

A network diagram featuring several blue human silhouettes connected by a web of blue lines. The background is a light beige color with a faint world map. The text '计算机组成原理课程设计' is overlaid in a green, bold font.

计算机组成原理课程设计

实验 6

微控器实验



实验 6 微控器实验

一、实验目的

二、相关单元

三、上位机软件

四、实验原理

五、实验要求

六、思考



一、实验目的

1. 掌握微控制器的工作原理
2. 掌握装入、执行微程序的方法，观察微程序的运行过程。
3. 掌握上位机软件的使用方法。





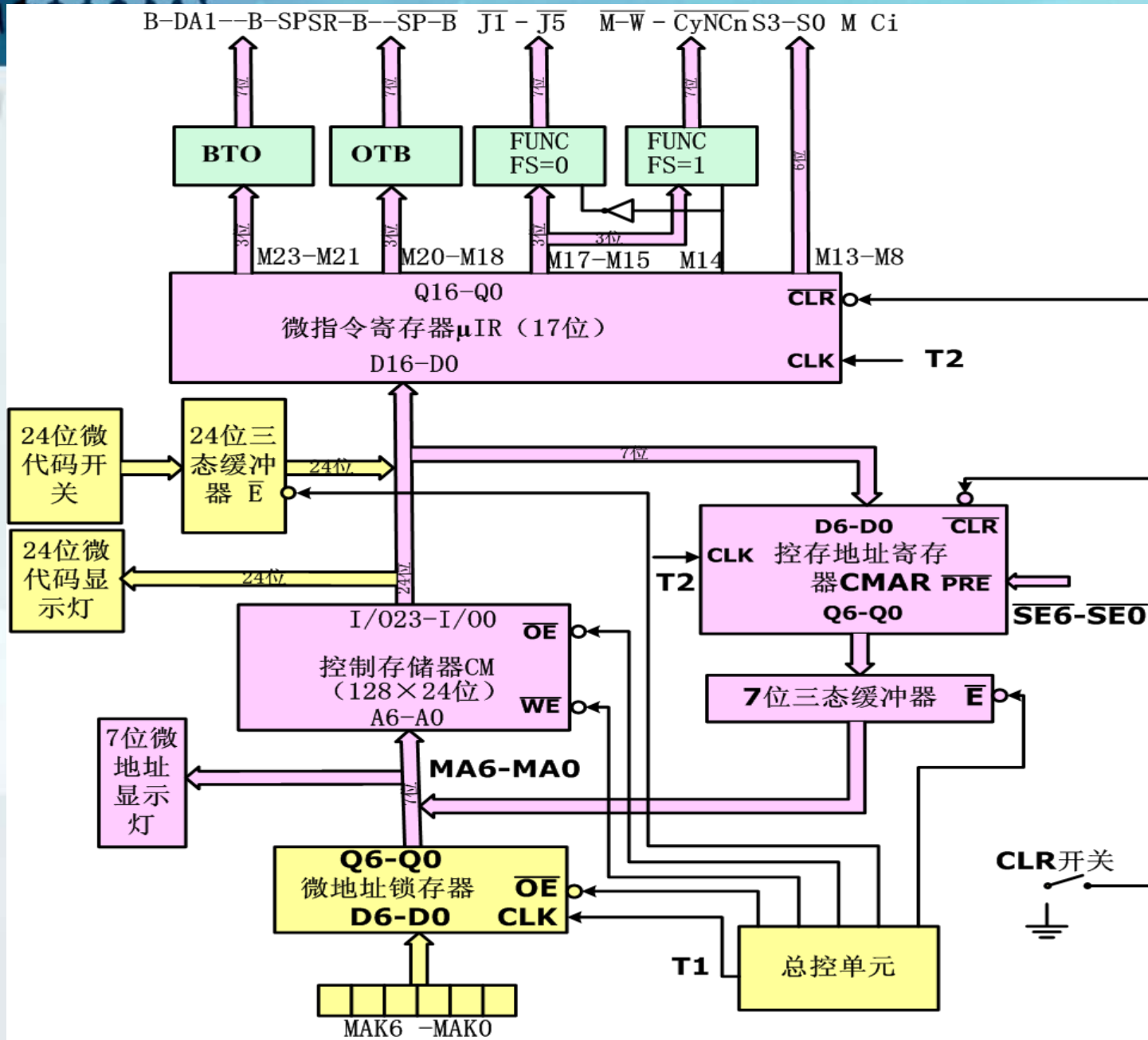
二、相关单元

1. 微控器单元（MAIN CONTROL UNIT）

2. 时序电路单元（CLOCK UNIT）、手动单元（MANUAL UNIT）、总线单元（BUS UNIT）、通用寄存器单元（REG UNIT）、运算器单元（ALU UNIT）、输入/输出单元（OUTPUT/INPUT UNIT）、地址单元（ADDRESS UNIT）、指令寄存与译码单元（INS UNIT）和存储器单元（MEM UNIT）。



