摘　要

在机载综合显示系统中，由于飞行状态复杂，对飞行数据的处理和显示都提出了比较严格的时间要求：需要在显示器的一帧时间内完全计算数据并完成显示任务。因而在工程中，硬件设计的运算速度成为至关重要的因素之一。

本研究论文在分析了国内外综合图形显示系统的发展情况后，提出了一种以FPGA+DSP 进行高速图形运算以及显示的方法，并以这种思路为基础，介绍了一种图形综合显示系统的解决方案。主要深入研究了在综合图形显示系统中比较重要的显示帧存控制器的设计。在仔细介绍了显示控制器的工作原理的同时，结合 FPGA 器件特点，给出了具体的解决方案和设计细节。

关键词：　机载综合显示系统、FPGA、帧存控制器、高速图形运算

Abstract

Because of the complicated flight state of the aircrafts, there rests a critical demand for high speed operation of flight data in the Synthetical Graphical Display System : We must finish all computations of the flight data and send them into the display panel in a very short time which is critically specified by the display instruments. Therefore, the operation speed of the hardware becomes to be very importance.

After the analyze of the current development situation of Synthetical Graphical Display System both at home and abroad , The thesis dissertates a solution of high speed graphical operation and display based on the DSP and FPGA instruments. Also, the thesis practices a further investigation on the design of the Frame Display Controller which is a very important part of the Synthetical Graphical Display System. Some details of this design is also be provided in this thesis.

Keyword : Synthetical Graphical Display System 、FPGA、Frame Display Controller、High-Speed Operation of Figure.

第一章　关于图形综合显示系统

1-1　机载图形显示系统综述

传统的飞机座舱显示系统中图形显示器较多，且大多为 CRT 的笔划式显示系统，这种显示方式有几个缺点：1、笔划式 CRT(cathode-ray tube)显示器只能显示文字和图形，而不能显示图像，当需要背景图像叠加时，用笔划式 CRT 很难设计。2、笔划式 CRT 受画线速度的限制，不能显示太复杂的图形。3、飞机座舱中的显示仪表较多，飞行员需要扫视和搜索仪表来获取信息，这种扫视和搜索对飞行员来说是一个沉重的负担。

在国外，飞机显示系统已发展到第六代：即采用光栅扫描式液晶显示器，并普遍采用综合显示技术。所谓综合显示是指：飞机的显示器上显示的图形根据飞机状态显示与当前飞行状态有关的图形信息，而那些与当前飞行状态关系不大的参数暂时消隐，这种显示图形的切换是自动或半自动进行的。

但是，光栅式显示器的应用使图形显示系统的设计难度大大增加。采用光栅式显示器的实时图形显示系统对系统的硬件和软件提出了更高的要求，国内研制飞机图形综合显示系统的技术难点主要是实时性的问题。光栅扫描式显示器要求的刷新率较高，一般要大于 50Hz。在做图形填充时，需要大量的二维矢量计算和反走样(Antialiasing)计算，一个 50Hz 刷新(即在 20ms 中要完成所有的处理功能)，1000x1000 个象素的显示帧需要每秒大于 2.6GFlOPS 的运算速度，即使现在最高速的 TMS320C67 系列和 ADSP2116X 系列的 DSP 也难于胜任。

1-2　国产机载综合显示系统研发现状

国外的图形综合显示系统中，使用了专用集成芯片 ASIC(Application Specific Integrated Circuit)，使得显示系统的图形处理速度大大提高。目前，国内在图形综合显示系统的研究中，基本上仍局限于算法的研究，诚然，先进的算法能产生事半功倍的效果，但硬件运算速度的瓶颈仍然没有突破。因此我们要研究新的图形生成电路结构和自己的专用集成芯片。

国内研制飞机图形综合显示系统的技术难点主要有下面两点：

(1) 实时性的问题。光栅扫描式显示器要求的刷新率较高，一般要大于 50Hz。而座舱图形综合显示系统是一个对实时性要求很高的系统，因为对于飞机来说，飞行状态变化快，座舱内图形显示出现可觉察的延迟或引起读数误差，都有可能酿成重大飞

行事故。以图形的显示刷新率为 50Hz 为例，要保证画面显示连续，所有的图形变换必须在 20ms 内完成。受硬件运算速度的限制，国内研制的飞机图形显示产品尚不能正常地显示全姿态指示器(或称天地球)等需要大量图形填充的画面。

(2) 反走样的问题。由于光栅扫描式显示器是用一个个离散的像素来近似地显示直线的，这里涉及到一个走样的问题，如果不做处理，一个斜的直线事实上会出现一个个小台阶，所以需要对每个点进行反走样处理。一个好的反走样算法能显著地改善屏幕上线段的视觉效果，但好的算法往往以增加运算复杂度为代价，这些算法都要大量的矩阵运算和函数运算，受硬件处理速度的限制，国内产品在罗盘旋转时，只能以牺牲旋转步距为代价。

1-3 用 DSP 加 FPGA 开发国产机载综合显示系统的设计

由此提出了一种新的设计思想：用 FPGA 设计专用的可重构的图形处理芯片。重构处理是指用现场可编程 FPGA(Field Programmable Gate Array)作为处理元素，用固定的硬件实现多种多样的可编程解法。在系统整个运行过程中能实时地改变硬件的配置，以实现不同的算法。一般由传统处理器执行主程序，特定的任务赋给以 FPGA 为基础的协处理器以加速它们的执行。在重构的过程中，根据需要，任务可以交换进入协处理器进行处理。重构的特性对提高电子信息系统的实时处理能力、自适应能力、可靠性、降低硬件系统的规模和功耗具有重大实际意义。

在座舱综合显示系统中，每个画面的处理算法有其特殊性，如在显示姿态画面时，将采用复杂的填充算法。而在显示全罗盘画面时，将采用罗盘刻度线和特殊的适合小号汉字旋转的反走样算法。由于不同画面是分时显示的，而且在切换画面时的时间要求是毫秒级的，可以设想，如果将填充算法、反走样算法等用 FPGA 硬件实现，并在实现中采用并行性设计，使实现这些算法的时间大大减少。在主处理器程序中，把 FPGA 实现的协处理器当成函数一样调用，并在显示画面切换时，由主处理器将不同的算法配置导入 FPGA，实现 FPGA 的动态重构。FPGA 的功能在整个显示过程中将动态地改变，这种方案既大大加快了图形处理的速度，同时又使硬件规模和成本大大降低。采用这种方案能有效地解决上述由于硬件处理速度瓶颈而引起的技术难点。

如前所述，国内在综合显示系统研制中的突出问题是硬件处理速度跟不上，必须研制自己的专用图形处理芯片。但 ASIC 设计周期长，成本高，风险大，并且结构固定，一般不具有再编程的能力，只能实现某个单一的算法，系统的可扩展性差。FPGA 是一种可动态再编程的器件，它一般基于 SRAM(Static Random Access Memory)，从外置存储器中导入不同的配置后，FPGA 的功能随即改变，是一种可在线编程的器件，一般的配置时间为几个毫秒到几十毫秒。用 FPGA 取代 ASIC 具有设计周期短、可重构和扩展性好等优点。座舱图形综合显示系统中，图形填充和线条的反走样计算是调用最多的计算，在全罗盘画面上，每个点都需要进行反走样计算后再写屏，如将反走样

算法的计算、旋转矩阵乘积运算及图形填充等用硬件实现，使 FPGA 成为一个专用协处理器，主处理器象调用函数一样调用该协处理器，则整个图形帧的处理时间将大大缩短。在专用芯片设计中，将重点研究填充算法与反走样算法如何在 FPGA 中实现，研究如何将算法设计成并行的多级流水线结构及 FPGA 的低功耗设计。

随着显示器的大规模化和分辨率的提高，高速帧存结构必须研究，因为最后要显示的像素都以帧为单位放在帧存中，随着所要传输的像素规模的提高，如何减少传输时间是一个值得研究的问题。图形显示系统设计是面向帧存的设计，帧存的设计在图形显示系统中是至关重要的。

将 FPGA 设计成协处理器，由主处理器将之当成函数来调用，可以大大加速整个系统的运行速度。协处理器花的时间主要在接口上，如何设计一个高速的接口也是本课题所要研究的一个内容。这里的高速接口研究不仅研究如何提高数据传输速度，还涉及到一个如何定义并行通讯协议的问题，即图形芯片与 DSP 之间的通讯模型的研究。

1-4 现场可编程逻辑 FPGA 在本课题中的主要应用

综上所述，本课题就是用 DSP（选用的是 ADSP2106 系列浮点 DSP）作为中央处理器，进行主要的计算处理。而使用 FPGA 做图形处理协处理器，把一些运算中常用到的，耗用大量 CPU 时钟的运算（如矩阵相乘运算，反走样运算，填充算法等）以软核的形式做到 FPGA 中去，利用动态可重构的原理，实现各个软核函数的分时复用，从而提高整个系统的性能，关于这一点，在上面的内容中已经有了介绍。

此外，在本课题中，使用 FPGA 设计 LCD 帧存显示控制器也占有相当重要的比重，光栅式液晶显示器是一种基于帧存缓冲的显示方式，也就是说显示器上面的每一个象素点都要对应于帧存缓冲中的一个单元，比如对于彩色液晶显示器上的每一个象素点来说，就需要在帧存缓冲中留出三个字节（3 * 8 bit）分别用于表示该象素的 RGB（物理三基色）值。在每一个显示周期中（一般为 50HZ,20ms）,LCD 在一定的时钟控制下，扫描全部帧存缓冲，从而决定整个显示屏的显示画面。在本系统中，所有的速度要求都是由 LCD 的显示周期来决定的，在 50HZ（20ms）的周期里，不光计算模块要根据飞机的飞行参数计算出 LCD 上每个象素相应帧存单元中应该放的值，还要将这个值正确地放入帧存缓冲中，由此可见，帧存控制器设计的好坏，将直接影响到系统的速度及稳定性。

选用 VirtexII 器件作为协处理器，需要考虑到动态重构时程序载入时间的大小，因为在动态重构时，DSP 处于等待状态，只有等到 FPGA 程序载入完成，DSP 才能把 FPGA 看作是可以使用的函数模块，因此这一段载入时间是无效时间，如何想办法减小这一段时间也是十分重要的。

在 VirtexII 器件的程序载入模式中，有串行模式和并行载入模式，比较常用的

是 BoundaryScan(JTAG)和 SelectMap 模式，其中 SelectMap 属于并行模式，并行模式的有点在于速度快，可以大量缩短由程序载入而带来的无效时间。在 VirtexII 器件的并行载入中需要根据载入时序专门设计载入控制器以实现动态可重构的功能，通常这种控制器可以用小规模的可编程逻辑实现，在本课题中选用的是 XILINX 公司的 CPLD 产品 XC9536 芯片。

图 1—1 是本图形综合显示系统的设计原理图，各部分详细解释如下：

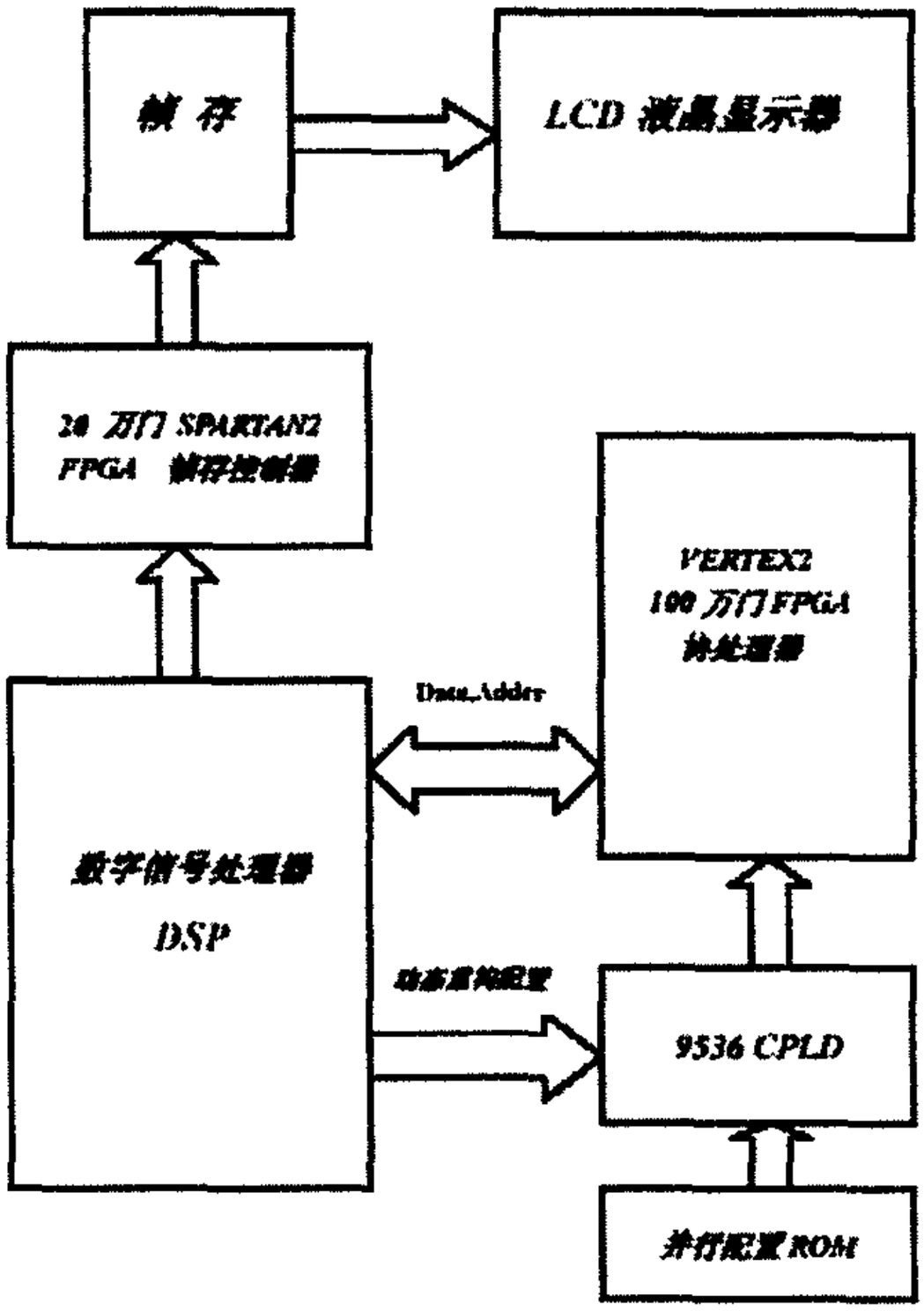


图 1—1

1）帧存由两片 512K × 8 的静态 SRAM，组成。由于帧存对速度的要求比较高，所以采用两片 RAM 交替使用的方式，在一片 SRAM 进行读操作，即向 LCD 送数的同时，对另一片 SRAM 进行写操作，把下一屏的数据写入，从而缓解显示速度对数据处理的压力。

2）采用 XILINX 公司的 SPARTAN2 器件实现帧存控制器，其功能控制两片帧存 SRAM 的自动交替，以及实现两片 SRAM 同 DSP 和 LCD 显示器的分时复用接口。

3）LCD 液晶显示器具有 640×480 个象素，即需要 307200 个存储单元，由此可见，所采用的两片 512K×8bit SRAM 满足容量要求。

4）数字信号处理器是整个系统的核心，它负责主要的数值计算任务，并且提供与帧

存控制器和图形协处理器的接口，还要提供控制信号控制协处理器动态可重构的功能。

5）由 XILINX 公司的 100 万门 VirtexII FPGA 芯片制作图形处理协处理器，VirtexII 系列芯片具有门数大，器件延迟小，具有内置式多位硬件乘法器，具备动态可重构功能等优点，比较适合算法的实现，可以尝试将图形计算中经常用到而且占用大量 CPU 工作时间的算法放入其中，作为函数，由 DSP 调用。

6）利用 XILINX 公司 9500 系列 CPLD 控制 VirtexII 期间的程序导入。VirtexII 器件有 4 中配置模式：主串(Master serial)、从串(Slave serial)、JTAG、SELECTMAP，在这里，选用 SELECTMAP 并行模式，并且由 CPLD 控制。

7）并行配置 ROM 中放有配置 VirtexII 图形处理芯片的载入程序，不同的程序放在不同的起始地址处，根据 DSP 运算的不同需要，由 CPLD 控制，将相应的程序载入到 VirtexII FPGA 中，从而实现动态重构。

1-5 论文主要研究内容

本人的硕士研究课题是上述项目的一部分，主要内容包括：

1）选用 XILINX 公司 SPARTAN2 芯片设计显示帧存控制器，用于控制 DSP 器件和协处理器之间的各种逻辑协调关系以及对后续显示器的显示控制。包括帧存控制器平台硬件设计和使用 VHDL 硬件描述语言编写帧存控制程序，并综合下载到 SPARTAN2 FPGA 中实现。

2）单片机模拟电路，由于本项目是由 DSP 和 FPGA 两部分构成，在 DSP 部分开发完成之前就要进行 FPGA 的开发，因此需要用单片机模拟简单的 DSP 功能，用于验证 FPGA 控制芯片的功能。为了验证帧存控制器的功能，设计一块单片机电路，模仿 DSP 控制显示电路,在 LCD 上进行图形显示。

3）选用 XILINX 公司 CPLD 器件 设计一片控制器，控制协处理器 FPGA 的 SRAM 配置。采用 SELECTMAP 配置模式，实现 VirtexII 器件的动态重构。

4）参与用 XILINX 公司 VirtexII 芯片开发协处理器的工作。

第二章　FPGA 在图形综合显示系统中的应用

2-1　使用 FPGA 制作帧存控制器，实现 LCD 的显示功能

2-1-1　显示器 LCD 的显示原理

在课题的研究中，对 LCD 工作模式的学习和分析是所有工作的基础，因为：首先研发的最终目的就是在 LCD 上没有凝滞地显示图形，对于系统的处理速度的要求主要取决于 LCD 的工作模式；其次，将 LCD 帧存中的数据正确显示到 LCD 上面，需要严格地遵守 LCD 的时钟标准。在这里首先以 NEC **NL6448AC33-18** 液晶显示器为例，对 LCD 的工作原理进行介绍。

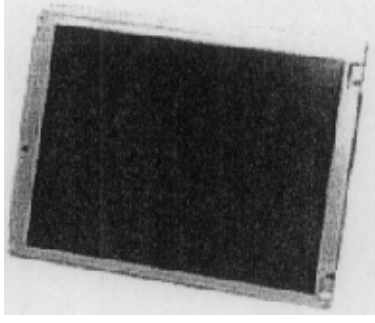


图 2－1 NEC NL6448AC33-18 液晶显示器

NEC NL6448AC33-18 液晶显示器是 26 cm 640×480 象素彩色液晶显示器，3.3V 供电，6bit×3 数字 RGB 数据信号在严格的时钟控制下被送到 LCD 面板，象素在 LCD 上面是按照矩阵的形式二维排列的，而在帧存中的存放形式是一维的，也就是说，LCD 中第一行的 640 个象素占据帧存中从 0 到 639 个地址单元；那么第二行的 640 个象素就要占据帧存中从 640 到 1279 个地址单元，第三行、第四行依次顺延，直到占满 640×480 个地址存贮空间。在 LCD 的显示中，在一帧的显示周期里，LCD 对所有的帧存空间进行一遍扫描，从而将帧存中存储的一维数据映射成 LCD 上的二维图象。

要使 LCD 正常地工作，严格地遵守显示时序非常重要，LCD 在四根重要的控制时序信号的作用下进行工作，如图 2－2 所示：

CLK：　点时钟信号，在这个信号的控制下，LCD 从帧存中读出一个数据单元，并将其显示到面板上的相应位置，频率范围在（21MHZ-29MHZ），标准频率是 25KHZ 时钟，占空比为 0.5。

Hsync:行同步信号，在一行显示完后（即显示了一行 640 个象素之后），LCD 需要一个行同步信号，通知 LCD 一行结束，另起一行显示。

Vsync:帧同步信号，在一帧显示完后（即显示了一帧 640×480 个象素后），LCD 需要一个帧同步信号，通知 LCD 一帧结束，清屏并重新开始显示新的一帧画面。

DE　:数据使能信号，只有在此信号为高电平的情况下，LCD 才能在时钟 CLK 的控制下工作，否则处于等待状态。

此外，还有三组数据信号线： R0-R5;G0-G5;B0-B5 分别代表 RGB 物理三基色值。下面就信号时序图进行详细分析：

行显示信号时序图：

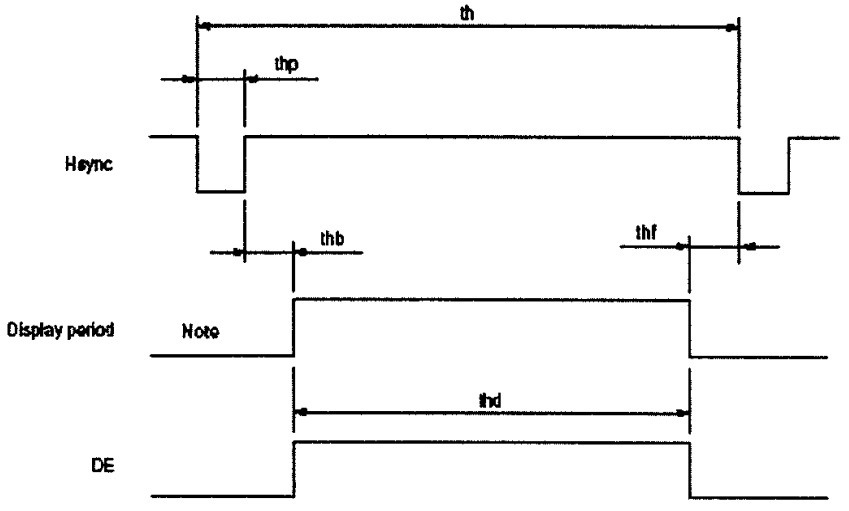


图 2－2　行显示信号时序图

th ： 行同步信号周期，一般为 800 个 CLK,约 32us。

thd: 显示周期，是 DE 信号有效的时间，应该为 640CLK。

thp: th 信号脉冲宽度，为 96CLK。

thf: 信号前间，为 16CLK。

thb: 信号后间，为 48CLK。

在行同步信号到来后，LCD 需要有一段时间准备，然后当 DE 信号有效时，开始接受数据。这一段时间就是信号后间 thb (back porch)。

同时，在一行数据接受完毕后，DE 信号由低变高，LCD 也需要一段时间准备，等待新的行同步信号的到来。这一段时间叫做信号前间 thf (front porch)。　图 2－2 就是行显示信号时序图。

帧显示信号时序图：

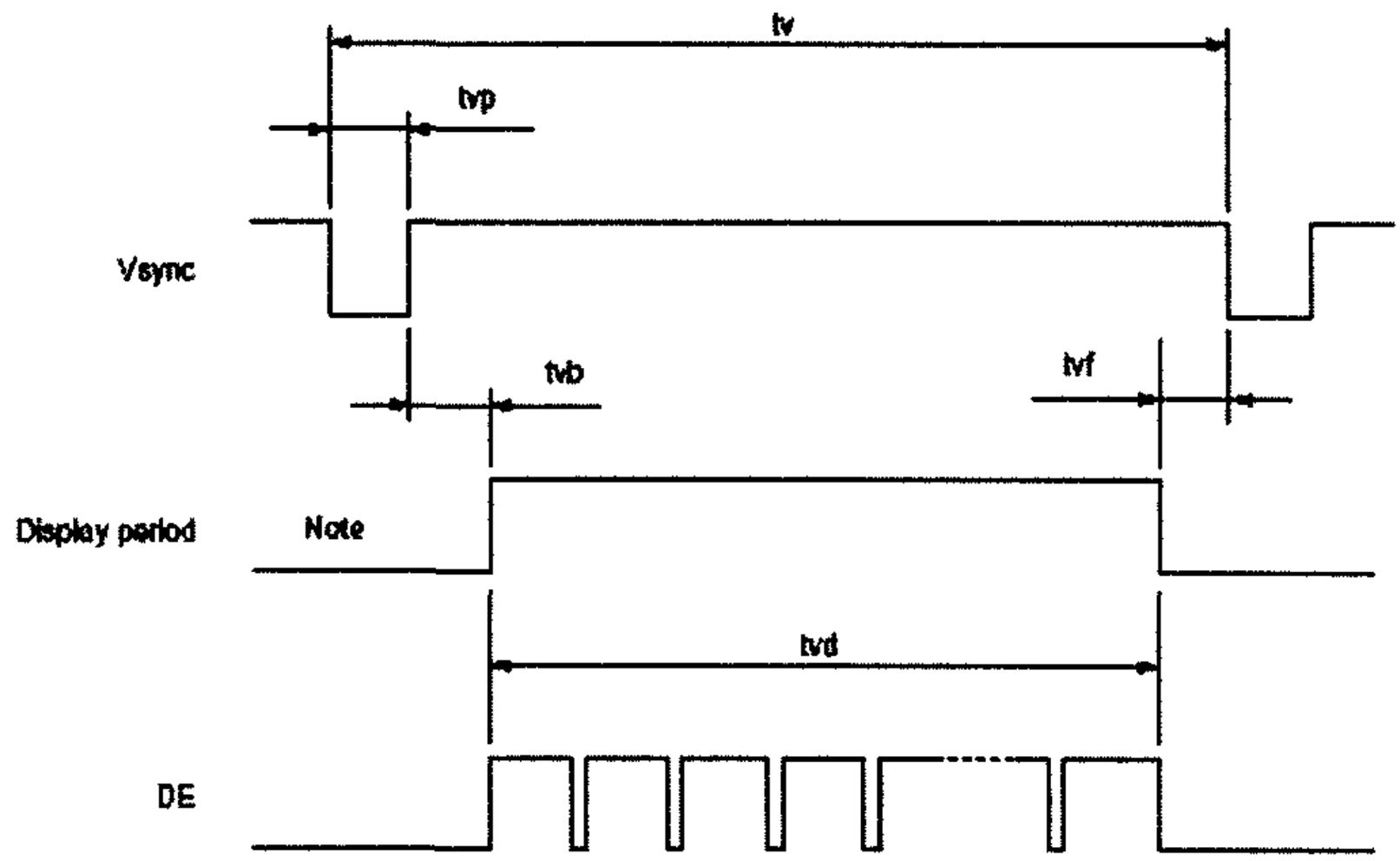


图 2—3 帧显示信号时序图

每一帧图象由 480 行组成，所以在完成了每一行信号的显示周期后，还需要了解帧显示信号时序。图 2—3 就是帧显示信号时序图。

tv ：帧显示周期，为 525 个行同步周期，约 16ms —20ms ，即满足大于 50HZ 的显示速度。这个参数对于整个系统都具有重要的意义，也就是说，要在这一段时间内(大约 59.3HZ)，完成所有的数值计算和数据显示。

tvd：数据使能时间，应该为 480 个行同步周期时间。

tvf：信号前间，为 12 个行同步周期时间。

tvb：信号后间，为 31 个行同步周期时间。

tvp：帧同步信号脉冲宽度，为 2 个行同步时间。

2-1-2 由帧存控制器实现 LCD 的显示

在上面一节中详细介绍了 LCD 的工作方式以及相应的时序模式。可以看到，LCD 只负责把外部送过来的满足一定格式的数据，在严格的时钟控制下，显示到 LCD 上面。需要注意到一下几个特点：

- ✧ LCD 本身不带数据产生单元，而需要外部器件将数据按照一定的格式和时序送给 LCD 面板。
- ✧ LCD 上没有数据存储单元，而需要外部 RAM 作为帧存。
- ✧ 外部处理后的显示数据需要首先存入帧存 RAM，然后再扫描显示到 LCD 上。
- ✧ 怎样将帧存 RAM 中的数据送给 LCD 面板，需要按照 LCD 的标准时序严格控制。

所以，需要设计专门的帧存控制器，把外部帧存同 LCD 连接起来，满足以下的功

能：

- ✧ 连接中央处理器 CPU 和帧存，从 CPU 中接收数据，放入帧存 RAM 中。
- ✧ 连接 LCD 和帧存，把帧存中的数据送到 LCD 面板。
- ✧ 产生 LCD 需要的各种时钟（比如行同步、帧同步信号等），并根据时钟信号把从帧存中读到的数据依次送往 LCD 面板。
- ✧ 为了提高帧存的效率，采用双 RAM 形式，即利用两片 SRAM 进行分时复用，当 CPU 通过帧存控制器对一片 SRAM 进行写操作时，帧存控制器同时对另一片 SRAM 进行读操作，并把读出的数值送往 LCD 面板显示。

