定时器 (Timer)

本节内容

- 定时器的基本工作模式
- MSP430定时功能及其实现
- 看门狗定时器
  - WDT的操作
  - 看门狗定时器的中断控制功能
  - 看门狗应用举例
- 16位定时器 A
  - 定时器A的特性
  - Timer_A结构
  - Timer_A工作原理
  - Timer_A典型应用

定时器 ——工作模式（1/1）

MSP430X5XX / 6XX系列单片机的通用定时器共有4种计数模式，以定时器A为例，（TAxCTL 寄存器中的 MCx 位）

MCx	模式	说明
00	停止模式	定时器停止
01	增计数模式	定时器重复从 0 计数到 TAxCCR0
10	连续计数模式	定时器器重复从 0 计数到 0FFFFh
11	增/减计数模式	定时器重复从 0 增计数到 TAxCCR0 再减计数到 0

定时器工作模式 —— 停止模式（1/1）

- 停止模式

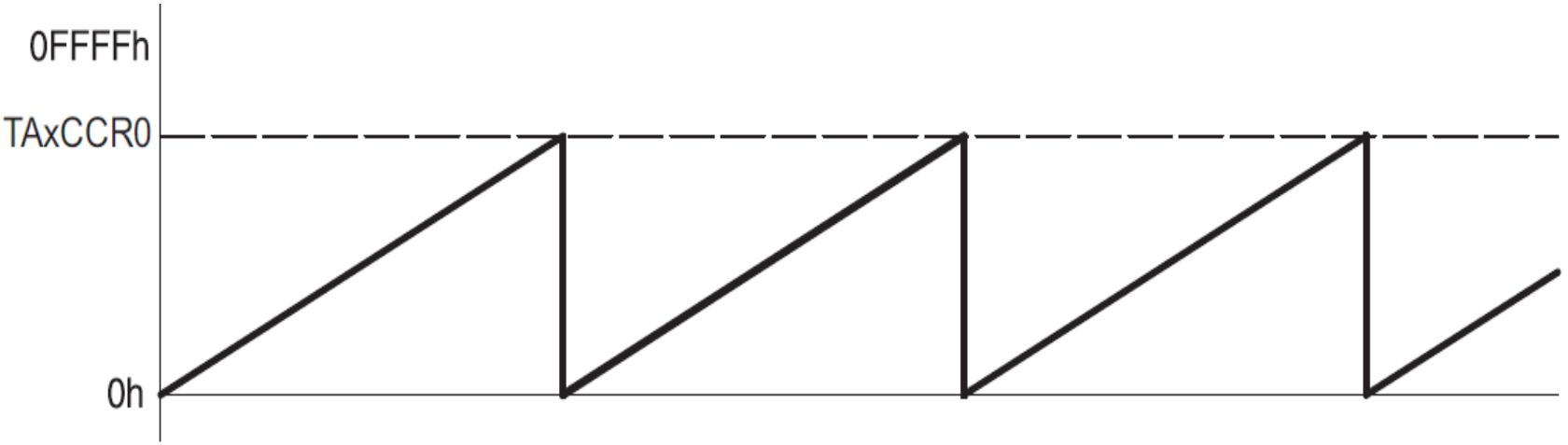
  - 用于定时器暂停，并不发生复位，所有寄存器现行的内容在停止模式结束后都可用。

  - 当定时器暂停后重新计数时，计数器将从暂停时的值开始以暂停前的计数方向计数。

  - 例如，停止模式前，Timer_A工作于增/减计数模式并且处于下降计数方向，停止模式后，Timer_A仍然工作于增/减计数模式。重新计数时，从暂停前的状态开始继续沿着下降方向开始计数。

定时器工作模式 ——增计数模式 （1/3）

- 捕获/比较寄存器TAxCCR0用作Timer_A增计数模式的周期寄存器。
- 计数器TAxR与TAxCCR0的值相等(或TAxR大于TAxCCR0的值)时，定时器 TAxR将立即重新从 0 开始计数。
- 下图说明了增计数模式的计数过程。



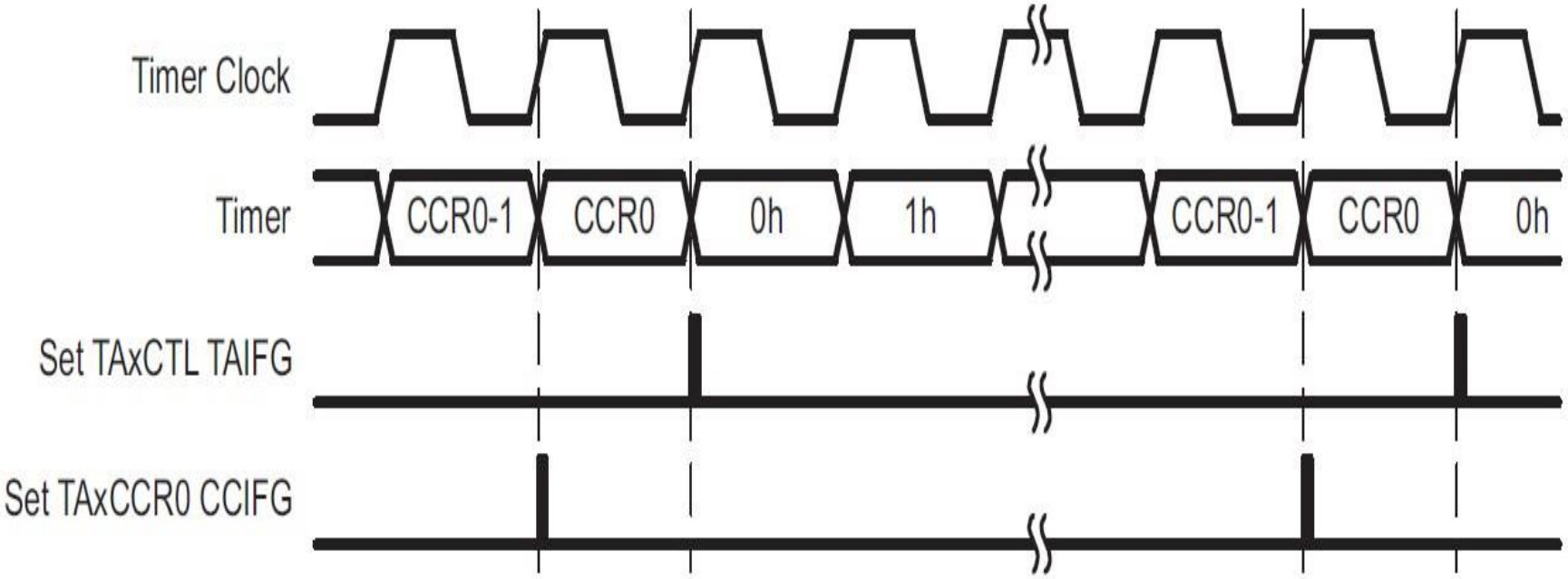
定时器工作模式 ——增计数模式 （2/3）

运行时改变捕获/比较寄存器TAxCCR0

- 如果新的计数周期大于或者等于旧的计数周期或者大于当前计数值，定时器将一直计数到新的计数周期。
- 如果新的计数周期小于当前计数值，计数值将减至 0。但是，在定时器回到 0 之前会有一次计数。

定时器工作模式 ——增计数模式 （3/3）

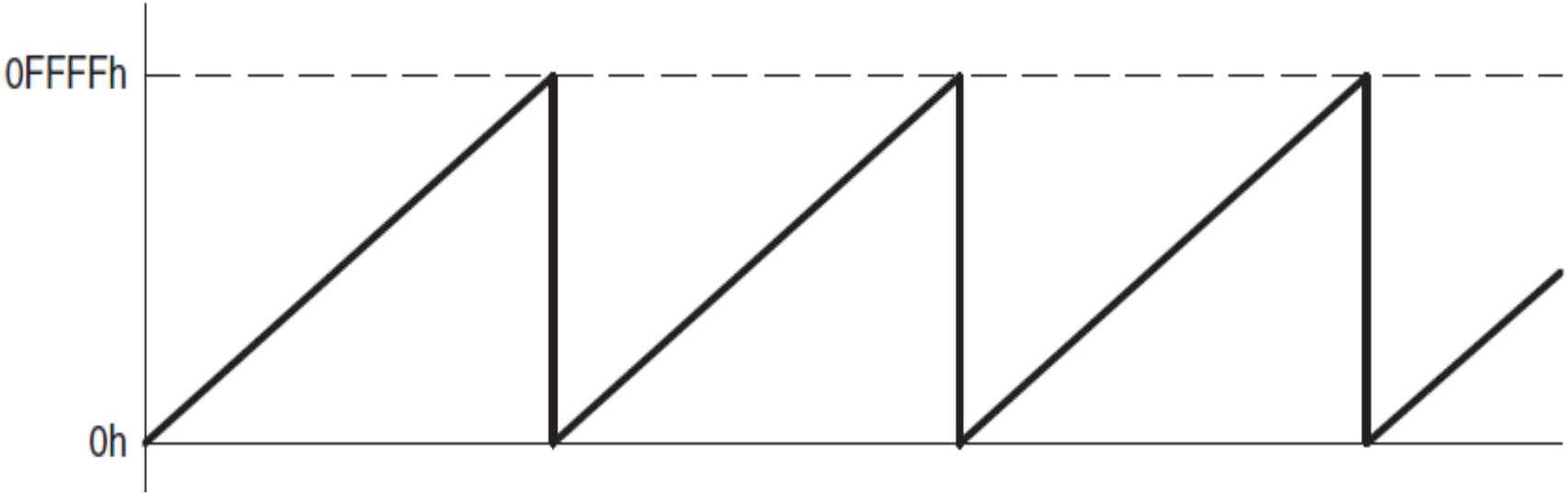
- 当定时器计数到 TAxCCR0 时，设置标志位TAxCCR0 CCIFG（捕获比较中断标志）为1，而当定时器从TAxCCR0计数到0时，设置标志位TAIFG（定时器溢出标志）位为1。中断标志位的设置过程，如下图所示。