微控器逻辑框图



Main

(4) 微指令译码器

(3) μ IR

(1) CM

(5) μ AR

(2) C

M

A

R

(6) 开关与显示灯

⑦ 编程开关及控制电路

点击①

~

⑦

打开链接





(1) 控制存储器 CM

✓CM 由 3 片 2816 ($2K \times 8$ 位) 组成, 存放 24 位的微指令。3 片 2816 的高位地址 $A_{10} \sim A_7$ 均接地, 因此, 控存的实际容量为 128×24 位, 即可以存放 128 条微指令。

(1) 控制存储器 CM

- 控存的地址端接有 7 个微地址显示灯，控存的地址

三态开关：=PROM 或者

①可以由微地址锁存器 μ AR^{=READ} 提供（手动“编程 PROM”或“校验 READ”状态下）；

②也可以由后继微地址修改逻辑来提供（“运行 RUN”状态下），

③由 PC 机控制送出（联机状态下）。

三态开关：
=RUN

- 控存中的 24 位微码，高 16 位送微指令寄存器 μ IR 保存并译码，低 7 位（下址字段）送散转逻辑电路。



(2) 控存地址寄存器 CMAR

- CMAR 功能是：由输入信号 $SE6\# \sim SE0\#$ 控制修改**当前**微指令的下址字段 $M6 \sim M0$ （即 $MA6 \sim MA0$ ），以产生即后继微地址。
- CMAR 由四片 74LS74（2 位带清零预置端的寄存器）和一片 74LS245（8 位三态缓冲器）连接而成，四片 74LS74 的清零端均接 **CLR 开关**，预置端则分别接自输入信号 $SE6\# \sim SE0\#$ ，因为预置端低电平有效，所以当 $SEi\#=0$ 时，相应的 Mi 即 **MAi** 被置 1。

(2) 控存地址寄存器 CMAR

- 四片 74LS74 的输入端接控存 CM 的数据输出端的最低 7 位 $M_6 \sim M_0$ ，经过预置端 $SE_6\# \sim SE_0\#$ 的修改，形成输出信号 $MA_6 \sim MA_0$ ，连接至三态缓冲器（74LS245）的输入，输出则作为后继微地址送至控存 2816 的地址输入端，以寻址下条微指令。三态门 74LS245 的使能信号 $E\#$ 由编程开关的控制电路产生。



(3) 微指令寄存器 μIR



- 微指令寄存器 μIR 的功能是存放从控存 CM 中读出的高 16 位的微码 $M23 \sim M8$ （低 7 位 $M6 \sim M0$ 是下址字段， $M7$ 为空）
- μIR 由两片 74LS273（8 位寄存器）组成的。
 - ① 清零端 $CLR\#$ 接 CLR 开关，在总清时 μIR 清零。
 - ② 数据输入端接控存 CM 的数据 I/O 端；
 - ③ 数据输出端，一面以排针 $M13 \sim M8$ 的形式直接引出，另一面， $M23 \sim M14$ 另送微指令译码器进行译码。 $M23 \sim M0$ 的定义见简单

(4) 微指令译码器

- 微指令译码器由两片 74LS138 和两片 GAL 芯片组成，其功能是根据微指令的格式及各字段的定义，将 μIR 送来的编码字段进行译码，产生全机所需要的各种微操作控制信号，以实现该条微指令功能。



(5) 微地址锁存器 μ AR

- 专供实验仪手动操作设置。
- μ AR 由一片 74LS374（8 位锁存器）构成，用于锁存手动操作时由开关拨入的微地址 MA6 ~ MA0，并提供给控存。
- μ AR 的输入端以排针的形式引出，标记为 MA6 ~ MA0，用以连接手动单元（MANUAL UNIT）的开关；输出端接至控存的地址输入端；但 μ AR 的输出使能 OE# 则同样由 GAL 芯片控制产生。