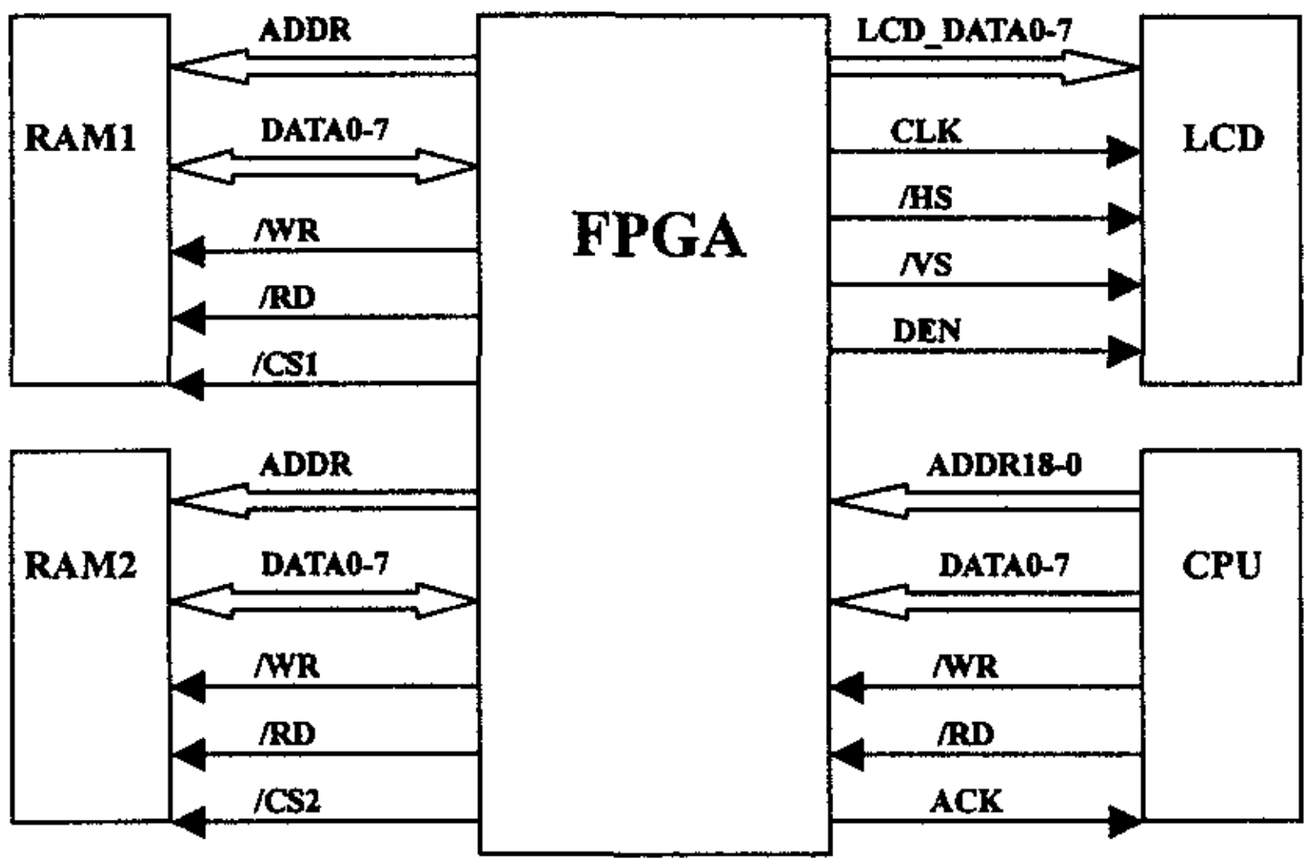


图 2—4 帧存控制器功能框图

如上图所示，一片 SPARTAN2 FPGA 同两片 512K×8bit 的 SRAM、中央处理器 CPU(高速 DSP 或单片机)、LCD 显示器连接。由 CPU 计算出 LCD 上每一点相应的值，然后送给 FPGA，FPGA 根据分时复用的原则，判断将此数据送往哪一片 SRAM，与此同时，将另一片 SRAM 中的数据送给 LCD。当一帧数据送完之后，FPGA 将两片 SRAM 的角色调换，再进行下一帧数据的接收和显示。

比如，在 T(0)时刻（周期为 1 帧的时间）。FPGA 把 SRAM1 中的数据（为一帧的显示数据）顺序读出并送给 LCD 面板显示，同时从 CPU 中接收数据（应为下一个帧周期显示的数据）顺序放入 SRAM2。当 T(0)周期结束，T(1)周期到来时，FPGA 转向 SRAM2，把 SRAM2 中的数读出并送给 LCD，同时把从 CPU 接收到的数据顺序放入 SRAM1。当 T(2)周期到来时，又重新回到了 T(0)的状态。

2-1-3 由单片机验证帧存控制器的工作

由于本项目是由 DSP 和 FPGA 两部分构成，在 DSP 部分开发完成之前就要进行 FPGA 的开发，而 DSP 部分的设计更为复杂和困难，不可能由 DSP 来对 FPGA 进行

功能验证。因此需要用单片机模拟简单的 DSP 功能，比如在 LCD 上画一条线或者画一个静态罗盘。为了验证帧存控制器的功能，设计一块单片机电路，模仿 DSP 控制显示电路,在 LCD 上进行图形显示。

在这里的单片机电路比较简单，只需要设计一个最小系统板。而比较重要的是设计接口，由图 2—5 可以看到，单片机同 FPGA 的接口线一共需要 30 根线，它们是：

- ✧ 19 根数据地址线，由于单片机要控制的帧存是两片 512K×8bit 的 SRAM，故需要访问到所有的地址空间，就需要用到 19 根地址线，这 19 根地址线可以由 P0 口、P2 口和 P1 口的低三位组成。
- ✧ 8 根数据线。要注意到单片机的数据线 P0 口是时分复用线，因此需要用到 373 地址锁存，但是在与 FPGA 接口时，需要将数据线和地址线分开来连接，也就是需要把 P0 口接到 373 地址锁存器的同时，还要直接连接到 FPGA 上。这一点在 MS51 单片机的相关资料上有详细的说明。
- ✧ /WR:用于对外部 RAM 进行写选通；
- ✧ /RD：用于对外部 RAM 进行读选通，它和/WR 信号是一对互逆信号；
- ✧ ACK：反馈信号，它的功能是 FPGA 通知单片机，一帧的数据显示完了，要求开始送下一帧的数据。

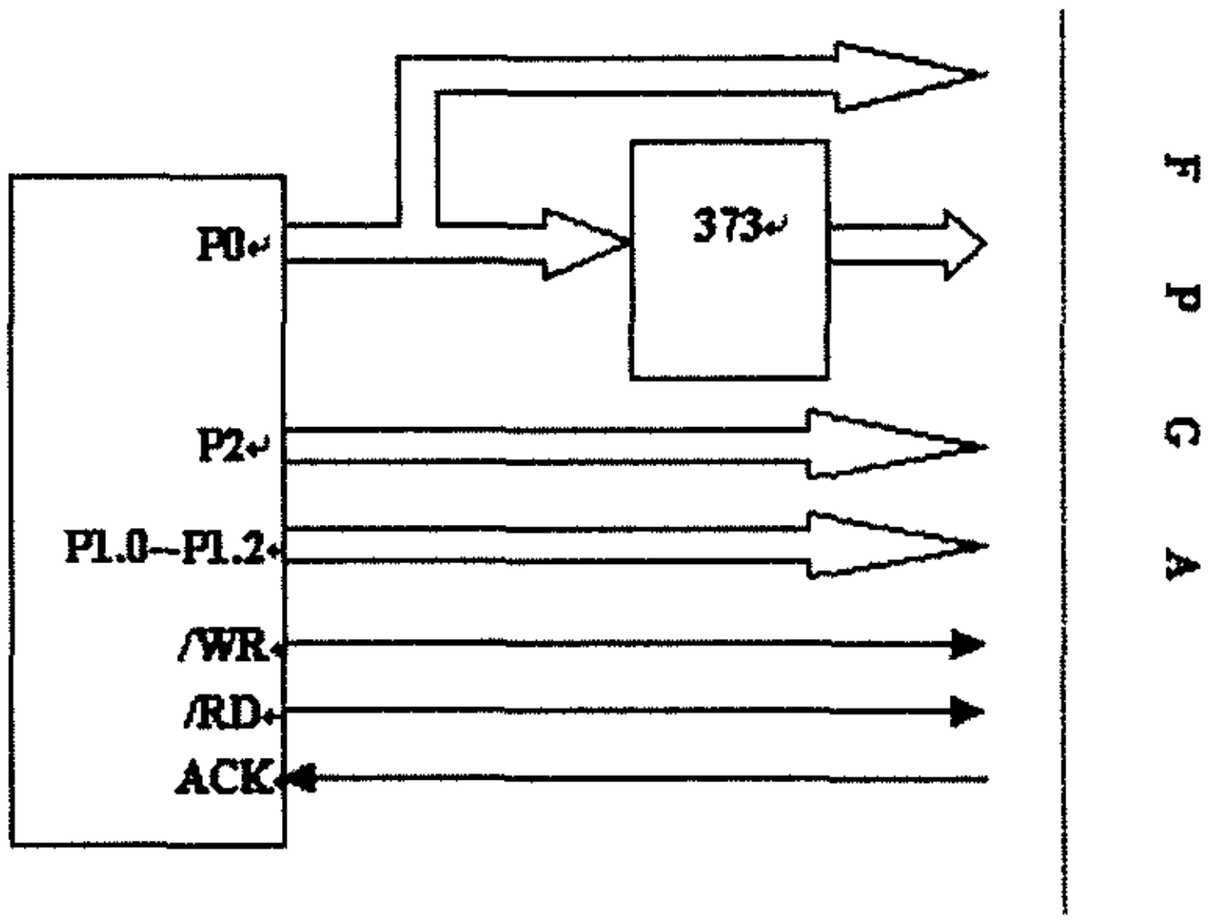


图 2—5 单片机同 FPGA 的接口设计

应该注意的是，由单片机代替 DSP，整个系统的工作速度并没有改变，还是由 LCD 的显示特性所决定的，帧同步信号（等同于反馈给单片机的 ACK 信号）频率应大于 50HZ。然而，由于单片机的工作频率比较低，速度较慢，在 20ms 的周期里不能完成大批量的数值计算，如反走样算法、汉字的旋转、天地球填充等复杂的计算都不可能用单片机来完成。单片机的作用仅仅是验证 FPGA 的设计能否满足 LCD 的显示时序逻

辑。比如用一个简单的画线算法，在 LCD 上显示直线，或者只是简单地向 FPGA 送一些简单的，事先计算好的数值。但是，除去了数据计算环节，单片机的速度已经足够满足 LCD 要求的时序规格，所以，用单片机代替 DSP 来测试 FPGA 的显示功能是完全可行的。

2-2 利用 FPGA 制作图形处理协处理器

2-2-1 协处理器实现的功能

研制的图形显示系统拟采用数字信号处理器 DSP+FPGA 的结构，DSP 拟采用 AD 公司的 ADSP2106X 系列浮点数字信号处理器，FPGA 将采用 XILINX 公司的 VirtexII 系列。DSP 通过高速接口电路访问 FPGA 协处理器，并行配置存贮器提供 FPGA 不同的算法配置，FPGA 中导入算法配置后，FPGA 就是一块专用的图形处理芯片，本系统中用 DSP 控制何时将什么算法导入 FPGA。FPGA 所实现的专用图形处理芯片，在整个系统工作过程中是动态变化的，也就是可重构的。

研究中将分析综合显示系统中所用到的各种填充和反走样等算法的调用情况，重点研究 FPGA 的算法实现。用 VHDL 语言现对处理算法做行为级的描述，并利用 ModelsimSE 或 Activ HDL 等仿真软件进行仿真分析，然后通过 XILINX 公司的 ISE 综合软件进行综合和门级仿真，在通过专用电缆下载到 VirtexII FPGA 芯片中。

2-2-2 协处理器与 DSP 的接口设计

协处理器与 DSP 的接口主要包括控制信号线和数据传输线，数据传输线需要根据 DSP 的工作模式和 FPGA 中所导入的具体算法来分析决定，在硬件设计上要将 DSP 的地址线和数据线以及一些外围控制 IO 同 FPGA 的通用 IO 连接起来，供以后设计具体算法时对 FPGA 编程决定其接口定义，具体内容已超出本毕业论文范围，在这里不作详述。

在这里主要讨论 FPGA 芯片的动态可重构设计，也就是要利用 CPLD 器件对 FPGA 器件的导入过程进行控制，这是 FPGA 同 DSP 接口最重要的部分之一。

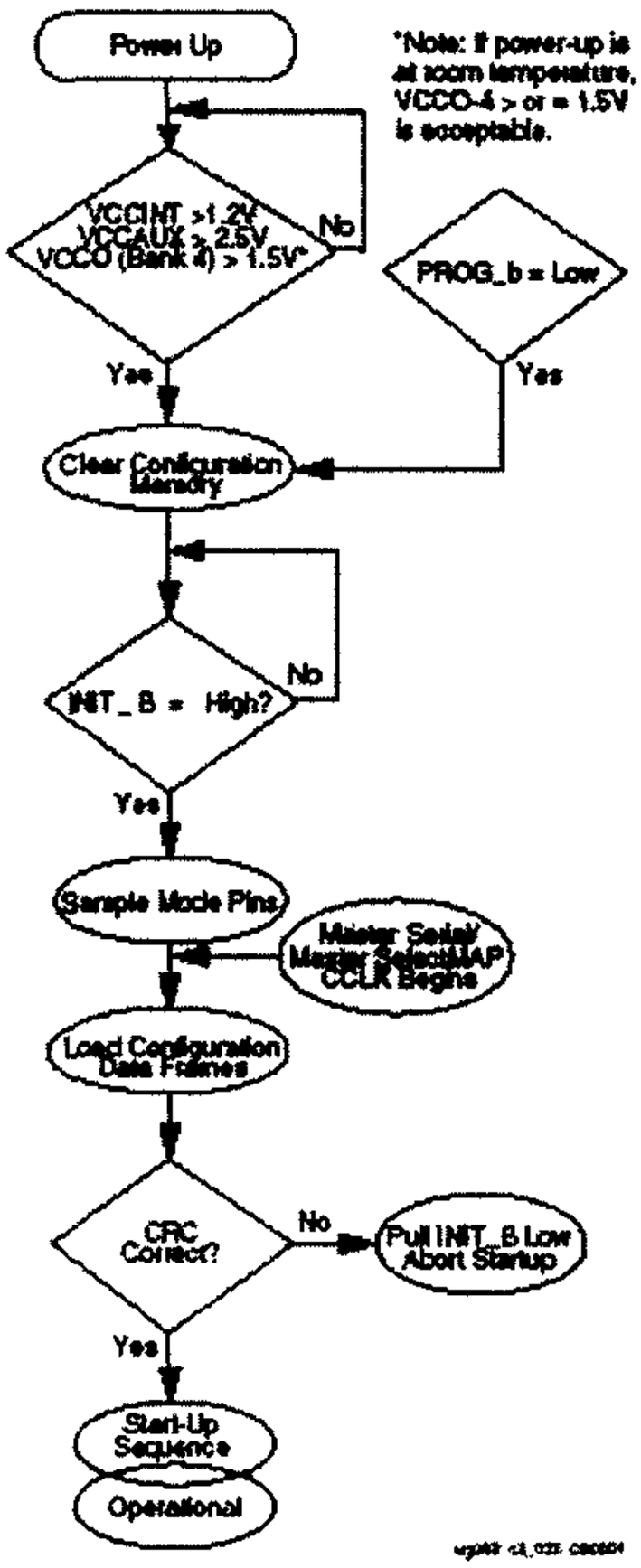
VirtexII 器件一共有四种配置模式：主串(MasterSerial)、从串(SlaveSerial)、SelectMAP、JTAG 边界扫描。这四种模式进行配置时，用户 I/O 可以悬空或指定一个值。首先，我们给出 VirtexII 的配置管脚，如表 2—1 所示：

表 2—1 VirtexII 配置管脚

名称	方向	功能描述
CCLK	输入/输出	配置时钟，主串模式时为输出
/PROG	输入	配置逻辑的异步复位
DONE	输入/输出	配置状态及启动控制
M2、M1、M0	输入	配置模式选择

TMS	输入	边界扫描节拍控制
TCK	输入	边界扫描时钟
TDI	输入	边界扫描数据输入
TDO	输入	边界扫描数据输出
DIN(D0)	输入/输出	串行配置数据输入
D0_D7	输入	SelectMAP 配置数据输入，回读数据输出
/CS	输入	片选 SelectMAP 专用
/WRITE	输入	写选通 SelectMAP 专用
BUSY/DOUT	输出	BUSY：SelectMAP 状态信号；DOUT:串行配置数据输出
INIT	输入/输出	延迟配置、配置错误信号

图 2—6 FPGA 配置时序



其中 CCLK、/PROG、DONE、M2、M1、M0、TMS、TCK、TDI、TDO 是专用的配置管脚，而 DIN、D0、D7、/CS、/WRITE、BUSY/DOUT、/INIT 是多用途管脚，在配置结束后可以用作用户 I/O。

VirtexII 的配置流程如图 2—6 所示。FPGA 器件的配置分为 4 步:清除配置存储器、初始化、配置、校验启动。 VCCINT 为 FPGA 的核心电压，VCCAUX 为 FPGA 的供电电压，如果 VCCINT>1.2V，并且同时满足条件 VCCAUX>1.5V,即对 FPGA 正常供电，VCCO 为 I/O BANK4 的 IOB 输出电压，就会满足 VCCO>1.5V。当 FPGA 达到这个操作电平且电路发送读写测试后开始延时，通常延时 16ms（当工作在主串模式时为 64ms），这种延时仅仅在上电复位时产生。

/PROG 为异步复位，当它为低时，清除配置存储器，使用片内振荡器，配置存储器的数据帧相继初始化，每帧初始化大约 1.3us，同时保持/INIT 为低；/PROG 变为高后，继续清除配置存储器直到完成，同时置/INIT 为高，在 2 个内部时钟后开始配置过

程，而此时我们也可以在外部保持/INIT 为低，延迟配置。配置的具体过程根据配置方式的不同而不同。

在整个配置过程中有两次 CRC 检查，一次是在最后一帧配置数据流装入之前，一次是在配置数据流装入的最后。如果出现错误，则/INTT 变低，放弃启动，这时就要重新设置/PROG 或重新加电来重新配置。如果 CRC 检查正确，则进入工作状态。在初始化和配置期间，HDC、/LDC、/INIT 和 DONE 四个引脚电平反映了系统接口的状态，FPGA 上电后，/LDC、/INIT 和 DONE 保持低电平，HDC 保持高电平，初始化完成后，/INIT 输出高电平，当配置完成 DONE 变高，芯片开始工作。

表 2—2 是 VirtexII 的四种配置模式对应的 M2、M1、M0 的值。

表 2—2 VirtexII 配置模式

配置模式	M2	M1	M0
主串	0	0	0
从串	1	1	1
SelectMAP	1	1	0
边界扫描	1	0	1

2-2-3 关于协处理器的动态可重构设计

在 Virtex 的四种配置模式中，主串、从串、边界扫描模式与 XILINX 以前的 FPGA 芯片没有什么不同，在这里我们不作详细讨论；SelectMAP 模式是一个 8 位的并行配置模式，类似与 XC4000XLA 和 SPARTANXL 系列中的快速（Express）模式，但又不尽相同，由于本课题中需要一种快速的配置方式，所以需要选择 SelectMAP 模式来进行配置。下面就详细介绍一下这种配置方法。

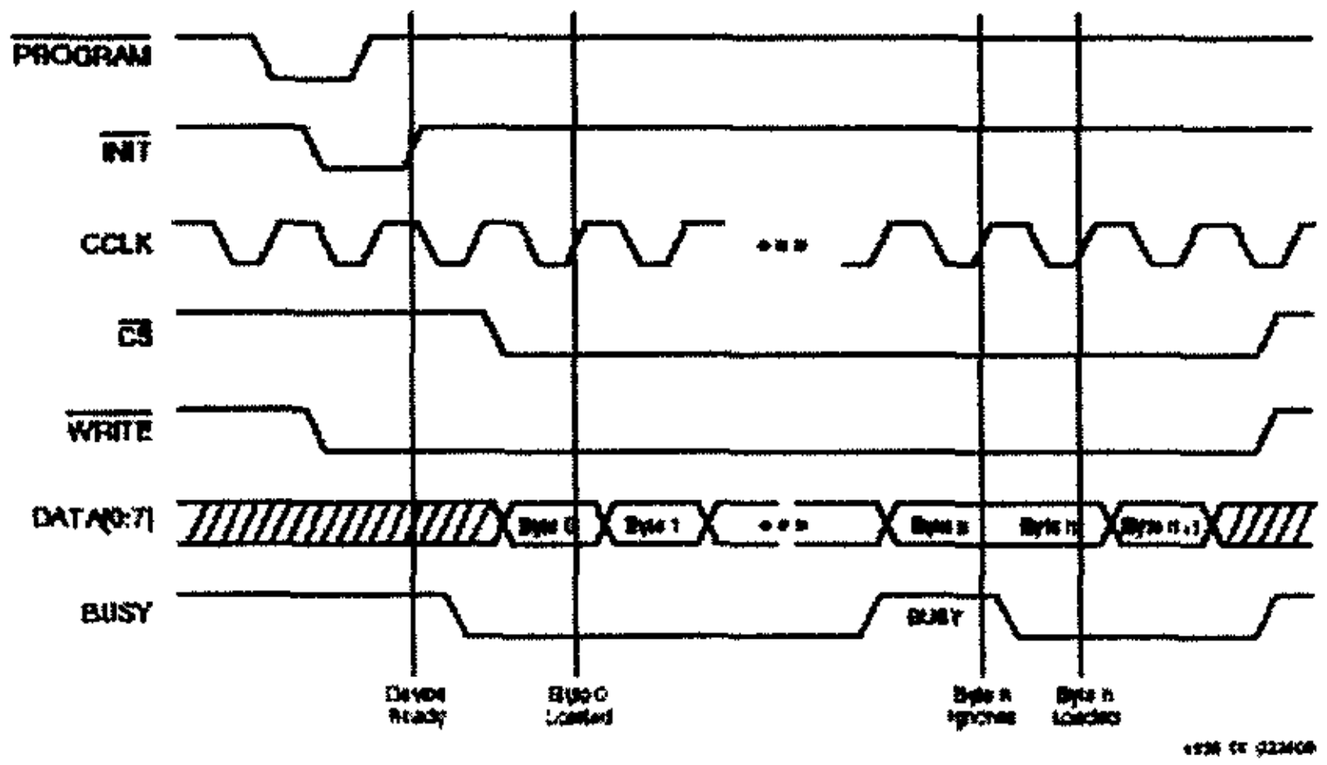


图 2—7 SelectMAP 端口配置时序图

SelectMAP 端口是一个 8 位的双向数据端口，可以通过它对 VirtexII 进行配置，也可以通过它回读（readback）配置数据。它是 VirtexII 的最快的配置方式，配置时钟可以达到 66MHZ。用 SelectMAP 进行配置需要 8 为数据线 D0-D7 和 7 个控制/状态信号，分别是 CCLK、/PROG、DONE、/INIT、/CS、/WRITE、BUSY。其中 BUSY 握手信号只有在 CCLK 大于 50MHZ 时才有用，在本课题中，CCLK 小于 50M，所以 BUSY 信号可以不用考虑。SelectMAP 端口配置时序图如图 2—7 所示。

在对 FPGA 进行配置时，大多数用户都选择从并行 EPROM 或者并行 FLASH 对 FPGA 进行配置，即用并行 EPROM 来存储配置数据流，因为并行 EPROM 通常拥有较大的存储空间，可以存储多个应用的配置数据流。这对需要重构的应用来说非常方便。目前在许多用户的设计中都是采用的这种方法。

为了方便应用，XILINX 提供了配套的并行 EPROM 系列 18V00，这是一种专门为了并行配置 XILINX 产品而设计的 EPROM 产品。它在对 FPGA 进行并行配置时，不需要进行地址控制，而默认从 EPROM 的最低地址开始配置，这种配置方法的原理图如图 2—8。

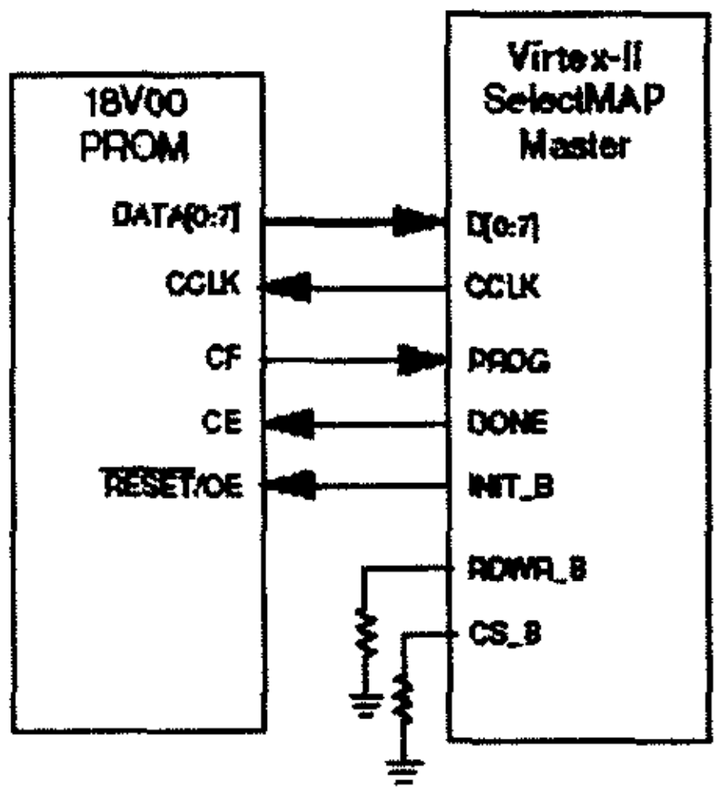


图 2—8 使用 18V00 并行配置 FPGA

这种配置方法简单有效，只需连 5 根线，就可以完成任务。但是，这种方法却有一种缺点：它不能把不同的程序都同时放入并行 EPROM（放在不同的起始地址处），然后根据需要调用不同地址处的相应程序，以实现重构功能。要解决这个问题，这里提供了一种从并行 EPROM 或 FLASH 对 Virtex 进行配置的方法。而这种方法相对来说复杂一点，需要加入地址控制功能，这种功能可以用 CPLD 来编程实现：要从并行 EPROM 配置 VirtexII，利用 SelectMAP 端口，但是 VirtexII 没有主并模式，这样就需要一个接口电路来产生 PROM 的地址和把配置数据装入 FPGA。我们用一片 XILINX CPLD 产品来实现，如图 2—9 和 2—10：

