


# 第7章

## 数字系统设计基础



# 概述

- 传统的真值表、卡诺图、状态转移图等方法设计电路需要凭设计者的经验，而且不适合大规模的数字系统设计。需要一种数字系统的设计方法，突破传统方法的局限性。
- 本章将详细介绍数字系统的描述工具：方框图，算法流程图、处理器明细表、**ASM**图。

# 7.1 概述

## 7.1.1 数字系统的基本模型

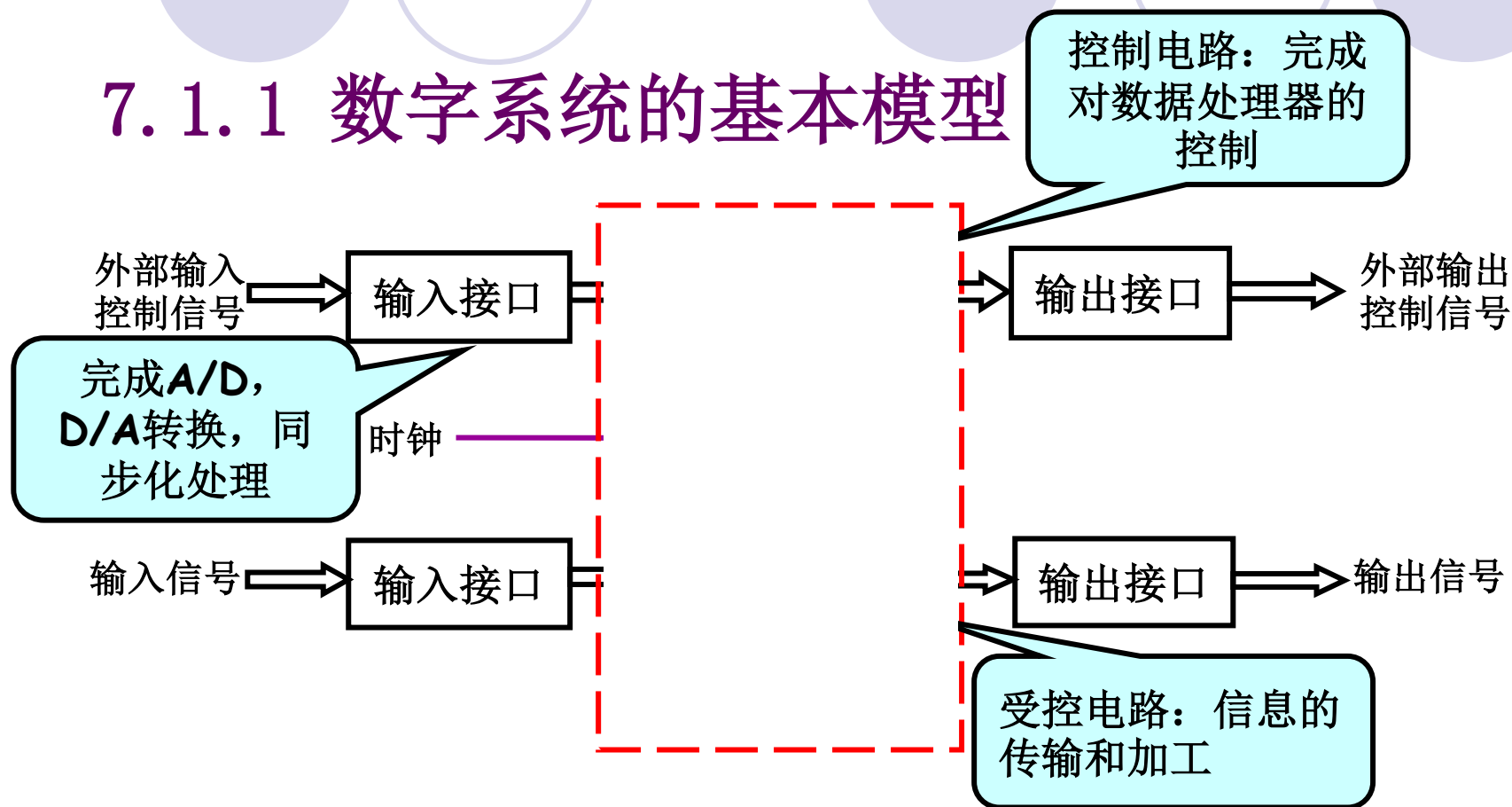


图7.1.1 数字系统的一般模型

# 7.1 概述

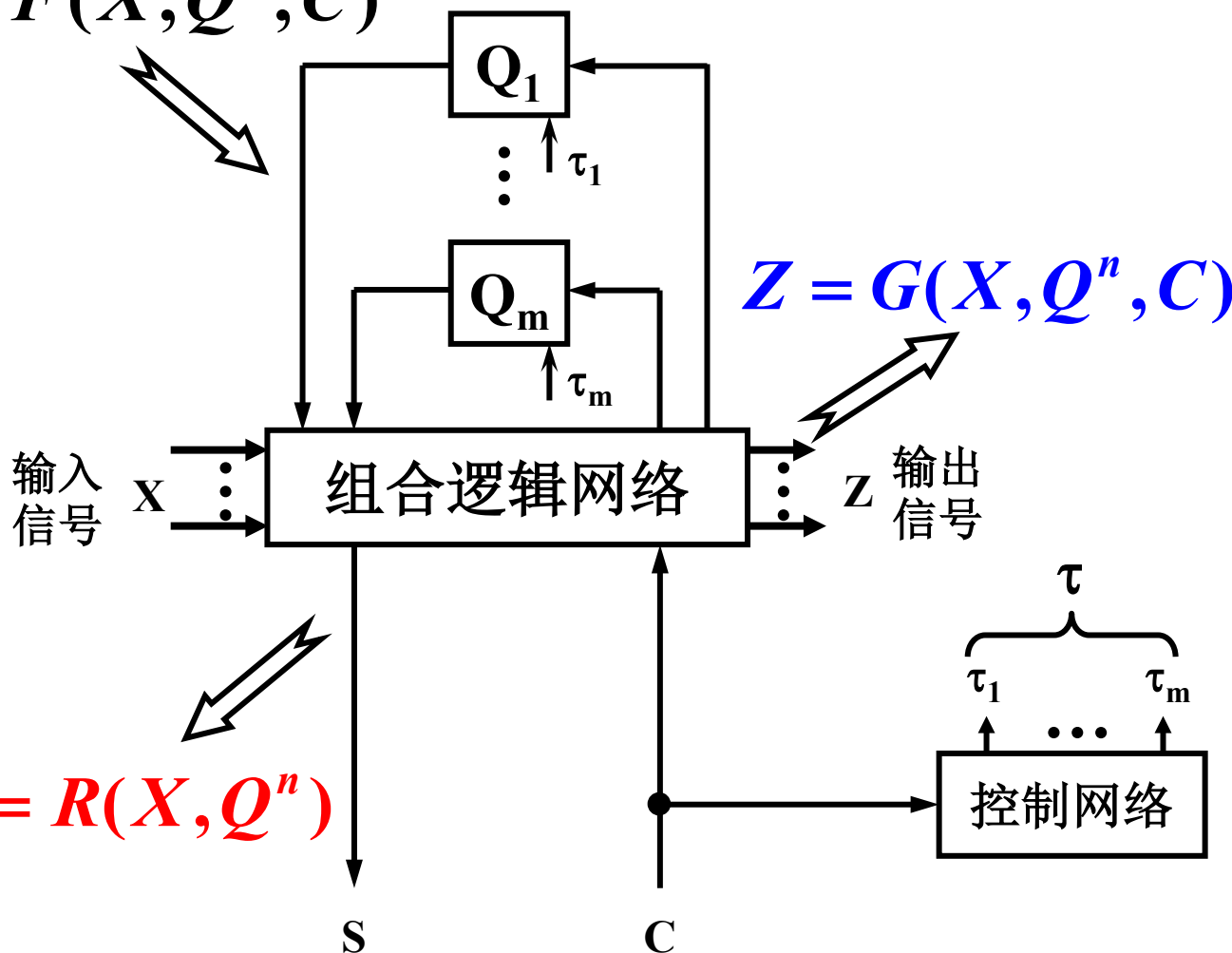
## 7.1.1 数字系统的基本模型

$$Q^{n+1} = F(X, Q^n, C)$$

$$Z = G(X, Q^n, C)$$

$$S = R(X, Q^n)$$

数  
据  
处  
理  
器  
模  
型



# 7.1 概述

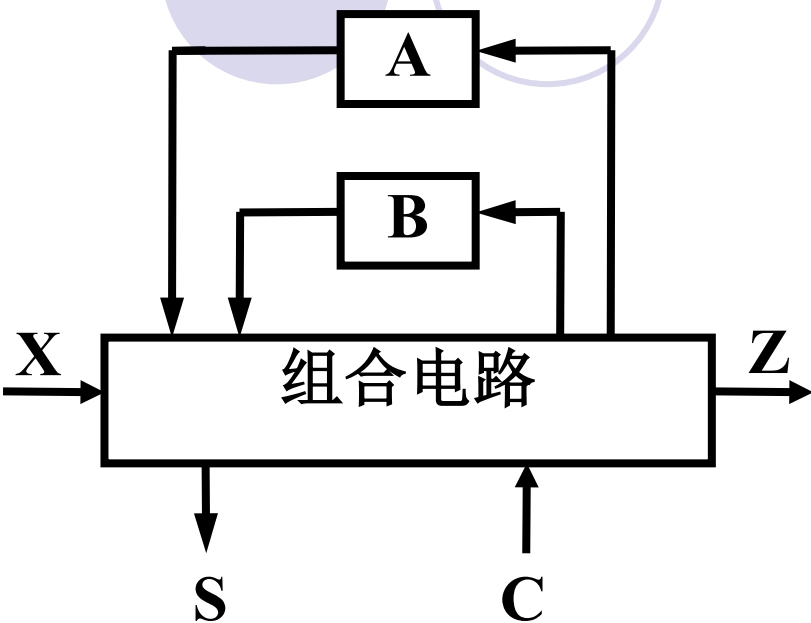
## 7.1.1 数字系统的基本模型

**数据处理器的描述方法：** 明细表——规定数据处理任务的表格。

明细表包括两个子表：

- **操作表：** 列出在控制信号下，数据处理器应实现的操作。
- **状态变量表：** 定义数据处理器输出的状态变量和信号。

设一个简单数据处理器，如图：



操 作 表		状 态 变 量 表	
控制信号	操 作	状态变量	定 义
NOP	无操作	$S_1$	$X > 0$
ADDA	$A \leftarrow A + X$	$S_2$	$X < 0$
ADDB	$B \leftarrow B + X$	输出 $Z = A$	
CLAB	$A \leftarrow 0, B \leftarrow 0$		

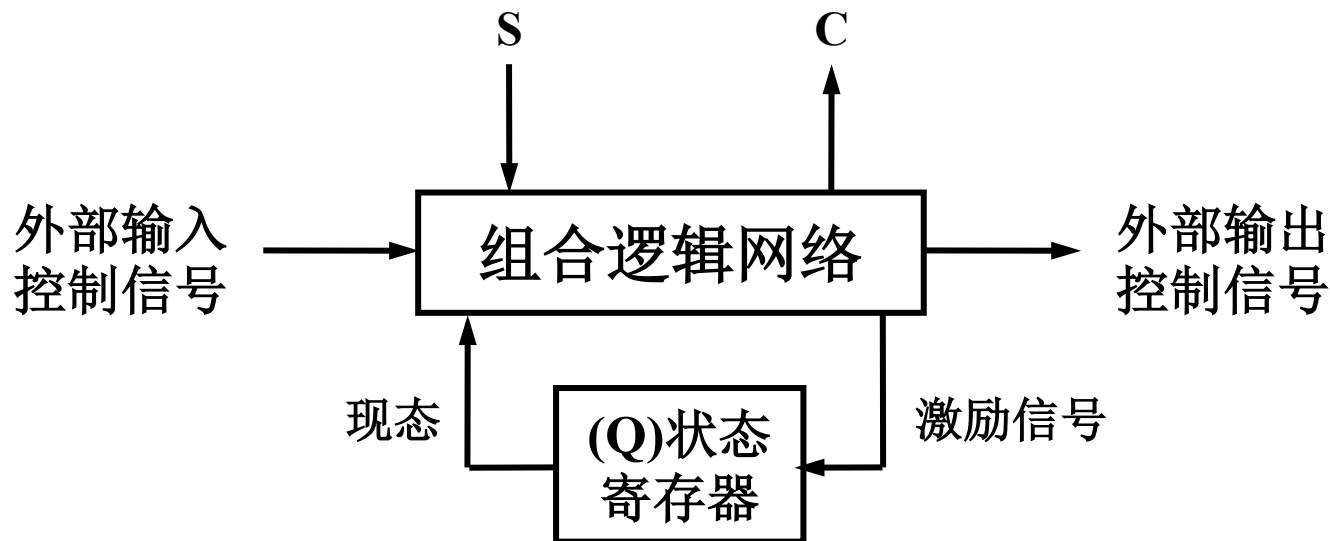
- 输入信号X；
- 控制信号C1、C2、C3和C4（分别记为NOP、ADDA、ADDB和CLAB）；
- 两个寄存器A、B。
- 输出状态信号S1、S2及信号Z。

# 7.1 概述

## 7.1.1 数字系统的基本模型

**控制器的描述方法：** 状态转移图或状态转移表

- 实现一个计算任务，必存在一个算法，控制器就是用来规定算法的步骤；
- 控制器决定算法步骤，必须有记忆能力，所以它是一个时序电路，应包含存储器。



# 7.1 概述

## 7.1.2 同步数字系统时序约定

假设为同步时序系统，满足以下条件：

- (1) 只有一个系统时钟；
- (2) 输入信号都与系统时钟同步；
- (3) 系统时钟同时到达所有存储元件的时钟脉冲输入端。

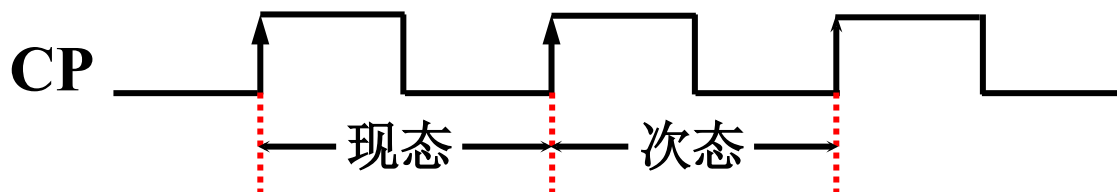


图7.1.4 系统时钟脉冲波形



# 7.1 概述

## 7.1.2 同步数字系统时序约定

### 1. 最小时钟周期

$CP \uparrow \rightarrow S$ （状态信号）稳定  $\rightarrow C$ （控制信号）稳定  $\rightarrow \tau$ （寄存器功能选择信号）、 $Z$ （输出）稳定  $\rightarrow CP \uparrow$ 。