(6) 开关与显示灯

- 微控器单元包含有一组逻辑开关和两组发光二极管显示灯：

①微代码输入开关 MK23 ~ MK0：通过三片三态缓冲器 74LS245（8 位）接在控存的数据总线上，用于手动拨入 24 位微码。三态门 74LS245 的 DIR 接地，即数据传送 B 到 A；其使能信号 E# 则由编程开关的控制电路产生，控制三态门的打开和关闭。专供实验仪

(6) 开关与显示灯

②微地址输入开关 MAK6 ~ MAK0 : 通过微地址锁存器 74LS 374 接控存的地址线 MA6 ~ MA0 。 **专供实验仪手动操作设置。**

③微代码显示灯 MD23 ~ MD0 : 经驱动接控存的数据总线上, 用于指示当前读出的微指令的 24 位微码。

④微地址显示灯 MA6 ~ MA0 : 经驱动接控存的地址线, 用于指示下条微指令的 7 位微地址 (修改过的后继微地址) 。





(7) 编程开关及控制电路

- 专供实验仪手动操作设置。
- 编程开关：有三种状态： PROM（编程）、 READ（校验）、 RUN（运行）。
- 根据编程开关的状态，控制电路产生各芯片的控制或使能信号，来实现各种不同的手动和联机操作。
 - ①编程开关关于编程状态“ PROM” 时：手动输入微码
 - ②编程开关关于校验状态“ READ” 时：手动校验微码
 - ③编程开关关于运行状态“ RUN” 时：运行微程序



(7) 编程开关及控制电路

- 专供实验仪手动操作设置。
- 编程开关 =**PROM**：装入微码（写控存）
 - ✓ 微地址由 μ AR（接微地址开关）提供，微码由微码开关提供，CM 执行写操作
- 编程开关 =**READ**：校验微码（读控存）
 - ✓ 微地址由 μ AR（接微地址开关）提供，CM 执行读操作，读出的微码在微码显示灯显示
- 编程开关 =**RUN**：执行微码（运行微程序）
 - ✓ 微地址由 CMAR 提供，CM 执行读操作，读出的微码送 μ IR（同时在微码显示灯显示）。

(7) 编程开关及控制电路

■ 在“RUN”状态下

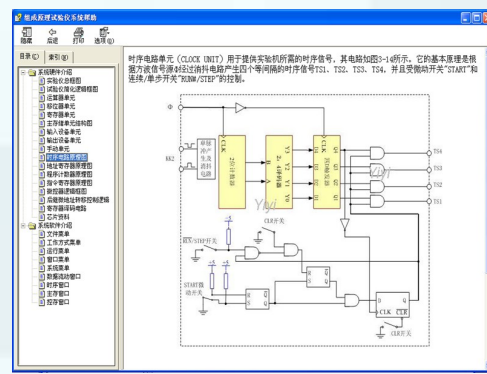
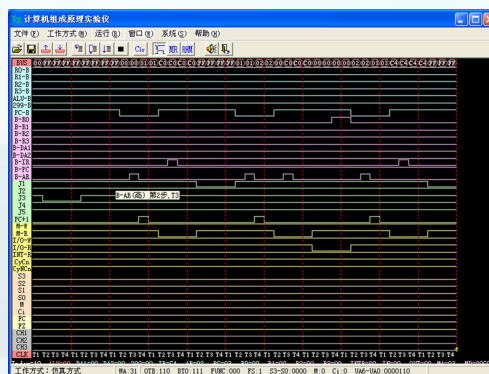
- ①微地址锁存器 μ AR (374) 的输出无效 (输出使能信号 $OE\#=1$)
- ②微代码输入开关的三态门 (245) 关闭 (使能信号 $E\#=1$)
- ③后续微地址三态门 (245) 打开 (使能信号 $E\#=0$)
- ④控存 (2816) 读有效 (片选 $CS\#=0$, 输出使能 $OE\#=0$, 写使能 $WE\#=1$) 。
- ⑤此时, 若启动时序电路, 即可从微地址显示灯所指示的微地址开始向下运行。