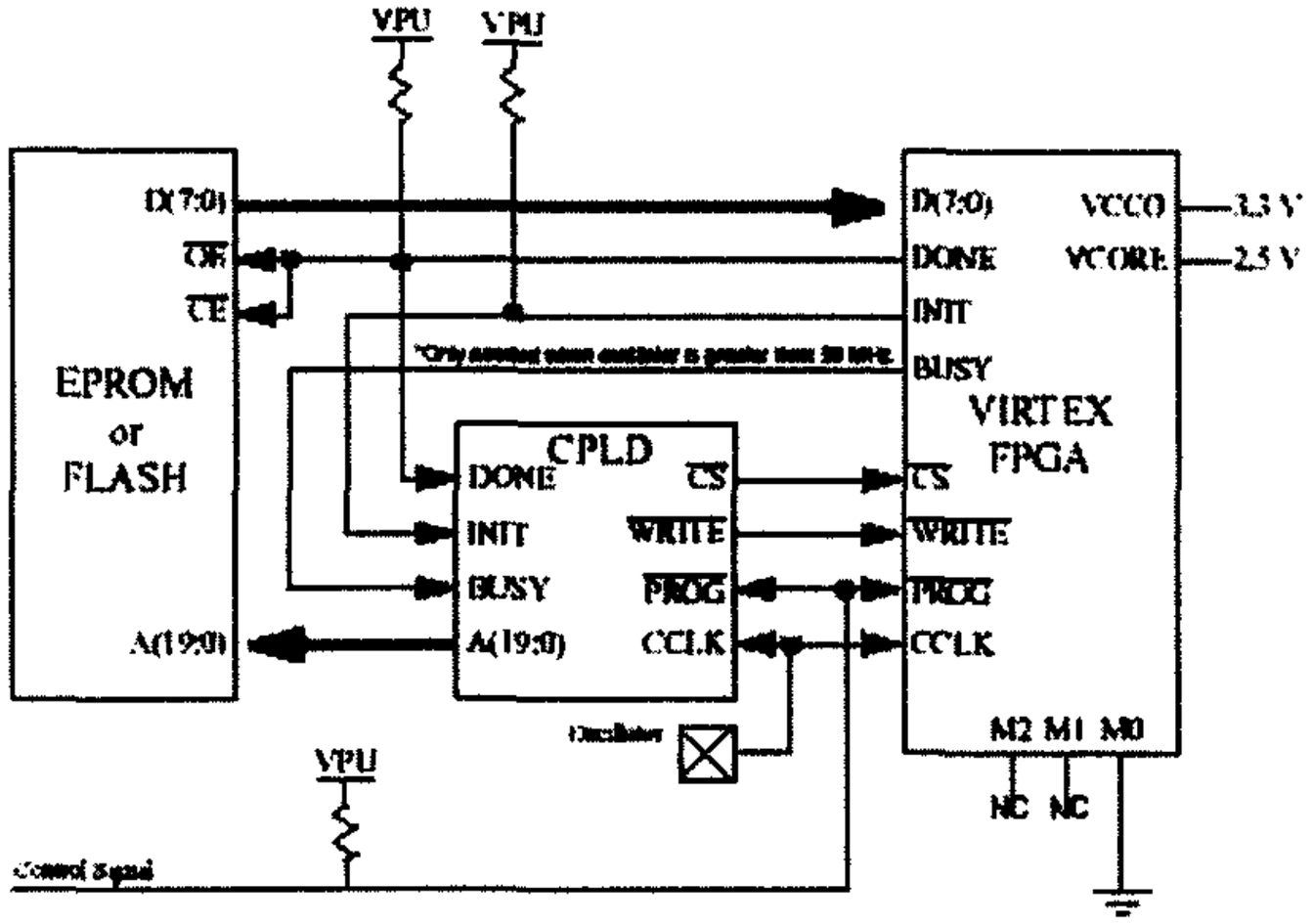


图 2-9 采用 CPLD 配置 VIRTEX FPGA 器件（单片 ROM）

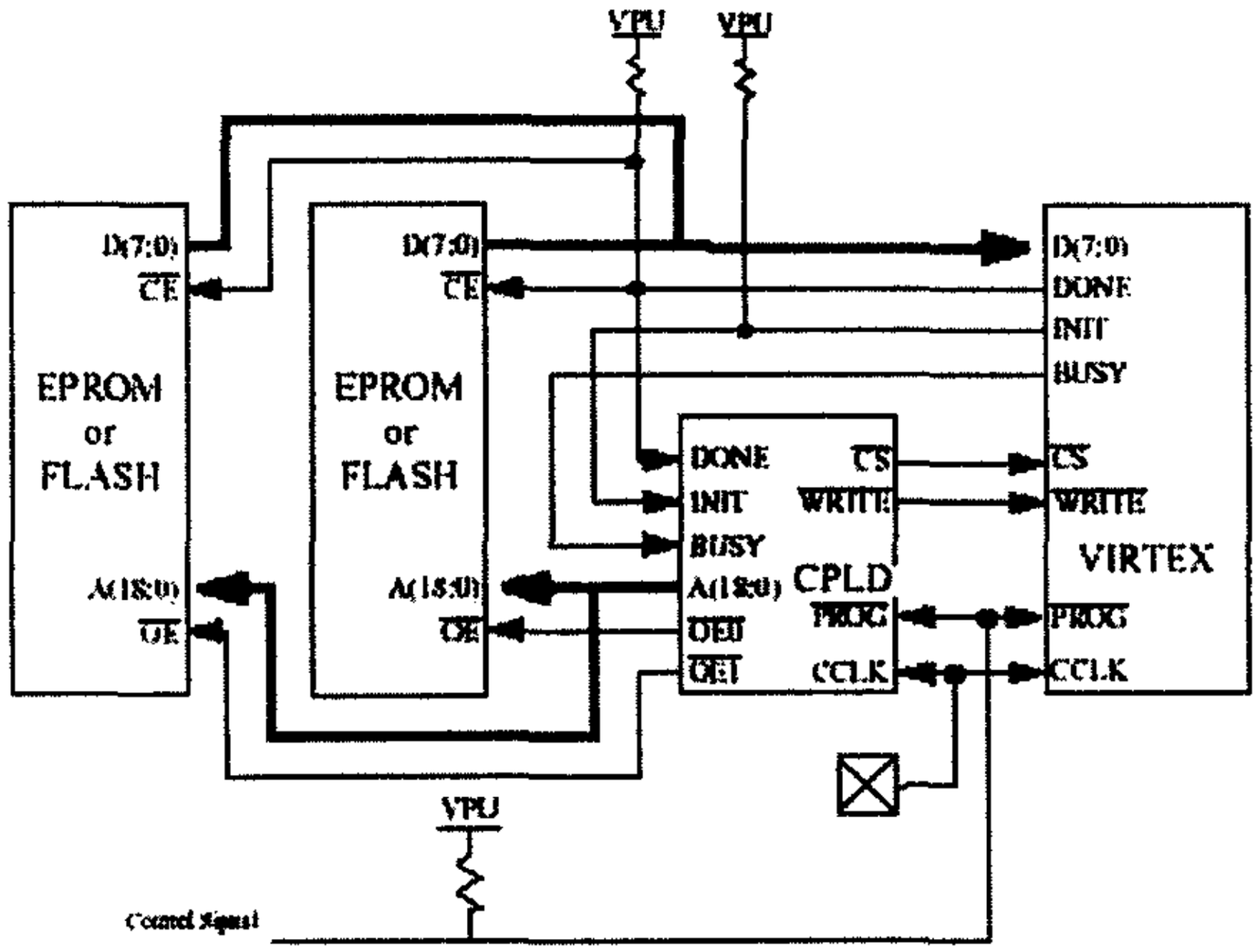


图 2—10 采用 CPLD 配置 VIRTEX FPGA 器件（多片 ROM）

图中 VPU 为 CPLD 和 EPROM 的工作电压，通常为 3.3V 或 4.5V，所接的上拉电阻一般采用 4.7 千欧。VirtexII 管脚内部接有上拉电阻，所以 M2 和 M1 悬空即为高电平。控制信号为外部产生（即由 DSP 产生），在配置失败时或重构时对配置逻辑进行异步复位。

振荡器的频率由下式确定：

$$振荡器频率=\frac{1}{Tacc-Tsmdcc}$$

其中 Tacc 为 EPROM 的存取时间，Tsmdcc 为 SelectMAP 端口输入数据的建立时间。典型的 EPROM 的 Tacc 为 100ns，VirtexII 的 Tsmdcc 为 2ns。所以振荡器的最大频率约为 9.6HZ，这个频率远远低于 VirtexII 所支持的 66MHZ 的配置频率和 CPLD 的 66MHZ 的工作频率，所以在这种方法下提高配置频率，关键是要采用高速 EPROM。

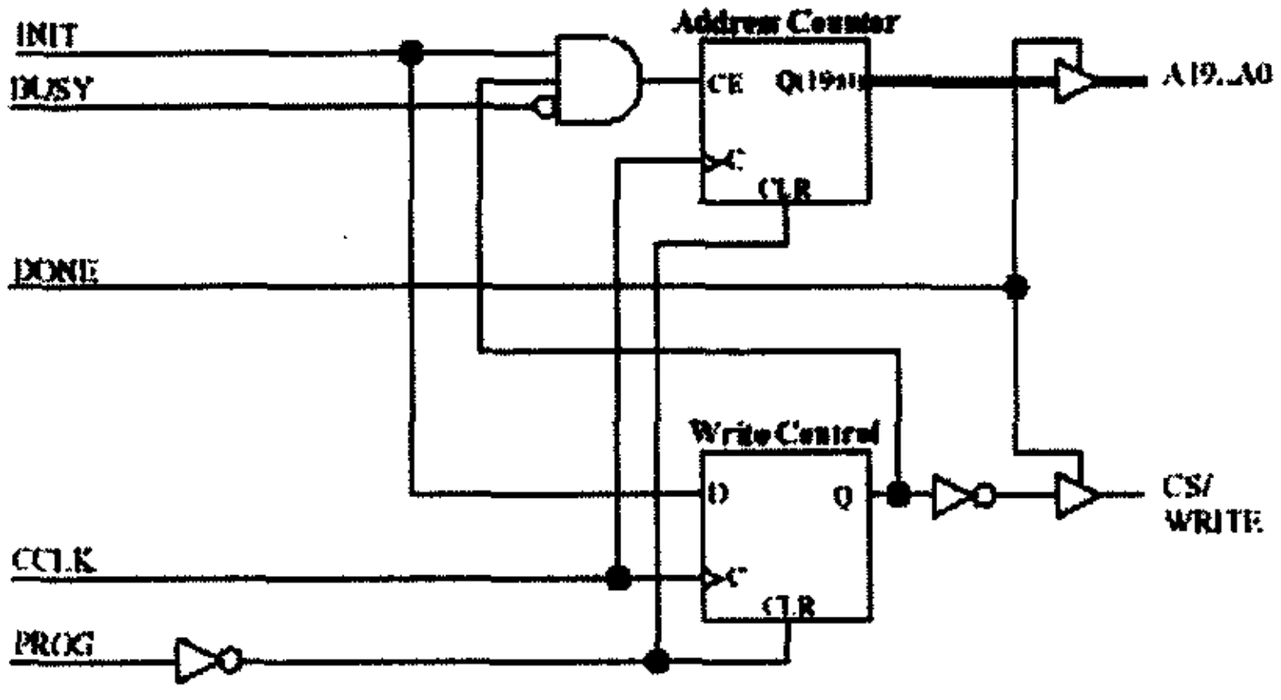


图 2—11 配置用 CPLD 内部电路

图 2—11 是 CPLD 内部电路，整个电路包含一个地址计数器、一个写控制寄存器、一些三态缓冲器和一些逻辑门。要注意的是地址计数器必须要为 EPROM 提供足够的地址位数，如果选用的 EPROM 是 1M 字节，总共有 20 根地址线。CS/WRITE 作为配置时的控制信号，在写配置数据时都要保持为低，对配置一片 FPGA 来说两者可以是同一个信号。

图 2—11 的 CPLD 内部电路也可以直接由 VHDL 语言编写下载，具体的内容见第五章内容。

第三章　XILINX公司FPGA产品介绍以及开发软件介绍

3-1　可编程逻辑器件简介

PLD是可编程逻辑器件（Programable Logic Device）[3]的简称，FPGA是现场可编程门阵列（Field Programable Gate Array）的简称，两者的功能基本相同，只是实现原理略有不同，所以我们有时可以忽略这两者的区别，统称为可编程逻辑器件或PLD/FPGA。数字集成电路本身在不断地进行更新换代。它由早期的电子管、晶体管、小中规模集成电路、发展到超大规模集成电路(VLSIC，几万门以上)以及许多具有特定功能的专用集成电路。但是，随着微电子技术的发展，设计与制造集成电路的任务已不完全由半导体厂商来独立承担。系统设计师们更愿意自己设计专用集成电路(ASIC)芯片，而且希望ASIC的设计周期尽可能短，最好是在实验室里就能设计出合适的 ASIC 芯片，并且立即投入实际应用之中，因而出现了现场可编程逻辑器件(FPLD)，其中应用最广泛的当属现场可编程门阵列(FPGA)和复杂可编程逻辑器件(CPLD)。

早期的可编程逻辑器件只有可编程只读存贮器(PROM)、紫外线可按除只读存贮器(EPROM)和电可擦除只读存贮器(EEPROM)三种。由于结构的限制，它们只能完成简单的数字逻辑功能。

其后，出现了一类结构上稍复杂的可编程芯片，即可编程逻辑器件(PLD)，它能够完成各种数字逻辑功能。典型的PLD由一个“与”门和一个“或”门阵列组成，而任意一个组合逻辑都可以用“与一或”表达式来描述，所以， PLD 能以乘积和的形式完成大量的组合逻辑功能。

这一阶段的产品主要有PAL(可编程阵列逻辑)和GAL(通用阵列逻辑)。 PAL由一个可编程的“与”平面和一个固定的“或”平面构成，或门的输出可以通过触发器有选择地被置为寄存状态。 PAL 器件是现场可编程的，它的实现工艺有反熔丝技术、EPROM 技术和 EEPROM 技术。还有一类结构更为灵活的逻辑器件是可编程逻辑阵列(PLA)，它也由一个“与”平面和一个“或”平面构成，但是这两个平面的连接关系是可编程的。 PLA 器件既有现场可编程的，也有掩膜可编程的。 在PAL的基础上，又发展了一种通用阵列逻辑GAL (Generic Array Logic)，如GAL16V8,GAL22V10 等。它采用了 EEPROM 工艺，实现了电可按除、电可改写，其输出结构是可编程的逻辑宏单元，因而它的设计具有很强的灵活性，至今仍有许多人使用。 这些早期的PLD器件的一个共同特点是可以实现速度特性较好的逻辑功能，但其过于简单的结构也使它们只能实现规模较小的电路。

为了弥补这一缺陷，20世纪80年代中期。 Altera和Xilinx分别推出了类似于

PAL 结构的扩展型 CPLD(Complex Programmable Logic Dvice)和与标准门阵列类似的 FPGA(Field Programmable Gate Array)，它们都具有体系结构和逻辑单元灵活、集成度高以及适用范围宽等特点。 这两种器件兼容了 PLD 和通用门阵列的优点，可实现较大规模的电路，编程也很灵活。与门阵列等其它 ASIC(Application Specific IC)相比，它们又具有设计开发周期短、设计制造成本低、开发工具先进、标准产品无需测试、质量稳定以及可实时在线检验等优点，因此被广泛应用于产品的原型设计和产品生产(一般在 10,000 件以下)之中。几乎所有应用门阵列、PLD 和中小规模通用数字集成电路的场合均可应用 FPGA 和 CPLD 器件。

由于 PLD 软件已经发展的相当完善，用户甚至可以不用详细了解 PLD 的内部结构，也可以用自己熟悉的方法：如原理图输入或 HDL 语言来完成相当优秀的 PLD 设计。所以对初学者，首先应了解 PLD 开发软件和开发流程。了解 PLD 的内部结构，将有助于提高我们设计的效率和可靠性。如果您打算使用 VHDL 或 Verilog HDL 硬件描述语言来开发 PLD/FPGA,通常还需要使用一些专业的 HDL 开发软件,这是因为 FPGA 厂商提供的软件的 HDL 综合能力一般都不是很强,需要其他软件来配合使用， 对于 PLD 产品，一般分为：基于乘积项（Product-Term）技术，EEPROM（或 Flash）工艺的中小规模 PLD，以及基于查找表（Look-Up table）技术，SRAM 工艺的大规模 PLD/FPGA。EEPROM 工艺的 PLD 密度小，多用于 5,000 门以下的小规模设计，适合做复杂的组合逻辑，如译码。SRAM 工艺的 PLD(FPGA)，密度高，触发器多，多用于 10,000 门以上的大规模设计，适合做复杂的时序逻辑，如数字信号处理和各种算法。

3-2 使用 FPGA 产品开发的流程

采用 FPGA 现场集成技术开发数字系统，其主要设计流程如图 3—1 所示[4][10]。

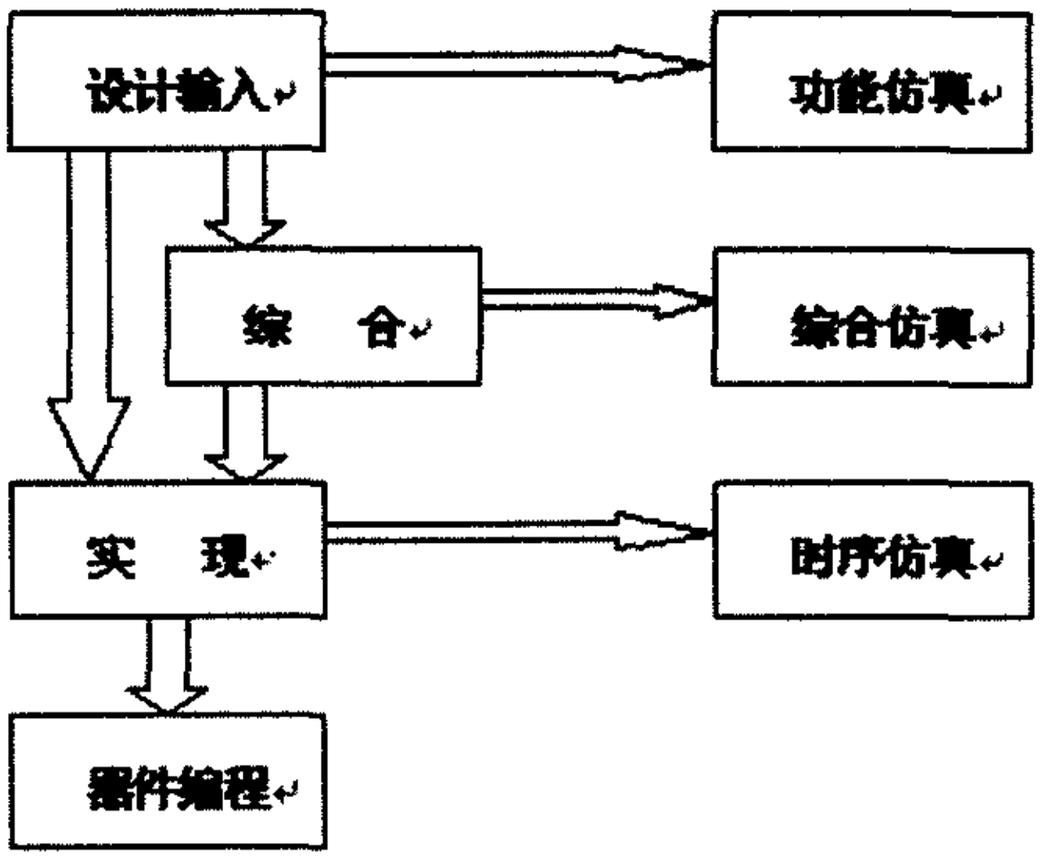


图 3—1 FPGA 设计流程

首先是设计输入，分别支持采用硬件描述语言的输入、采用原理图的输入以及状态机图标输入等三种输入方式（视不同的 EDA 软件而有所不同），不仅便于具有不同的设计习惯的人选择适合自己的设计输入工具。也可以采用混合输入的方式来进行设计。

对于采用硬件描述语言输入的设计，一般 EDA 工具支持 VHDL、VERILOG HDL 和 ABEL 语言输入方式，但是 ABEL 语言输入一般不用于 FPGA 的设计。对于 HDL 输入方式，一般的 EDA 软件都有良好的语法检测功能（这一点对于 VHDL 和 VERILOG HDL 的使用者非常重要）。

对于原理图输入的设计，一般 EDA 工具可以方便地调用器件库中的基本元件进行原理图输入，这就方便那些不懂或者不熟悉 VHDL 和 VERILOG HDL 等的开发者或者是更喜欢使用原理图输入的开发者进行设计。毕竟在 VHDL 和 VERILOG HDL 还没有流行起来之前，原理图输入设计方式是主要的输入设计方式。

对于状态机图标输入方式，其可以简单地把设计者的思路用状态图的方式加上状态之间的转换条件描述出来，使用这种方式的设计也可以被 EDA 工具转换为 VHDL 和 VERILOG HDL 语言。

其次是综合过程，这一步只是当采用硬件描述语言的输入方法时才需要，所谓综合就是把用硬件描述语言 VHDL 或 VERILOG HDL 语言编写的文件编译成可以具体实现的电路和布线形式，形成网表。现在市面上常用的 EDA 综合工具有 SYNOPSYS 公司的 FPGA EXPRESS、FPGA COMPLIER, SYNPLICITY 公司的 SYNPLIFY, XILINX 公司的 XST 等。它们的功能都是一样的，并且几乎支持市面上所有的可编程逻辑器件。可以看出，所谓综合，其实就是一种计算机智能化，通过计算机，自动地把硬件描速语言转换成电路的形式。

由于硬件描述语言功能强大，几乎所有用高级语言可以编写的程序都可以用硬件描述语言来重写，所以给硬件工程师带来了很大的方便，很多复杂的设计过程，原来只能通过数字电路设计的传统方法来进行，现在可以通过硬件描述语言描述功能，然后通过综合工具来变成实际的电路。但是，计算机的智能化程度仍然是有限的，在目前的编译技术下，想通过综合工具，把所有的硬件描述语言写出来的功能都变成电路可实现的形式几乎是不可能的。所以用硬件描述语言来设计电路，并不是“为所欲为”，想怎么写就怎么写。可能写出来的东西语法编译是没有错误的，但是综合工具却没有办法将其变成电路形式。因此，用硬件描述语言（以 VHDL 语言为例）来设计电路，需要注意以下几点：

- ✧ 设计 VHDL 语言程序时，应尽量考虑到电路实现的具体情况，尽量使用已经知道可以进行成功综合的 VHDL 语句或者程序段，而不要使用比较复杂的、功能描述性较强的语句。
- ✧ 要考虑到具体使用的芯片，结合具体芯片的情况编写程序。比如，有的 FPGA

芯片上面自带比较大的 RAM 块，要正确地使用它们，在编程时就要采取相应的风格，这样，综合工具才能根据程序语句利用这些 RAM 块。

- ✧ 选用不同的综合工具，综合出来的结果不同，效率也不同。有的语句在这个综合工具下不能综合，却可以在另外的综合工具下使用。所以在编程时，需要紧密结合综合器的特性。
- ✧ 需要结合数字电路设计的传统方法和设计者的以往的设计经验来编程，这样有利于编写出利于综合、效率高的 VHDL 语言。

综合一般还有另外一个重要功能，就是设计约束，可以控制设计的实现。设计约束有两种不同的类型：定位约束和时序约束。

定位约束控制布局规则、以及在目标器件中确定逻辑元件的位置。最通用的定位约束是引脚约束。为了确保各个修订版的引脚位置的一致性，用它们可以锁定特殊 I/O 位置的设计引脚。

时序约束告知软件哪些路径是临界的，因此需要紧密的布线和快速的布线。相反，时序约束也告知软件哪些路径不是临界的，因此不需要紧密的布局和快速的布线。时序约束可以驱动布局器和布线器。

可以使用以下工具在设计过程中输入约束：

- ✧ 原理图编辑器
- ✧ HDL 编辑器
- ✧ UCF 文件

可以用一下方法手工或者使用 Constraints 编辑器输入约束：

- ✧ 综合工具
- ✧ Constraints 文件
- ✧ Floorplanner

综合完成后，就是实现过程，实现过程就是根据综合出来的网表文件，结合相应的可编程逻辑器件，生成可以最后对器件进行硬件编程的 bit 文件。实现工具的流程对于 FPGA 和 CPLD 不同的器件是不同的。对于 FPGA 而言，可以分为以下几个阶段：编译规划、布局布线、程序 bit 流文件的生成。对于 CPLD 而言，可以分为：编译、配置、bit 流文件产生几个阶段。

设计实现进程的第一步是编译，合并所有的输入网表。下一步是规划映射。进行门级最优化的分析以及删除在合并的网表中没有用的逻辑。这一步也是逻辑资源的规划映射，把设计中的逻辑布局和目标器件硅片上的资源进行映射性规划，并检查物理设计规则。

规划映射完后，就是布局和布线。在布局步骤中，主要在于把整个的逻辑块，包括逻辑配置块（CLB）和输入/输出块（IOB）分配到电路片上的特定位置上去。如果在部分的逻辑元件中放置了时序约束，那么布局器就会通过移动相应的逻辑块去使它

们靠近，以满足它们的约束。

在布线阶段中，把逻辑网络分配到电路片中与逻辑元胞互联的物理线段上去。如果在部分的逻辑元件放置了时序约束，那么布线器就会尝试选择快速的连线以满足它们的约束。

设计实现后产生的一系列报告文件用于设计者了解设计实现过程的有关信息。CPLD 配置器实现 XC9500、XC9500XV 等器件的设计后，也会输出一些类似的报告文件，如：配置报告（设计文件名为.rpt），静态时序延时报告（设计文件名为.tim），向导文件（设计文件名为.gyd），程序文件（设计文件名为.jed）和时序时延仿真数据库（设计文件名为.nga）等。

要成功地进行对 FPGA 应用系统的设计，至少需要二次仿真。在图 3－1 中列出了三种仿真。第一中是逻辑功能仿真，这种仿真是对 VHDL 语言进行语法检查，然后进行逻辑仿真。由于这一级仿真是在设计输入之后、综合之前进行的，由于没有进行综合，仅仅是对 VHDL 语言功能上的仿真，所以速度比较快，而且只要语法上正确，仿真就可以顺利进行。功能仿真对于设计比较大的数字系统时显得比较重要。首先保证功能上的正确，这样才有价值进行下面的综合和电路实现。

第二级仿真是综合仿真，即在综合过后，对经过合并处理的网表文件进行逻辑功能的验证，这时不考虑布线及不同分区规划而产生的时延系统功能影响，仅仅验证综合过后电路系统设计的功能。

最后一级仿真是时序仿真，这一级仿真是在综合和实现过程完成后进行的，能够得到包括目标器件的详细的时序信息。可以通过创建使用图标表示工作台的测试台文件。这一级仿真的结果，可以看作是最终下载到芯片中运行后的结果。是接近与真实运行结果的仿真。如果这一级仿真结果正确，则设计是正确的，就可以下载到目标芯片上运行了。

在所有的设计完成后，就需要器件编程。器件编程指的是用编程下载工具将已有的设计转换成能够下载到器件上的形式，来对器件进行配置，从而将设计编程实际的器件实现的形式。用来对器件编程的工具主要有三种：JTAG 编辑器、PROM 文件格式器和硬件调试器。

在现场可编程集成电路的应用设计中，针对具体目标器件，需要不同的编程方式来实现目标数字系统的下载。根据不同的器件结构，目前常用的下载可以分为 3 种：

- ✧ 在系统可编程技术 ISP。就是器件在下载时没有专门的编程器，可直接在已制成的的系统板或 PCB 板上下载，且调电后程序保存。CPLD 芯片就是采用的这种技术。
- ✧ 在系统可重配技术 ISR，同 ISP 不同之处在与调电后程序不保存。一般的 ISR 器件采用的是 SRAM 编程技术，如 FPGA 就是采用了这种编程技术。
- ✧ 一次性编程技术。具备这种编程技术的 FPGA 采用反熔丝制造工艺，一旦编程就不可以改变，适用于高可靠性低功耗的场合使用。

3-3 FPGA 芯片的程序导入过程

在第二章中曾经讨论了 FPGA 芯片的程序导入并行配置问题，这里再讨论它的几种程序导入方法[2][6]。

具有 ISR 功能的 FPGA 器件采用了 SRAM 制造工艺，由 SRAM 存储配置数据，也称作 SRAM 现场可编程门阵列。这一特征使得相应 FPGA 器件在掉电后（或工作电压低于额定电压时）将丢失所存储的信息，采用这一类 FPGA 的数字系统在每次接通电源后，首先必须对改器件的 SRAM 加载数据，即重新装入器件功能配置数据。FPGA 芯片所具有的逻辑功能将随着置入配置数据的不同而不同。配置器件的过程也是在用户的目标系统或 PCB 板上进行的，故成为系统可重构技术。

在器件结构设计时，SRAM 现场可编程阵列(FPGA)就设置了 3 各模式控制信号 M0、M1、M2，由它们的状态可决定重构模式。表 2—2 是 VirtexII 器件的重构配置模式，下表是 SpartanII 器件的配置模式：

配置模式	M2	M1	M0
主串	X	0	0
从串	X	1	1
从动并行	X	1	0
边界扫描	X	0	1

表 3—1 SpartanII 器件的配置模式

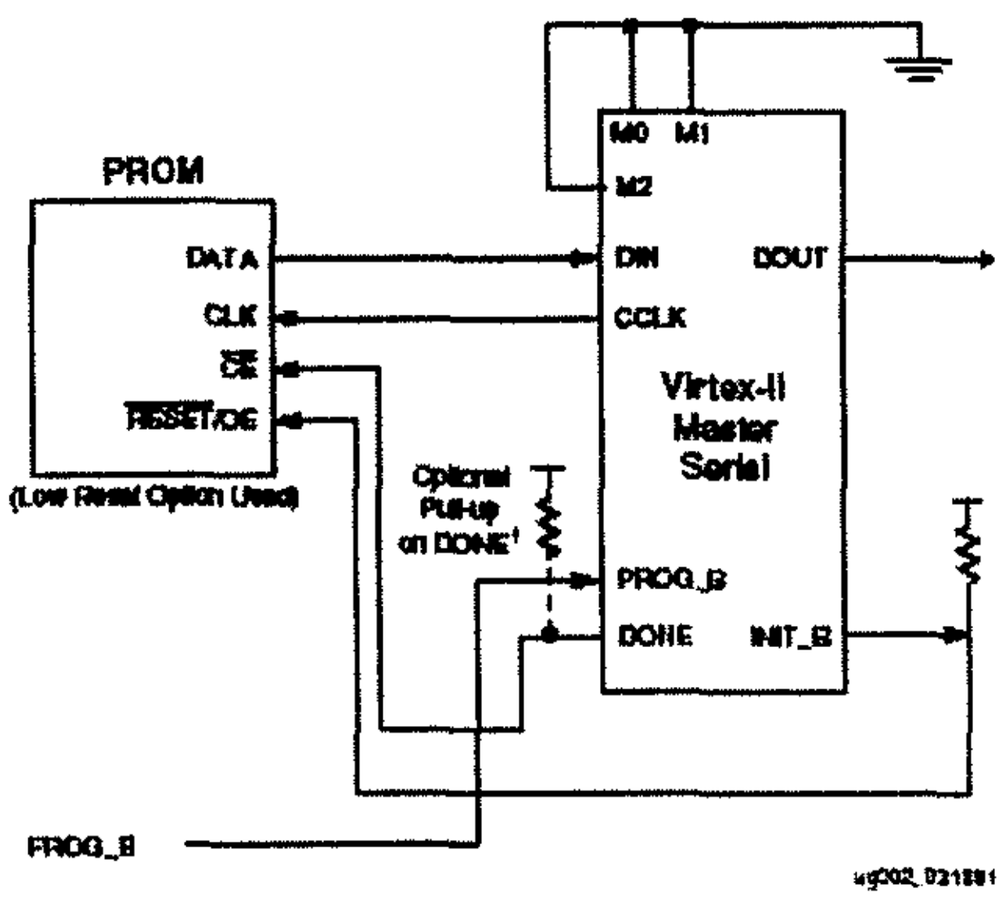


图 3—2 FPGA 芯片的主串配置模式

主动和从动的串行模式：主动串行模式，FPGA 驱动时钟输出控制配置过程。而从动串行模式，FPGA 被动接收 CCLK 作为输入，来控制配置过程。在两个模式中，FPGA 在每个时钟周期下载一位数据，配置数据的最高位首先被写到 DIN 脚。在主动串行模式下，FPGA 的时钟 CCLK 输出驱动一个串行 PROM，并将配置数据反馈给 DIN 脚。主动串行模式下，当 FPGA 器件被加电时，前 60BYTE 的数据以 2.5MHZ 的速率下载到 FPGA 芯片，直到配置速率设定数据并被载入 FPGA，配置速率才根据用户的设定而改变。缺省的配置速率为 4MHZ，最大为 60MHZ。

