UNIVERSITY OF MICROCHIP
UOFM
MASTERS 2007
第八届中国技术精英年会

11009 PRC

PIC18F外设

课程目标

- 学习完本课程后，您将会了解如下信息：
  - **PIC18F架构**
  - **PIC18F基本外设和中断**
  - 如何用**PIC18F**开发应用程序
  - **PIC18F的扩展指令集优点**
  - **PIC18F的特殊功能**

日程安排

- 架构概述
  - 编程模型
    - 程序存储器
    - 堆栈
    - 数据存储器
  - 指令集概述
- 中断处理和延时
- 基本外设讨论和实验演示
  - I/O端口
  - 开发工具
    - 实验 1： 开发工具的使用、初始化以及读和写I/O端口
  - 模拟外设： 比较器 、 基准电压源和ADC
    - 实验 2： 初始化和ADC转换

日程安排（续）

- 基本外设讨论和实验演示（续）
  - 定时器
    - 实验 3：计数器
  - **CCP （捕捉、比较和PWM）**
    - 实验 4和5：产生PWM脉冲并测量频率/占空比
  - **MSSP （I²C和SPI / Microwire）**
    - 实验 6：I²C™的初始化以及与温度传感器通信
  - 表读和表写操作
  - **USART**
    - 实验 7：初始化、与主机间通信以及表读/表写操作

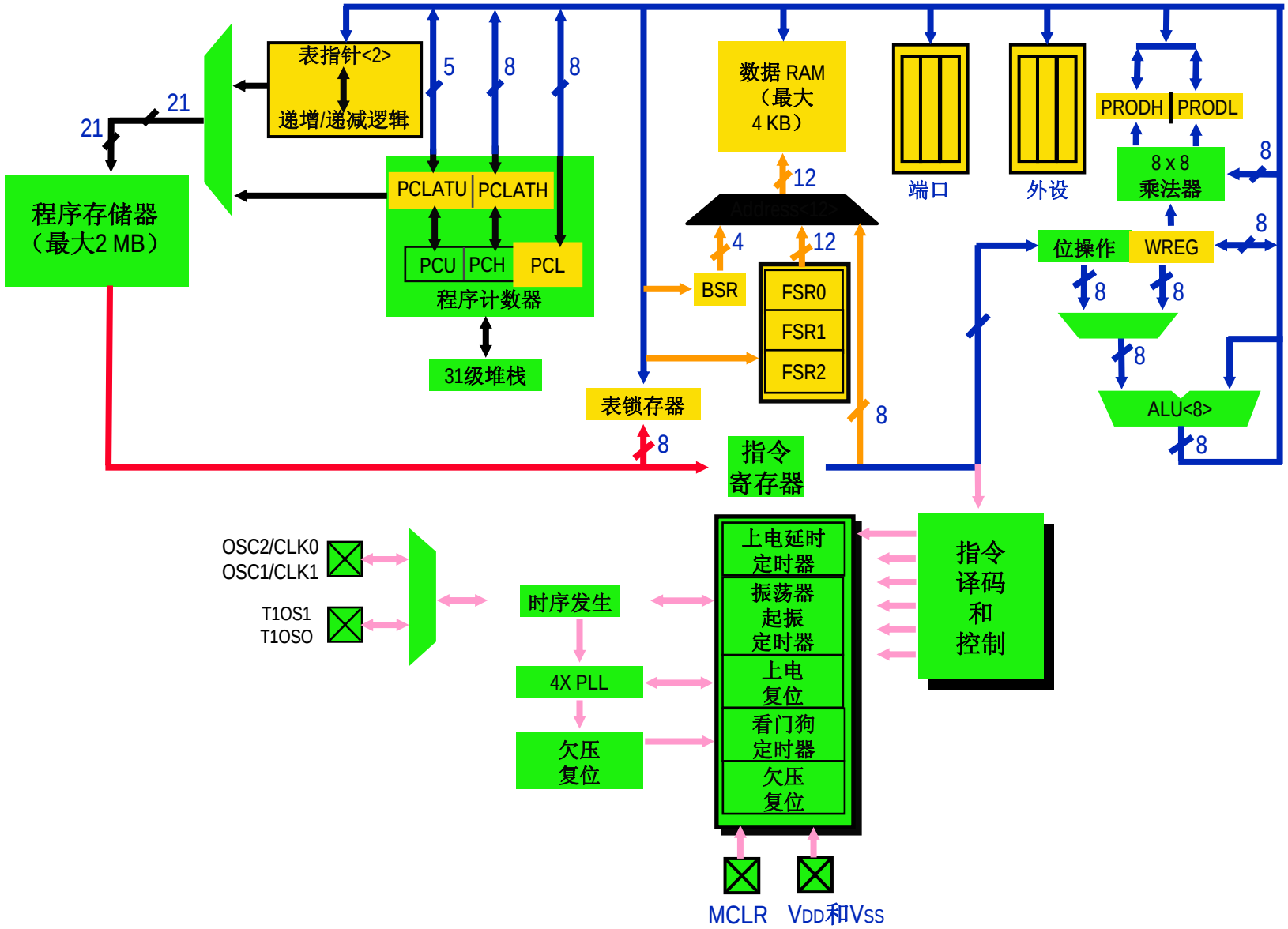
日程安排（续）

- 扩展架构优点
  - 实验 8
    - 扩展的指令集优点
- 振荡器和省电模式
- 特殊功能
  - **PLVD**
  - **PBOR**
  - **ICSP™编程能力**
  - **WDT**
  - 复位
    - 实验 9
      - 示例：如何确定在实际应用中导致复位的原因

UNIVERSITY OF MICROCHIP
UOFM
MASTERS 2007
第八届中国技术精英年会

PIC18F
基本架构

PIC18F框图



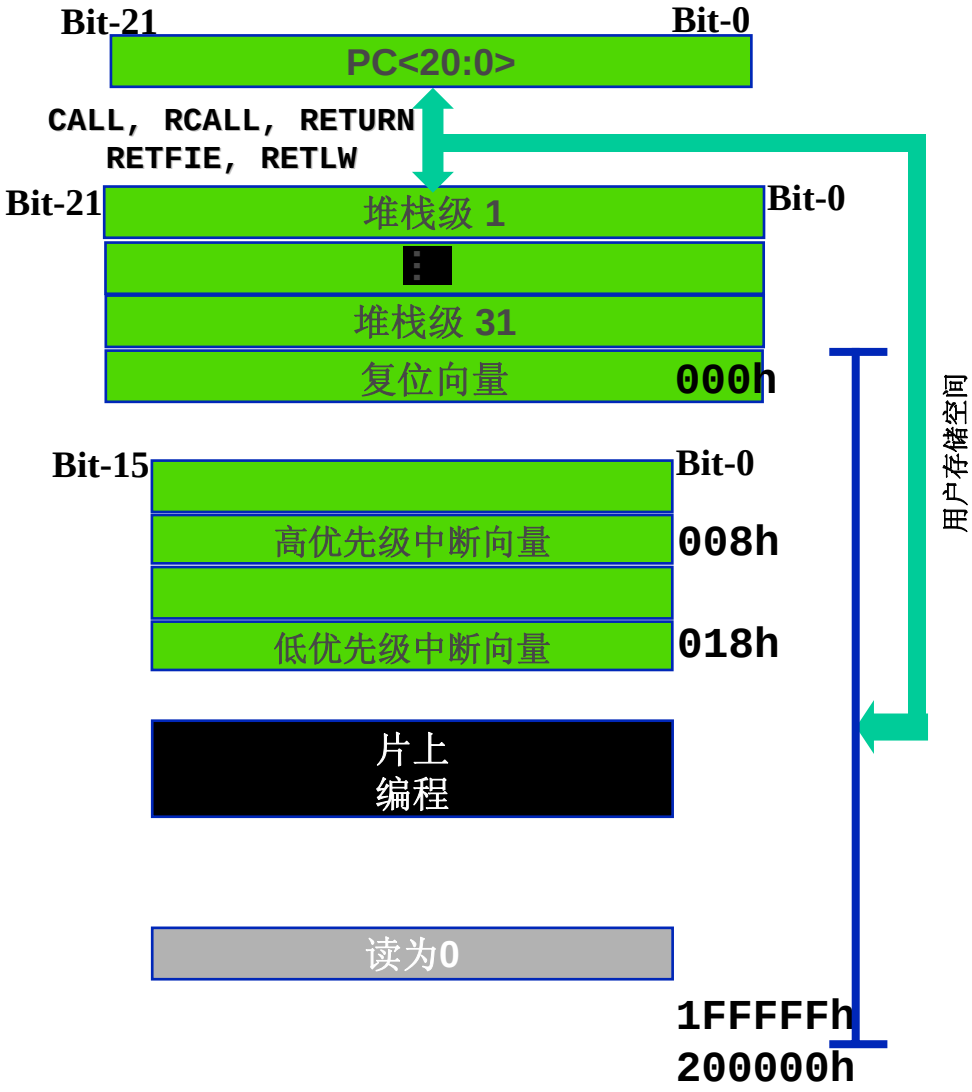
UNIVERSITY OF MICROCHIP
UofM
MASTERs 2007
第八届中国技术精英年会

PIC18F
程序存储器和堆栈

PIC18F MCU 架构
程序存储器

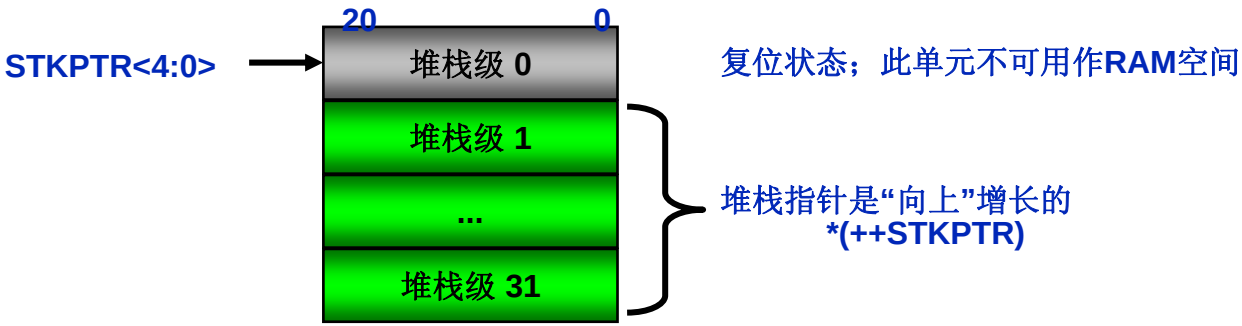
- 最大2 MB的程序存储空间
  - 21位PC （PCU PCH PCL）
- 复位向量位于0000h
- 中断向量位于0008h和0018h
- PC按字递增

  LSB总是为 0
- PC<0>不能被更改



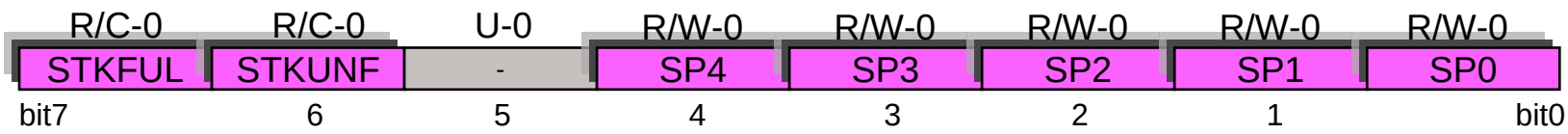
PIC18F 架构
堆栈存储器

- 硬件堆栈 — 31级深
  - 由**STKPTR**指向的独立存储区
  - 可供 CALL、RCALL、RETURN和RETFIE指令以及中断使用



- **5**位堆栈指针寻址**31**级深的堆栈
  - 堆栈指针到达堆栈的底端后不会计满归零
- 栈顶 = **TOSU:TOSH:TOSL**
  - 可读可写

STKPTR — 堆栈指针寄存器



bit 7： **STKFUL**：堆栈满标志位

1 = 发生堆栈满或者溢出

0 = 由用户软件复位或清零

bit 6： **STKUNF**：堆栈下溢标志位

1 = 发生堆栈下溢

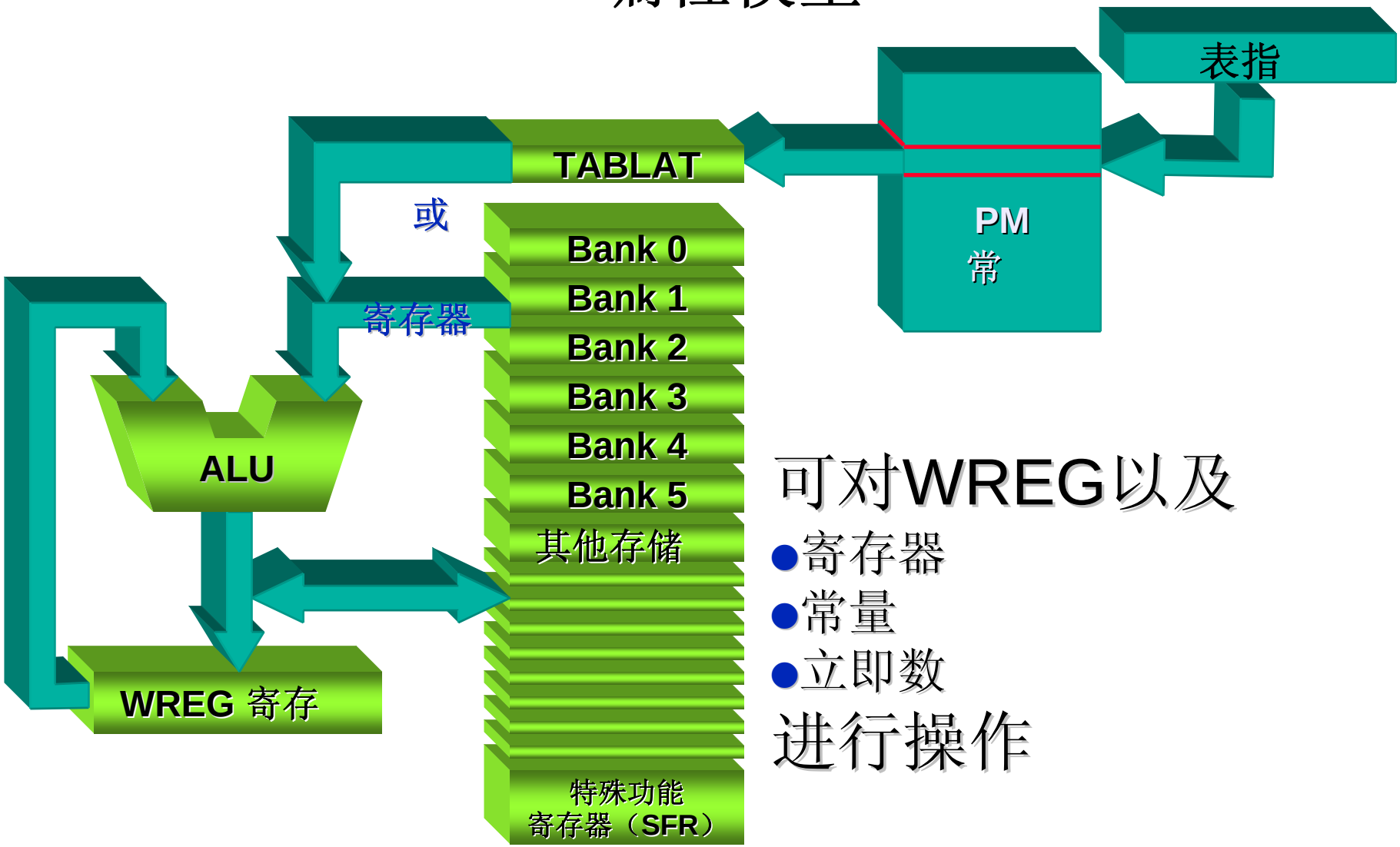
0 = 由用户软件复位或清零

bit 4-0： **SP4:SP0**： 堆栈指针位

UNIVERSITY OF MICROCHIP
UOFM
MASTERS 2007
第八届中国技术精英年会

PIC18F
数据存储器

PIC18F 架构
编程模型

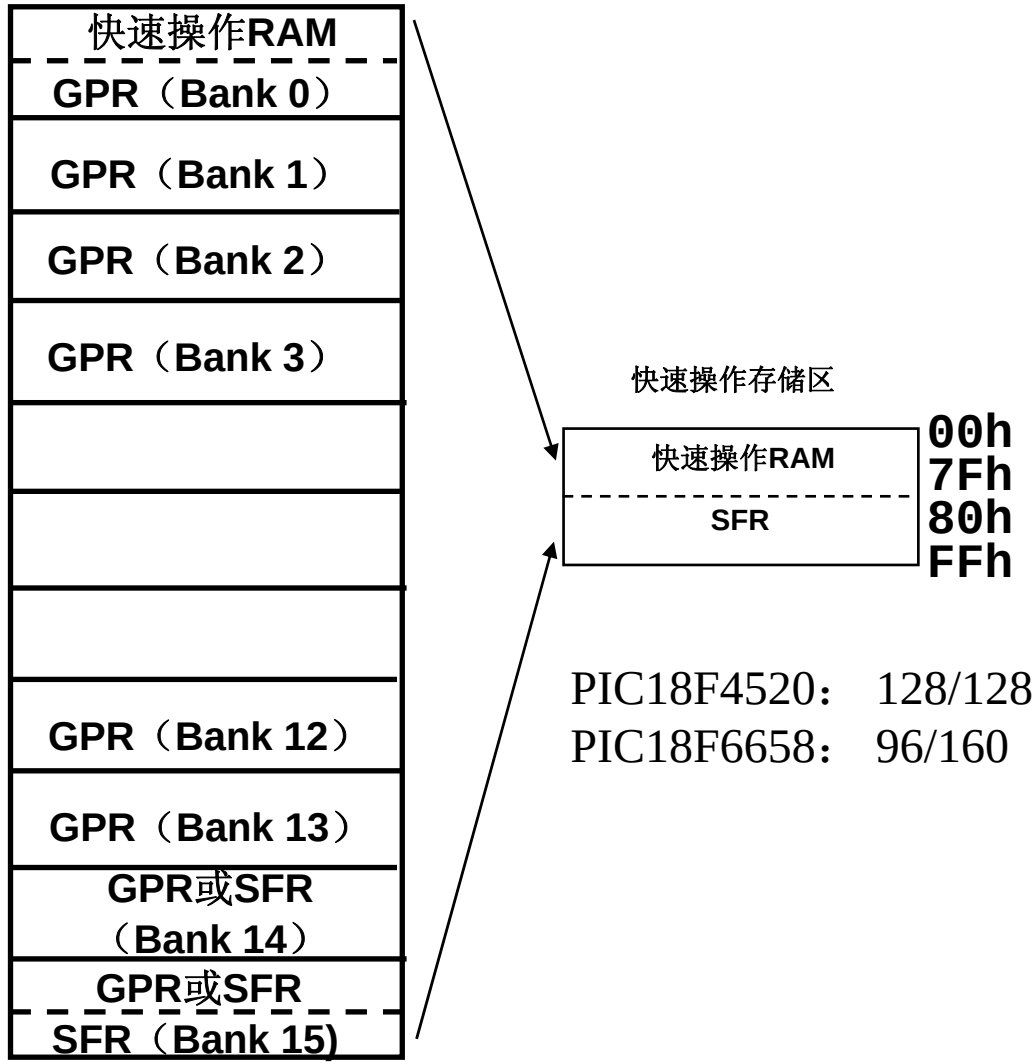


可对WREG以及
- 寄存器
- 常量
- 立即数

进行操作

PIC18F 架构
数据存储器

- 快速操作存储区
  - **Bank 0**的一部分
  - **Bank 15**的一部分
- 无论**BSR**的状态如何，可从任何存储区访问
- 最小化存储区切换
- 用于全局变量、现场的保存和恢复等
- 快速操作存储区的大小取决于器件



UNIVERSITY OF MICROCHIP
UOFM
MASTERS 2007
第八届中国技术精英年会

PIC18F
指令集
概述

PIC18F： 指令集概述

- 数据传送指令
  - 由数据存储器移动到数据存储器
  - 由数据存储器移动到程序存储器（或相反的操作）
  - 立即数
- 算术、逻辑和移位指令
- 单周期**8 x 8**乘法 （**100 ns**）
- 强大的位操作
  - 单周期位置**1**、清零和取反
  - 可直接对包含**I/O**在内的所有寄存器执行操作
- 跳转**/**条件跳转

