

版权声明

本用户手册的版权归西安唐都科教仪器开发有限责任公司所有，保留一切权利。
非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本用户手册的部分或全部，并以任何形式传播。

西安唐都科教仪器开发有限责任公司，1999-2020(C)，All right reserved.

TDX-CMX 实验系统用户手册

©版权所有 非经许可 严禁复制

技术支持邮箱：tangdukejiao@126.com

唐都公司网址：<http://www.tangdu.com/>

目 录

第 1 章	TDX-CMX 系统概述	1
1.1	系统功能及特点	1
1.2	系统构成	1
1.3	系统主要实验项目	5
第 2 章	TDX-CMX 系统软件安装	7
2.1	软件的安装	7
2.2	驱动程序的安装	7
第 3 章	TDX-CMX 系统硬件环境	8
3.1	系统硬件布局图	8
3.2	系统电源	9
3.3	系统实验单元电路	9
3.4	注意事项	25
第 4 章	TDX-CMX 系统集成操作软件	26
4.1	与 PC 联机说明	26
4.2	软件操作说明	26
第 5 章	TDX-CMX 系统检测功能说明	37
第 6 章	TDX-CMX 系统常见故障的分析及处理	38
附录 1	微程序流程图编程方法	39
附录 2	控制器单元和扩展单元对应 FPGA 引脚配置说明	46

第 1 章 TDX-CMX 系统概述

“TDX-CMX 计算机组成原理与系统结构教学实验系统”是西安唐都科教仪器公司推出的新一代计算机组成原理与系统结构的教学实验设备，是按照计算机组成原理和系统结构完整教学内容而精心研发设计的，与 TD-CMA 系统相比，实验教学内容更加丰富，实验观测手段更加完善，实验操作更加便捷，实验效果由此提高到了新的层次。

1. 对计算机组成原理的实验教学，增加了时序观测的内容，实验过程既可以基于通路图，也可以基于时序图来观测，使得实验教学内容进一步得到丰富。

2. 计算机系统结构实验教学内容全面升级，系统配置了多通路多总线的部件单元电路，可以并行计算机结构为主体来完整开展计算机系统结构的实验。

3. 各部件单元电路及整体结构进一步优化升级，在保证开放性的同时，使计算机各部件和各种复杂模型机电路的搭接更加简捷、高效和直观，极大提高了实验效率和成功率，更易于学生实验操作和学习掌握。

1.1 系统功能及特点

一、完整丰富的教学实验内容，确保实验室建设的先进性

1. 全面支持计算机组成原理实验教学

(1) 具有运算器、控制器、存储器、系统总线及总线接口设计等各部件教学实验内容，还具有简单模型机和复杂模型机的整机实验，以及输入、输出系统方面的实验，特别是提供了具有中断处理功能的模型计算机设计和具有 DMA 处理功能的模型计算机设计等实验，可使学生对计算机的组成结构和原理能有一个完整的认识和掌握。

(2) 对计算机组成原理的实验教学，还增加了时序观测的内容，使得实验过程既可以基于通路图，也可以基于时序图来观测，实验教学内容进一步得到丰富。

2. 计算机系统结构实验教学内容全面升级

系统配置了多通路多总线的部件单元电路，可以并行计算机结构为主体来完整开展计算机系统结构的实验。

(1) 支持并行结构计算机的部件电路实验：运算器和寄存器堆电路采用多端口多总线的电路设计，使学生掌握并行计算机运算部件和通路的设计及实现方法。存储器电路采用双端口结构，使学生掌握并行计算机存储系统的实现方法。

(2) 通过多端口多总线的部件电路，可灵活构建各种并行结构的模型计算机。通过 CISC 和 RISC 模型机实验，使学生学习 CISC 和 RISC 指令系统的特点，掌握这两种不同的指令系统及模型机的结构和设计特点。通过重叠结构模型机和三级流水模型机的实验，学习掌握以时间并行性为特征的计算机系统结构和设计特点。通过具有两条流水线的超标量模型机的实验，学习掌握以指令并行性为特征的计算机系统结构和设计特点。

二、具有新型结构的部件电路，可构成各种并行结构的模型计算机

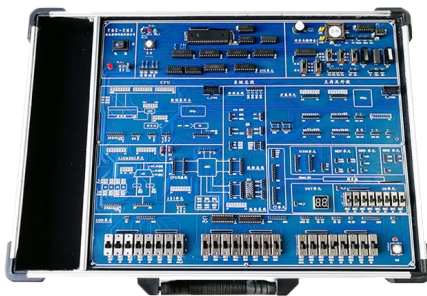
1、新型结构的部件电路

各部件电路结构创新设计，完全满足计算机组成原理和系统结构的教學要求，部件单元对用户完全开放，使学生顺利学习并掌握计算机各部件的功能和工作原理。

- (1) 具有多端口多通路的运算器和寄存器堆单元。
- (2) 微程序控制与硬布线组合逻辑控制复用的控制器单元。
- (3) 包含指令预取部件和双端口存储控制部件的 ABI 先进总线接口单元。
- (4) 先进系统总线单元，包含完整的数据总线、地址总线和控制总线，具有中断、DMA、存储、I/O 控制等各种总线信号。
- (5) 先进双端口存储器单元。
- (6) Super IO 芯片组，集成各种外部接口器件，包含中断控制器、DMA 控制器、定时计数器、输入输出控制器等。

2、实现各种并行结构的先进模型计算机

以各部件单元为基础，可实现不同结构和不同复杂程度的模型计算机：简单模型机，复杂结构模型机，精简指令模型机，重叠结构模型机，三级流水结构模型机，超标量模型机。



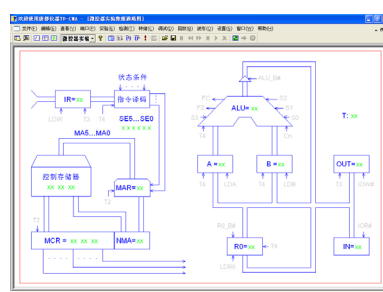
三、新型电路布局 and 结构，提高信号观测能力，电路设计和连接更加简捷和高效

- 1、计算机各部件进一步优化升级，整体结构更加规整完善，具有更好的开放性和设计空间，全面高效支持计算机各部件和各种模型计算机的开放性设计实验。
- 2、在保证电路开放性的同时，计算机各部件和各种复杂模型机电路的连接更加简捷和高效，极大提高了实验效率和成功率。
- 3、实验电路上布有清晰的计算机组成通路图，各部件的数据、地址和控制信号都具有 LED 显示，可充分显示各部件和模型计算机的当前工作状态和下一步的信息，不论单机使用还是连机使用，都能更加顺利地開展实验。
- 4、采用最新一代 Cyclone 10 系列器件和新一代设计工具软件，支持计算机部件和模型计算机的高效设计。

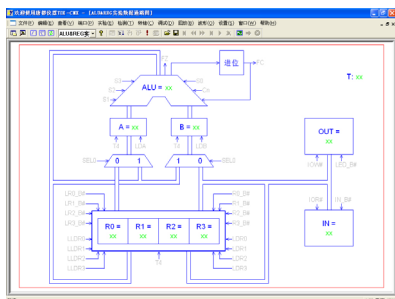
四、先进的实时动态图形观测调试，使学生更容易学懂计算机组成原理

(1) 先进部件实时通路图调试手段

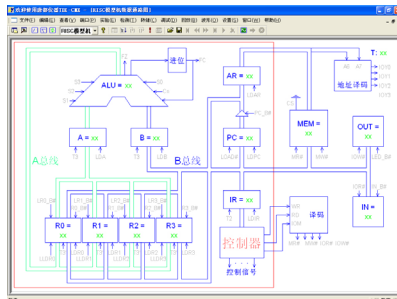
系统为各计算机部件（运算器、存储器、控制器）分别提供了基于计算机通路图的实时动态图形调试工具，使得学生可以轻松了解计算机部件的内部结构和操作方法，并可实时跟踪部件的工作状态。



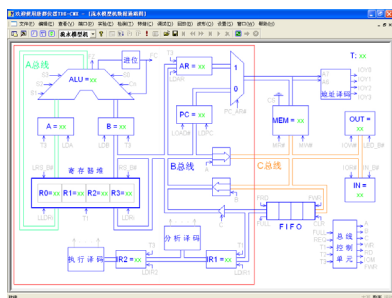
机、三级流水结构模型机、超标量结构模型机的实时动态观测和调试，使原本很复杂的具有并行结构的部件及计算机的运行状况和调试过程直观可见，使学生非常容易学习和掌握计算机系统结构的教学内容。



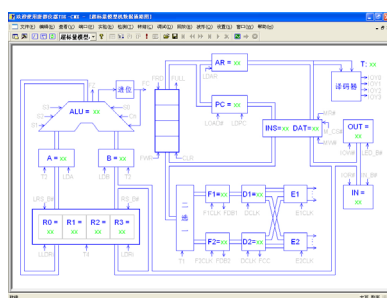
运算器与寄存器堆



精简指令集模型机



流水模型机



超标量模型机

六、扩展基于 SOPC 的“计算机系统结构”实验（需选配 MIPS32 计算机系统结构开发板）

系统提供了一个完整的 MIPS32 架构的处理器软核和基于 SOPC 的计算机系统结构实例。该处理器具有标准 32 位 5 级流水线架构，具备常用的 60 余条指令，与 MIPS 指令系统相兼容，并具有指令 CACHE 和数据 CACHE，解决了流水线中的数据相关、结构相关、控制相关等系统结构和设计中的各种核心问题。

系统采用了 MIPS 的编译器和链接器，同时还配套提供了编译、链接、下载的综合开发环境，该开发环境除支持“计算机系统结构”课程中的各种先进结构模型机的通路图调试功能之外，还可以编辑、编译、链接基于 MIPS32 机器的 C 语言程序，并提供基于 MIPS 的计算机系统程序的下载，以及程序的运行和停止等系统控制功能。

这一设计实例可以在 20MHz 工作主频下非常稳定的运行，非常适合以 IP 软核的形式作为嵌入式系统开发的核心，便于学生学习掌握计算机系统的设计方法和设计流程。

1.2 系统构成

TDX-CMX 实验系统硬件内容如表 1-1 所示。

表 1-1 TDX-CMX 系统硬件内容

控制器单元	指令寄存器 微程序控制器（微程序存储器，微命令寄存器，后续微地址寄存器，译码电路，指令预取控制部件） 基于 FPGA 的硬布线控制器
ALU® 单元	具有多端口多总线的运算器及寄存器堆
ABI 单元	地址寄存器 AR，程序计数器 PC，基于流水的指令预取部件，基于超标量的指令预取部件
控制总线	读写译码逻辑，CPU 中断使能寄存器，DMA 控制逻辑
数据总线	LED 显示灯，数据排线座
地址总线	LED 显示灯，地址译码电路，数据排线座
IN 单元	8 位开关，LED 显示灯
OUT 单元	2 位 7 段数码管
MEM 单元	基于 FPGA 的双端口存储器
8259 单元	基于 FPGA 的 8259 中断控制器
8237 单元	基于 FPGA 的 8257 DMA 控制器
8253 单元	基于 FPGA 的 8253 定时计数器
CON 单元	3 组 8 位开关，系统清零按钮
时序与操作台单元	时序发生电路，555 多谐振荡电路，单脉冲电路 本地主/控存编程、校验电路，本地机器调试及运行操作控制电路
SYS 单元	系统监视电路，总线竞争报警电路
逻辑测量单元	4 路逻辑示波器
扩展单元	基于 Intel 公司先进开放的 FPGA 设计单元，JTAG 下载电路，LED 显示灯，+3.3V 电源，GND

1.3 系统主要实验项目

（一）计算机组成原理实验

1. 运算器及设计实验

- （1）基本运算器实验
- （2）超前进位加法器设计实验
- （3）阵列乘法器设计实验

2. 存储系统及设计实验

- (1) 静态随机存储器实验
- (2) Cache 控制器设计实验

3. 控制器及设计实验

- (1) 时序发生器设计实验
- (2) 微程序控制器实验

4. 系统总线与总线接口实验

- (1) 系统总线和具有基本输入输出功能的总线接口实验
- (2) 具有中断控制功能的总线接口设计
- (3) 具有 DMA 控制功能的总线接口设计

5. 模型计算机的设计实验

- (1) CPU 与简单模型机设计实验
- (2) 硬布线控制器模型机设计实验
- (3) 复杂模型机设计实验

6. 输入、输出系统实验

- (1) 具有中断处理功能的模型计算机设计实验
- (2) 具有 DMA 处理功能的模型计算机设计实验
- (3) 典型 I/O 接口 8253 扩展设计实验

(二) 计算机系统结构实验

1. 多通路的运算器和寄存器堆

多通路的运算器和寄存器堆设计实验

2. 指令系统设计

基于 CISC 指令系统的模型机设计实验

基于 RISC 技术的模型机设计实验

3. 存储系统设计

FIFO 先进先出存储器实验

CACHE 控制器设计实验

4. 时间并行性为特征的计算机系统

具有指令预取功能的模型机设计实验

具有三级流水的模型机设计实验

5. 指令并行性为特征的计算机系统

具有两条流水线的超标量模型机设计实验

(三) 计算机系统设计实验 (需选配 MIPS32 计算机系统设计开发板)

- 1. 基于 MIPS32 的 CPU 设计及实现
- 2. 计算机系统设计及实现

第 2 章 TDX-CMX 系统软件安装

2.1 软件的安装

1. TDX-CMX 系统与 PC 微机相连

用 USB 电缆一根, 按图 2-1 所示, 将 PC 微机 USB 接口和 TDX-CMX 系统中的 USB 接口连接在一起。

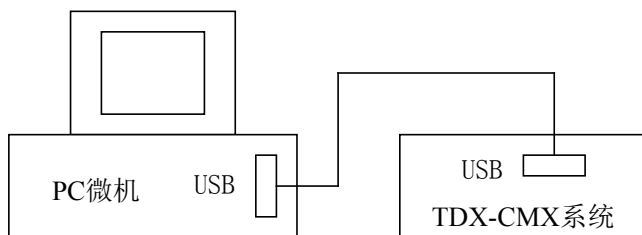


图 2-1 TDX-CMX 系统与 PC 微机联机示意图

2. TDX-CMX 系统联机软件的安装

(1) 软件运行环境。

操作系统: Windows XP/7/8/10

最低配置: CPU: 奔腾 300MHz

内存: 64MB

显示卡: 标准 VGA, 256 色显示模式以上

硬盘: 20MB 以上

(2) 安装软件与运行。

可以通过“资源管理器”, 找到光盘驱动器本软件安装目录下的‘安装 CMX.EXE’, 双击执行它, 按屏幕提示进行安装操作。“TDX-CMX”软件安装成功后, 在“开始”的“程序”里将出现“CMX”程序组, 点击“CMX”即可执行程序。