从动并行模式。于第二章介绍的 VirtexII 器件的 SelectMAP 模式基本一置，在这里不作详细介绍。

边界扫描（JTAG）模式：在采用 JTAG 模式来对 FPGA 器件配置或者回读时不需要使用非专用脚，仅仅需要通过器件固有的基于 IEEE1149.1 的测试端，TAP 即可进行。通过 TAP 进行数据配置时，需要采用专门的 CFG、IN 指令，这个指令可使到达 TDI 的输入数据转换成内部配置总线的数据包。其配置步骤主要如下：

- ✧ 载入 CFG-IN 指令进入边界扫描指令寄存器（IR），并进入移位数据寄存器（SDR）；
- ✧ 将标准配置数据串移到 TDI 端，并回到测试运行闲置（RTI）状态。
- ✧ 载入 TSTART 指令进入 IR，并进入 SDR 状态；
- ✧ 启动时钟序列（该序列长度是可编程的）TCK 后再回到测试运行闲置状态（RIT）。

采用边界扫描模式比其它模式简单。再回读时，通过配置数据可获得 FPGA 中的所有触发器。LUT RAM 和 BLOCK RAM 的数据。

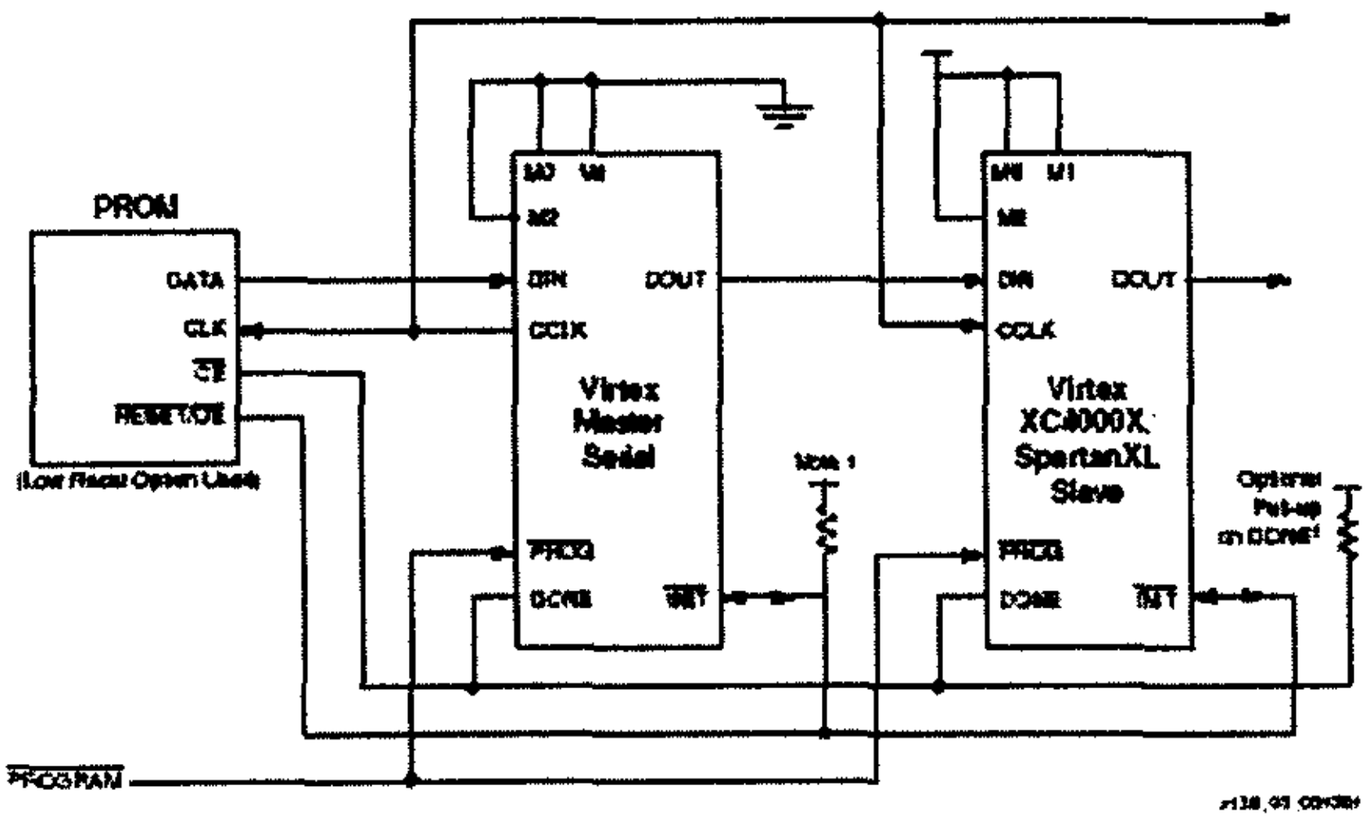


图 3—3 FPGA 芯片的主串/从串配置模式

3-4 XILINX SpartanII 系列现场可编程逻辑器件 FPGA 介绍

SpartanII 系列 FPGA[8]是 XILINX 公司生产的第二代 FPGA 产品。该系列 FPGA 密度高达 5，293 个逻辑单元，即 20×10000 个系统门，采用基于 Virtex 结构的流水线新结构，片内还有嵌入式 RAM，采用先进的 0.22/0.18um 半导体工艺，6 层板结构。具有无限的可重编程性以及很低的价格。

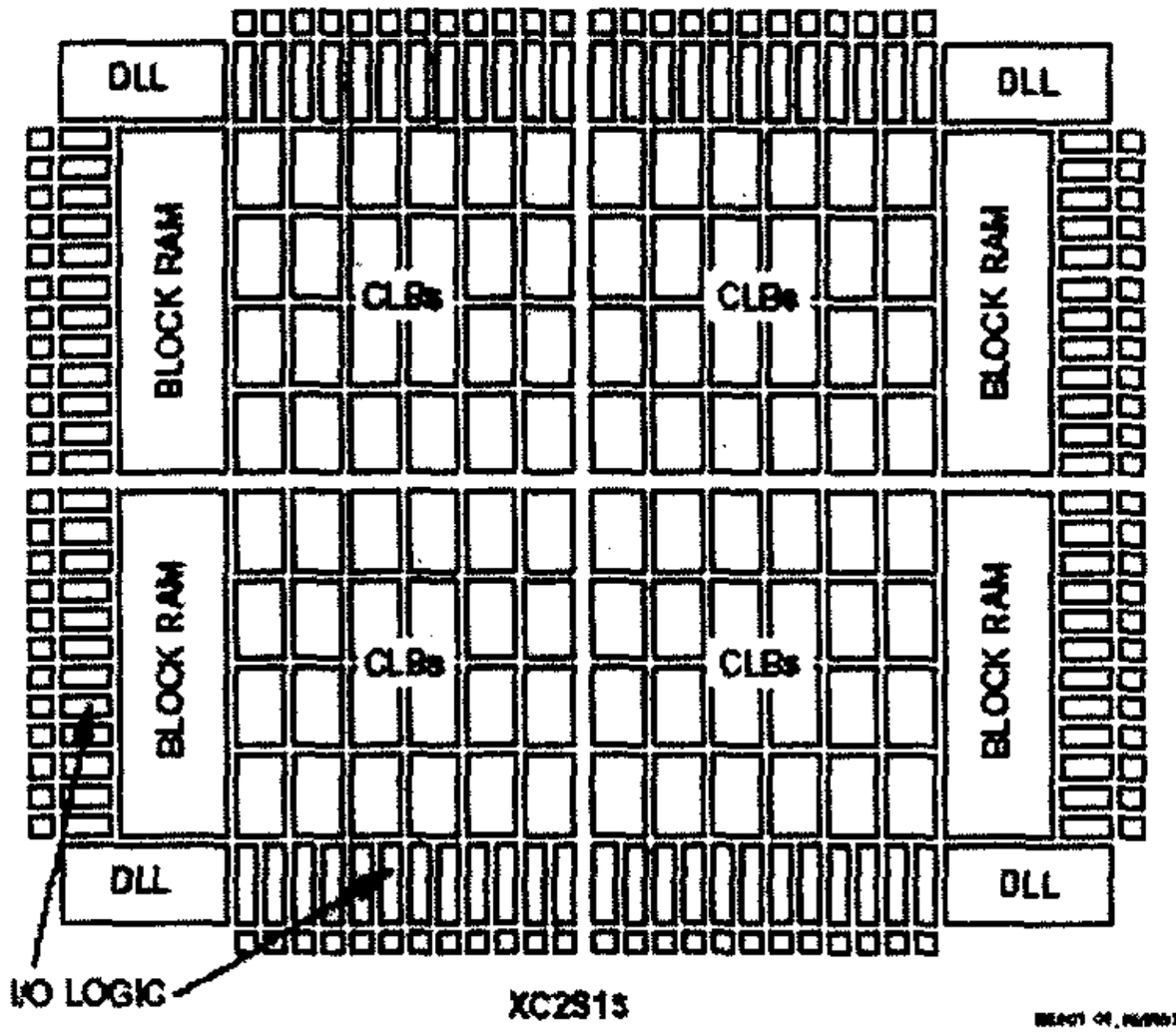


图 3—4 SpartanII 系列 FPGA 的基本结构

SpartanII 系列 FPGA 具有系统级特性：该系列 FPGA 芯片采用低压布线结构。片内还有丰富的寄存器资源、时钟使能信号、同步、异步置为/复位信号。为增强时钟控制，提供了 4 个精确的延迟锁相环（DLLS），含有 4 个主要的全局低偏移时钟分配网络，以及 24 个次全局网络；有两种类型的片上随即存储内存（SelectRAM）：块状 RAM、分布式 RAM。为满足高速运算设计的进位逻辑提供精确的乘法器。

SpartanII 系列 FPGA 的基本结构如图 3—4 所示。其主要包括 5 个可配置部分：

a) 可配置逻辑块（CLBS），结构如图 3—5 所示，用于实现大部分逻辑功能。构成 CLB 的基本结构是逻辑元胞（LC）。一个 LC 包括一个 4 输入的函数发生器，进位逻辑和一个存储部分。在每个 LC 中，函数发生器的输出既是 CLB 的输出又是 D 触发器的输入。每个 SpartanII 系列 FPGA 的 CLB 包含 4 个 LC。每个 CLB 由相似的两个单元构成。除了 4 个基本的 LC 外，SpartanII 系列 FPGA CLB 还包含可以提供 5 输入和 6 输入的函数发生器。在估计所给器件的系统门数时，应以每个 CLB 包含 4.5 个 LC 来计算。

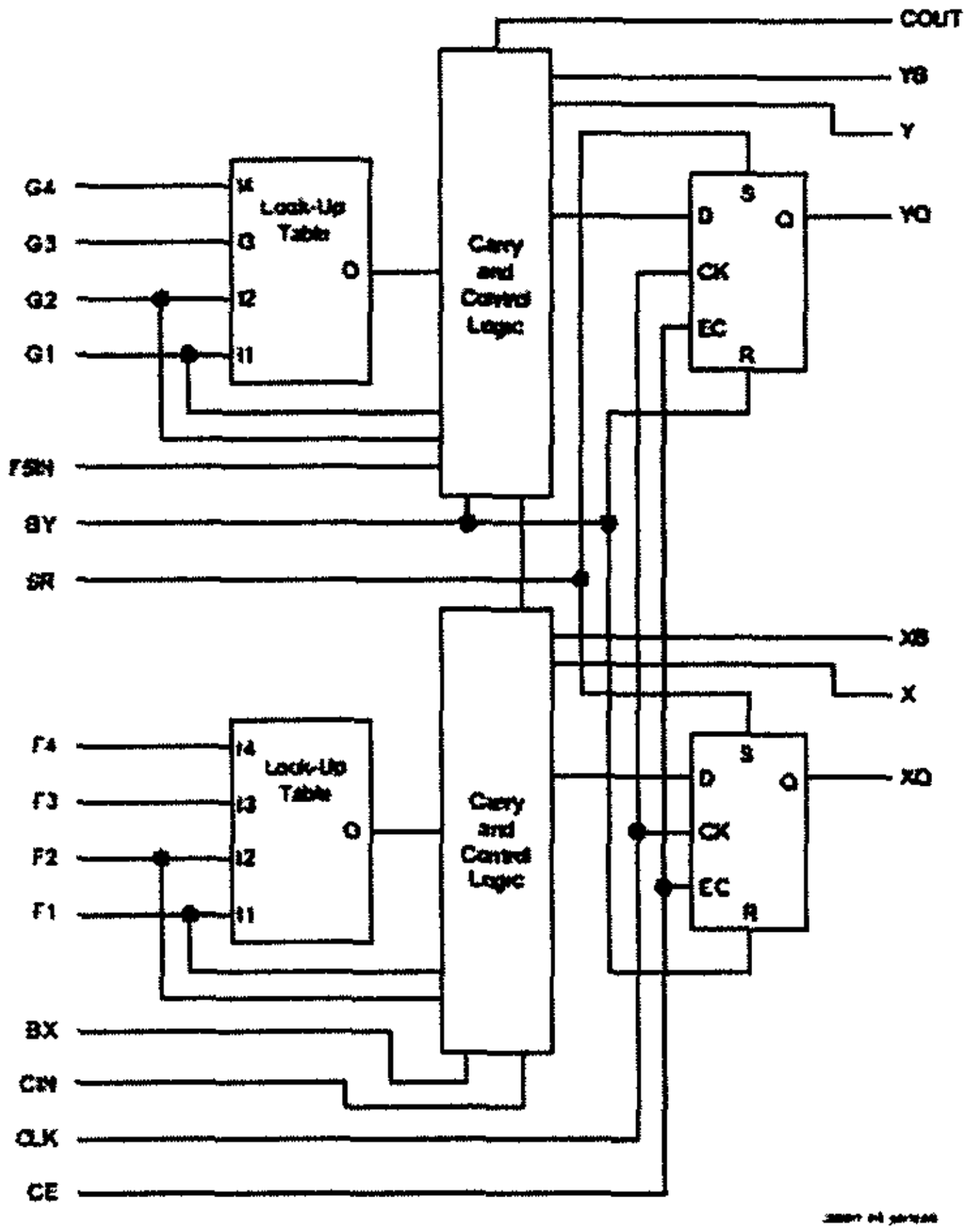


图 3—5 SpartanII 系列 FPGA CLB 一个单元的原理框图

SpartanII 系列 FPGA 的函数发生器用 4 输入的查找表(LUT)实现。除了作为函数发生器，每个 LUT 可以作为一个 16×1bit 的同步 RAM。进而，在同样单元中的两个 LUT 可以组合产生一个 16×2bit 或者是 32×1bit 的同步 RAM，或者是一个 16×1 双端同步 RAM。

SpartanII 系列 FPGA 中的时序逻辑部分可配置成边沿触发的 D 型触发器或是电平触发器或是电平触发的锁存器。D 型触发器的输入既可以由同一单元中的函数发生器驱动，也可以由该片的输入直接驱动。除了时钟信号和时钟使能信号，每一单元都有同步的置位和复位信号(SR 和 BY)。所有这些控制信号都可以独立反向，并由同一单元中的两个触发器共享。

每个 SpartanII 系列 FPGA 的 CLB 包含两个三态缓冲器（BUFT)。每个 SpartanII 系列 FPGA 的 BUFT 都有一个独立的三态控制引脚和一个独立输入引脚。

b) 可编程输入/输出块（IOB)，结构如图 3—6 所示。该系列 FPGA 支持多种 I/O 标准。一个 IOB 寄存器可以实现两种功能：边沿触发的 D 型出发器；电平触发的锁存器。在每个 IOB 内，三个寄存器共享一个时钟信号(CLK)，但是每个寄存器都有独立

的时钟使能信号。此外，三个寄存器还共享置位/复位信号（SR）。每个寄存器可独立配置 SR 信号，可将其配置成一个同步置位、同步复位、异步置位或者是异步清零信

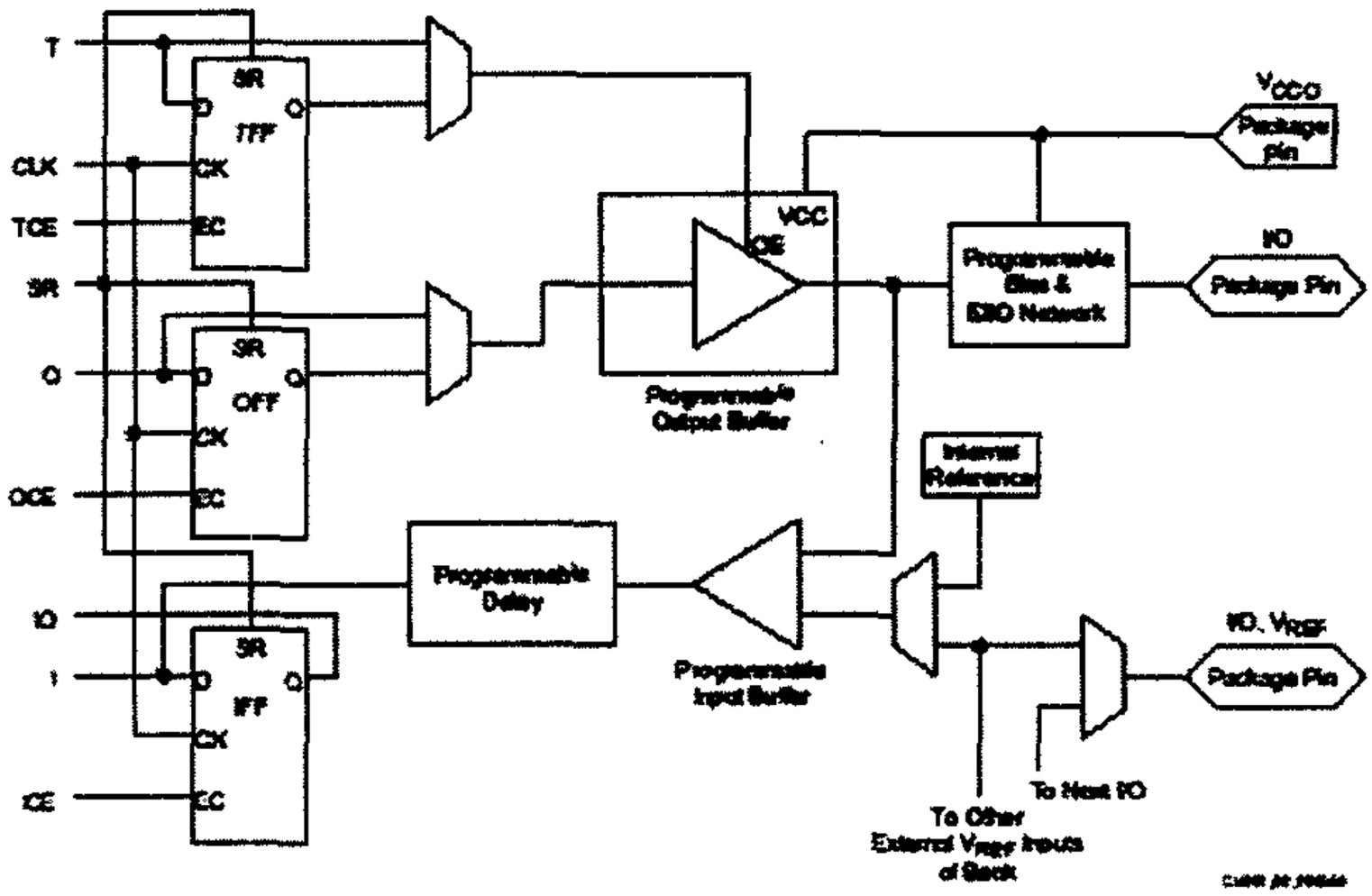


图 3—6 SpartanII 系列 FPGA 的 IOB 结构图

号。在 IOB 的输入途径上有一个缓冲器，它控制输入信号是直接输入到内部逻辑还是通过一个可选的输入触发器输入。每个输入缓冲器都可以配置成该系列支持的任何一种电压标准。每个输入信号都有可供选择的下拉、上拉电阻。他们在芯片配置之后使用。在 SpartanII 系列 FPGA 的 IOB 输出路径上有一个三态输出缓冲器，输出信号可以配置成直接从内部逻辑输出或者是通过一个可选择的 IOB 输出触发器输出。每个输出驱动器可独立配置成该系列支持的任何一种电压标准。输出高电压取决于外部提供的电压 Vcco。

Vcco	可兼容的标准
3. 3V	PCI, LVTTL, SSTL3 I, SSTL3 II, CIT, AGP, GTL, GTL+
2.5V	SSTL2 I, SSTL2 II, LVCMOS2, GTL, GTL+
1.5V	HSTL I, HSTL III, HSTL IV, GTL, GTL+

表 3—2 IOB 可兼容的电压标准

上面提到的一些 I/O 标准需要电压 Vcco 与/或 Vref。这些外部电压与器件引脚相连，这些器件引脚是一组 IOB，成为组。在一个组中，哪些 I/O 标准可以相连是有严格规定的。把 FPGA 每边分成两个组，这样整个 FPGA 芯片共有 8 个 I/O 组，见图 3—7。每个组包含多个 Vcco 引脚，这些引脚必须与相同的电压相连。该电压大小由使用的输出标准确定。在一个组内部，如果输出标准都使用相同的 Vcco，则它们可以

混合。表 3—2 列出了可兼容的电压标准。

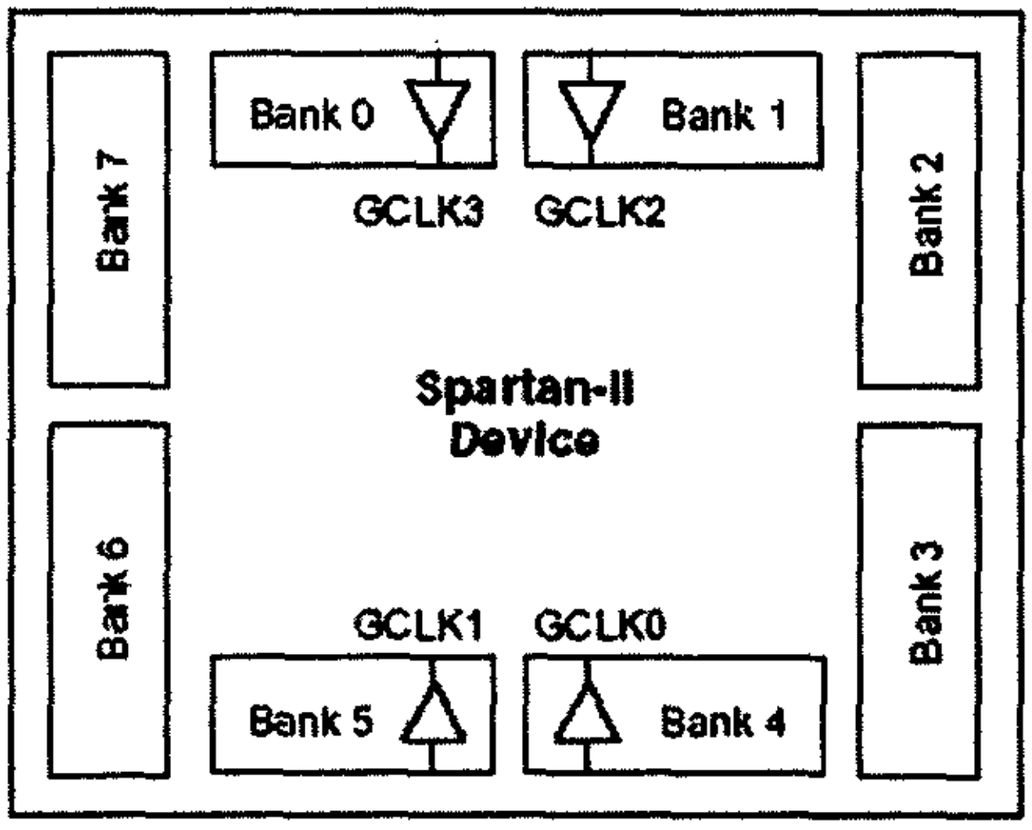


图 3—7 SpartanII FPGA 的 I/O 组

c) 布线通道。SpartanII 系列 FPGA 的布线通道主要包括可编程的布线矩阵、局域布线、精细布线、全局布线以及时钟布线网络和 I/O 布线等丰富的布线资源。可编程的布线矩阵是一条最长的延迟线，它给出了设计最坏情况的速度门限。SpartanII 系列 FPGA 的布线结构和它的布局、布线软件是按单项优化过程设计的，这种优化设计，减少了长距离信号的延迟，增强了系统的性能。

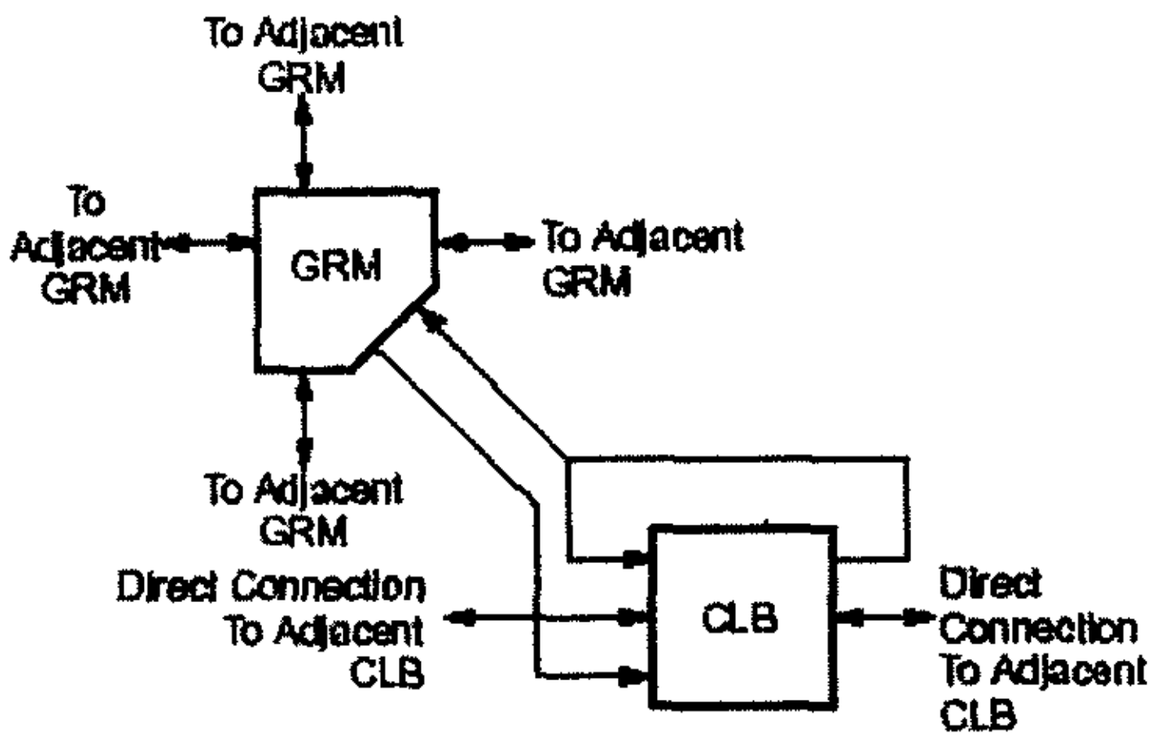


图 3—8 SpartanII 系列 FPGA 的局部布线

局域布线的框图如图 3—8。其中给出了 3 种连接方式：a LUTs、触发器和 GRM（通用布线矩阵）之间的连接；b 内部的 CLB 回读路径，提供了在同一个 CLB 内与 LUTs 的高速连接；c 直接路径，为水平相邻的 CLBs 之间提供了高速连接。

大多数 SpartanII 系列 FPGA 的信号布线用于通用目的线上。因此，大多数资源互联与这一布线层相关。通用布线资源包含：a 与每一个 CLB 相邻有一个通用布线矩阵（GRM）。 b 24 条单长线把 GRM 信号布于相邻的 GRM 的四周；c 96 条缓冲的十六进

制线把 GRM 信号布于相距 6 块 GRM 的任意 GRM 的四周。其中三分之一的十六进制线是双向的，其余的是单向的；d 12 条长线是带缓冲的双向线。它们可以快速有效地对信号布线。

