

高速 A/D 与微处理器间的数据缓存技术

杜凌云, 黄士坦

(西安微电子技术研究所, 陕西 西安 710075)

摘要: 由于数字信号处理的种种优点, 现在多数时候是将模拟信号转换成数字信号再进行处理。在雷达系统中往往产生高频信号, 要对这类信号进行数字处理, 根据奈奎斯特采样定律, 要求 A/D 采样率高达 Gbps 量级。对此例高频信号进行采样的系统, 就是所谓的高速数据采集系统。高速数据采集具有系统数据吞吐率高的特点, 要求系统在短时间内能够传输并存储采集结果。模数转换后的数据快速存储能力在一定程度上制约着 A/D 转换的频率和最大采集时间。故高速数据采集系统中的数据存储是一个热点和难点。文中研究讨论了一种高达 1Gbps 的 A/D 与微处理器间的数据缓存技术。

关键词: 高速数据采集系统; 模数转换器; 采样率; 微处理器; 缓存; 交叉存储

中图分类号: TP311

文献标识码: A

文章编号: 1673-629X(2007)04-0167-04

Technology of Data Buffer Between High-Speed ADC and Microprocessor

DU Ling-yun, HUANG Shi-tan

(Xi'an Microelectronics Technology Institute, Xi'an 710075, China)

Abstract: Generally convert analog signals into digital signals for the advantages of the digital signals processing. The signals in the radar system has high frequency. So need the ADC that has the sampling rate of Gbps grade to sample the radar signals according to the Nyquist law. The sampling system that sample this high frequency signals is a high-speed data acquisition system. The high-speed data acquisition system that has a high data throughput demands quick data transmission and storage. The A/D converter's frequency and continuous sampling time is limited by the ability of the data transmission and storage. The technology of data buffer between high-speed ADC and microprocessor is important and difficult. This paper proposes the technology of data buffer between high-speed ADC and microprocessor that has 1Gbps grade sampling rate.

Key words: high-speed data acquisition system; A/D converter(ADC); sampling rate; microprocessor; data buffer; interleaved

1 高速数据采集系统简介

高速数据采集系统按照功能, 通常分为模拟信号调理电路、模数转换器、数据存储接口、大容量存储器、时钟电路和系统时序及控制逻辑电路等部分^[1,2]。本系统是以美国国家半导体公司的 ADC081000 作为模数转换器的。此款转换器采用独特的折叠/内插结构, 以高达 1.6GHz 的取样率将模拟信号转为 8 位分辨率的数字信号, 而且只需 1.9 伏(V)的额定供电, 功耗不会超过 1.4W。芯片还采用了美国国家半导体的低电压差分信号传输(LVDS)接口, 因此可以为高速信号提供一个低噪音、低失真的传送环境, 确保传送过程稳定可靠。ADC081000 的结构框图如图 1 所示。大容

量存储器选用 16 片 CYPRESS 公司的双端口静态异步存储器 CY7C006A。数据存储接口、时钟电路和系统时序及控制逻辑电路等部分用 xilinx 公司的 virtex4 系列 FPGA 实现。一般高速模数转换器的速度比微处理器的速度要快。不论什么样的高速数据采集系统, 只要是 ADC 的工作频率和微处理器的工作频率不匹配, 就需要缓存装置。笔者就此缓存技术进行研究讨论。