三、上位机软件

上位机虚拟软件（联机实验）

下载地址：

http://jpkc.hdu.edu.cn/computer/zcyl/kczy_view.asp?id=117



三、上位机软件



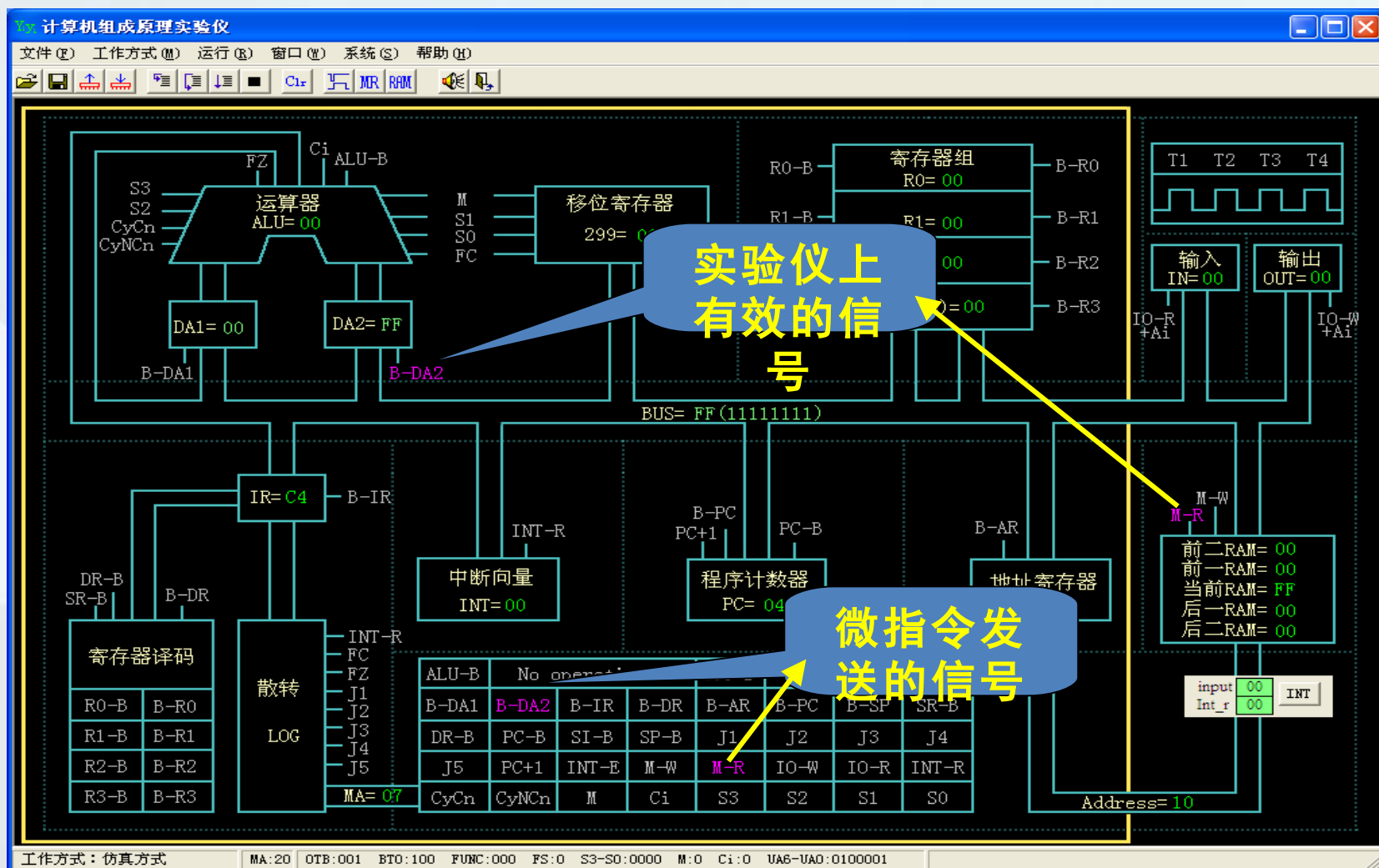
1. 三种工作方式

■ 联机主控（M）

①实验仪上的所有操作都必须在上位机软件的控制下完成。例如，实验仪上 START、K K2 等控制键失效。

②在联机主控方式下，通过观察上位机软件的动态流程界面，可以检查出实验仪接线是否正确。

在联机主控方式下，检查实验仪接线是否正确



三、上位机软件



1. 三种工作方式

- 联机从控（S）：上位机软件只作为一个上下载程序及微程序的工具，所有的时序信号和控制操作都由实验仪自己控制发出。
- 仿真方式（E）：**完全脱离实验仪**，由上位机软件和仿真机一起仿真地完成各个实验。对上位机软件而言，其操作与在联机主控方式下的操作完全一样。仿真机由仿真机软件实现，**仿真机软件包含在上位机软件中**。

