5)$、$x(n-6)$与 $x(n-7)$；寄存器 A2 中存放的数据时 $x(n-8)$；移位寄存器 B 中存放的是 $w(n,0)$、$w(n,1)$、$w(n,2)$、$w(n,3)$、$w(n,4)$、$w(n,5)$、$w(n,6)$与 $w(n,7)$；寄存器 G 中存放的是 $y(n-1)$；$G2$ 中存放的是 $e(n-1)$。进入 T1 状态以后，A1 与 A2 的片选信号 CS1 继续有效；数据选择器 C1 的控制信号 SEL 变为高电平，C1 选择的是 A1 最右端送过来的数据 $x(n-7)$；移位寄存器 B 的片选信号 CS2 与累加器 E1 的片选信号 CS3 变为有效；累加器的清零信号 CLR 变为高电平，E1 不再被清零，可以开始计算 $y(n)$。在此期间，乘法器 D1 完成乘法运算 $x(n-7)$ $w(n,7)$，累加器完成加法运算，等到 T2 上升沿将加法结果送入累加器中的寄存器。同时由乘法器 D2、数据选择器 C2 以及加法器 E2 对权系数 $w(n,7)$进行更新。T1 状态结束时，加法器 E2 输出的数据是 $w(n,7)+2\mu x(n-8)e(n-1)$，T2 的上升沿将其送入移位寄存器 B，完成了对 $w(n,7)$的更新。

(3)T2 ~ T8 状态。这 7 个状态滤波器的运行状况与 T1 状态完全一致,不同的是各寄存器中存放的数据。滤波器的运算情况在这里不再详述。

(4)T9 状态。在一个运算周期中的前 8 个时钟周期内,$y(n)$已经计算完毕,$w(n,0)$ ~ $w(n,7)$也被更新完毕,T9 的上升沿将累加器内部的加法器计算结果(等于$y(n)$)送入其内部的寄存器,同时将更新后的$w(n,0)$送入 B,权系数至此全被更新。此后寄存器 G 的时钟信号 CLK1 将$y(n)$送入 G,寄存器 G2 的时钟信号 CIK2 将减法器 F 计算的误差$e(n)$送入 G2。T9 状态还必须调整控制信号,为下个运算周期做准备。A1 与 A2 的片选信号 CS1,B 的片选信号 CS2,累加器的片选信号 CS3 都变为无效;数据选择器 C1 的控制信号 SEL 变为低电平,C1 再次选择外部送进来的采样信号。累加器 E1 的清零信号 CLR 变为低电平(CLR 的低电平出现在 CLK1 的上升沿之后)。此后,滤波器进入等待状态,直到下一个 READY 信号的上升沿到来才使它重新启动工作。

4 仿真结果

本设计用 VHDL 编写代码,使用 ALTERA 公司的 Max+plusII 进行编译、综合和仿真。图 2,3,4 分别是 CLOCK 的周期为 50ns,40ns,30ns 时滤波器的仿真结果,可以看出,当 CLOCK 周期取值 50ns 与 40ns 时滤波器能工作正常,而 CLOCK 周期取 30ns 时,滤波器已不能正常工作了。这意味着,如果 CLOCK 的周期只有 30ns,那么,一个时钟周期内,滤波器不能完成对权系数的更新或不能完成对$y(n)$的计算,CLOCK 周期取值 40ns 时,滤波还能很好的完成。