PIC18F： 指令集概述

- 77条标准指令
  - 73条单字指令
  - 4条双字指令，它们分别是MOVFF、MOVLB、CALL和 GOTO
- 大多数指令都是单周期指令
  - 4条双字指令是双周期指令
  - 10条条件跳转转移、无条件转移和调用指令的执行时间为1个或2个周期
  - 8条表操作指令是双周期指令
  - 3条返回指令的执行时间是2个周期
  - 10条条件跳过指令的执行时间为1个或2个周期
- 表读/表写指令
- 8条扩展指令

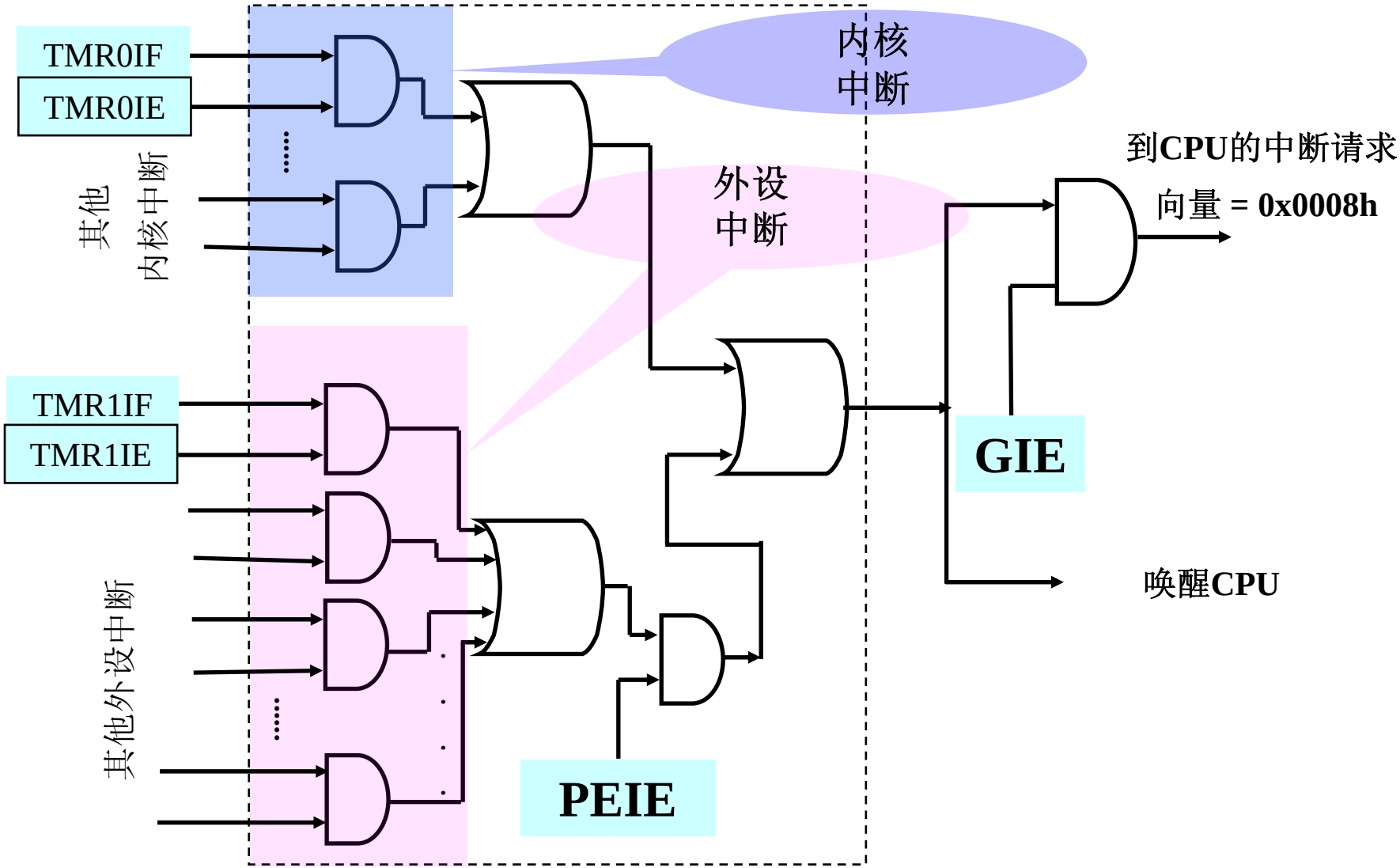
UNIVERSITY OF MICROCHIP
UOFM
MASTERS 2007
第八届中国技术精英年会

PIC18F
中断结构

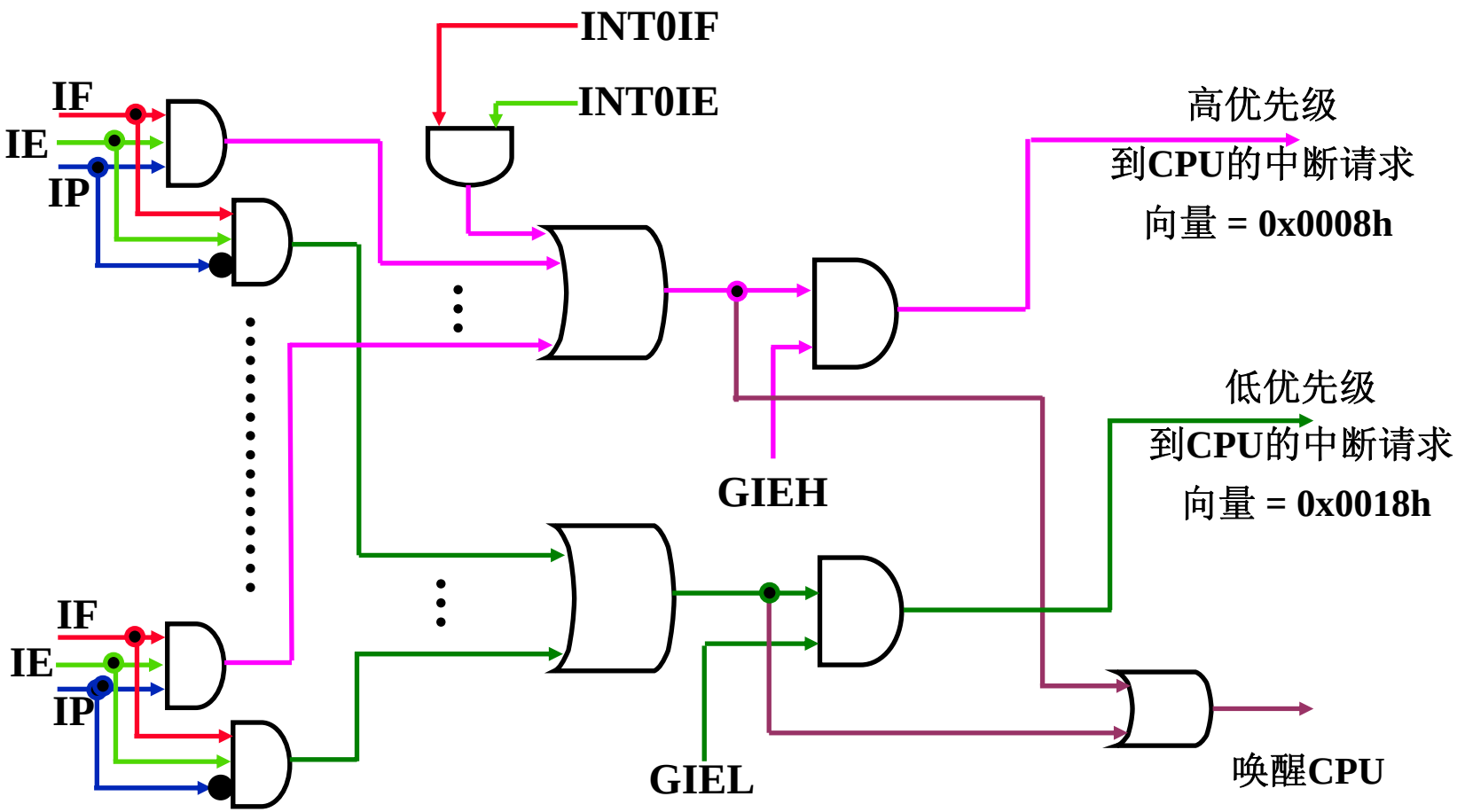
PIC18F 中断概述

- 多个内部和外部中断源
- 每个中断具有独立的中断标志位和屏蔽位
- 可通过**GIE**位屏蔽所有中断
- 具有两种模式
  - 自然模式 （通过清零**RCON** 中的**IPEN**位）
  - 优先级模式 （通过置**1 RCON**中的**IPEN**位）
- 在优先级模式中，每个外设中断可被配置为高/低硬件优先级
  - 高优先级向量位于000008h （默认）
  - 低优先级向量位于000018h
- 中断时的快速现场保护
- 大部分中断会将处理器从休眠模式唤醒
- 中断延时为**3/4**个指令周期

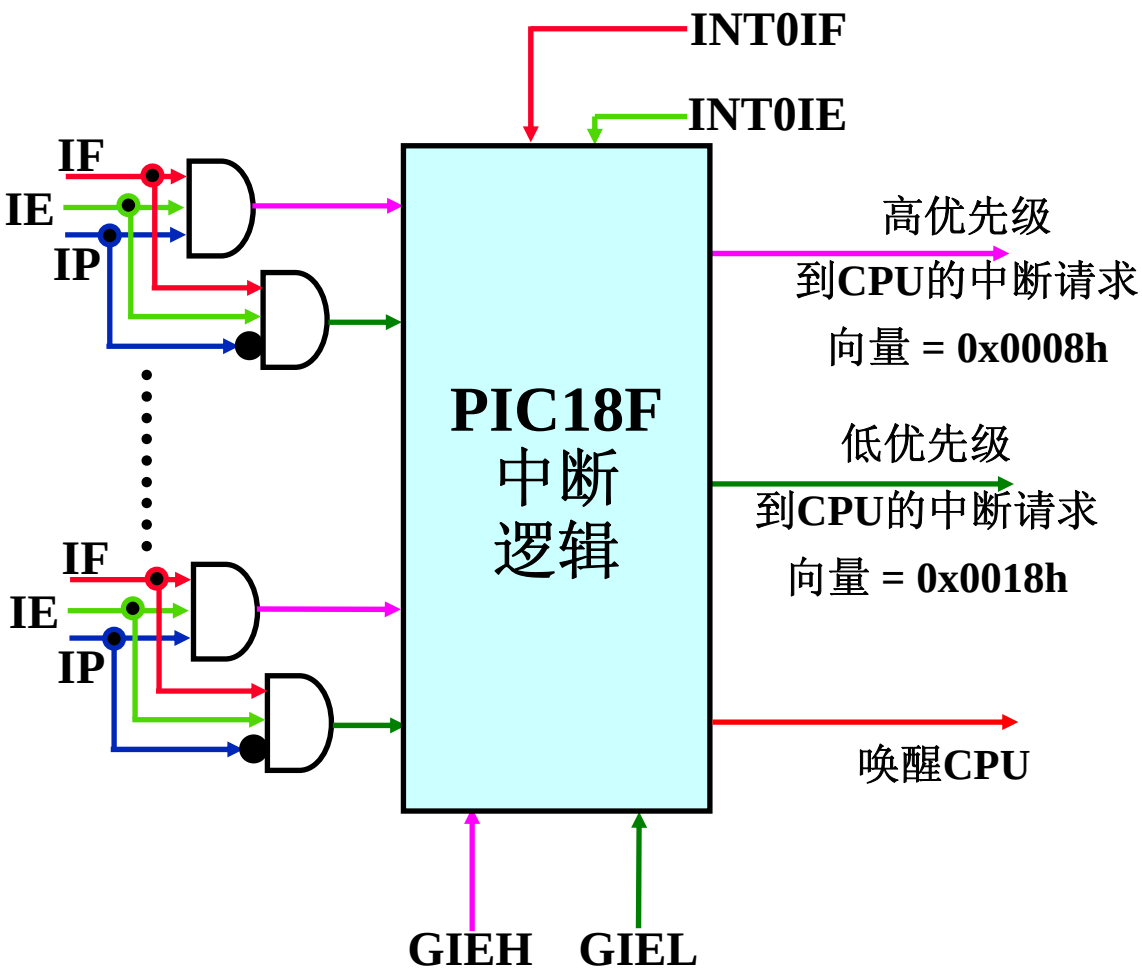
PIC18F
中断配置（自然模式）



PIC18F
中断配置（优先级模式）



PIC18F
中断配置（优先级模式）



PIC18F
中断概述

- 中断
  - 多个中断优先级
    - 高优先级和低优先级
  - 与优先级对应的向量
  - 四个外部中断引脚

    INT0、INT1、INT2和INT3
  - 每个中断具有允许位、标志位和优先级位
    - INT0除外，它总是高优先级

PIC18F
中断概述

- 影子寄存器： 快速调用/返回
  - 调用时可选择将关键寄存器压入影子寄存器，返回时弹出
  - 一级深,用于保存**W**寄存器、**STATUS**寄存器和**BSR**寄存器
  - 在调用/返回指令中通过“**S**”位选择

    CALL FAST
- 影子寄存器： 中断
  - 任何中断都将压入影子寄存器
  - 如果使用**RETFIE**弹出影子寄存器，则禁止**ISR**被中断（低优先级）
  - 影子寄存器仅对于高优先级有用

影子寄存器

- 一级深度
- **call**指令中的“**S**”位使能“快速” 调用/返回过程
- 如果**s = 1**，则将**WREG**、**STATUS**和**BSR**寄存器的内容保存在影子寄存器内
- **s = 1**的指令将把影子寄存器的内容重新恢复到**WREG**、**STATUS**和**BSR**寄存器中
  - **RETURN FAST**
  - **RETFIE FAST**

外设

PIC18F外设

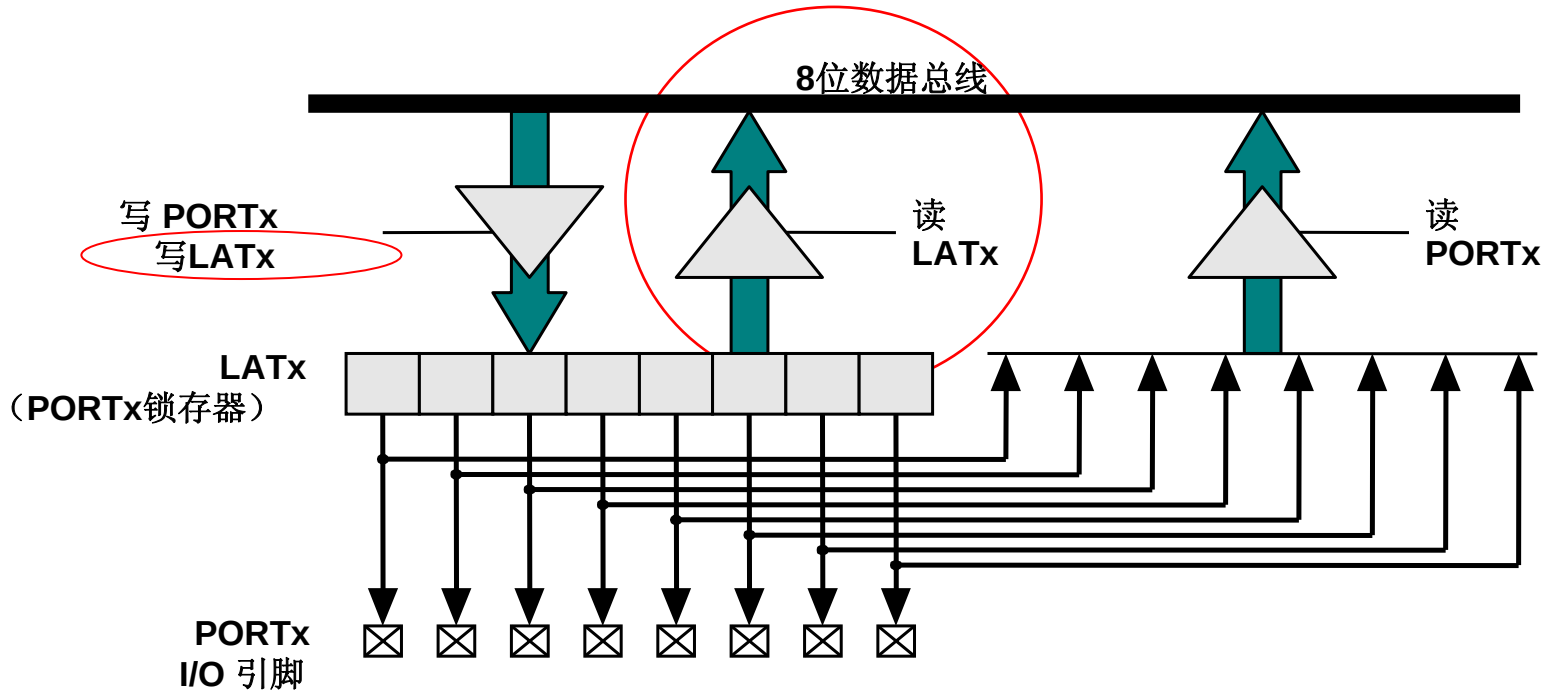
- 数字I/O端口
- 模拟比较器
- 模数转换器
- Timer0、1、2和3
- 比较/捕捉/PWM（CCP）
- 可寻址USART（AUSART）
- 主同步串行端口（MSSP）
- 并行从端口（PSP）

PIC18F外设
数字I/O端口

- 最多66个双向I/O引脚，某些引脚与外设功能复用
- 高驱动能力：
  - 每个引脚的最大灌电流为25 mA
  - 每个引脚的最大拉电流为25 mA
- 可直接驱动LED
- 直接的单周期位操作
- 每个引脚具有单独的由软件控制的方向控制
- 所有引脚均具有ESD保护二极管
  - 基于人体模型的电压为4 kV
- 引脚RA4通常为漏极开路引脚 ——请查看具体器件
- 最初默认所有I/O引脚为输入引脚 （高阻态）
- 与模拟功能复用的引脚默认为模拟输入引脚

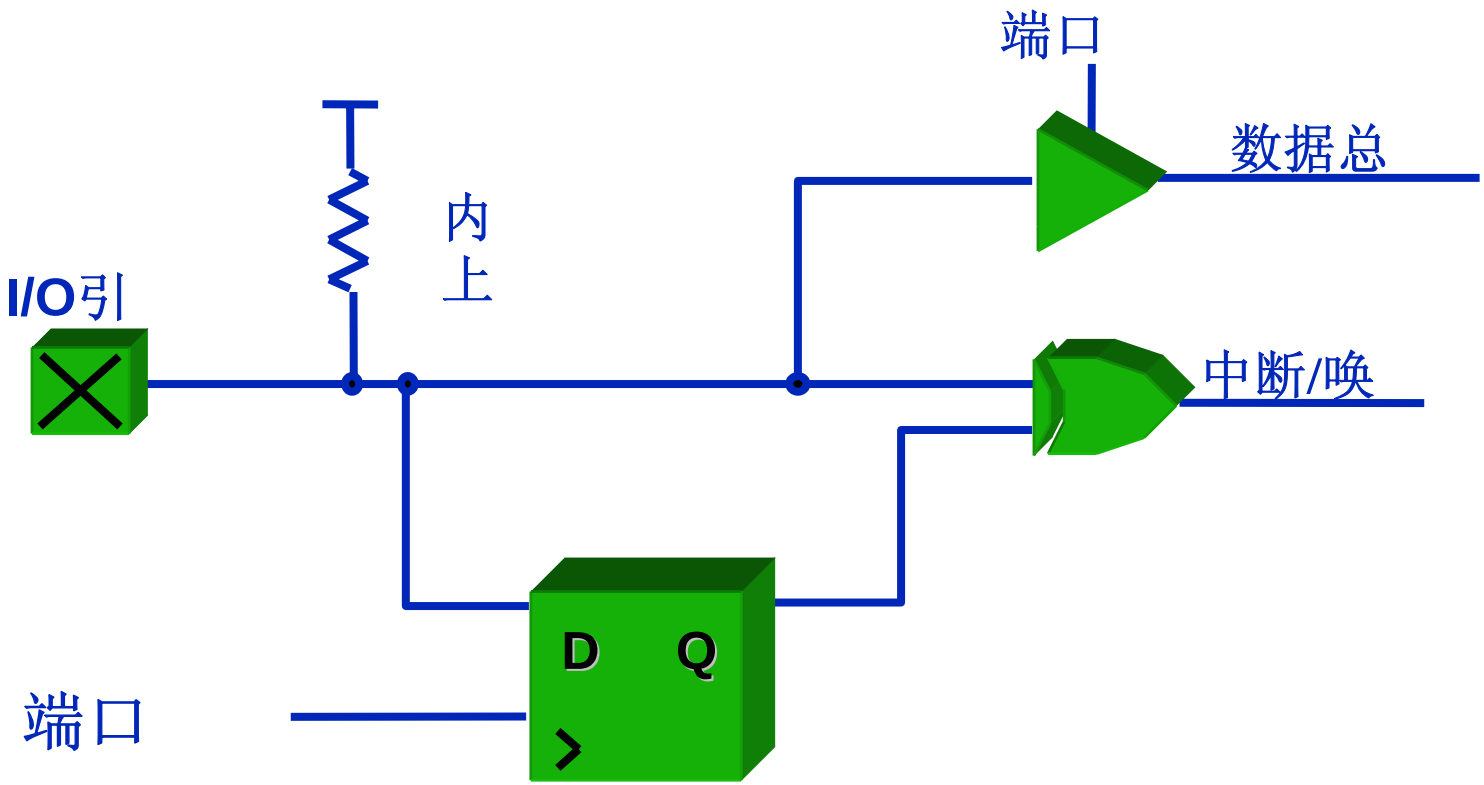
PIC18F 外设
数字I/O端口：使用LATx寄存器

- 将锁存寄存器添加到PIC18存储器映射以消除RMW情形。
  - **MOVF PORTX,w** ；读PORTX*引脚*（引脚的实际状态）
  - **MOVF LATX,w** ；读PORTX*输出锁存寄存器*（所需的引脚状态）
  - **MOVWF PORTX** ；写PORTX/LATX
  - **MOVWF LATX** ；写PORTX/LATX



PIC18F外设
PORTB：电平变化中断

- 内部上拉和唤醒/电平变化中断特性



PIC18F外设
PORTB：配置电平变化中断

INTCON2：中断控制寄存器2

R/W-0	R/W-0	R/W-0	R/W-0		R/W-0		R/W-x
RBPU	INTEDG0	INTEDG1	INTEDG2	—	TMR0IP	—	RBIP
Bit - 7	6	5	4	3	2	1	

使能内部上拉。

当未外部连接上拉电路时使能

中断优先级位

1：高优先级

0：低优先级

允许电平状态变化中断

INTCON：中断控制寄存器

中断状态位

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
GIE/GIEH	PEIE/GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF
Bit - 7	6	5	4	3	2	1	

UNIVERSITY OF MICROCHIP
UOFM
MASTERS 2007
第八届中国技术精英年会

实验1

PIC18F：开发工具

我们首先回顾一下**Microchip**开发工具的基本用法...