定时器工作模式 ——连续数模式 （1/3）

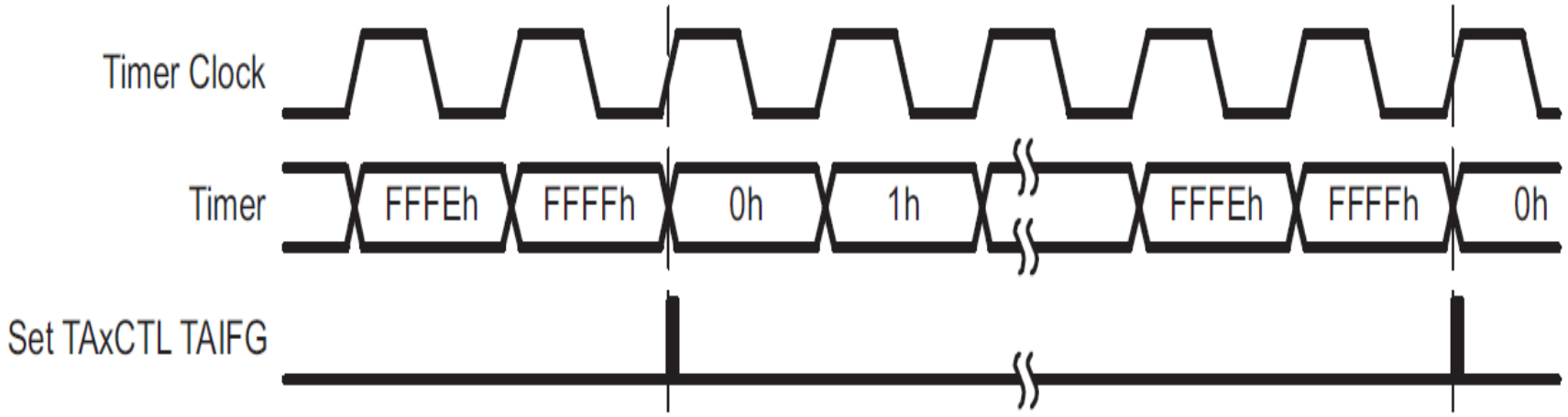
- 此模式下，定时器从当前值计数到0FFFFH后，又从0开始重新计数。

- 如下图所示，此时捕获/比较寄存器 TAxCCR0 和其它捕获/比较寄存器的工作方式相同。



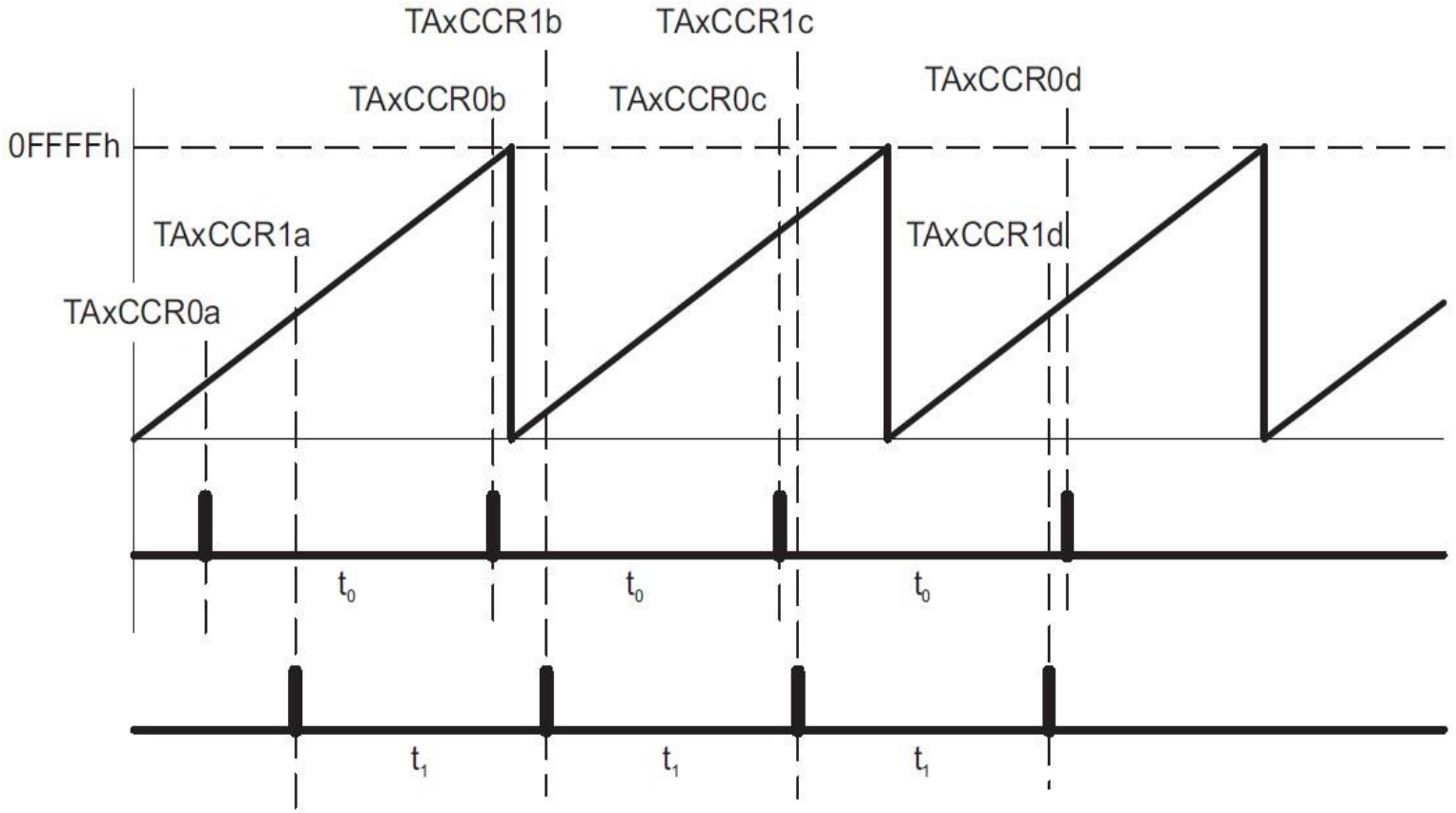
定时器工作模式 ——连续数模式 （2/3）

- 标志位的设置过程，如下图所示：当定时器从 0FFFFh 计数到 0 时，中断标志 TAIFG 置位。



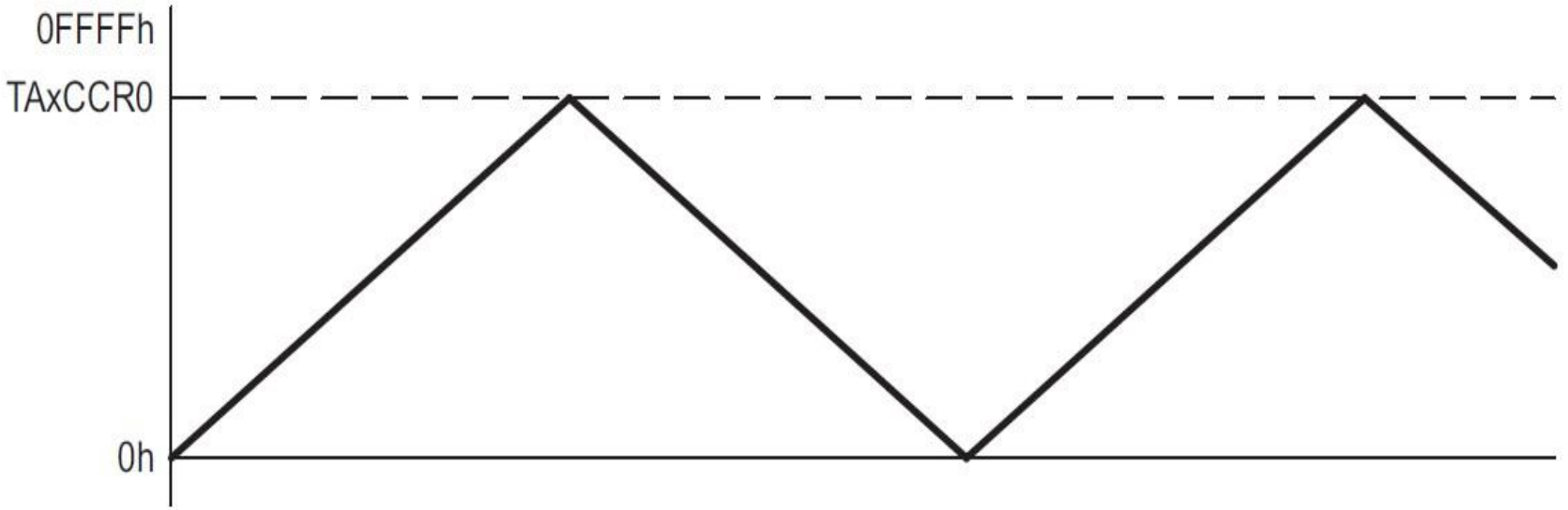
定时器工作模式 ——连续数模式 （3/3）

➢连续计数模式的典型应用：产生多个定时信号：通过中断处理程序在相应的比较寄存器TAxCCRx上加上一个时间差来实现。这个时间差是当前时刻（即相应的TAxCCRx中的值）到下一次中断发生时刻所经历的时间，如下图所示。



定时器工作模式 ——增/减计数模式 （1/3）

- 该模式下，定时器先增计数到TAxCCR0的值，然后反向减计数到0。
- 计数周期仍由TAxCCR0定义，它是TAxCCR0计数器数值的2倍。
- 增/减计数模式时计数器中数值的变化情况如图所示。



定时器工作模式 ——增/减计数模式 （2/3）

运行时改变寄存器 TAxCCR0周期

- 当计数器正在运行且在减计数方向时改变 TAxCCR0 的值，定时器将会继续减计数方向到 0。定时器减到 0 后，新的周期才有效。

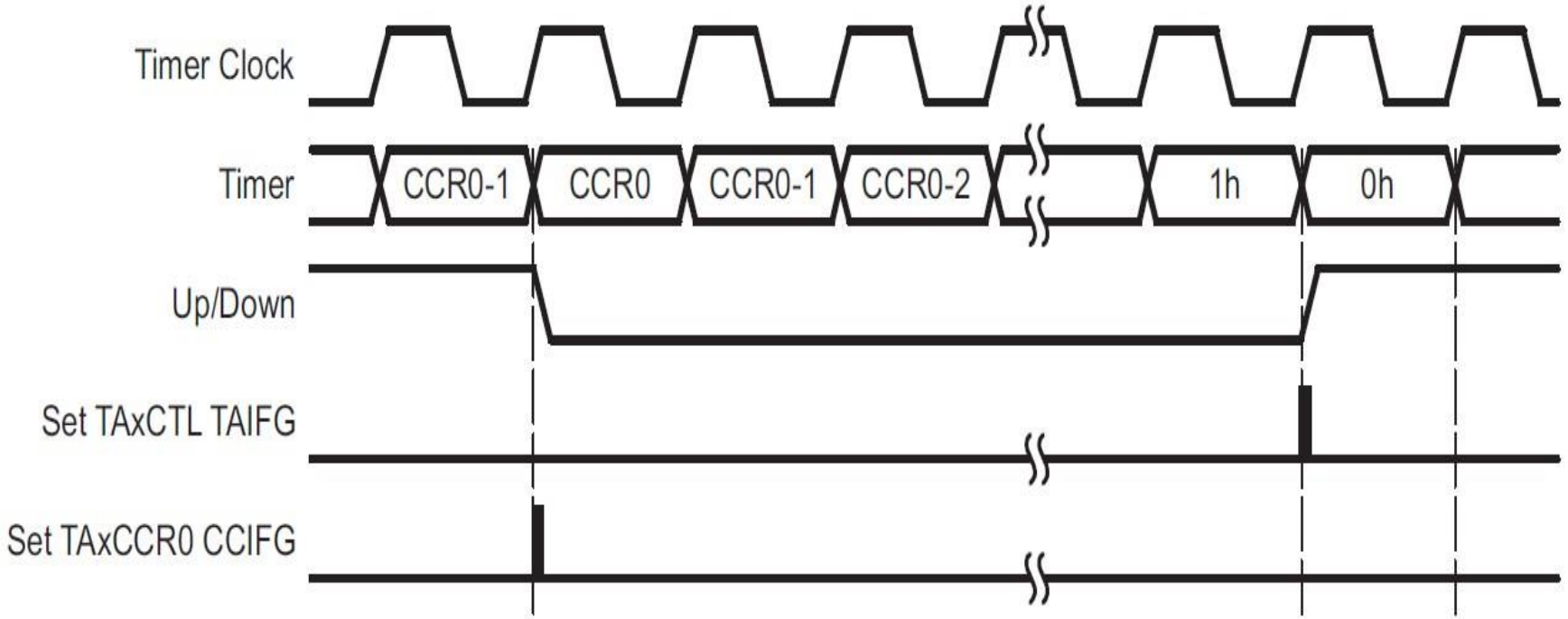
- 当定时器在增计数方向时。

  如果新的计数周期大于或者等于原来的计数周期，或者比当前的计数值大，定时器会增计数到新的计数周期，再反向计数。

  如果新的计数周期小于当前的计数值，则定时器将立即开始减计数。但是，在定时器减计数之前有一个额外的计数(即当前计数器的值加1后再进行减计数)。

定时器工作模式 ——增/减计数模式 （3/3）

- 定时器TAxCCR的值从TAxCCR0 - 1增计数到TAxCCR0时，中断标志TAxCCR0 CCIFG置位；当定时器从0001h减计数到0000h时，中断标志TAIFG置位。标志位的设置情况如图所示。



MSP430定时功能及其实现（1/2）

- 定时功能模块是MSP430应用系统中经常用到的重要部分，可用来实现定时控制、延迟、频率测量、脉宽测量和信号产生、信号检测等等。

- 一般来说，MSP430所需的定时信号可以用软件和硬件两种方法来获得。

- MSP430系列有丰富定时器资源：看门狗定时器（WDT），定时器A（Timer_A），定时器B（Timer_B）和定时器D（Timer_D）等。

MSP430定时功能及其实现（2/2）

MSP430系列定时器部件功能，如下表所示：

定时器	功能
看门狗定时器	基本定时、当程序发生错误时执行一个受控的系统重启动
基本定时器	基本定时、支持软件和各种外围模块工作在低频率、低功耗条件下
定时器A	基本定时、支持同时进行的多种时序控制、多个捕获/比较功能和多种输出波形（PWM），可以以硬件方式支持串行通信。
定时器B	基本定时、功能基本同定时器A,但比定时器A灵活，功能更强大
定时器D	基本定时、功能基本同定时器A,但比定时器A灵活，功能更强大

看门狗定时器 —— 概述（1/1）

◆ 看门狗定时器，主要作用：

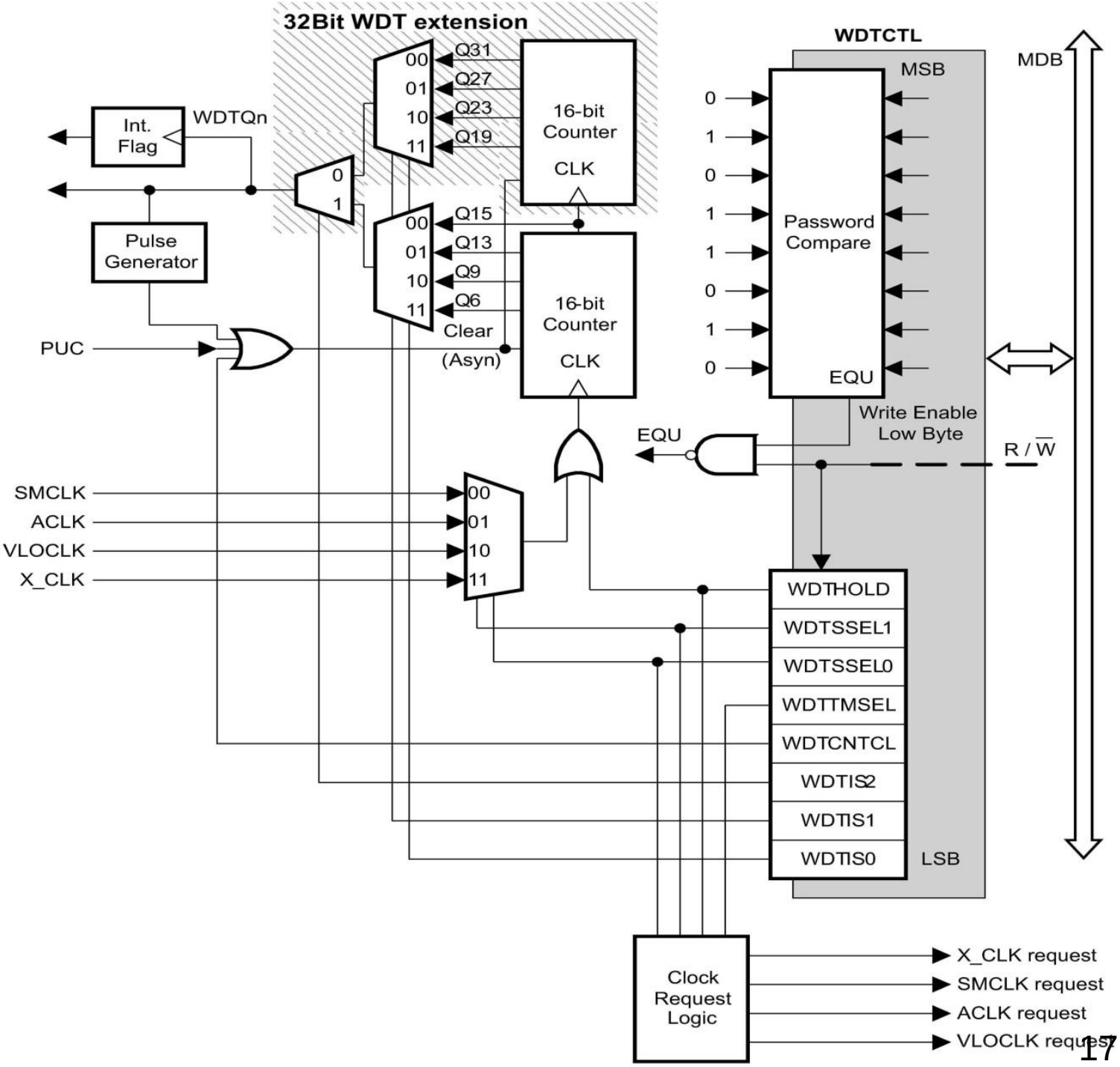
用于在“程序跑飞”时，WDT就会产生溢出，从而产生系统复位，CPU需要重新运行用户程序，这样程序就可以又回到正常运行状态。