### 2. 异步输入信号转换成同步输入信号

异步输入信号：早于或晚于系统时钟有效沿出现的输入信号。

# 7.1 概述

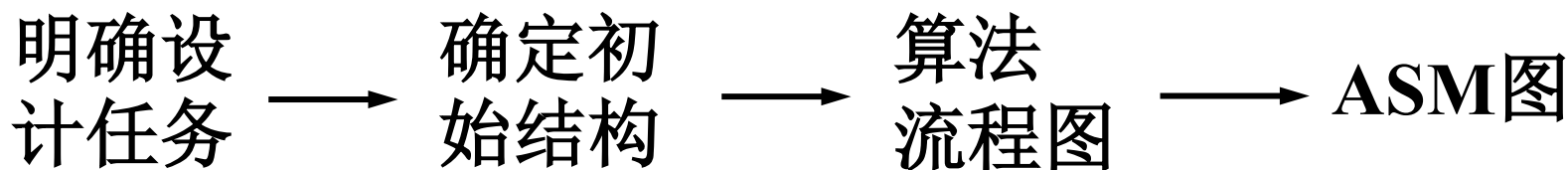
## 7.1.4 数字系统的设计步骤

1. 系统设计
- ↓
2. 逻辑设计
- ↓
3. 电路设计
- ↓
4. 物理设计

# 7.1 概述

## 7.1.4 数字系统的设计步骤

### 1. 系统设计



✘ 确定输出和输入之间的关系，找到实现数字系统的设计原理和方法。

✘ 划分系统的控制单元和受控单元，确定初始结构框图。

✘ 建立算法流程图，表示解决问题的步骤。

✘ 根据一定的规则将算法流程图转换成ASM图。

# 7.1 概述

## 7.1.4 数字系统的设计步骤

### 2、逻辑设计

当系统中各个子系统（指最低层子系统）或部件的逻辑功能和结构确定后，采用比较规范的形式来描述系统的逻辑功能。

#### ①数据处理器设计

建立操作明细表

#### ②控制器设计

建立状态转移表

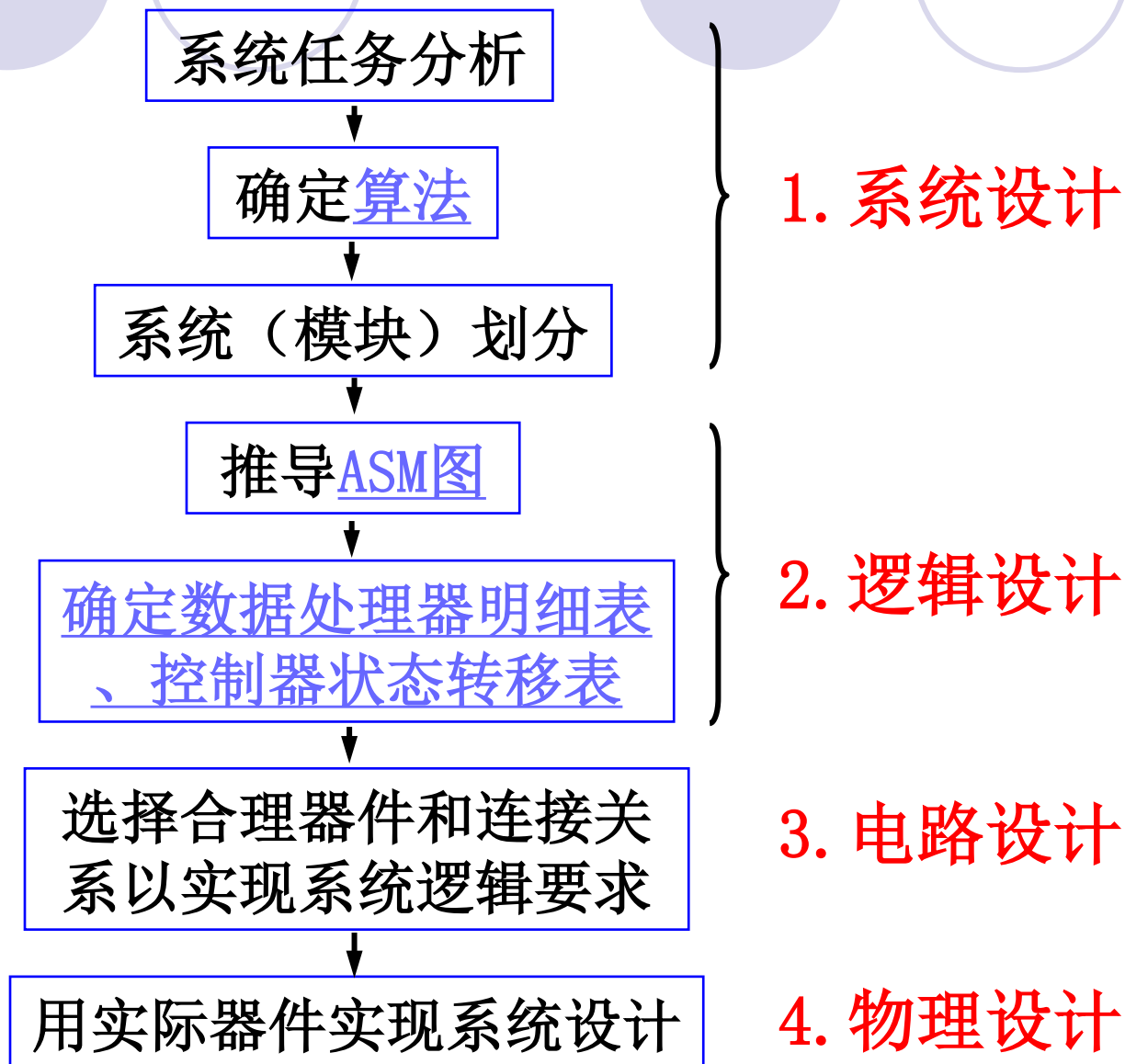
# 7.1 概述

## 7.1.4 数字系统的设计步骤

### 3、电路设计

选择合理的器件和连接关系，以实现系统逻辑要求。电路设计的结果常采用两种方式来表达：**电路图方式、硬件描述语言方式**。

# 7.1 概述



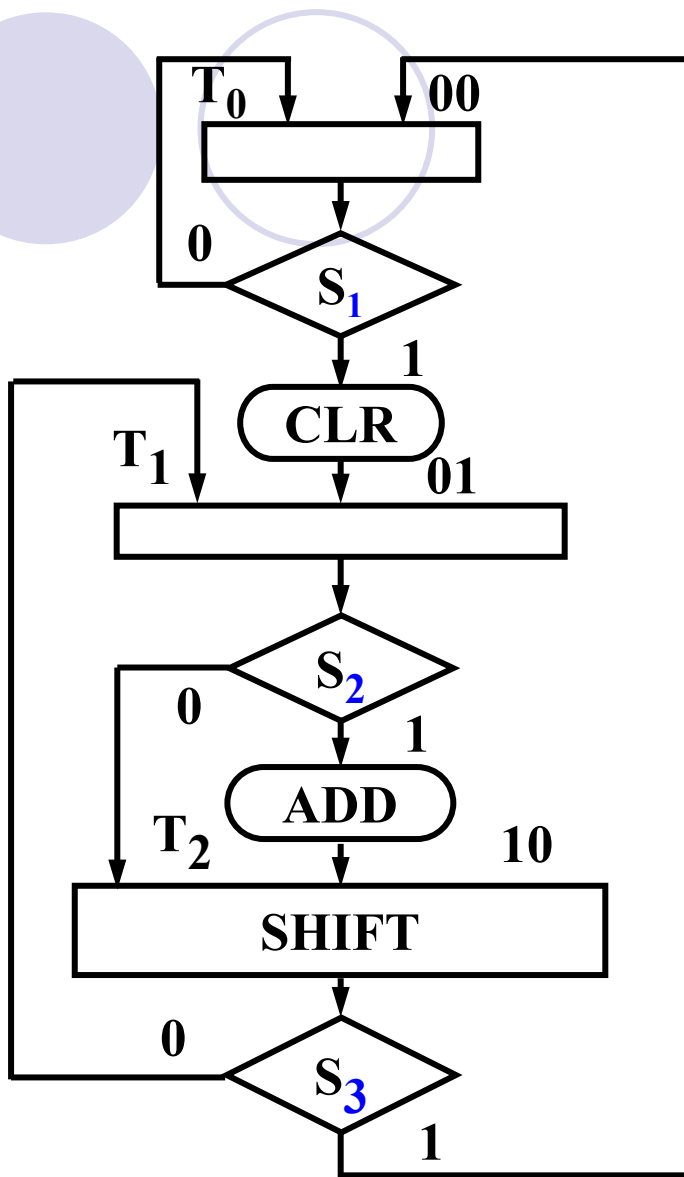
## 7.3 数字系统中控制器的设计

### 7.3.1 每态一个触发器的方法

控制器有多少状态就有多少触发器，每一个状态对应一个触发器，某一触发器出1表示进入该状态。

优点：

- a、无须分配状态。
- b、控制器的逻辑图易于读懂，调试维护方便，只要根据哪个触发器输出1，就知道进入哪个状态。
- c、不用列状态转移表，直接根据**ASM**图求得触发器得输入。