PIC18F：开发工具
MPLAB® IDE

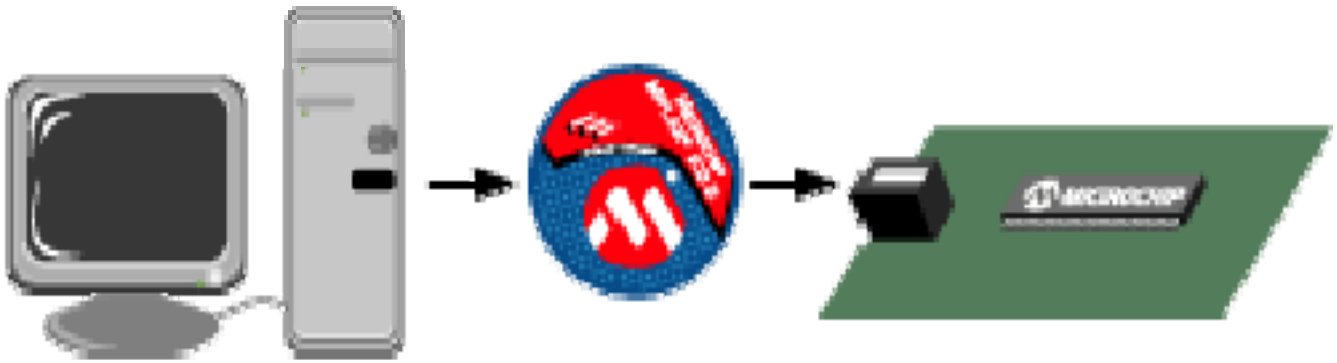
- **MPLAB IDE：集成开发环境**
- **它集成了不同的Microchip和第三方工具**
  - 代码编辑器
  - 交叉编译器
  - 汇编器
  - 软件模拟器、在线调试器和仿真器
  - 编程器

MPLAB® ICD 2 （在线调试器）

- **MPLAB® ICD 2**是一种低成本的实时调试器和编程器，它能提供如下功能：
  - 实时后台调试
  - 读/写目标单片机的存储空间和**EEDATA**区域
  - 编程配置位
  - 擦除带校验的程序存储空间

PIC18F：开发工具
MPLAB® ICD 2
（在线调试器）

连接MPLAB ICD



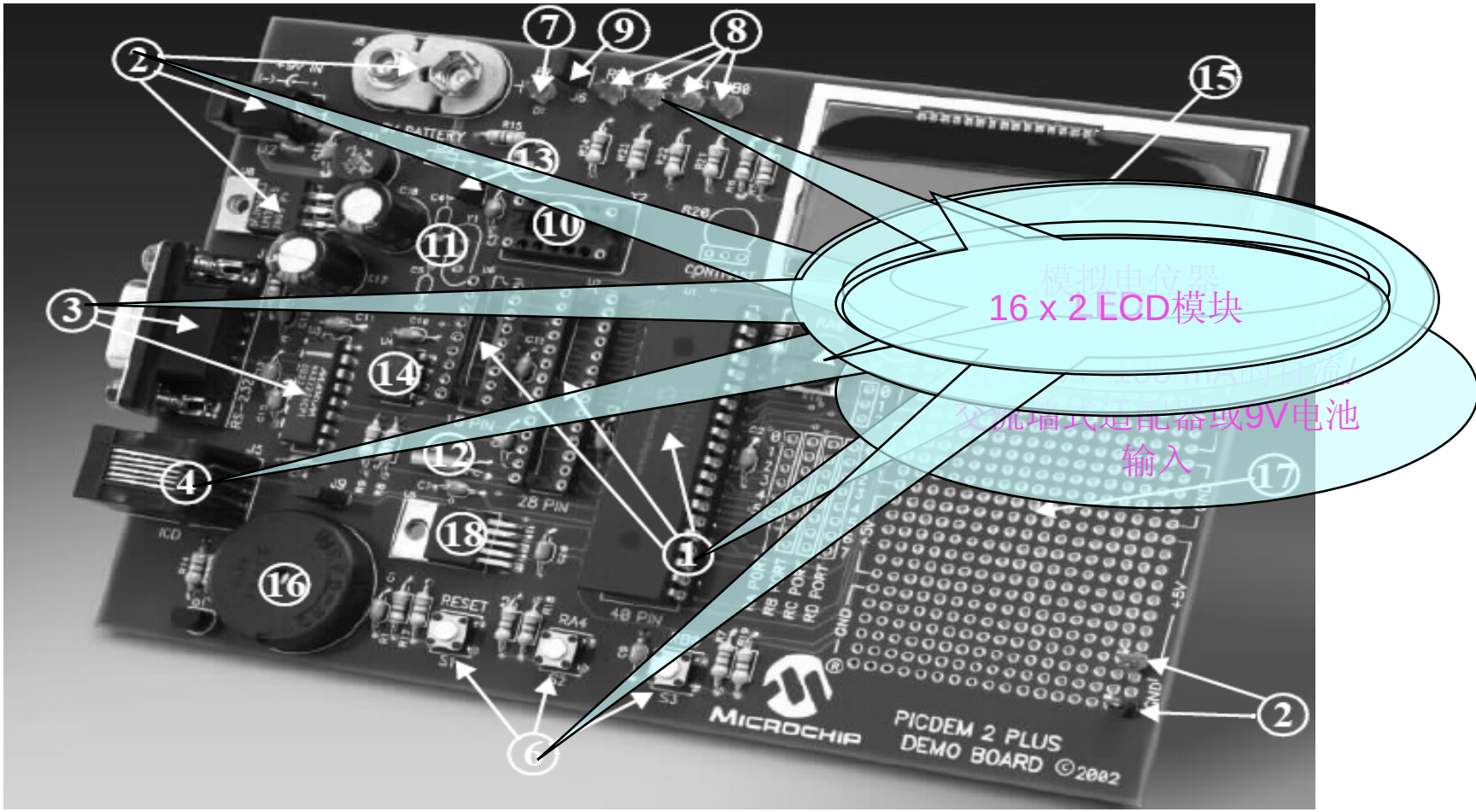
对于所有的演示实验，我们将会把MPLAB ICD 2调试器与PICDEM™ 2板一起使用。

PIC18F：开发工具
PICDEM™ 2 Plus板

- **PICDEM 2 Plus板**
  - **2 x 16 LCD显示器**
  - 蜂鸣器
  - **RS 232端口**
  - 板上温度传感器
  - **4个LED**
  - **2个按钮开关和主复位**
  - **18/28/40引脚控制器插槽**
  - **MPLAB ICD 2连接器**
  - 广阔的实验布线区
  - 使用**9V**电池或直流电源组工作

PIC18F：开发工具 PICDEM™ 2 Plus板

PICDEM 2 Plus板硬件详细信息



完成实验的一般指导

- 所有实验均位于下列文件夹中 **C:\Masters\11009_PRC\LabN**
  - 其中**X**是与实验相关的实验编号。

完成实验的一般指导

在每个项目中搜索关键语句：

Your Code Here

并遵循所提供的说明完成实验

- 使用提供给您的讲义了解寄存器定义

实验1： 初始化、读/写I/O口

- 目标：
  - 了解端口配置
  - 了解读和写端口或端口引脚
  - 了解LAT寄存器的重要性
  - 了解端口电平状态变化中断的用法
  - 了解外部中断

实验1：初始化、读/写I/O口

- 实验1的代码：
  - 打开项目**Lab1.mcp**
  - 在文件**Lab1_main.c**中编写C源代码以配置
    - PORTB<7:1>作为输出引脚
    - PORTB<0>作为输入引脚
    - 允许电平状态变化中断（PORTB<0>上的开关SW3）

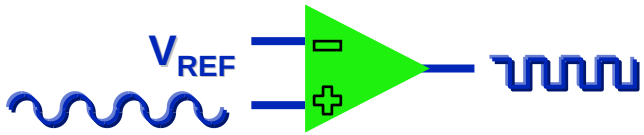
    TRISB = 0x01, INTCON2.RBPU = 1

    INTCON2.RBIP = 1, INTCON. GIE/GIEH = 1
  - 在**PORTB**电平状态变化中断服务程序（**ISR**）中
    - 读PORTB<0>，且如果引脚状态为零则递增计数
  - 在**main**函数中
    - 写一个函数以在板上LCD上显示计数值

实验1：初始化、读/写I/O口

- 预期的结果：
  - 完成实验后，将预期得到如下结果：
    - 在LCD上显示一个数值
    - 每次按下SW3时，该数值都应递增
    - 数值应当是顺序递增的，中间没有跳过任何数字

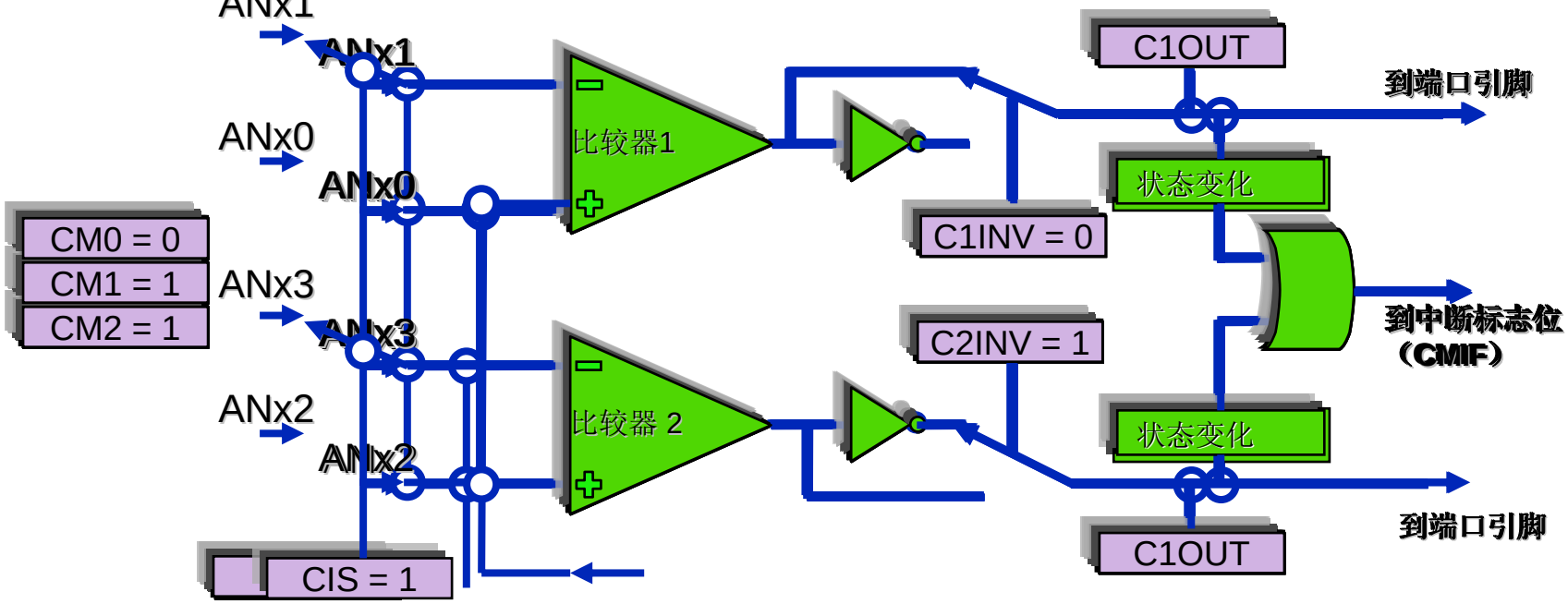
PIC18F外设
模拟比较器模块



- 2个模拟比较器
- 在“休眠”模式下工作
- 在输出电平变化时产生中断/唤醒
- 具备比较器输出引脚
- 8种可编程工作模式

PIC18F外设
模拟比较器模块配置

两个比较器复用4路输入 CM2:CM0=110



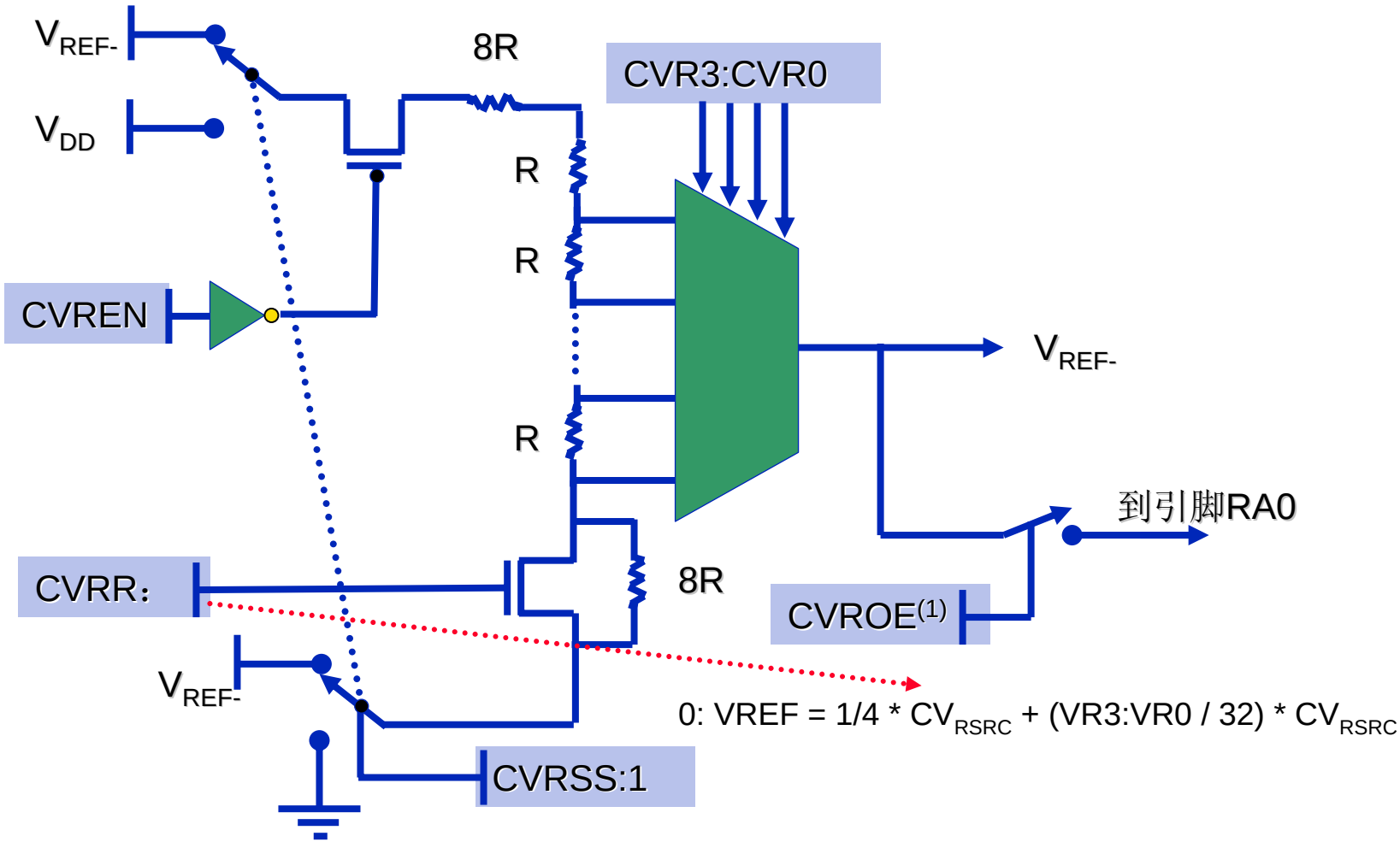
CMCON：比较器控制寄存器

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
bit7	6	5	4	3	2	1	0

PIC18F外设
内部VREF：框图

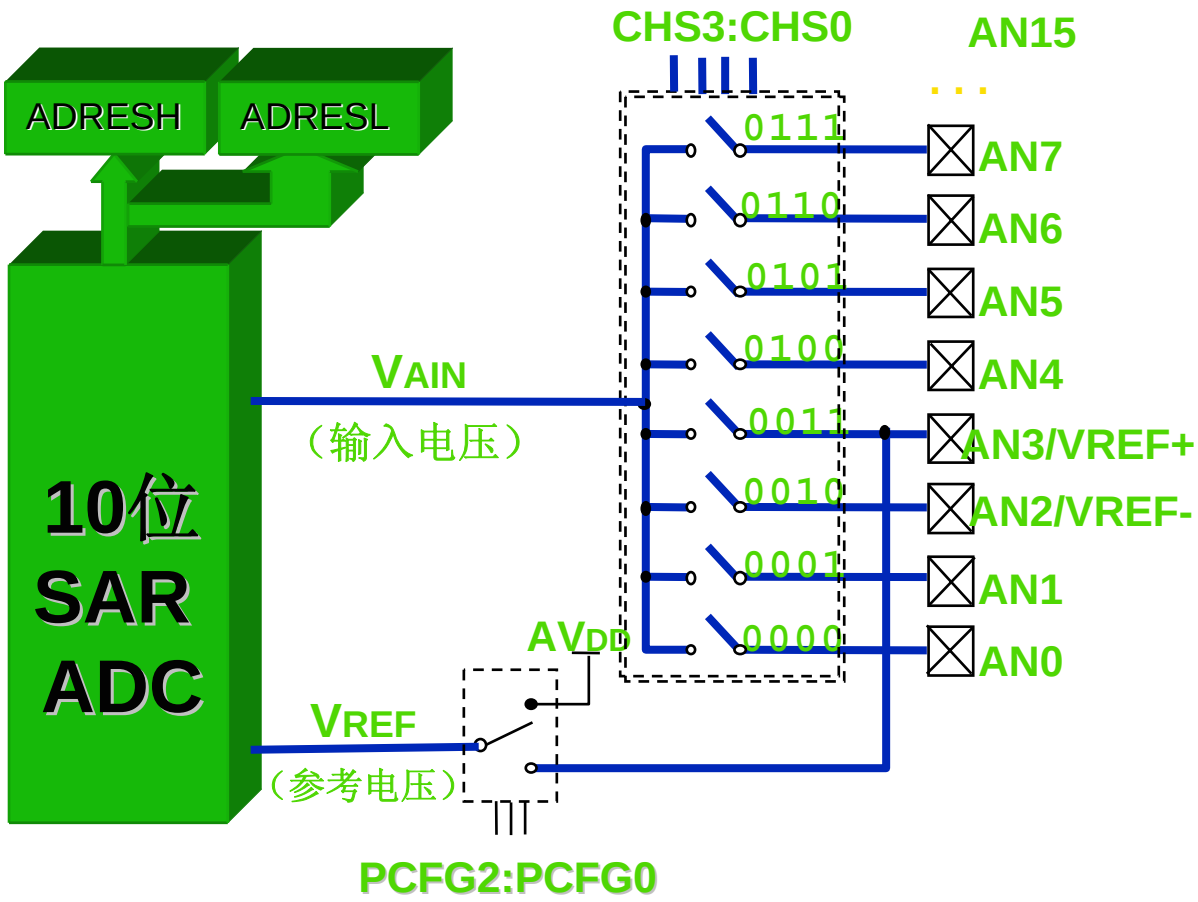
- 提供片上参考电压
- 16阶梯形电阻网络；提供可编程参考电压
- 24或32阶大小
- 基准电压源可以取自片内或片外电压
- 可被用作D/A转换器
- VREF 可直接连接到输出引脚