三、上位机软件

2. 文件下装



- 下装（D）把上位机中的程序和微程序分别装入实验仪 / 仿真机的主存和控存中。
- 程序和微程序 **只有被下装后，才能被执行。**
- 此功能与主存、控存窗口中写入功能的不同是：
下装（D）功能是一次性将文件装入实验仪 / 仿真机，主存、控存窗口中的写入功能更方便于在修改某个单元内容后，将修改过的内容写入实验仪 / 仿真机。





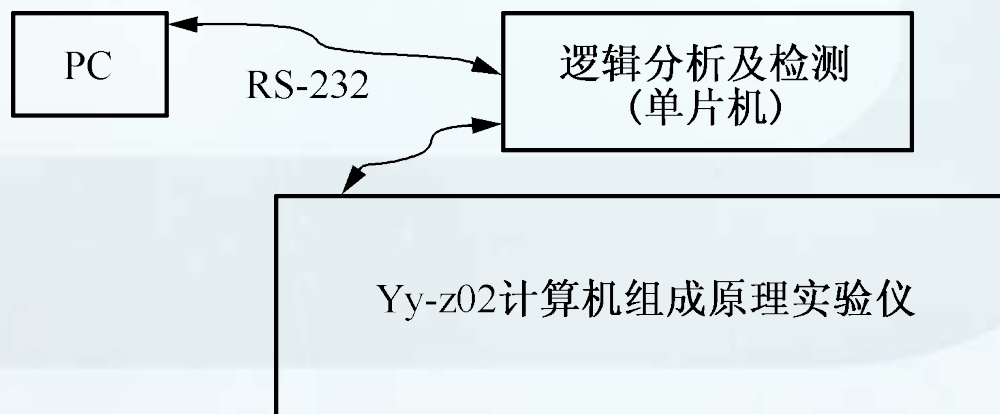
四、实验原理

1. 模型机与 PC 机的连接
2. 微程序流程图和微指令编码
3. 装入微码的方法
4. 微码的校验方法
5. 执行微程序方法



1. 实验仪与 PC 的连接

- 单片机监测单元放在电路板背面，它采集信号使上位机能**实时反映实验仪状态**
- 联机实验前，应将实验仪断电，通过串口与 PC 连接，然后接线，打开实验仪电源开关，开始实验