$$D_0 = T_0 \bar{S}_1 + T_2 S_3$$

$$D_1 = T_0 S_1 + T_2 \bar{S}_3$$

$$D_2 = T_1 \bar{S}_2 + T_1 S_2 = T_1$$

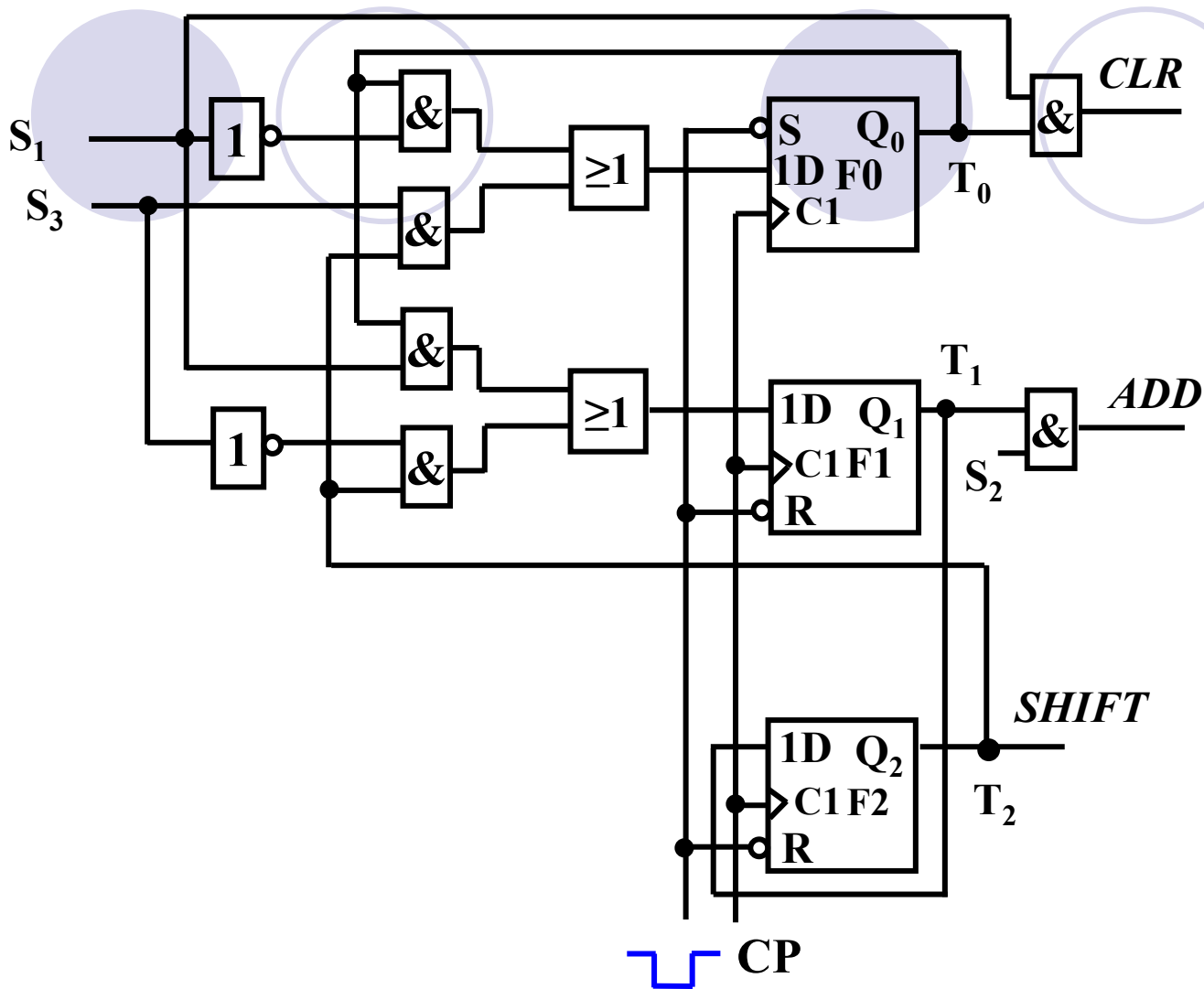
$$CLR = T_0 S_1$$

$$ADD = T_1 S_2$$

$$SHIFT = T_2$$

例 乘法器的ASM图



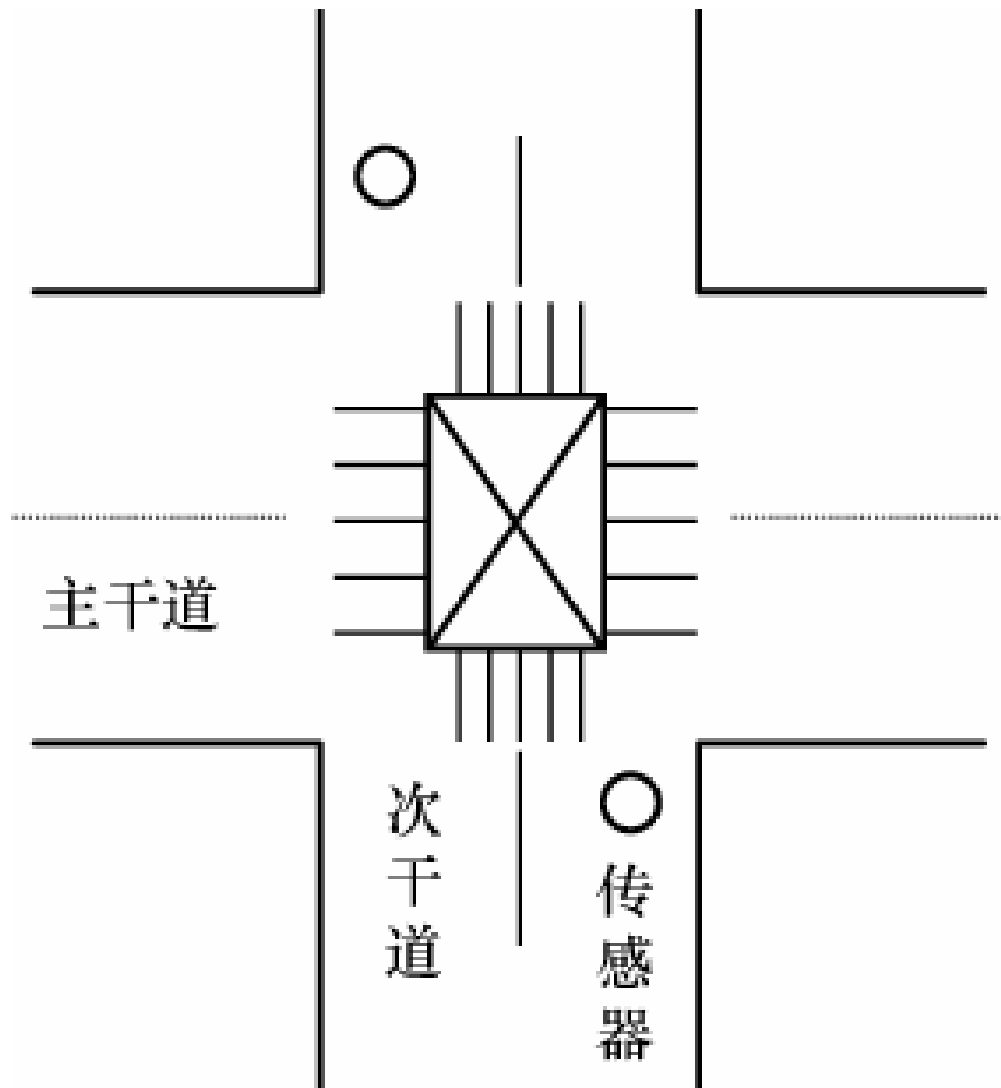


用每态一个触发器的乘法控制器逻辑图

## 7.4 数字系统设计举例

**例 7.4.2** 在主干道和次干道的十字交叉路口，设置交通灯管理系统，管理车辆运行。其示意图如图7.4.7所示，在次干道设置传感器，当次干道上有车时，传感器（sensor）输出 $SEN=1$ 。主干道车辆通车有优先权，次干道无车时始终保持主干道车辆畅通，主干道绿灯亮，次干道红灯亮。当次干道上有车时，系统开始计时，当主干道通车时间达到16s时，主干道交通灯由绿经黄变红，次干道交通灯由红变绿。若次干道继续有车要求通行时，其绿灯可以继续亮，但最长时间被限定为16s；若次干道已无车辆或有车辆但16s计时到，则次干道交通灯由绿经黄变红，主干道交通灯由红变绿。黄灯亮的时间设定为4s。

## 7.4 数字系统设计举例



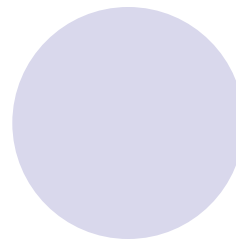
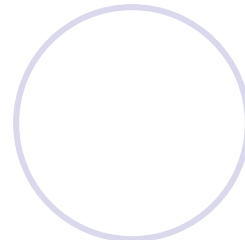
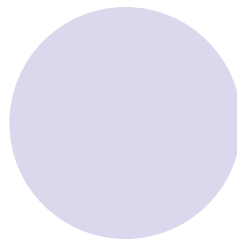
# 一、系统设计

## 1. 系统任务分析

## 2. 确定初始结构图

### (1) 结构图

## 3. 建立算法流程图



## 二、逻辑设计

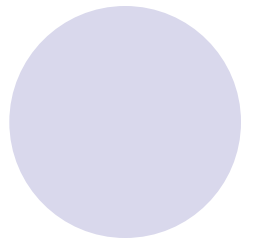
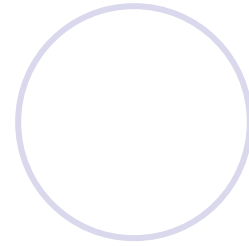
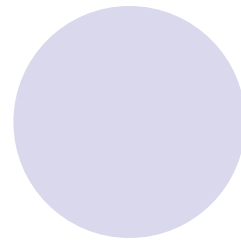
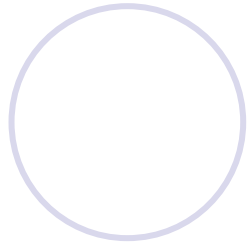
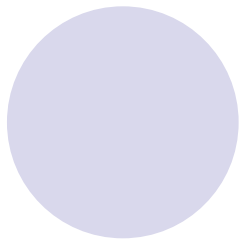
### 1. 导出ASM图