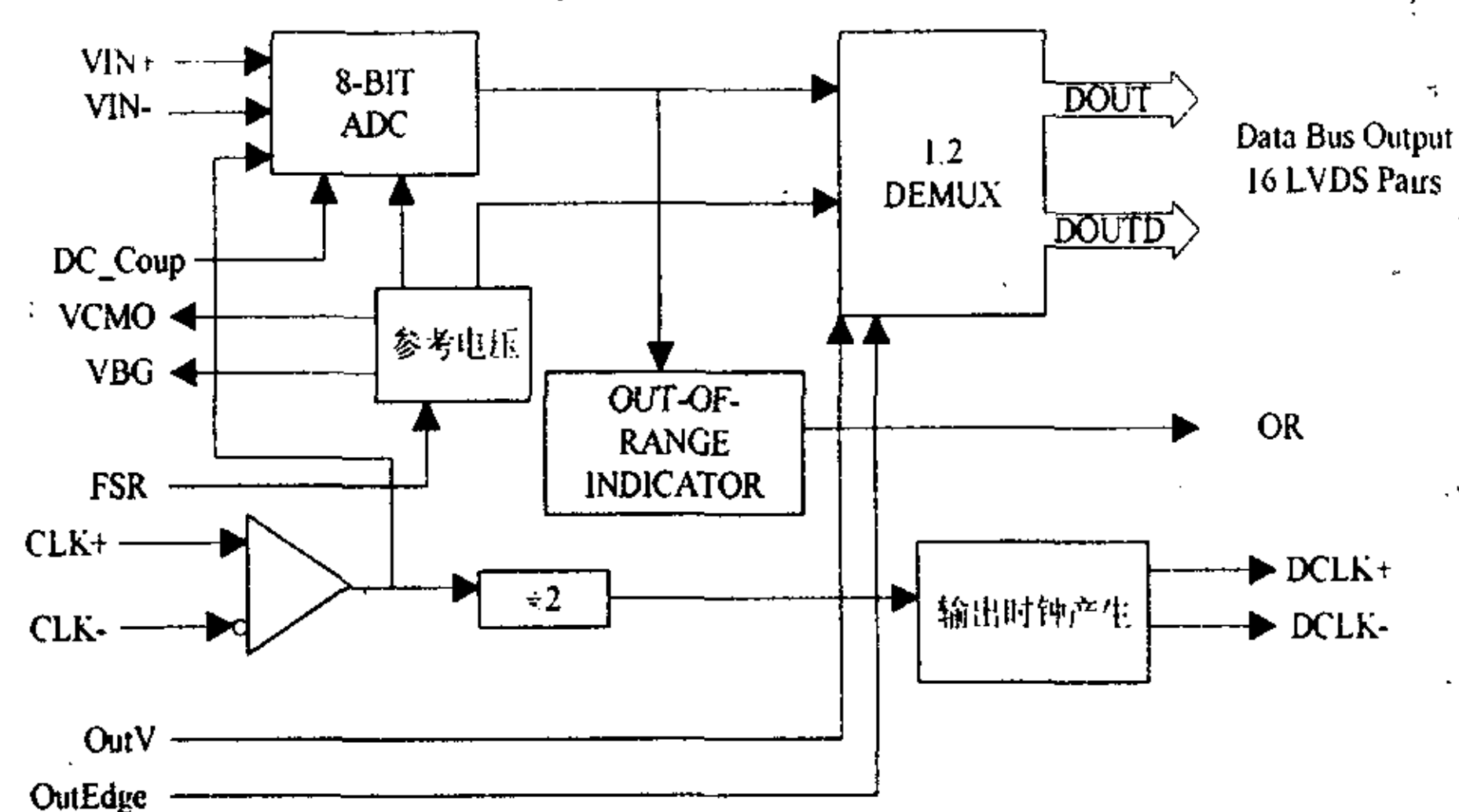


图 1 ADC081000 结构框图

收稿日期: 2006-06-21

作者简介: 杜凌云(1977-), 男, 陕西韩城人, 硕士研究生, 研究方向为高速数据采集系统中的缓存技术; 黄士坦, 研究员, 研究方向为空间计算机系统结构。