(3) 卸载软件。

联机软件提供了自卸载功能, 使您可以方便地删除“TDX-CMX”的所有文件、程序组或快捷方式。单击【开始】/【程序】打开“CMX”的程序组, 然后运行“卸载”项, 就可执行卸载功能, 按照屏幕提示操作即可以安全、快速地删除“TDX-CMX”。

2.2 驱动程序的安装

按 2.1 步骤连接 PC 机和实验箱, 打开实验箱电源。双击软件光盘中的 CH341SER.EXE, 点击 Install 安装驱动程序, 打开“设备管理器”---“端口”, 可以看到 USB-SERIAL CH341A (COM3), 驱动安装完成, 双击桌面的 CMX.EXE, 选择 COM3。联机软件左下角提示下位机已复位, 说明驱动安装成功, 实验箱联机正常。

第3章 TDX-CMX 系统硬件环境

3.1 系统硬件布局图

系统硬件的电路布局是按照计算机组成结构来设计的，如图 3-1 所示，最上面一部分是 SYS 单元，这个单元是非操作区，其余单元均为操作区，时序与操作台单元位于 SYS 单元的右侧。所有构成 CPU 的单元放在中间区域的左边，并标注有‘CPU’，CPU 对外表现的是三总线：控制总线、数据总线和地址总线，三总线并排位于 CPU 右侧。与三总线挂接的主存和各种 IO 设备，都集中放在系统总线的右侧，逻辑测量单元位于扩展单元的左侧。在实验箱中上部对 CPU、系统总线、主存及外设分别有清晰的丝印标注，通过这三部分的模块可以方便地构造各种不同复杂程度的模型计算机。

系统独立运行时，为了对微控器或是主存进行读写操作，在实验箱下方的 CON 单元中安排了一个开关组 SD17~SD10，专门用来给出主/控存的地址。在进行部件实验时，有很多的控制信号需要用二进制开关模拟给出，所以在实验箱的最下方安排的是控制开关单元 CON 单元。

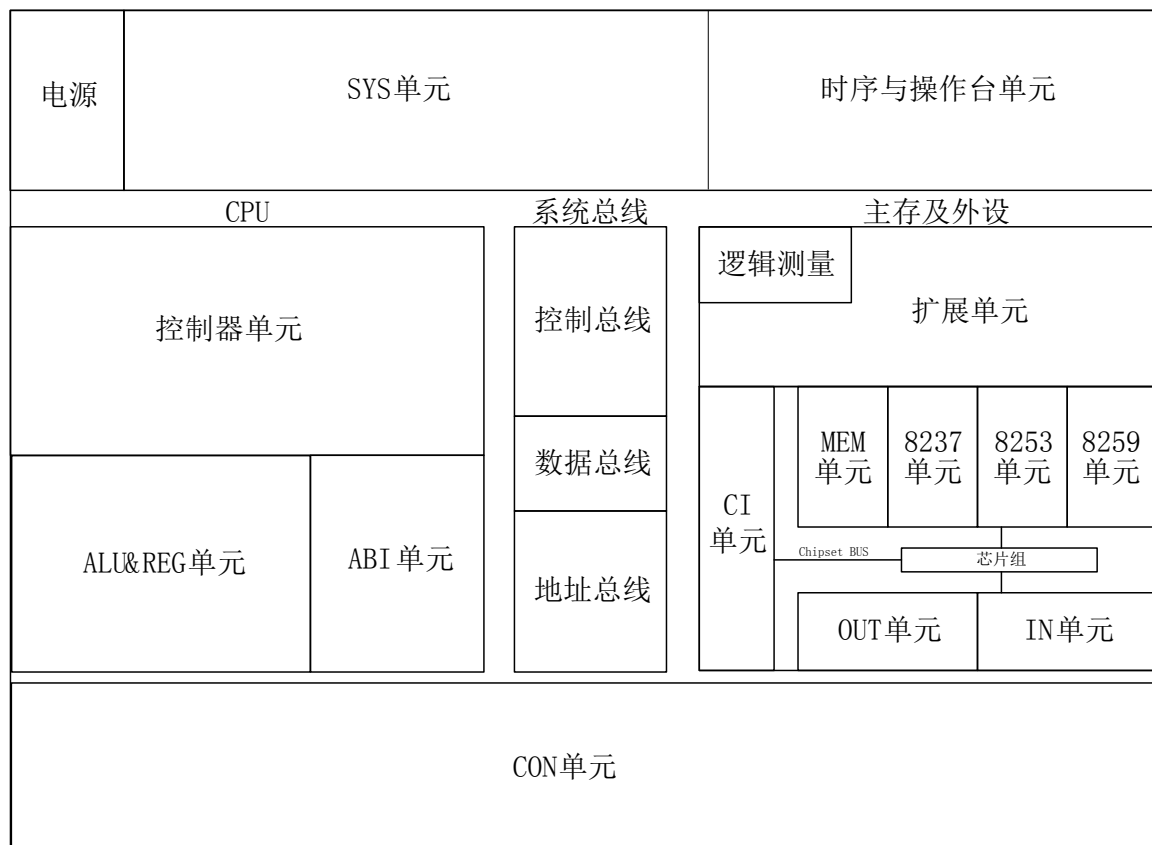


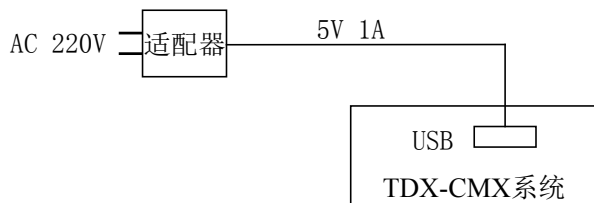
图 3-1 TDX-CMX 系统布局图

3.2 系统电源

TDX-CMX 系统通过 USB 电缆供电。

如果本实验室配有电脑，由 PC 机的 USB 口通过 USB 电缆向实验系统提供 5V、0.5A 的电源。

如果本实验室不配电脑，可选配 5V、1A 适配器通过 USB 电缆向实验系统提供电源。



3.3 系统实验单元电路

1. ALU® 单元

ALU® 单元主要由一片 CPLD 来实现的，ALU 内部构成如图 3-2 所示。

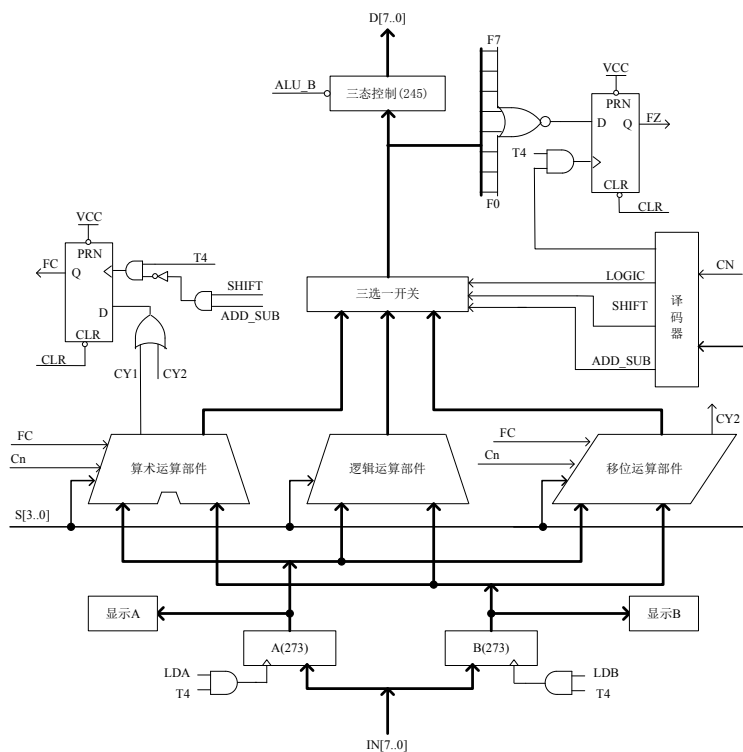


图 3-2 ALU 内部原理图

运算器内部含有三个独立运算部件，分别为算术、逻辑和移位运算部件，要处理的数据存于暂存器 A 和暂存器 B。其中暂存器 A 和暂存器 B 中的数据能在 LED 灯上实时显示，但图中有两部分不在 CPLD 中实现，而是在外围电路中实现，这两部分为图中的‘显示 A’和‘显示 B’。本实验箱上所有的 LED 灯均为正逻辑，即‘1’时亮，‘0’时灭，原理如图 3-3 所示（以 A0 为例，其它相同）。本单元中的进位标志 FC 和零标志 FZ 显示原理也是如此。

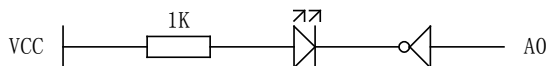


图 3-3 A0 显示原理图

ALU 和 REG（右口）的连接如图 3-4 所示。

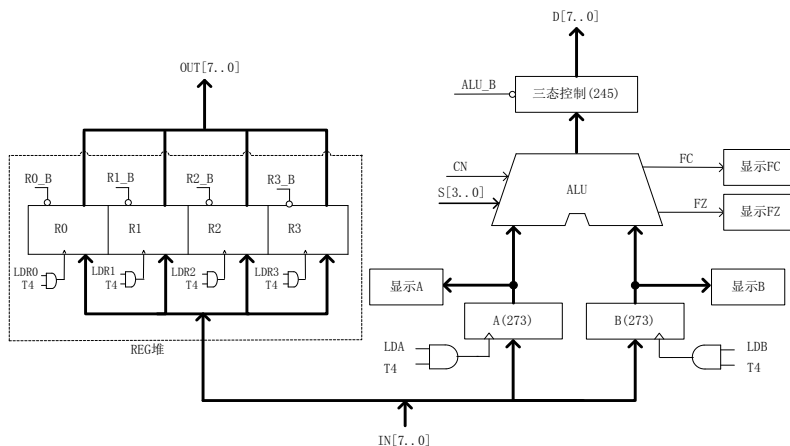


图 3-4 ALU 和 REG（右口）连接原理图

ALU® 单元由运算器和双端口寄存器堆构成，双端口寄存器堆由 R0、R1、R2、R3 组成，它们用来保存操作数及中间运算结果等，其中 R2 还兼做变址寄存器，R3 兼做堆栈指针。

SEL0 和 SEL1 用于选择运算器和寄存器堆的通路：

(1) 当 SEL1=0、SEL0=0，运算器和寄存器堆的结构如图 3-5 所示，寄存器堆只能从右口进行操作，相当于只有一组控制线的单端口寄存器堆。

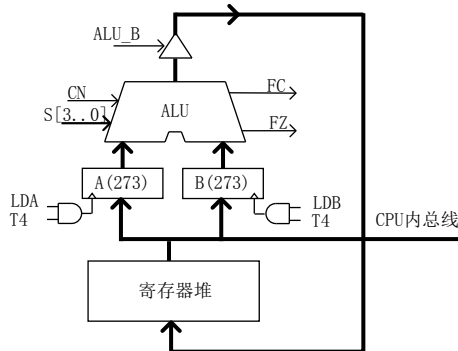


图 3-5 单通道单端口的运算器和寄存器堆的结构

(2)当 SEL1=1、SEL0=0，运算器和双端口寄存器堆的结构如图 3-6 所示，寄存器堆由两组控制信号来分别进行控制，每组控制信号都可以相对独立的对寄存器堆进行读写操作，同时增加了执行专用通道 A 总线，以利于提高指令执行的效率。

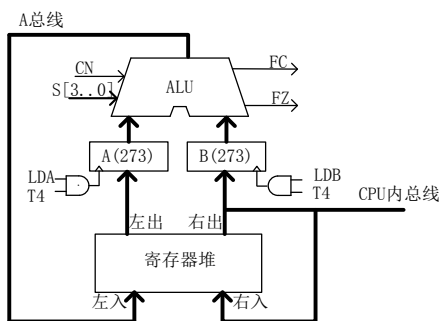


图 3-6 双通道双端口的运算器和寄存器堆的结构

(3)当 SEL1=1、SEL0=1 时，运算器和双端口寄存器堆的结构如图 3-7 所示，在双通道双端口运算器和寄存器堆的基础上增加了暂存器旁路，把运算结果写回到寄存器堆的同时也可以写到暂存器 A、暂存器 B 中。

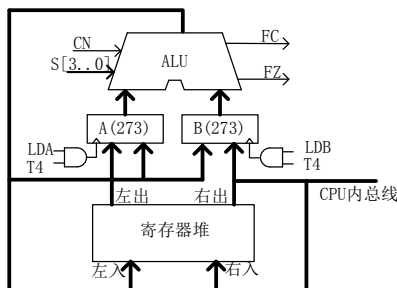



图 3-7 增加暂存器旁路的双通道双端口的运算器和寄存器堆的结构

ALU 与 REG 堆（右口）的输入以排针形式引出 IN7...IN0，运算器的控制信号（CN、S3、S2、S1、S0、LDA、LDB、ALU_B）也以排针形式引出，ALU 的输出、REG 堆（右口）的输出已连接到 CPU 内总线，寄存器堆的左口控制信号（LLDR0、LLDR1、LLDR2、LLDR3）、寄存器堆的右口控制信号（LDR0、LDR1、LDR2、LDR3）、寄存器堆的左口输出控制信号（LR0_B、LR1_B、LR2_B、LR3_B）、寄存器堆的右口输出控制信号（R0_B、R1_B、R2_B、R3_B）、运算器和寄存器堆的通路选择信号（SEL1、SEL0）、进位标志 FC 和零标志 FZ 都已连接到控制器单元的控制信号上。

请注意：实验箱上凡丝印标注有马蹄形标记 ‘’，表示这两根排针之间是连通的。另外，该单元的 JP1 跳线 1、2 脚相接时，表示系统的 T4 节拍接入 ALU® 单元，2、3 脚相接时，表示系统的 T4 节拍未接入 ALU® 单元。图中除了上述提到的信号，实验箱中所有部件单元的 CLR 都连接至 CON 单元的 CLR 按钮。