PIC18F外设
内部VREF：配置



注： 1. CVROE改写TRISA<0>位的设置

PIC18F外设
10位ADC框图



- 最多16个通道
- 10位±1 **LSb**
- 可在“休眠”期间转换
- 输入捕捉时的转换
- 内部或外部参考电压
- **20到100ksps***
- 容易配置

注：转换速度请参见器件数据手册

PIC18F外设
ADC模块配置

ADCON0： A/D控制寄存器0

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CHS3	CHS3	CHS3	CHS3	GO/DONE	ADON
Bit7	6	5	4	3	2	1	0

此配置寄存器将帮助选择所需的通道并控制转换

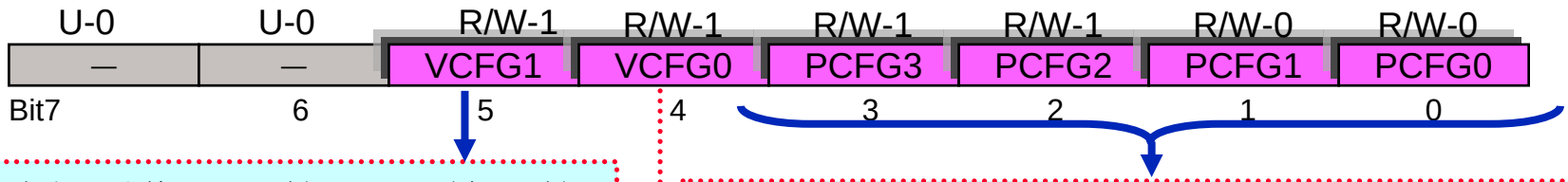
3	2	1	0	选中的通道
0	0	0	0	CH – 0
0	0	0	1	CH – 1
0	0	1	0	CH – 2
0	0	1	1	CH – 3
0	1	0	0	CH – 4
;				;
;				;
1	1	1	1	CH – 15

开始转换

使能ADC

PIC18F外设
ADC模块配置

ADCON1：A/D控制寄存器1



选择用作ADC的V_{REF-} 输入的参考电压
0： V_{REF-}为AV_{SS}
1： V_{REF-}为AN2

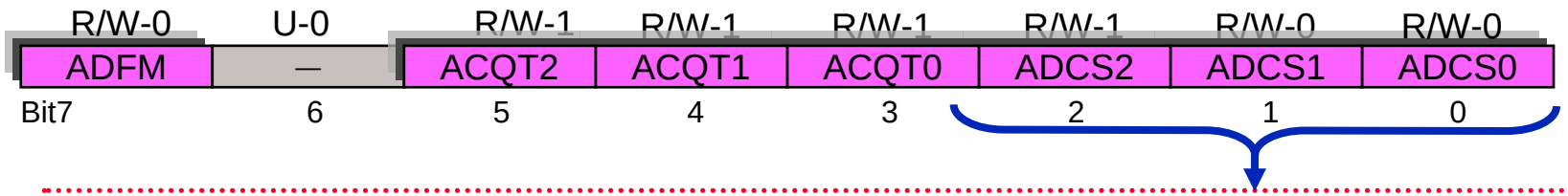
选择用作ADC的VREF +输入的参考电压
0：VREF+为AVDD
1：VREF+为AN3

将可用的模拟端口引脚配置为模拟输入引脚
0000 – 无模拟通道
0001 – 1个模拟输入通道（AN0）
“
“
“
0111 – 7个模拟输入通道（AN0至AN6）
“
1111 – 16个模拟输入通道（AN0至AN15）

当输入传感器使用不同电压的电压源时，参考电压选择帮助您获得满量程的ADC值。

PIC18F外设
ADC模块配置

ADCON2：A/D控制寄存器2



SAR需要11个时钟来转换10位数据，且每个时钟周期应大于2 μS（近似值）
以下位通过选择输入时钟的频率范围来帮助实现以上要求

ADCS2:ADCS0 – 000 → T_{AD} = 2 T_{OSC}
ADCS2:ADCS0 – 100 → T_{AD} = 4 T_{OSC}
ADCS2:ADCS0 – 001 → T_{AD} = 8 T_{OSC}
ADCS2:ADCS0 – 101 → T_{AD} = 16 T_{OSC}
ADCS2:ADCS0 – 010 → T_{AD} = 32 T_{OSC}
ADCS2:ADCS0 – 110 → T_{AD} = 64 T_{OSC}
ADCS2:ADCS0 – x11 → T_{AD} = RC**
根据V_{DD}的变化， T_{AD}的变化范围为4到6** ms

PIC18F外设
ADC模块配置

ADCON2：A/D控制寄存器2

R/W-0	U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-0	R/W-0
ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
Bit7	6	5	4	3	2	1	0

为了得到良好的转换精度，采样保持电容必须充电至模拟输入电压的值
下面是给电容充电时的延时：
- 放大器稳定时间（TAMP = 5 µS，近似值）
- 保持电容充电时间（TC = 9.61 µS，近似值）
- 温度系数（TCOFF = 1.25 µS，近似值）

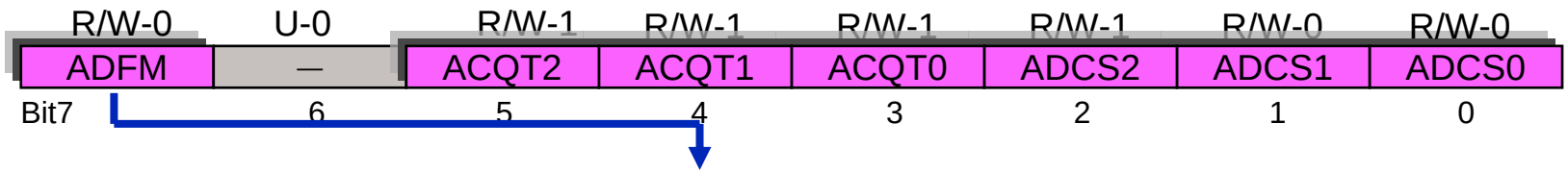
因此所需的总采集时间为TACQ = 12.86 µS

ACQT2：ACQT0在转换开始之前，将提供此延时（将GO位置1后）
采集延时 = $(2^n * T_{AD})$**
其中n = ACQT2:ACQT0

** 请参见器件数据手册以获得精确的延迟时间

PIC18F外设
ADC模块配置

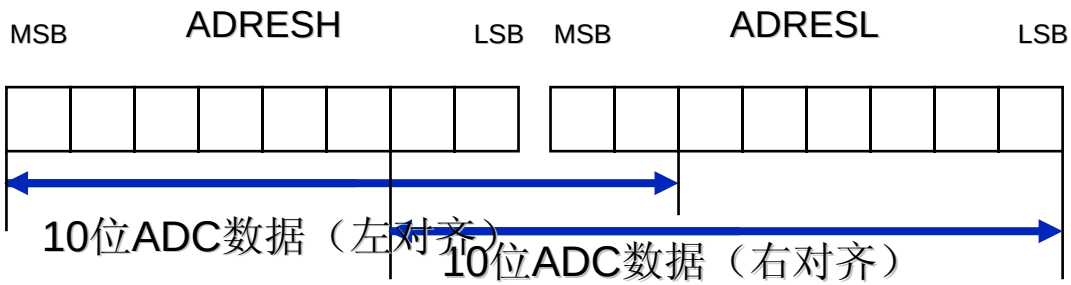
ADCON2：A/D控制寄存器2



该位将允许ADC的结果以左/右对齐的方式存储在ADRES寄存器中

当ADFM = 0时：结果是左对齐的，这将帮助您通过只读ADRESH寄存器获得低分辨率数据

当ADFM = 1时：结果是右对齐的，这将帮助您通过只读ADRESL寄存器获得高分辨率数据



实验2：初始化和ADC转换

- 目标：
  - 理解ADC配置
  - 理解读ADC通道
  - 理解选择ADC通道
  - 理解切换ADC通道
  - 理解调整ADC的值

实验2：初始化和ADC转换

- 实验2的代码：
  - 打开项目**Lab2.mcp**
  - 在文件**Lab2_main.c**中写入C源代码以配置
    - PORTA<0>引脚作为模拟输入引脚
    - 转换时钟为FOSC/8
    - 采集时间为$12T_{AD}$
    - 结果格式采用右对齐
    - 选择ADC的内部参考电压

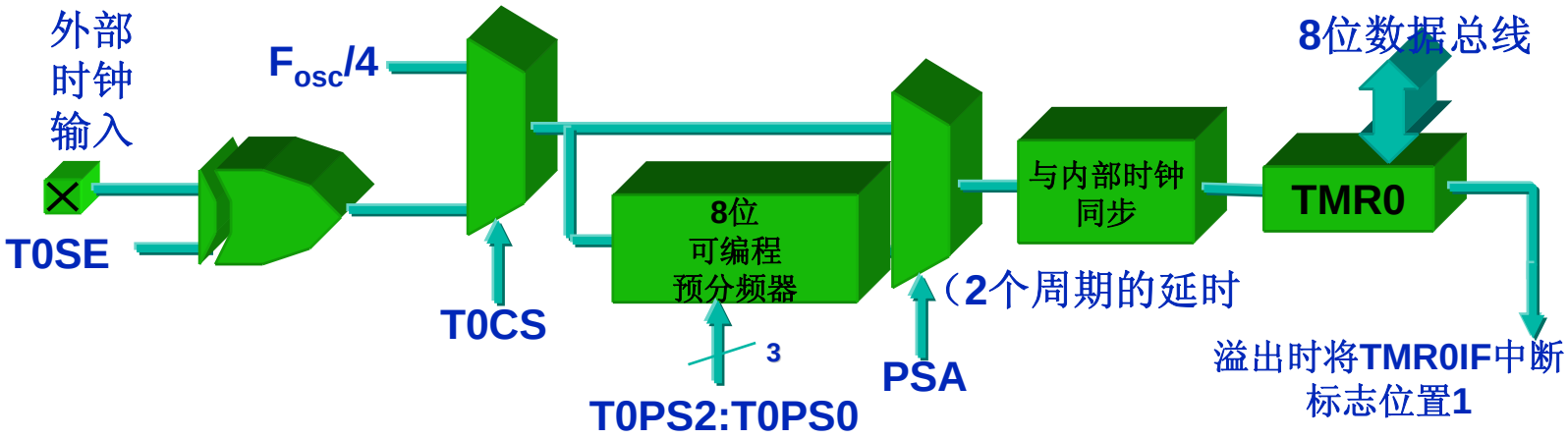
    ADCON0 = 0x01，ADCON1 = 0x0E，ADCON2 = 0x29
  - 选择**ADC通道0**用于转换
  - 启动**ADC转换**并等待转换完成
  - 读**ADC结果寄存器**并显示结果

实验2：初始化和ADC转换

- 预期的结果：
  - 完成实验后，将预期得到如下结果：
    - LCD显示出电压
    - 最小电压应当为0V （电位器位于最小值的位置）
    - 最大电压应当为5V （电位器位于最大值的位置）

PIC18F外设
Timer0

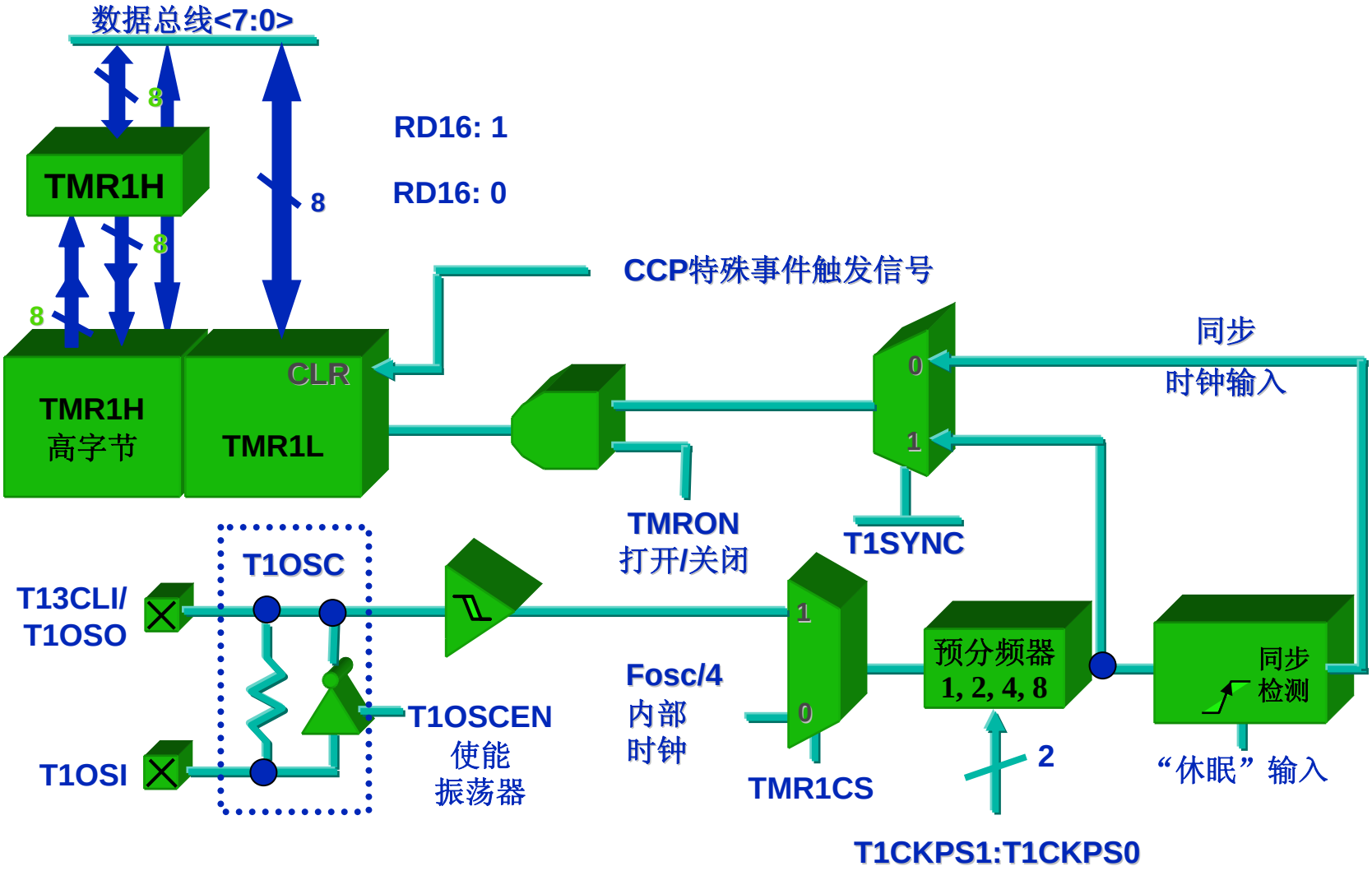
- 8位/16位定时器/计数器
  - 16位读和写操作
- 8位软件可编程预分频器
- 内部或外部时钟选择
- 当从FFh/FFFFh溢出到00h时产生中断



PIC18F外设
Timer1和Timer3

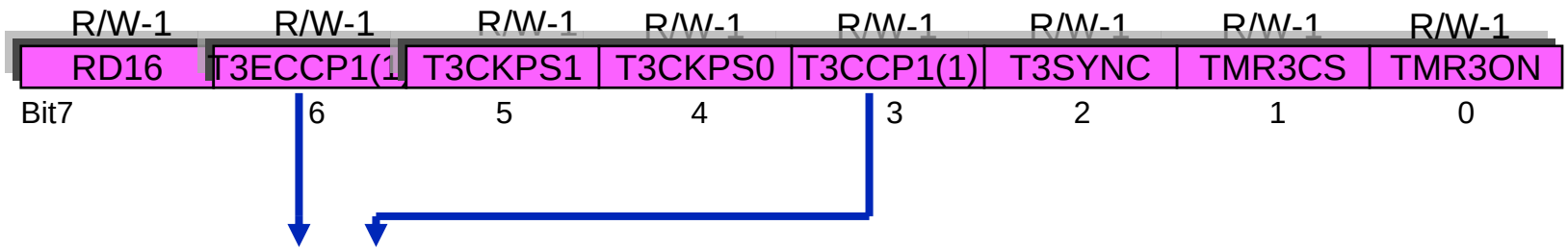
- 16位定时器/计数器
- 定时器、同步计数器或异步计数器
- 允许内置振荡器通过外部晶振工作
- 由2个可读/可写8位寄存器组成
- 1、2、4或8分频预分频器
- 当从FFFFh溢出到0000h时产生中断

PIC18F外设
Timer1和Timer3（续）



PIC18F外设
Timer 3模块配置

T3CON：TIMER0控制寄存器



1　X：Timer3是CCP和ECCP模块的捕捉/比较时钟源

0　1：Timer3是ECCP1的捕捉/比较时钟源；Timer1是CCP1的捕捉/比较时钟源

0　0：Timer1是CCP和ECCP模块的捕捉/比较时钟源

CCP和ECCP的时钟源选择位

注：1 这些位仅在PIC18F4X80器件中可用

实验3：使用计数器

- 目标：
  - 了解计数器模式下的定时器配置
  - 了解频率测量技术

实验3：使用计数器

- 实验3的代码：
  - 打开项目**Lab3.mcp**
  - 在文件**Lab3_main.c**中编写C源代码以配置
    - Timer1工作在16位同步计数器模式
    - 选择Timer1的预分频比为1:1
    - 在定时器计数器中预装载8000h（ d32768 ）以提供1秒的时间间隔
    - 允许Timer1溢出中断