### 2. 确定数据处理器的明细表

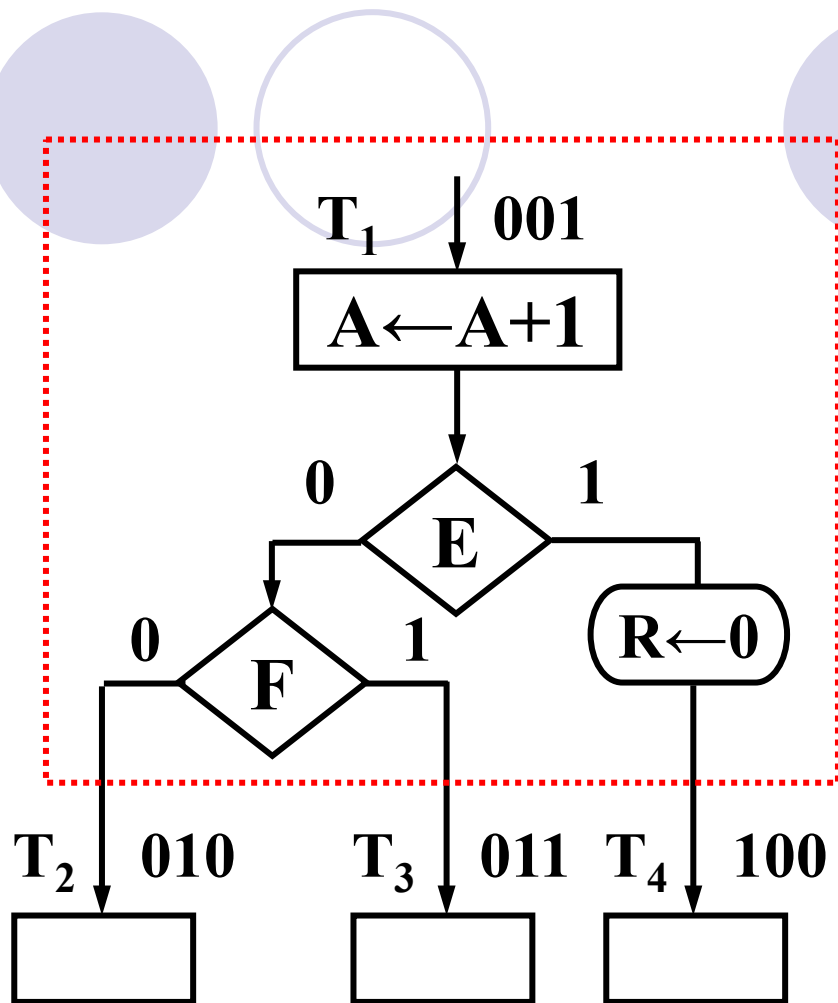
数据处理器设计。

### 3. 确定控制器的转换表

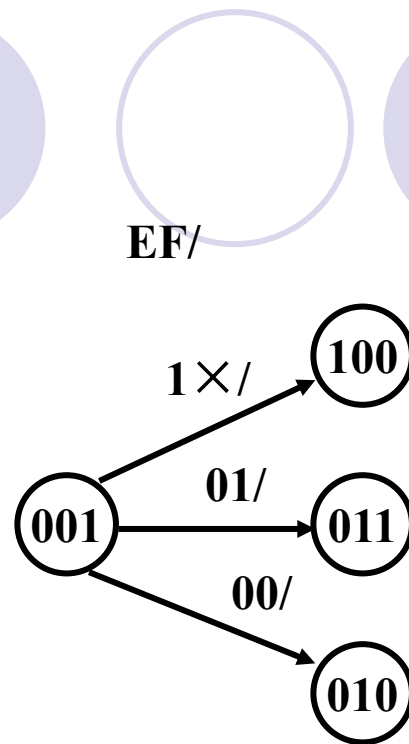
采用每态一个触发器的方法，可直接利用ASM图，避免建立转换表，简化设计过程。



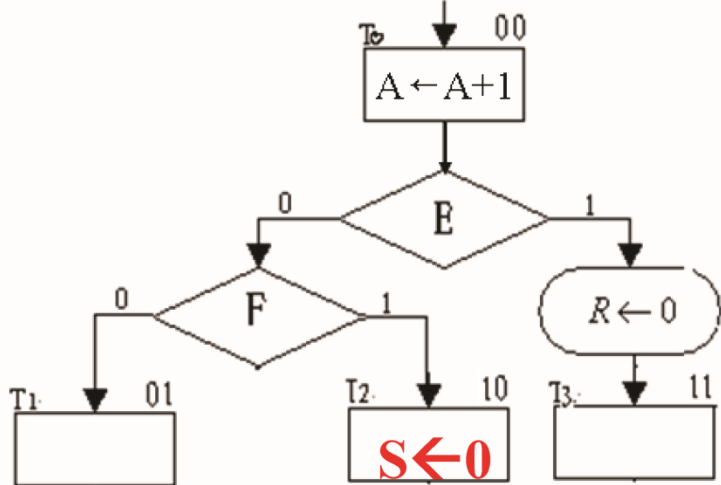
# 作业



ASM块

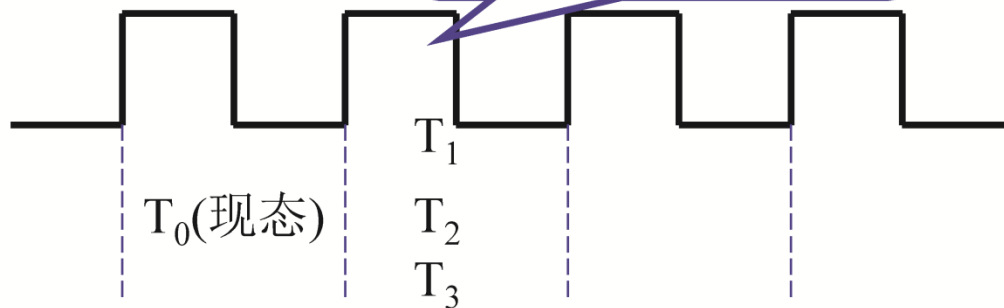


等效状态转移图

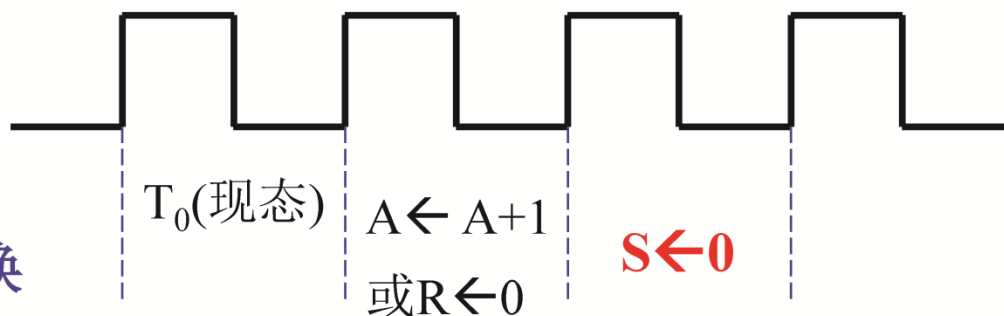


根据EF不同，  
选择不同状态

控制器的时间表



处理器的时间表



控制器：进行状态转换

处理器：进行数据处理

现态  $T_0$  与状态框内的操作不在同一个CLK内



## 7.2 数字系统的描述工具

### 7.2.3 算法流程图

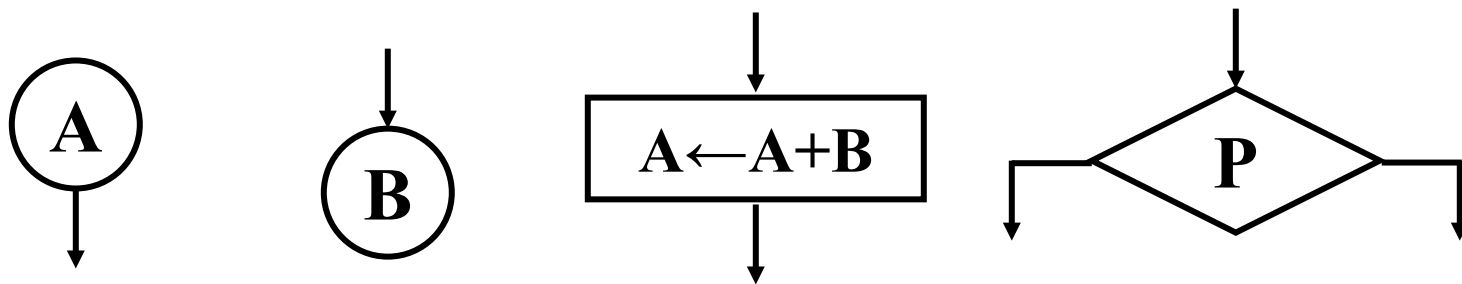
#### 1. 作用

描述算法。

**注意：按照事件的先后次序排列的，与电路的时序无对应关系。**

#### 2. 基本符号

入口点；出口点；传输框；判断框



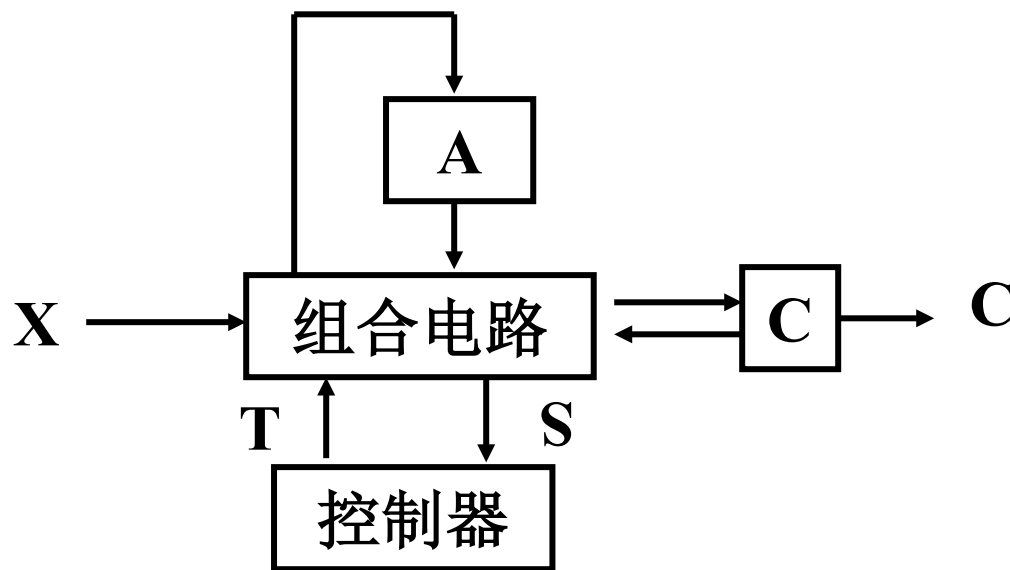
(a) 入口点 (b) 出口点 (c) 传输框 (d) 判断框

图7.2.10 流程图符号

## 7.2 数字系统的描述工具

### 7.2.3 算法流程图

例 7.2.2 绝对值计算，计算  $Z = |X_1| + |X_2| + |X_3|$ 。



系统结构图

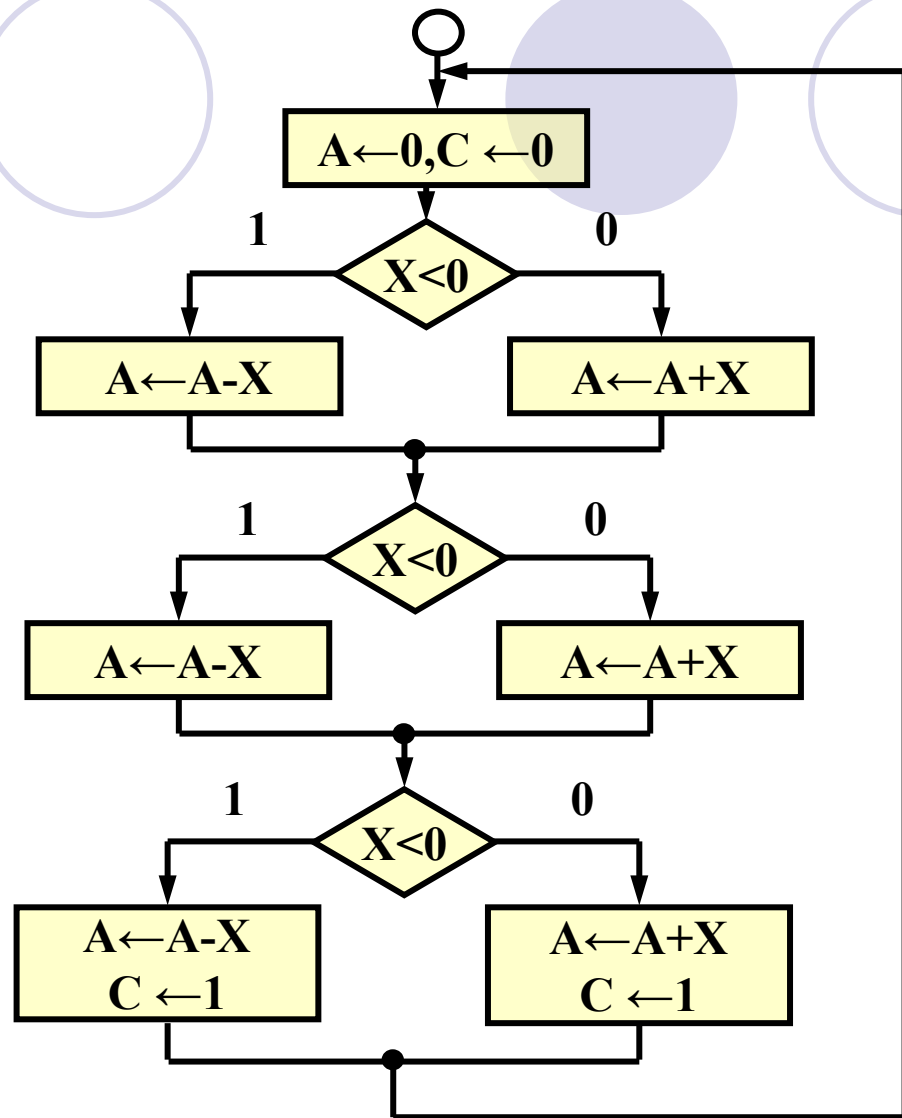


图7.2.12 算法流程图

## 7.2 数字系统的描述工具

### 7.2.1 寄存器传输语言（RTL）

#### 1. 寄存器传输操作

所存信息的处理和存贮

#### 2. 寄存器传输语言

既表示了寄存器传输操作，又和硬件间有个简单的对应关系的一种方便的设计工具。

#### 3. 寄存器具有广义的概念

既包括暂存信息的寄存器，也包括移位寄存器、计数器、存储器等

## 7.2 数字系统的描述工具

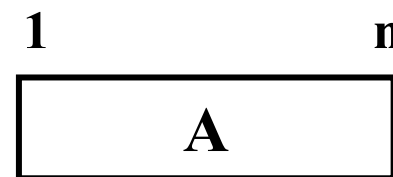
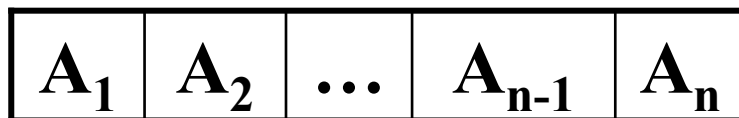
### 7.2.1 寄存器传输语言 (RTL)

#### 一、寄存器间的信息传输

##### 1. 寄存器的表示方法

①大写英文字母

②方块图



(a)寄存器A (b)寄存器A的各个位表示 (c)寄存器位编号表示

图7.2.1 寄存器方块图表示

## 7.2 数字系统的描述工具

### 7.2.1 寄存器传输语言 (RTL)

#### 一、寄存器间的信息传输

2. 传输操作  $\bar{X} \cdot T_1 : A \leftarrow B$

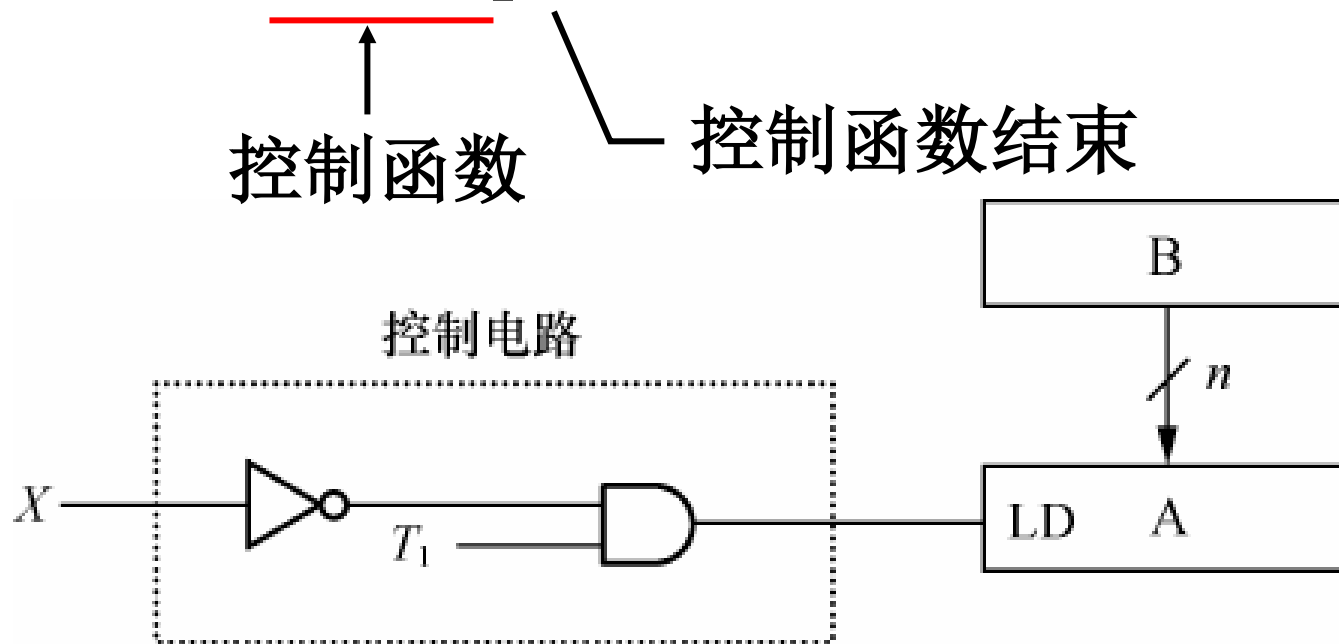


图 7.2.2 实现语句  $\bar{X} \cdot T_1 : A \leftarrow B$  的逻辑图

## 7.2 数字系统的描述工具

### 7.2.1 寄存器传输语言（RTL）

#### 二、算术操作

$T_2 : A \leftarrow A+B$

$T_5 : A \leftarrow A+1$

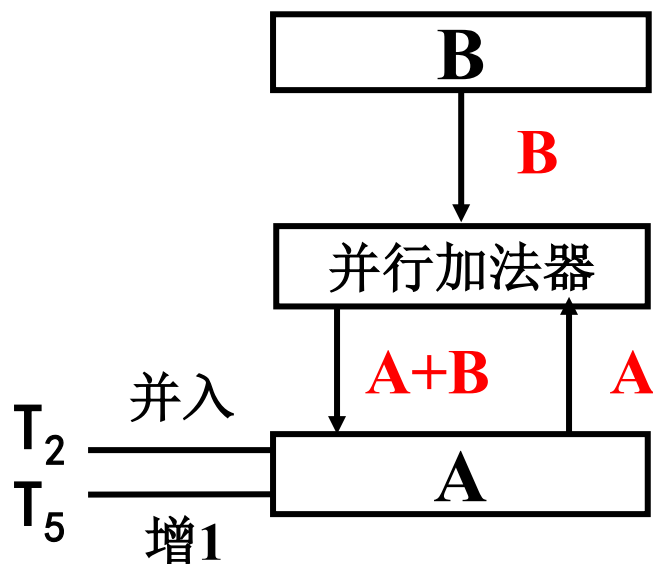


图 7.2.5 完成加和增“1”操作的方框图

## 7.2 数字系统的描述工具

### 7.2.1 寄存器传输语言 (RTL)

#### 三、逻辑操作

与运算符 “ $\wedge$ ” ； 或运算符 “ $\vee$ ”

——为了与算术运算的符号 “ $.$ ”、“ $+$ ”区别。

$T_1+T_2$ :  $A \leftarrow A+B, \quad C \leftarrow D \vee F$

---



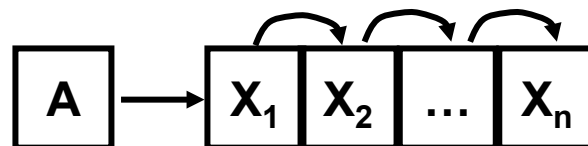
两个操作同时实现（并行关系）



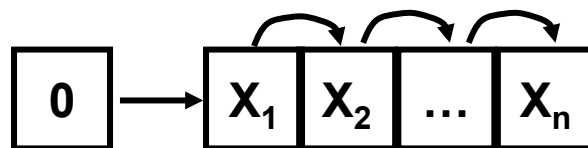
## 7.2 数字系统的描述工具

### 7.2.1 寄存器传输语言 (RTL)

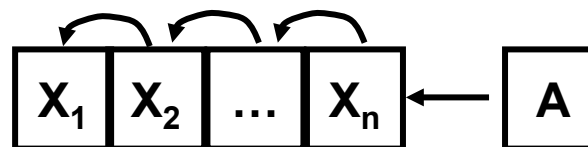
#### 四、移位操作 1. 右移操作: $X \leftarrow SR(A, X)$



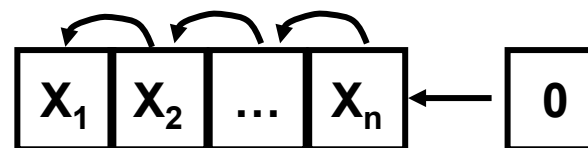
$X \leftarrow SR(X)$



#### 2. 左移操作: $X \leftarrow SL(X, A)$



$X \leftarrow SL(X)$



## 7.2 数字系统的描述工具

### 7.2.1 寄存器传输语言 (RTL)

#### 五、条件控制语句

**P: IF (条件) Then (微操作1) Else (微操作2)**

←  
控制函数

例:

**T2: IF (C=0) THEN (F←1) ELSE (F←0)**

可以写成两个一般语句:

**$\overline{C}$ ·T2: F←1**

**C·T2: F←0。**



## 小结

一条**RTL** 语句：描述数字系统所处的一个状态。

其操作：说明数据处理器要实现的操作。

控制函数：说明控制器发出的命令。

## 7.2 数字系统的描述工具

### 7.2.4 算法状态机图

#### 1. 作用

按系统时序来描述系统的工作过程。

#### 2. ASM图符号

(1)状态框

(2)判断框

(3)条件框(ASM图特有)