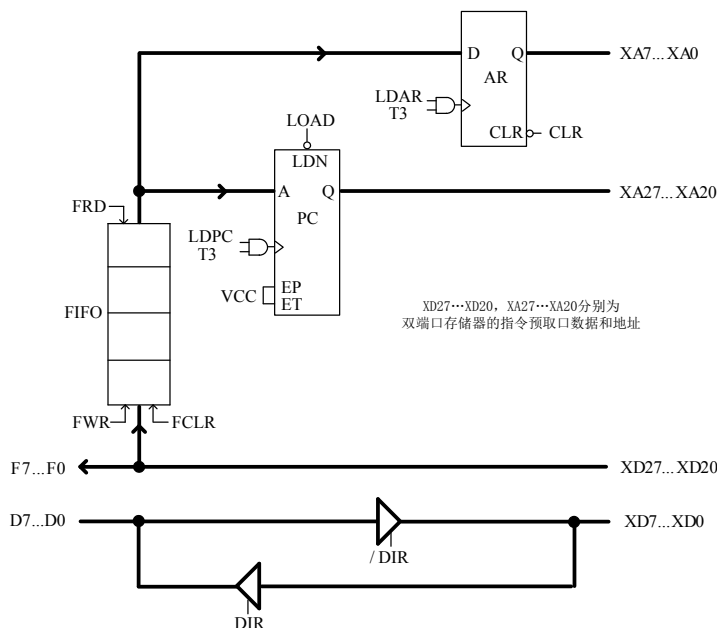


图 3-8-3 支持超标量的总线接口

3. 控制器单元

本系统的控制器单元包含三部分：微程序控制器、指令寄存器及译码、基于 FPGA 的硬布线控制器。

(1) 微程序控制器

微程序控制器包含微程序存储器、微命令寄存器、后续微地址寄存器、译码电路、指令预取控制部件。这里的微程序存储器具有本地编程校验和联机编程校验功能，且具备掉电保护特性。

本地编程由 SD15...SD10 给出微程序地址，IN 单元 D7...D0 给出 8 位微代码。通过编程开关的不同状态及由 T2 引入的节拍脉冲来进行低、中、高共 24 位微代码的编程、校验、运行。

图 3-9 是核心微程序控制器电路原理图：

· 微地址显示灯（MA5...MA0）显示的是后续微地址，而 24 位显示灯（M23...M0）显示的是后续地址的二进制控制位。

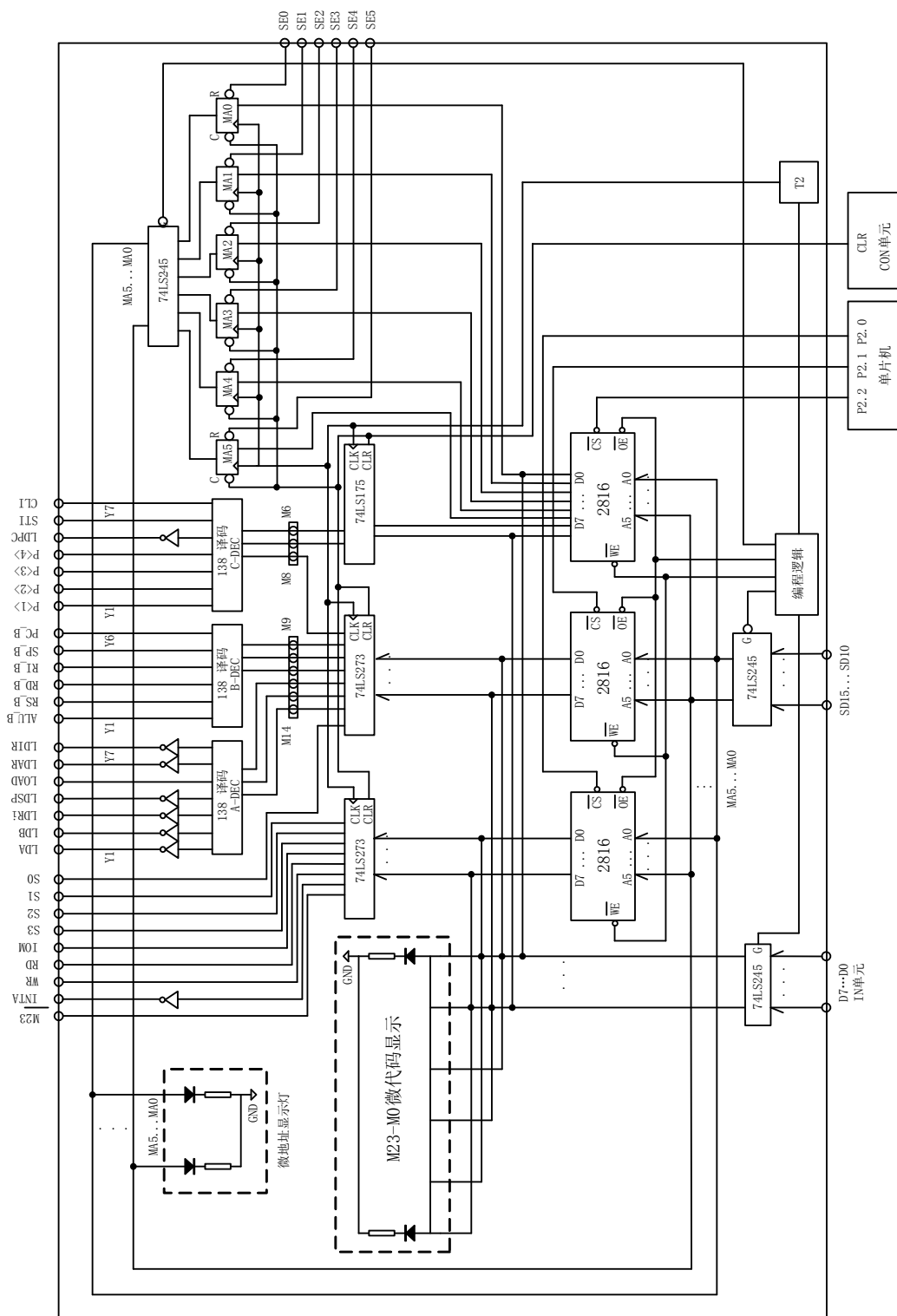
· T2 为微地址锁存器和微命令锁存器的时钟信号。

· 片选信号（MC2、MC1、MC0）在手动状态下一直为‘0’，而在和 PC 联机状态下，受 SYS 单元的单片机来控制。

联机编程通过联机软件来进行 24 位微代码的编程、校验、运行。

微程序控器主要完成接收机器指令译码器送来的代码，使控制转向相应机器指令对应的首条微代码程序，对该条机器指令的功能进行解释或执行的工作。更具体讲，就是通过接收 CPU 指令译码器发来的信号，找到本条机器指令对应的首条微代码的微地址入口，再通过由 T2 引入的时序节拍脉冲的控制，逐条读出微代码。实验板上的微控器单元中的 24 位显示灯（M23—M0）

显示的状态即为读出的微指令。然后，其中几位再经过译码，一并产生实验板所需的控制信号，将它们加到数据通路中相应的控制位，可对该条机器指令的功能进行解释和执行。指令解释到最后，再继续接收下一条机器指令代码并使控制转到对应的微地址入口，这样周而复始，即可实现机器指令程序的顺序、分支、循环运行。所以，有效地定义 24 位微代码对系统的设计至关重要。



为了能够实现基于微程序控制器的机器指令预取功能，这里的微控器还在上面电路的基础上增加了“指令预取控制部件”，使指令预取与指令执行的工作可以重叠进行。

“指令预取控制部件”包含两部分：先进先出指令缓冲栈 FIFO 的控制信号（图 3-10-2）、外部读写控制信号（图 3-10-3）。

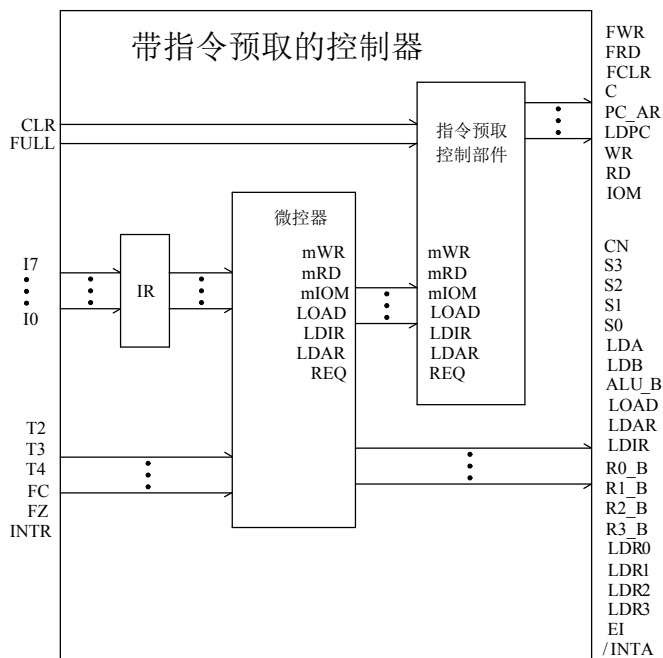


图 3-10-1 带指令预取的微控制器原理图

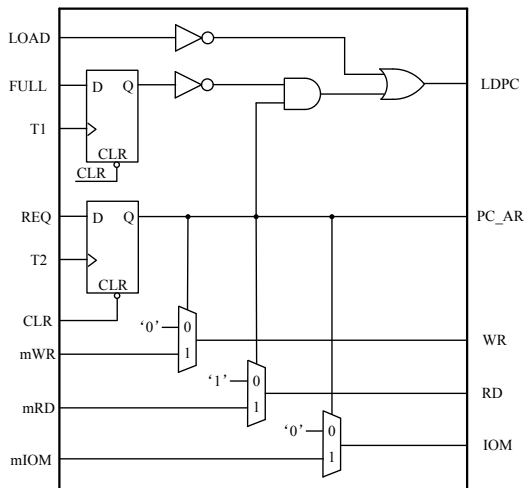
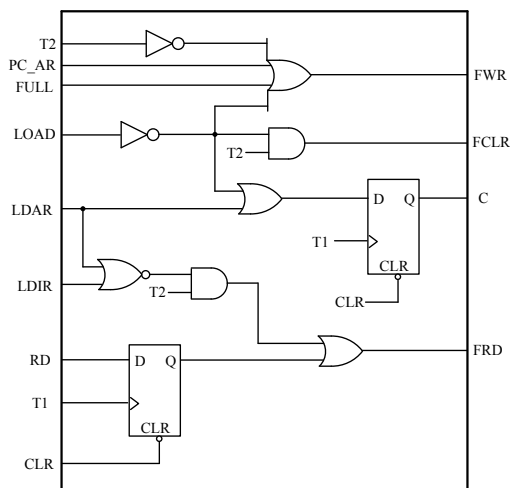


图 3-10-2 指令预取控制部件的 FIFO 控制原理图

图 3-10-3 指令预取控制部件的读写控制原理图

(2) 指令寄存器及译码

CLR 为清零信号的引出端，实验板中已接至 CON 单元中最右边的 CLR 按钮上，此二进制按钮为 CLR 专用。SE5-SE0 端挂接到 CPU 的指令译码器（图 3-12-1）的输出端，通过译码器确定相应机器指令的微代码入口。

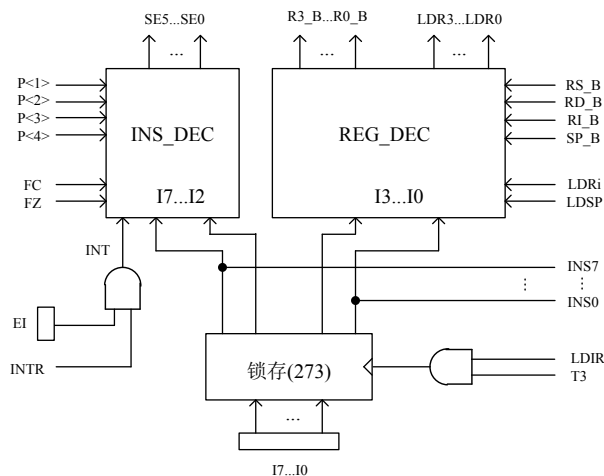


图 3-11 机器指令、寄存器译码原理图

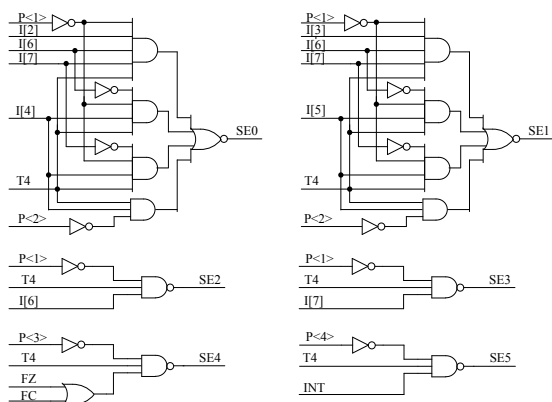


图 3-12-1 INS_DEC 原理图

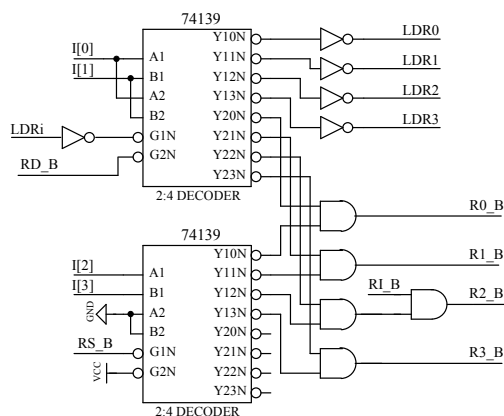


图 3-12-2 REG_DEC 原理图

(3) 基于 FPGA 的硬布线控制器

硬布线控制器本质上是一种由门电路和触发器构成的复杂树形网络，它将输入逻辑信号转换成一组输出逻辑信号，即控制信号。硬布线控制器的输入信号有：指令寄存器的输出、时序信号和运算结果标志状态信号等，输出的就是所有各部件需要的各种微操作信号。