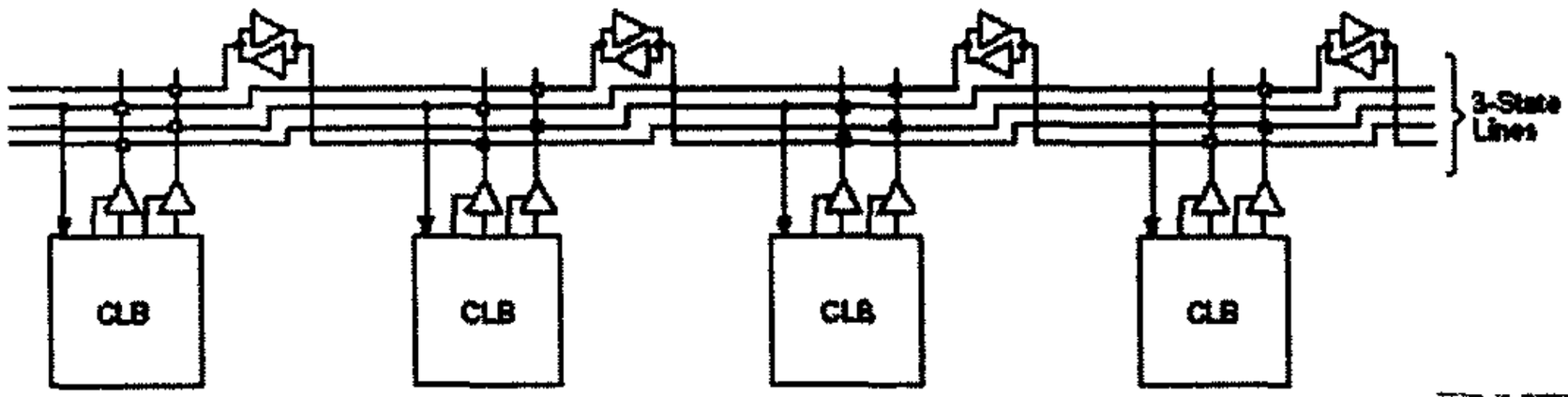


图 3—9 与精细水平总线连接的 BUFT

精细布线：一些信号需要精细的布线资源以增强其功能。在 SpartanII 系列 FPGA 的结构中，精细布线资源为以下两种信号提供布线：a 水平布线资源为片上三态总线提供布线，在 CLB 的每一行，有 4 条可分离的总线，因此在一行中有多条总线，如图 3—9。 b 每个 CLB 中有两个精细布线网格，它们将进位信号与相邻的 CLB 垂直相连。

全局布线资源和时钟分布网络：全局布线资源主要用于时钟信号和其它有大扇区的信号布线。SpartanII 系列 FPGA 有两级全局布线资源：主全局布线资源和次全局布线资源。主全局布线资源是 4 个精细的全局网络，带有精细的输入引脚。他们为时钟信号布线。每个全局时钟网络可以驱动所有的 CLB、IOB 和块状 RAM 的时钟信号引脚。主全局网络仅仅能由全局缓冲器驱动，共有四个全局缓冲器，每个全局缓冲器分配一个。其中两个位于芯片中心的顶部，两个位于芯片中心的底部。他们驱动 4 个主全局网络，进而驱动所有的时钟信号引脚。

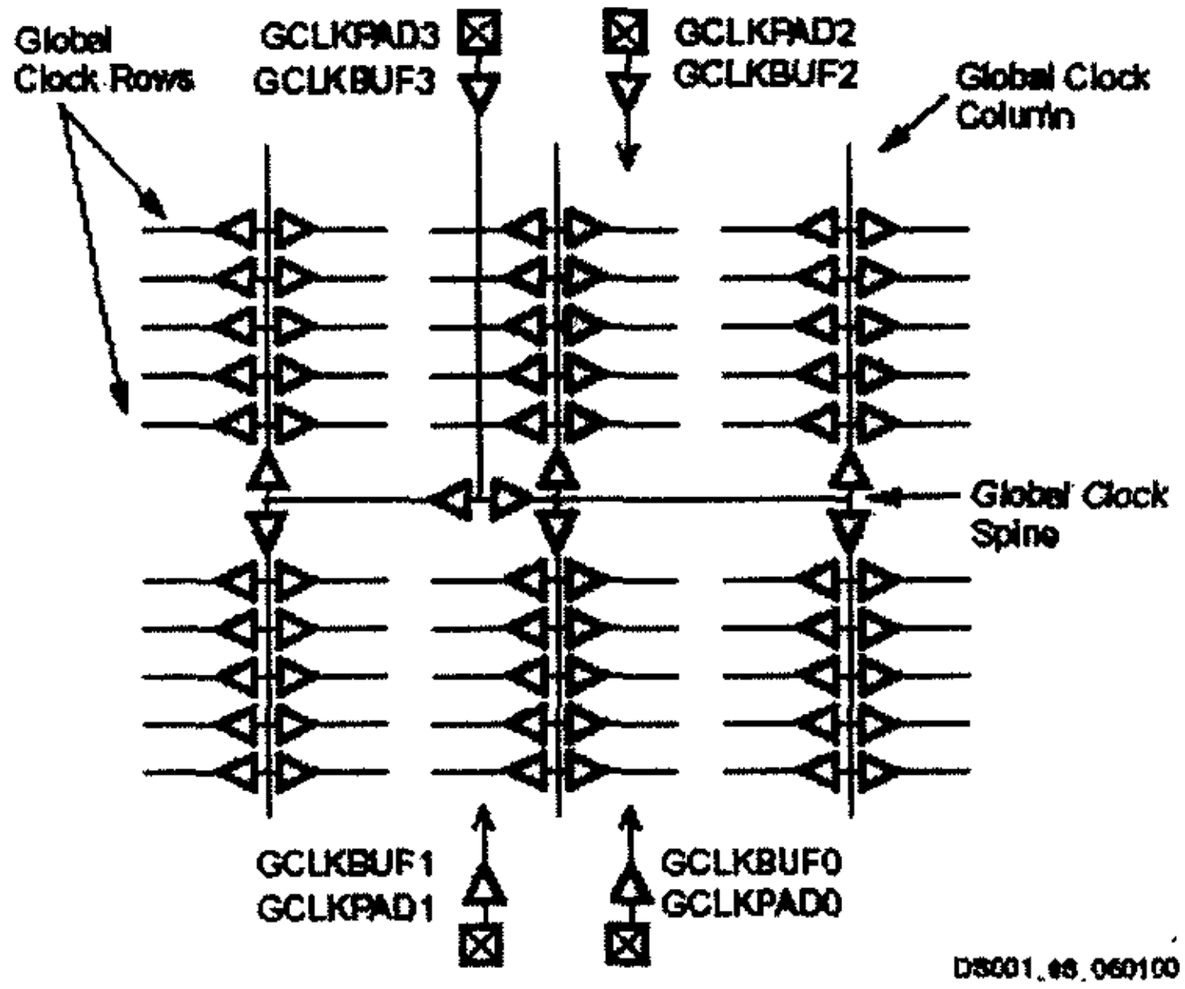


图 3—10 全局分布时钟网络

次全局布线资源包括 24 条骨干线，其中 12 条位于芯片的顶部，12 条位于芯片的底部。这些次全局布线资源比主全局资源更灵活，他们不仅能对时钟信号布线，还可以对其它信号布线。图 3—10 给出了一个典型的时钟分布网络。

d) 块状 RAM。

在 SpartanII 系列 FPGA 中有几个大的块状 RAM。这些 RAM 是按照列排列的。所有的 SpartanII 系列 FPGA 器件都有这样的两个列，它们沿着芯片的垂直边排列。每个块状 RAM 有 4 个 CLB 高，一个有 8 个 CLB 高的 SpartanII 系列 FPGA 器件每列有 2 个块状 RAM ，整个芯片共有 4 块 RAM。

如图 3—11 所示，每个块状 RAM 是一个完全同步的 4096bit 的双端 RAM，其中每一端都有独立的控制信号。可独立配置两个端口的数据宽度。

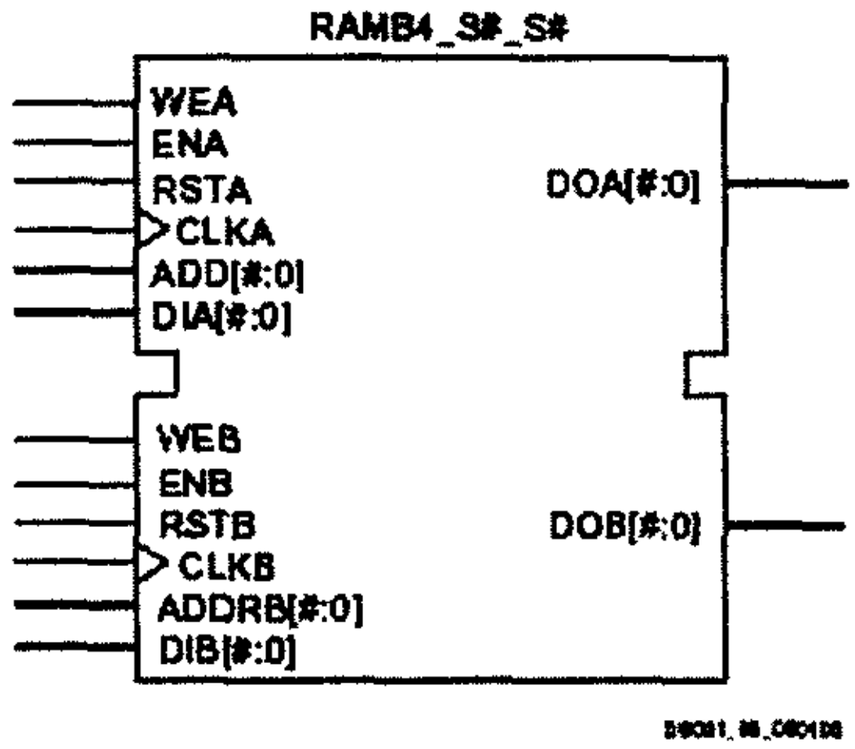


图 3—11 双端块状 RAM 结构框图

e) 延迟锁相环（DLL）

在 SpartanII 系列 FPGA 器件中，与每个全局时钟输入缓冲器相连有一个全数字式的延迟锁相环（DLL）。DLL 可以使时钟输入信号与整个芯片内部的时钟输入引脚之间的偏差减少到最小。每个 DLL 可以驱动两个全局时钟网络。DLL 控制输入时钟、分布的时钟并可以自动调整时钟的延迟。该闭环系统确保时钟边沿到达内部触发器与其到达输入引脚同步，有效地消除了时钟分配时地延迟。

除了消除时钟分布延迟，DLL 还提供先进的多时钟控制。DLL 可以使时钟信号倍频，或将时钟信号按 1.5, 2, 2.5, 3, 4, 5, 8, 16 分频输出。为了保证在芯片配置之后，FPGA 启动之前得到正确的时钟，DLL 可以在锁定正确的时钟之前，延迟芯片配置的实现过程。

3-5 XILINX VirtexII 系列现场可编程逻辑器件 FPGA 介绍

VirtexII 1.5V 系列 FGPA[7]是高性能、高密度的可编程逻辑器件。其采用了先进的 8 层金属 1.8um 的 CMOS 工艺，优化的新型结构，使硅片得到了有效的利用。高速、高密度的VirtexII 1.5V 系列FGPA专门为低压操作设计。其系统门级数从40K到10M；可以提供高达420M的内部性能；840M的IO速度；PCI适于3.3V、32/64位、33/64MHZ。该系列FPGA还有高度灵活的SelectIO技术，可以支持20种高性能的I/O接口标准并具有高达1108种单端I/O口。支持9种单端I/O标准和6种差分I/O标准，支持差分信号LVDS、BLVDS、LVPECL；差分的IO信号口可以作输入、输出或者是IO口；并兼容标准的差分设备；其中，LVPECL和LVDS的时钟输入是300MHZ。并且，该系列FPGA还拥有SelectRAM技术。芯片内包括1.9Mb的内部分布式RAM以及高达3.6Mb的内部全双端块状RAM；并提供400Mb/s DDR SRAM 外部RAM接口、400 Mb/s FCRAM 外部RAM接口、333 Mb/s QDR™ -SRAM 外部RAM接口、600 Mb/s Sigma RAM 外部RAM接口。该系列FPGA还具有高性能的ActiveInterconnect 技术，使得内部布局和连线更加有效和容易控制；片内有12个高级时钟管理模块（DCM），16个全局时钟多路选择缓冲器。该系列FPGA还提供强大的算术处理硬核：18bit×18bit的乘法器和快速前向逻辑进位链。

VirtexII 1.5V 系列FGPA器件的基本结构图如图3－12所示：

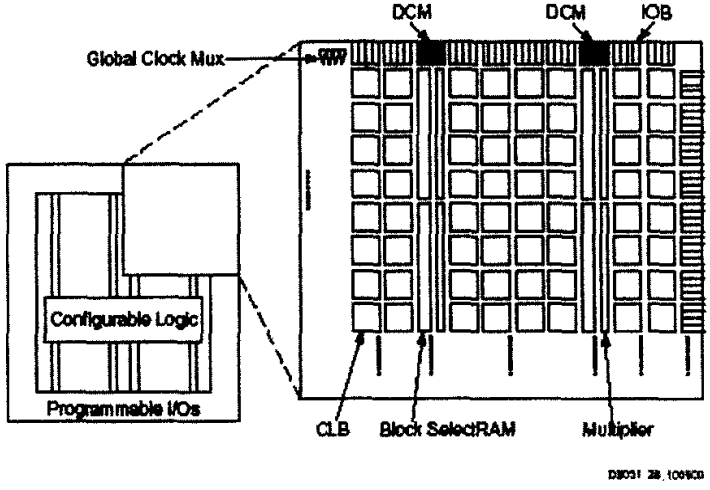


图3－12 VirtexII 1.5V 系列FGPA器件的基本结构图

主要包括

a) 可配置逻辑块（CLB）

VirtexII 1.5V 系列 FGPA 器件的 CLB 在器件内部成阵列分布，用于实现复杂的同步逻辑设计。每一个 CLB 单元都和一个转换矩阵（Switch Matrix）相连接，并和通用布线矩阵（GRM）相连，如图 3－13。一个 CLB 单元由 4 个相同的片段（Slice）和

一些快速的本地反馈电路组成。四个片段被分为 2 列和 2 行排列，它们有两个独立的进位逻辑链和一个公用的移位链。

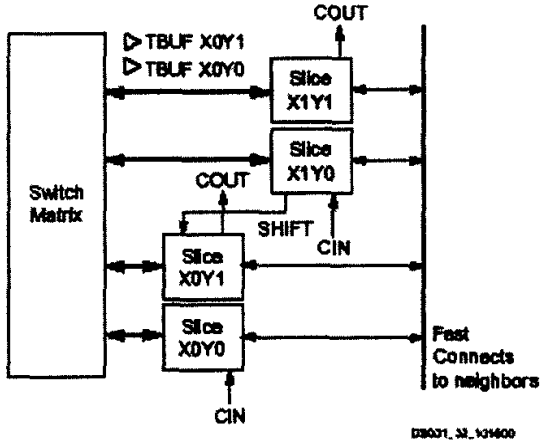


图 3－13 VirtexII 1.5V 系列 FGPA CLB 单元

每一个片段包含有两个 4 输入的函数发生器，进位逻辑，算术逻辑门电路，多功能数据选择器和两个存储单元，如图 3－14 所示。每一个 4 输入函数发生器可能是一个可编程的 4 输入查找表（LUT），16bit 的分布式 SelectRAM 或者是一个 16bit 移位寄存器。在一个片段中，函数发生器的输出同时驱动该片段的输出和存储单元的输入。图 3－15 是更加详细的片段的原理框图。

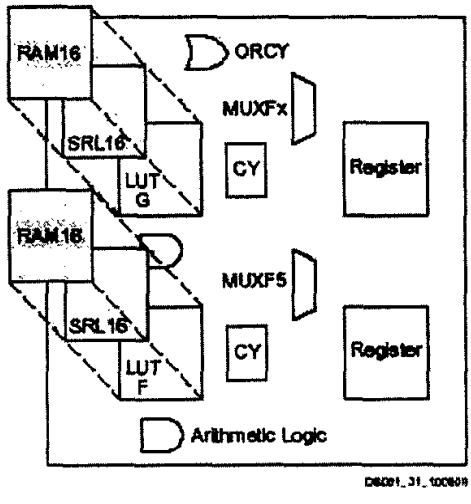


图 3－14 VirtexII CLB 中的片段（Slice）配置

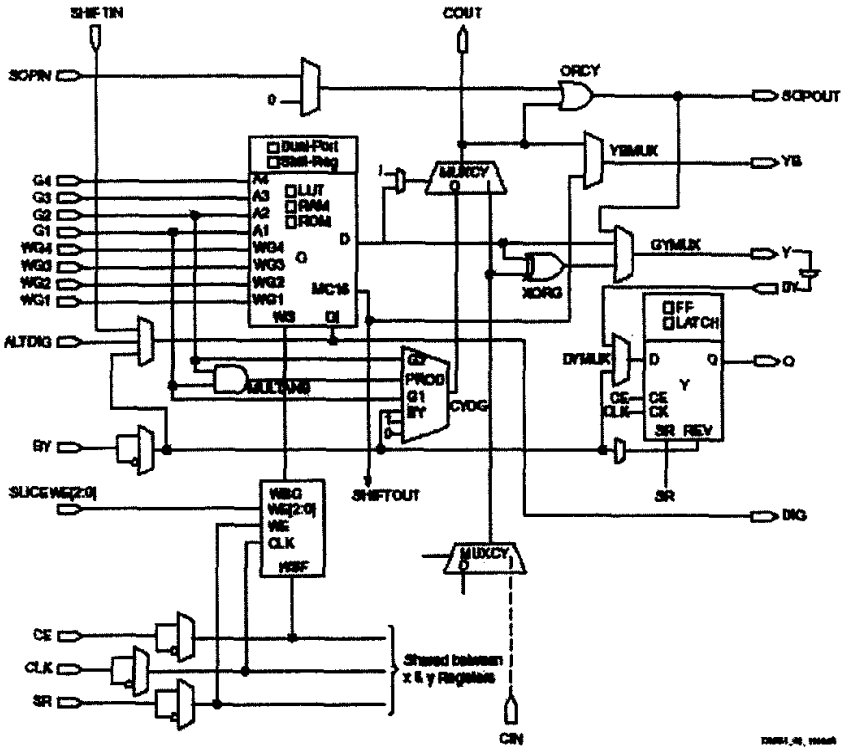


图 3—15 VirtexII CLB 中的片段（Slice）原理框图

CLB 中的函数发生器由查找表 LUT 实现，除了作为函数发生器，每个查找表 LUT 还可以作为一个 16×1bit 的同步 RAM。进而，在同一个 CLB 中的分布式 RAM 可以组成：单端口 16×8bit、32×4bit、64×2bit、128×1bit 和双端口 16×4bit、32×2bit、64×1bit RAM。

CLB 中的时序逻辑部分可以配置成一个边沿触发的 D 型触发器或是一个电平触发的锁存器。

b) VirtexII I/O 模块（IOB）是两个一组或四个一组。每一个 IOB 都可以单独作为输入或输出。两个 IOB 在一起使用可以作为一个差分对，它们通常与内部的同一个转换矩阵(SwitchMatrix)相连。

IOB 支持多种高性能 IO 标准，包括 LVDS, LDT, Bus LVDS, LVPECL 等差分信号。

c) 18Kbit BLOCK RAM

VirtexII 器件集成了大量的 18kbit block RAM。这些 RAM 弥补了 CLB 中的分布式 RAM 的不足。每一个 Block RAM 块都含有 18kbit 的纯双端口 RAM，每一个端口都支持以下输入方式：时钟和时钟使能、写使能、清零/置位、地址写、地址读取。

作为单端口 RAM 使用时，每个 Block RAM 块中的 18KbitRAM 可以配置成 2K×9bit、

1K×18bit、512×32bit。或者，也可以将 18Kbit 中的 16K 配置成 16K×1bit、8K×2bit、4K×4bit。这些丰富的配置模式为灵活地设置数据的宽度提供了极大的方便。

当作为纯双端口 RAM 使用时，分别由独立的控制信号控制两个端口，使之可以独立地工作。这两个端口都是时钟同步的。关于两个端口（A 口和 B 口）数据宽度的配置较单端口情况复杂一点，图 3－16 给出了双端口数据宽度配置的可能情况：

Port A	16K x 1	16K x 1	16K x 1	16K x 1	16K x 1	16K x 1
Port B	16K x 1	8K x 2	4K x 4	2K x 9	1K x 18	512 x 36
Port A	8K x 2	8K x 2	8K x 2	8K x 2	8K x 2	
Port B	8K x 2	4K x 4	2K x 9	1K x 18	512 x 36	
Port A	4K x 4	4K x 4	4K x 4	4K x 4		
Port B	4K x 4	2K x 9	1K x 18	512 x 36		
Port A	2K x 9	2K x 9	2K x 9			
Port B	2K x 9	1K x 18	512 x 36			
Port A	1K x 18	1K x 18				
Port B	1K x 18	512 x 36				
Port A	512 x 36					
Port B	512 x 36					

图 3－16 双端口 Block RAM 配置模式

需要注意，对于不同的情况，有可能从某个端口不能访问到所有的 18kbit 空间，而只能访问到 16kbit 空间。需要严格按照图 3－16 配置。

关于 Block RAM 的读写时序也是需要考虑的问题。对于单端口工作模式，比较简单，在 CLK、EN、WE 和 RST 等控制信号的作用下，当 CLK 上升沿到来时，数据输出口输出相应地址的数据，同正常的同步单端口 RAM 操作没有什么不同。

当工作在双端口模式下时，就有可能出现两个端口同时操作一个存储地址的情况发生，这样就存在着竞争，必须要有一定的规则来约束：

- ✧ 当两个端口的地址相同并且两个端口都在进行写操作时，存储单元的内容视为无效
- ✧ 当两个端口地址相同，一个端口在写，一个端口在读时，存储单元中的内容改变，但是输出口的数据是无效的。

Block RAM 在 VirtexII 器件中的分布是按列纵向分布的，一共有 4 列或者是 6 列，而每一列中各有多少 Block，就要根据具体的器件大小来决定了。

d) 18bit 硬件乘法器

VirtexII 器件有一个重要的特点就是内部的一个 18bit×18bit 的硬件符号数乘法器，由于在数字电路设计中，乘法器的设计是比较耗费资源的，所以硬件乘法器给设计者提供了很大的方便。这个硬件乘法器可以单独使用，也可以配合 18Kbit Block RAM 使用，提供高效、低功耗的乘法计算。图 3－17 是这个硬件乘法器的功能框图，为了和 VirtexII 器件中的 18Kbit Block RAM 配合使用，每一个硬件乘法器都和一个

Block RAM 块相连接，并且在二者配合使用时，Block RAM 只能配置成 18bit 宽度。

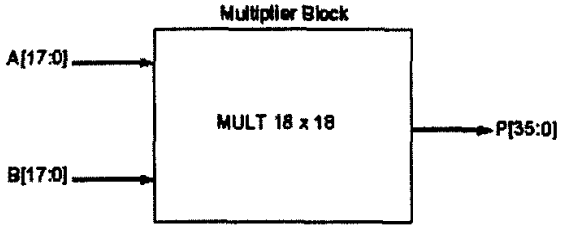


图 3－17 硬件乘法器功能框图

硬件乘法器的个数随着器件大小的不同而不同。

e) 全局时钟选择缓冲器(Global Clock MUX)

VirtexII FPGA 器件一共有 16 个时钟输入引脚，它们也可以被用来作为用户 IO 脚。在器件的顶部和底部分别各有 8 个时钟输入引脚。如图 3－18 所示。

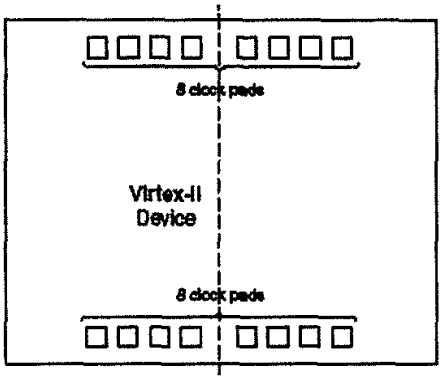


图 3－18 VirtexII 器件时钟输入引脚

全局时钟选择缓冲器负责将时钟输入连接到器件内部低延迟的树状分布 CLOCK 布线上。与时钟输入引脚一样，在器件的顶部和底部分别各有 8 个全局时钟选择缓冲器。

全局时钟选择缓冲器可以看作是两个部分：全局时钟缓冲器和选择器。作为全局时钟缓冲器，它把外部时钟同器件里部分或者所有的同步逻辑器件（比如 CLB、IOB、寄存器、SelectRAM Block）连接起来，其输入可以直接是外部的时钟输入，也可以是数字时钟管理器（DCM）的输出。而作为选择器，全局时钟选择缓冲器可以看作是两个时钟输入、一个控制信号选择的二选一器件，如图 3－19－a 所示。I0 和 I1 分别是两个时钟信号，它们可以是外部时钟直接输入，也可以是 DCM 的输出。S 是选择信号，当 S 为 0 时，输出端 O 为时钟 I0，当 S 为 1 时，输出端 O 为时钟 I1。

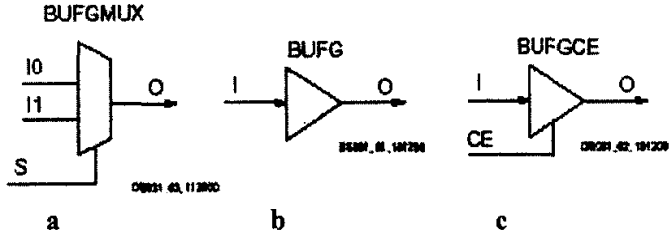


图 3－19 全局时钟选择缓冲器的三种工作方式

I1 和 I0 以及控制信号 S 可以是互相异步的，但是当 S 异步变化时，器件内部的同步功能一定要等到新的时钟的下降沿（或者上升沿）到来时才改变输出时钟，如图 3－20 所示。

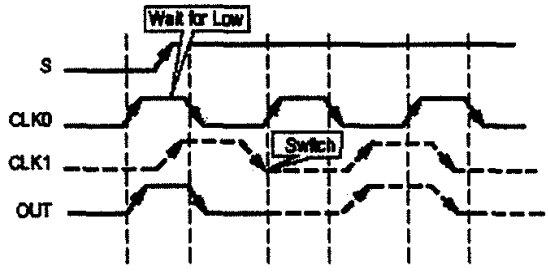


图 3－20 时钟选择波形图

根据需要，全局时钟选择缓冲器还可以配置成如图 3－19－b（简单的输入缓冲）和图 3－19－c（时钟三态缓冲）的形式。

VirtextII 器件一共有 16 个时钟输入引脚。如果以器件的中心为原点，把器件分为 4 个象限（如图 3－21 所示，为别计为 NE、NW、SW、SE），则每一个象限 中可以允许同时有 8 个不同的时钟。这 8 个时钟可以是 16 个时钟输入通过 8 个全局时钟选择缓冲器后的输出。为了降低功耗，当不需要用到那么多时钟输出时，没有用到的输出端被关闭。

f） 数字时钟管理器（DCM）

不同于前面介绍的 SpartanII 器件通过数字锁相环（DLL）来实现对时钟的管理，VirtexII 器件在时钟管理方面也有了比较大的改进：通过一种全新的数字时钟管理器（DCM）来实现时钟的管理，DCM 比原来的 DLL 功能更加强大，更能够满足高速复杂数字电路设计的需要。图 3－22 是 DCM 的功能框图。

DCM 可以实现以下的功能：

- 减小时钟偏差。DCM 根据输入时钟自动调整数字延迟线来控制延迟，通过在内部传输的时钟信号中自动加入延迟的方法使得到达内部逻辑单元的时钟信号

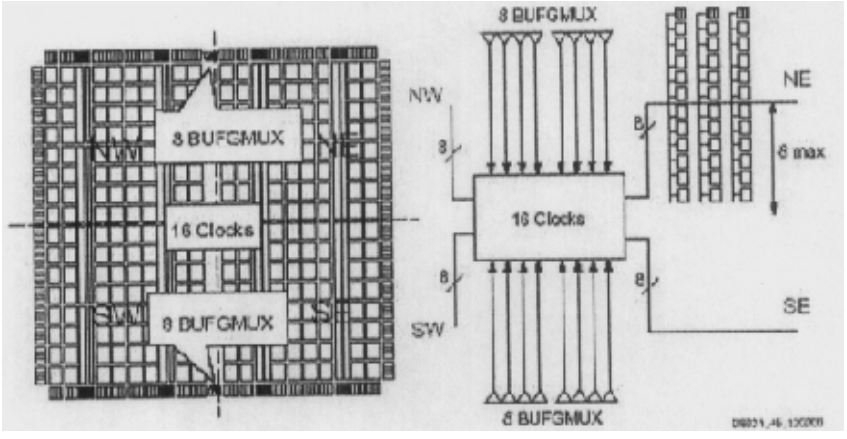


图 3－21 VirtexII 的时钟分布

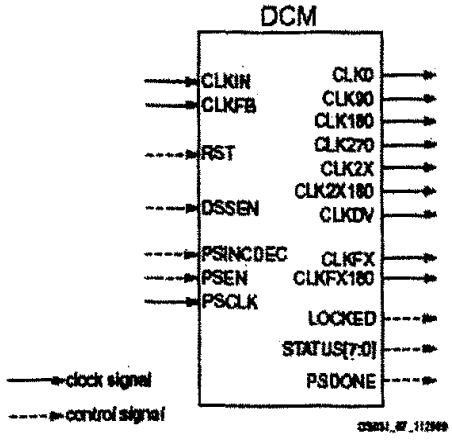


图 3－22 DCM 功能框图

与外部输入时钟信号保持严格的相位一置。

✧ 频率综合。DCM提供了强大的功能，可以产生新的、不同频率、不同AC特性的综合频率。CLK2X和CLK2X180可以提供2倍扩频信号，CLKDV输出可以提供1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6, 6.5, 7, 7.5, 8, 9, 10, 11, 12, 13, 14, 15和16分频的时钟。此外CLKFX和CLKFX180输出端可以按照下面的公式提供相应的扩频信号，M和D是两个整数，范围是1－4096：

$$FREQ_{CLKFX} = (M/D) * FREQ_{CLKIN}$$

✧ 相位偏移。通过 DCM 上的四个输出引脚 CLK0、CLK90、CLK180、CLK270 可以得到与原来时钟信号相位偏移差 0 度、90 度、180 度和 270 度的时钟信号。