#### 3. ASM块

#### 4. ASM图的建立

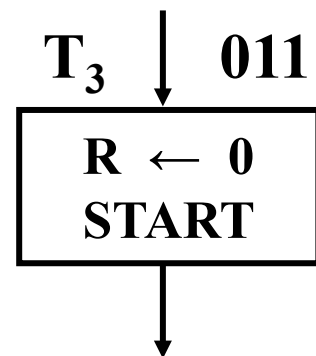
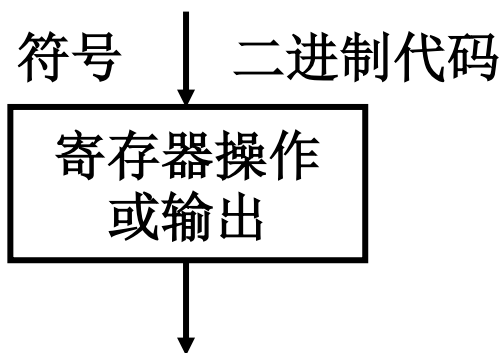
#### 5. 由ASM图推导处理器明细表 and 控制器状态转移图

## 7.2 数字系统的描述工具

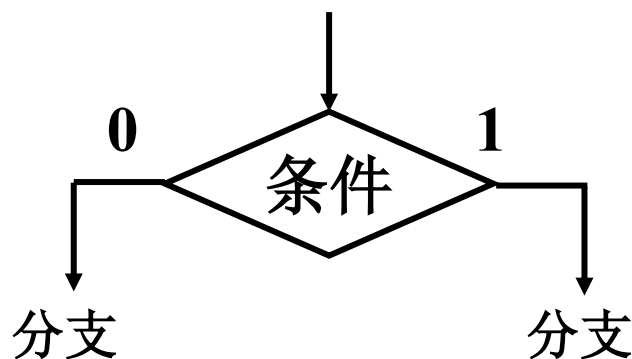
### 7.2.4 算法状态机图

#### 2.ASM图符号

状态框



判断框

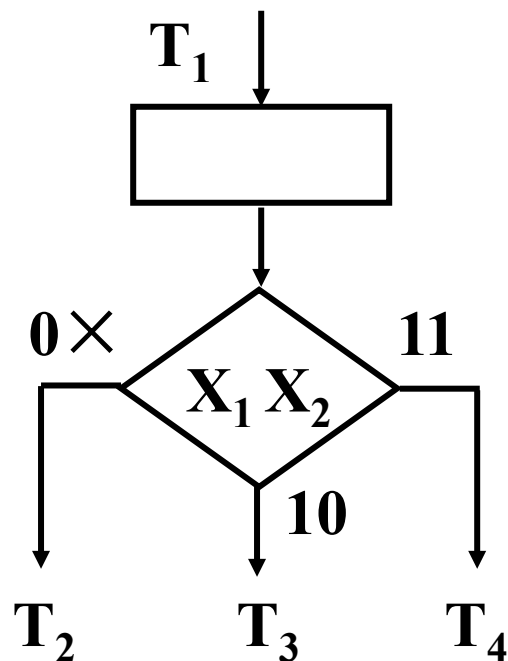


## 7.2 数字系统的描述工具

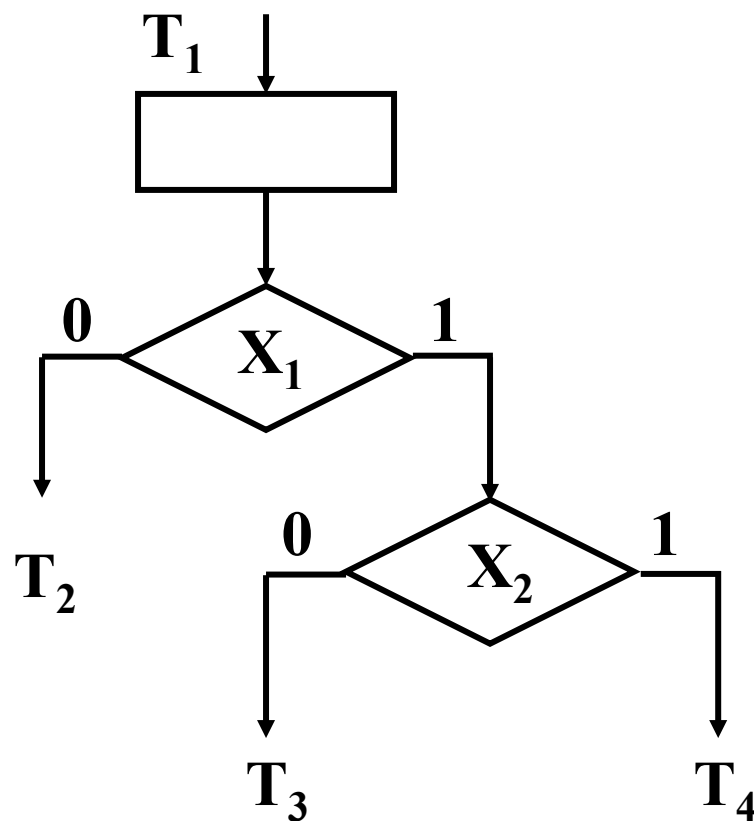
### 7.2.4 算法状态机图

#### 2.ASM图符号

判断框



(a) 真值表图解分支表示



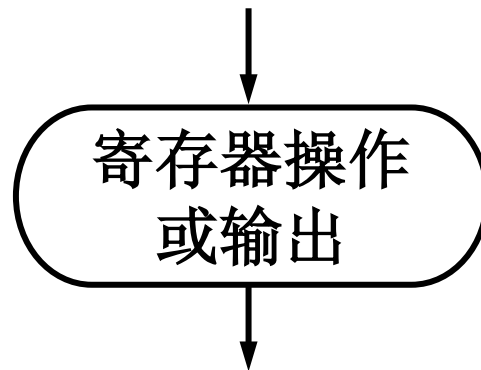
(b) 变量优先级分支表示 38

## 7.2 数字系统的描述工具

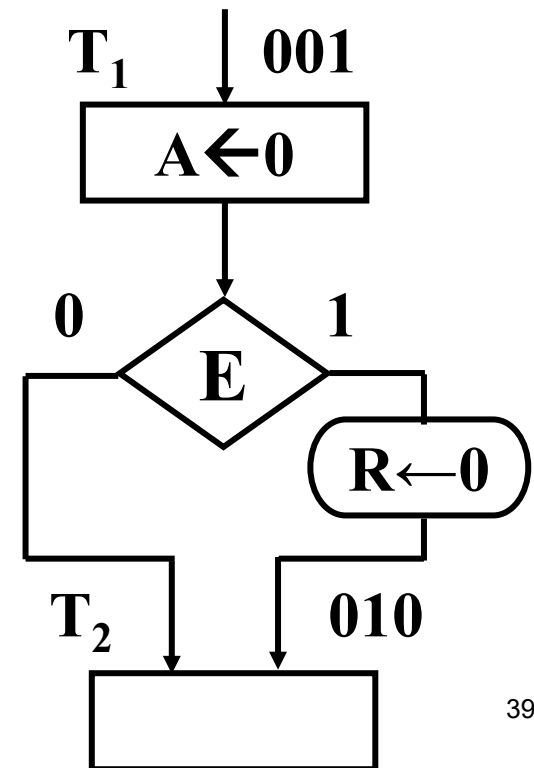
### 7.2.4 算法状态机图

#### 2.ASM图符号

条件框



条件框为ASM图所特有的，条件框内的操作和输出是在给定条件下，判断条件被满足时才发生的，所以条件框的输入必定与判断框的分支相连。



## 7.2 数字系统的描述工具

### 7.2.4 算法状态机图

#### 3. ASM块

- (1) 包含且仅包含一个状态框（可能包含若干其他框）；
- (2) 表示一个时钟周期内系统的状态；

#### (3) ASM图与状态转移图；

状态转移图无法表示**操作**和**输出变量**  
与**输入变量**的函数关系。

- (4) 各种逻辑框之间的时间关系（见例题）



## 7.2 数字系统的描述工具

### 7.2.4 算法状态机图

#### 4.ASM图的建立

从算法流程图→ASM图

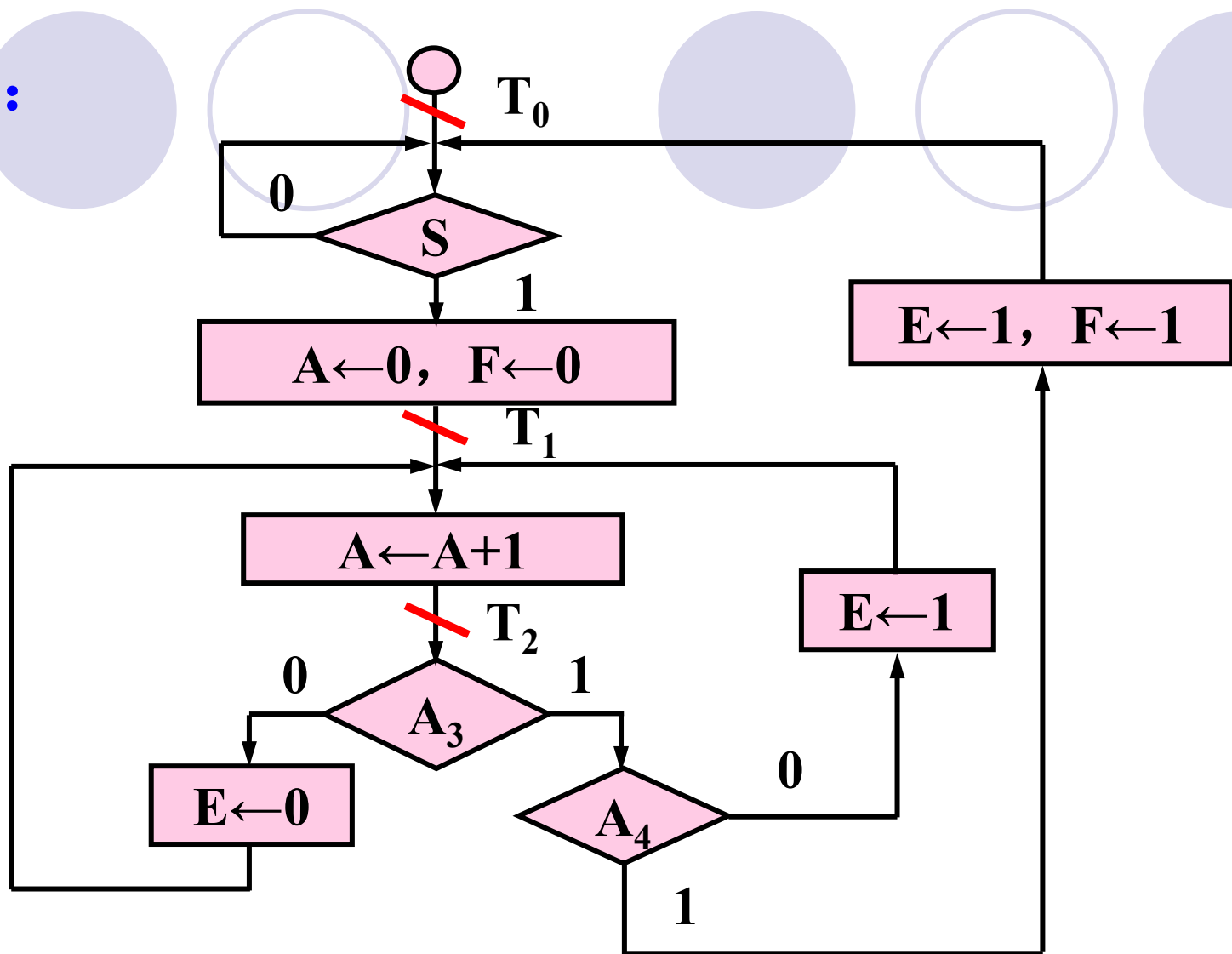
原则1：在算法的起始点安排一个状态；

原则2：必须用状态来分开不能同时实现的寄存器传输操作；

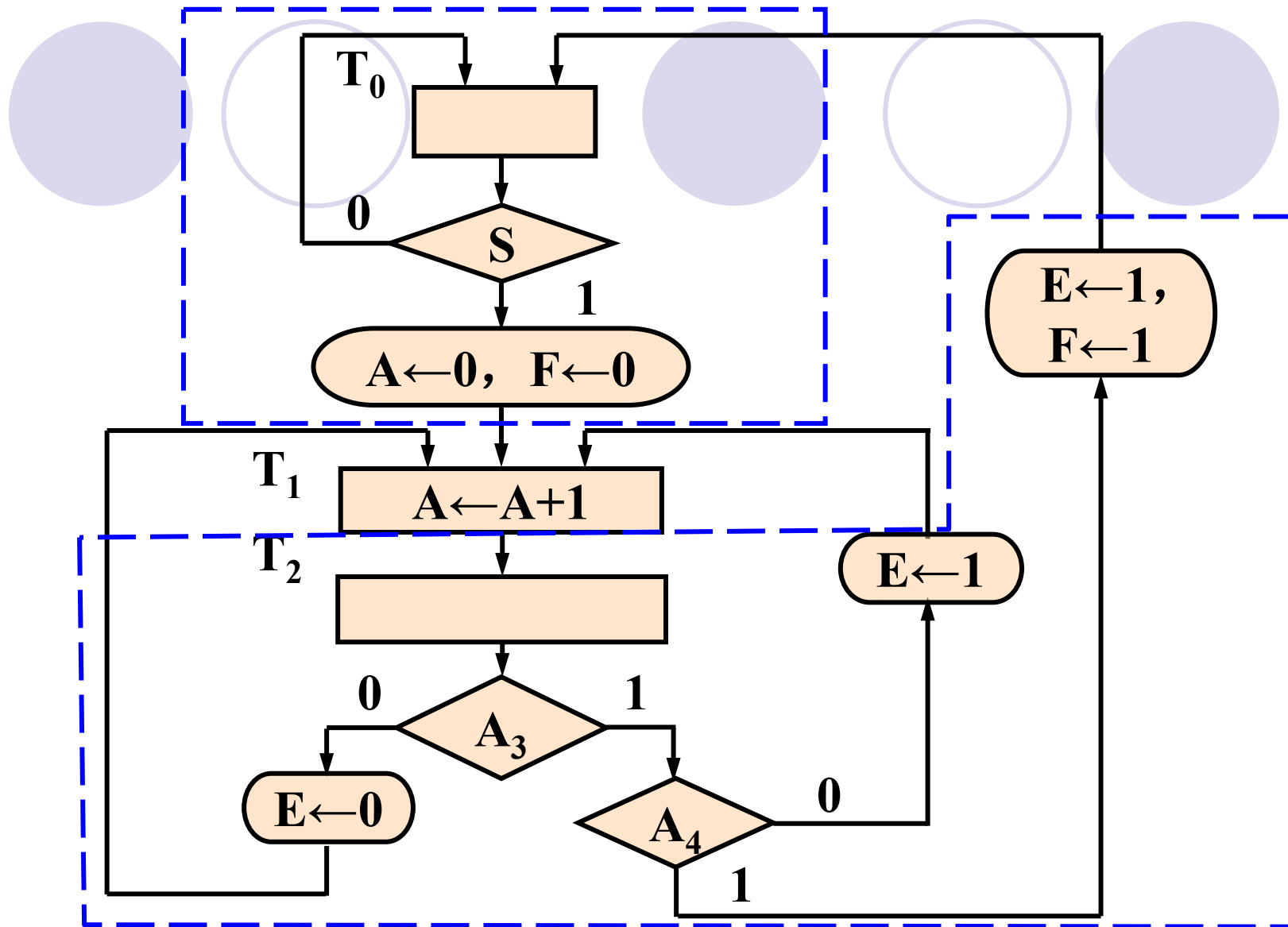
原则3：判断如果受寄存器操作的影响，应在它们之间安排一个状态。

补充原则4：使状态框尽量少。

例:



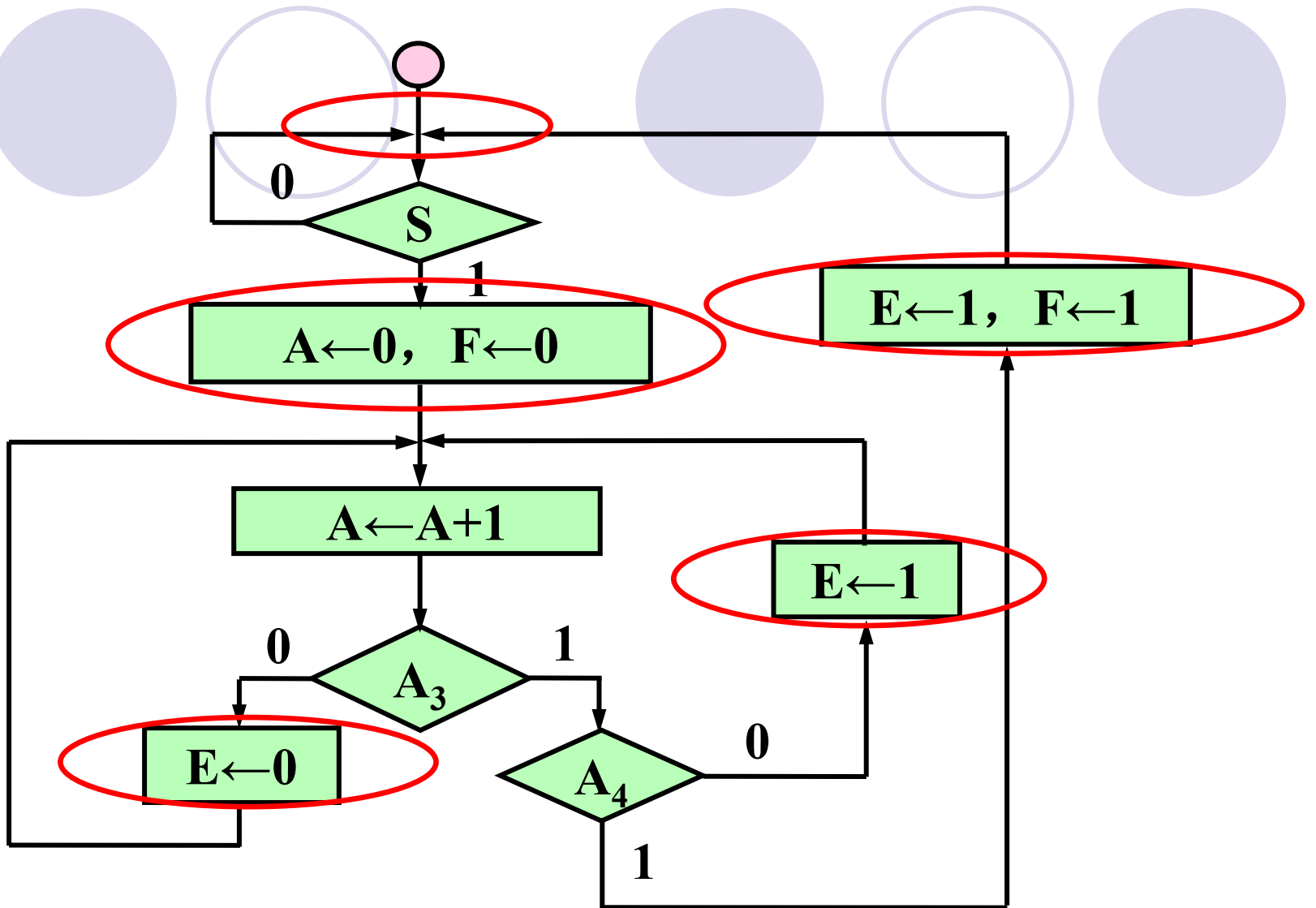
算法流程图



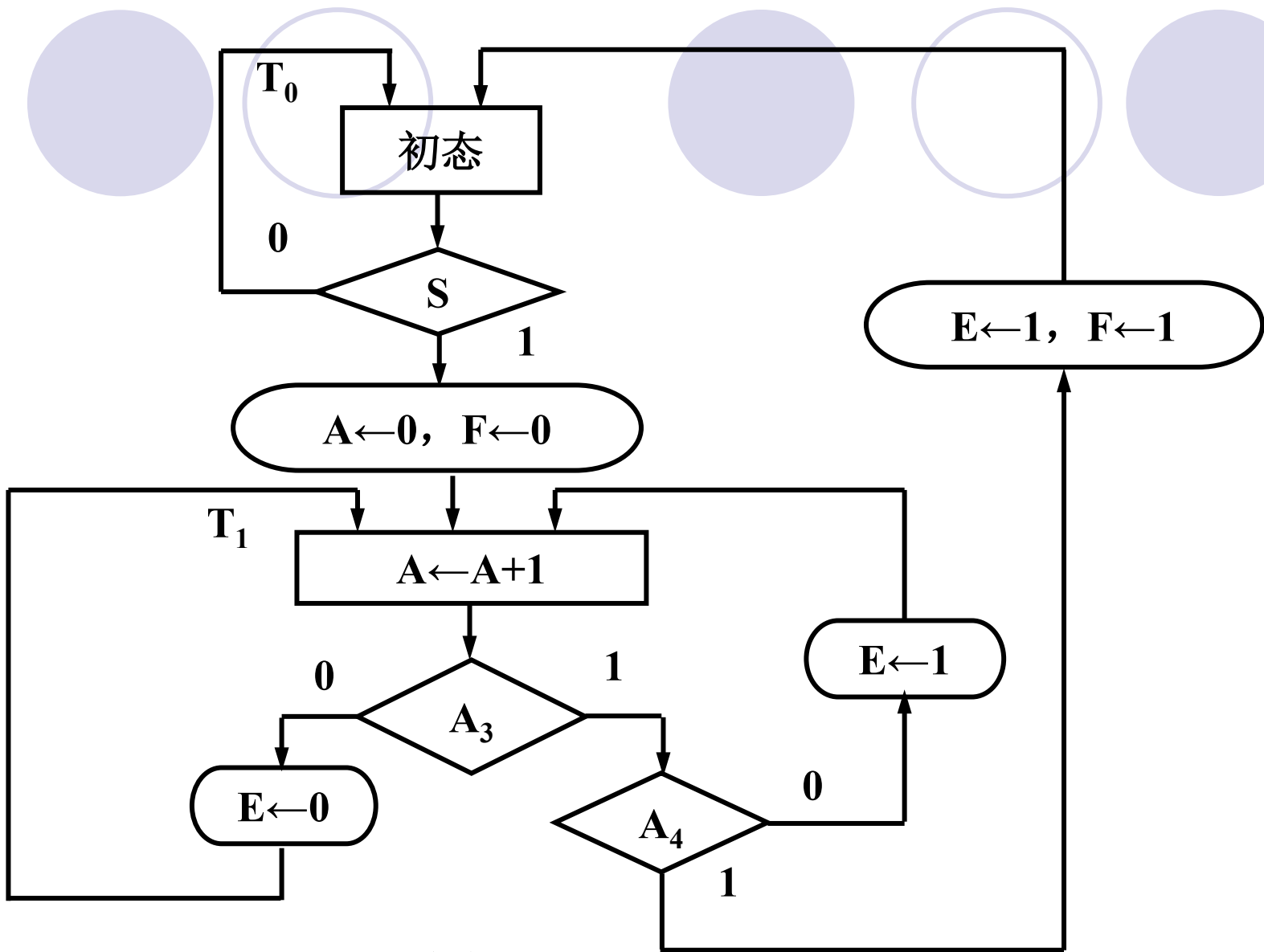
ASM图

**例：**一个数字系统的数据处理器有2个触发器E和F及1个二进制计数器A，计数器的各个位分别用 $A_4$ 、 $A_3$ 、 $A_2$ 、 $A_1$ 标记， $A_4$ 为最高位， $A_1$ 为最低位。启动信号S使计数器A和触发器F清“0”，从下一个时钟脉冲开始，计数器增1，一直到系统停止工作为止。系统的操作序列由 $A_3$ 和 $A_4$ 之值决定，即：

- ① $A_3=0$ ，触发器E清“0”，并继续计数。
- ② $A_3=1$ ，触发器E置“1”，并检验 $A_4$ ，若 $A_4=0$ ，继续计数；若 $A_4=1$ ，触发器F置“1”，系统停止计数。



算法流程图



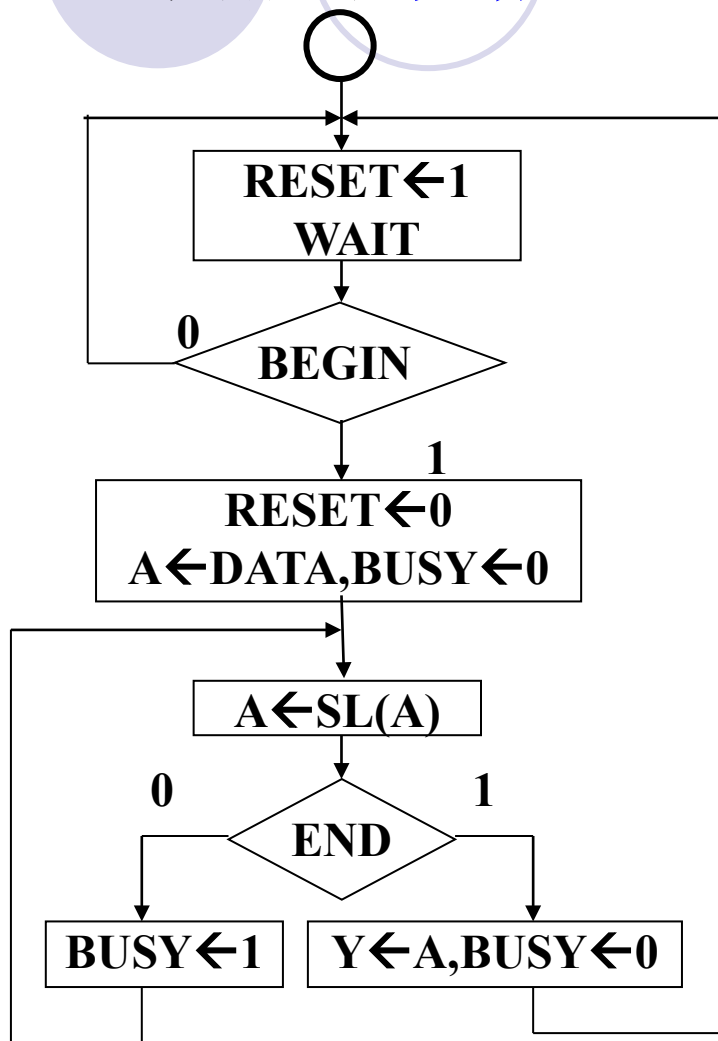
ASM图

A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	E	F	条 件	状 态
----------------	----------------	----------------	----------------	---	---	-----	-----

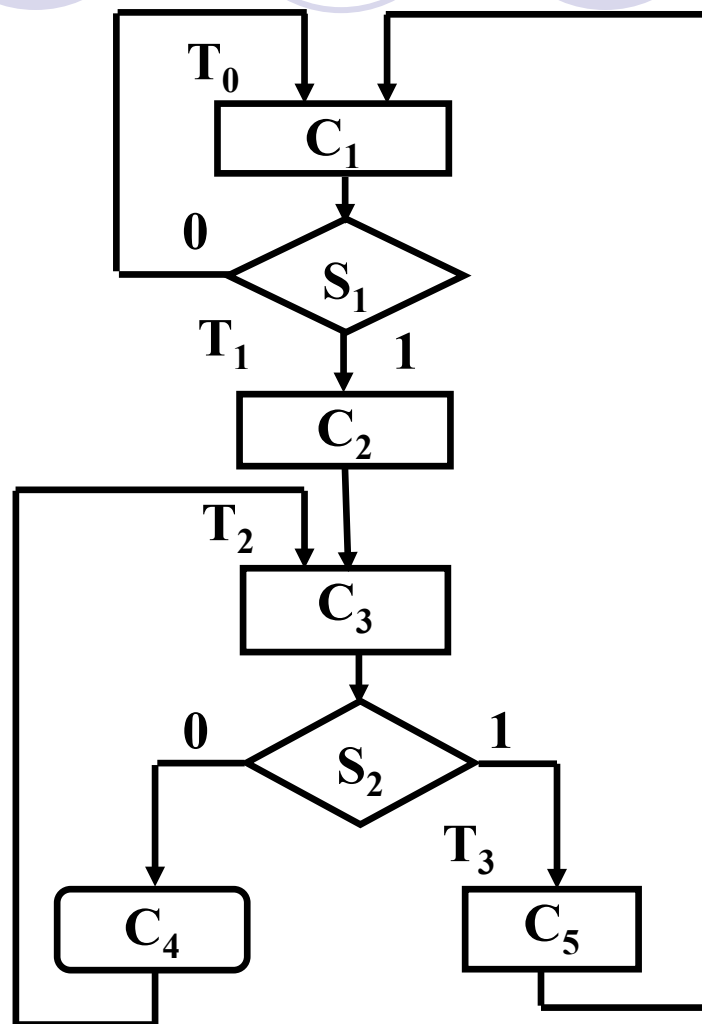
[illegible]