硬布线控制器的设计思想是：在硬布线控制器中，操作控制器发出的各种控制信号是时间因素和空间因素的函数。各个操作定时的控制构成了操作控制信号的时间特征，而各种不同部件的操作所需要的不同操作信号则构成了操作控制信号的空间特征。硬布线控制器就是把时间信号和操作信号组合，产生具有定时特点的控制信号。

(4) 微程序控制器和硬布线控制器的关系

本控制器出厂默认配置是微程序控制器，硬布线控制器是在控制器单元中的 FPGA 设计实现的，可通过 C_JTAG 接口进行下载调试。控制器单元中的 FPGA 与微程序控制器的部分信号复用，复用信号参考附录 2。

4. 时序与操作台单元

时序单元可以提供单脉冲或连续的时钟信号：KK 和 Φ 。其中的 Q 为 555 构成的多谐振荡器的输出，其原理如附图 3-13 所示，经分频器分频后输出频率大约为 30Hz、300Hz、占空比为 50% 的 Φ 信号。

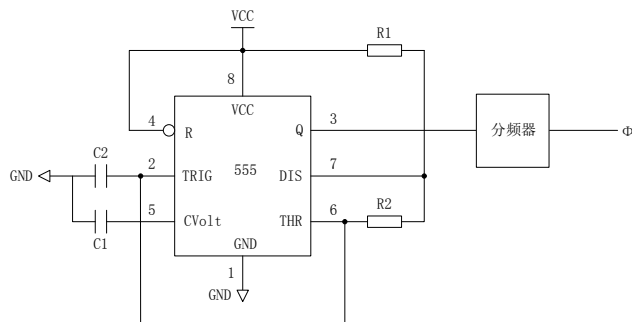


图 3-13 555 多谐振荡器原理图

时序与操作台单元的“MODE”短路块短路，系统工作在四节拍模式；“MODE”短路块拨开，系统工作在三节拍模式。

时序与操作台单元的“SPK”短路块短路，系统具有总线竞争报警功能；“SPK”短路块拨开，系统无报警功能。

每按动一次 KK 按钮，在 KK+ 和 KK- 端将分别输出一个上升沿和下降沿单脉冲。其原理如图 3-14 所示：

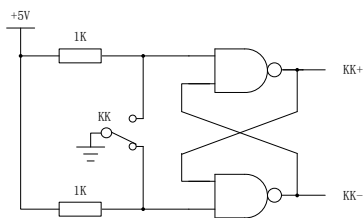


图 3-14 KK 单脉冲电路原理图

每按动一次 ST 按钮，根据时序开关档位的不同，在 TS1、TS2、TS3、TS4 端输出不同的波形。当开关处于‘连续’档时，TS1、TS2、TS3、TS4 输出的是图 3-15 所示的连续时序。开关处于‘单步’档时，TS1、TS2、TS3、TS4 只输出一个 CPU 周期的波形，如图 3-16。开关处于‘单拍’档时，TS1、TS2、TS3、TS4 交替出现，如图 3-17 所示。

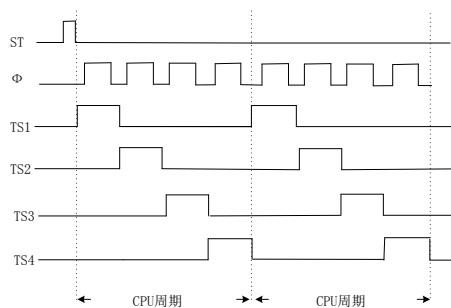


图 3-15 连续时序

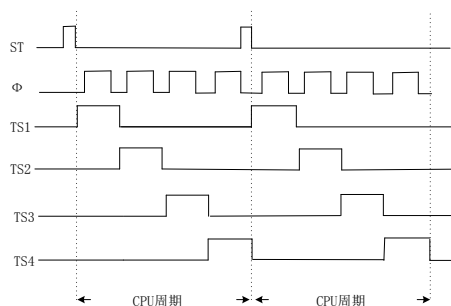


图 3-16 单步时序

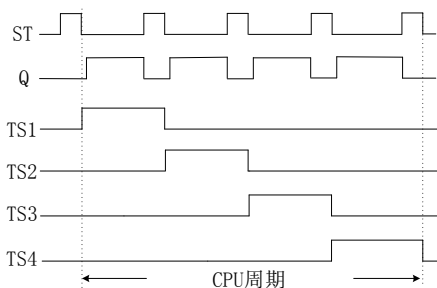
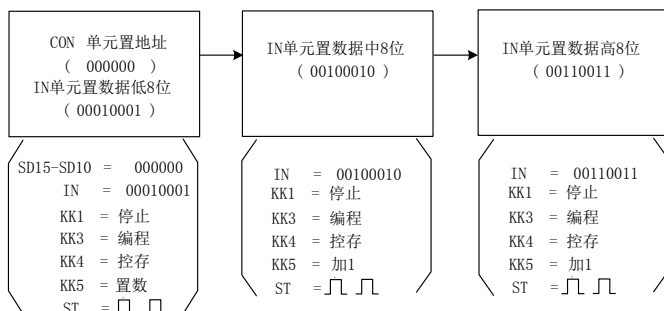


图 3-17 单拍时序

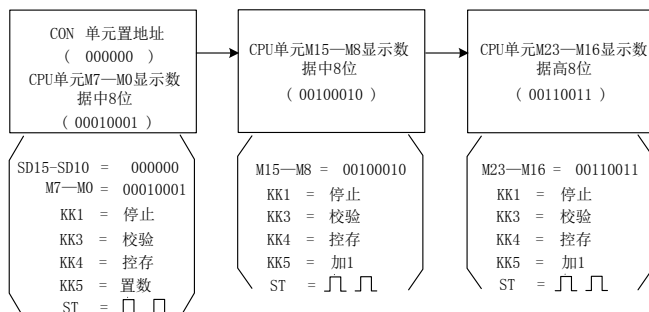
当 TS1、TS2、TS3、TS4 输出连续波形时，有四种方法可以停止输出：将时序状态开关 KK1 拨至停止挡、将 KK2 打到‘单拍’或‘单步’档、按动 CON 单元的 CLR 按钮或是系统单元的复位按钮。CON 单元的 CLR 按钮和 SYS 单元的复位按钮的区别是，CLR 按钮完成对各实验单元清零，复位按钮完成对系统及时序发生器复位。

在实验平台中设有一组编程控制开关 KK1、KK2、KK3、KK4、KK5（位于时序与操作台单元），可实现对存储器（包括程序存储器和控制存储器）的三种操作：编程、校验、运行。考虑到对于存储器（包括程序存储器和控制存储器）的操作大多集中在一个地址连续的存储空间中，实验平台提供了便利的手动操作方式。以向 00H 单元中写入 332211 为例，对于 2816 进行编辑的具体操作步骤如下：首先将 KK1 拨至‘停止’档、KK3 拨至‘编程’档、KK4 拨至‘控存’档、KK5 拨至‘置数’档，由 CON 单元的 SD15——SD10 开关给出需要编辑的控存单元首地址（000000），IN 单元开关给出该控存单元数据的低 8 位（00010001），连续两次按动时序

与操作台单元的开关 ST（第一次按动后控制器单元低 8 位显示该单元以前存储的数据，第二次按动后显示当前改动的数据），此时控制器单元的指示灯 MA5——MA0 显示当前地址（000000），M7——M0 显示当前数据（00010001）。然后将 KK5 拨至‘加 1’档，IN 单元开关给出该控存单元数据的中 8 位（00100010），连续两次按动开关 ST，完成对该控存单元中 8 位数据的修改，此时 CPU 单元的指示灯 MA5——MA0 显示当前地址（000000），M15——M8 显示当前数据（00100010）；再由 IN 单元开关给出该控存单元数据的高 8 位（00110011），连续两次按动开关 ST，完成对该控存单元高 8 位数据的修改，此时 CPU 单元的指示灯 MA5——MA0 显示当前地址（000000），M23——M16 显示当前数据（00110011）。此时被编辑的控存单元地址会自动加 1（01H），由 IN 单元开关依次给出该控存单元数据的低 8 位、中 8 位和高 8 位配合每次开关 ST 的两次按动，即可完成对后续单元的编辑。



编辑完成后需进行校验，以确保编辑的正确。以校验 00H 单元为例，对于 2816 进行校验的具体操作步骤如下：首先将 KK1 拨至‘停止’档、KK3 拨至‘校验’档、KK4 拨至‘控存’档、KK5 拨至‘置数’档。由 CON 单元的 SD15——SD10 开关给出需要校验的控存单元地址（000000），连续两次按动开关 ST，CPU 单元指示灯 M7——M0 显示该单元低 8 位数据（00010001）；KK5 拨至‘加 1’档，再连续两次按动开关 ST，CPU 单元指示灯 M15——M8 显示该单元中 8 位数据（00100010）；再连续两次按动开关 ST，CPU 单元指示灯 M23——M16 显示该单元高 8 位数据（00110011）。再连续两次按动开关 ST，地址加 1，CPU 单元指示灯 M7——M0 显示 01H 单元低 8 位数据。如校验的微指令出错，则返回输入操作，修改该单元的数据后再进行校验，直至确认输入的微代码全部准确无误为止，完成对微指令的输入。



同样的，操作控制开关 KK1、KK2、KK3、KK4、KK5（KK4 拨至‘主存’档），可实现对存储器的操作。

5. 输入设备单元 (IN 单元)

此单元使用 8 个拨动开关作为输入设备, 其电路原理如图 3-18 所示, 左边表示的是 IN 单元的整体连接原理, 右边表示的是一个拨动开关的连接原理, 拨动开关采用的是双刀双掷开关, 一刀用来输出数据, 一刀用来在 LED 灯上显示开关状态。

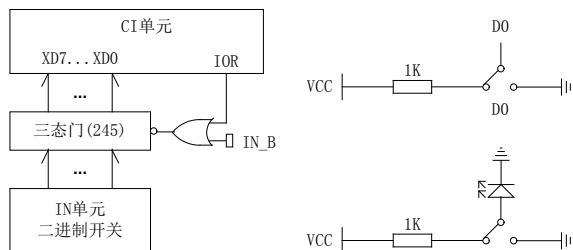


图 3-18 IN 单元原理图

6. 输出设备单元 (OUT 单元)

在 OUT 单元, 数据由 FPGA 在内部通过 74LS273 进行锁存, 并进行显示译码, 形成数码管显示的驱动信号, 具体电路原理如图 3-19 所示。

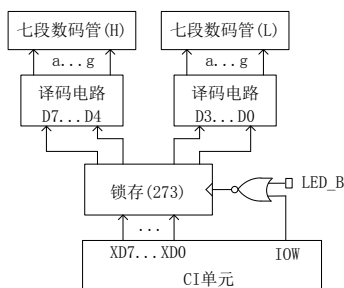


图 3-19 OUT 单元原理图

7. 系统总线单元

此单元由三部构成: 控制总线、数据总线和地址总线。其中控制总线包含 CPU 对存储器 and IO 进行读写时的读写译码电路 (如图 3-20 所示)、外部中断请求电路 (如图 3-21 所示)、中断请求指示灯 INTR、CPU 中断使能指示灯 EI, 还有 CPU 各部件正常工作所需要的时序信号 T1~T4, 已连接至对应部件单元。

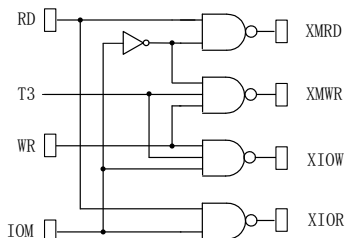


图 3-20 读写译码原理

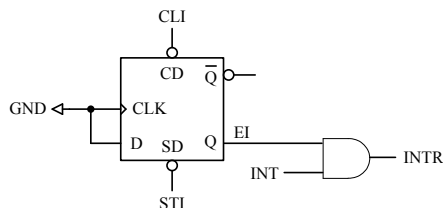


图 3-21 中断请求原理图

数据总线是 CPU 和主存以及外设之间数据交换的通道，其包含两排 8 线排针，排针的相应位已和 CPU 内总线连通。

地址总线由两排 8 线排针，I/O 地址译码芯片 74LS139，地址指示灯组成。两排 8 线排针已连通，为了选择 I/O，产生 I/O 片选信号，还需要进行 I/O 地址译码，系统的 I/O 地址译码原理见图 3-22 所示。

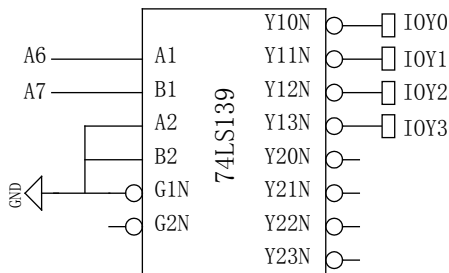


图 3-22 I/O 地址译码原理图

由于用的是地址总线的高两位进行译码，I/O 地址空间被分为四个区，如表 3-1 所示。

表 3-1 I/O 地址空间分配

A7 A6	选定	地址空间
00	IOY0	00-3F
01	IOY1	40-7F
10	IOY2	80-BF
11	IOY3	C0-FF

8. 存储器单元 (MEM 单元)

存储器单元包括一片 IDT7130 双端口存储器（内嵌于 FPGA 中）如图 3-23 所示。该存储器可以由系统对其进行本地或联机读写操作。

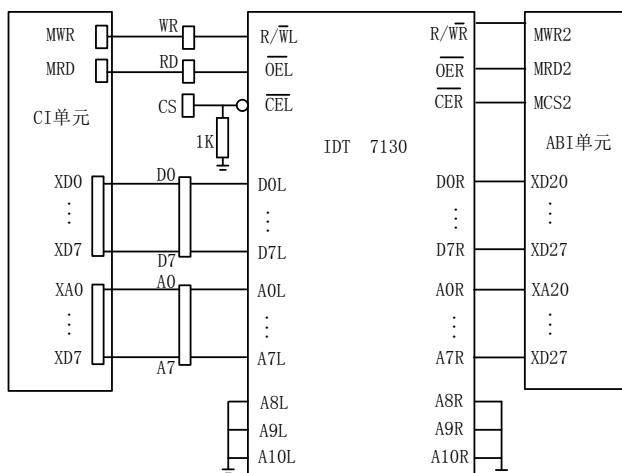


图 3-23 存储器原理图

9. 8253 单元

8253 单元是内嵌于 FPGA 内部的，数据线、地址线、IO 读写通过 Chipset BUS 连接到 CI 单元通过排针引出，其余信号线在 8253 单元以排针引出，其中 GATE0 接到高电平。如图 3-24 所示。