◆ MSP430x5xx 看门狗模块具有以下特性：

- 32种软件可选的定时时间
- 看门狗工作模式
- 定时器工作模式
- 带密码保护的 WDT 控制寄存器
- 时钟源可选择
- 为降低功耗，可停止
- 时钟失效保护

看门狗

MSP430 X5XX / 6XX系列单片机的看门狗定时器原理，如右图所示



看门狗定时器 —— WDT的操作（2/3）

用户可以通过 WDTCTL 寄存器中的 WDTTMSEL 和 WDTHOLD 控制位设置 WDT 工作在看门狗模式、定时器模式和低功耗模式。

- 看门狗模式
  - PUC后，WDT 进入默认状态。如果系统不用看门狗功能，应该在程序开始处禁止看门狗功能。
  - 在看门狗模式下，如果计数器超过了定时时间，就会产生复位和激活系统上电清除信号。
  - 用户软件一般都需要进行如下操作：
    - 进行WDT的初始化：设置合适的时间。
    - 周期性地对WDTCNT清零：防止WDT溢出。

看门狗定时器 —— WDT的操作（3/3）

- 定时器模式

  WDTTMSEL 设置为 1 时，WDT 工作在定时器模式。
  在定时器模式下，定时间隔到以后，WDTIFG 标志位置 1

- 低功耗模式

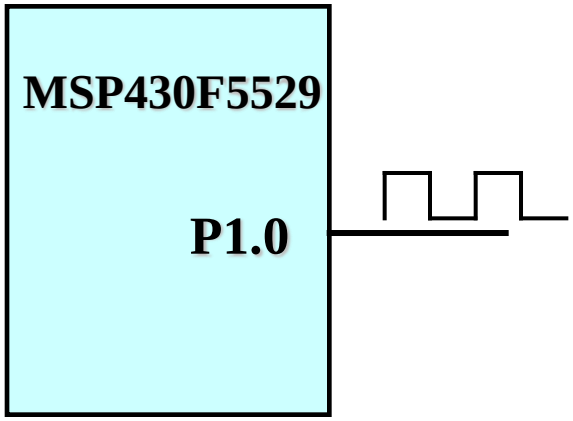
  当不需要看门狗定时器时，可使用 WDTHOLD 位来停止看门狗计数器 WDTCNT，以降低功耗。

看门狗定时器 —— WDT应用举例 （1/2）

例，使用看门狗定时功能产生一个方波（周期性的取反P1.0）

程序代码如下：

```
# include  <msp430.h>
void main(void)
{
  WDTCTL = WDT_MDLY_32;  // 定时周期为32ms
  SFRIE1 |= WDTIE;       // 使能WDT中断
  P1DIR |= BIT0;         // P1.0输出
  __enable_interrupt();  // 系统中断允许
  for (;;)
  {
        // 进入 LPM0
    __bis_SR_register(LPM0_bits);
    __no_operation();
  }
}
```



看门狗定时器 —— WDT应用举例 (2/2)

```
    // 看门狗中断服务子程序
#pragma vector= WDT_VECTOR
__interrupt void watchdog_timer (void)
{
        P1OUT ^= 0x01;          // P1.0取反
}
```

定时器 A —— 主要内容

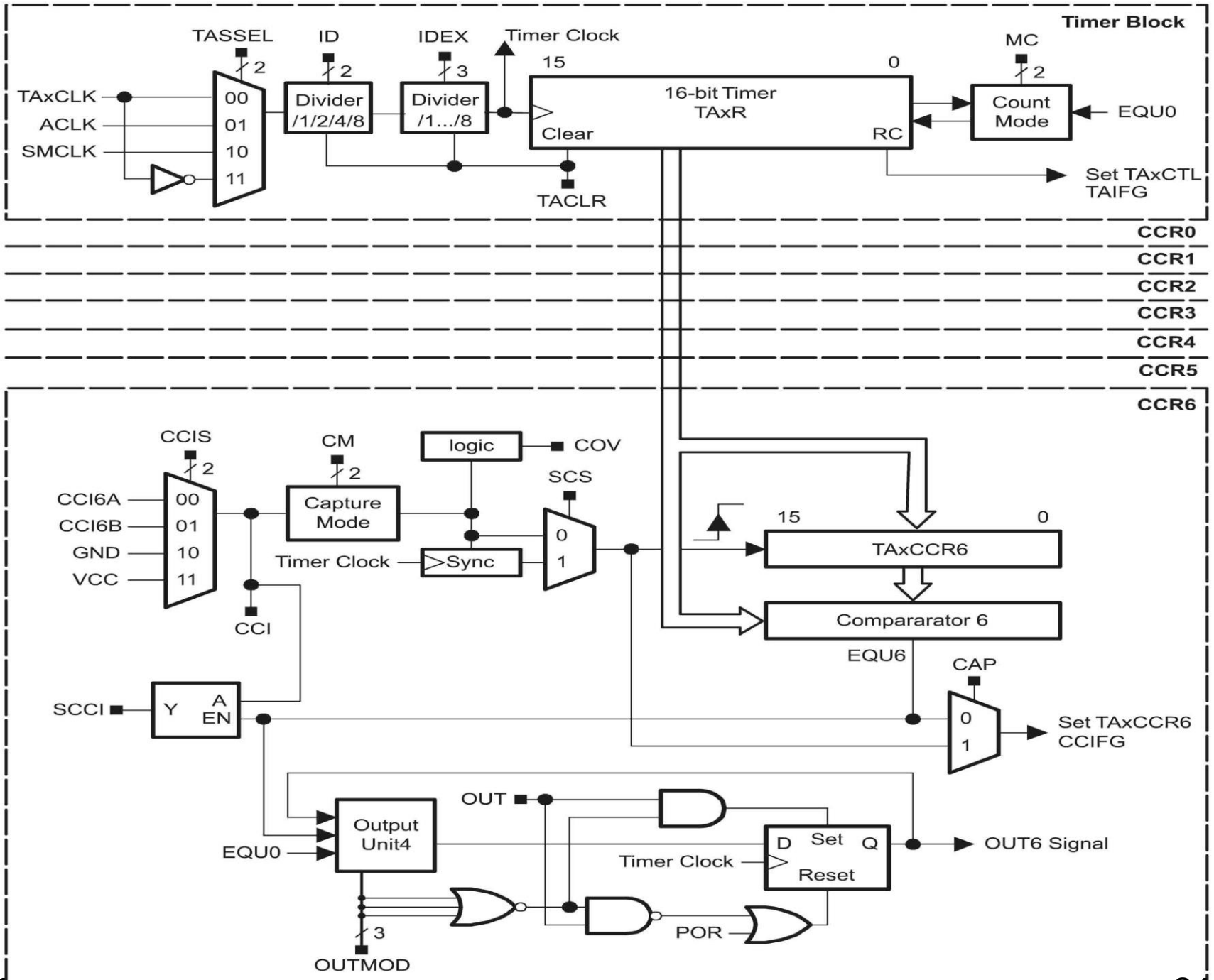
- 定时器A的特性
- 定时器A的结构
- 定时器A的工作原理
  - 定时器工作模式
  - 捕获/比较模块
  - 输出单元
  - Timer_A中断
- 定时器A的典型应用

定时器 A —— 特性（1/1）

- 定时器 Ax 由一个16位定时器和多路捕获/比较通道组成。

- MSP430X5XX / 6XX系列单片机的Timer _A有以下特性：
  - 带有 4 种操作模式的异步 16 位定时/计数器;
  - 输入时钟可以有多种选择，可以是慢时钟，快时钟以及外部时钟;
  - 可配置捕获/比较寄存器数多达 7 个;
  - 可配置的PWM（脉宽调制）输出;
  - 异步输入和同步锁存。不仅能捕获外部事件发生的时间还可锁定其发生时的高低电平;
  - 完善的中断服务功能。快速响应Timer_A中断的中断向量寄存器;
  - 8种输出方式选择;
  - 可实现串行通讯。

结构


Timer Block
TASSEL
ID
IDEX
Timer Clock
MC
TAxCLK
ACLK
SMCLK
00
01
10
11
Divider /1/2/4/8
Divider /1.../8
15
0
16-bit Timer TAxR
Clear
RC
Count Mode
EQU0
TACLR
Set TAxCTL TAIFG
CCR0
CCR1
CCR2
CCR3
CCR4
CCR5
CCR6
CCIS
CM
logic
COV
SCS
CCI6A
CCI6B
GND
VCC
Capture Mode
Timer Clock
Sync
TAxCCR6
Compararator 6
EQU6
CAP
CCI
SCCI
Y
A
EN
Set TAxCCR6 CCIFG
OUT
Output Unit4
EQU0
D
Set
Q
Reset
Timer Clock
OUT6 Signal
POR
OUTMOD

定时器 A —— 结构（2/2）

- 从上图可以看出，Timer_A有以下部分组成：
  - 定时计数器：16 位定时/计数寄存器——TAxR
  - 时钟源的选择和分频：定时器时钟 TACLK 可以选择 ACLK，SMCLK 或者来自外部的 TAxCLK。选择的时钟源，可以通过软件选择分频系数（2、3、4、5、6、7、8）。
  - 捕获/比较器：用于捕获事件发生的时间或产生时间间隔，捕获比较功能的引入主要是为了提高I/O 端口处理事务的能力和速度。
  - 输出单元：具有可选的8种输出模式，用于产生用户需要的输出信号，支持PWM。