此外，还可以通过设置一个相位偏移值 PS(是一个取值从－255 到 255 的整数)来实现任意相位的偏移：

$$\text{Phase shift} = (PS/256) * PERIOD_{CLKIN}$$

✧ 扩频特性（EMI Reduction)。DCM 提供了一种对输出时钟进行扩展频谱处理的功能，用以降低输出时钟对外的电磁干扰和能量辐射。

3-6 XILINX XC9500 系列 CPLD 器件介绍

XC9500 系列 CPLD[9]采用了 ISP 技术。采用 ISP 技术之后，器件编程不再需要硬件编程器，只需要一根下载电缆通过下载软件和器件的编程接口相连就可以实现。而且，可以在产品设计和制造的任何一个环节，甚至在产品卖给最终用户后，仍可以方便地改写 ISP 器件中的逻辑功能，即可以通过软件进行硬件升级。应该说 ISP 产品在 CPLD 产品中占有主要地位。

XC9500 系列产品是一种采用先进的 FastFLASH ISP 技术的 CPLD。其采用基于 Flash 的 0.35um 技术，可以提供 10 000 次以上的编程/擦除周期。该系列 CPLD 的宏单元数目从 36 个到 288 个。器件封装的引脚数从 44 个到 352 个，有 PLCC、VQFP、CSP、TQFP、PQFP、HQFP、BGA、CSP 等芯片封装形式。在这里我们采用的 XC9536XL 44 个引脚的 PLCC 封装。

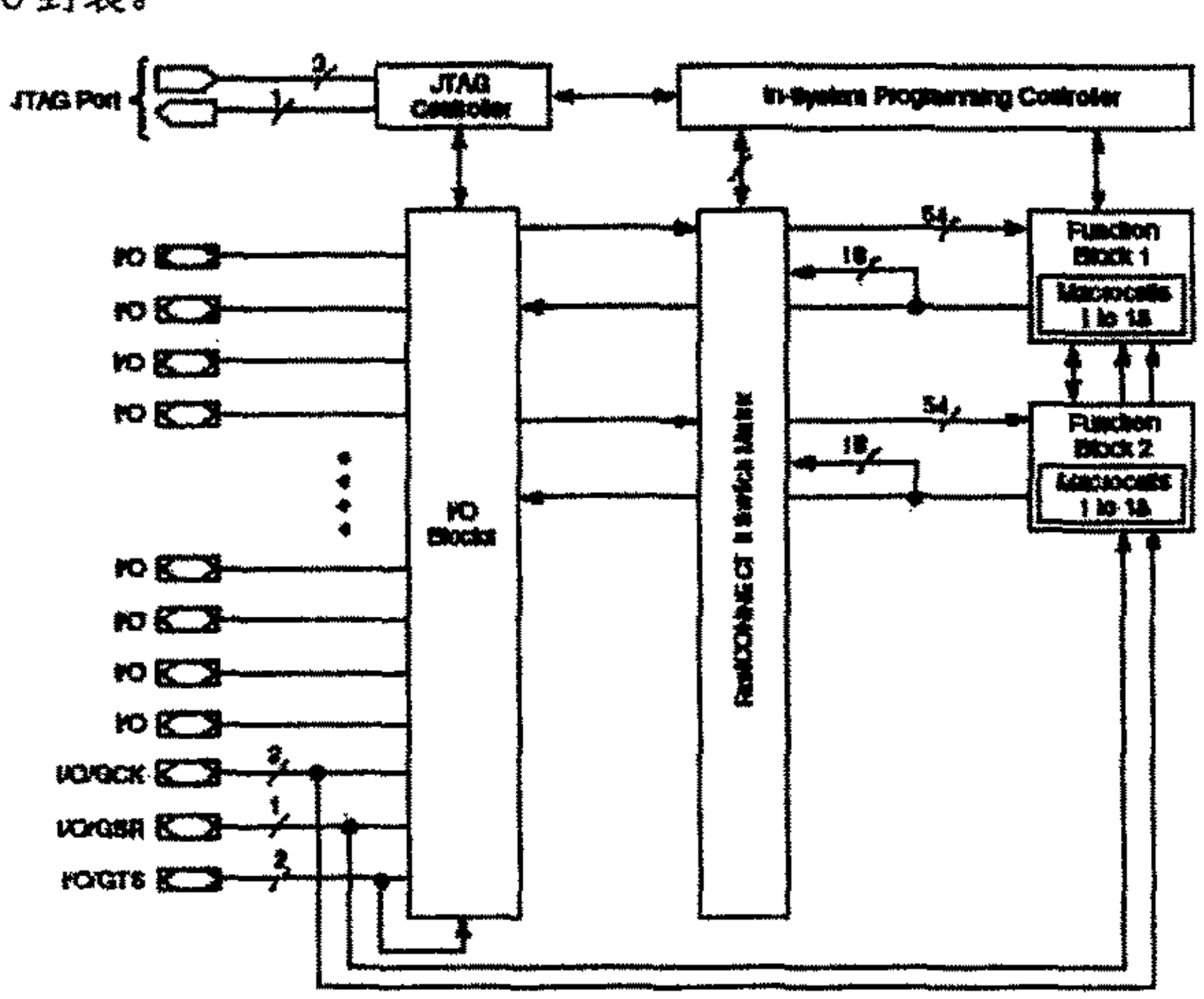


图 3－23 XC9500XL 结构框图

XC9500 系列 CPLD 共分为 5.0V、3.3V、2.5V 三种系列：

XC9500XV 系列 CPLD 为 2.5V；XC9500XL 系列为 3.3V；XC9500 系列为 5V。下面以 XC9536XL 3.3V 为例介绍其结构原理。

XC9500XL 系列 CPLD 的结构框图如图 3－23 所示。每一个 XC9500XL 系列 CPLD 由多个功能块 FB 和 I/O 块（IOB）组成，可用开关矩阵 FastCONNECTII 完全互联的子系统。IOB 提供器件输入和输出的缓冲，每个 FB 提供具有 54 个输入和 18 个输出的可编程逻辑的容量。可用开关矩阵 FastCONNECTII 连接所有的 FB 的输出、输入信号。每个 FB，有高达 18 个输出和相对应的输出使能信号直接驱动 IOB。

a） 功能块（FB）

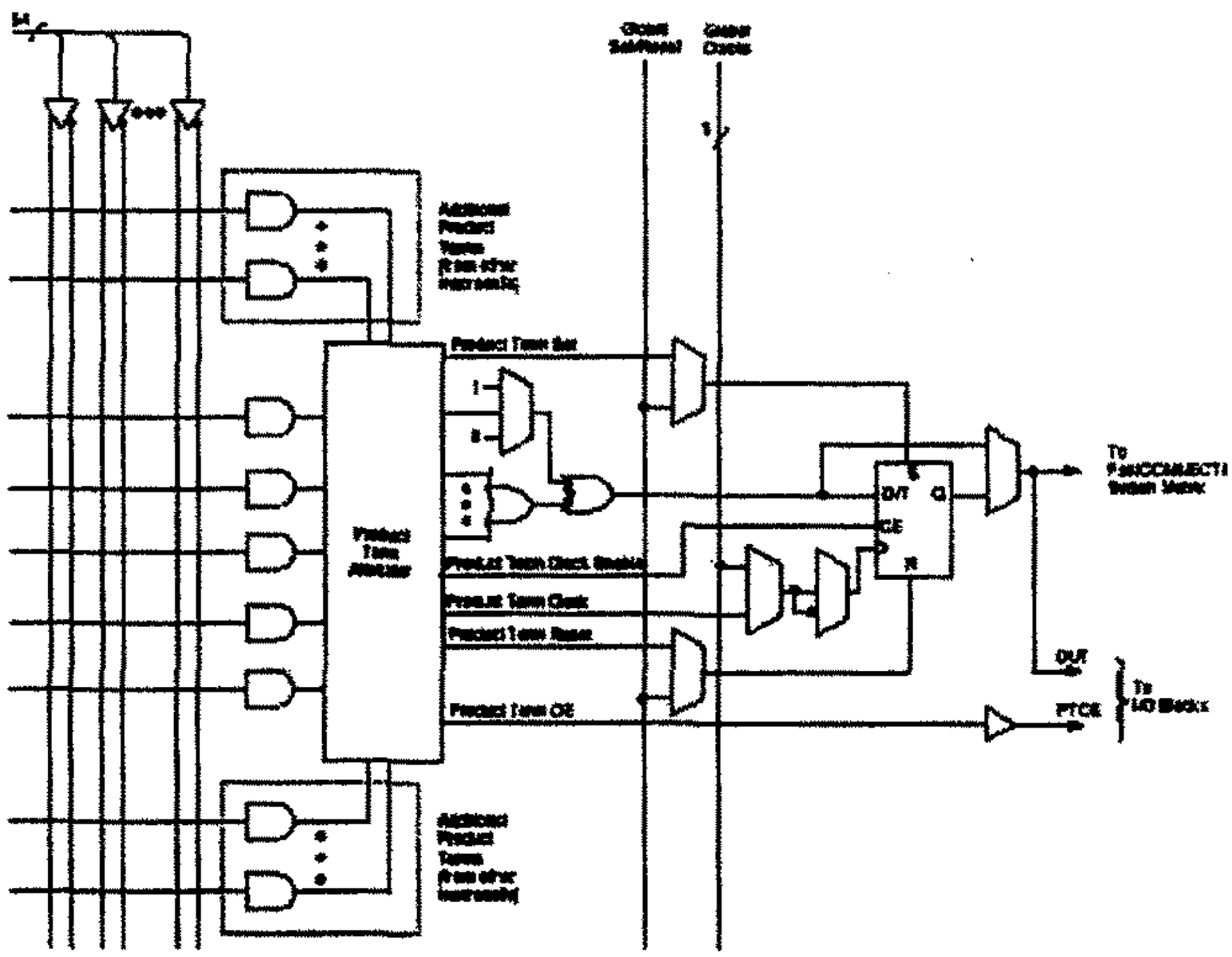


图 3－24 XC9500XL 宏单元框图

每个功能块由 18 个独立的宏单元组成。其中每一个宏单元都可以实现一定的组合逻辑或者是寄存器功能。FB 也可以接受全局时钟信号、输出使能信号和置位/复位信号。FB 一共由 18 路输出，用于驱动开关矩阵 FastCONNECTII。这 18 路输出以及它们的相应输出使能信号也用于驱动 IOB。

在 FB 内的逻辑利用一个乘积和表达式实现功能。54 个输入提供了 108 种信号输入给于阵列产生 90 个乘积项。这些乘积项可以通过乘积项分配器分配给每个宏单元。

b） 宏单元

每个 XC9500XL 器件的宏单元都可以独立配置实现一种组合逻辑功能或者是一种时序逻辑功能。宏单元以及相关的 FB 逻辑结构如图 3－24 所示。

来自与阵列的 5 个直接乘积项作为主要的数据输入，它们用来实现组合功能或者是作为时钟、时钟使能、置位/复位信号、输出使能等控制信号。与每个宏单元相连

的乘积项分配器决定如何使用这 5 个输入信号。

宏单元中的寄存器可以配置成一个 D 型、T 型触发器，或是实现某种组合逻辑功能。每个寄存器均支持异步的置位和复位操作。在上电时，所有用户寄存器均被初始化为用户定义的预加载状态。

每个独立的宏单元可以获得所有的全局控制信号。这些控制信号包括时钟信号、置位/复位信号和输出使能信号。宏单元寄存器的时钟来自于三个全局时钟信号或者是一个乘积项时钟信号。一个 GSR 输入可使用户寄存器处于用户定义的状态。

c) 开关矩阵 FastCONNECTII

开关矩阵 FastCONNECTII 将信号与 FB 输入连接在一起。所有的 IOB 输出和所有的 FB 输出驱动开关矩阵 FastCONNECTII。

d) I/O 块

I/O 块（IOB）为 CPLD 的内部逻辑和芯片的 I/O 引脚之间提供接口。每个 IOB 包括一个输入缓冲器、输出驱动器、输出使能多路选择器以及用户可编程的接地控制。

输入缓冲器可以兼容 5V TTL, 5V CMOS, 3.3V CMOS 以及 2.5V CMOS 信号。输入缓冲器使用内部的 3.3V 电压（V_{CCINT}），以确保输入门限固定不随电压 V_{CCIO} 改变。每个输入缓冲器提供滋滞（典型为 50mV），这样可以减少系统噪音。

输出使能信号有以下 4 中选择：来自宏单元的乘积项；任意一个全局输出使能信号（GTS）；总保持‘1’或者是‘0’。含有 72 或更少的宏单元器件有两个全局输出使能信号，而含有 144 个或更多宏单元的器件有 4 个全局输出使能信号。

每个 IOB 还提供了用户可编程的接地引脚。这可以使器件的 I/O 引脚配置成附加的接地引脚，以使未使用的 I/O 引脚出于低电平，同时还可以提供附加的器件接地能力。该接地引脚利用内部逻辑实现，其内部逻辑迫使产生一个与内部宏单元信号无关的逻辑低输出，这样内部宏单元逻辑就不受可编程接地引脚的影响。

每个 IOB 还提供总线保持电路，该电路在有效的用户操作期间被激活。总线保持的特点使不需要将未使用的引脚固定在前一个输入状态（无论是高电平还是低电平）直到新的输入信号到来为止。

3-7 VHDL 硬件描述语言简介

VHDL[1][5] 的英文全名是 Very High Speed Integrated Circuit Hardware Description Language, 诞生于 1982 年。1987 年底，VHDL 被 IEEE 和美国国防部确认为标准硬件描述语言 。自 IEEE 公布了 VHDL 的标准版本，IEEE-1076（简称 87 版）之后，各 EDA 公司相继推出了自己的 VHDL 设计环境，或宣布自己的设计工具可以和 VHDL 接口。此后 VHDL 在电子设计领域得到了广泛的接受，并逐步取代了原有的非标准的硬件描述语言。1993 年，IEEE 对 VHDL 进行了修订，从更高的抽象层次和系统描述能力上扩展 VHDL 的内容，公布了新版本的 VHDL，即 IEEE 标准的 1076-1993 版本，

（简称 93 版）。现在，VHDL 和 Verilog 作为 IEEE 的工业标准硬件描述语言，又得到众多 EDA 公司的支持，在电子工程领域，已成为事实上的通用硬件描述语言。有专家认为，在新的世纪中，VHDL 与 Verilog 语言将承担起大部分的数字系统设计任务。VHDL 主要用于描述数字系统的结构、行为、功能和接口。除了含有许多具有硬件特征的语句外，VHDL 的语言形式和描述风格与句法是十分类似于一般的计算机高级语言。VHDL 的程序结构特点是将一项工程设计，或称设计实体（可以是一个元件，一个电路模块或一个系统）分成外部（或称可视部分,及端口）和内部（或称不可视部分），既涉及实体的内部功能和算法完成部分。在对一个设计实体定义了外部界面后，一旦其内部开发完成后，其他的设计就可以直接调用这个实体。这种将设计实体分成内外部分的概念是 VHDL 系统设计的基本点。应用 VHDL 进行工程设计的优点是多方面的。

与其他的硬件描述语言相比，VHDL 具有更强的行为描述能力，从而决定了他成为系统设计领域最佳的硬件描述语言。强大的行为描述能力是避开具体的器件结构、从逻辑行为上描述和设计大规模电子系统的重要保证。

一个完整的 VHDL 语言程序通常包含实体(Entity)、结构体（Architecture)、配置（Configuration)、包集合（Package)、库（Library）5 个部分。前 4 种是可分别编译的源设计单元。实体用于描述所设计的系统的外部接口信号；结构体用于描述系统内部的结构和行为；包集合存放个设计模块都能共享的数据类型、常数和子程序等。配置用于从库中选取所需单元来组成系统设计的不同版本；库存放已经编译的实体、构造体、包集合和配置。库可由用户生成或由 ASIC 芯片制造商提供，以便在设计中为大家所共享。电路基本结构都由实体说明（Entity Declaration）和构造体（Architecture Body）两部分构成。实体说明部分规定了设计单元的输入输出接口信号和引脚，而构造体部分定义了设计单元的具体构造和操作（行为）。

实体说明：

Entity 实体名 IS

[类型参数说明][端口说明]

END 实体名

一个基本单元设计的实体说明以“Entity 实体名 IS”开始至“END 实体名”结束。

构造体：

ARCHITECTURE 构造体名 OF 实体名 IS

[定义语句]内部信号，常数，数据类型，函数等的定义；

BEGIN[并行处理语句]；

END 构造体名；

一个构造体从“ARCHITECTURE 构造体名 OF 实体名 IS” 开始至“END 构造体名”结束。

VHDL 语言有 3 种形式的子结构描述语句：BLOCK 语句结构、PROCESS 语句结构、SUBPROGRAMS 语句结构。

例如最常用的 PROCESS 语句结构：

[进程名]：PROCESS（信号 1，信号 2，…）

BEGIN…

END

PROCESSPROCESS 语句从 PROCESS 开始至 END PROCESS 结束。

关于 VHDL 程序编写的具体方法，请参考相关书籍，在这里不做详述。

3-8 XILINX FPGA 开发软件简介

XILINX 现场集成技术[1][2][10]所依赖的开发系统包括 Alliance 系列开发系统，Foundation 系列开发系统，ISE 系列开发系统，免费 Web Powered 软件，ModelSim XILINX 版，XILINX Modular Design, ChipScopeILA 等等。在这里首先介绍 XILINX ISE 系列开发系统，其主要的构成工具有：

- ✧ Foundation ISE 项目导航器，是总的项目管理器，包含设计过程中要用到的软件工具。在项目导航器中包含以下几种工具：设计输入、综合、设计实现、器件编程以及设计验证工具（设计验证包括功能仿真和时序时延仿真，以及静态时序分析）。
- ✧ 设计输入工具。包括硬件描述语言编辑器、ECS 原理图编辑器和状态机设计工具。基于 HDL 的 ISE 语言文本编辑器是十分灵敏的，它包括有色代码和对于保留词的即时帮助。HDL 编辑器支持 Verilog、VHDL 和 ABEL。可以自定义编辑器的显示方式，输入并格式化选项。ECS 原理图编辑器有以下三种功能：内置的设计规划检查功能、产生 VHDL 和 Verilog 结构网表的功能和可以产生匹配字符的功能。ISE 的状态机设计工具是 StateCAD Xilinx 版本和 StateBench Xilinx 版本，它们都是由 Visual Software Solutions 公司出品。可以帮助用户开发状态机。
- ✧ 设计实现工具，包括综合和实现两个部分。ISE 软件自带 XST 综合工具，XST 是由 XILINX 公司开发的，它支持 XILINX 公司 SpartanII，SpartanXL, Virtex 系列 FPGA 以及 XC9500 系列 CPLD 产品；此外，ISE 还留有于其它公司综合工具的接口，比如 Synplicity 公司的 Synplify 以及 Synopsys 公司的 FPGA Express 等。
- ✧ ISE 本身不带设计仿真工具，包括功能仿真、综合仿真、时序仿真都需要采用 Mentor 公司的 Modelsim 仿真工具完成。
- ✧ 编程工具。包括 JTAG 编程器，硬件调试器，时序分析器等。
- ✧ 辅助工具。比如 LogiBLOX 和 Core 产生器模块，是专门用于自动设计高级模

块的工具。

在 FPGA 产品的研发过程中，软件设计的仿真流程也是比较复杂的过程，很多初学者往往对此不是很清楚，下面就作一个简单的介绍：

如图 3-25 所示，设计的.vhd 原文件可以直接或结合 testbench 文件进行功能仿真；.vhd 原文件通过综合工具综合以后，形成了.vhm 测试文件和.edf 网表。其中.vhm 测试文件是用来作综合仿真的，也可以结合 testbench 文件进行仿真。.edf 文件则是综合结束后的网表，是下一步实现过程的输入文件。.edf 文件输入到实现工具，经过实现过程，产生了.vhd 结构文件和.sdf 延迟文件。这两个文件可以作为时序仿真的输入文件，进行时序仿真，其中.sdf 就是规定器件引脚间延迟时间的文件，.vhd 是源程序经过实现后的结构文件。

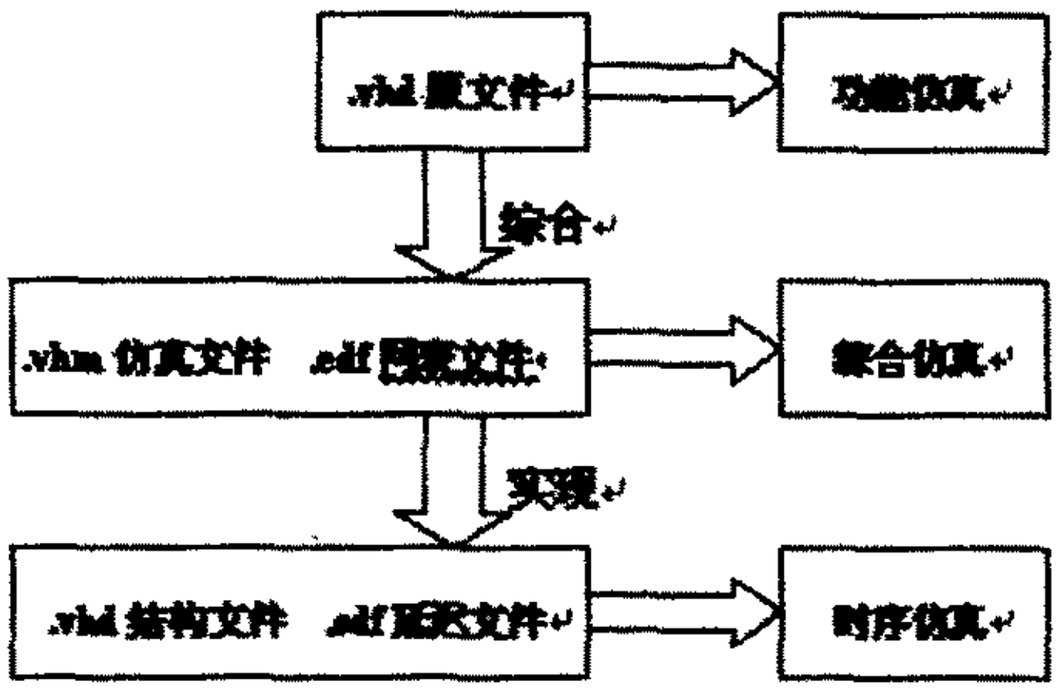


图 3—25 EDA 设计软件仿真流程

第四章　应用研究中的软硬件设计

4-1　硬件设计过程

4-1-1　帧存控制器的总体设计

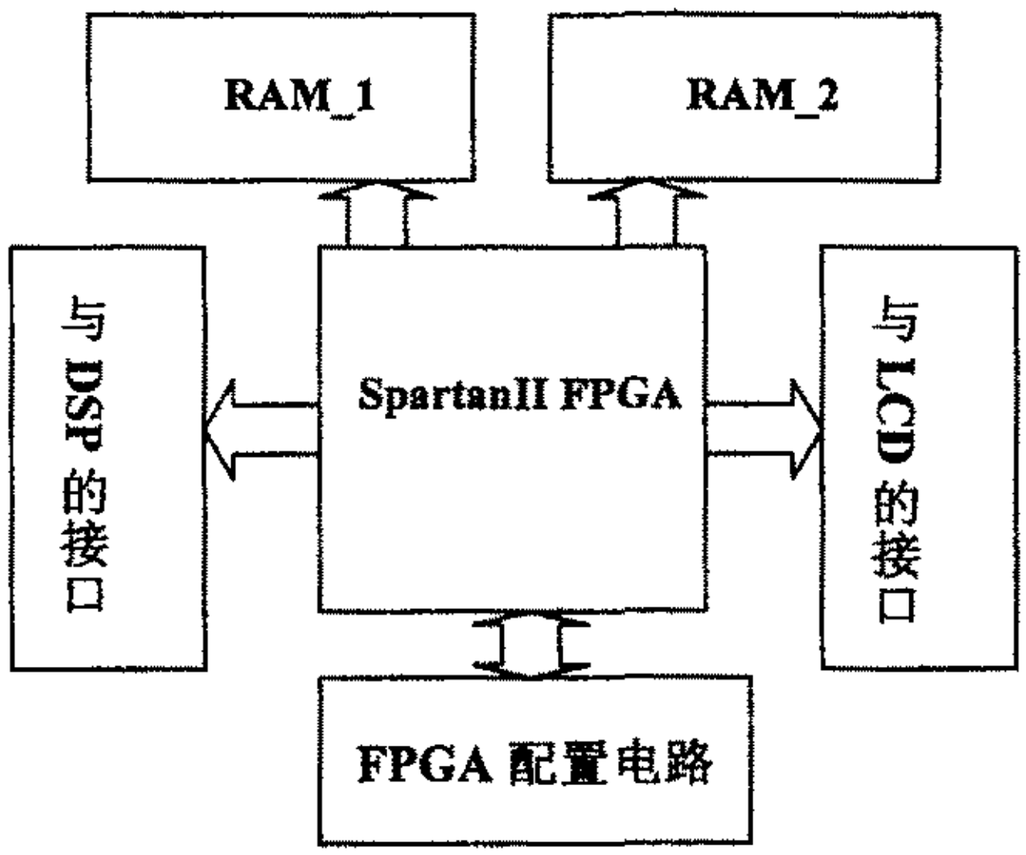


图 4—1 帧存控制器组成原理图

如图 4—1 所示，帧存控制器主要由五个部分组成。其中由 SpartanII FPGA 芯片制作帧存控制器主控制器。它分别与 LCD 和 DSP 有高速接口，可以从 DSP 中高速地读取数据，也可以将数据高速地送给 LCD。

另外，帧存控制器主控制器（SpartanII）还连接有两块高性能静态 RAM 作为帧存。正如前面章节所述，为了提高系统的性能，两片 SRAM 是分时复用的。LCD 具有 640×480＝307200 个象素，故每一片 SRAM 至少需要有 307200 个存储单元，所以我们选用了两片 512K×8bit 的静态 RAM 作为帧存。可见，要寻址到 307200 个空间，必须要用到 19 根地址总线，所以在 FPGA 与 SRAM 的接口中，需要准备 8 根数据线和 19 根地址线。

FPGA 的配置电路设计也是比较重要的一部分内容。由于在帧存控制器的使用中，对程序的初始配置时间要求不高，所以我们选用结构简单、速度较慢的 JTAG 模式。

根据图 4—1，可以比较容易地理解帧存控制器的工作原理：由 DSP 计算出一帧图象之后，按照顺序将这一帧图象上每一点的显示数值送给帧存，所有的这 307200

个象素值必须要在 20ms 中计算出来并全部送到一片帧存中。以 20ms 为周期 T，则在这个周期 T0 中，SpartanII 将 SRAM_1 的控制权完全交给 DSP，也就是说在 SpartanII 内部简单地将 SRAM1 器件的所有地址线和数据线和 DSP 的相应引脚相连。对于 DSP 来说，就相当与简单地控制一片片外 RAM，并且在 20ms 里算出 307200 个象素值并把它们都放到这片 RAM 中。

当 T0 周期里 DSP 将一帧数据放入 SRAM_1 后，在下一个 T1 周期里，SpartanII 从 DSP 那里收回 SRAM_1 的控制权，并把 SRAM_2 的控制权完全交给 DSP，DSP 又在 T1 这个周期里计算并传送新一帧的数据到 SRAM_2 中。所以由于动态切换 RAM 的功能，对于 DSP 来说，只是相当与外面挂有一片 512K×8bit 的存储器，它只要以 20ms 为周期，不断地计算一帧的显示数据并送给片外 RAM 就可以了。

同样在 T1 周期里，SpartanII 从 DSP 那里收回 SRAM_1 的控制权后，把其中的数据总线交给 LCD。由于 SRAM_1 中已经存在一帧显示数据，所以只要按照一定的时序，将 SRAM_中的数据读出并送给 LCD，就可以实现显示功能。由此可见，SpartanII FPGA 主控制器实现的功能除了动态切换两片 SRAM 的控制线之外，还需要提供一个时序发生器，用来提供 LCD 的显示时序。

在 DSP 计算过程中，只是计算出哪些点需要赋值，而对于背景来说，它们的值是不需要 DSP 改动的，比如显示一条直线，DSP 只需要计算出这条直线上点的位置，并把它们的象素值送给帧存就可以了，而不是要将所有点的象素值都送给帧存。这样，DSP 进行赋值操作的次数就大大减少了（实际上 DSP 或单片机也一直是这样工作的），其工作效率也得到了提高。但是，一个新的问题也出现了：在一帧数据显示完成后，帧存中仍然存储着这一帧的数据，当 DSP 要写入下一帧数据时，必须要先清除 SRAM 中留下的上一帧的数据。那么 DSP 又要对每一个单元先送“0”，然后再送相应的显示点，这样不是要耗费更多的时间吗？

实际上，对 SRAM 清零的工作不需要 DSP 来作，它完全可以由 FPGA 内部来完成：比如对于 SRAM_1 来说，完成了一帧的显示之后，它的控制权被 FPGA 收回并立刻又交给 DSP，新的一个工作周期又开始了。DSP 需要先耗用一段时间 T_count 来计算新的一帧数据，然后才开始向 SRAM_1 送数。T_count 是一段可以利用的时间，我们可以在这一段时间里，先不把 SRAM_1 的控制权交给 DSP，而是由 FPGA 对 SRAM_1 进行清零处理，在清零处理完成后，再将 SRAM_1 的控制权交还给 DSP。如果对 SRAM_1 清零所用的时间是 T_clear。现在的问题就在于，能保证 T_clear 不会耗用太长的时间吗？