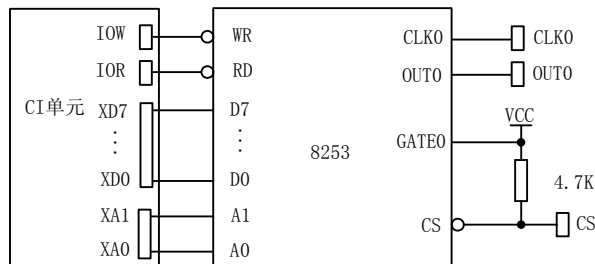


图 3-24 8253 连接图

10. 8259 单元

8259 单元是内嵌于 FPGA 内部的，数据线、地址线、IO 读写通过 Chipset BUS 连接到 CI 单元通过排针引出，其余信号线在 8259 单元以排针引出。如图 3-25 所示。

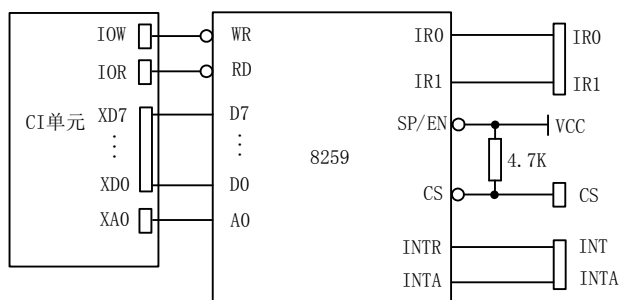


图 3-25 8259 连接图

11. 8237 单元

8237 单元是内嵌于 FPGA 内部的，数据线、地址线、IO/存储器读写通过 Chipset BUS 连接到 CI 单元通过排针引出，其余信号线在 8237 单元以排针引出。如图 3-26 所示。

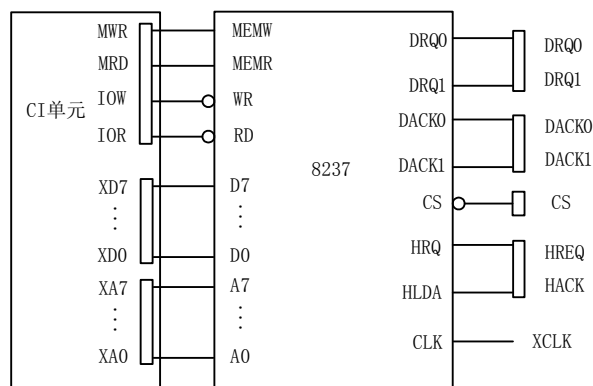


图 3-26 8237 连接图

12. 控制台开关单元 (CON 单元)

此单元包含一个系统总清按钮 CLR 和 24 个双刀双掷开关，开关分成三组，分别为：SD27-SD20、SD17-SD10 和 SD07-SD00。有部分开关有双重丝印，为的是方便接线，一个开关可能对应两个排针，根据丝印就能找到开关和排针的对应关系。开关为双刀双掷，一刀用来提供数据，一刀用来显示开关值，其原理如图 3-27 所示（以 SD20 为例，其它相同）。

CLR 按钮连接如图 3-28 所示，平时为高，按下后 CLR 输出变为低，为系统部件提供清零信号，按下 CLR 按钮后会清零的部件有：程序计数器 PC、地址寄存器 AR、暂存器 A、暂存器 B、指令寄存器 IR、微地址寄存器 MAR 等。

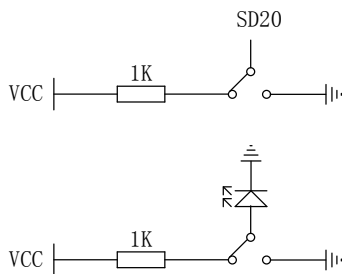


图 3-27 双刀双掷开关原理图

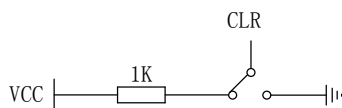


图 3-28 CLR 按钮原理图

13. 扩展单元

扩展单元由一片 Intel 公司的先进 FPGA（资料参考光盘芯片手册）、E_JTAG 下载座、及两组 16 只 LED 显示灯构成。该 FPGA 有 144 个引脚，开放的 E_JTAG 下载座支持下载 SOF 文件，掉电后还原出厂。LED 灯可在调试时观测数据，为正逻辑，1 时亮，0 时灭。

扩展单元排针的丝印分为两部分，一是连线标号，以 H、U、X、Y、Z 打头，如 H0，一是芯片引脚号，是纯数字，如 2，它们表示的是同一个引脚。在 Quartus 软件中分配 I/O 时用的是引脚号，而在实验接线图中，都以连线标号来描述。本单元引出了 39 个 IO 引脚，供实验使用，其中 5 个引脚为纯输入引脚，参考附录 2。

14. 逻辑测量单元

此单元包含四路逻辑示波器 CH3-CH0，四路示波器的电路一样，如图 3-31（以 CH0 为例）。逻辑示波器启动后，通过四路探笔可以测得被测点逻辑波形，并将采样所得数据通过 USB 口发送到 PC 机，PC 机再根据收到的数据，在屏幕上绘制波形。

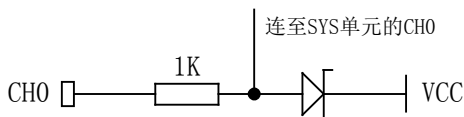


图 3-31 CH0 采样电路

15. 系统单元 (SYS 单元)

此单元是为了和 PC 联机而设计，其原理是通过单片机的串口和 PC 机的 USB 口相连，PC 以命令形式和单片机进行交互，当单片机接收到某命令后，产生相应的时序，实现指定操作。SYS 单元还安排了一个检测电路，当总线上数据发生竞争时，蜂鸣器会发出‘嘀’警报声。SYS 单元还有一个重要职责：当 ST 按钮按下时会对单片机的 INT1 产生一个中断请求，此时单片机根据时序单元状态开关的档位，产生相应的时序。

3.4 注意事项

- (1) 使用前后均应仔细检查主机板，防止导线、元件等物品落入装置内导致线路短路、元件损坏。
- (2) 电源线应放置在机内专用线槽中。
- (3) 注意系统的日常维护，经常清理灰尘和杂物。
- (4) 电源关闭后，不能立即重新开启，关闭与重新开启之间至少应有 30 秒间隔。

第 4 章 TDX-CMX 系统集成操作软件

4.1 与 PC 联机说明

TDX-CMX 计算机实验系统安装有一个标准的 USB 接口 (Type B to A) 插座, 使用随机配套的 USB 电缆分别插在 TDX-CMX 及 PC 微机的 USB 口, 即可实现系统与 PC 的联机操作。

本系统软件是通过 PC 机 USB 口向 TDX-CMX 上的 SYS 单元发送指令, 从而使用单片机直接对程序存储器、微程序控制器进行读写, 并可实现单步微程序, 单步机器指令和程序连续运行等操作。

4.2 软件操作说明

(一) 界面窗口介绍

主界面如图 4-1 所示, 由指令区、输出区和图形区三部分组成。

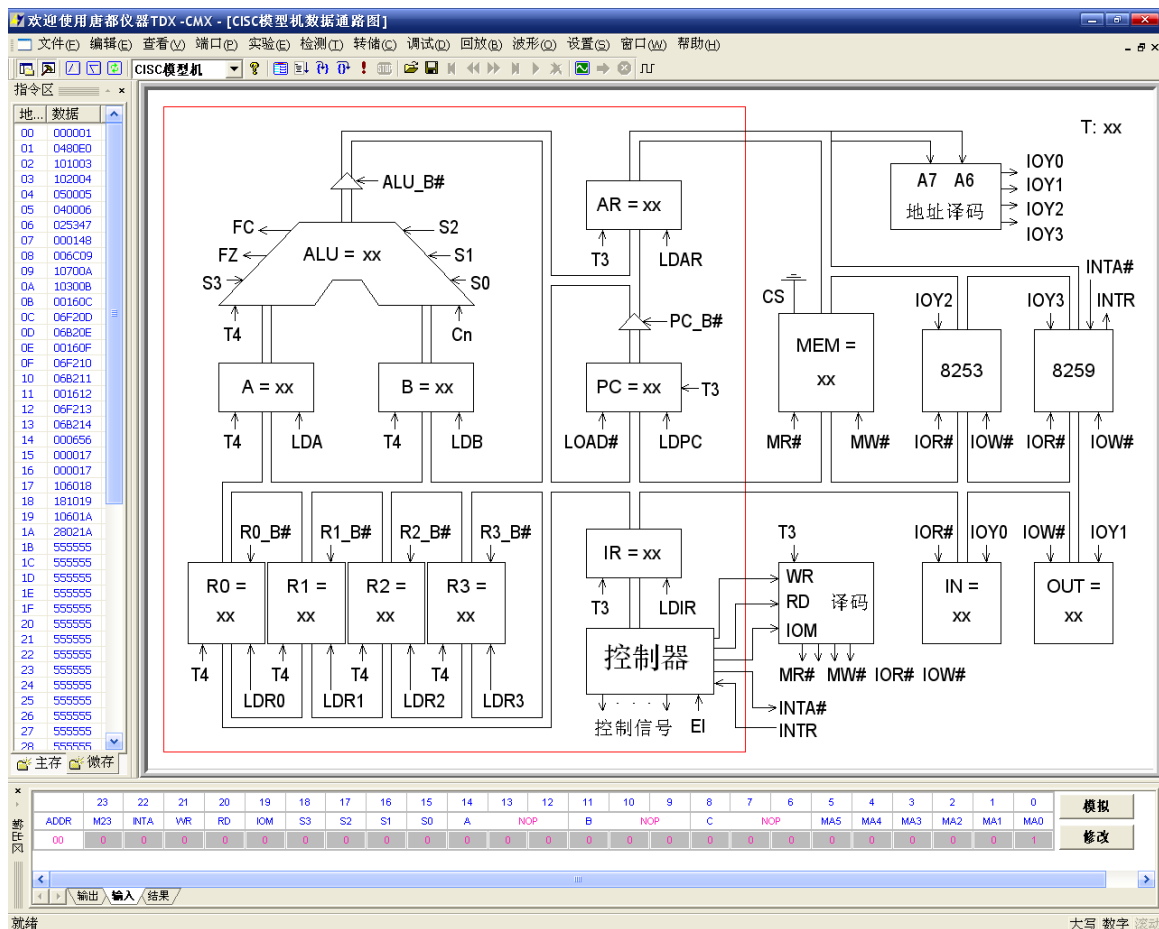


图 4-1 软件主界面

指令区：

分为机器指令区和微指令区，指令区下方有两个 Tab 按钮，可通过按钮在两者之间切换。

机器指令区：分为两列，第一列为主存地址（00—FF，共 256 个单元），第二列为每个地址所对应的数值。USB 口通讯正常且 USB 口无其它操作，可以直接修改指定单元的内容，用鼠标单击要修改单元的数据，此时单元格会变成一个编辑框，即可输入数据，编辑框只接收两位合法的 16 进制数，按回车键确认，或用鼠标点击别的区域，即可完成修改工作。按下 ESC 键可取消修改，编辑框会自动消失，恢复显示原来的值，也可以通过上下方向键移动编辑框。

微指令区：分为两列，第一列为微控器地址（00—3F，共 64 个单元），第二列为每个地址所对应的微指令，共 6 字节。修改微指令操作和修改机器指令一样，只不过微指令是 6 位，而机器指令是 2 位。

输出区：

输出区由输出页、输入页和结果页组成。

输出页：在数据通路图打开，且该通路中用到微程序控制器，运行程序时，输出区用来实时显示当前正在执行的微指令和下条将要执行的微指令的 24 位微码及其微地址。当前正在执行微指令的显示可通过菜单命令“【设置】—【当前微指令】”进行开关。

输入页：可以对微指令进行按位输入及模拟，鼠标左键单击 ADDR 值，此时单元格会变成一个编辑框，即可输入微地址，输入完毕后回车，编辑框消失，后面的 24 位代表当前地址的 24 位微码，微码值用红色显示，鼠标左键单击微码值可使该值在 0 和 1 之间切换。在数据通路图打开时，按动‘模拟’按钮，可以在数据通路中模拟该微指令的功能，按动‘修改’按钮则可以将当前显示的微码值下载到下位机。

结果页：用来显示一些提示信息或错误信息，保存和装载程序时会在这区域显示一些提示信息。在系统检测时，也会在这区域显示检测状态和检测结果。

图形区：

可以在此区域编辑指令，显示各个实验的数据通路图、示波器界面等。

（二）菜单功能介绍

1. 文件菜单项

文件菜单提供了以下命令：

① 新建 (N)：

在 CMX 中建立一个新文档。在文件新建对话框中选择您所建立的新文件的类型。

② 打开 (O)

在一个新的窗口中打开一个现存的文档。您可同时打开多个文档。您可用窗口菜单在多个打开的文档中切换。

③ 关闭 (C)

关闭包含活动文档的所有窗口。CMX 会建议您在关闭文档之前保存对您的文档所做的改动。如果您没有保存而关闭了一个文

新建 (N)	Ctrl+N
打开 (O)...	Ctrl+O
关闭 (C)	
保存 (S)	Ctrl+S
另存为 (A)...	
打印 (P)...	Ctrl+P
打印预览 (V)	
打印设置 (R)...	
最近文件	
退出 (X)	

档，您将会失去自从您最后一次保存以来所做的所有改动。在关闭一无标题的文档之前，CMX 会显示另存为对话框，建议您命名和保存文档。

④ 保存 (S)

将活动文档保存到它的当前的文件名和目录下。当您第一次保存文档时，CMX 显示另存为对话框以便您命名您的文档。如果在保存之前，您想改变当前文档的文件名和目录，您可选用另存为命令。

⑤ 另存为 (A) ...

保存并命名活动文档。CMX 会显示另存为对话框以便您命名您的文档。

⑥ 打印 (P) ...