定时器 A —— 工作原理（1/1）

- 定时器工作模式
  - 停止模式
  - 增计数模式
  - 连续计数模式
  - 增/减计数模式
- 捕获/比较模块
- 输出单元
- Timer_A中断

定时器 A —— 捕获/比较模块 （1/5）

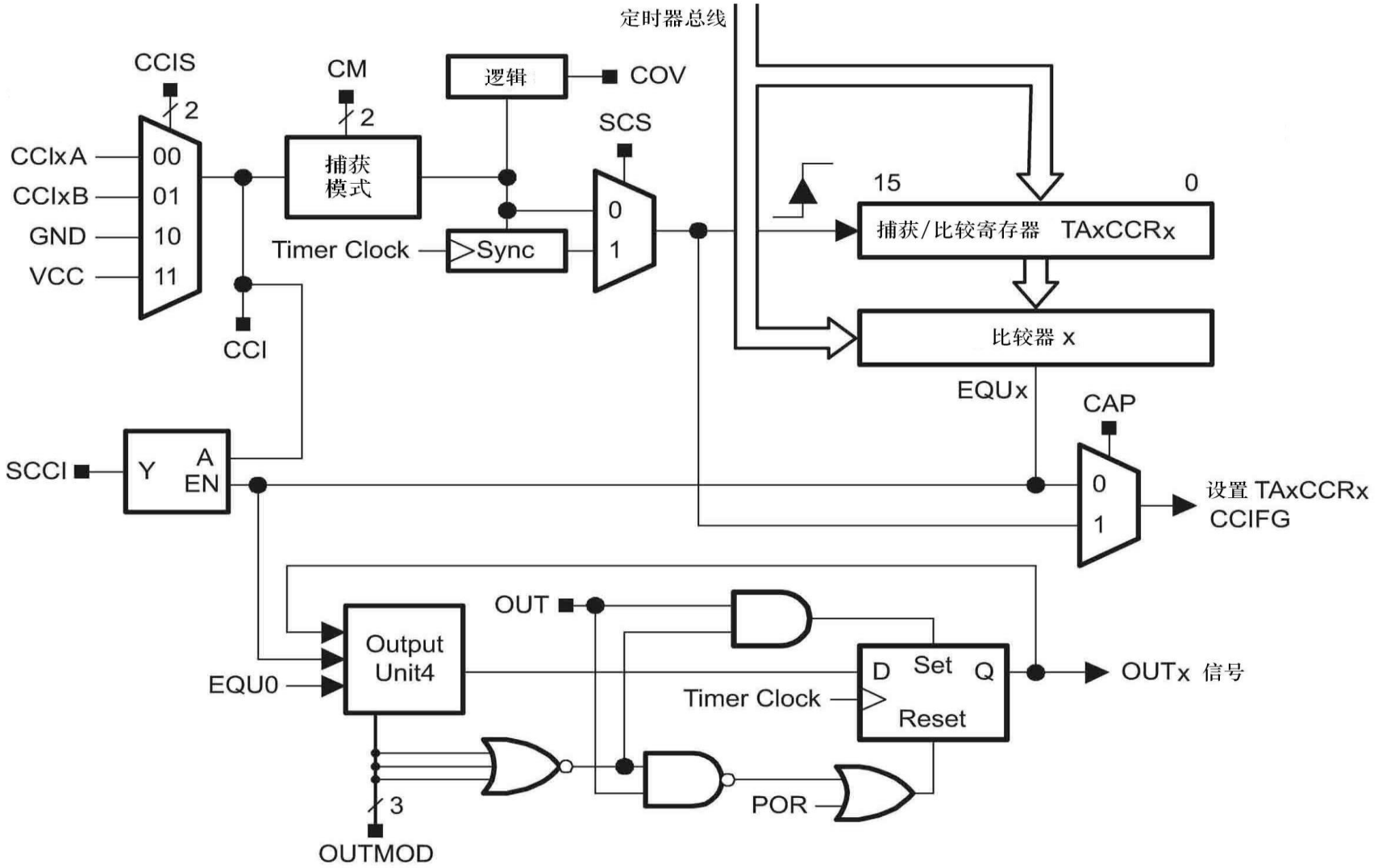
- Timer_A有多个相同的捕获/比较模块，为实时处理提供灵活的手段，每个模块都可用于捕获事件发生的时间或产生定时间隔。

- 通过TACCTLx中的CAP位选择模式，该模块既可用于捕获模式，也可用于比较模式。

- 当发生捕获事件或定时时间到都将引起中断。

- 捕获/比较模块的结构，如下图所示。

定时器 A —— 捕获/比较模块 （2/5）

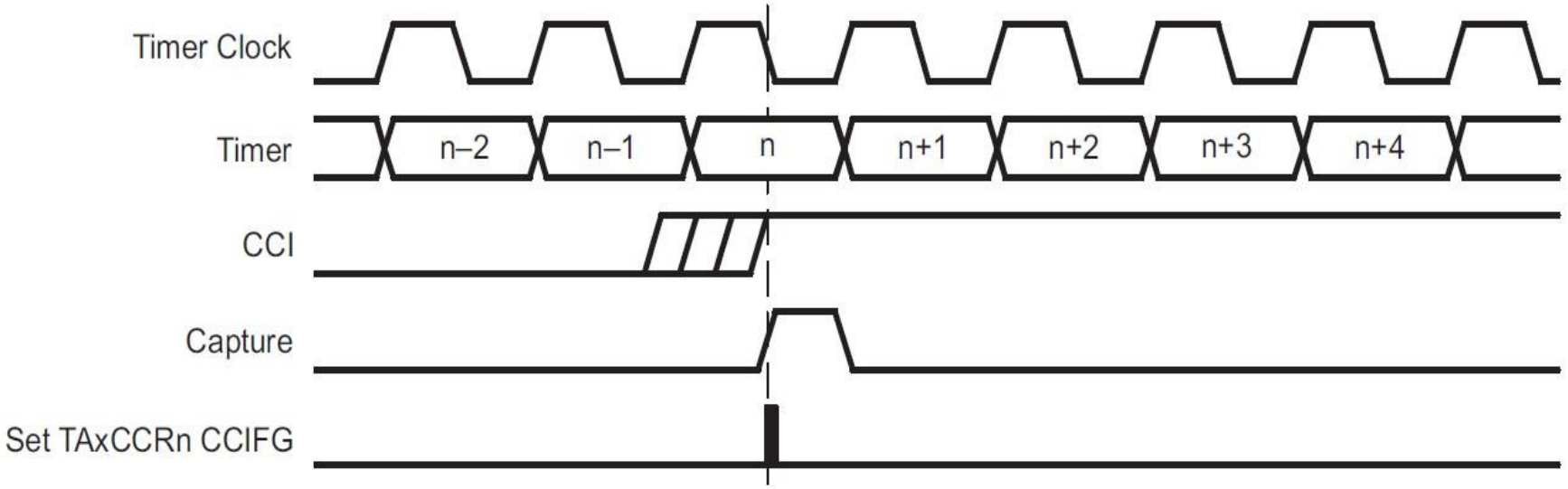


捕获/比较模块的逻辑结构

定时器 A —— 捕获/比较模块 （3/5）

- 捕获模式
  - 当TACCTLx中的CAP = 1，该模块工作在捕获模式。
  - 每个捕获/比较寄存器可以用来记录时间事件，例如：
    - ▲ 测量软件程序所用时间
    - ▲ 测量硬件事件之间的时间
    - ▲ 测量系统频率
  - 用CM1和CM0 位选择捕获条件，可以选择禁止捕获、上升沿捕获、下降沿捕获或者上升沿下降沿都捕获。
  - 当捕获完成后，定时器的值被复制到 TAxCCRn 寄存器，并且中断标志 CCIFG 置位。如果总的中断允许位GIE允许，相应的中断允许位CCIE也允许，则将产生中断请求。如下图所示：

定时器 A —— 捕获/比较模块 （4/5）



捕获模式的信号

定时器 A —— 捕获/比较模块 （5/5）

◆ 比较模式

➢当TACCTLx中的CAP = 0，该模块工作在比较模式。

➢比较方式主要用于为软件或应用硬件产生定时，还可为D/A转换功能或者马达控制等各种用途产生脉宽调制（PWM）输出信号。

➢在计数器TAxR计数到TAxCCRn（n 代表具体的捕获比较寄存器）的值时：

▲ 中断标志 CCIFG 置位
▲ 内部信号 EQUx=1
▲ EQUx 根据输出模式影响输出
▲ 输入信号 CCI 被锁存在 SCCI

定时器 A —— 输出单元（1/5）

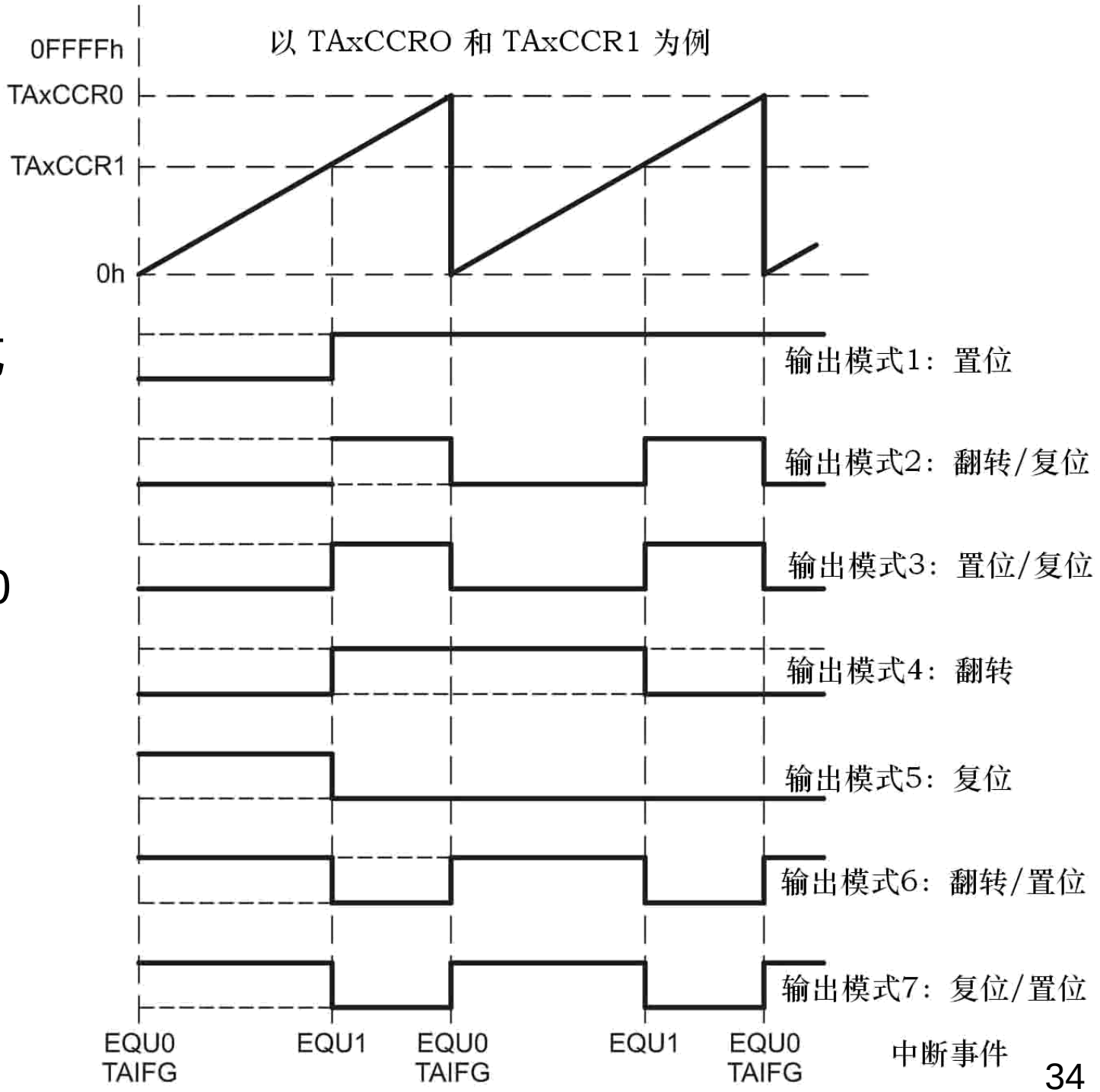
- 每个捕获/比较模块都包含一个输出单元，用于产生输出信号。
- 每个输出单元有8种工作模式，可产生基于EQUx的多种信号。
- 除模式0外，其他模式的输出都在定时器时钟上升沿时发生变化。
- 输出模式2，3，6，7不适合输出单元0，因为EQUx=EQU0。
- 输出单元在输出控制位OUTMODx的控制下，有8种输出模式输出信号。这些模式与TAxR、TACCTLx、TAxCCR0的值有关，如下表所示。

定时器 A —— 输出单元（2/5

OUTMODx	模式	说明
000	输出模式0：输出	输出信号取决与寄存器 TACCTLx 中的 OUT位。当 OUT位更新时，输出信号立即更新。
001	输出模式1：置位	输出信号在TAxR等于TAxCCRn时置位，并保持置位到定时器复位或选择另一种输出模式为止。
010	输出模式2：翻转/复位	输出在TAxR的值等于TAxCCRn时翻转，当TAxR的值等于TAxCCR0时复位。
011	输出模式3：置位/复位	输出在TAxR的值等于TAxCCRn时置位，当TAxR的值等于TAxCCR0时复位。
100	输出模式4：翻转	输出电平在TAxR的值等于TAxCCRn时翻转，输出周期是定时器周期的2倍。
101	输出模式5：复位	输出在TAxR的值等于TAxCCRn时复位，并保持低电平直到选择另一种输出模式。
110	输出模式6：翻转/置位	输出电平在TAxR的值等于TAxCCRn时翻转，当TAxR值等于TAxCCR0时置位。
111	输出模式7：复位/置位	输出电平在TAxR的值等于TAxCCRn时复位，当TAxR的值等于TAxCCR0时置位。