2 多体交叉存储技术

ADC081000 虽然是以 1Gbps 对模拟信号进行采样,但器件内部存在 1:2 的 DEMUX 使数据驱动两路差分总线,从而使频率下降为 500MHz。采样速度有多高,存储的速度就要求有多快,否则将丢失信号的信息。一般的存储技术是不能接受 500MHz 的数据的,所以采用多体交叉存储技术^[3]。多体交叉存储技术中在一个存储体内有多个容量相同的存储模块,而且各存储模块都有各自独立的地址寄存器、译码器和数据寄存器,各模块可独立进行工作。多体交叉存储技术采用交叉编址方式,这种编址方式使用地址码的低位字段,经过译码选择不同的存储模块,而高位字段指向相应的模块内部的存储字。这样,连续地址分布在相邻的模块内,而同一模块内的地址都是不连续的。当微处理器对存储体进行存取操作时,由时序控制线路对各模块分时启动、分时控制,最后所有的模块并行地工作。如对 8 个模块,在第一个存储周期的开始时刻启动模块 M_0 ,在 $T/8, 2T/8, 3T/8 \cdots 8T/8$ 时刻分别启动 $M_1, M_2, M_3 \cdots M_7$,经 T 后 8 个模块进入并行工作状态,各模块存储周期互相重叠。与一个存储体相比较,数据传送的平均速度可提高 8 倍,整个主存的有效周期缩小到原来每个模块存储周期的 $1/8$ 。所以可以用低速的存储模块采用多体交叉存储技术构成高速的存储体。

3 高速 A/D 与微处理器间的缓存技术

本系统采用的缓存技术就是上文所讨论的多体交叉存储技术^[1,4,5]。这里模块数取 8,有效存储时间为 2ns,则单个存储体的有效存储时间为 16ns。有这么一个数据缓存装置则相对微处理器高速的 A/D 和微处理器之间实现了异步的数据交流。本系统的交叉存储原理框图如图 2 所示。ADC081000 器件内部存在 1:2 的 DEMUX 使数据驱动两路差分总线,对每一路进行模 8 交叉存储,则两路总共需要 16 个存储体。这里选用 CYPRESS 公司的双端口静态异步存储器 CY7C006A,其存储时间最大为 12ns。因为每路每 2ns 就输出一个采样所得数据,所以要在 2ns 中将数据锁存到锁存器中。锁存器选用德州仪器 SN74AUC16374DGGR。可编程器件选用 xilinx 公司的 virtex4 系列。其中双端口静态异步存储器的写时序如图 3 所示,而 ADC081000 产生数据的时间矢量图如图 4 所示。在提供时钟信号 CLK 以及 PD 脚置低的情况下,ADC081000 就一直工作着。在 CLK+ 的下降沿 ADC081000 得到数据,然后在 7 个时钟周期之后从第一路送出数据,或者 8 个时钟周期之后从第二路送出。

送出时用 DCLK 同步。所以在此系统中,产生写入双端口的时序逻辑是比较重要的部分。这些时序用 verilog 硬件电路描述如下:

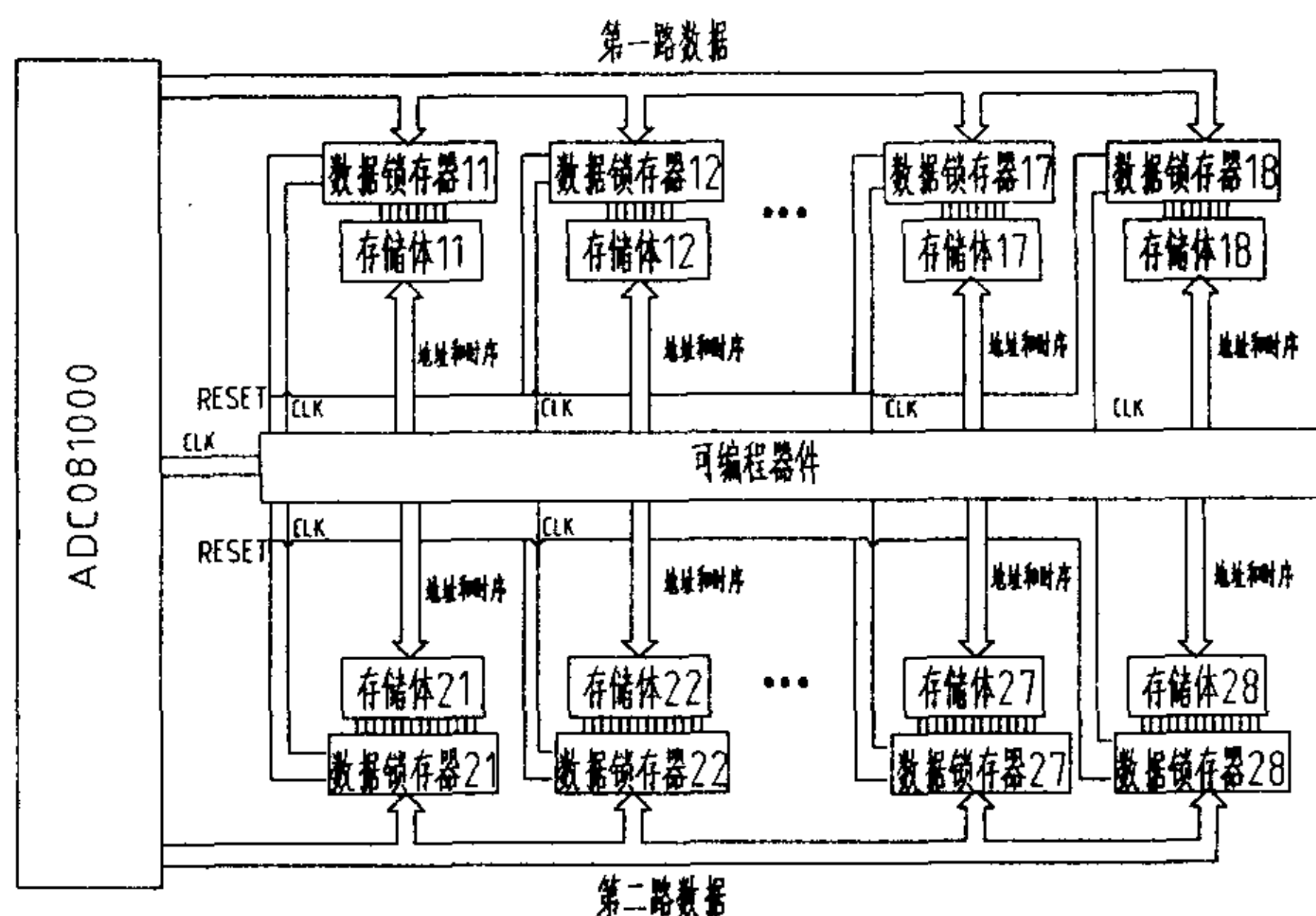


图 2 交叉存储结构图

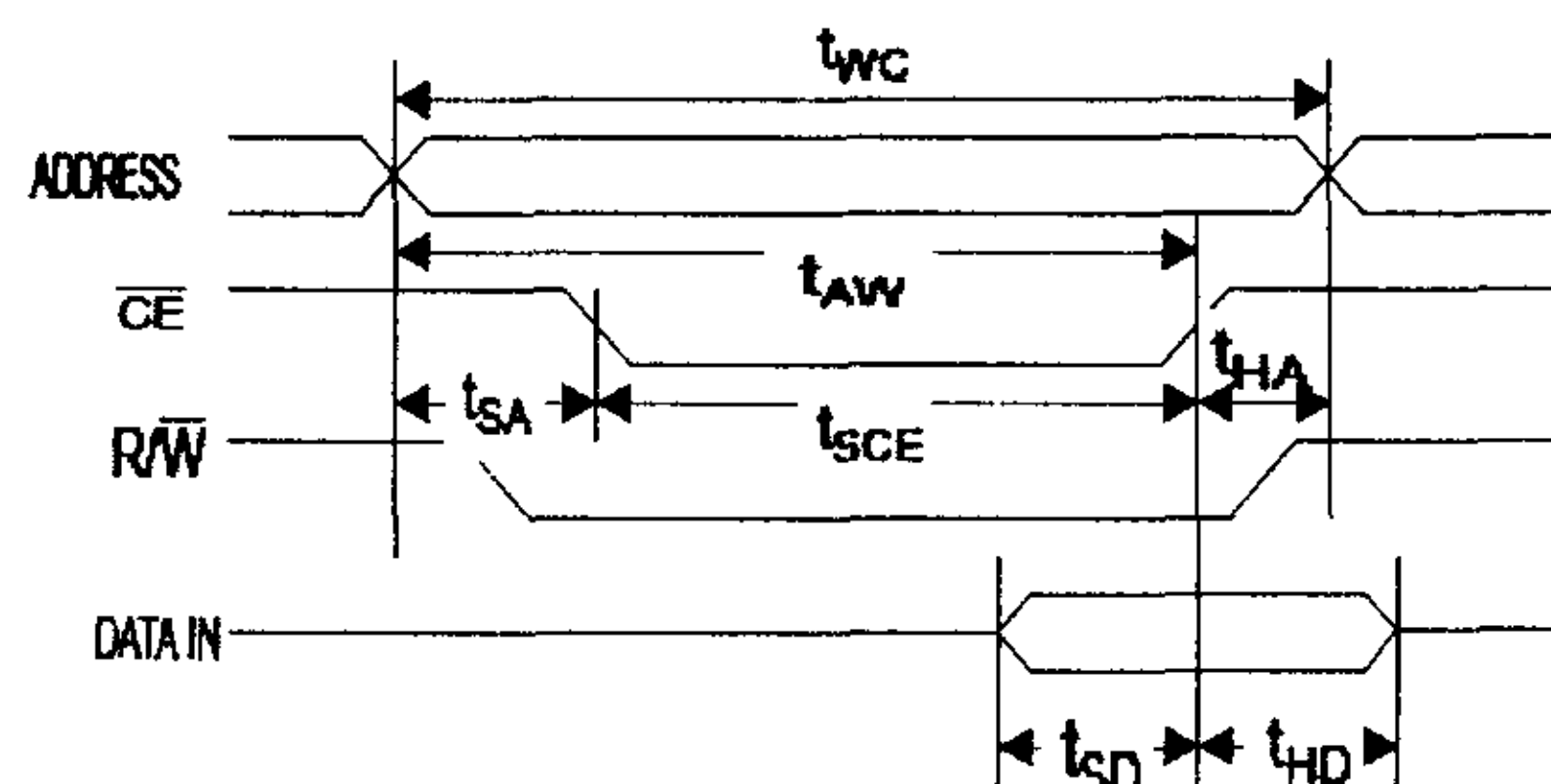


图 3 CY7006A 写时序

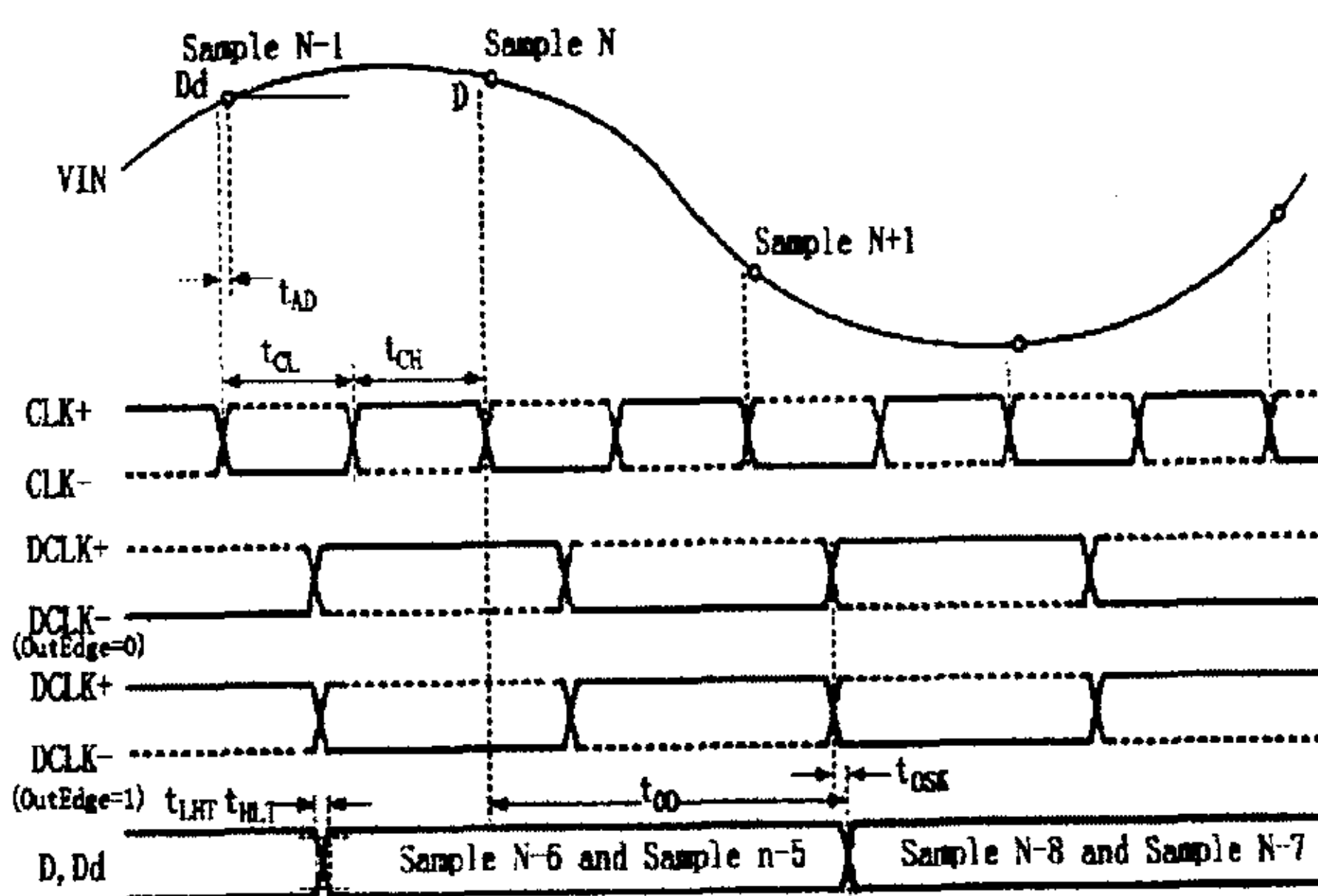


图 4 ADC081000 时间矢量图

'timescale 1 ps/1 ps

```
module shixu (clk, reset, ce, samp, address0, address1, address2, address3, address4, address5, address6, address7);
```

```
input clk;
```

```
input reset;
```

```
output[7:0] ce;
```

```
output[7:0] samp;
```

```
output[13:0]
```

```
address0, address1, address2, address3, address4, address5, address6,
address7;
```

```
wire clk;
```

```
wire reset;
```

```
reg fout;
```



```

reg[2:0] j;
reg[2:0] i;
reg[2:0] k;
reg[7:0] ce;
reg[7:0] samp;
reg[13:0] address0, address1, address2, address3, address4, address5, address6, address7;
always@(posedge clk)
    if(! reset)
        begin
            ce[0] <= 0;
            j <= 0;
        end
    else
        begin
            if(j < 2)
                begin
                    j <= 0;
                    ce[0] <= 1;