打印一个文档。在此命令提供的打印对话框中，您可以指明要打印的页数范围、副本数、目标打印机，以及其它打印机设置选项。

⑦ 打印预览 (V)

按要打印的格式显示活动文档。当您选择此命令时，主窗口就会被一个打印预览窗口所取代。这个窗口可以按它们被打印时的格式显示一页或两页。打印预览工具栏提供选项使您可选择一次查看一页或两页，在文档中前后移动，放大和缩小页面，以及开始一个打印作业。

⑧ 打印设置 (R) ...

选择一台打印机和一个打印机连接。在此命令提供的打印设置对话框中，您可以指定打印机及其连接。

⑨ 最近使用文件

您可以通过此列表，直接打开最近打开过的文件，共四个。

⑩ 退出 (X)

结束 CMX 的运行阶段。您也可使用在应用程序控制菜单上的关闭命令。

2. 编辑菜单项

① 撤销 (U)

撤销上一步编辑操作。

② 剪切 (I)

将当前被选取的数据从文档中删除并放置于剪贴板上。如当前没有数据被选取时，此命令则不可用。

③ 复制 (C)

将被选取的数据复制到剪切板上。如当前无数据被选取时，此命令则不可用。

④ 粘贴 (P)

将剪贴板上内容的一个副本插入到插入点处。如剪贴板是空的，此命令则不可用。

撤销 (U)	Ctrl+Z
剪切 (I)	Ctrl+X
复制 (C)	Ctrl+C
粘贴 (P)	Ctrl+V

3. 查看菜单项

查看菜单提供了以下命令：

① 工具栏 (T)

显示和隐藏工具栏，工具栏包括了 CMX 中一些最普通命令的按钮。当工具栏被显示时，在菜单项目的旁边会出现一个打勾记号。

② 指令区 (W)

<input checked="" type="checkbox"/>	工具栏 (T)
<input checked="" type="checkbox"/>	指令区 (W)
<input checked="" type="checkbox"/>	输出区 (O)
<input checked="" type="checkbox"/>	状态栏 (S)

显示和隐藏指令区，当指令区被显示时，在菜单项目的旁边会出现一个打勾记号。

③ 输出区 (O)

显示和隐藏输出区，当输出区被显示时，在菜单项目的旁边会出现一个打勾记号。

④ 状态栏 (S)

显示和隐藏状态栏。状态栏描述了被选取的菜单项目或被按下的工具栏按钮，以及键盘的锁定状态将要执行的操作。当状态栏被显示时，在菜单项目的旁边会出现一个打勾记号。

4. 端口菜单项

端口菜单提供了以下命令：

① 端口选择…

选择通讯端口，选择该命令时会弹出图 4-2 所示对话框。该命令会自动检测当前系统可用的串口号，并列于组合框中，选择某一串口后，按确定键，对选定串口进行初始化操作，并进行联机测试，报告测试结果，如果联机成功，则会将指令区初始化。

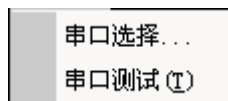


图 4-2 串口选择对话框

② 端口测试 (T)

对当前选择的串口进行联机通讯测试，并报告测试结果，只测一次，如果联机成功，则会将指令区初始化。如串口不能正常初始化，此命令则不可用。

5. 实验菜单项

实验菜单提供了以下命令：

①. 运算器实验

打开运算器实验数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

②. 存储器实验

打开存储器实验数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

③. 微控器实验

打开微控器实验数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

④. 简单模型机实验

打开简单模型机实验数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

运算器实验
存储器实验
微控器实验
简单模型机
ALU®实验
CISC模型机
RISC模型机
指令预取模型机
三级流水模型机
超标量流水模型机

示。

⑤. ALU® 实验

打开 ALU® 实验数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

⑥. CISC 模型机实验

打开 CISC 模型机实验数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

⑦. RISC 模型机实验

打开 RISC 模型机数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

⑧. 指令预取模型机

打开指令预取模型机数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

⑨. 三级流水模型机

打开流水模型机数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

⑩. 超标量流水模型机实验

打开超标量流水模型机实验数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

6. 检测菜单项

检测菜单提供了以下命令：

① 连线检测 (C)

(a) 简单模型机

对简单模型机的连线进行检测，并在‘输出区’的‘结果页’显示相关信息。

(b) 复杂模型机

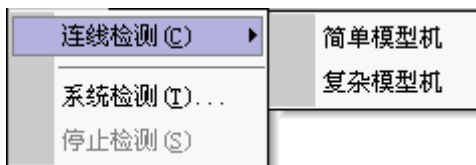
对复杂模型机的连线进行检测，并在‘输出区’的‘结果页’显示相关信息。

② 系统检测 (T) ...

启动系统检测，可以进行部件或是整机检测。

③ 停止检测 (S)

停止系统检测。



7. 转储菜单项

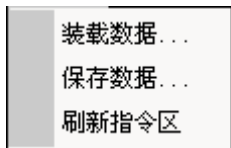
转储菜单提供了以下命令：

① 装载数据...

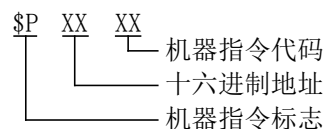
将上位机指定文件中的数据装载到下位机中，您选择该命令会弹出打开文件对话框。

可以打开任意路径下的*.TXT 文件，如果指令文件合法，系统将把这些指令装载到下位机中，装载指令时，系统提供了一定的检错功能，如果指令文件中有错误的指令，将会导致系统退出装载，并提示错误的指令行。

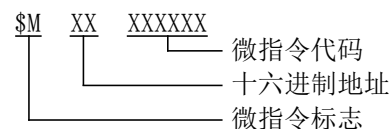
指令文件中指令书写格式如下：



机器指令格式说明:



微指令格式说明:



例如机器指令\$P00FF, “\$”为标记号, “P”代表机器指令, “00”为机器指令的地址, “FF”为该地址中的数据。微指令\$M00AA77FF, “\$”为标记号, “M”代表微指令, “00”为机器指令的地址, “AA77FF”为该地址中的数据。

② 保存数据…

将下位机中(主存, 微控器)的数据保存到上位机中, 选择该命令会弹出一个保存对话框, 如图 4-3 所示:

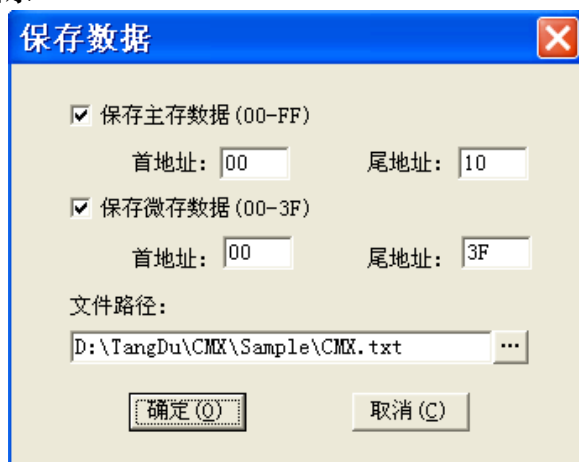


图 4-3 保存数据对话框

可以选择保存机器指令, 此时首尾地址输入框将会变亮, 否则首尾地址输入框将会变灰, 在允许输入的情况下您可以指定需要保存的首尾地址, 微指令也是如此, 数据到保存指定路径的*.TXT 格式文件中。

③ 刷新指令区

从下位读取所有机器指令和微指令, 并在指令区显示。

8. 调试菜单项

调试菜单提供了以下命令:

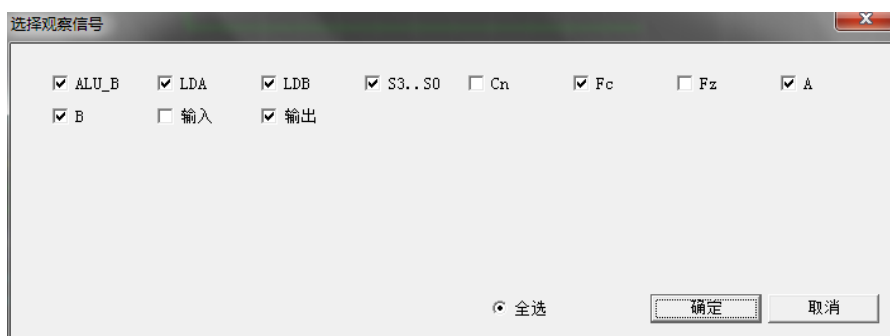
① 微程序流程图…

当微控器实验、简单模型机和综合性实验中任一数据通路图打开时, 可用此命令来打开指定的微程序流程图, 选择该命令会弹出打开文件对话框。

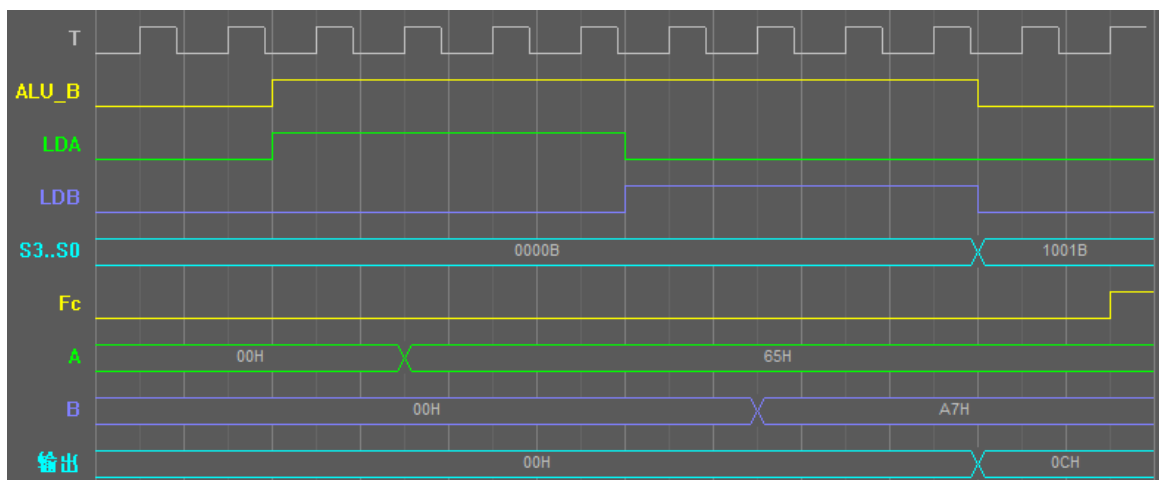
② 时序观测窗

当运算器、存储器、简单模型机和综合性实验中任一数据通路图打开时, 可用此命令来打开指定的时序观测窗。打开后弹出“选择观察信号”, 勾选需要观察的信号, 点击确定。

微程序流程图…
时序观测窗
单节拍
单周期
单机器指令
连续运行
停止运行



参考实验步骤单拍或者单步调试，可观察到时序图。



③ 单节拍

向下位机发送单节拍命令，下位机完成一个节拍的工作。

④ 单周期

向下位机发送单周期命令，下位机完成一个机器周期的工作。

⑤ 单步机器指令

向下位机发送单步机器指令命令，下位机运行一条机器指令。

⑥ 连续运行

向下位机发送连续运行命令，下位机将会进入连续运行状态。

⑦ 停止运行

如果下位机处于连续运行状态，此命令可以使得下位机停止运行。

9. 回放菜单项

回放菜单提供了以下命令：

① 打开…

打开现存的数据文件。

② 保存…

保存当前的数据到数据文件。

③ 首端

跳转到首页。

④ 向前

向前翻一页。

⑤ 向后

向后翻一页。

⑥ 末端

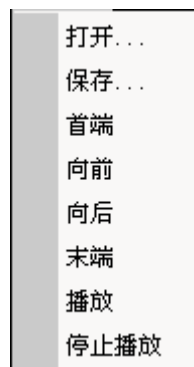
跳转到末页。

⑦ 播放

连续向后翻页。

⑧ 停止播放

停止连续向后翻页。



10. 波形菜单项

波形菜单提供了以下命令：

① 打开 (O)

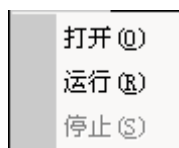
打开示波器窗口。

② 运行 (R)

启动示波器，如果下位机正运行程序则不启动。

③ 停止 (S)

停止处于启动状态的示波器。



11. 设置菜单项

设置菜单提供了以下命令：

① 流动速度 (L) ...

设置数据通路图中数据的流动速度，选择该命令会弹出一个流动速度设置对话框，如图 4-4 所示。拖动滑动块至适当位置，点击‘确定’按钮即完成设置。

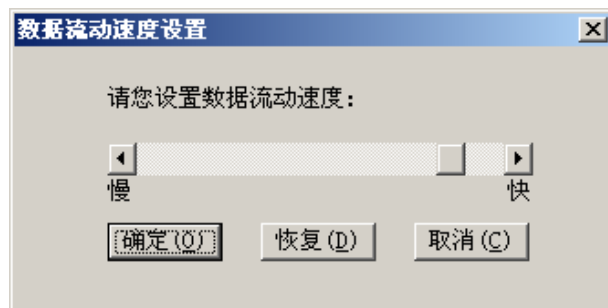
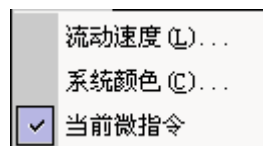


图 4-4 流动速度设置对话框

② 系统颜色 (C) ...