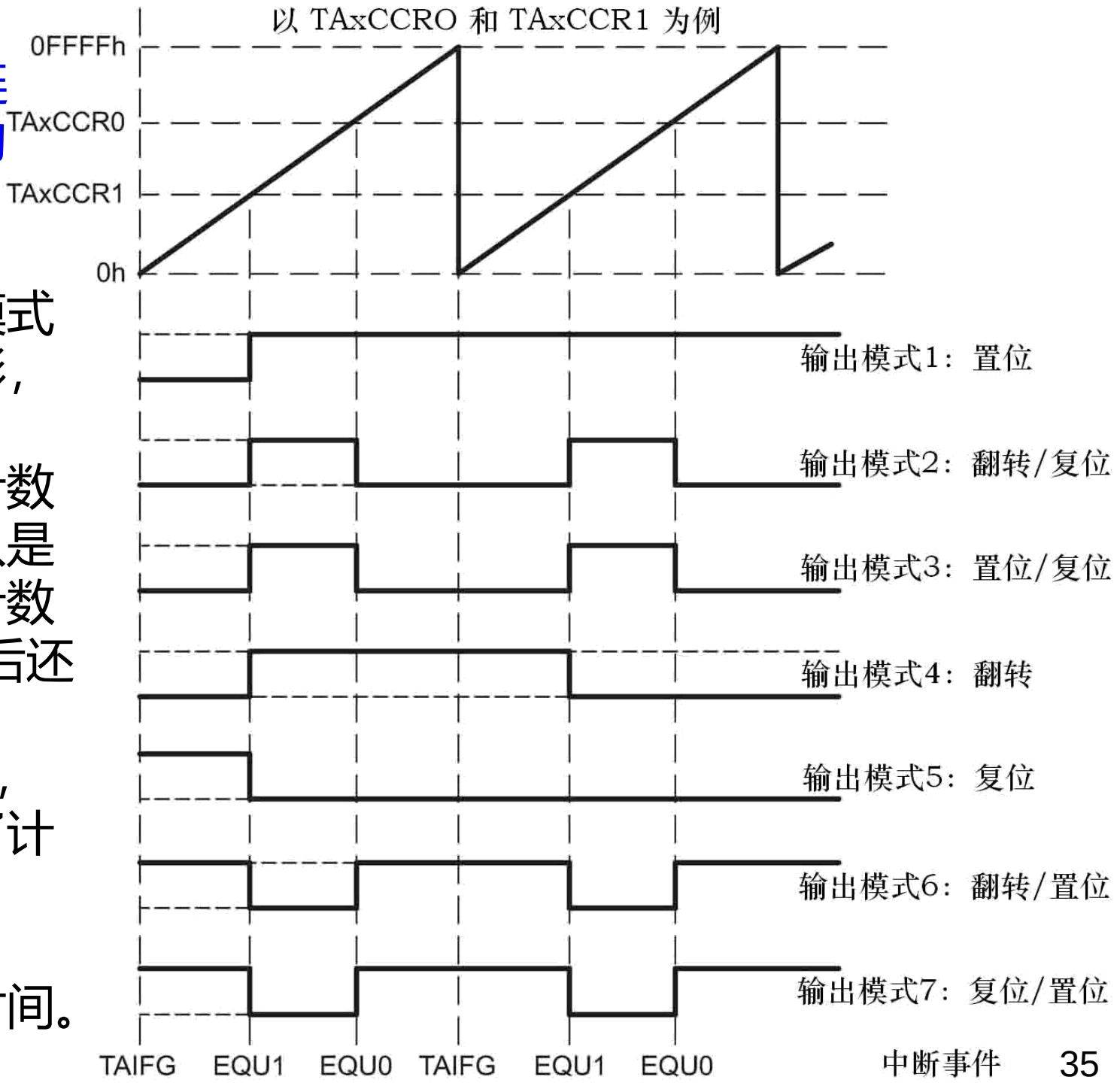
◆ 定时器在增计数模式的输出实例

在增计数模式下，当计数器TAxR增加到TAxCCRx或从TAxCCR0计数到0时，OUTn信号按选择的输出模式发生变化。实例如右图所示。



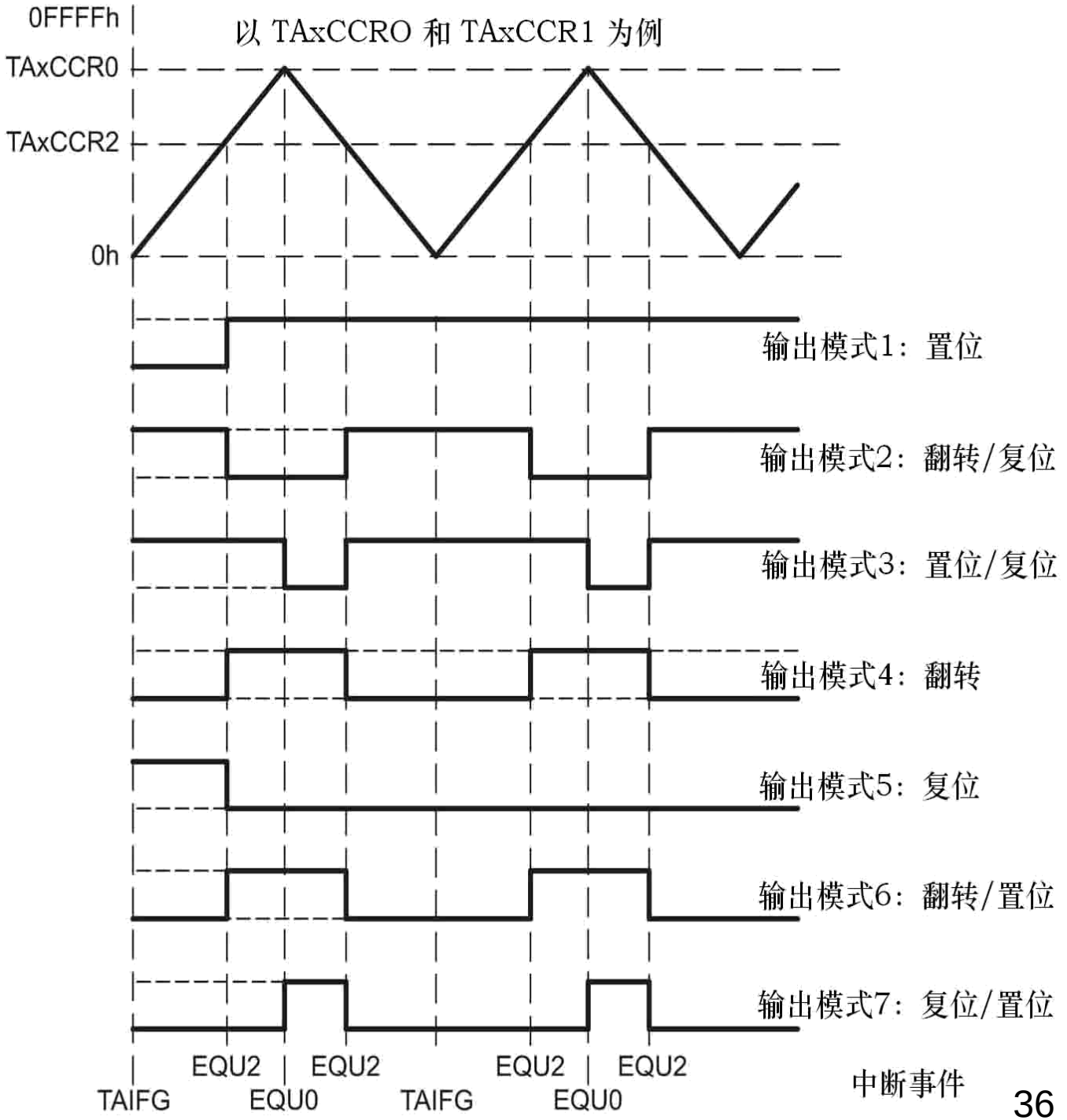
◆ 定时器在连续计数模式的输出实例

在连续计数模式下的输出波形，如右图所示。其波形与增计数模式一样，只是计数器在增计数到TAxCCR0后还要继续增计数到0FFFFH，这样就延长了计数器计数到TAxCCR1的数值后的时间。



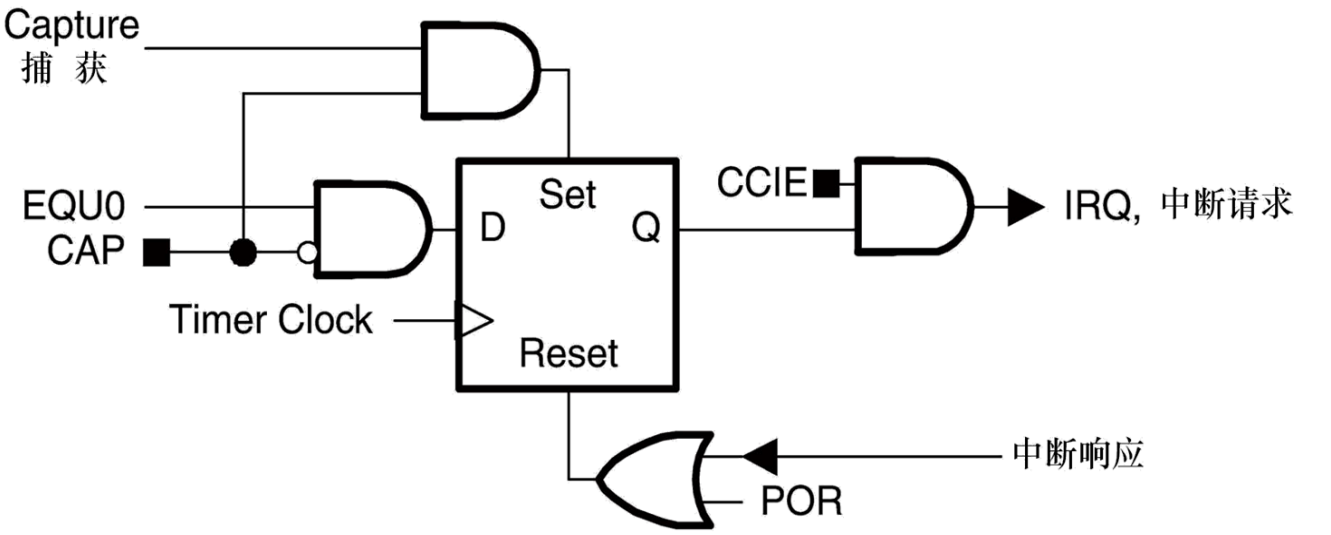
◆ 定时器在增/减计数模式的输出实例

在增/减计数模式下的输出实例，如右图所示。这时的各种输出波形与定时器增计数模式或连续计数模式不同。当定时器在任意计数方向上等于TAxCCRx时，OUTn信号都按选择的输出模式发生改变。



定时器 A —— 中断（1/2）

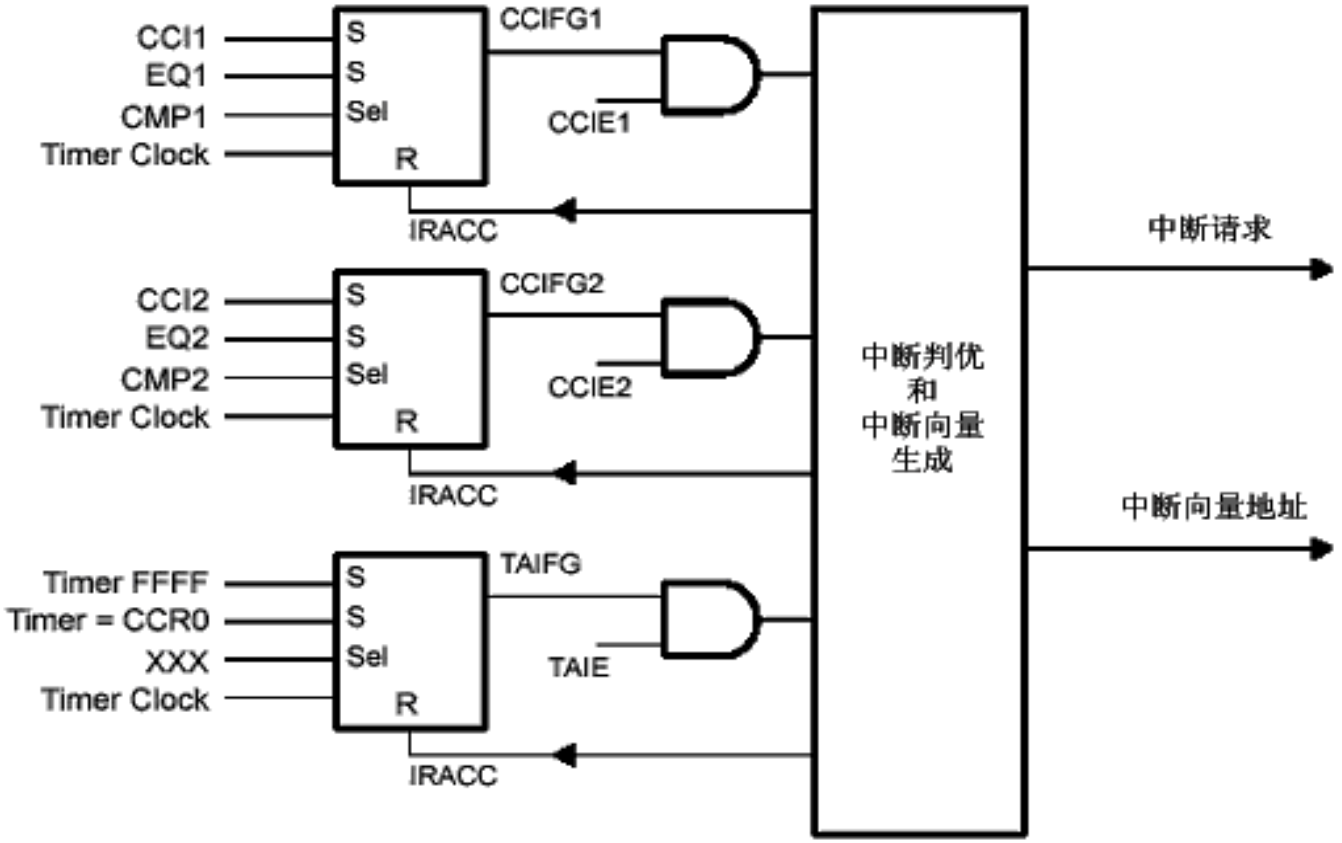
- Timer_A中断可由计数器溢出引起，也可以来自捕获/比较寄存器。每个捕获/比较模块可独立编程，由捕获/比较外部信号以产生中断。
- Timer_A模块使用两个中断向量：
  - 一个单独分配给捕获/比较寄存器TAxCCR0；
  - 另一个作为共用中断向量用于定时器和其他的捕获/比较寄存器（TAIV）。
- TAxCCR0中断如下图所示：