2. 微程序流程图和微指令编码

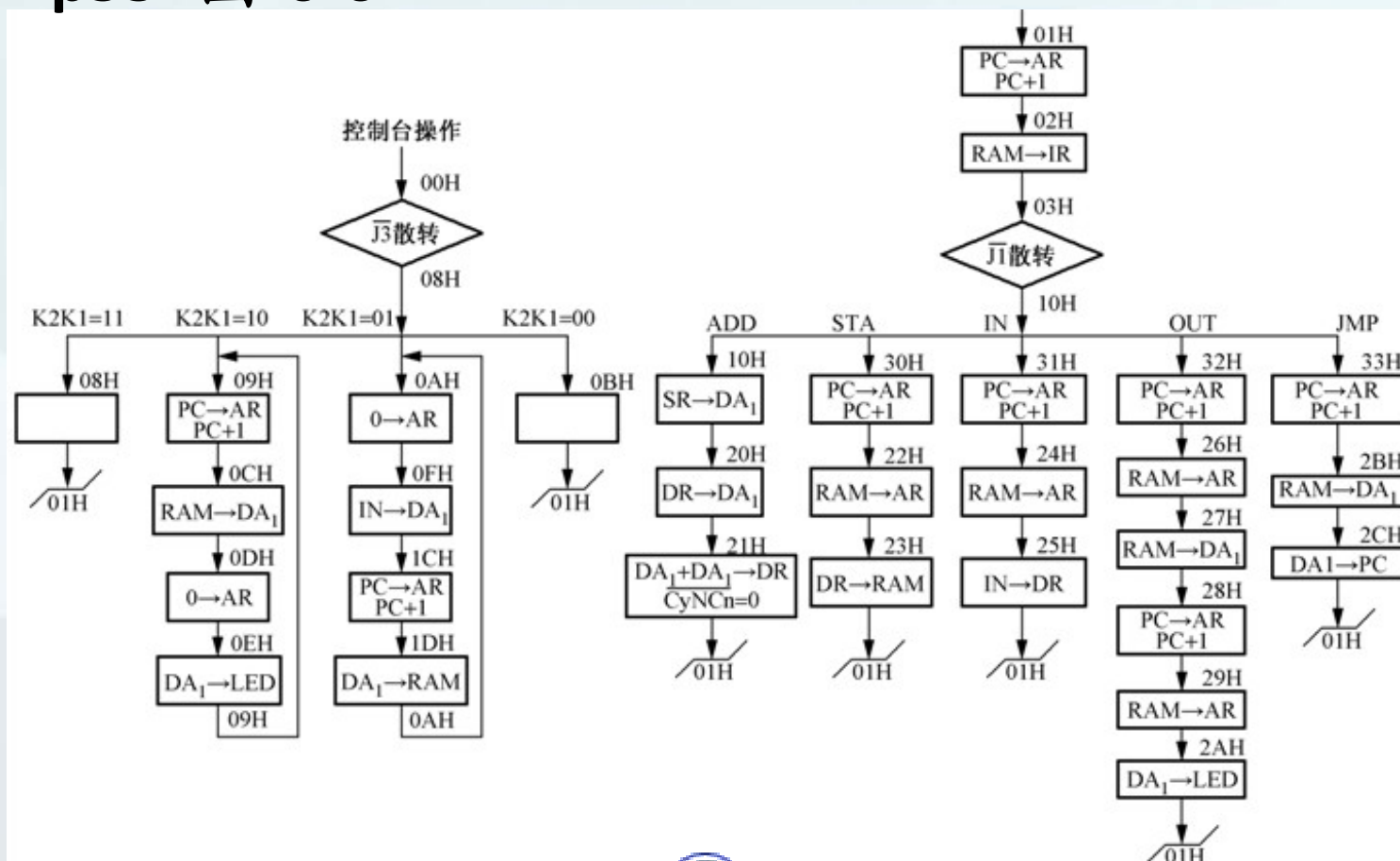
■ p54 表 6-4

微地址	微代码	BTO	OTB	FUNC	FS	S3 S2 S1 S0 M Ci	N	下址	微指令注释
00H	01C008	000	000	011	1	0 0 0 0 0 0	0	0001000	$\overline{J_3}$
01H	DC4002	110	111	000	1	0 0 0 0 0 0	0	0000010	PC→AR, PC=PC+1
02H	610003	011	000	010	0	0 0 0 0 0 0	0	0000011	RAM→IR
03H	00C010	000	000	001	1	0 0 0 0 0 0	0	0010000	$\overline{J_1}$
08H	000001	000	000	000	0	0 0 0 0 0 0	0	0000001	NULL
09H	DC400C	110	111	000	1	0 0 0 0 0 0	0	0001100	PC→AR, PC=PC+1
0AH	C40E0F	110	001	000	0	0 0 1 1 1 0	0	0001111	0→AR
0BH	000001	000	000	000	0	0 0 0 0 0 0	0	0000001	NULL
0CH	21000D	001	000	010	0	0 0 0 0 0 0	0	0001101	RAM→DA ₁
0DH	C40E0E	110	001	000	0	0 0 1 1 1 0	0	0001110	0→AR
0EH	058109	000	001	011	0	0 0 0 0 0 1	0	0001001	DA ₁ →OUT
0FH	22001C	001	000	100	0	0 0 0 0 0 0	0	0011100	IN→DA ₁

.....