    T1CON = 0x0B，TMR1H = 0x80，TMR1L = 00
    - 写Timer1 ISR以实现软件RTCC

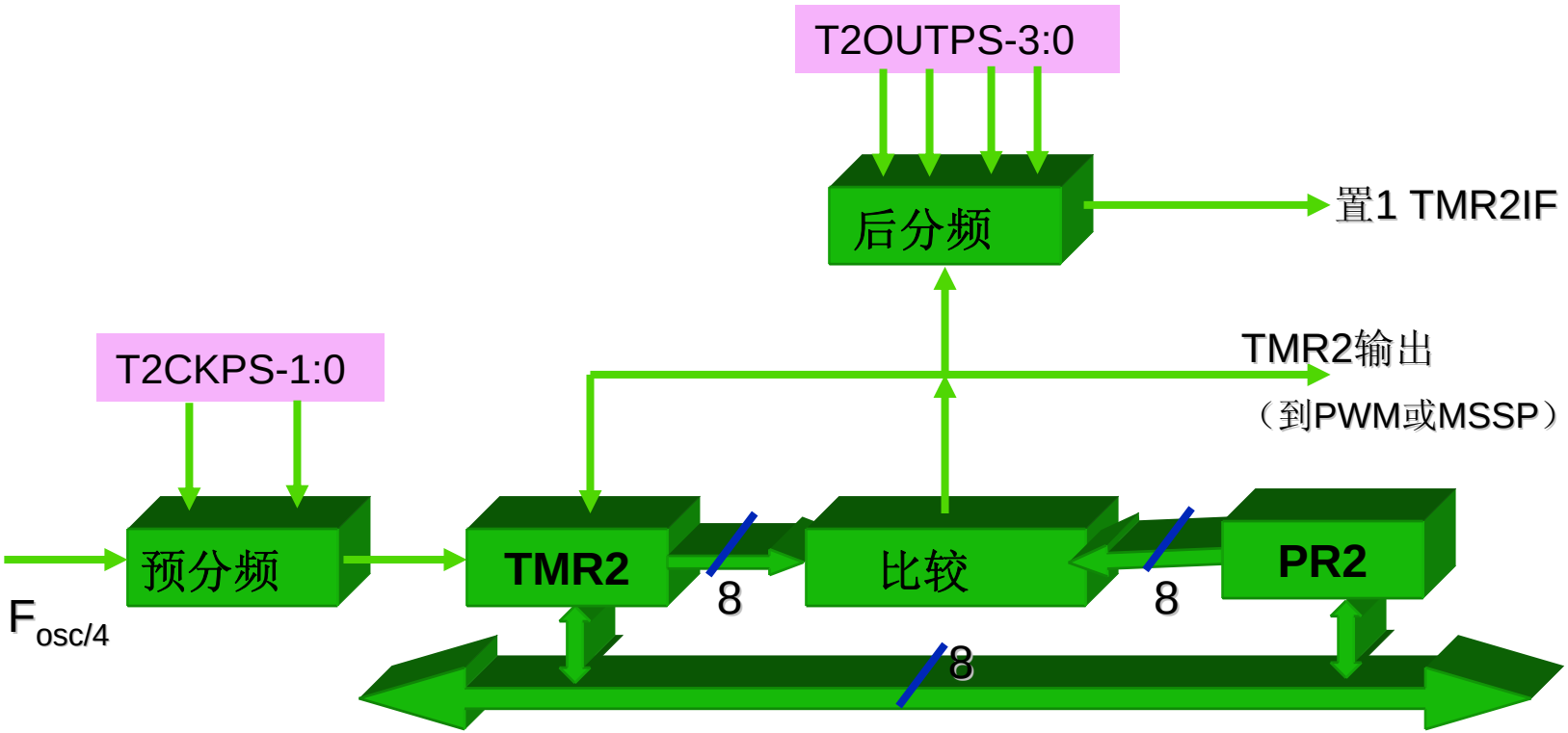
实验3：使用计数器

- 预期的结果：
  - 完成实验后，将预期得到如下结果：
    - LCD以HH:MM:SS的格式显示时间

PIC18F外设 Timer2

- 带预分频器和后分频器的8位定时器
- 用作CCP模块的PWM模式的时基
- TMR2是可读和可写的
- TMR2递增直至其与周期寄存器PR2匹配，然后复位为00h
- 若TMR2与PR2匹配，则通过后分频器产生中断
- 用作MSSP的波特率时钟（SPI）

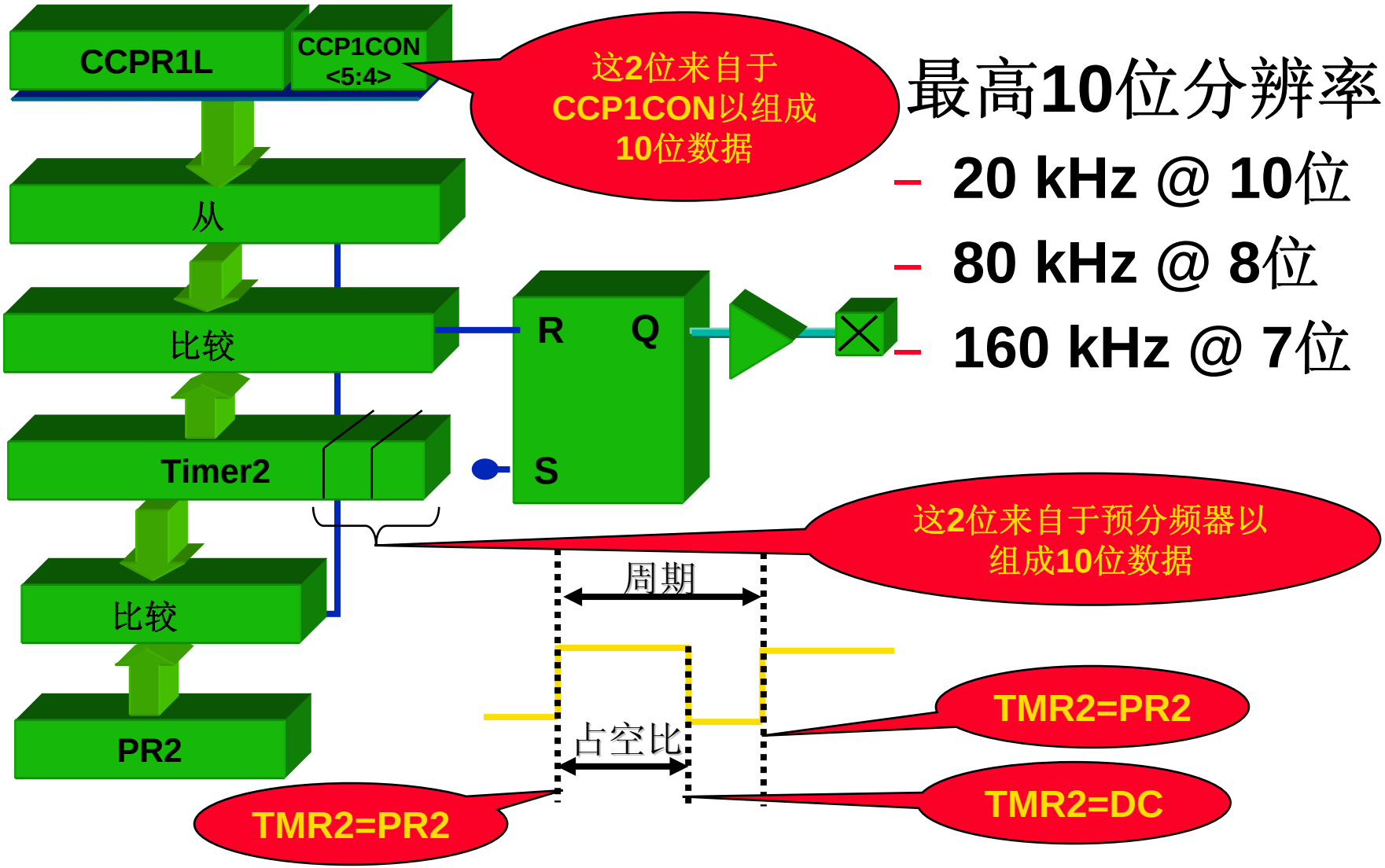
PIC18F外设 Timer 2模块配置



T2CON：TIMER0控制寄存器

U-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
—	T2OUTPS3	T2OUTPS2	T2OUTPS1	T2OUTPS0	TMR2ON	T2CKPS1	T2CKPS0
Bit7	6	5	4	3	2	1	0

PIC18F外设
CCP模块：PWM模式



最高10位分辨率

- 20 kHz @ 10位
- 80 kHz @ 8位
- 160 kHz @ 7位

实验4：使用PWM

- 目标：
  - 理解用于**PWM**的定时器配置
  - 理解**TIMER2**的**PWM**工作模式
  - 学习由系统频率计算周期值的方法

实验4：使用PWM

- 实验4的代码：
  - 打开项目**Lab4.mcp**
  - 在文件**Lab4_main.c**中写入**C**源代码以配置
    - Timer2用于单输出PWM工作模式
    - 将预分频比设置为1:4
    - 将默认频率设置为4.9 KHz
    - 将默认占空比设置为50%

  CCP1CON = 0Ch，T2CON = 05h，PR2 = 80h，CCP1CON<5:4> = 3
  - 按下**SW2**或**SW3**时启动/停止**PWM**
  - 按照与模拟输入的比例更改周期的值

实验4：使用PWM

- 实验4的代码：
  - 使用下面的公式计算PWM频率（1/周期）

$$\text{PWM 周期} = [(\text{PR2}) + 1] \cdot 4 \cdot \text{Tosc} \cdot (\text{TMR2 预分频值})$$

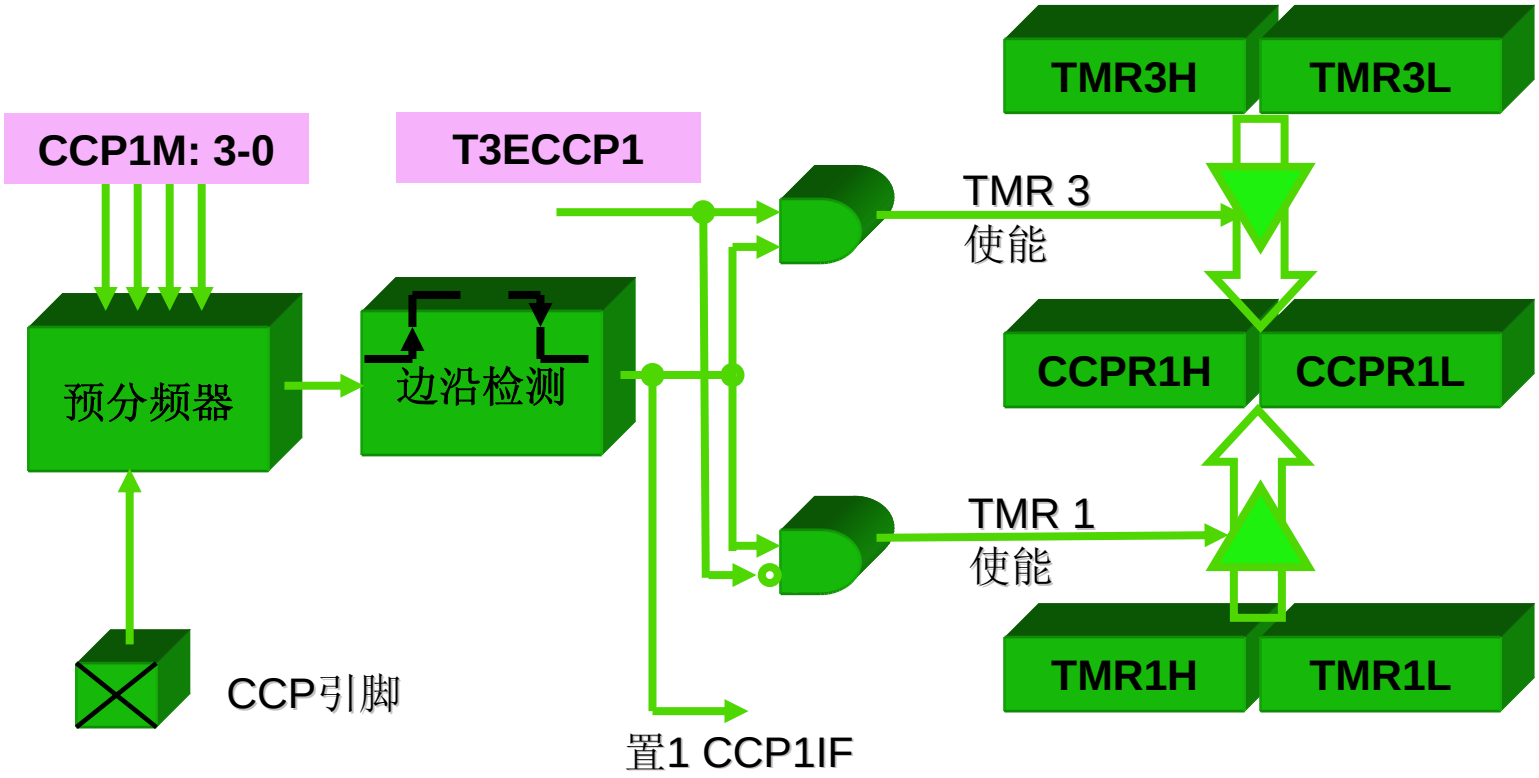
实验4：使用PWM

- 预期的结果：
  - 完成实验后，将预期得到如下结果：
    - 蜂鸣器上发出响声
      - 通过按SW2或SW3启动或停止PWM
      - 如果调节电位器的话，发出的声音的频率将发生改变

PIC18F外设
CCP模块：输入捕捉模式

- 当在**CCPx**引脚上发生事件时，捕捉**16**位**Timer1/Timer3**的值：
  - 每个下降沿捕捉
  - 每个上升沿捕捉
  - 每**4**个上升沿捕捉
  - 每**16**个上升沿捕捉
- 每次捕捉产生一次中断

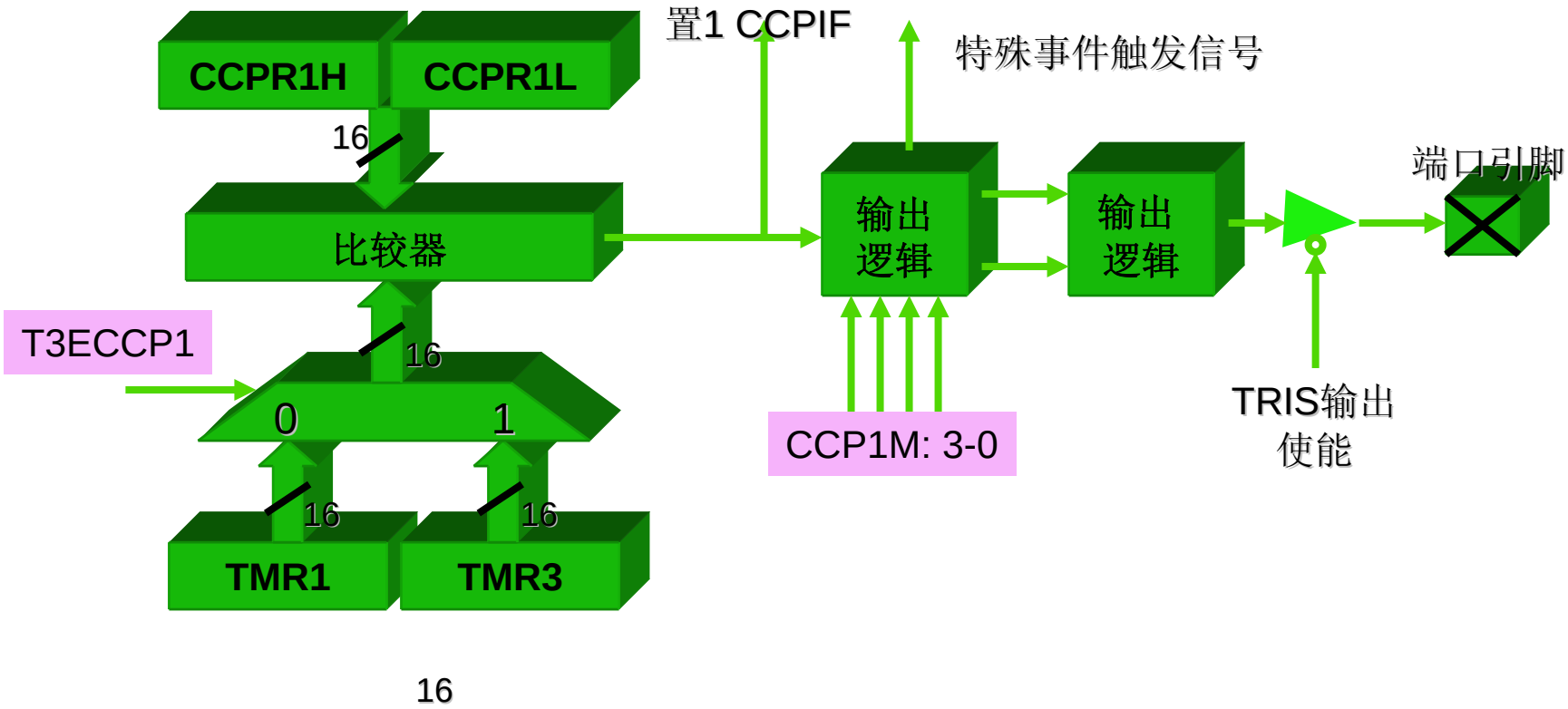
PIC18F外设
CCP模块：输入捕捉模式



PIC18F外设
CCP模块：输出比较模式

- 将16位CCPRn寄存器的值与TMR1/3比较，如果匹配，则将CCPn引脚：
  - 驱动为高电平/低电平
  - 翻转
  - 保持不变
- 比较匹配产生中断
- 比较匹配可触发一个特殊事件
- 可以启动A/D转换

PIC18F外设
CCP模块：输出比较模式



实验5：使用CCP

- 目标：
  - 理解用于输入捕捉的定时器配置
  - 理解**TIMER1**的捕捉工作模式
  - 学习由捕捉到的计数值计算周期的方法

实验5：使用CCP

- 实验5的代码：
  - 打开项目**Lab5.mcp**
  - 在文件**Lab5_main.c**中编写C源代码以配置
    - Timer3用作CCP时钟源输入
    - 设置Timer3以内部时钟源（FOSC/4）运行
    - 配置并允许在上升沿和下降沿各产生一次捕捉中断

    T3CON = 0x49，CCP2CON = 0x05（对于上升沿）

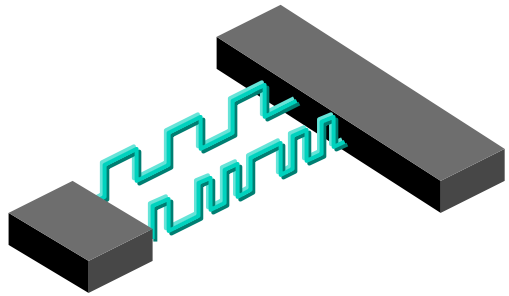
    CCP2CON = 0x04（对于下降沿）
  - **将PWM端口引脚与捕捉端口引脚连接**
  - 写捕捉**ISR**以
    - 读捕捉寄存器和端口引脚的状态
    - 计算周期和占空比

实验5：使用CCP

- 预期的结果：
  - 完成实验后，将预期得到如下结果：
    - LCD上显示出周期/频率以及占空比
      - 通过按SW2或SW3启动或停止PWM
      - 如果调节电位器，则周期将发生变化

PIC18F外设
主同步串行端口（MSSP）

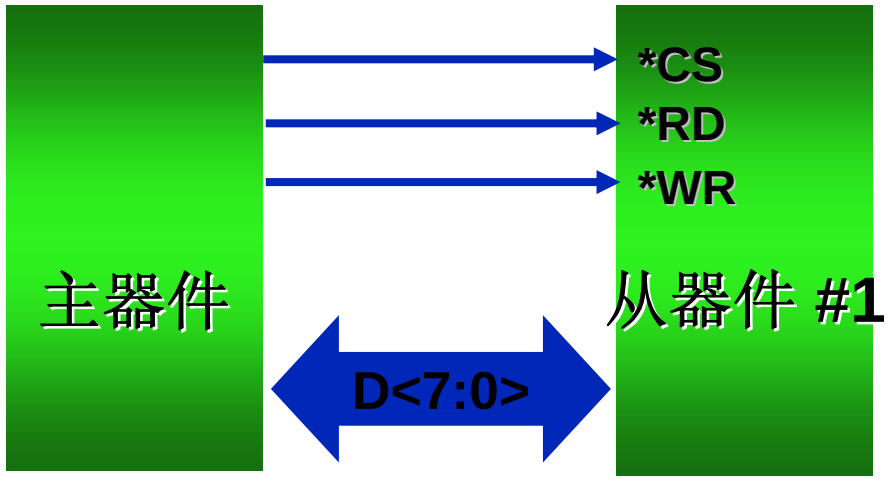
- 可工作在SPI或I²C™模式下
- SPI模式
  - 可编程波特率
  - 最大波特率（@ 40MHz）
    - 主器件： 10 Mbps
    - 从器件： 2.5 Mbps单字节发送
- I²C模式
  - 支持标准（100kHz）、快速（400kHz）和Microchip的1 MHz I²C标准
  - 硬件实现主器件/从器件
  - 在SCL和SDA引脚上的毛刺滤波器和斜率控制