对 SRAM_1 的清零处理是通过对每一个存储地址送 0 来实现的，所以清零时间的长短完全由存储器的存取时间来决定。我们选用的 SRAM 是 IDT 公司的 EDT71V424S 系列 3.3V 高速 CMOS 静态 RAM（图 4—2），一个读/写操作完全可以在 20ns 内完成，可以以 50M 的高速进行存取。由于在本系统中所用的时钟周期是 40M，所以在 40M 的时钟下，只需要 307200 个周期（大约 8ms）就可以完成对一片 SRAM 的清零。在这样一个时间里，DSP 是不能完成计算任务，而且还有 20—8=12ms 留给 DSP，所以这样的

设计是完全可行的。

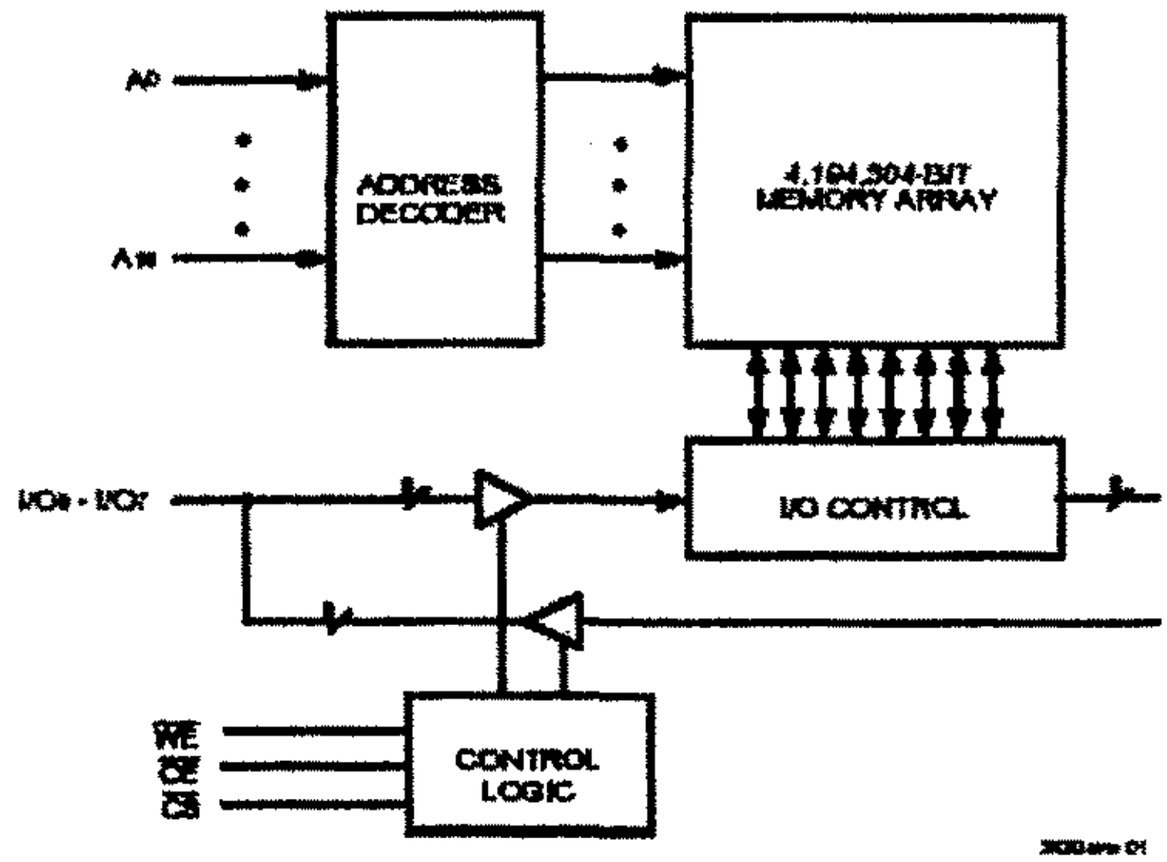


图 4—2 EDT71V424S SRAM 结构原理图

4-1-2　　帧存控制器和 LCD 以及单片机的接口设计

帧存控制器同 LCD 的接口主要由以下信号线构成:8 根数据线(LCD0-LCD7)，用来向 LCD 传送显示数据；时钟线 CLK，用来向 LCD 提供显示时钟，这个时钟是由 FPGA 内部产生的；行同步信号 HS，用来向 LCD 提供行同步信号；帧同步信号（VS)，用来向 LCD 提供帧同步信号；数据使能信号 DEN；它和行同步信号、帧同步信号也都是由 FPGA 内部产生的。如图 4—3 所示。

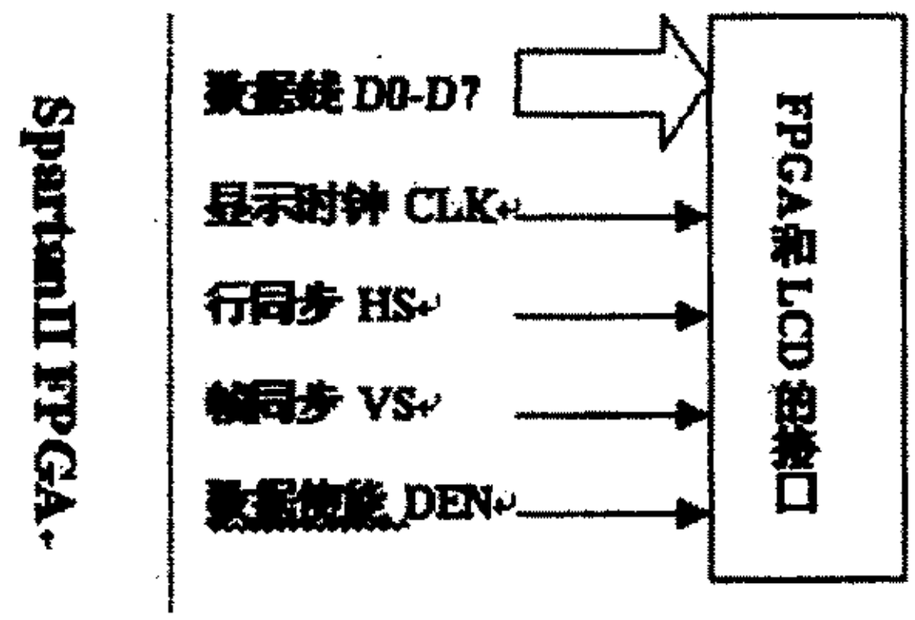


图 4—3 FPGA 同 LCD 的接口设计

为了减少干扰，需要在数据显示线、行同步线、帧同步线和数据使能线上连接电阻，并且在显示时钟信号 CLK 上加去耦电容，电容大小可为 33pF。

帧存控制器同 DSP 的接口由以下信号线构成：DSP 的 19 根地址总线(AD0-AD18)、读使能控制线 RD、写使能控制线 WR、帧显示完成的反馈信号线 ACK。其原理图如图 4—5 所示。

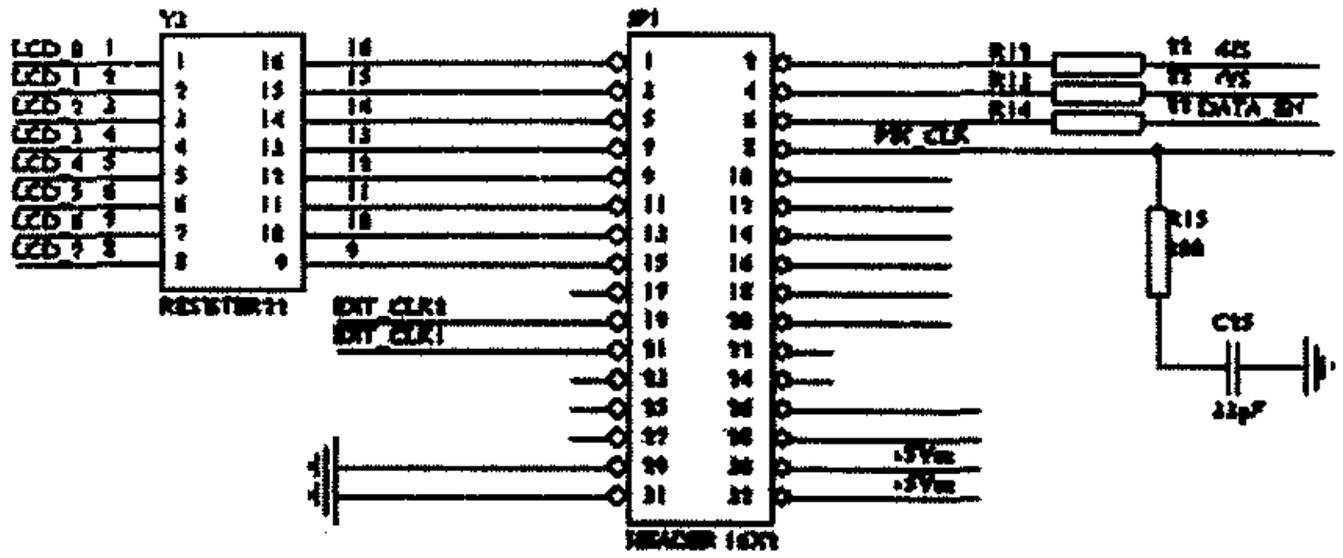


图 4—4 FPGA 同 LCD 接口的 PCB 原理图设计

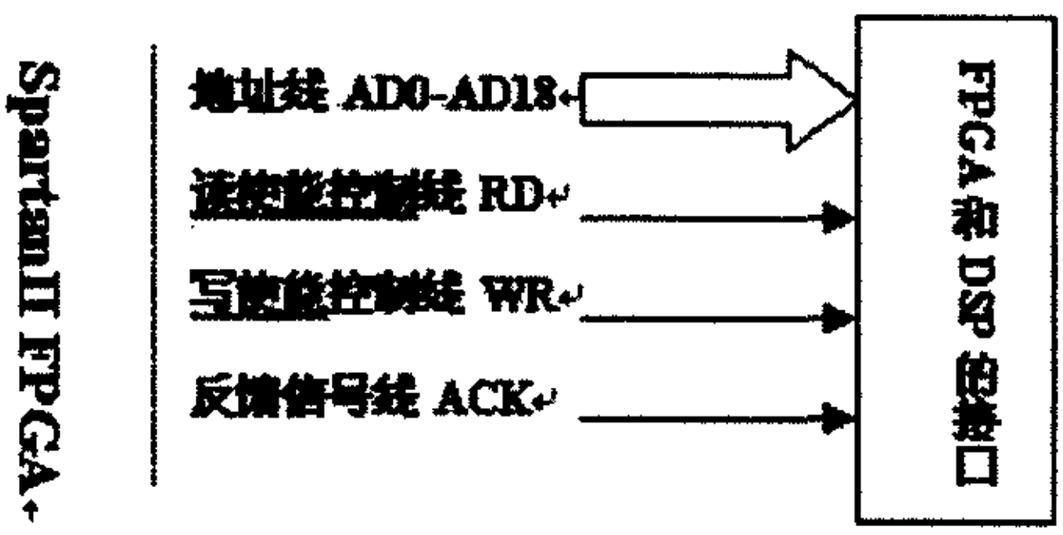


图 4—5 FPGA 和 DSP 的接口原理

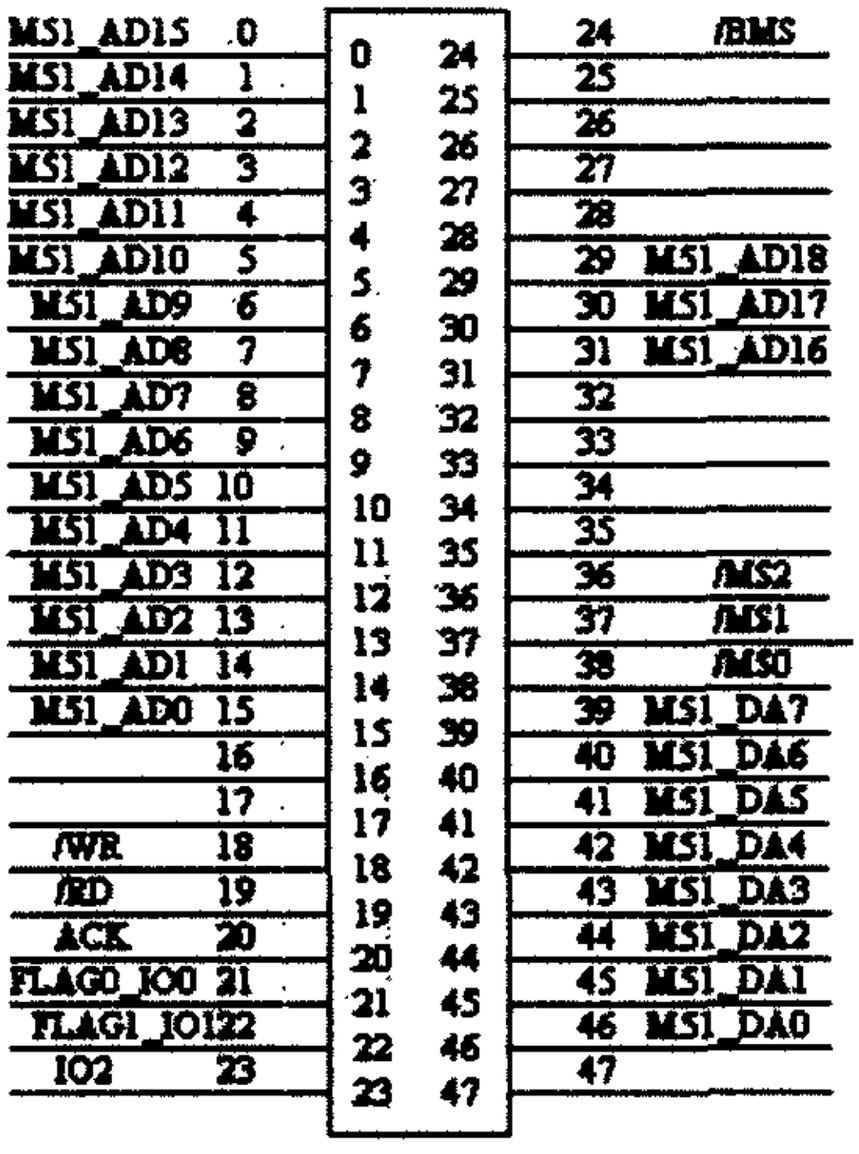


图 4—6 FPGA 和 DSP 接口的 PCB 原理图设计

（注：图中 FLAG0 等多余信号为设计预留线，在此不作解释）

4-1-3　　帧存与 FPGA 的接口设计

SpartanII FPGA 与帧存 SRAM 是帧存控制器中最主要的组成部分，它们之间的接口设计由以下信号线组成：19 根地址线（AD0-AD18）、8 根数据线(D0-D7)、读使能控制线 RD、写使能控制线 WR 和 SRAM 片选信号线 CS。如图 4—7 所示。

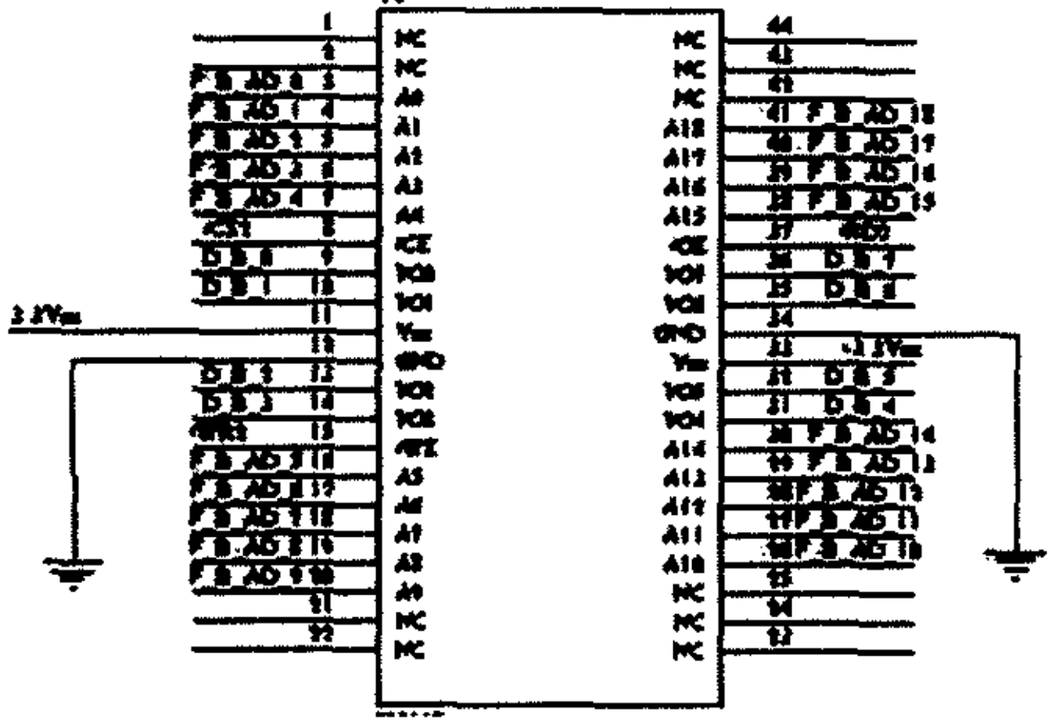


图 4—7 帧存 RAM 的 PCB 原理图设计

4-1-4　　芯片的程序导入电路设计

显示帧存控制器对程序导入的速度要求比较低。所以在这里我们采用 JTAG 的配置模式。JTATG 口一共由 4 根信号线控制：TDI、TDO、TMS、TCK。JTAG 数据可以有两种来源,一种是由上位机（电脑）通过 JTAG 编程电缆，直接由相应的编程软件下载到 SpartanII 器件中，这种方法通常用在调试阶段，在上位机上调试程序，然后立刻下载到器件当中去。到了具体应用阶段，就不能用电脑直接对器件进行编程了，这时需要用一片片外 ROM 或 FLASH 存储程序，在上电时自动配置或者是手动对 FPGA 进行编程，这就是现场可编程的概念。在进行电路设计的时候，需要了解这两种配置的方法并设计电路兼容这两种方法，这可以由跳线来解决，如图 4—8 所示，当 S4 接到 FPGA_TDI 并且 S3 断开时，电路执行由电脑上位机下载的配置方式；当跳线 S4 接到 ROM_TDI 并且 S3 合上时，电路执行的是直接由 ROM 或 FLASH 进行编程。我们可以通过跳线的设计，在程序设计阶段采用前一种方式，而当设计定型，进入产品运行阶段时，采用后一种运行方式。这中电路设计形式在很多场合都是常用的。

用上位机形式进行配置时，只需要将 JTAG 测试口的相应引脚同 FPGA 上的专用 JTAG 引脚连接起来就可以了。图 4—8 是用片外 ROM 进行 JTAG 配置的设计原理图。注意里面有一个按键 S5，是用来进行人工触发配置程序的，如果 S5 被按下，则新的配置过程就开始了。在这里使用的 ROM 是 XILINX 公司出品的 XC18V 系列的 FLASH 芯片，它是专门为 XILINX 公司 FPGA 芯片设计的 FLASH ROM，其工作时序都是按照 FPGA

的配置时序设计的，在使用时只需要将相应的引脚连接起来就可以了，因而比较简单。

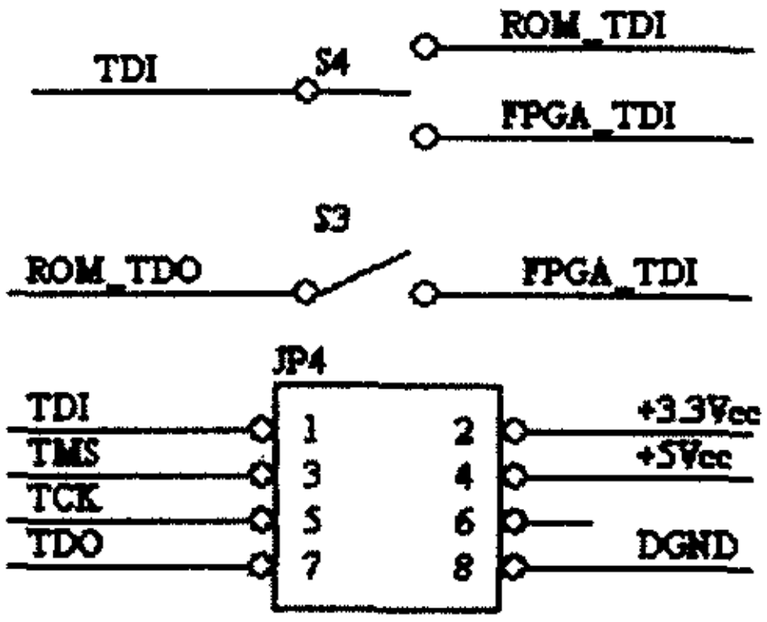


图 4—8 FPGA 配置的跳线设计

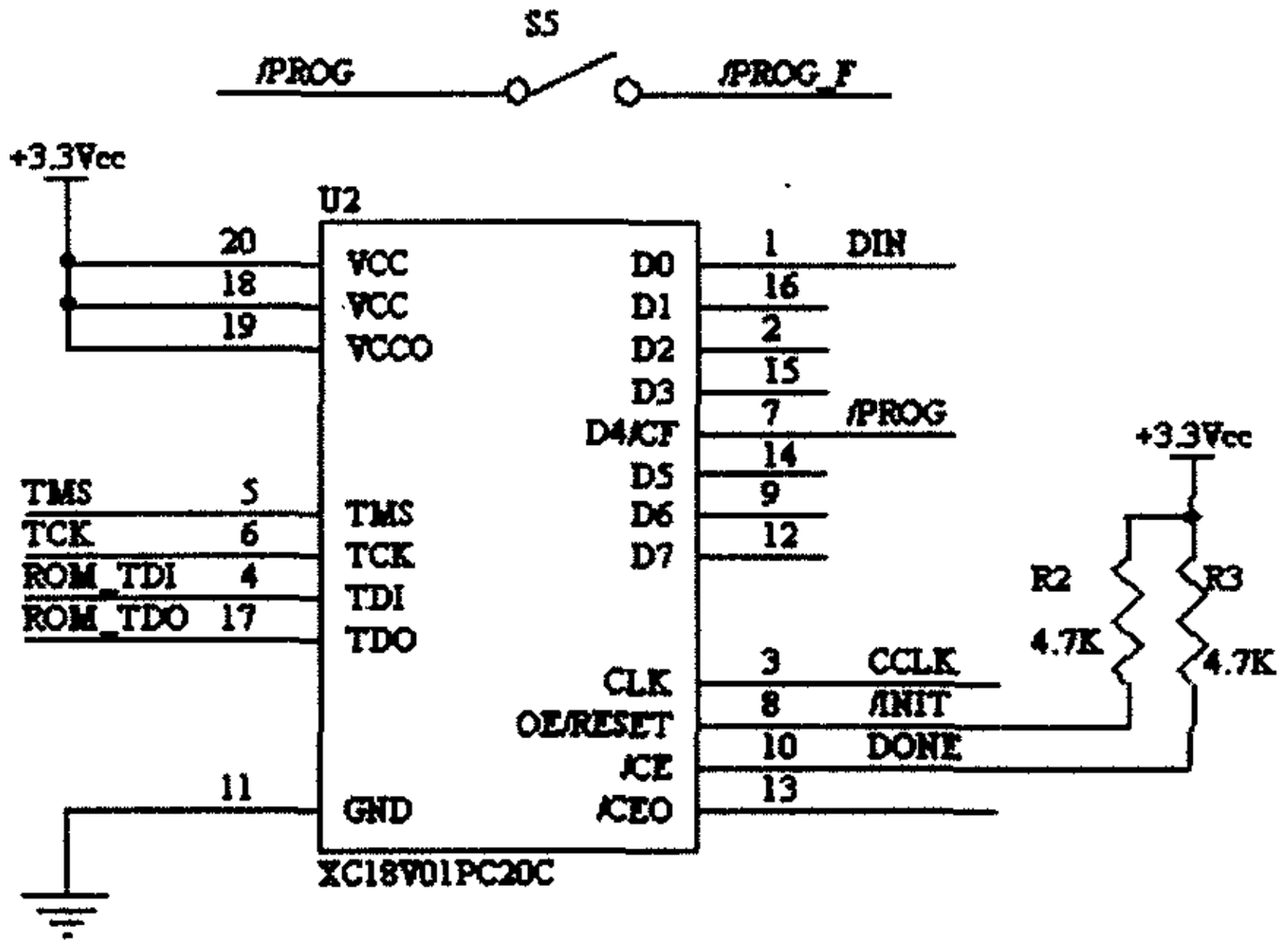


图 4—8 用片外 ROM 进行 JTAG 配置的 PCB 原理图设计

4-1-5　　电源的设计

在第三章介绍的芯片资料中，SpartanII 的工作电压是 3.3v 和 2.5v（其中核心工作电压是 2.5v，引脚供电电压是 3.3v）而 VirtexII 的工作电压是 3.3v 和 1.5v（核心工作电压是 1.5v，引脚供电电压是 3.3v）。在普通常用的电源中，大于 5v 的电压是常见的，而低与 5V 的专用电源就不多见了，所以我们需要设计一个供电系统，可以提供由 5V 到 3.3v、2.5v、1.5v 的转换。

提供这种转换的方法比较多，我们采用了模拟集成电源转换芯片。Linear 公司生产的 LT1083_3.3 模拟集成线性稳压芯片可以提供稳定的 3.3v 电压输出。它需要的电压输入是 5V，可以稳定地输出 3.3v 的直流电压。并且，采用这种线性稳压电源芯

片，外部不用另接分立元件，使用起来稳定、可靠。图 4—10 是 LT1083_3.3 的工作原理图：

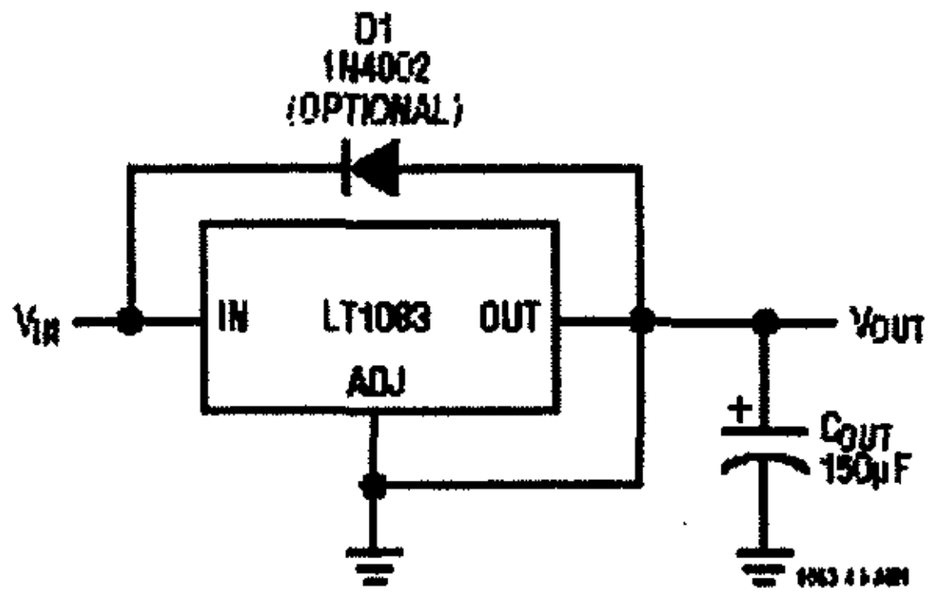


图 4—10 LT1083_3.3 工作原理图

2.5v 和 1.5v 电压的产生，我们同样选用了模拟集成 LT1083 可调电压芯片。图 4—11 是它的工作原理图。

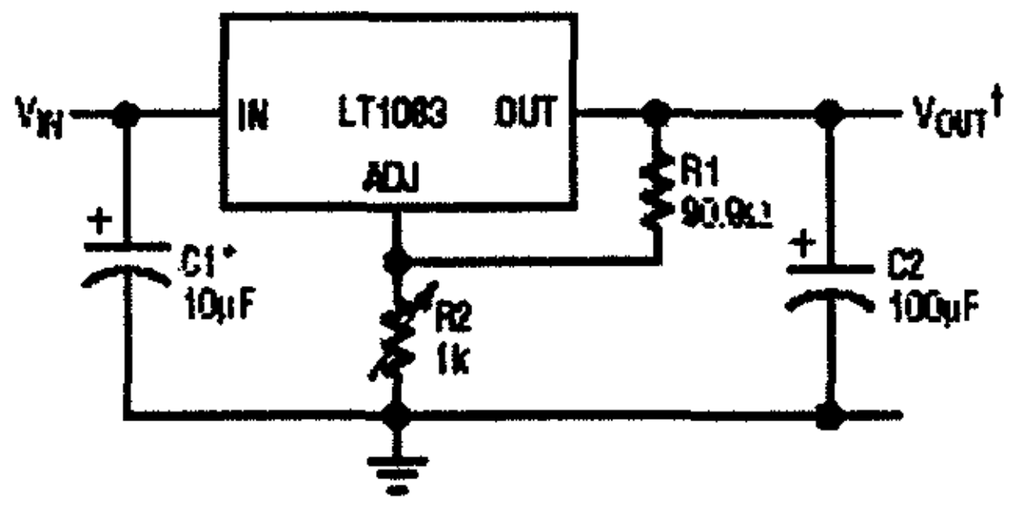


图 4—11 LT1083 可调电压原理图

由图可见，2.5v 和 1.5v 电压生成电路是一样的，都有一个可变电阻器 R2，根据可变电阻器阻值的不同，可以得到不同的输出电压。我们可以根据试验，确定出输出 2.5v 和 1.5v 电压各自对应的可变电阻值。