2. 微程序流程图和微指令编码

■ p55 图 6-6



3. 装入微码的方法

- **手动装入：**通过 24 位微代码开关和 7 位微地址开关（不做要求）
- **联机装入：**通过上位机（虚拟仿真）软件，打开文件→下装





手动装入微码方法

- (1) 将微控器单元 (MAIN CONTROL UNIT) 右上角的编程开关置于 “ **PROG** ” 状态。
- (2) 将时序电路单元 (CLOCK UNIT) 中的 **RUN#/STEP** 开关置于 “ **STEP** ” 状态。
- (3) 从手动单元 (MANUAL UNIT) 中的开关 “ **MA6 ~ MA0** ” 上拨入微地址 **MA6 ~ MA0** 。

手动装入微码方法

- (4) 从微代码开关 MK23 ~ MK0 上拨入微代码，24 位开关对应着 24 位显示灯（灯亮表示“1”，灯灭表示“0”）
- (5) 按动 START 按键，启动时序，即将 24 位微代码写入到控存（2816）的相应单元（由微地址 MA6 ~ MA0 所指定）中。
- (6) 重复步骤 3 ~ 5，即可将微指令代码一条条装入控存。



4. 微码的校验方法


- 手动校验：（不做要求）

- 联机校验：

- ①直接打开上位机软件的**控存窗口**校验；

- ②如果微码是直接从实验仪上输入，则在上位机软件“联机从控”模式下“上传”文件，实现校验。





微码手动校验方法

- (1) 将编程开关置于“**READ**”状态。
- (2) 将时序电路单元（**CLOCK UNIT**）中的**RUN#/STEP**开关置于“**STEP**”状态。
- (3) 从手动单元（**MANUAL UNIT**）中的开关“**MA6 ~ MA0**”上拨入微地址 **MA6 ~ MA0**。

微码手动校验方法

- (4) 按动 START 按键，启动时序，读出微代码，观察显示灯 MD23 ~ MD0 的状态（灯亮表示“1”，灯灭表示“0”），检查读出的微代码是否与写入的相同。如果不同，则将编程开关重新置于“**PROG**”状态，再执行装入操作，装入正确的微代码即可。
- (5) 重复步骤 1 ~ 4，即可校验每一条微指令代码。





联机装入微码、程序代码方法

■ 上位机软件工作方式：

- ①**联机主控**：可以通过软件来**控制**和**动态显示**指令在实验仪上执行过程；上下载程序、微程序；执行程序、微程序、单步程序、单步微程序、连续执行、停止执行
- ②**联机从控**：软件只用来上下载程序、微程序。
- ③**仿真方式**：不使用实验仪，所有部件用**软件模拟仿真实现**。

联机装入微码、程序代码方法

- 编辑微码：控存窗口
- 装入微码：下载按钮、菜单
- 编辑程序代码：主存窗口
- 装入程序代码：下载按钮、菜单
- 按照模型机的硬件机理（Reset 后，PC 清零，CMAR 清零），取指令微程序段应从 0 号控存开始存放，而程序代码应从 0 号内存开始存放。





手动装入程序机器码

- 执行控制台“写内存”操作，通过 8 位输入开关输入程序和数据

①将编程开关置于“**RUN**”状态，RUN#/STEP 开关置于“**STEP**”状态。

②操作 CLR 开关，使 CLR 信号“ $1 \rightarrow 0 \rightarrow 1$ ”，程序计数器 PC 清零，微地址清零。

③开关 K2K1 置为“0 1”，即写内存状态。



手动装入程序机器码

- ④ 按动启动键 **START** 一次，则从控存 **00H** 单元开始执行微指令，微地址显示灯显示 “0001010”，第二次按动 **START** 键，微地址显示灯显示 “001111”，此时，将数据开关置为要写入的机器指令代码或数据，再按动 **START** 键两次，即完成该条指令的写入，同时 PC 指向下一个内存单元。继续按动 **START** 键，当且仅当微地址显示灯显示 “0001111” 时，才从开关上置入指令代码，直至所有的程序代码写入完毕。



手动校验程序机器码

- 执行控制台“读内存”操作，通过输出设备（8 位 LED 显示灯）显示程序和数据代码

① 将编程开关置于“RUN”状态，RUN#/STEP 开关置于“STEP”状态。

② 操作 CLR 开关，使 CLR 信号“ $1 \rightarrow 0 \rightarrow 1$ ”，程序计数器 PC 清零，微地址清零。



手动校验程序机器码

④按动启动键 START 一次，则从控存 00H 单元开始执行微指令，微地址显示灯显示“0001001”，第二次按动 START 键，微地址显示灯显示“0001100”，第三次按动 START 键，微地址显示灯显示“0001101”，第四次按动 START 键，微地址显示灯显示“0001110”，此时，输出设备（OUTPUT DEVICE）发光管上将显示内存 00H 号单元的内容，检查是否与写入的数据相同。




手动校验程序机器码

⑤继续按动 START 键，当且仅当微地址显示灯显示“0001001”时，发光管上显示的内容才是内存的数据。每个循环 PC 会自动增 1，由此，可检查后续单元的内容是否正确。



Why?