SPI是摩托罗拉半导体的商标
I²C是菲利浦半导体的商标

PIC18F外设
并行从端口

- 8位多MCU总线
  - 一个主器件 — 多个从器件
  - 主器件可读/写一个或多个从器件
- 异步通信
- 高工作速度



实验6：使用I²C™

- 目标：
  - 理解MSSP配置
  - 理解I²C主工作模式
  - 理解与I²C从器件通信所需的步骤

I²C是菲利浦半导体的商标

实验6：使用 I²C™

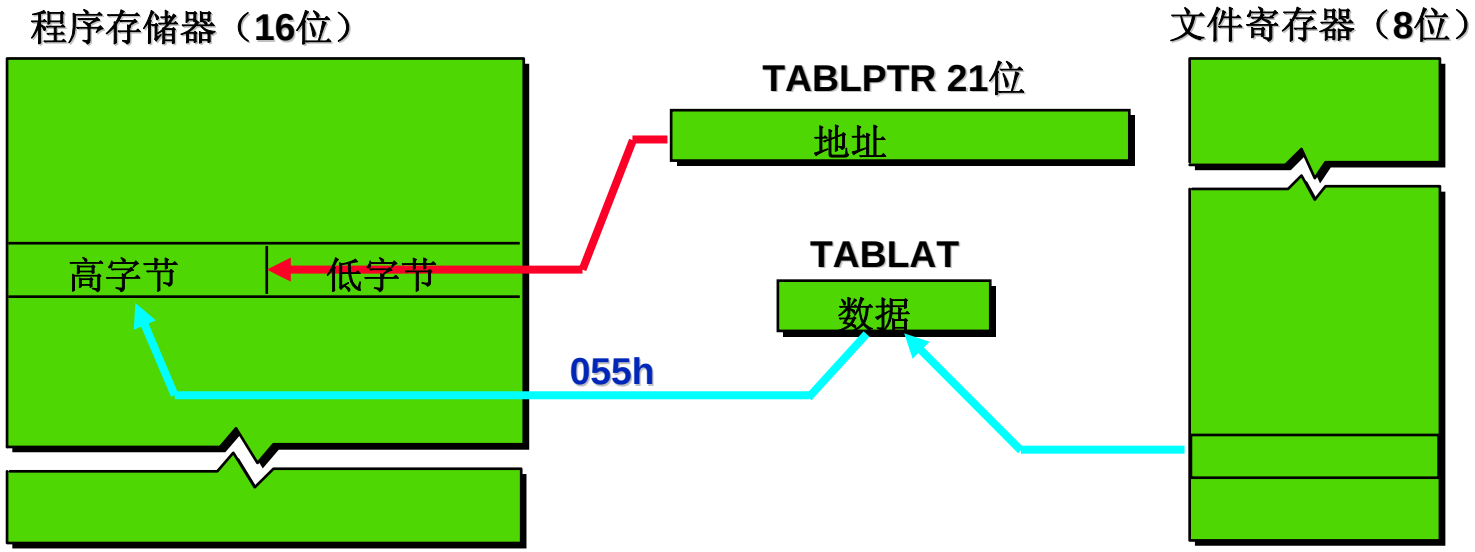
- 实验6的代码：
  - 打开项目**Lab6.mcp**
  - 在文件**Lab6_main.c**中编写C源代码以配置
    - CLK和SDA作为输入TRISC（TRISC<3:4> = 11）
    - 配置为主模式
      - SSPCON2 = 0x28；
    - 主模式的时钟 = (FOSC /(4 x (SSPADD + 1))
      - SSPADD = 0x63；（注：选择任何所需的波特率）
  - 使用**I2C_read()**函数发送读温度请求命令到温度传感器
  - 将接收到的温度数据转换为**BCD**（使用**Bin2BCD()**函数）
  - 使用**LCD**写函数在板上**LCD**上显示接收到的温度值

实验6：使用 I²C ™

- 预期的结果：
  - 完成实验后，将预期得到如下结果
    - LCD上显示出温度数据

PIC18F表操作

```
MOVLW   055h
MOVF    TABLAT
TBLWT*
```



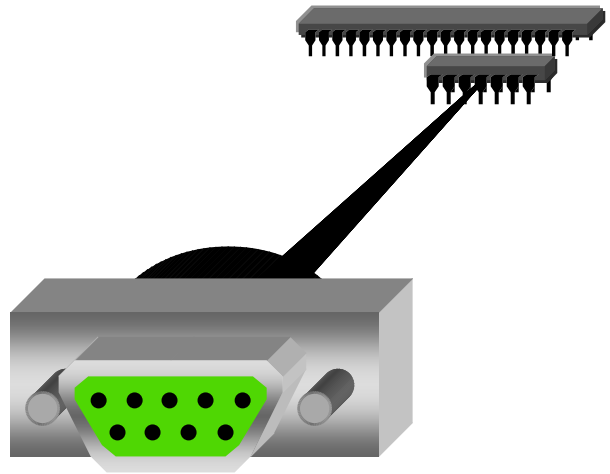
- TBLPTR：保存程序存储器地址的寄存器
- TABLAT：保存待读或写的数据的寄存器
- 从程序存储器读取传输数据到文件寄存器
- 从文件寄存器写传输数据到程序存储器
- 每次传输占用4个周期

PIC18F表指针操作

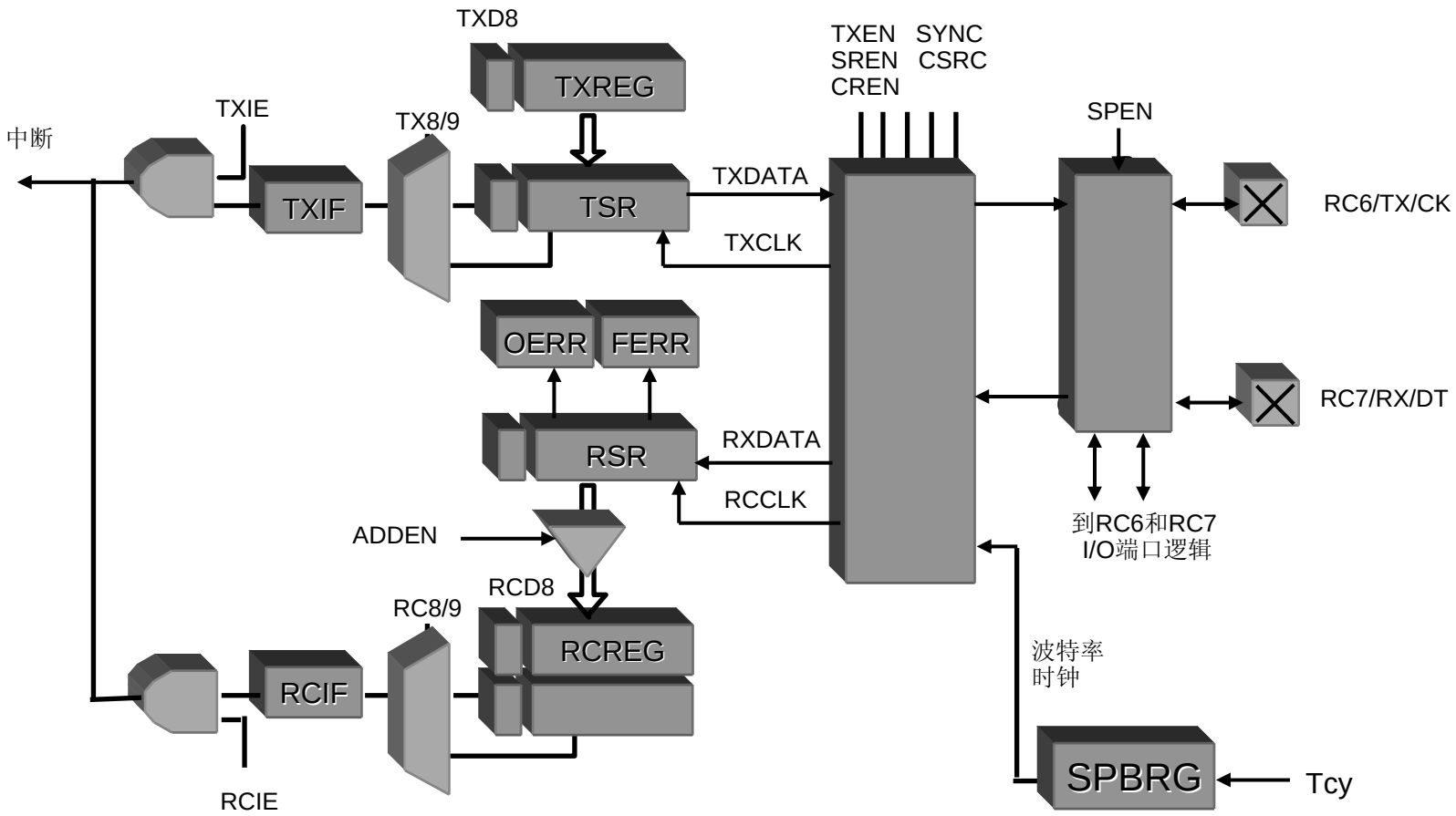
- TBLRD *　　表指针不会发生改变
- TBLRD *+　　表指针自动后递增
- TBLRD *-　　表指针自动后递减
- TBLRD +*　　表指针自动预递增

PIC18F外设
串行通信接口
（SCI/USART）

- 全双工异步通信或半双工同步通信
- 传输**8**位或**9**位数据
- 双重缓冲的发送和接收缓冲器
- 独立的发送和接收中断
- 首先发送和接收**LSB**
- 专用**16**位波特率发生器
- 最大波特率@ **40 MHz**
  - 同步：**10 Mbaud**
  - 异步：**625 Kbaud/2.5 Mbaud**
- **9**位可寻址模式
- 接收到字符时自动唤醒
- 自动波特率校准
- **12**位间隔字符发送



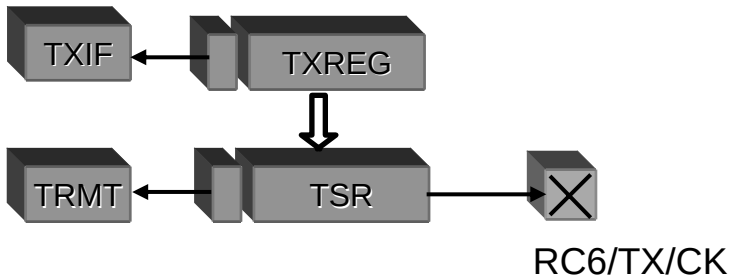
PIC18F外设
串行通信接口
USART框图



PIC18F外设
串行通信接口
TXIF和TRMT操作

- TXREG为空，将置1 TXIF
- 装载TXREG，复位TXIF

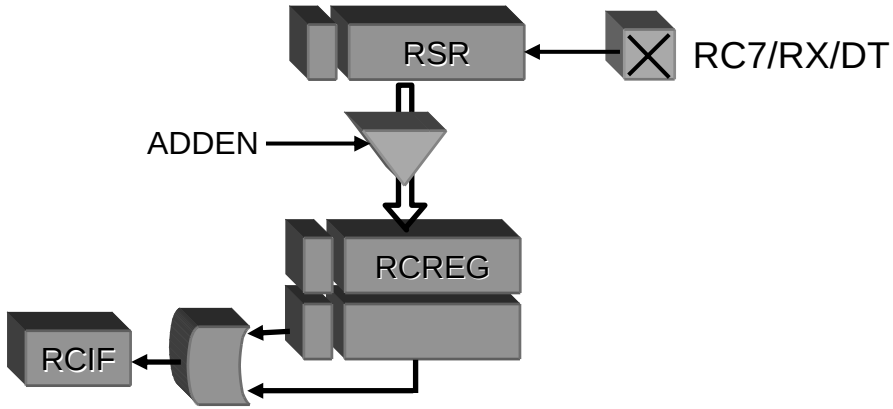
- TSR为空，将置1 TRMT
- 装载TSR，复位TRMT



- 如果TXREG被装载，且TRMT被置1，则数据将会被立即装入TSR。启动串行数据移位且TXIF将置1

PIC18F外设
串行通信接口
RCIF操作

- **RSR**接收带有有效启动/停止位的数据
- 数据被装入**RCREG FIFO**，且**RCIF = 1**
- 如果在处理第**1**个数据之前接收到**2**个数据，则新数据将被放置在**FIFO**的第**2**个单元
- 当接收中断时，读取第**1**个字节后如果另有字节位于**FIFO**中，则产生第**2**个**RCIF**中断



2级深FIFO

实验7：使用USART

- 目标：
  - 初始化**SFR**以设置**USART**外设
  - 理解使用**USART**发送和接收数据
  - 编写代码以处理中断事件
  - 理解表读/表写指令的用法

实验7：使用USART

- 实验7的代码：
  - 打开项目**Lab_7.mcp**
  - 在文件**Lab7_main.c**中编写C源代码以配置 **USART**
    - 异步通信模式，波特率为19,200 bps @ 10 MHz
    - 发送使能，连续接收
    - 禁止地址检测
    - 配置为高优先级中断源
  - 添加**I²C™**文件到项目（以读取温度数据）
  - 在‘**HighISR**’中添加代码以完成下列操作：
    - 检查USART接收中断
    - 将PC终端发来的数据回传
    - 当接收到键‘R’时发送温度数据
    - 当接收到键‘K’时发送常量字符串
  - 使用超级终端校验结果

实验7：使用USART

- 实验7的代码：
  - 这里是计算波特率的公式

SYNC	BRGH = 0 （低速）	BRGH = 1 （高速）
0	（异步）波特率 = Fosc/(64(X+1))	（波特率）= Fosc/(16(X+1))
1	（同步）波特率 = Fosc/(4(X+1))	NA

X = SPBRG中的值（0-255）

  - 单击如下链接以打开超级终端
    - 链接到超级终端

实验7：使用USART

- 预期的结果：
  - 完成实验后，将预期得到如下结果：
    - 在超级终端上将看到按下每个键的回传数据
    - 如果按下R键，将接收到温度数据
    - 如果按下S键，将接收到预先存储的常量字符串

PIC18F： 架构优点

- 优化的C编译器
  - 架构
    - 线性程序存储器寻址范围为2 MB
    - 线性数据存储器寻址范围为4 KB
    - 具有5种寻址模式的3个数据指针
  - 扩展指令集
    - 软件堆栈指针操作
    - 动态分配和取消分配软件堆栈
    - 操作位于软件堆栈中的变量
    - 直接函数指针调用

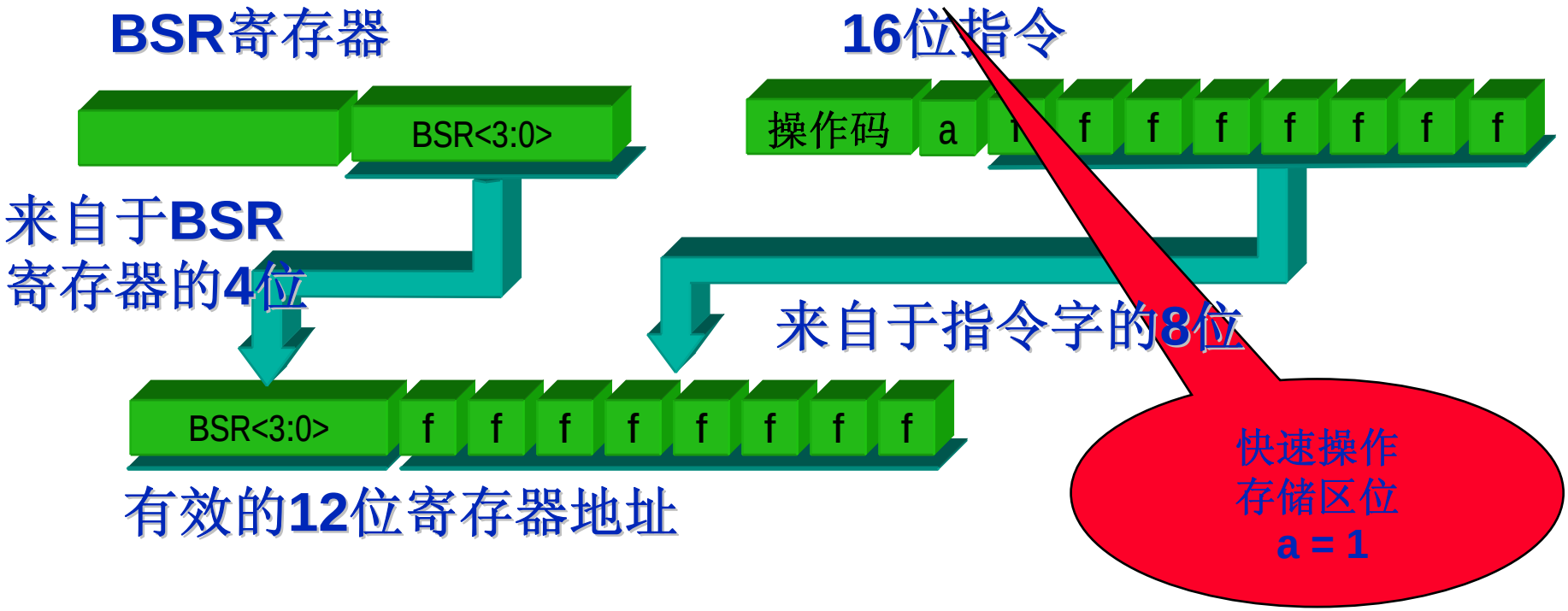
PIC18F：扩展模式

- 通过*XINST* 配置位使能/禁止
- 立即数偏移量变址寻址模式和...