设置数据通路图、微程序流程图和示波器的显示颜色，选择该命令会弹出一个设置对话框，如图 4-5 所示。



图 4-5 系统颜色设置对话框

分为三页，分别为通路图、微流图和示波器，按动每页的 TAB 按钮，可在三页之间切换。选择某项要设置的对象，然后按下‘更改’按钮，或直接用鼠标左键点击要设置对象的颜色框，可弹出颜色选择对话框，选定好颜色后，点击‘应用’按钮相应对象的颜色就会被修改掉。

③ 当前微指令

设置‘输出区’的‘输出页’是否显示当前微指令，当前微指令用灰色显示，并在地址栏标记为‘C’，下条将要执行的微指令标记为‘N’。

12. 窗口菜单项

窗口菜单提供了以下命令。这些命令使您能在应用程序窗口中安排多个文档的多个视图：

① 新建窗口 (N)

打开一个具有与活动的窗口相同内容的新窗口。您可同时打开数个文档窗口以显示文档的不同部分或视图。如果您对一个窗口的内容做了改动，所有其它包含同一文档的窗口也会反映出这些改动。当您打开一个新的窗口，这个新窗口就成了活动的窗口并显示于所有其它打开窗口之上。

② 层叠 (C)

按相互重叠形式来安排多个打开的窗口。

③ 平铺 (T)

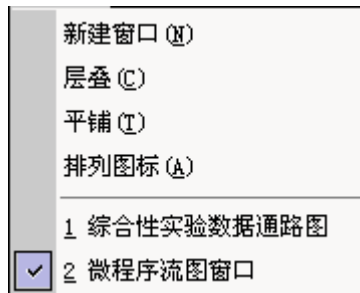
按互不重叠形式来安排多个打开的窗口。

④ 排列图标 (A)

在主窗口的底部安排被最小化的窗口的图标。如果在主窗口的底部有一个打开的窗口，则有可能会看不见某些或全部图标，因为它们在这个文档窗口的下面。

⑤ 窗口选择

CMX 在窗口菜单的底部显示出当前打开的文档窗口的清单。有一个打勾记号出现在活动的窗口的文档名前。从该清单中挑选一个文档可使其窗口成为活动窗口。



13. 帮助菜单项

帮助菜单提供以下的命令，为您提供使用这个应用程序的帮助：

① 关于 (A) CMX...

显示您的 CMX 版本的版权通告和版本号码。

② 实验帮助 (E) ...

显示实验帮助的开场屏幕。从此开场屏幕，您可跳到关于 CMX 所提供实验的参考资料。

③ 软件帮助 (S) ...

显示软件帮助的开场屏幕。从此开场屏幕，您可跳到关于使用 CMX 设备的参考资料。

(三) 工具栏命令按钮介绍：



显示或隐藏指令区。



显示或隐藏输出区。



保存下位机数据。



向下位机装载数据。



刷新指令区数据。



打开实验帮助。



打开微程序流程图。



单节拍运行。



单周期运行。



单机器指令运行。



连续运行。



停止运行。



打开实验数据文件。



保存实验数据。



跳转到首页。



向前翻页。



向后翻页。



跳转到末页。



连续向后翻页。

关于(A)CMX...
实验帮助(E)...
软件帮助(S)...



停止向后翻页。



打开示波器窗口。



启动示波器。



停止示波器。



打开时序观测窗。

第5章 TDX-CMX 系统检测功能说明

为了更好地维护设备，TDX-CMX 提供了两种系统检测功能，一种针对实验接线的检测，一种是针对维护的系统检测。

对接线相对比较复杂的实验——复杂模型机实验，提供了实验接线检测，其余实验要么接线较少，要么是在复杂模型机实验接线基础上增加少量线，所以没有安排连线检测，连线检测结果会在软件‘输出区’的‘结果页’显示。

对于系统检测，提供了两种检测功能，包括基本检测和整机检测。基本检测无须接线就可实现对实验平台的 ALU® 单元、ABI 单元、CPU 单元、MEM 单元的初级检测，基本检测主要用于设备维护性检测；整机检测需如图 5-1 所示连接线路后进行，一般在系统检测发现问题后使用，以进一步确定故障所在。

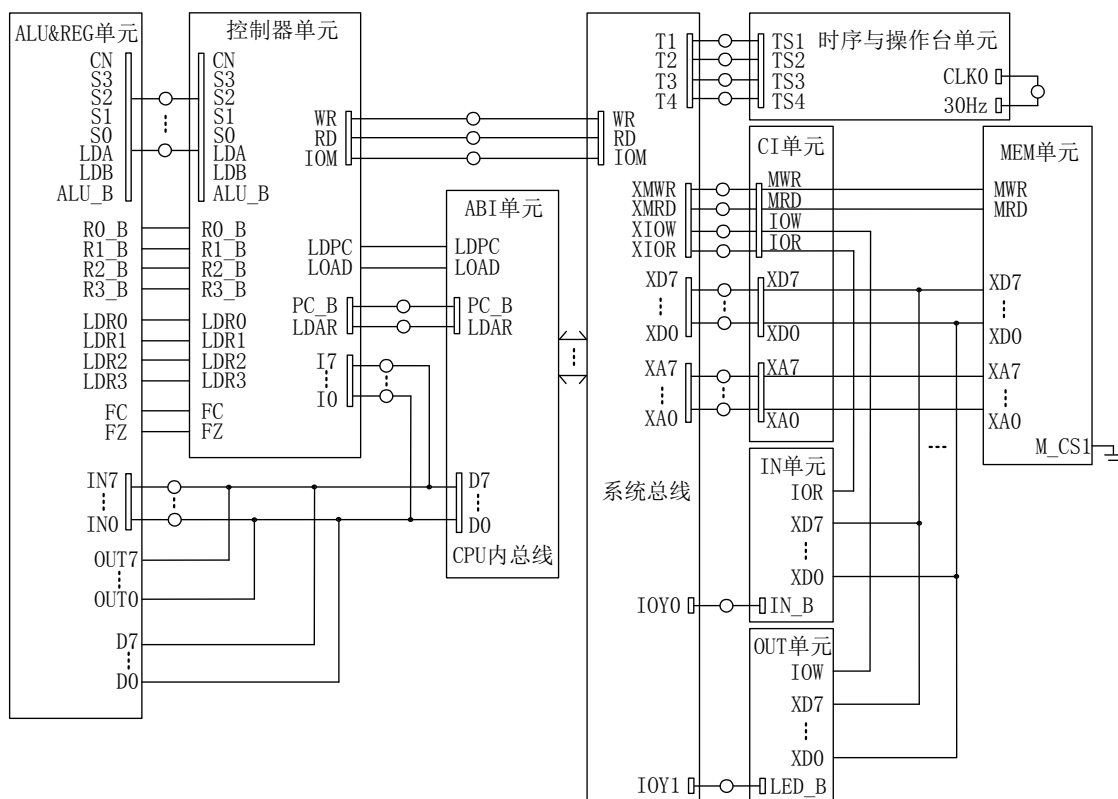


图 5-1 整机检测接线图

第 6 章 TDX-CMX 系统常见故障的分析及处理

1. 与 PC 联机失败

- (1) 首先按复位键，检查单片机复位是否正常。
- (2) 检查 USB 口是否接触良好，(包括 PC 机的 USB 口是否完好,通讯电缆是否完全接触)。
- (3) 在前述(1)、(2)均正常的情况下，请检查+5V 电压是否正常，若正常，请检查实验接线或查看线路板上有无散落的导线或元器件，导致短路。
- (4) 若以上均正常，有可能是 SYS 单元的 89S51 或晶振有问题。

2. 微程序无法写入

- (1) 若手动写入程序时，检查时序与操作台单元的 KK3 开关应处于‘编程’状态。
- (2) 若联机写入程序时，检查单片机复位是否正常。

3. 实验无法通过

这种情况一般出现在一次或几次实验之后，其原因是：

- (1) 实验时芯片损坏，只需更换相应单元的芯片即可。
- (2) 实验接线错误，应仔细检查接线。

如果您在实验中遇到无法解决的问题，请与我公司售后服务部联系。

附录 1 微程序流程图编程方法

程序以行为单位，每一行代表一条指令，并允许有注释，凡“;”后的所有内容都认为是被注释掉的内容。程序中可以有任意多的空格或是空行，即使是同一个数字之间有空格也认为是合法的，如“45”或是“4 5”，认为是等价的，因为在生成图形文件的过程中，这些注释和空格或是空行都将被忽略掉。但是有一点，凡是双引号中的空格将不会被除掉，因为这些可能是您有意添加的。

1. 指令种类说明

(1) “T 指令”。

格式为：T：“列号：字符串”。如：T：“9:复杂模型机微程序流程图”。

说明：“T:”为指令标志，指明该行为标题行，字母 T 必须为大写，双引号中的内容为标题，其中内容又分为两部分，冒号前的数字为标题显示的列号，可以由‘0’打头，如写成‘09’也是合法的。冒号后面的内容即为用来显示的字符串，因而指令行：“T：“9:复杂模型机微程序流程图””的作用就是以第 9 列的中心，在窗口客户区顶部显示标题：复杂模型机微程序流程图。

如果不指明列号，则以窗口总宽度的一半为中心显示标题，如果标题指令中出现多个冒号则只认第一个，如果字符串以冒号打头则认为列号为 0。该指令行必须位于逻辑上的第 1 行，所谓逻辑上的行是指去除注释行和空行以后的行号。

(2) “R 指令”。

格式：R：行数，如：R：020。

说明：“R:”为指令标志，指明该指令行为总行数，字母 R 也必须为大写，其后的数字为总行数，可以以‘0’打头。该指令行必须位于逻辑上的第 2 行，且行数最大不能超过 500。

(3) “C 指令”。

格式：C：列数，如：C：017。

说明：“C:”为指令标志，指明该指令行为总列数，字母 C 也必须为大写，其后的数字为总列数，可以以‘0’打头。该指令行必须位于逻辑上的第 3 行，且列数最大不能超过 500。

(4) “\$ 指令”。

格式：\$：行号，列号，微地址，图片类型，测试字，指令说明，如：\$：04, 09, 02, 1, 80, "RAM->BUS:BUS->IR"。

2. 指令使用说明

其中微地址、测试字和指令说明字段可以为空。

(1) “\$:”为指令标志，指明该指令为一般性指令。

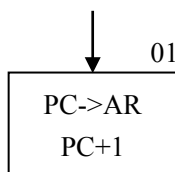
(2) 行号、列号指定了该图形节点所在的位置，不论是哪种图形，其所占有的空间大小是固定的，这样便于一般性处理。

(3) 微地址指明了该节点所对应的微地址，如果该节点图片类型为 1，也就是普通意义上的一条微指令（下位机每执行一次所对应的一个节点），其值是 16 进制的，因而取值范围是 00-3F，共 64 个单位，而如果该节点图片类型为其他类型的话，其取值最好不要在 00-3F 范围之内，以

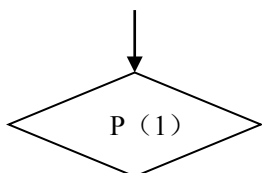
免发生冲突。每个结点的微地址应该是唯一的，如果有重复的情况将会导致错误。如果该字段为空则将其视为-1。

(4) 图片类型指明该节点以什么方式绘图，而且不同的图形类型，其后的微程序说明格式也不一样，因而每种图形的微程序说明格式也在此表述，系统共提供了 9 种图形类型，在您的程序中可以包含其中的任意种。这 9 种图形分别为：

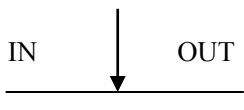
I、图形号为 1，整个图形高度所占用的象素为逻辑上的一行，而整个图形宽度所占用的象素为逻辑上的一列，以后的图形中的行列都以此为标准。图形中箭头所占用的高度为图形总高度的 1/3，矩形中的文字为微程序说明，居中显示，最多可显示 3 行，且图形高度能自动调整一级，但超过 3 行将不显示，微程序说明的书写方法是以“:”来换行，如果一行长度过长将会导致错误，右上角的数字为微地址，这就是用得最多的一种图形，其每个图形和下位机一条微指令相对应，而其它图形都只是做一些辅助性的工作。



II、这是测试字图形，图形号为 2，其行列号同图形 1，菱形中的文字为测试字说明，无论您输入多少行，其只会标出第一行，书写方式也同图形 1，其‘微地址’不加以显示，但要求用户在编程序时加以关联，否则运行时可能达不到预期的效果。

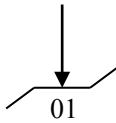


III、这是跟在测试字后的助记符说明，图形号为 3，其行号同图形 1，但其列号是有别的意义的，列号指的是箭头所在列的列号，因为一般的助记符显示位置为该列的正中间，而对于列号所在的列在靠左的位置加以显示，而且在绘画运行轨迹时，也是从列号所在的列绘到当前的列。助记符的书写方法是：在微指令说明字段加以描述，如：8:10:KWE(01):KRD(00):RP(11)，第一个冒号前的数字代表图形的起始列号，第二个冒号前的数字代表图形的结束列号，且应大于起始列号，这两部分必须加以描述清楚。后面用冒号隔开的每个字段代表一个助记符，助记符应按从左到右的次序依次书写，如果中间某项为空，也得用冒号隔开，终止列号和起始列号之差必须和助记符的个数相等。



IV、这是转公共的图形，图形号为 4，行列号和图形 1 相同，微地址一般为空，微指令说

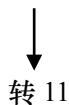
明字段中的内容为转公共的微地址，如下图中显示的 01 即为转公共的微地址。



V、这是每个微程序流图分支的标题，图形号为 5，其行号同图形 1，而列号是指该标题所显示的中心所在的列号，显示颜色同主标题，微地址和测试字一般为空，微指令说明字段中的内容即为要显示的标题。

运行微程序

VI、这是转移图形，图形号为 6，如果遇到有循环之类的图形，可以用转移图形来表示，该图形和图形 4 很相似，指令也类似，下图中的文字描述指的是要转移的地址，其内容来自微指令说明字段中。



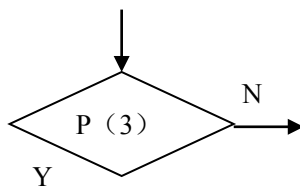
VII、该图形在平时用得不多，图形号为 7，只有在复杂的微程序流图中才可能用到，主要用于填充空白，使整幅图在视觉上变得美观，如复杂模型机微程序流图。其一般用在图形 1 之后，利用图形 1 或其它图形的测试字字段和其建立起关联。



VIII、图形号 8，该图形和上面的一样，只是在较为复杂的微程序流图中才会用到，具体可见复杂模型机微程序流图文件中的范例。



IX、带分支的测试字图形，图形号为 9，该图形和图形 2 很类似，编程方式也是一样的，具体见示例。



您可以在任意的纯文本编辑器中编写微程序流图文件，但一定要以*.MFP 的格式存储，您

也可以在系统中带有的编辑器进行编辑，当您编辑好并存盘后，就可以生成微程序流图了。方法是选择菜单：[运行\微程序流图]命令来解释您编写的微程序流图文件，如果您的程序没有错误，系统就会自动为您生成一个微程序流图，这一过程相当于预览功能，您可以边编写边预览，但是有一点，您必须打开复杂模型机或是重叠模型机数据通路图后才可以启用上述功能。

3. 系统检错类型

如果您的微程序流图文件有错误，不用紧张，系统会为您找出错误，并提供错误的一些信息，如：程序的第 10 行是这样一行代码：\$: 16, 05, v82, 9, , "P(3)", 那么系统在解释程序时会提供下面的错误：

Error At:第 10 行，错误的微地址号！

\$: 16, 05, v82, 9, , "P(3)"

您可以根据系统提供的错误信息很容易改掉程序中的错误，将 82 前的字母 v 去掉就可以了。

系统提供的检错类型有：

(1) 双引号不匹配！

要求微指令说明必须用成对的双引号括起来，而且只能出现一对，不能出现多对双引号或是成单的双引号。

(2) 错误的标题！

第一行程序即为标题行，如果不以“T:”打头，或者双引号外还有别的字母或数字等都可能导致该错误。

(3) 错误的总行数或总行数溢出！

第二行程序即为总行数，如果不以“R:”打头，或者总行数值小于 1，或大于 500，或总行数中含有非数字的字母等都可能导致该错误。

(4) 错误的总列数或总列数溢出！

第三行程序即为总列数，如果不以“C:”打头，或者总列数值小于 1，或大于 500，或总列数中含有非数字的字母等都可能导致该错误。

(5) 错误的指令,参数不全！

从第四行开始，就是普通的指令了，如果指令不是以“\$:”打头，或是指令中的“,”不为 5 个就会导致该错误。

(6) 错误的行号！

指令中行号含有非数字字母等，会导致该错误。

(7) 行号溢出或为空！

行号值小于 0，或行号数大于 500 会导致该错误。

(8) 错误的列号！

指令中列号含有非数字字母等，会导致该错误。

(9) 列号溢出或为空！

列号值小于 0，或列号数大于 500 会导致该错误。

(10) 错误的微地址号！

指令中微地址含有非数字字母等，会导致该错误。

(11) 微地址号溢出！

微地址值小于 0，或微地址数大于 2 位数会导致该错误。

(12) 错误的图形号!