定时器 A —— 中断（2/2）

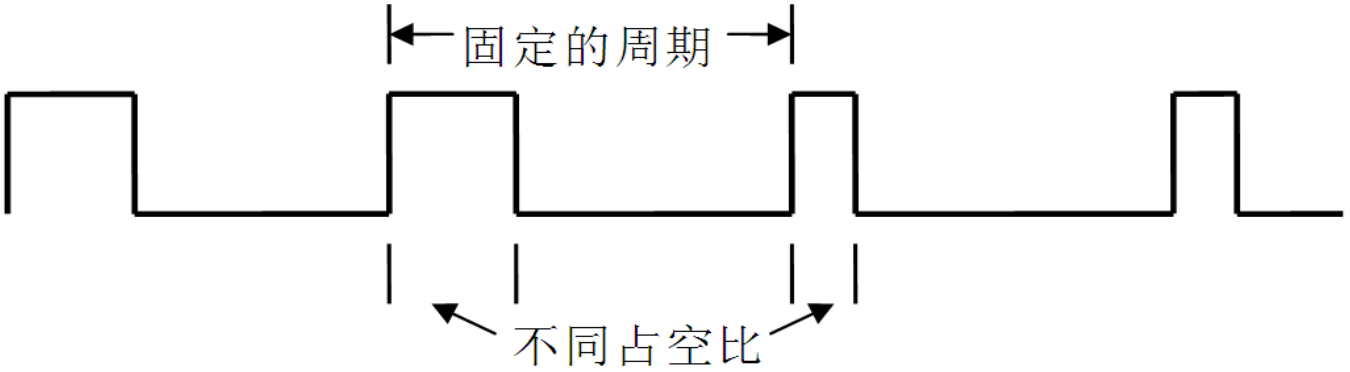
◆ TAxCCR1 ~ TAxCCRx和定时器按照优先次序结合共用一个中断向量，属于多源中断。中断向量寄存器（TAIV）用于确定哪个标志请求中断。

◆ TAxCCR1 ~ TAxCCRx中断，如下图所示：

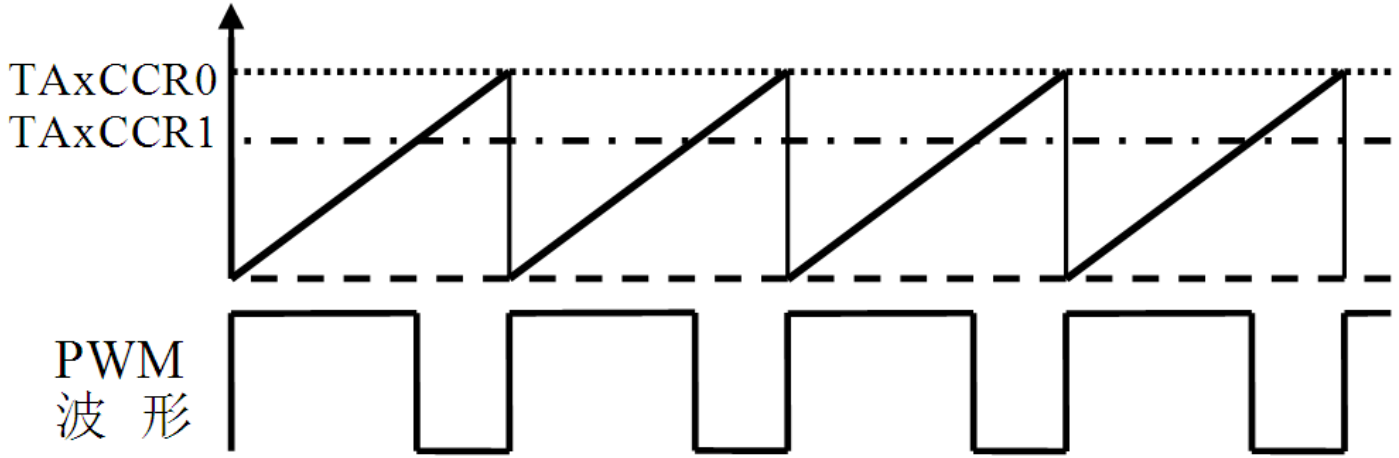


定时器 A典型应用——实现PWM （1/5）

- PWM信号是一种具有固定周期不定占空比的数字信号，如下图所示：



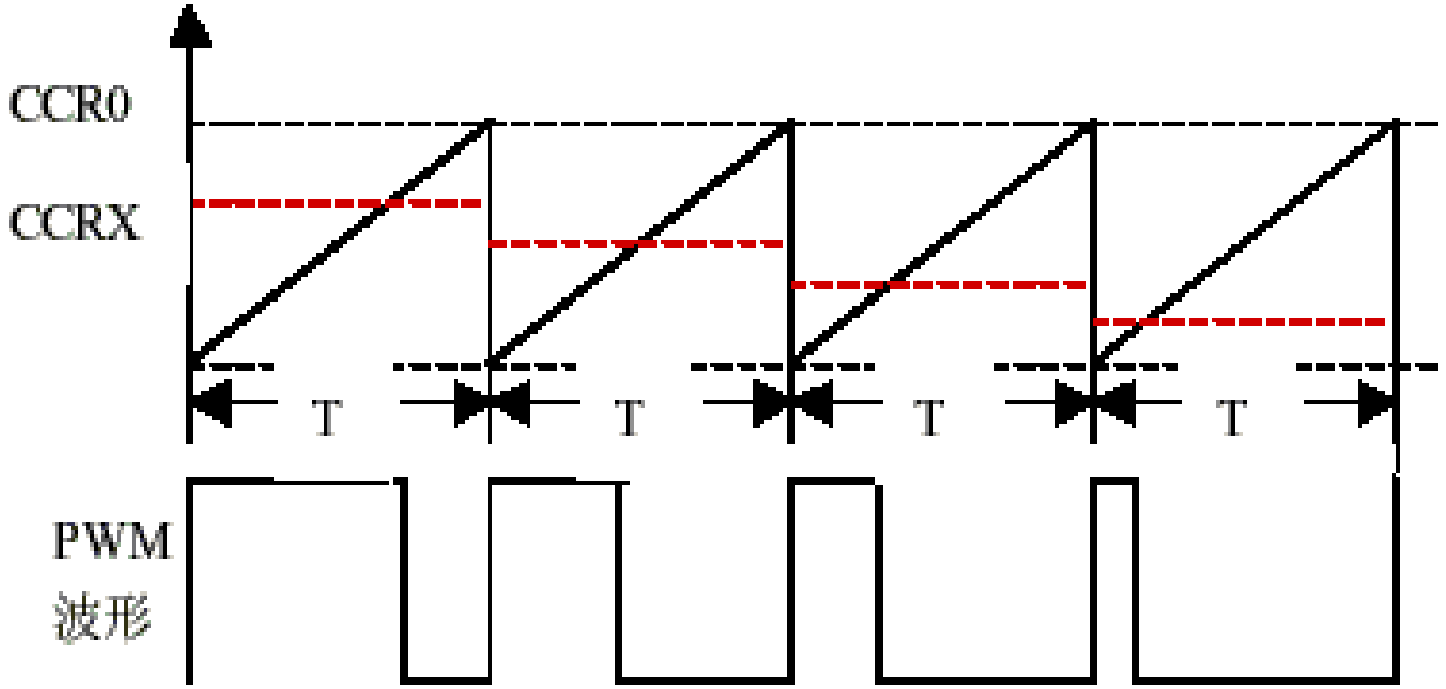
- 如果Timer_A定时器的计数器工作在增计数方式，输出采用输出模式7（复位/置位模式），则可利用寄存器TAxCCR0控制PWM波形的周期，用某个寄存器TAxCCRx控制占空比。这样Timer_A就可以产生出任意占空比的PWM波形。如下图所示：



定时器 A典型应用——实现PWM （2/5）

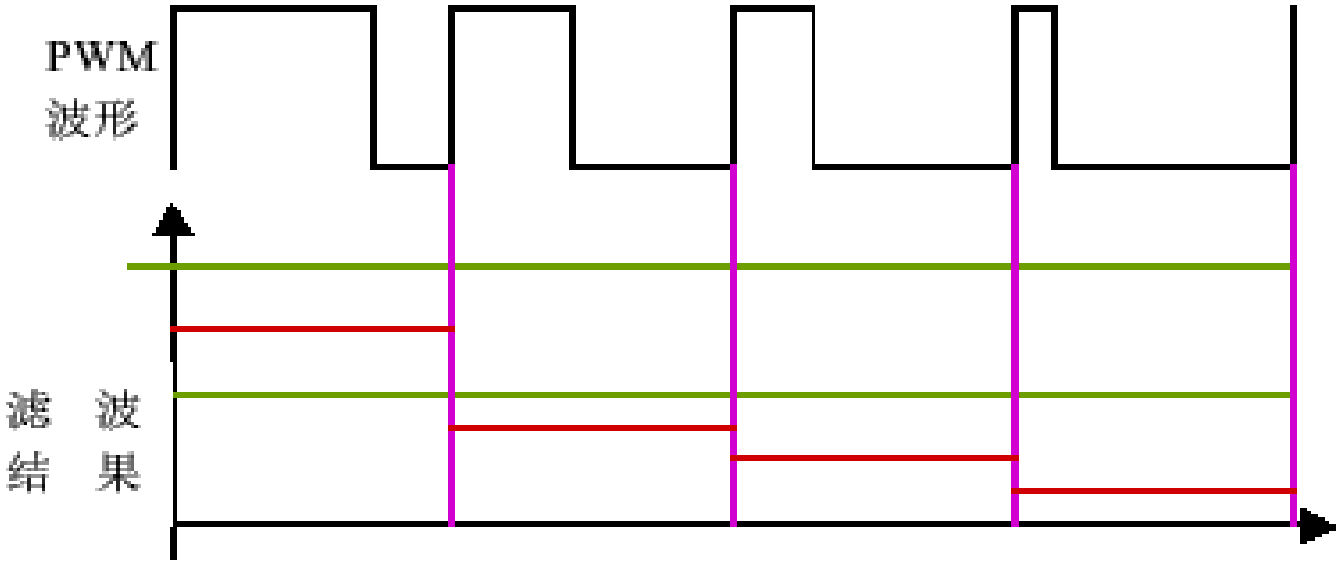
- 可以随时间变化任意改变PWM信号的占空比，具体做法：
  - 保持CCR0值（周期不变）；
  - 改变CCRx值（改变占空比）。

如下图所示：



定时器 A典型应用——实现PWM （3/5）

◆ 如果PWM信号占空比随时间变化，那么经过滤波之后的输出信号就是幅度变化的模拟信号，因此通过控制PWM信号的占空比，就可以产生不同的模拟信号，实现D/A转换。如下图所示：