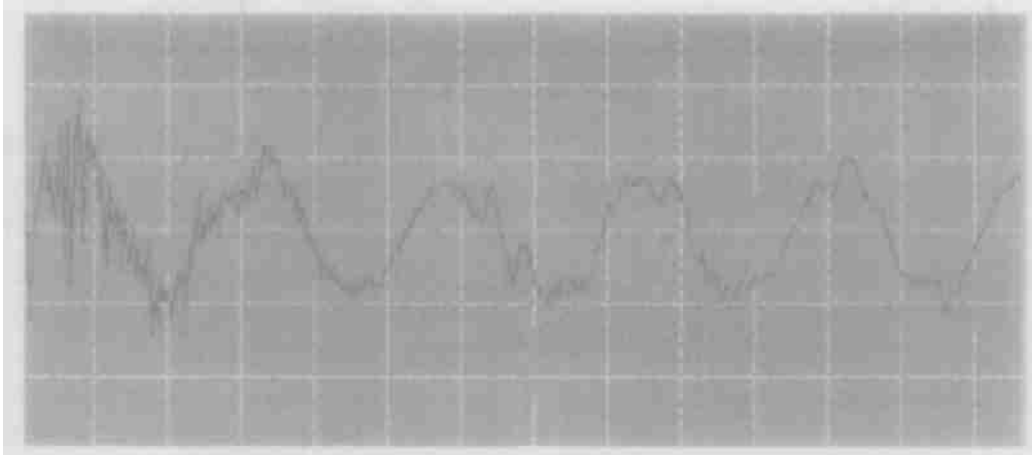

图 2 CLOCK 周期为 50ns 时的滤波情况

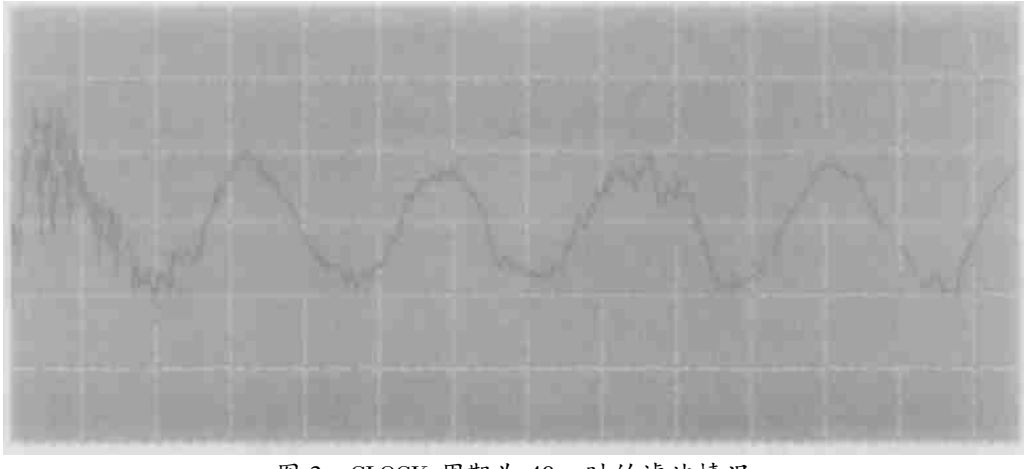

图 3 CLOCK 周期为 40ns 时的滤波情况

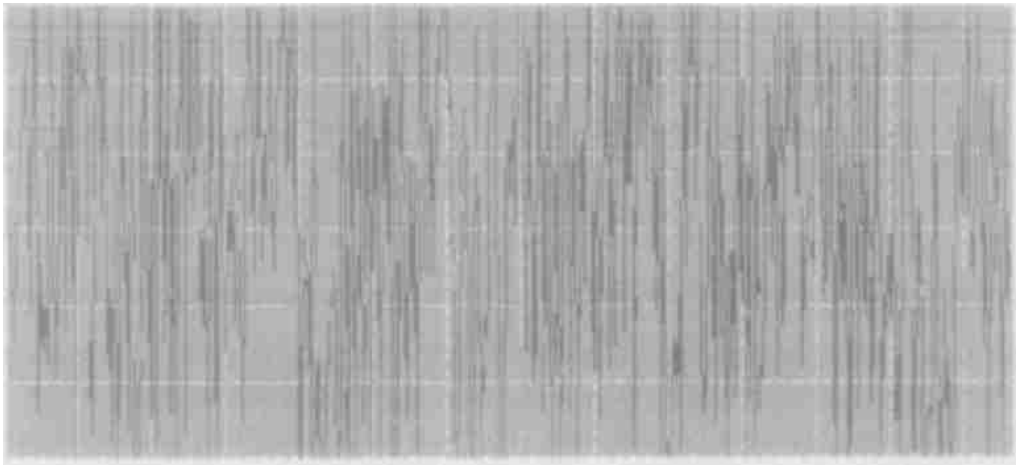

图 4　CLOCK 周期为 30ns 时的滤波情况

5　结　论

本文采用自上而下的设计思想，用 FPGA 实现了自适应滤波器。采用 ALTERA 公司的 FLEX10 系列 3 万门的芯片，综合后使用 70％的 LC 单元。对综合后的网表进行了仿真，仿真结果表明，自适应滤波器能够很好的消除噪声。

参考文献：

[1] 沈福民. 自适应信号处理，西安：西安电子科技大学出版社，2001.

[2] 张贤达. 现代信号处理，北京：清华大学出版社，1999.

[3] 高清运，李学初. 适合硬件实现的自适应滤波算法，微电子学与计算机，第 20 卷，2003 年，第 7 期，34～36.

作者简介：

高清运

高清运：1965 年出生，博士，南开大学微电子科学系副教授。主要研究方向为集成电路设计。

李学初：1979 年出生，南开大学微电子科学系硕士研究生。主要研究方向为集成电路设计。