指令	说明
ADDFSR f, k	立即数加到**FSR**
ADDULNK k	立即数加到**FSR2**并返回
CALLW	使用**WREG**调用子程序
MOVSF Zs, fd	将以**zs**为偏移量（源地址为**FSR2+Zs**）的地址中的内容送入**fd(**目标**),** ((FSR2) + zs) → fd
MOVSS Zs, Zd	将以**zs**为偏移量（源地址为**FSR2+Zs**）的地址中的内容送入以**fd**为偏移量（目地址为**FSR2+Zd**）的地址中 ((FSR2) + zs) → ((FSR2) + zd)
PUSHL k	在**FSR2**指向的空间上存储立即数，**FSR2**指针递减
SUBFSR f, k	从**FSR**中减去立即数
SUBULNK k	从**FSR2**中减去立即数并返回

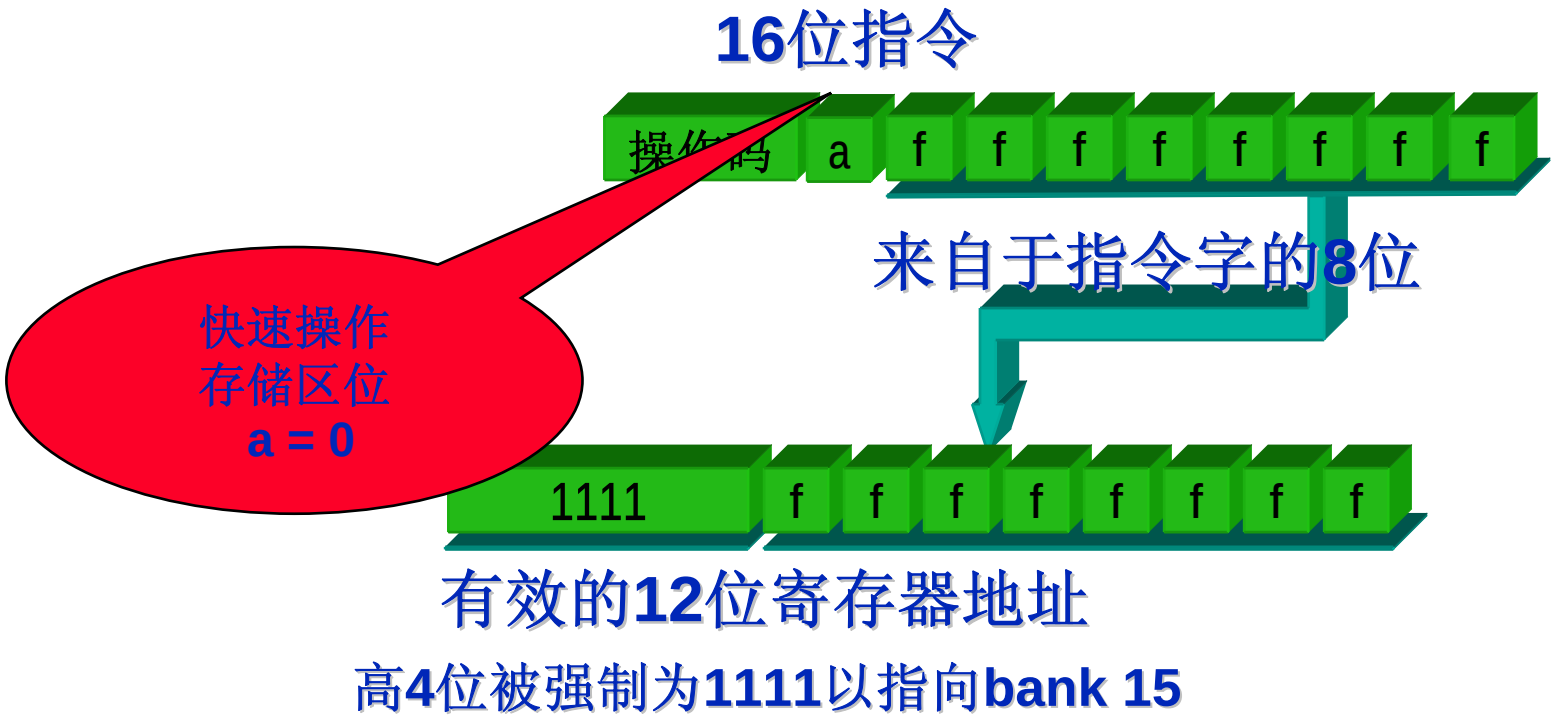
PIC18F架构
访问数据存储器
（扩展模式*XINST = 1*）

- 当**a**位（快速操作存储区位）**= 1**时的指令格式示例



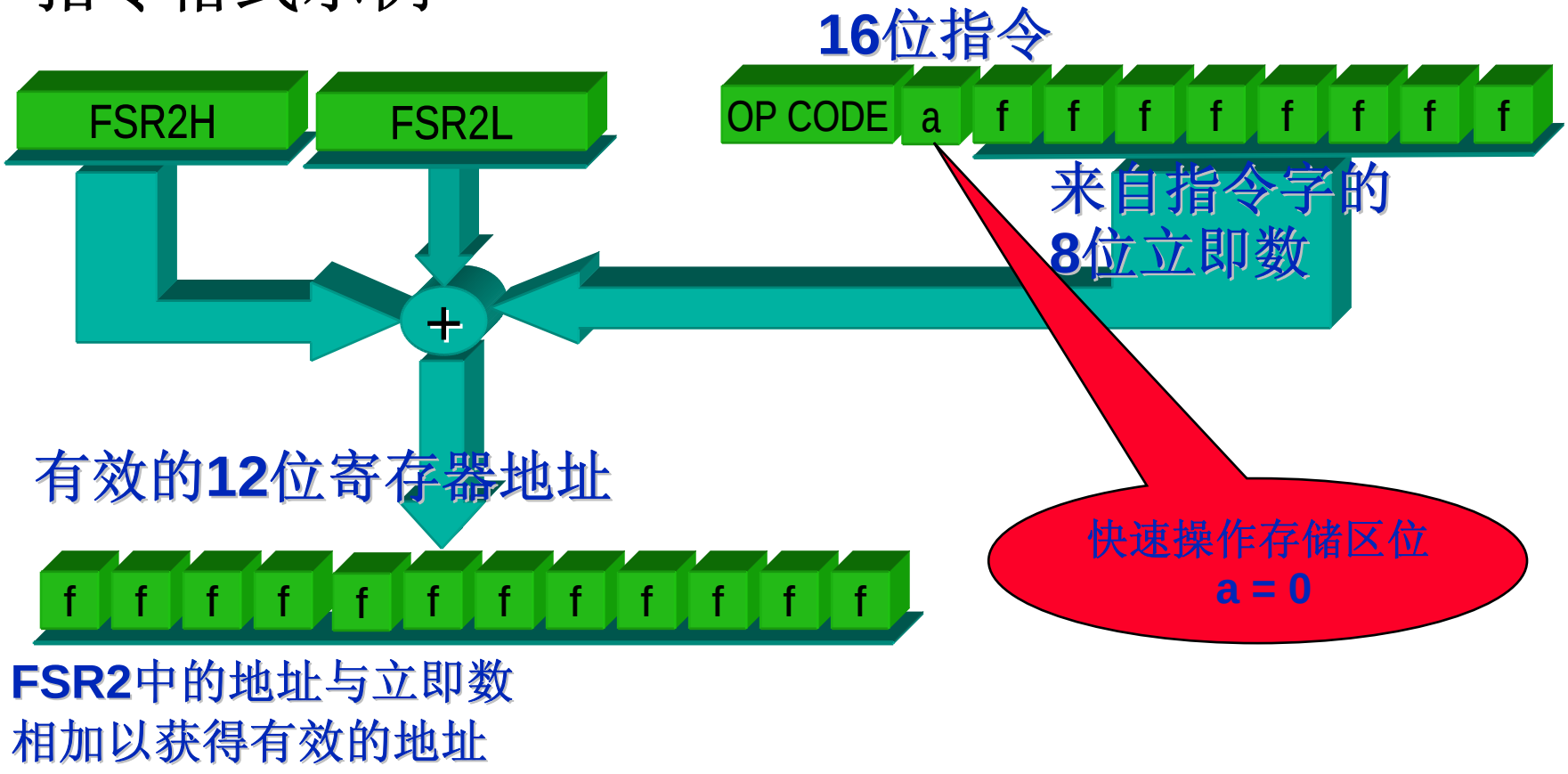
PIC18F架构
访问数据存储器
（扩展模式*XINST = 1*）

- 当**a**位（访问存储区位）**= 0**且**f ≥ 60h**时的指令格式示例



PIC18F架构
立即数偏移量变址寻址
（扩展模式*XINST = 1*）

- 当**a**位（快速操作存储区位）**= 0**且**f ≤ 5Fh**时的指令格式示例



实验8： 扩展架构的优点演示

- 目标：
  - 第1部分：使用MPLAB® IDE
    - 设置选项以配置为标准模式和扩展模式
  - 第2部分：理解和评估扩展架构的优点

实验8： 扩展架构的优点演示

- 说明：
  - 打开**Lab8_Extended.mcp**
    - 在配置中禁止扩展模式选项
    - 在MCC18编译器中禁止扩展模式
    - 包含标准模式链接器文件‘18f4520i_e.lnk’
    - 编译项目
  - 打开**Lab8_Standard.mcp**
    - 在配置中使能扩展模式选项位
    - 在MCC18编译器中使能扩展模式
    - 包含扩展模式链接器文件‘18f4520i.lnk’
    - 编译项目
  - 比较代码大小和执行时间

实验8： 扩展架构的优点演示

- 表格：

	标准模式	扩展模式
代码大小		
执行时间		

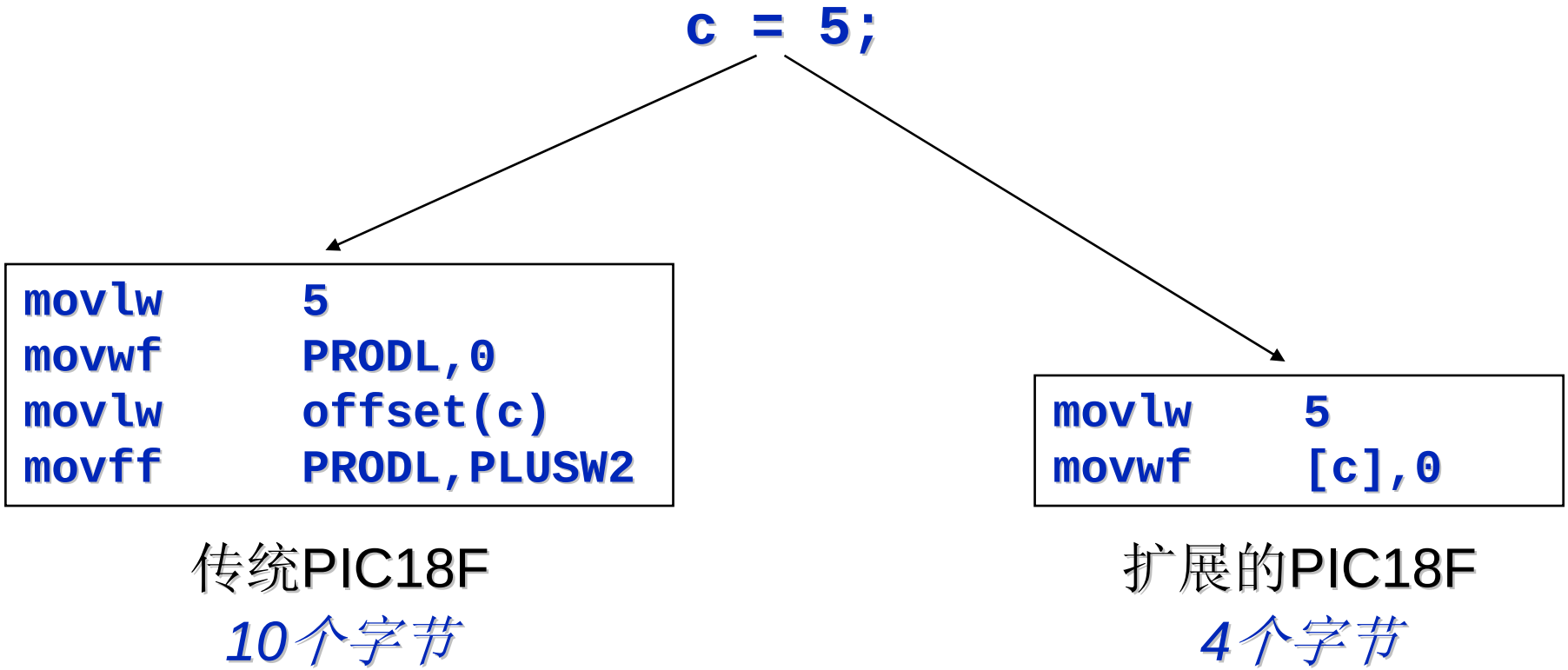
- 结论：
  - 扩展模式减小了代码大小
  - 扩展模式缩短了程序的执行时间

UNIVERSITY OF MICROCHIP
UOFM
MASTERS 2007
第八届中国技术精英年会

实验8讲解

PIC18F：扩展架构的优点

改进简单赋值的示例：



PIC18F：扩展架构的优点

改进函数序言的示例：

单存储区堆栈：

```
movlw     FrameSize
addwf     FSR1L,1,0
```

多存储区堆栈：

```
movlw     FrameSize
addwf     FSR1L,0,0
bnc       PC+6
setf      FSR1L,0
movf      POSTINC1,1,0
movwf     FSR1L,0
```

传统PIC18F

4至12字节

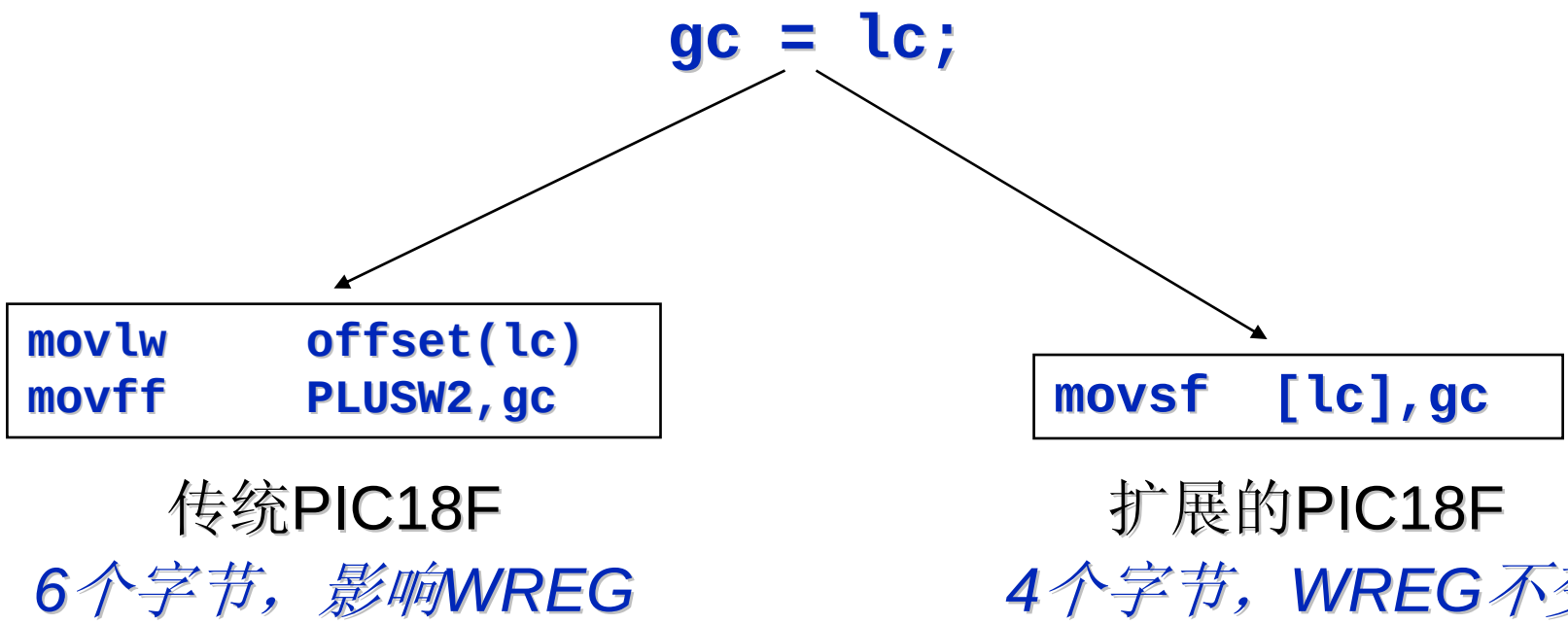
```
addfsr FrameSize,1
```

扩展的PIC18F

2个字节

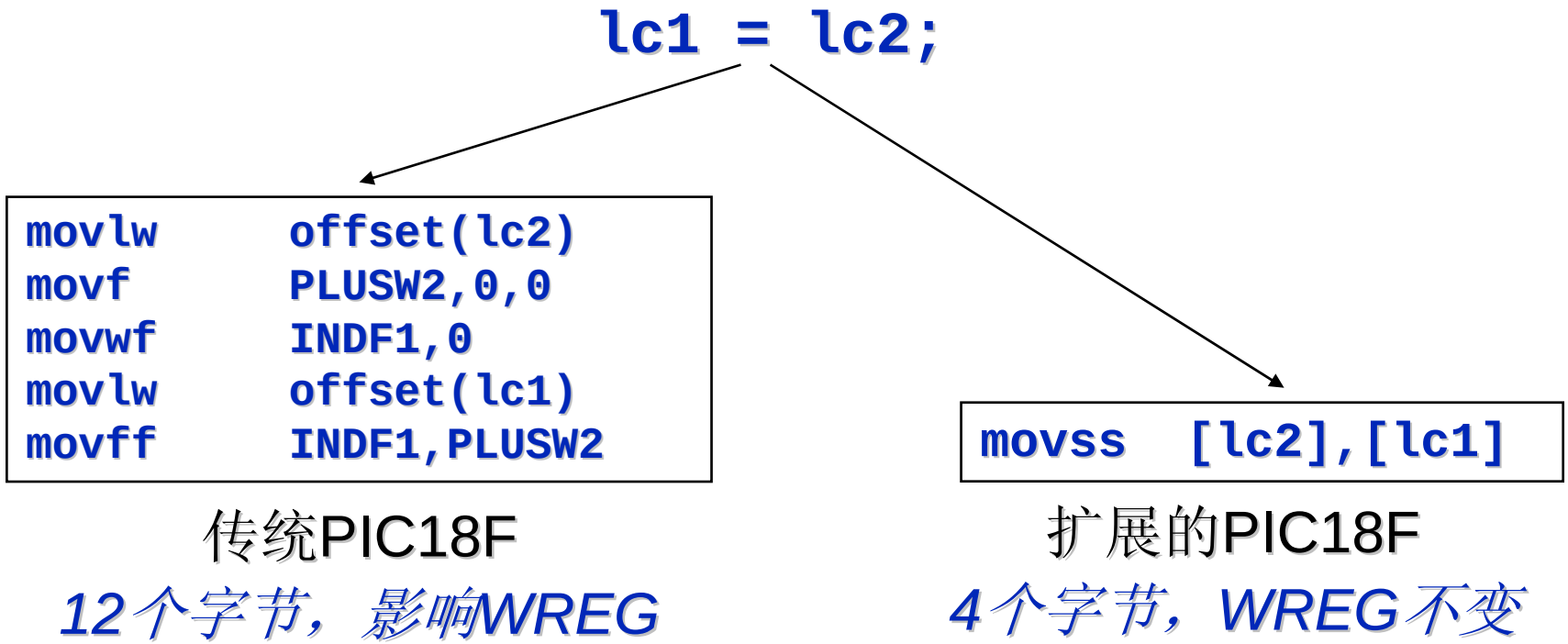
PIC18F：扩展架构的优点

对堆栈到全局变量的传送操作进行改进的示例：



PIC18F：扩展架构的优点

对堆栈到堆栈的传送操作进行改进的示例：



PIC18F：扩展架构的优点

改进函数指针调用的示例：

```
fn();
```

```
bra     PC+12
movff   fn+2,PCLATU
movff   fn+1,PCLATH
movlb   fn
movf    fn,0,1
movwf   PCL,0
rcall   PC-10
```

传统PIC18F

18个字节，2次转移

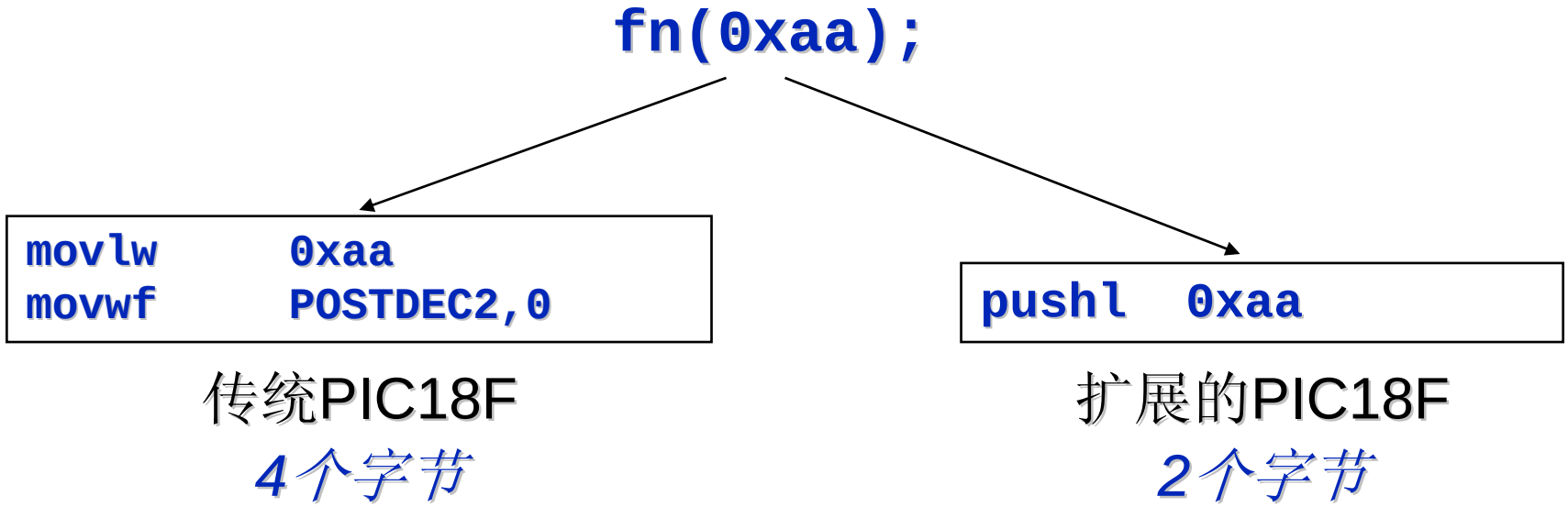
```
movff fn+2,PCLATU
movff fn+1,PCLATH
movlb fn
movf  fn,0,1
callw
```