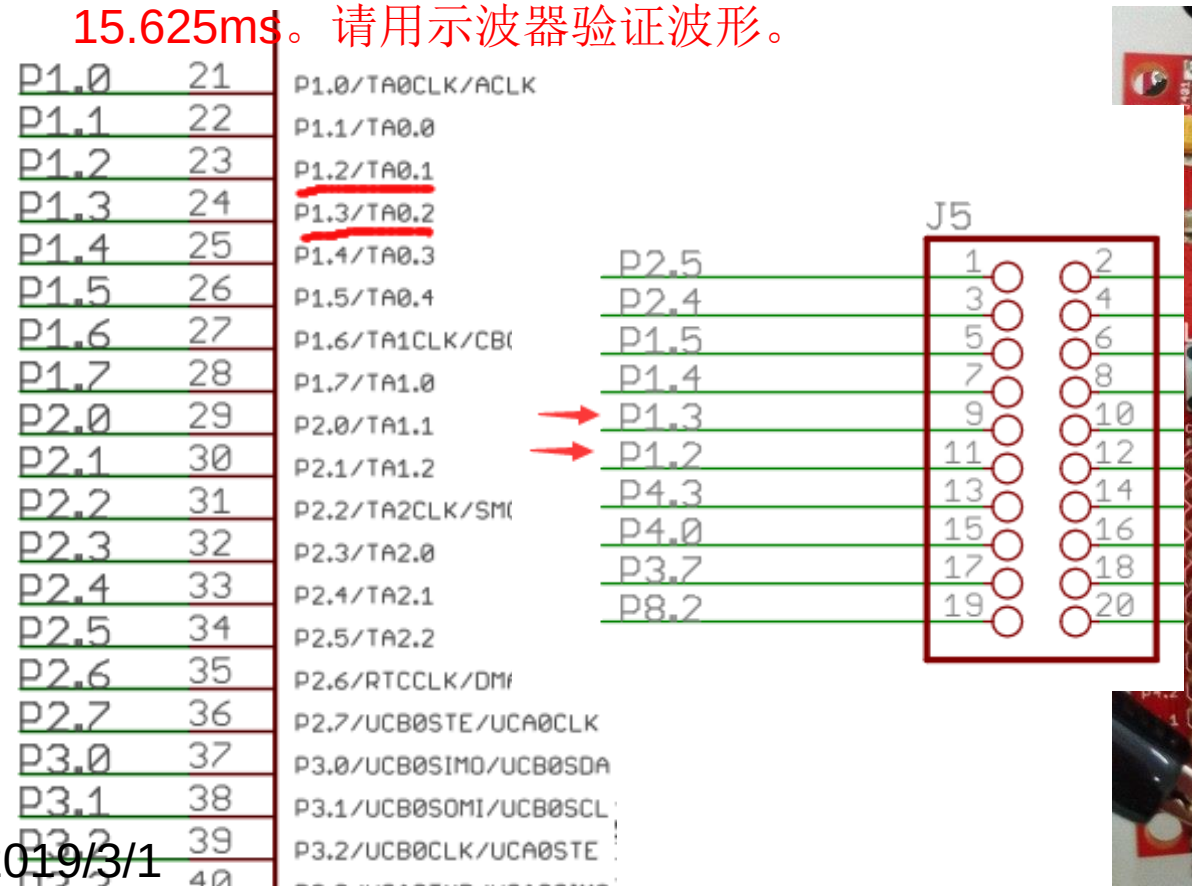


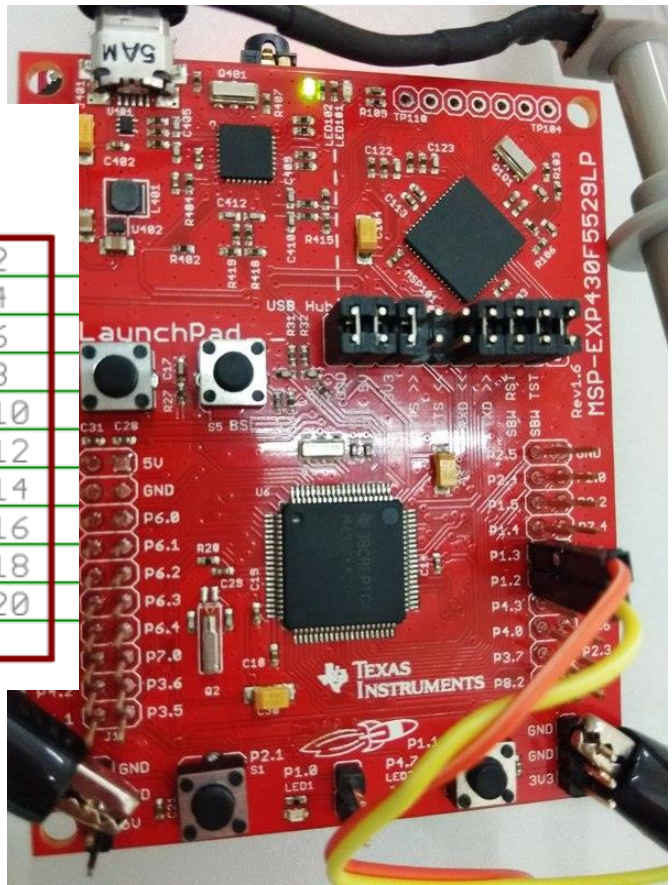
◆ PWM不需要修改占空比和时间时，CPU在做完Timer_A初始化工作之后，Timer_A就能自动输出PWM，而不需利用中断维持PWM输出，此时CPU就可以进入低功耗状态。

定时器 A典型应用——PWM 例程（4/5）

- 例：设ACLK = TACLK = LFXT1 = 32768Hz, MCLK = SMCLK = DCOCLK = 32×ACLK = 1.048576MHz，利用Timer_A输出周期为512 /32768 = 15.625ms、占空比分别为75%和25%的PWM矩形波。

由程序知：P1.2—> CCR1 - 75% PWM，P1.3—> CCR2 - 25% PWM。周期15.625ms。请用示波器验证波形。





定时器 A典型应用——PWM 例程（5/5）

```
int main(void)
{
// stop watchdog timer
WDTCTL = WDTPW | WDTHOLD;

TA0CTL=____;      // ACLK, 清除 TAR
TA0CCR0 =__;      // PWM周期
TA0CCTL1 = ___; // 输出模式7
TA0CCR1 = ___;  //占空比384/512=0.75
TA0CCTL2 = ___; // 输出模式7;
TA0CCR2 = ___;  //占空比128/512=0.25
_____________;    // P1.2 方向为输出
_____________;    // P1.2端口为外设
                  // 定时器TA0.1
_____________;    // P1.3 方向为输出
_____________;    // P1.3端口为外设
                  // 定时器TA0.2
TA0CTL |= MC0;  // Timer_A 增计数模式
// 接右面
```

```
// 接左面
    for (;;)
    {
        // 进入    LPM3
        _BIS_SR(LPM3_bits );
        _NOP();
    }

     return 0;
}
```