## 7.2.4 算法状态机图

### 5、ASM图推导数据处理器明细表和控制器状态转移图



(a) 算法流程图

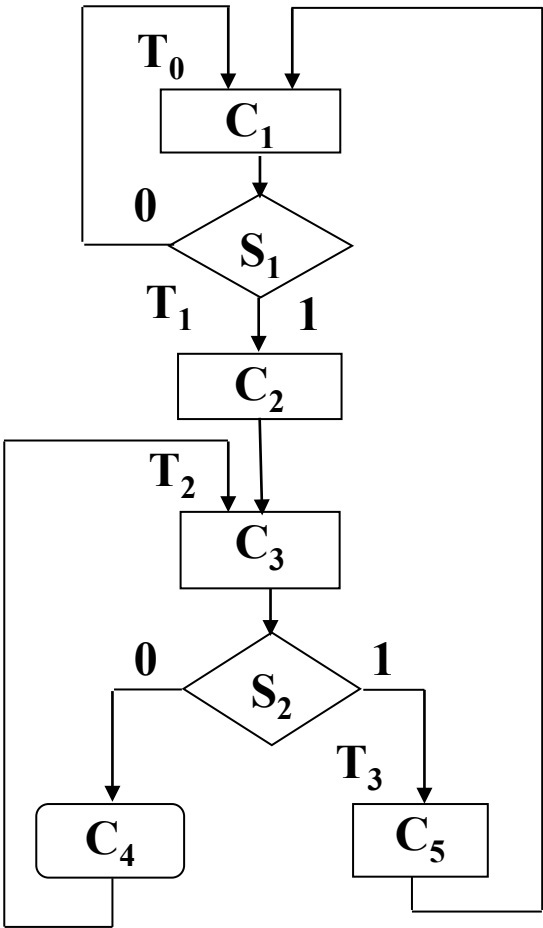


(b) ASM图



# 7.2.4 算法状态机图

## 5、ASM图推导[数据处理器明细表](#)和控制器状态转移图



(b) ASM图

### 数据处理器明细表

操 作 表		状态变量表	
控制信号	<u>操 作</u>	状态变量	定 义
C <sub>1</sub> C <sub>2</sub>  C <sub>3</sub> C <sub>4</sub> C <sub>5</sub>	<b>RESET←1 WAIT</b> <b>RESET←0</b> <b>A←DATA,BUSY←0</b> <b>A←SR(A)</b> <b>BUSY←1</b> <b>Y←A,BUSY←0</b>	S <sub>1</sub> S <sub>2</sub>	<b>BEGIN</b> <b>END</b>

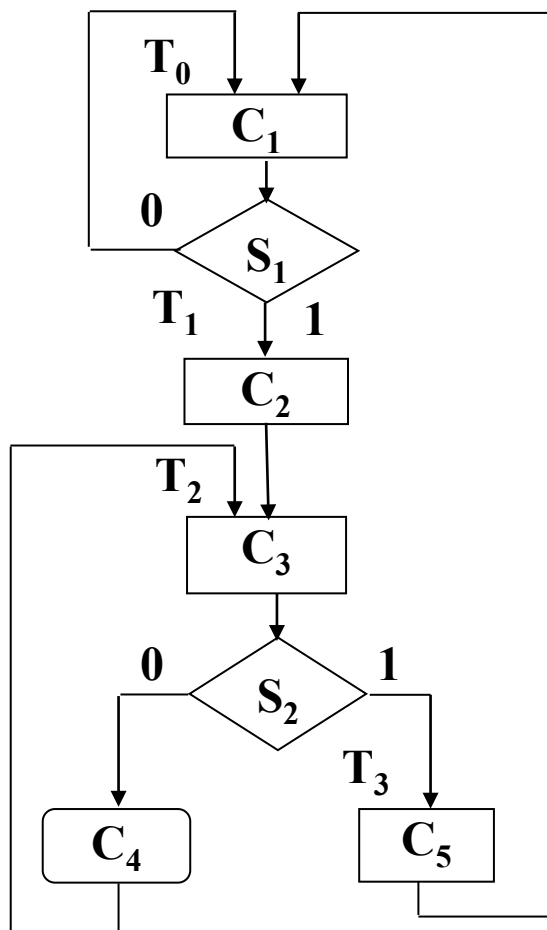
## 7.2.4 算法状态机图

### 5、ASM图推导数据处理器明细表和控制器状态转移图

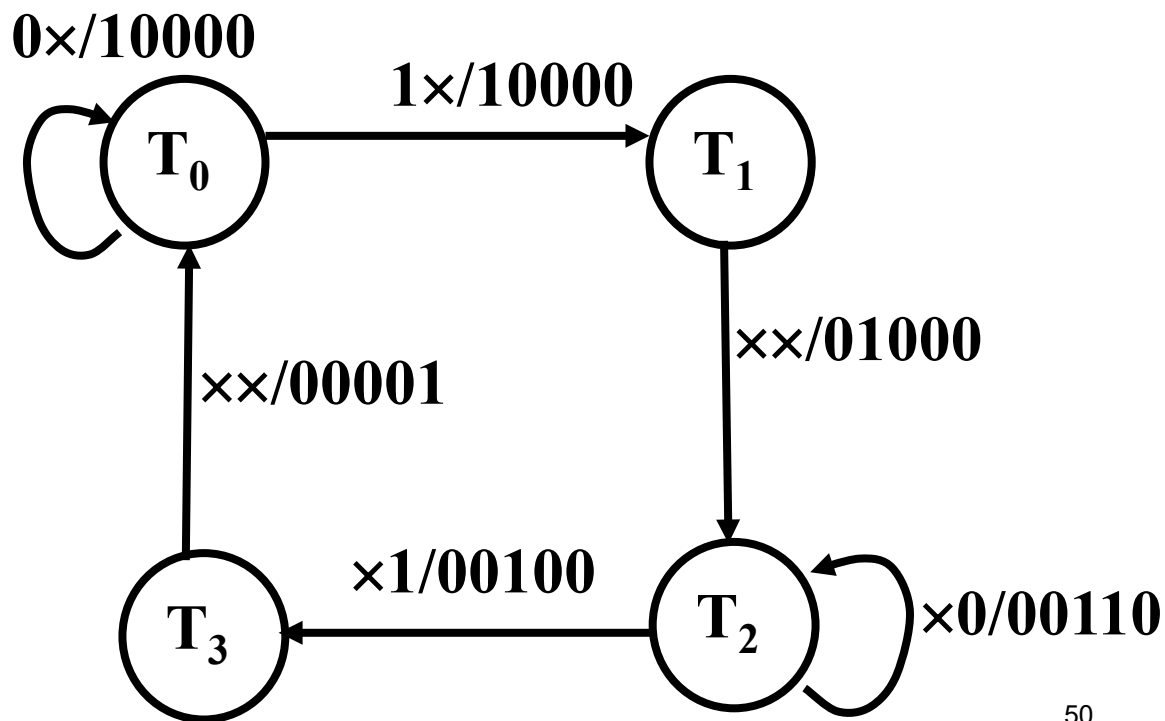
#### 控制器的状态转移图

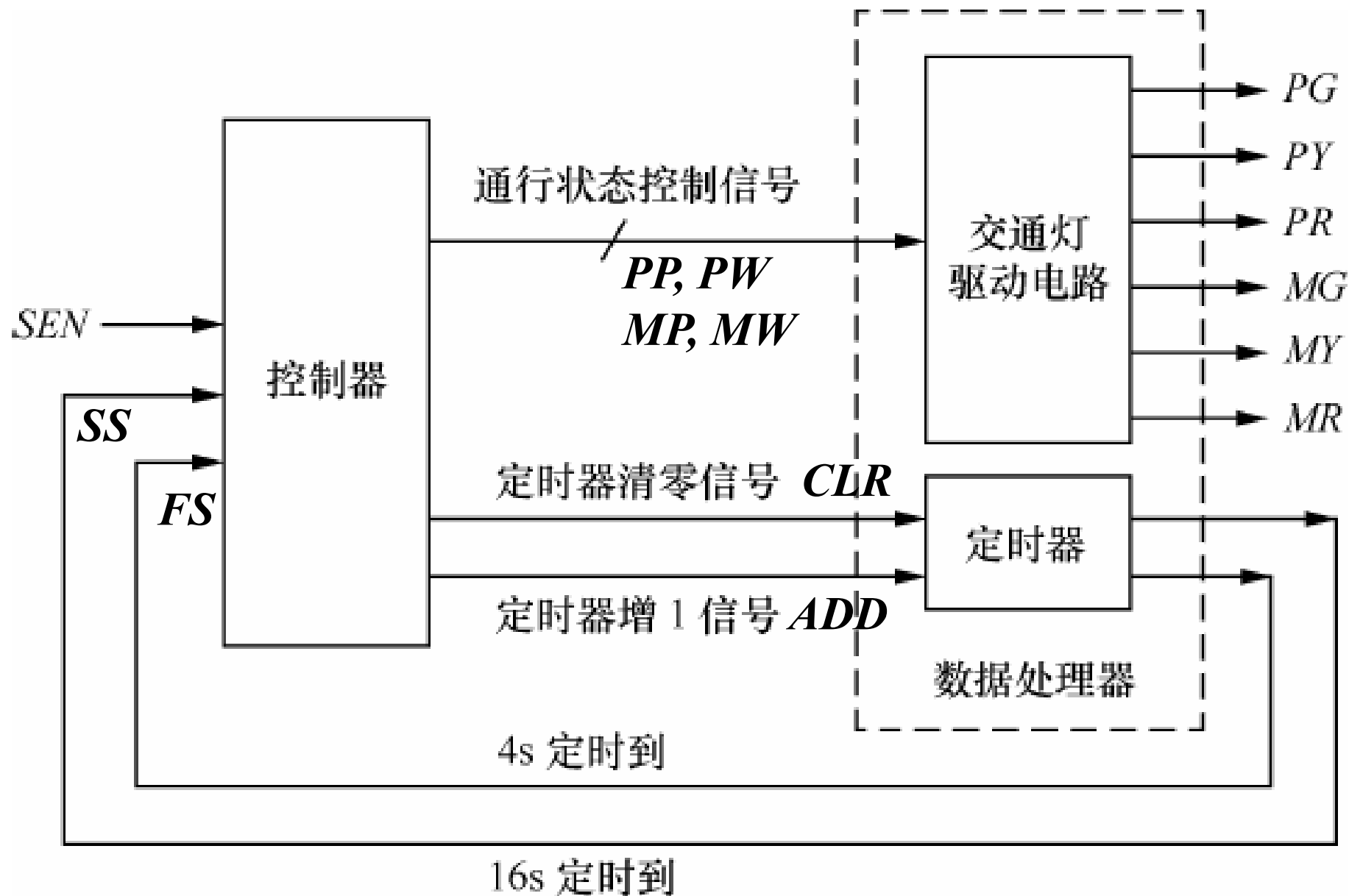
输入/输出:

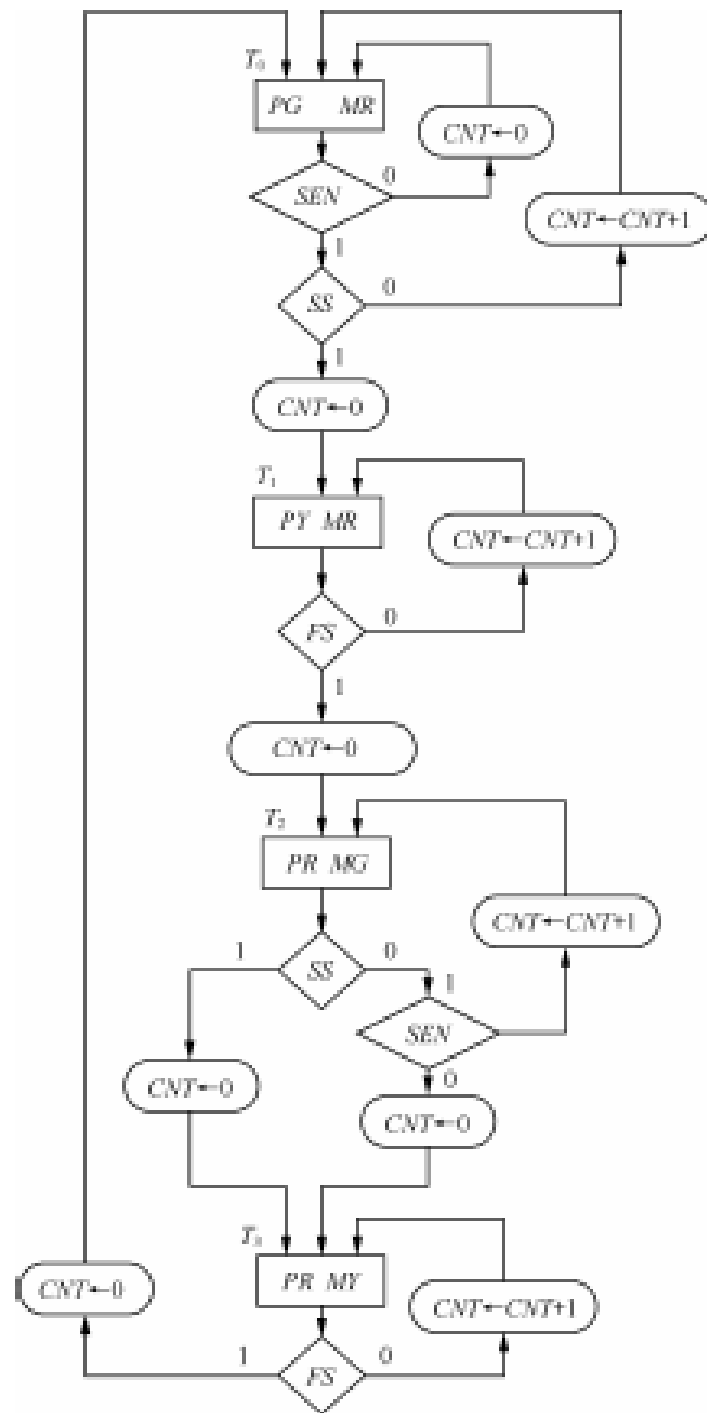
$S_1S_2/C_1C_2C_3C_4C_5$



(b) ASM图







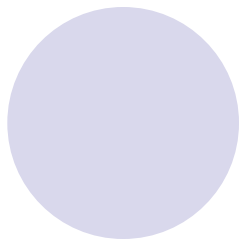
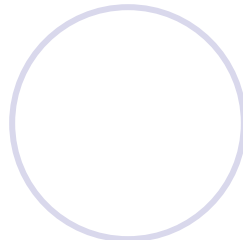
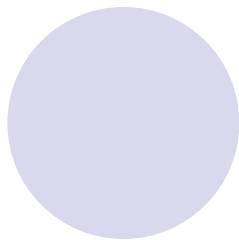
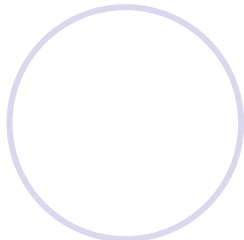
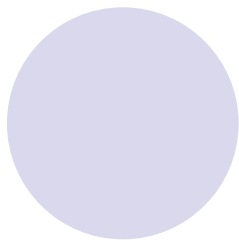