                end
            else
                begin
                    ce[0] <= 0;
                end
            j <= j + 1;
        end
always@(posedge clk)
    if(! reset)
        begin
            address0 <= 14'b11111111111111;
            i <= 7;
        end
    else
        begin
            if(i == 7)
                begin
                    address0 <= address0 + 1;
                end
            i <= i + 1;
        end
always@(posedge clk)
    if(! reset)
        begin
            ce[7:0] <= 0;
        end
    else
        begin
            ce[1] <= ce[0];
            ce[2] <= ce[1];
            ce[3] <= ce[2];
            ce[4] <= ce[3];
            ce[5] <= ce[4];
            ce[6] <= ce[5];
            ce[7] <= ce[6];
        end
end
always@(posedge clk)
    if(! reset)
        begin
            address1 <= 14'b11111111111111;
            address2 <= 14'b11111111111111;
            address3 <= 14'b11111111111111;
            address4 <= 14'b11111111111111;
            address5 <= 14'b11111111111111;
            address6 <= 14'b11111111111111;
            address7 <= 14'b11111111111111;
        end
    else
        begin
            address1 <= address0;
            address2 <= address1;
            address3 <= address2;
            address4 <= address3;
            address5 <= address4;
            address6 <= address5;
            address7 <= address6;
        end
end
always@(negedge clk)
    if(! reset)
        begin
            samp[0] <= 0;
            k <= 7;
        end
    else
        begin
            if(k == 7)
                begin
                    samp[0] <= 1;
                end
            else
                begin
                    samp[0] <= 0;
                end
            k <= k + 1;
        end
end
always@(negedge clk)
    if(! reset)
        begin
            samp[7:0] <= 0;
        end
    else
        begin
            samp[7:0] <= 0;
        end
end