5. 执行微程序的方法

- 脱机执行：执行控制台“启动程序”操作指令（不做要求）

① K2 K1 = 00/11，编程开关（三态）= RUN

✓ 单步执行：RUN#/STEP = STEP

✓ 连续执行：RUN#/STEP = RUN #

② CLR 信号“1→0→1”，使微地址清零，PC 清零，即程序首址为 00H

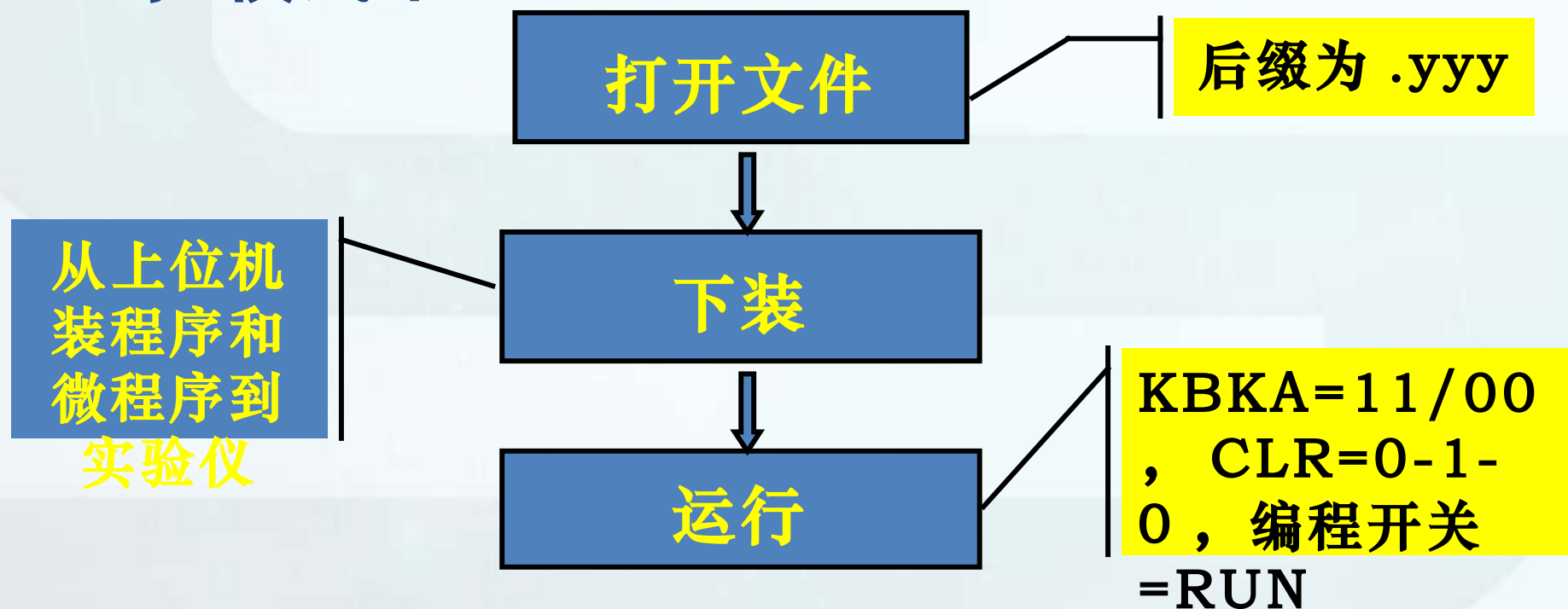
③ 按动 START 键，单步执行（微指令）或者连续执行。


- 联机执行：

① 联机主控：通过软件控制

使用上位机软件装入程序和微程序

主控模式下：





五、实验要求

1. 根据实验原理和相关单元电路，画出**实验接线图**，连接线路，开始实验；
2. 从零地址开始执行微程序，观察并记录微地址的变化顺序，解释其原理。





六、思考

若要强制使**取指令**微程序从 **07H** 单元开始执行，请结合微控器的结构和原理，说明你的方法，并实现。





The End !