表 7.4.8

交通灯管理系统中数据处理器明细表

操作表		变量表	
控制信号	操作	变量	定义
<i>PP</i>	<i>PG=1,MR=1</i>	<i>SS</i>	<i>CNT=15</i>
<i>PW</i>	<i>PY=1,MR=1</i>	<i>FS</i>	<i>CNT= 3</i>
<i>MP</i>	<i>PR=1,MG=1</i>		
<i>MW</i>	<i>PR=1,MY=1</i>		
<i>CLR</i>	<i>CNT←0</i>		
<i>ADD</i>	<i>CNT←CNT+1</i>		

表 7.4.9

交通灯驱动电路真值表

输入	输出					
	$PG$	$PY$	$PR$	$MG$	$MY$	$MR$
$PP$	1	0	0	0	0	1
$PW$	0	1	0	0	0	1
$MP$	0	0	1	1	0	0
$MW$	0	0	1	0	1	0

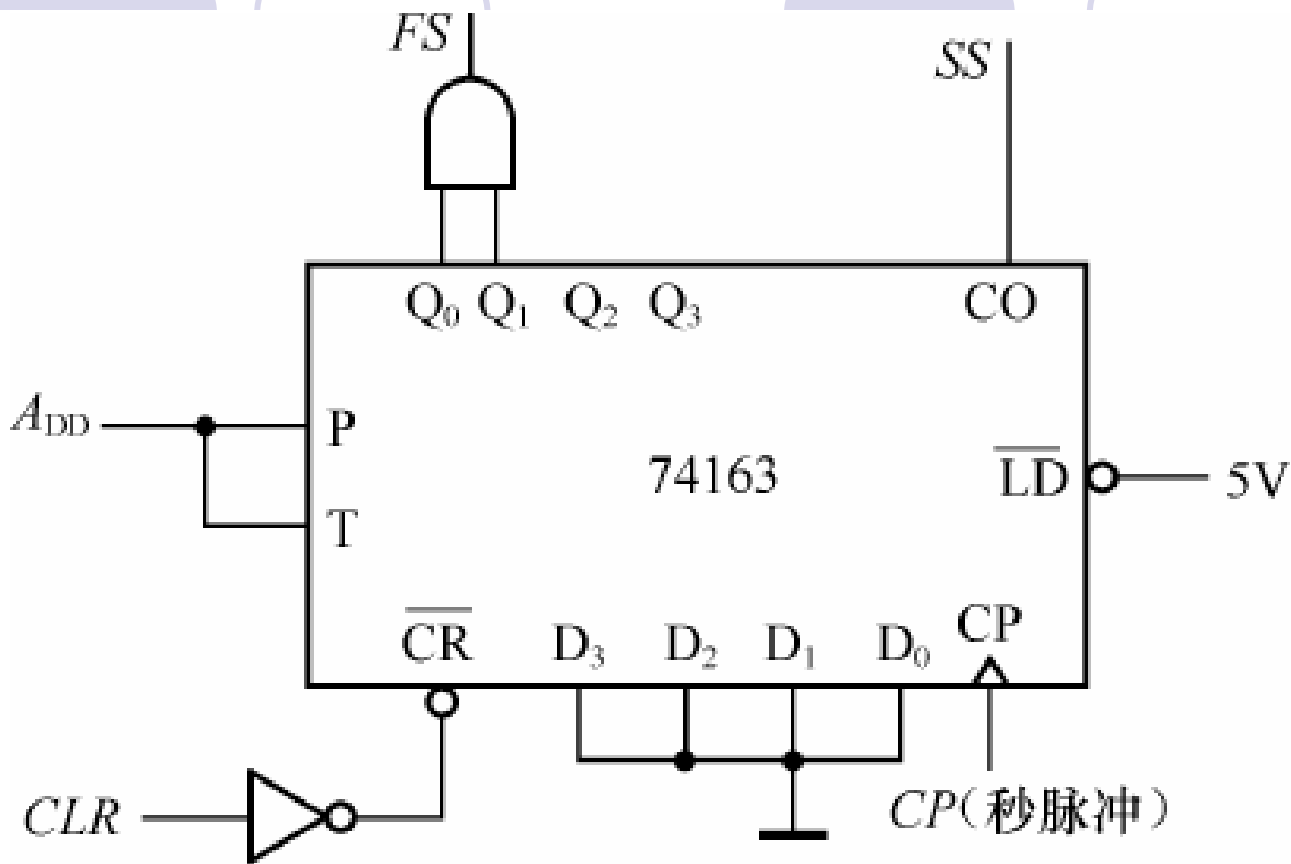


图 7.4.10 定时器逻辑图

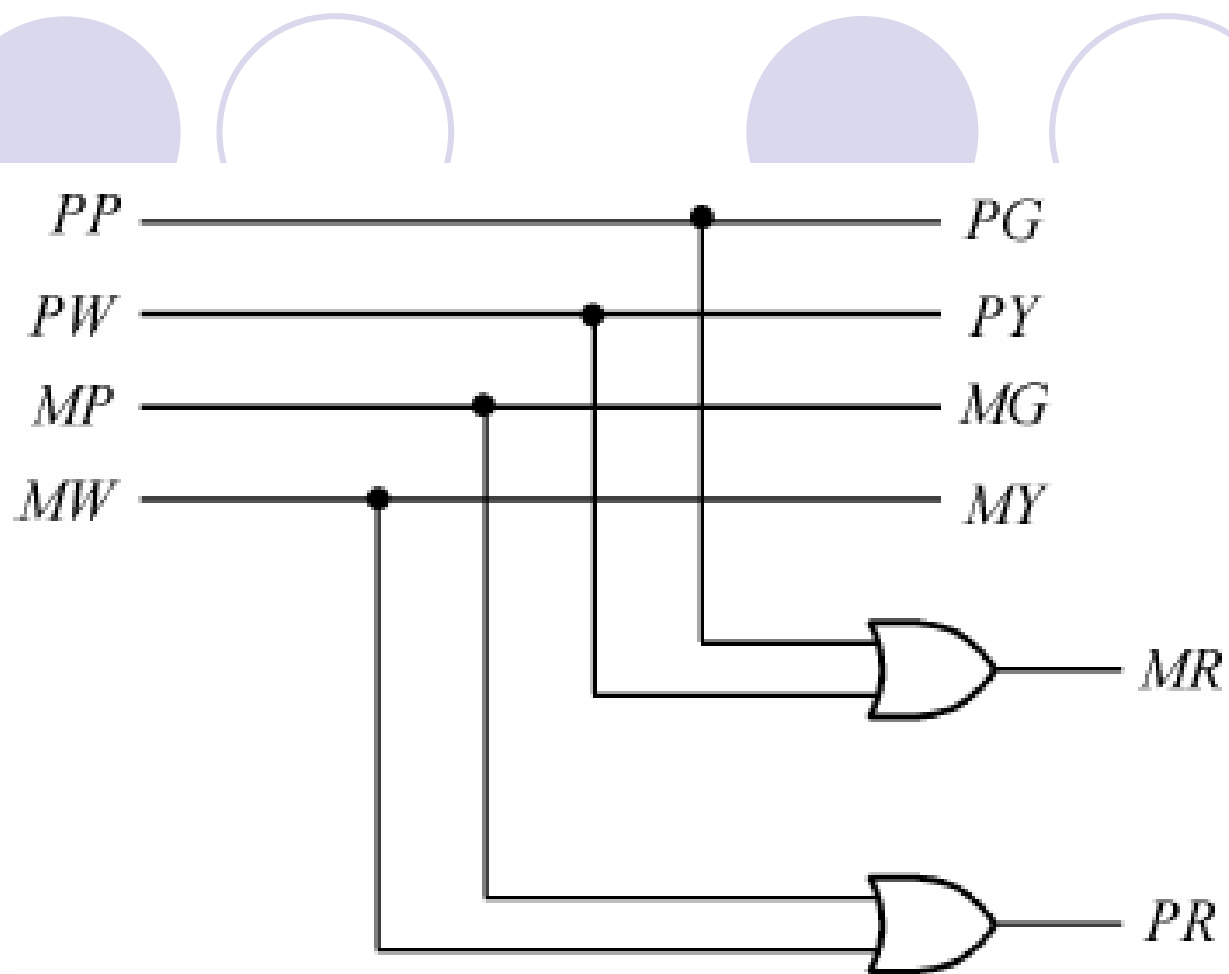
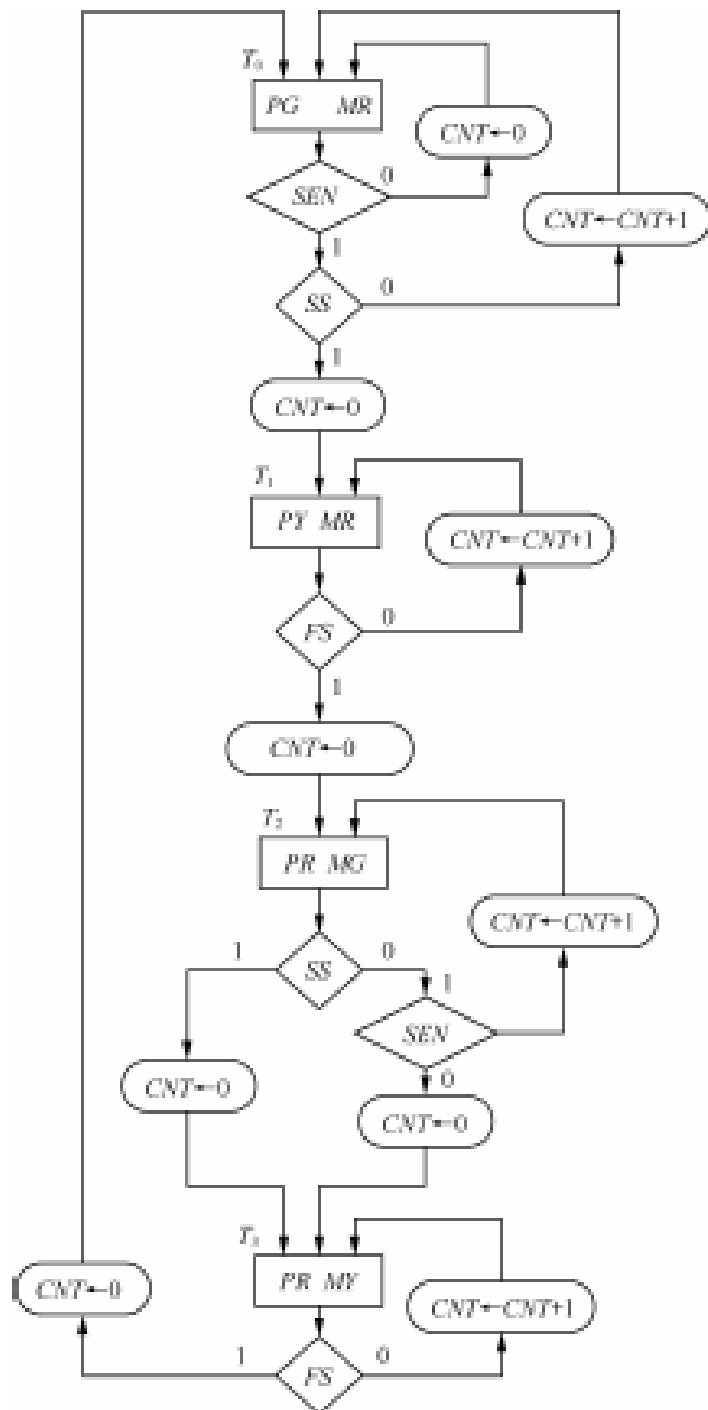


图 7.4.11 交通灯驱动电路逻辑图





$$D_0 = T_0 \cdot (\overline{SEN} + \overline{SS}) + T_3 \cdot SS$$

$$D_1 = T_0 \cdot SEN \cdot SS + T_1 \cdot \overline{FS}$$

$$D_2 = T_1 \cdot FS + T_2 \cdot \overline{SS} \cdot SEN$$

$$D_3 = T_2 \cdot (SS + \overline{SEN}) + T_3 \cdot \overline{FS}$$

$$PP = T_0$$

$$PW = T_1$$

$$MP = T_2$$

例： T1:  $C \leftarrow A$

T5:  $C \leftarrow B$

T6:  $D \leftarrow B$ : 试画出逻辑图

解： 1) 两个目标寄存器： **C、D**。两个源寄存器：**A、B**。

表 7.2.1

真值表

$T_1$	$T_5$	$T_6$	$A_0$	$LD_C$	$LD_D$
0	0	0	$\emptyset$	0	0
0	0	1	1	0	1
0	1	0	1	1	0
0	1	1	$\emptyset$	$\emptyset$	$\emptyset$
1	0	0	0	1	0
1	0	1	$\emptyset$	$\emptyset$	$\emptyset$
1	1	0	$\emptyset$	$\emptyset$	$\emptyset$
1	1	1	$\emptyset$	$\emptyset$	$\emptyset$