```



```
begin
  samp[1] <= samp[0];
  samp[2] <= samp[1];
  samp[3] <= samp[2];
  samp[4] <= samp[3];
  samp[5] <= samp[4];
  samp[6] <= samp[5];
  samp[7] <= samp[6];
end
endmodule
```

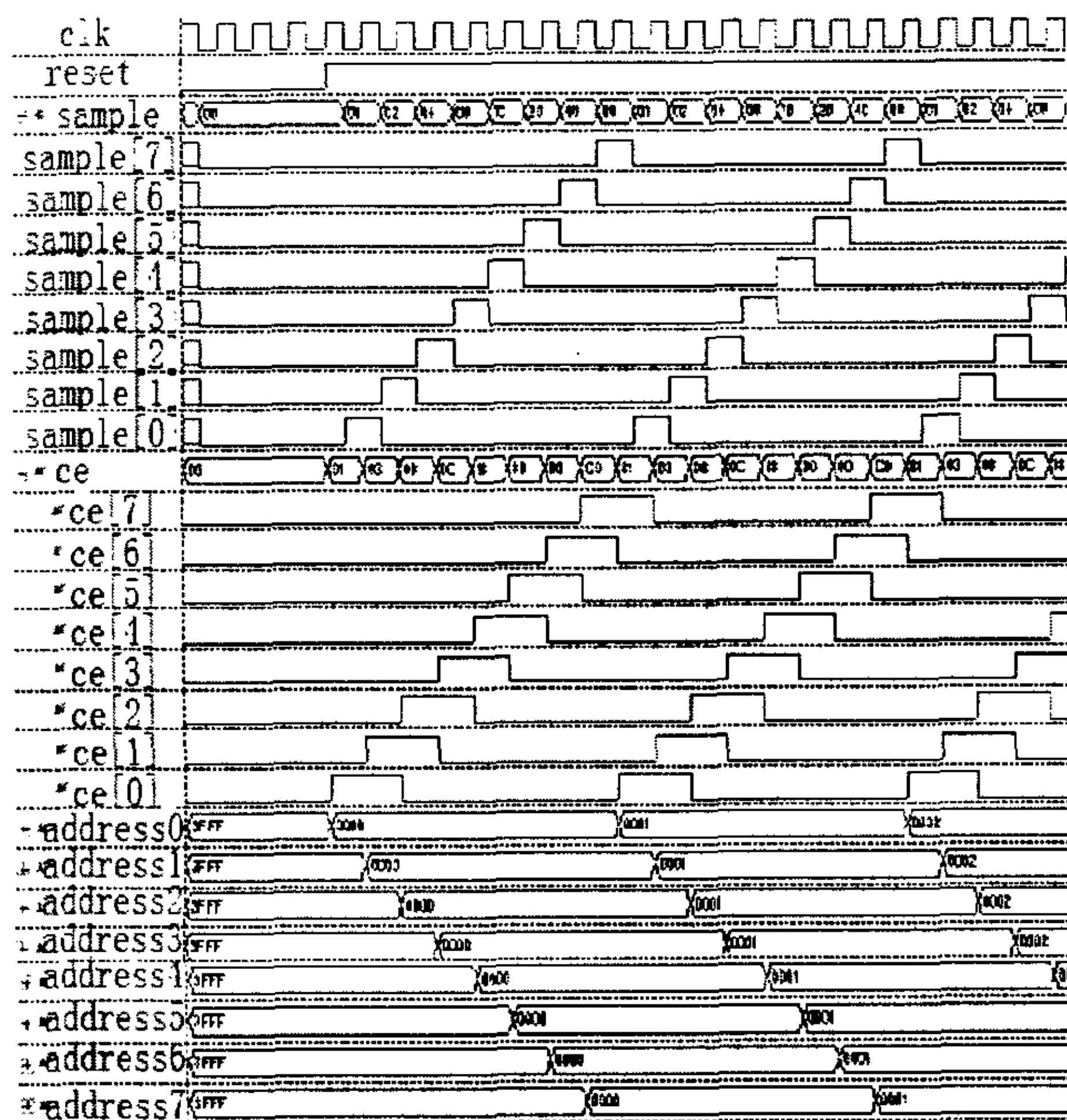


图 5 仿真波形

此段程序描述了第一路数据写入时所需要的时序,由于第二路与第一路差一个 CLK 时钟周期,这里不再赘述。其中的信号 ce 为片选, address 为地址信号, samp 为锁存信号。其波形如图 5 所示。从图中可以看出打入锁存的信号 samp[0]~samp[7]依次有效,从而将数据顺次地打入到锁存器 11 到锁存器 18;而 ce[0]~ce[7]依次有效,从而将存储器 11 到存储器 17 依次选通;地址 address0~address7 也是依次有效。这些都是为了实现数据的交叉存储。其原理也就是一个 8 分频分频器和一个 8 位移位寄存器。

4 结束语

由于此系统电路的速度大部分时钟已超过 50MHz,所以应该考虑使用高速电路设计的技术,也就是所谓的传输线、串扰等信号完整性问题。

参考文献:

- [1] National Semiconductor Corporation. ADC081000 Data Sheet [EB/OL]. 2004. <http://www.national.com>.
- [2] 马明建. 数据采集与处理技术[M]. 西安:西安交通大学出版社,1997.
- [3] 张代远. 计算机组成原理[M]. 北京:北京邮电大学出版社,2002.
- [4] Cypress Semiconductor Corporation. CY7C006A Data Sheet [EB/OL]. 2005. <http://www.cypress.com>.