指令中图形号含有非数字字母等，会导致该错误。

(13) 图形号溢出或为空!

图形号值小于 0，或图形号大于 9 会导致该错误。

(14) 错误的测试字号!

指令中测试字号含有非数字字母等，会导致该错误。

(15) 测试字号溢出!

测试字号值小于 0，或测试字号大于 2 位数会导致该错误。

(16) 微指令说明错误!

微指令说明没有被引号括住，或者引号外有其它内容会导致该错误。

(17) 微指令说明中出现非法双引号!

微指令说明中出现多余的双引号会导致该错误。

(18) 微指令说明中字符数超标(大于 16)!

微指令说明中每行的字符是有限制的，不能大于 16 个字节（一个汉字为两个字节），如果过了就会导致该错误。

(19) 指令中行号大于总行数

一旦总行数指定后，指令中所有的行号是不能超过总行数的，如果超过就会导致该错误。

(20) 指令中列号大于总列数

一旦总列数指定后，指令中所有的列号是不能超过总列数的，如果超过就会导致该错误。

(21) 助记符说明错误!

如果助记符说明时指定的总列数（最大列和最小列之差）和实际的助记符个数不相等就会导致该错误。

(22) 找不到测试字中对应的结点

因为测试字字段的用途就是将当前执行的节点和其紧密关联的下个结点关联起来，所以如果指定的测试字号在所有结点中找不到与之对应的结点就会导致该错误。

(23) 第 X 行和第 Y 行和微地址相冲突

我们必须保证每个节点微地址的唯一性，这样才会使得将上位机中的节点和下位机中相应的微地址单元对应起来，如果有两个结点的微地址一样就会导致该错误。

4. 例程说明

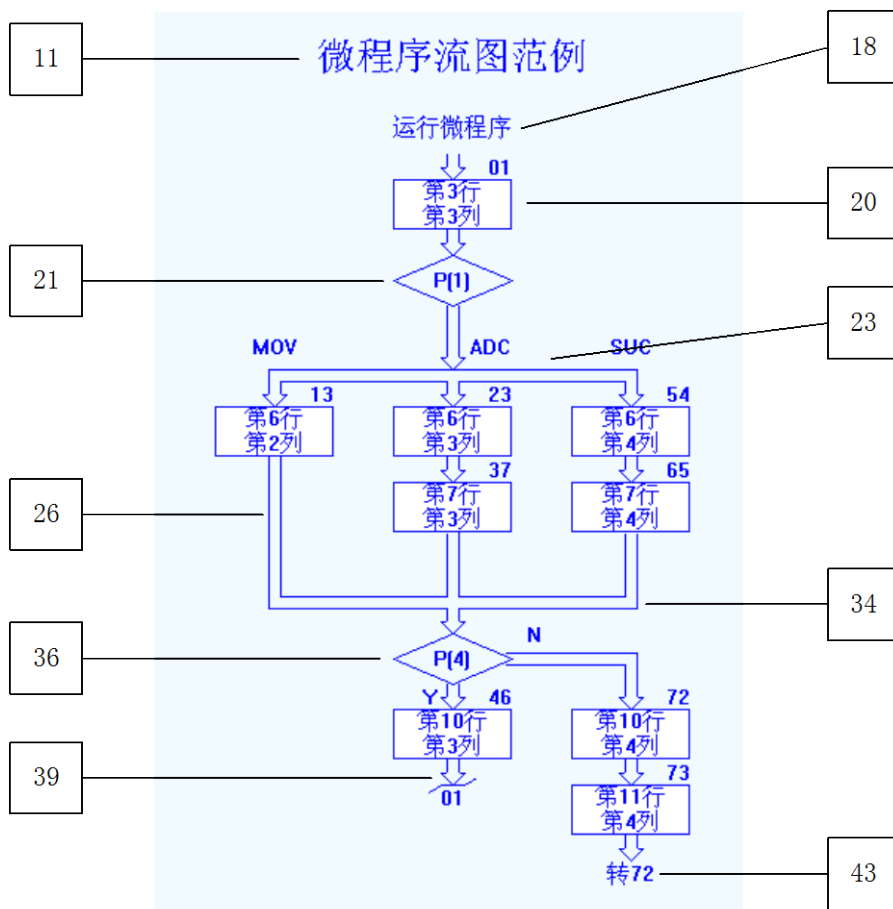
现在假设要生成如附图 1-1 所示的微程序流图，看一看其生成过程。

您可以在安装目录下的 Sample 子目录下找到一个叫微程序流图范例.MFP 的文件，就是该范例，微程序流图文件的内容如下（行号为后来人为加上的）：

```

01 ;//*****
02 ;//
03 ;//      微程序流程图范例文件      ;//
04 ;//
05 ;// FileName: Sample.MFP By TangDu Co.,LTD ;//
06 ;//*****

```



附图 1-1 微程序流程图示例

```

07
08 ;//*****
09 ;//      标题及总行、列数
10
11 T : "3:微程序流程图范例"
12 R : 13
13 C : 05
14
15 ;//*****
16 ;//      运行微程序
17
18 $ : 02, 03, , 5, , "运行微程序"
19

```

```

20 $ : 03, 03, 01, 1, 80, "第 3 行:第 3 列"
21 $ : 04, 03, 80, 2, , "P(1):Poilk:oifu"
22
23 $ : 05, 03, 81, 3, , "2:4:MOV:ADC:SUC"
24
25 $ : 06, 02, 13, 1, 82, "第 6 行:第 2 列"
26 $ : 07, 02, 82, 7, 83, ""
27
28 $ : 06, 03, 23, 1, , "第 6 行:第 3 列"
29 $ : 07, 03, 37, 1, 83, "第 7 行:第 3 列"
30
31 $ : 06, 04, 54, 1, , "第 6 行:第 4 列"
32 $ : 07, 04, 65, 1, 83, "第 7 行:第 4 列"
33
34 $ : 08, 03, 83, 8, 91, "2:4"
35
36 $ : 09, 03, 91, 9, , "P(4)"
37
38 $ : 10, 03, 46, 1, , "第 10 行:第 3 列"
39 $ : 11, 03, , 4, , "01"
40
41 $ : 10, 04, 72, 1, , "第 10 行:第 4 列"
42 $ : 11, 04, 73, 1, , "第 11 行:第 4 列"
43 $ : 12, 04, , 6, , "72"
44
45 ;//*****
46 ;// End of All

```

对于每种图形类型在上图中都有体现，而每种图形和实际程序行号的对应关系也已在图中标注出来。第 01-06 行为注释行，第 07 行为空行，第 11 行为标题行，第 12、13 分别对总的行列数。

图形号 1 对应程序中的第 20、25 行等，对应图中的 20；

图形号 2 对应程序中和图中的 21；

图形号 3 对应程序中和图中的 23；

图形号 4 对应程序中和图中的 39；

图形号 5 对应程序中和图中的 18；

图形号 6 对应程序中和图中的 43；

图形号 7 对应程序中和图中的 26；

图形号 8 对应程序中和图中的 34；

图形号 9 对应程序中和图中的 36；

注意：在微程序流图文件的微指令说明字段中不能含有：逗号、引号、分号以及星号 ‘*’。每个微程序流图文件至少要有 4 行，前三行分别为标题、总行数和总列数，再就是一行 \$ 指令。

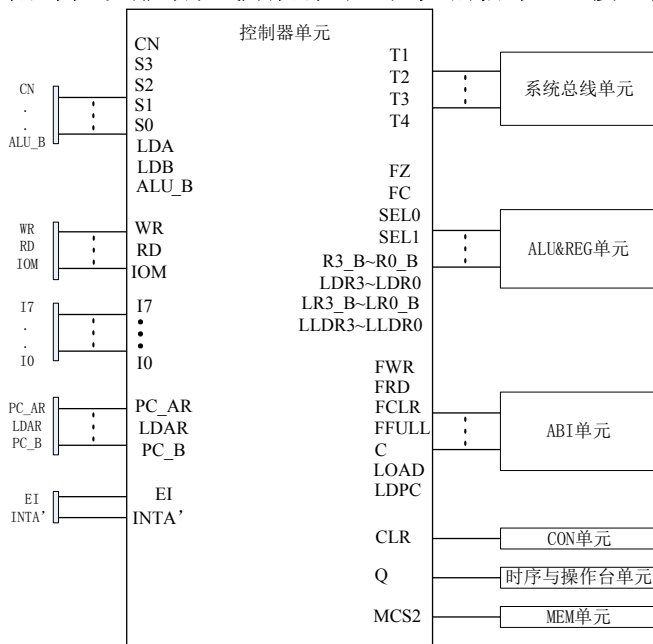
附录 2 控制器单元和扩展单元对应 FPGA 引脚配置说明

本实验平台提供了两块 FPGA，一块位于控制器单元，一块位于扩展单元，芯片都是 Intel 公司的 FPGA。

1、控制器单元中的 FPGA 引脚分配如下表所示：

引脚	信号	引脚	信号	引脚	信号	引脚	信号	引脚	信号
3	CLR	52	T1	84	R0_B	91	I0*	137	FRD
11	PC_B*	54	T2	86	R1_B	90	I1*	135	FWR
23	Q	58	T3	142	R2_B	88	I2*	121	FCLR
44	LDR0	60	T4	144	R3_B	89	I3*	132	FULL
51	LDR1	49	LLDR0	75	LR0_B	138	I4*	125	C
55	LDR2	53	LLDR1	73	LR1_B	136	I5*	115	FC
67	LDR3	59	LLDR2	71	LR2_B	133	I6*	119	FZ
126	LOAD	65	LLDR3	69	LR3_B	129	I7*	141	S0*
127	LDIR	98	WR*	120	LDPC	128	MCS2	143	S1*
114	LDAR*	100	RD*	87	LDA*	111	SEL0	99	S2*
124	PC_AR*	103	IOM*	85	LDB*	113	SEL1	101	S3*
106	EI*	112	/INTA*					105	CN*

注：带*的信号在控制器单元以排针形式开放引出，不带*的信号已连接至各单元，如下图所示：



其中带*的信号有：

- CN、S3~S0、LDA、LDB、ALU_B 是运算器控制信号；
- WR、RD、IOM 是总线写、读、IO/存储器选择信号；
- I7~I0 是指令寄存器输入信号；

- PC_AR、LDAR、PC_B 是 ABI 控制信号。
- EI, INTA' 是中断控制信号。

不带*的信号有:

- CLR 是总清零信号, 已与 CON 单元的 CLR 连接;
- LOAD、LDPC 是 ABI 控制信号, 已与 ABI 单元的 LOAD、LDPC 连接;
- LDIR 是指令寄存器请求信号, 已与指令寄存器 IR 的 LDIR 连接;
- FRD、FWR、FCLR、FULL、C 是 FIFO 控制信号, 已与 ABI 单元的 FRD、FWR、FCLR、FULL、C 连接;
- SEL0、SEL1 是运算器和寄存器堆的通路选择信号, 已与 ALU® 单元的 SEL0、SEL1 连接;
- FZ、FC 是运算器标志信号, 已与 ALU® 单元的 FZ、FC 连接;
- MCS2 是存储器右路片选信号, 已与 MEM 单元的 MCS2 连接;
- Q 是时序信号, 已与时序操作台的 Q 连接;
- T1~T4 是时序信号, 已与系统总线的 T1~T4 连接;
- LDR3~LDR0、R3_B~R0_B 是寄存器堆右路输入、输出控制信号, 已与 ALU® 单元的 LDR3~LDR0、R3_B~R0_B 连接;
- LLDR3~LLDR0、LR3_B~LR0_B 是寄存器堆左路输入、输出控制信号, 已与 ALU® 单元的 LLDR3~LLDR0、LR3_B~LR0_B 连接。

2、扩展单元中的 FPGA 引脚分配如下表所示:

引脚	排针	引脚	排针	引脚	排针	引脚	排针	引脚	排针
70	X0	1	Y0	106	U0	24	Z0	2	H0
68	X1	28	Y1	103	U1	32	Z1	7	H1
66	X2	136	Y2	100	U2	34	Z2	3	H2
60	X3	138	Y3	98	U3	39	Z3(IN)	43	H3
58	X4	46	Y4	144	U4	91	Z4(IN)	44	H4
54	X5	76	Y5	142	U5	89	Z5(IN)	11	H5
52	X6	74	Y6	86	U6	88	Z6(IN)	77	H6
50	X7	72	Y7	84	U7	90	Z7(IN)		

注: 扩展单元的排针名称后有 (IN) 只能作为输入引脚, 其余可作为输入/输出引脚。