4-1-6　　PCB 板的设计

为了减小系统调试的难度，我们把帧存显示控制器的电源模块和验证模块专门拿出来单独设计。这样，为了设计并验证帧存控制器，就需要制作 3 块电路板。图 4—12 是它们的硬件组成框图。

- ✧ 帧存控制器 PCB 板，上面主要包括一片 SpartanII FPGA 芯片，两片 SRAM，同 LCD 的接口以及同中央处理器的接口。
- ✧ 电源模块，如上面一节介绍，我们需要制作一块电源板，实现从 5V 电压到 3.3v、2.5v、1.5v 的转换。

✧ 验证电路，为了验证帧存控制器的显示功能，可以先用结构比较简单的单片机代替 DSP 进行一些简单的运算功能，主要用于验证帧存控制器的工作是否正常。（对于帧存控制器的功能来说，简单的和复杂的图形没有什么区别）。这样还需要设计一片单片机电路 PCB 板。

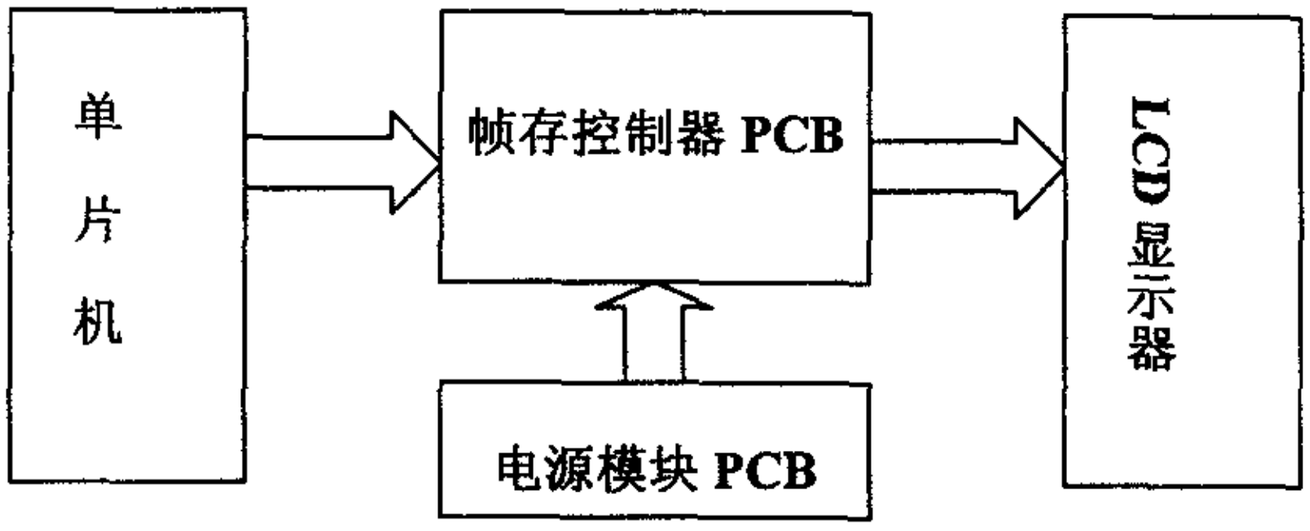


图 4—12 帧存控制器系统的硬件组成框图

4-2 软件设计过程

4-2-1 帧存控制器软件编写的模块划分

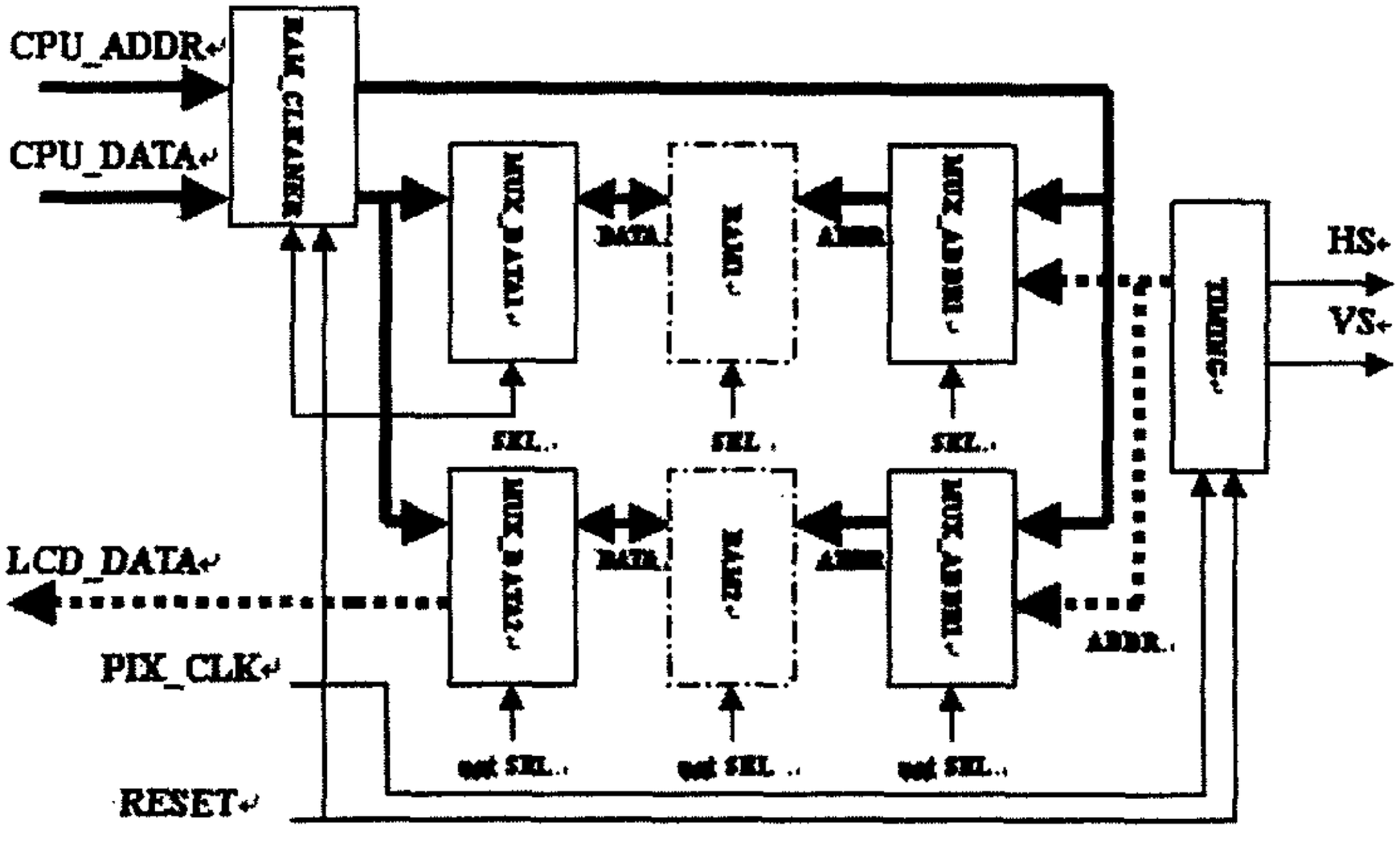


图 4—13 帧存控制器软件模块构成(其中加粗线为 8 位或 19 位总线)

在编写帧存控制器程序的过程中，需要将整个功能分成几个部分，如图 4—13 所示。可见整个帧存控制器主要由下面几个模块组成：

✧ 一个时序产生模块 timing。这个模块的功能主要是产生 LCD 显示需要的各种

时序信号，如行同步信号 HS、帧同步信号 VS。并且还要产生内部的显示地址信号，用以在点时钟 PIX_CLK 的驱动下，将帧存 RAM 中的数据顺序读出并馈给 LCD 显示板。

- ✧ 两个地址选择器 MUX_ADDR1 和 MUX_ADDR2。这两个模块分别控制两片帧存 RAM 的地址总线，在不同的工作周期里，轮流地将帧存 RAM 的地址总线和外部 CPU 的地址总线或和 timing 模块产生的内部显示地址总线相连。
- ✧ 两个数据选择器 MUX_DATA1 和 MUX_DATA2。这两个模块分别控制两片帧存 RAM 的数据总线，在不同的工作周期里，轮流地将帧存 RAM 地数据总线和外部 CPU 的数据总线或和 LCD 显示板的数据总线相连。需要注意的是，和地址选择器不同，在数据选择器中，两种不同的数据总线连接的数据流向是不同的，所以在编写程序的时候，需要遵循可综合的双向数据缓冲器的程序风格。
- ✧ 帧存 RAM 硬件擦除模块 RAM_CLEANER。在一帧数据显示完毕后，存放这一帧数据的 RAM 就紧接着被 CPU 控制，在下一个显示周期里，CPU 将下一帧的数据写入这片 RAM。但是，CPU 只是传送那些需要显示的点，而背景是不需要传送的，所以在 480×640 个存储单元中，CPU 只需要改变其中的一部分。可是，这片 RAM 中还存在着上一帧的显示数据，CPU 如果只是简单地把新的数据送入，就会出现两帧图象重叠的情况，所以，在送数之前，必须对 480×640 个单元全部清零。CPU 对片外存储器的操作指令一般都大于三个周期，这个工作如果由 CPU 来完成，要耗费大量的时间，这是不可忍受的，所以我们考虑采用 FPGA 内部对 RAM 进行清零处理：在完成了一帧的显示数据之后，FPGA 立刻控制这片 RAM 的地址和数据总线，以两倍时钟（比如 80M）对 RAM 每一个单元进行高速写零。由于我们采用的是 IS61LV 高速 RAM，其存取速度小于 10 个 ns,所以完全可以用 80M 的高速对其写零。这样，在不到 4 个 ms 的时间内，就可以对 RAM 进行完全清零。我们设计 RAM_CLEANER 模块，就是要在系统启动和每一帧显示完成后对 RAM 进行自动硬件高速清零。

在图 4—13 中，还有用虚线画的两个帧存 RAM，由于它们是帧存控制器主控制器外部的器件，所以用虚线表示，代表它们是外部器件。

除了图 4—13 中表示的几个模块外，在帧存控制器中，还需要有时钟产生模块，用来根据系统时钟产生 LCD 显示所需要的点时钟信号 PIX_CLK，这个时钟产生模块可以用 SpartanII 器件中的 DLL 模块来实现，DLL 模块可以实现时钟的分频和扩频，并且还可以提供给每个片内的寄存器或锁存器无延迟相位偏差的时钟信号。由于系统的工作时钟和 LCD 显示时钟 PIX_CLK 是不同的，此外还要产生对 RAM 进行高速硬件清零的时钟信号，所以需要用 DLL 的分频功能进行频率转换并提供稳定的、无相位偏差的时钟。

另外，还需要一个选择信号 SEL 的产生器，根据不同的帧显示周期改变两个地址选择器和两个数据选择器的选通状态。

下面结合图 4－13 来举例说明帧存控制器是怎样工作的：

如果 SEL 信号为 1，数据线多路选择器 MUX_DATA1 连通的是 LCD 的数据线和 RAM1 的数据线，地址线多路选择器 MUX_ADDR1 连通的是 TIMING 模块的地址线和 RAM1 的地址线，这时 RAM1 的工作模式是从 RAM1 中读取数据，并在一定的时序驱动下送到 LCD 上显示。

与此同时，not SEL 信号为 0，数据线多路选择器 MUX_DATA2 连通的是 CPU 的数据线和 RAM2 的数据线，地址多路选择器 MUX_ADDR2 连通的是 CPU 的地址线和 RAM2 的地址线，这时 RAM2 完全被 CPU 所控制，它的工作模式是 CPU 写入数据到 RAM2 中。

当下一个显示周期到来时，SEL 信号由 VS 信号驱动进行反向操作，RAM1 和 RAM2 的工作模式也同时进行交换，从而实现了两片帧存 RAM 的分时复用。

在编写 VHDL 程序之前，我们还需要讨论一下面向综合的 VHDL 语言编写方式。在第三章曾经提到过，用 VHDL 语言编写的程序要经过综合软件综合成电路的具体的电路可实现形式，而因为现在的综合编译器还不能将所有的 VHDL 表述完全翻译成具体电路，所以我们在编写面向综合的 VHDL 语言时，不能象写 C 语言那样随心所欲，而必须尽量按照电路的具体构成来组织硬件描述代码。这里就要用到自顶向下的设计思想，将整个功能分成由若干个子模块组成，分别完成每个模块的功能设计，然后将它们连接起来。并且在每个模块的设计过程中，尽量使用已经存在的数字电路模块和已知的能够被综合器综合的语法。

下面举一个这方面的例子：在众多高级语言如 C 语言中，数组的作用是十分强大的，很多功能都可以用数组来完成。同样，VHDL 语言中也提供了数组，并且对数组的操作也是相当灵活的，支持一维、二维数组。但是，当编写面向综合的程序时，就需要注意，大多数综合器是不支持二维数组的，所以存在二维数组的 VHDL 程序一般是不能综合的。并且，在对一维数组的使用也要十分地小心，因为对于综合器来说，大多数情况下是将数组结构综合成寄存器的形式，也就是说数组中每一个 bit 的数值将被综合成一个寄存器，一个简单的数组就有可能占用大量的寄存器。在 CPLD 和 FPGA 器件中，寄存器的数目是十分有限的，所以最好不要在程序中简单地使用数组，而是尽量将原来用数组表达的语句编译为用 FPGA 中的 RAM 实现。要将数组综合成 RAM 形式，就需要遵循特殊的编程风格，这些风格在不同的综合器中有不同的规定，需要查阅综合器的帮助文档。

4-2-2　　基于 LCD 显示特性的时钟发生器 timing 模块设计

时序产生模块是在帧存控制器设计中相对比较复杂的一个模块，首先需要明确地确定几个重要的参数：（具体的时序图如图 2－3、图 2－4、图 2－6 所示）

th ：行同步信号周期，一般为 800 个 CLK，约 32us。

thd：显示周期，是 DE 信号有效的时间，应该为 640CLK。

thp：th 信号脉冲宽度，为 96CLK。

thf：信号前间，为 16CLK。

thb：信号后间，为 48CLK。

tv ：帧显示周期，为 525 个行同步周期，约 16ms －20ms ，即满足大于 50HZ 的显示速度。这个参数对于整个系统都具有重要的意义，也就是说，要在这一段时间内(大约 59.3HZ)，完成所有的数值计算和数据显示。

tvd：数据使能时间，应该为 480 个行同步周期时间。

tvf：信号前间，为 12 个行同步周期时间。

tvb：信号后间，为 31 个行同步周期时间。

tvp：帧同步信号脉冲宽度，为 2 个行同步时间。

在程序中，多处使用了计数器来控制各个信号的周期。模块的框图和 VHDL 程序如下所示：

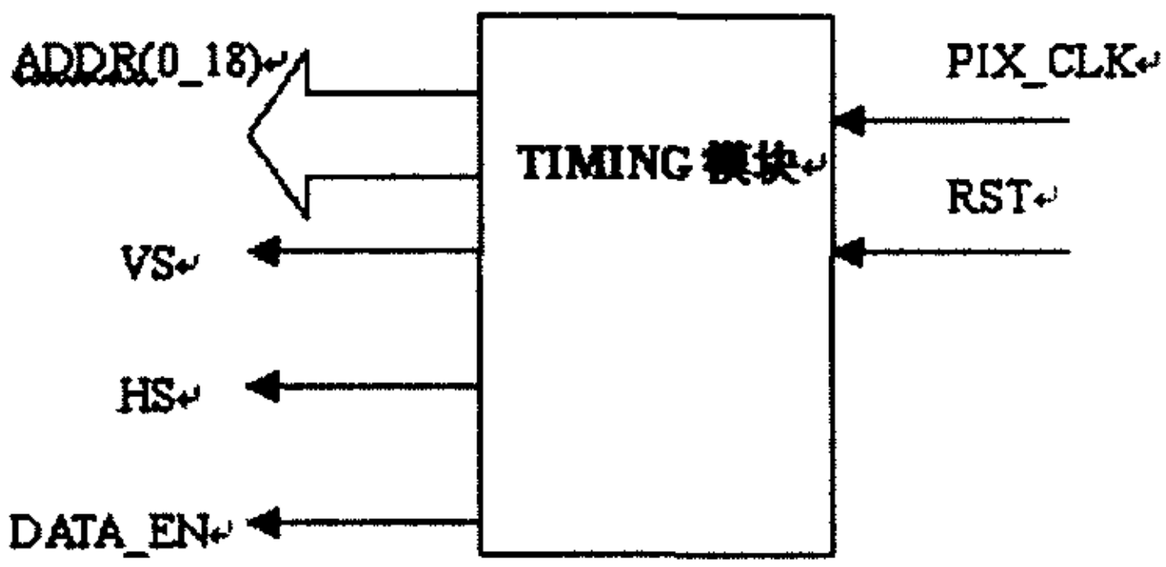


图 4－14 TIMING 模块框图

4-2-3　　用于控制 RAM 地址总线的数据多路选择模块设计

地址总线的数据多路选择模块框图如下图：

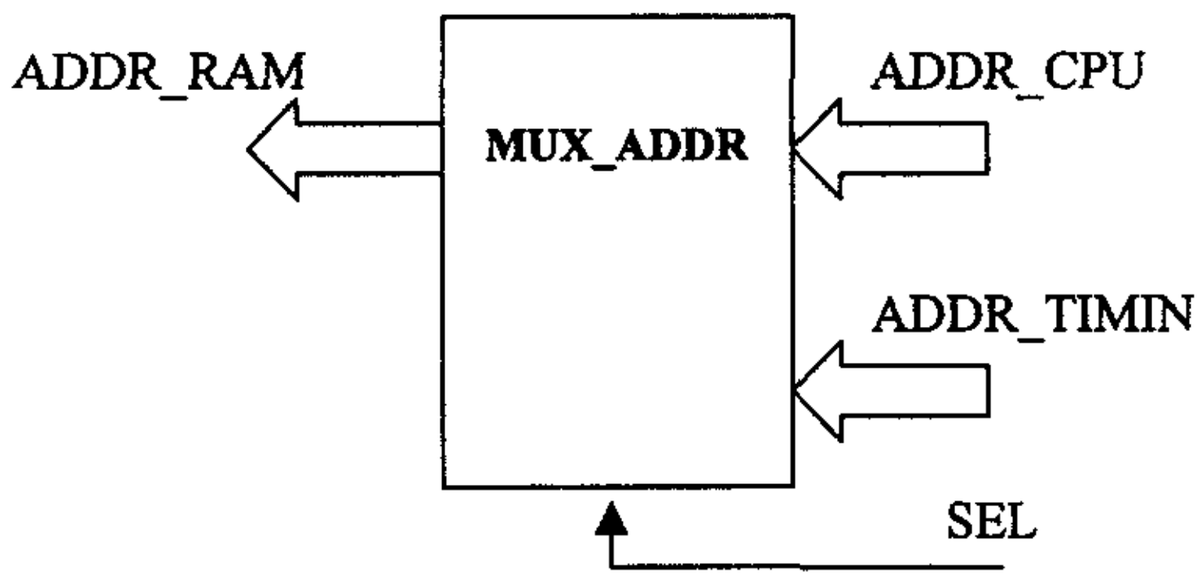


图 4－15 MUX_ADDR 模块框图

由 SEL 控制，在 SEL 为 1 时，输出 8 位总线 ADDR_RAM 与输入八位总线 ADDR_CPU 相连；当 SEL 为 0 时，输出 8 位总线 ADDR_RAM 与输入八位总线 ADDR_TIMING 相连。

4-2-4　　用于控制 RAM 数据总线的数据选择模块设计

数据总线的数据多路选择模块框图如下图：

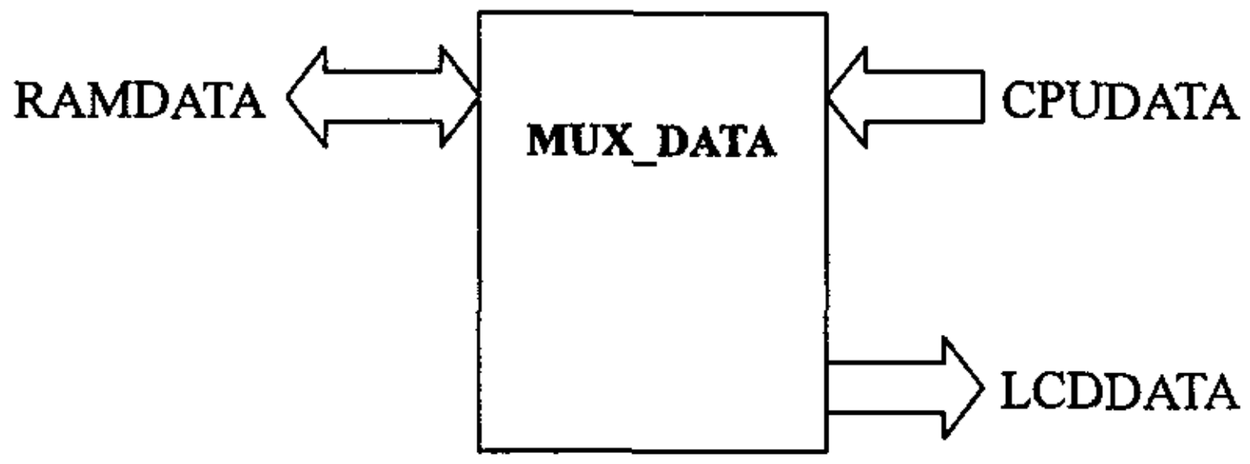


图 4－16 MUX_DATA 模块框图

由 SEL 控制，在 SEL 为 1 时，双向 8 位总线 RAMDATA 与输入八位总线 CPUDATA 相连；当 SEL 为 0 时，双向 8 位总线 RAMDATA 与输出八位总线 LCDDATA 相连。需要特别注意在不同的情况下，数据的流向不同。

4-2-5　　时钟产生模块的设计

这个模块用于产生无系统偏差的 LCD 显示点时钟信号和用于 RAM 高速擦除的高频信号，为了调试方便，在实际的系统中，我们采用的系统时钟是 40M，要产生的点时钟是 20M，硬件擦除高速信号时钟是 80M。所以可以采用 SpartanII 中的数字锁相环 DLL 来产生这两个信号。

4-2-6　　帧存 RAM 硬件擦除模块的设计

图 4－17 是 RAM 硬件擦除模块的框图，reset 置位或者当新的显示周期到来，sel 信号变化时，输出 addr 和 data 总线在 clk 的控制下向 RAM 中送 0，并且 ack 信号为 0，通知 CPU 不能送数。当 RAM 被清零以后，输出 addr 为输入 cpu_addr 的值，data 为输入 cpu_data 的值，并且 ack 信号变为 1，通知 CPU 可以送数。

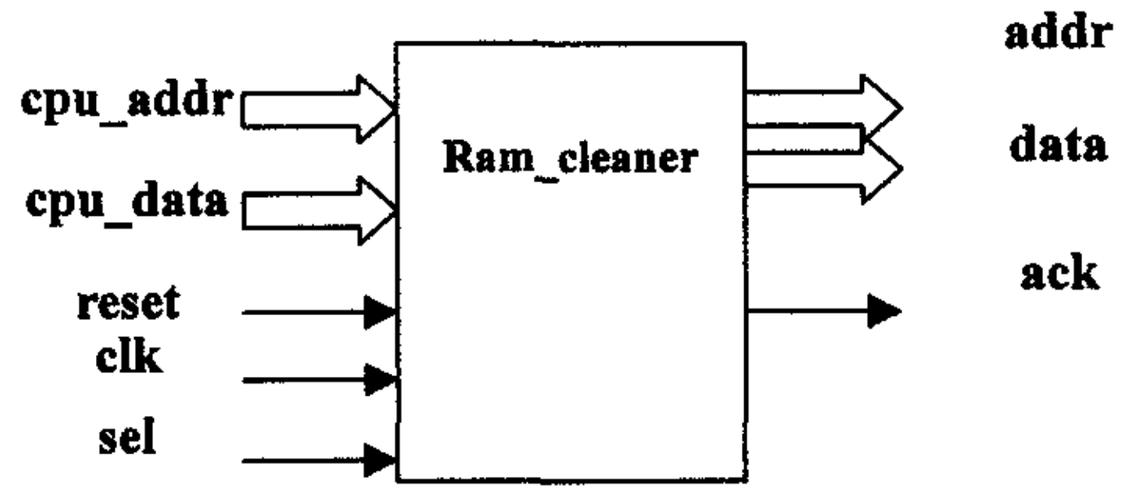


图 4－17 RAM_CLEANER 模块

4-2-7　　顶层模块的设计

完成了所有的子模块设计，只需要按照图 4－13，将各个子模块连接起来就可以

了。

4-2-8　　VirtexII 图形协处理器并行配置的软件设计

在第二章中曾经相信介绍了采用 VirtexII 芯片构成图形协处理器的原理，并提到了一种利用 XC9500 系列 CPLD 对 VirtexII 进行并行配置、从而实现动态重构功能的方法，配置原理如图 2－12。

第五章　结论

在本研究论文中，主要讨论了 FPGA 在图形综合显示系统中的应用。首先，在分析了国内外综合图形显示系统的发展情况后，提出了一种以 FPGA+DSP 进行高速图形运算以及显示的方法，并以这种思路为基础，介绍了一种图形综合显示系统的解决方案。

在深入了解了 Xilinx 公司各种 PLD 产品（FPGA 和 CPLD）的特性以及软件设计方法之后，主要深入研究了在综合图形显示系统中比较重要的显示帧存控制器的设计。在仔细介绍了显示控制器的工作原理的同时，结合 FPGA 器件特点，给出了具体的解决方案和设计细节。在实际工程应用中，成功地设计出基于 Xilinx SpartanII FPGA 器件的帧存控制器，与 LCD 显示器、单片机及 DSP 的配合使用中工作正常。在本论文中，对于显示帧存控制器的软硬件设计有比较详细的介绍。

由于研究时间有限，本人没有能够继续深入到使用 Xilinx VirtexII 器件设计高速图形处理芯片的研究中去，只是根据 VirtexII 器件的工作特点，解决了其在高速工作状态中的动态重构问题。同时，也没有能够对怎样使用 DSP 进行高速图形运算做深入研究。相信随着时间的推移和大家的努力，这种综合图形显示系统一定会取得令人满意的实用效果。

攻读硕士学位期间已发表的学术论文

1　《用 XILINX 公司 FPGA 器件实现异步 FIFO 的一种方法》 宋伯炜、张焕春，南京市第四届青年学术年会——信息化与先进制造技术 2002.10 江苏南京

参 考 文 献

[1] 《CPLD/FPGA 的开发与应用》 徐志军 电子工业出版社

[2] 《XILINX 数字系统现场集成技术》朱明程 东南大学出版社

[3] 《数字系统设计与 PLD 应用技术》蒋璇 臧春华 电子工业出版社

[4] 《VHDL 与 FPGA 设计》 胡凯华 中国铁道出版社

[5] 《VHDL 程序设计》 曾繁森 陈美金 清华大学出版社

[6] The programmable logic databook. XiLinx .Inc 1999

[7] Virtex-II Platform FPGA User Guide. Xilinx .Inc 2002

[8] Spartan-II 2.5V FPGA Family:Functional Description. Xilinx . Inc 2001

[9] XC9500XL High-Performance CPLDFamily Datasheet. Xilinx.Inc 2002

[10] Xilinx 5 Software Manuals. Xilinx. Inc 2002

[11] 512K x 8 HIGH-SPEED CMOS STATIC RAM. ISSI. Inc JULY 2001

[12] TFT COLOR LCD MODULE NL6448AC33-18 Datasheet. NEC. Inc

[13] LT1083/LT1084/LT1085 Fixed 3A, 5A, 7.5A Low DropoutPositive Fixed Regulators. LINEAR Technology.

[14] LT1083/LT1084/LT1085 7.5A, 5A, 3A Low Dropout Positive Adjustable Regulators. LINEAR Technology.

[15] High Speed Digital Design . Johnson&Graham . AMD Inc.

[16] 《EDA 技术与数字系统设计》 包明 赵明富 陈渝光 北京航空航天大学出版社

[17] 《EDA 工程概论》 曾繁泰 李冰 李晓林 清华大学出版社

[18] 《EDA 技术基础》郭勇 机械工业出版社

[19] 《EDA 工具 Protel98 及其设计应用》韩力 李晋炬 齐春东 北京理工大学出版社

[20] 《EDA 技术及应用》谭会生 张昌凡 西安电子科技大学出版社

[21] 《FPGA 设计及应用》褚振勇 翁木云 西安电子科技大学出版社

致　　谢

本课题的研究是在导师张焕春老师和经亚枝老师的关怀指导下完成的。他们治学一丝不苟、严格要求和诲人不倦的作风将使我终生受益。在此衷心感谢老师们对我学习、研究和生活的无私帮助。

感谢教研室其它老师和同学给我的帮助和支持，他们给我提出了宝贵的参考意见。

特别要感谢计算机测控教研室的朱耀东博士，他的指导贯穿于本课题研究的整个过程中。

最后要感谢我的父母、同学和朋友，他们的鼓励和帮助是我完成学业的动力和保障。

感谢所有关怀帮助我的人。