扩展的PIC18F

14个字节，0次转移

PIC18F：扩展架构的优点

对传递立即数参数的操作进行改进的示例：



UNIVERSITY OF MICROCHIP
UOFM
MASTERS 2007
第八届中国技术精英年会

振荡器和功耗管理模式

PIC18F：时钟系统

- 时钟源
  - 主时钟源
    - 固定选择
    - LP、XT、HS、RC、EC和内部RC振荡器
  - 辅助时钟源
    - Timer1振荡器 — 固定频率
    - 需要用于实时时钟时基
  - 内部RC振荡器
    - INTOSC（8 MHz）时钟源
      - 4、2和1 MHz；500、250、125和31 kHz
    - INTRC（31 kHz）时钟源

PIC18F：时钟系统

- 新的内部**RC**振荡器
  - **2**个独立的**RC**时钟源
    - 8 MHz（INTOSC）
    - 31 kHz （INTRC）
  - **2-31 kHz**时钟源
  - 内部**8**和**4 MHz**的**INTOSC**可通过**PLL**倍频为
    - 16或32 MHz
  - 更改**IRCF<2:0>**位直接选择不同的**INTOSC** 后分频比值

PIC18F：功耗管理模式

3个类别

运行 – 3个时钟源

空闲 – 3个时钟源

休眠 – 无时钟源

总共 = 7种模式

PIC18F：功耗管理模式

- **PRI_RUN模式**
  - 配置字定义主时钟源
    - FOSC3:FOSC0 （_CONFIG1H<3:0>）10种模式
      - 晶振 – LP、XT、HS和HSPLL
      - 外部时钟 – EC和ECIO
      - 外部RC振荡器 – RC和RCIO
      - 内部RC振荡器 – INTIO1和INTIO2

- **SEC_RUN模式**
  - 其他**PIC18**控制器中的时钟切换机制
  - 使用**Timer1**时钟源，主振荡器被禁止
- **RC_RUN模式**
  - **IRCF<2:0>**选择时钟速率
  - 如果频率 ≠ **31 kHz**，则在**1us**（典型值）延时后置**1 IOFS**

UNIVERSITY OF MICROCHIP
UOFM
MASTERS 2007
第八届中国技术精英年会

PIC18F
的特殊性能

PIC18F的特殊性能

- 宽工作电压范围： **2.0V至5.5V**
- 可承受**100,000**次（典型值）擦/写的增强型闪存程序存储器
- 可承受**1,000,000**次（典型值）擦/写的数据**EEPROM**存储器
- 闪存/数据**EEPROM**数据保存时间：**100**年（典型值）

PIC18F的特殊性能

- 扩展型看门狗定时器（**WDT**）：
  - 从**4 ms**到**131s**的可编程周期
- 在线串行编程 （ **In-Circuit Serial Programming™**，**ICSP™**） 编程能力
- 可编程高/低电压检测（**HLVD**）模块
  - 支持在检测到高/低电压条件时中断
- 可编程欠压复位
  - 在配置过程中使能或
  - 通过软件使能选项使能

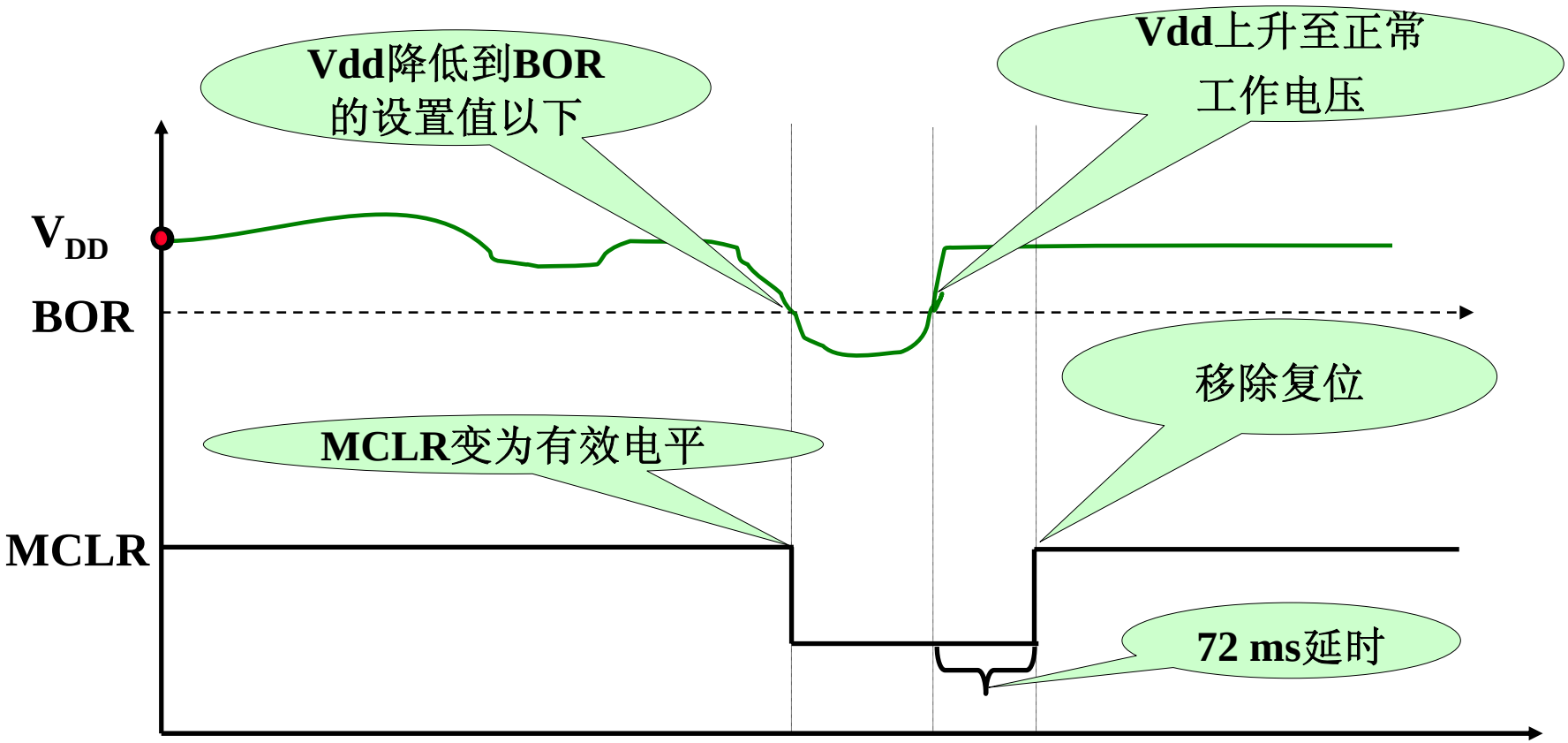
PIC18F的特殊性能
可编程低电压检测

- 提供“预警”
- 可编程内部或外部参考电压
  - 最多**14**个内部参考电压（**2 – 4.77V**）
- 在休眠模式下工作
  - 低电压条件唤醒/中断**MCU**
- 软件控制使能/禁止
  - 对于低功耗应用有用

PIC18F的特殊性能
可编程欠压复位

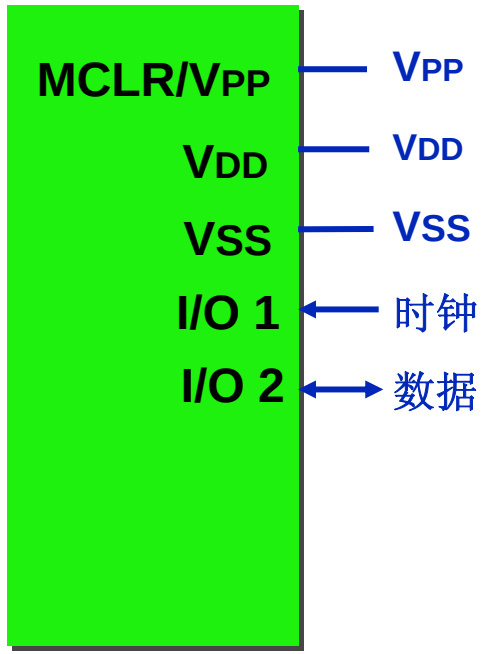
- 监视工作电压范围
- 在预定义范围内复位**MCU**
  - 当**Vdd**恢复正常工作电压后（**+ 72 ms**）移除复位
- 可编程内部参考电压
  - 最多**4**个参考电压（**2.0**、 **2.7**、**4.2**和**4.5**）
- 通过配置寄存器使能

PIC18F的特殊性能
可编程欠压复位



PIC18F的特殊性能
ICSP™编程能力

- 在线编程方法
- 仅使用2个引脚发送/接收数据
- 不影响正常工作
- **ICSP™的优点**
  - 降低现场升级的成本
  - 在生产时校准和序列化系统
  - 减少处理



PIC18F的特殊性能
看门狗定时器（WDT）

- 从软件故障中恢复
- 若未能准时处理，则复位MCU
  - 软件必须周期性的将其清零（CLRWDT）
- 可编程周期
  - **4 ms到131.0 s**（典型值）
- 配置控制的后分频器
- 通过配置寄存器或软件使能
- 将CPU从休眠/空闲模式唤醒

PIC18F的特殊性能
复位

- PIC18复位
  - 上电复位（POR）
  - 正常工作期间的MCLR复位
  - 可编程欠压复位（BOR）
  - 看门狗定时器（WDT）复位（执行期间）
  - RESET指令
  - 堆栈满复位
  - 堆栈下溢复位
- 对于所有复位，PC均跳转到地址0处的向量

PIC18F的特殊性能
复位寄存器

- 复位后，**PC将指向地址0x000000**
- 每次复位后，将对下列位产生影响：
  - **POR = ‘0’**：*上电复位*
  - **BOR = ‘0’和POR = ‘1’**： *BOR复位*
  - **TO = ‘0’**：*WDT复位*
  - **RI = ‘0’**：*RESET指令*
  - **STKFUL = ‘1’**：*堆栈上溢复位*
  - **STKUNF = ‘1’**：*堆栈下溢复位*
  - **POR、BOR、 TO和RI = ‘1’ 且STKFUL和STKUNF = ‘0’**：*MCLR复位*

实验9：如何了解在实际应用中导致复位的原因

- 目标：
  - 理解PIC18复位源
  - 理解每个复位的意义
  - 识别应用中的复位源

- 此演示由2部分组成：
  - 第1部分：编写软件以处理复位类型
  - 第2部分：模拟产生复位的条件

实验9：如何了解在实际应用中导致复位的原因

- 实验9的代码：
  - 打开项目**Lab_9.mcp**
  - 在文件**Lab_main.c**中编写**C**源代码以识别并处理每种复位，如下所示：
    - POR复位
      - 检查RCON寄存器中的POR和BOR，如果仅POR位未被置1，则为POR复位
      - 向RCON和STKPR寄存器中写入无效值
      - 调用名为“POR_Occured” 的函数，它将在板上显示器上显示出复位类型

实验9：如何了解在实际应用中导致复位的原因

- 实验9的代码：（续…）
  - BOR复位
    - 检查RCON寄存器中的POR和BOR位，如果两者都未被置1，则为BOR复位
    - 向RCON和STKPR中写入无效值
    - 调用名为“BOR_Occured”的函数，它将在板上显示器上显示出复位的类型
  - MCLR复位
    - 检查RCON和STKPTR寄存器中的POR和BOR位，如果2个寄存器的复位状态位都不变，则为MCLR复位
    - 向RCON和STKPR中写入无效值
    - 调用名为“MCLR_Occured”的函数，它将在板上显示器上显示出复位的类型

实验9：如何了解在实际应用中导致复位原因

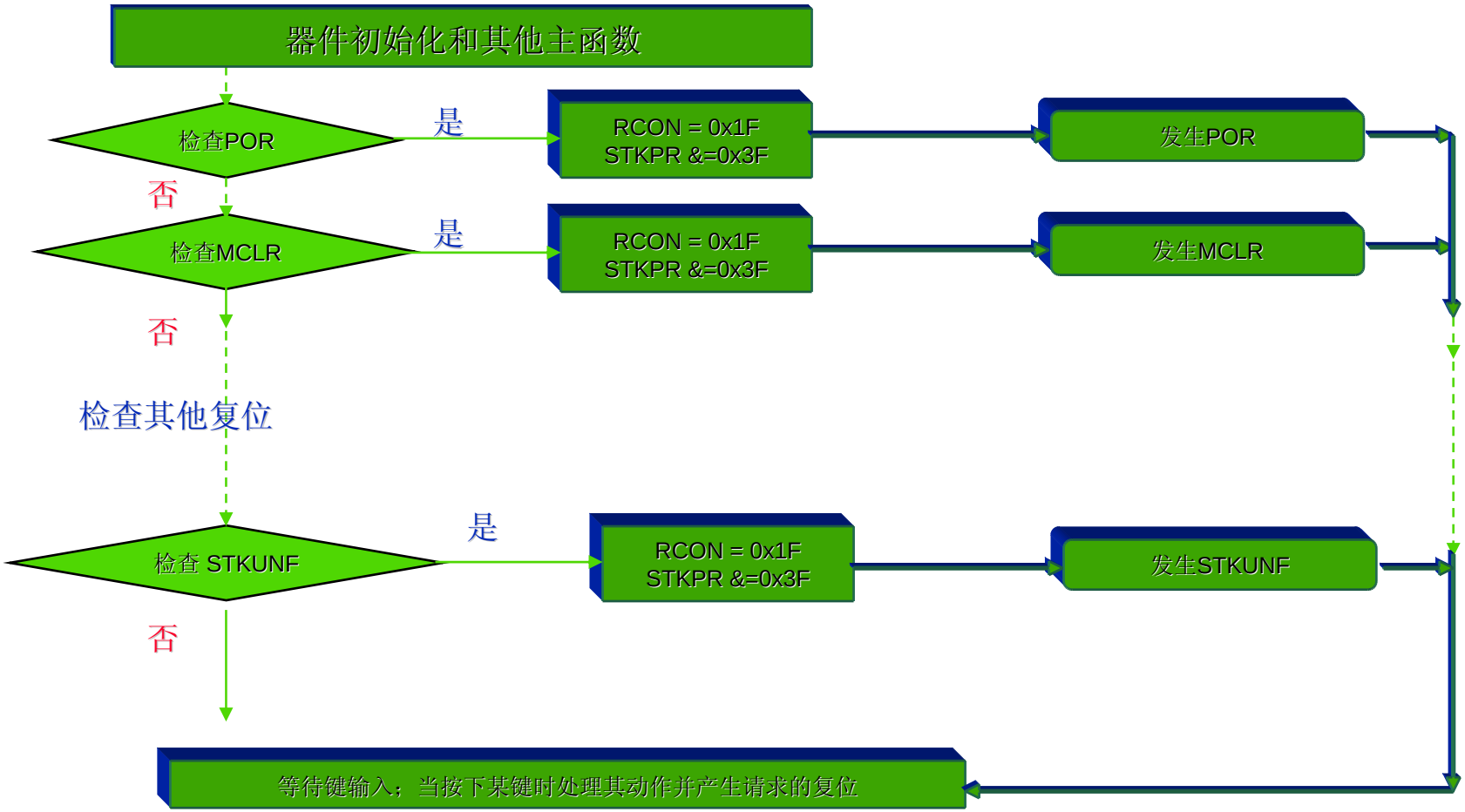
- 实验9的代码：（续…）
  - 看门狗定时器复位
    - 检查RCON寄存器中的TO位，如果该位未被置1，则为看门狗复位
    - 向RCON和STKPR中写入无效值
    - 调用一个名为“WDT_Occured”的函数，它将在板上显示器上显示出复位的类型
  - 软件复位
    - 检查RCON寄存器中的RI位，如果该位未被置1，则表示此复位是由于执行了软件复位指令引起的
    - 向RCON和STKPR中写入无效值
    - 调用一个名为“RI_Occured”的函数，它将在板上显示器上显示出复位的类型

实验9：如何了解在实际应用中导致复位的原因

- 实验9的代码（续…）
  - 堆栈满复位
    - 检查STKPTR寄存器中的STKFUL位，如果该位置1，则为堆栈溢出复位
    - 向RCON和STKPR中写入无效值
    - 调用一个名为“StkOvFlo_Occured”的函数，它将在板上显示器上显示出复位的类型
  - 堆栈下溢复位
    - 检查STKPTR寄存器中的STKUNF位，如果该位置1，则为堆栈下溢复位
    - 向RCON和STKPR中写入无效值
    - 调用一个名为“StkUnFlo_Occured”的函数，它将在板上显示器上显示出复位的类型

实验9：如何了解在实际应用中导致复位的原因

实验9的流程图：



实验 9：如何了解在实际应用中导致复位的原因

- 实验 9： 第2部分模拟复位条件
  - 编译项目
  - 使用MPLAB® ICD 2烧写单片机
  - 切断演示板的电源
  - 从演示板上拔下ICD电缆

实验 9：如何了解在实际应用中导致复位的原因

- 实验 9： 第2部分模拟复位条件

现在打开演示板的电源

在LCD上将显示“Power ON ”复位消息

实验 9：如何了解在实际应用中导致复位的原因

- 实验 9： 第2部分模拟复位条件

现在按下再松开演示板上的“复位”按钮

在LCD上将显示“MCLR”复位消息

实验 9：如何了解在实际应用中导致复位的原因

- 实验 9： 第2部分模拟复位条件

现在按演示板上的“SW1”开关

在LCD上将显示“Watchdog”复位消息

实验 9：如何了解在实际应用中导致复位的原因

- 实验 9： 第2部分模拟复位条件

现在按演示板上的“SW1”开关

在LCD上将显示“Software”复位消息

实验 9：如何了解在实际应用中导致复位的原因

- 实验 9： 第2部分模拟复位条件

现在按演示板上的“SW1”开关

在LCD上将显示“Stack Full ” 复位消息

实验 9：如何了解在实际应用中导致复位的原因

- 实验 9： 第2部分模拟复位条件

现在按演示板上的“SW1”开关

在LCD上将显示“Stack Under flow”复位消息

总结

- 讨论了架构概述
  - **PIC18F**的架构
  - 编程模型
  - 指令集概述
  - 中断处理和中断响应延时
- 回顾了基本外设
  - **I/O**端口
  - 开发工具
  - 比较器和参考电压
  - **ADC**

总结

- 定时器
- **CCP （捕捉、比较和PWM）**
- **MSSP（I²C™和SPI/Microwire）**
- 表读和表写操作
- **USART**

- 还讨论了：
  - 扩展的架构
  - 振荡器和节能模式
  - **PIC18F控制器的特殊性能**

总结

- 执行了下列实验：
  - IO口初始化以及读/写端口
  - 初始化和ADC转换
  - 时间配置
  - 产生PWM
  - 捕捉输入信号
  - I²C™模块配置
  - 标准指令集和扩展指令集的对比
  - 复位识别

谢谢！

商标

Microchip的名称和徽标组合、**Microchip**徽标、**Accuron**、**dsPIC**、**KeeLoq**、**KeeLoq**徽标、**microID**、**MPLAB**、**PIC**、**PICmicro**、**PICSTART**、**PRO MATE**、**rfPIC**和**SmartShunt**均为**Microchip Technology Incorporated**在美国和其他国家或地区的注册商标。

AmpLab、**FilterLab**、**Linear Active Thermistor**、**Migratable Memory**、**MXDEV**、**MXLAB**、**SEEVAL**、**SmartSensor**和**The Embedded Control Solutions Company**均为**Microchip Technology Incorporated**在美国的注册商标。

Analog-for-the-Digital Age、**Application Maestro**、**CodeGuard**、**dsPICDEM**、**dsPICDEM.net**、**dsPICworks**、**dsSPEAK**、**ECAN**、**ECONOMONITOR**、**FanSense**、**FlexROM**、**fuzzyLAB**、**In-Circuit Serial Programming**、**ICSP**、**ICEPIC**、**Mindi**、**MiWi**、**MPASM**、**MPLAB Certified**徽标、**MPLIB**、**MPLINK**、**PICkit**、**PICDEM**、**PICDEM.net**、**PICLAB**、**PICtail**、**PowerCal**、**PowerInfo**、**PowerMate**、**PowerTool**、**REAL ICE**、**rfLAB**、**Select Mode**、**Smart Serial**、**SmartTel**、**Total Endurance**、**UNI/O**、**WiperLock**和**ZENA**均为**Microchip Technology Incorporated**在美国和其他国家或地区的商标。

SQTP是**Microchip Technology Incorporated**在美国的服务标记。

在此提及的所有其他商标均为各持有公司所有。