- [5] 袁俊泉. Verilog HDL 数字系统设计及其应用[M]. 西安:西安电子科技大学出版社,2002.

(上接第 166 页)

的相关经济、地理数据,对数据仓库中的数据进行挖掘,发现一些对借用国外贷款有关的联系、知识,帮助决策者做出有利于发展建设的决定。

5 结束语

借用国外贷款决策系统结合本项目特点不同于其他商业智能系统,加入了信息输入功能,使信息的输入和分析决策工作全部通过浏览器完成,避免了过去文件下发、填报数据、提交文件、数据分析等繁琐的工作,提高了工作的效率和精确度,也是对政府办公自动化的一个推动。它能够很好地辅助借用国外贷款项目评价人员来解决项目评价过程中的重大决策问题,是决策者的得力助手。该系统在开发的过程中参加了山西省发改委组织的《山西省借用国外贷款 25 年回顾与总结》项目系统的部分分析模型,评价指标借鉴了该项目的分析方法,并将本系统应用于该项目中,帮助了对数

据的收集、分析,取得了良好的效果,得到了项目分析人员的认可。本系统的开发是将商务智能方案应用于政府决策的一次有意义的尝试,对以后的此类系统的开发起了指导作用。

参考文献:

- [1] INMON B. 数据仓库[M]. 北京:机械工业出版社,2001.
- [2] 王 珊. 数据仓库技术与联机分析处理[M]. 北京:科学出版社,1998.
- [3] 王昭义,刘 斌,蔡瑞英. Web 数据仓库及其在办公自动化系统中的应用[J]. 计算机技术与发展,2006,16(2):17-19.
- [4] ROY - FADERMAN A, DORSEY P. Oracle JDeveloper 10g 开发手册[M]. 北京:清华大学出版社,2006.
- [5] HOBBS L, HILLSON S. Oracle Database 10g Data Warehousing[M]. USA: Elsevier Digital Press, 2005.