定时器 A典型应用——PWM 例程（4/5）

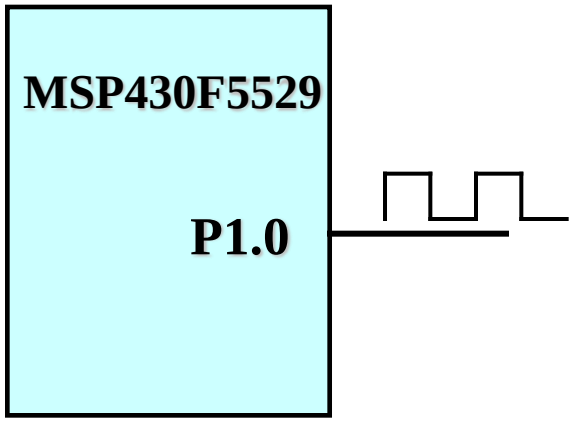
看门狗定时器实验任务

- 课上实验1：

  实验PPT例程，使用看门狗定时功能产生一个周期为64ms方波（周期性的取反P1.0）

  要求（1）用示波器查看其波形；

  （2）测量其频率 。



- 课上实验2：

  指导书，完成代码，包括IO初始化，中断服务程序语句等。

  要求（1）观察LED1灯状态，按键S1在1s前后按下的实验现象；

看门狗定时器实验作业

- 作业1：

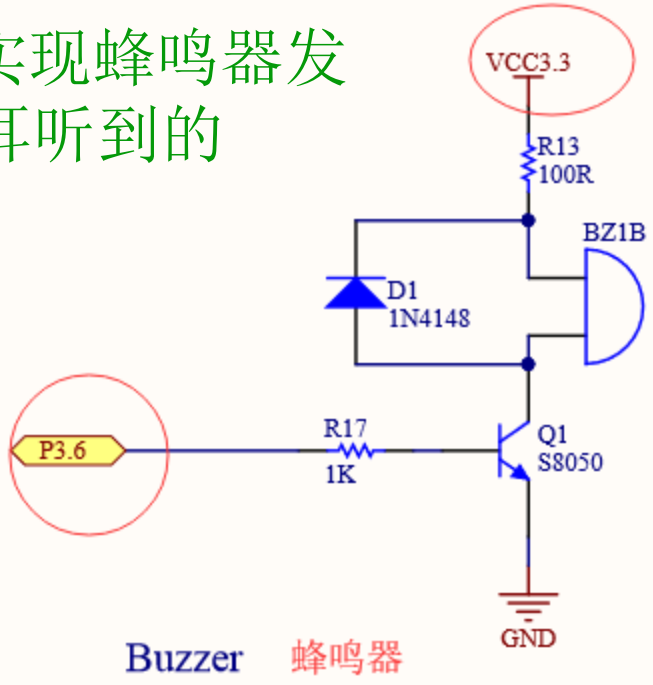
  指导书，完成课上实验2后，添加一个功能。

  要求

（1）蜂鸣器安装在P3.6引脚后，编程实现蜂鸣器发声，发声频率为1Hz，周期为1s。人耳听到的声音是类似钟表的滴答声。

（2）按键S1按下后，观察实验现象，蜂鸣器发声情况。

（3）编程实现改变发声频率为4Hz，周期为250ms。



提示：

P8.1引脚在定时中断后翻转，LED1灯闪烁，那么P3.6引脚跟P8.1引脚功能一样，只不过驱动的是蜂鸣器。

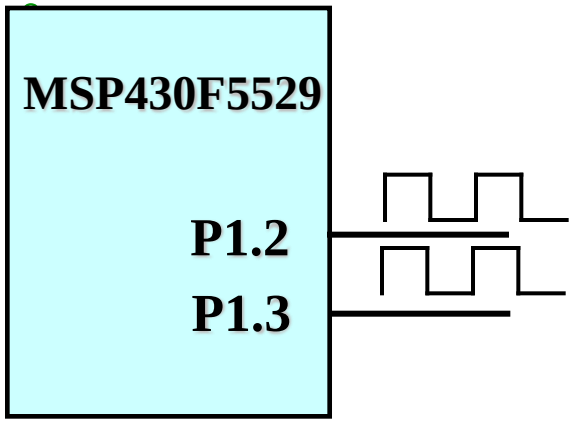
定时器A实验任务

- 课上实验1：

  定时器 A典型应用， 完成PWM 例程。

  要求（1）用示波器查看P1.2、P1.3输出波形；

  （2）测量波形周期和频率。



- 课上实验2：

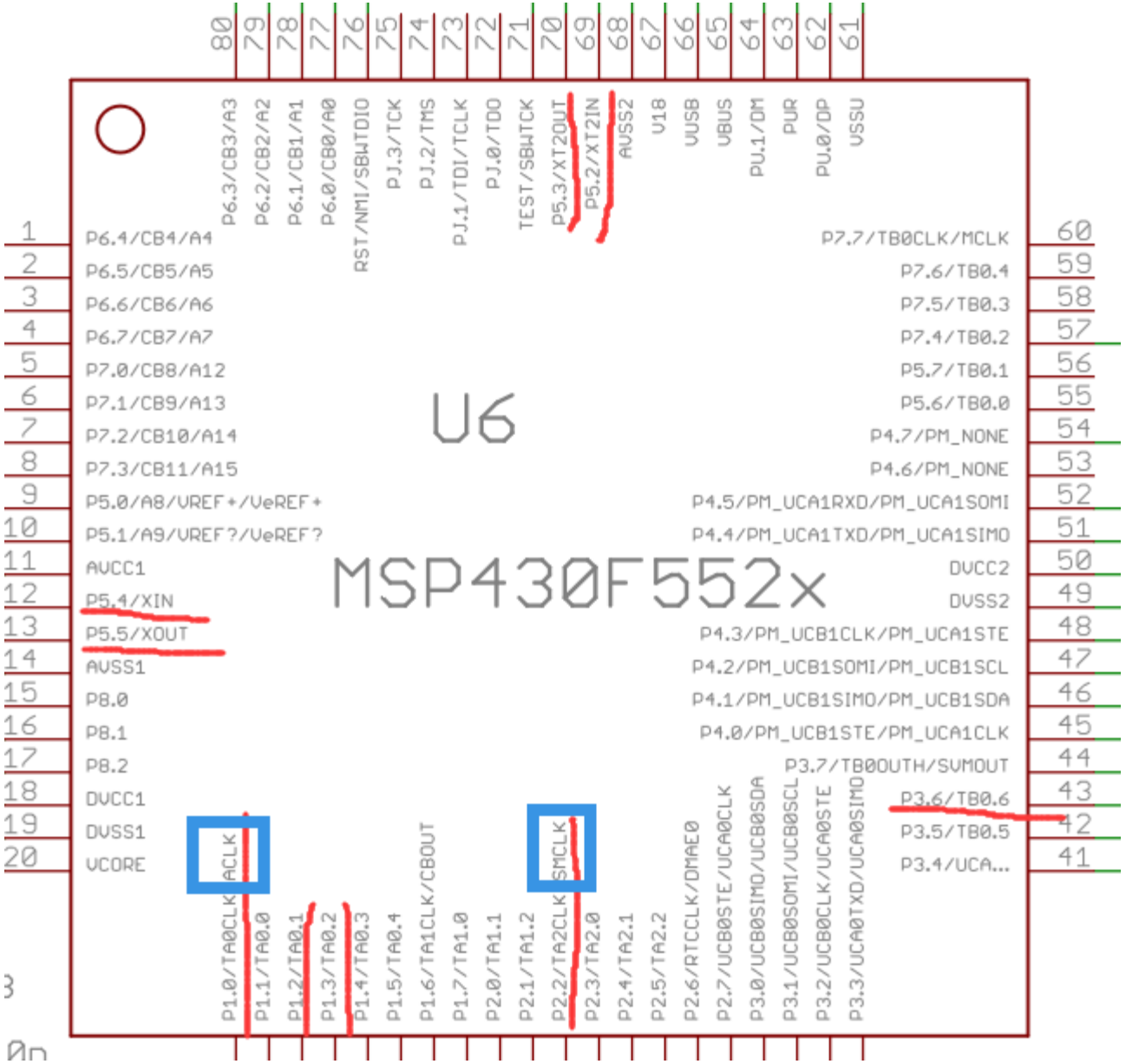
  指导书，完成代码，包括IO初始化，中断服务程序语句等。

  要求：观察LED1灯状态，按键S1和S2按下的实验现象，蜂鸣器的鸣叫效果；

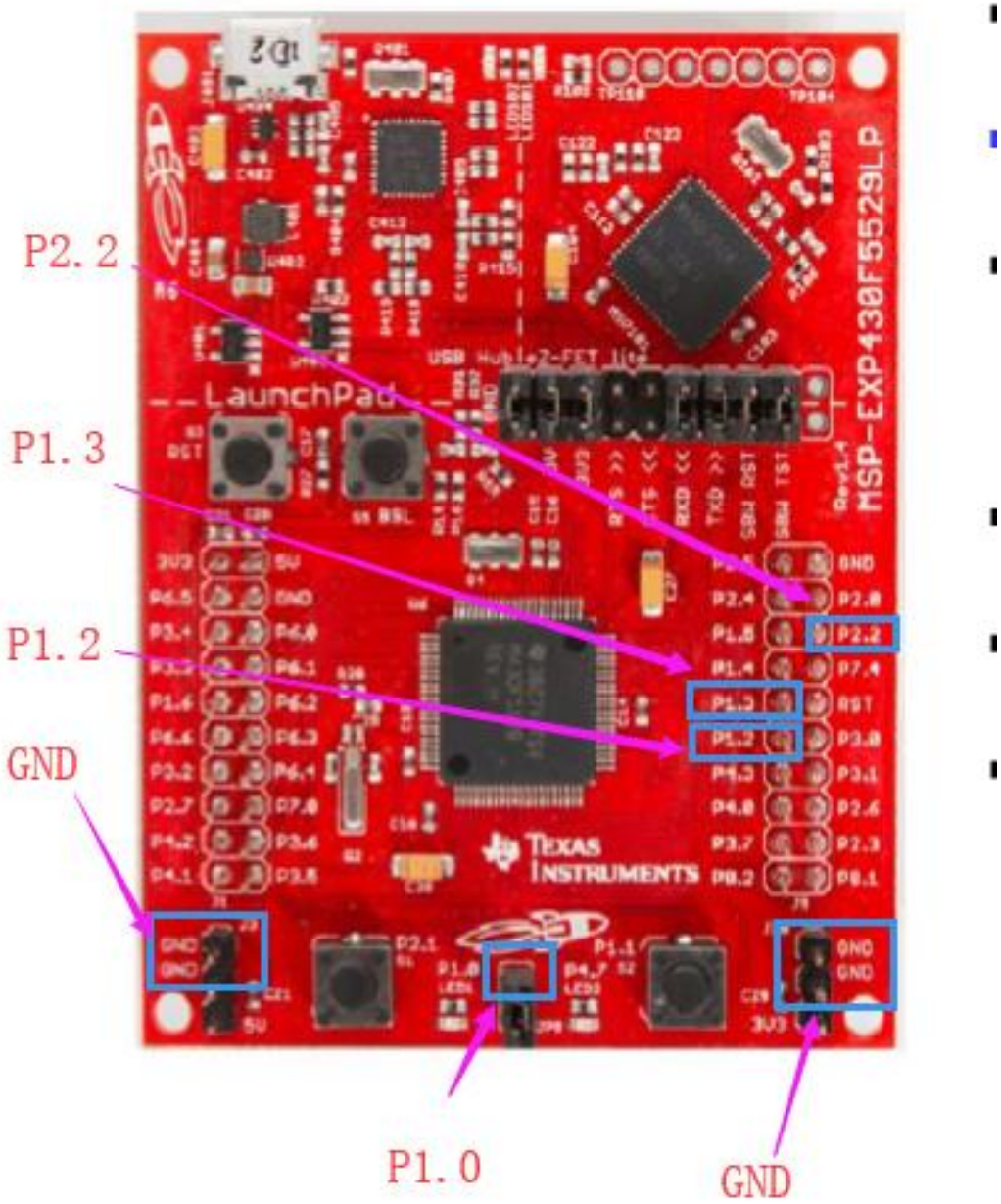
定时器A实验作业

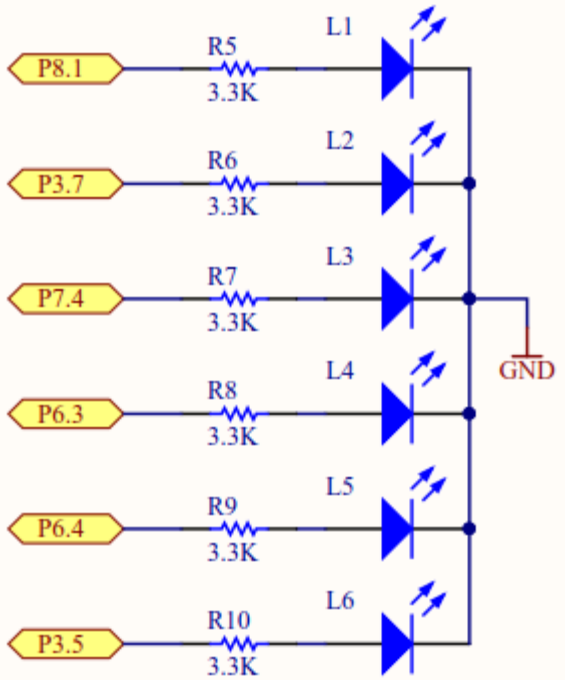
- 作业1：

  参考课上实验2，通过定时器设定改变蜂鸣器发声的音调（频率），高声调一个，低声调一个，做对比，分析蜂鸣器发生频率和定时器中断频率之间的关系。

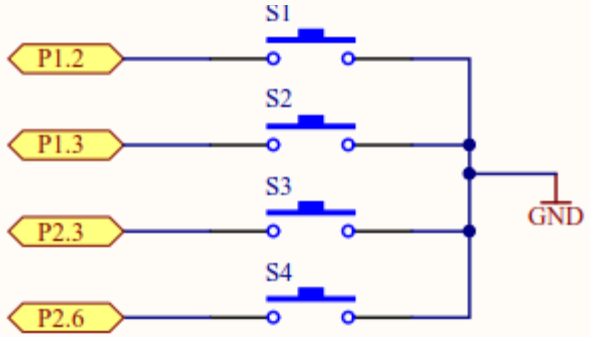

U6
MSP430F552x
P6.4/CB4/A4
P6.5/CB5/A5
P6.6/CB6/A6
P6.7/CB7/A7
P7.0/CB8/A12
P7.1/CB9/A13
P7.2/CB10/A14
P7.3/CB11/A15
P5.0/A8/VREF+/VeREF+
P5.1/A9/VREF?/VeREF?
AVCC1
P5.4/XIN
P5.5/XOUT
AVSS1
P8.0
P8.1
P8.2
DVCC1
DVSS1
VCORE
P7.7/TB0CLK/MCLK
P7.6/TB0.4
P7.5/TB0.3
P7.4/TB0.2
P5.7/TB0.1
P5.6/TB0.0
P4.7/PM_NONE
P4.6/PM_NONE
P4.5/PM_UCA1RXD/PM_UCA1SOMI
P4.4/PM_UCA1TXD/PM_UCA1SIMO
DVCC2
DVSS2
P4.3/PM_UCB1CLK/PM_UCA1STE
P4.2/PM_UCB1SOMI/PM_UCB1SCL
P4.1/PM_UCB1SIMO/PM_UCB1SDA
P4.0/PM_UCB1STE/PM_UCA1CLK
P3.7/TB0OUTH/SVMOUT
P3.6/TB0.6
P3.5/TB0.5
P3.4/UCA...
P6.3/CB3/A3
P6.2/CB2/A2
P6.1/CB1/A1
P6.0/CB0/A0
RST/NMI/SBWTDIO
PJ.3/TCK
PJ.2/TMS
PJ.1/TDI/TCLK
PJ.0/TDO
TEST/SBWTCK
P5.3/XT2OUT
P5.2/XT2IN
AVSS2
V18
VUSB
VBUS
PU.1/DM
PUR
PU.0/DP
VSSU
P1.0/TA0CLK/ACLK
P1.1/TA0.0
P1.2/TA0.1
P1.3/TA0.2
P1.4/TA0.3
P1.5/TA0.4
P1.6/TA1CLK/CBOUT
P1.7/TA1.0
P2.0/TA1.1
P2.1/TA1.2
P2.2/TA2CLK/SMCLK
P2.3/TA2.0
P2.4/TA2.1
P2.5/TA2.2
P2.6/RTCCLK/DMAE0
P2.7/UCB0STE/UCA0CLK
P3.0/UCB0SIMO/UCB0SDA
P3.1/UCB0SOMI/UCB0SCL
P3.2/UCB0CLK/UCA0STE
P3.3/UCA0TXD/UCA0SIMO

接线部分引脚





LED



USER Key

16.3.1 WDTCTL Register

Watchdog Timer Control Register

Figure 16-2. WDTCTL Register

15	14	13	12	11	10	9	8
WDTPW							
rw-0	rw-0	rw-0	rw-0	r0(w)	rw-1	rw-0	rw-0

7	6	5	4	3	2	1	0
WDTHOLD	WDTSSEL		WDTTMSEL	WDTCNTCL	WDTIS		
rw-0	rw-0	rw-0	rw-0	r0(w)	rw-1	rw-0	rw-0

				0b = Watchdog timer is not stopped. 1b = Watchdog timer is stopped.
6-5	WDTSSEL	RW	0h	Watchdog timer clock source select 00b = SMCLK 01b = ACLK 10b = VLOCLK 11b = X_CLK; VLOCLK in devices that do not support X_CLK
4	WDTTMSEL	RW	0h	Watchdog timer mode select 0b = Watchdog mode 1b = Interval timer mode
3	WDTCNTCL	RW	0h	Watchdog timer counter clear. Setting WDTCNTCL = 1 clears the count value to 0000h. WDTCNTCL is automatically reset. 0b = No action 1b = WDTCNT = 0000h
2-0	WDTIS	RW	4h	Watchdog timer interval select. These bits select the watchdog timer interval to set the WDTIFG flag and/or generate a PUC. 000b = Watchdog clock source /(2^31) (18h:12m:16s at 32.768 kHz) 001b = Watchdog clock source /(2^27) (01h:08m:16s at 32.768 kHz) 010b = Watchdog clock source /(2^23) (00h:04m:16s at 32.768 kHz) 011b = Watchdog clock source /(2^19) (00h:00m:16s at 32.768 kHz) 100b = Watchdog clock source /(2^15) (1 s at 32.768 kHz) 101b = Watchdog clock source /(2^13) (250 ms at 32.768 kHz)

Table 17-3. Timer_A Registers

Offset	Acronym	Register Name	Type	Access	Reset	Section
00h	TAxCTL	Timer_Ax Control	Read/write	Word	0000h	Section 17.3.1
02h	TAxCCTL0	Timer_Ax Capture/Compare Control 0	Read/write	Word	0000h	Section 17.3.3
04h	TAxCCTL1	Timer_Ax Capture/Compare Control 1	Read/write	Word	0000h	Section 17.3.3
06h	TAxCCTL2	Timer_Ax Capture/Compare Control 2	Read/write	Word	0000h	Section 17.3.3
08h	TAxCCTL3	Timer_Ax Capture/Compare Control 3	Read/write	Word	0000h	Section 17.3.3
0Ah	TAxCCTL4	Timer_Ax Capture/Compare Control 4	Read/write	Word	0000h	Section 17.3.3
0Ch	TAxCCTL5	Timer_Ax Capture/Compare Control 5	Read/write	Word	0000h	Section 17.3.3
0Eh	TAxCCTL6	Timer_Ax Capture/Compare Control 6	Read/write	Word	0000h	Section 17.3.3
10h	TAxR	Timer_Ax Counter	Read/write	Word	0000h	Section 17.3.2
12h	TAxCCR0	Timer_Ax Capture/Compare 0	Read/write	Word	0000h	Section 17.3.4
14h	TAxCCR1	Timer_Ax Capture/Compare 1	Read/write	Word	0000h	Section 17.3.4
16h	TAxCCR2	Timer_Ax Capture/Compare 2	Read/write	Word	0000h	Section 17.3.4
18h	TAxCCR3	Timer_Ax Capture/Compare 3	Read/write	Word	0000h	Section 17.3.4
1Ah	TAxCCR4	Timer_Ax Capture/Compare 4	Read/write	Word	0000h	Section 17.3.4
1Ch	TAxCCR5	Timer_Ax Capture/Compare 5	Read/write	Word	0000h	Section 17.3.4
1Eh	TAxCCR6	Timer_Ax Capture/Compare 6	Read/write	Word	0000h	Section 17.3.4
2Eh	TAxIV	Timer_Ax Interrupt Vector	Read only	Word	0000h	Section 17.3.5
20h	TAxEX0	Timer_Ax Expansion 0	Read/write	Word	0000h	Section 17.3.6

Timer_Ax Control Register

Figure 17-16. TAxCTL Register

15	14	13	12	11	10	9	8
Reserved						TASSEL	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ID		MC		Reserved	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	w-(0)	rw-(0)	rw-(0)

Bit	Field	Type	Reset	Description
15-10	Reserved	RW	0h	Reserved
9-8	TASSEL	RW	0h	Timer_A clock source select 00b = TAxCLK 01b = ACLK 10b = SMCLK 11b = INCLK
7-6	ID	RW	0h	Input divider. These bits along with the TAIDEX bits select the divider for the input clock. 00b = /1 01b = /2 10b = /4 11b = /8
5-4	MC	RW	0h	Mode control. Setting MCx = 00h when Timer_A is not in use conserves power. 00b = Stop mode: Timer is halted 01b = Up mode: Timer counts up to TAxCCR0 10b = Continuous mode: Timer counts up to 0FFFFh 11b = Up/down mode: Timer counts up to TAxCCR0 then down to 0000h
3	Reserved	RW	0h	Reserved
2	TACLR	RW	0h	Timer_A clear. Setting this bit resets TAxR, the timer clock divider logic, and the count direction. The TACLR bit is automatically reset and is always read as zero.
1	TAIE	RW	0h	Timer_A interrupt enable. This bit enables the TAIFG interrupt request. 0b = Interrupt disabled

Timer_Ax Capture/Compare Control n Register

Figure 17-18. TAxCCTLn Register

15	14	13	12	11	10	9	8
CM		CCIS		SCS	SCCI	Reserved	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r-(0)	r-(0)	rw-(0)
7	6	5	4	3	2	1	0
OUTMOD			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

10	SCCI	RW	0h	Synchronized capture/compare input. The selected CCI input signal is latched with the EQUx signal and can be read via this bit.
9	Reserved	R	0h	Reserved. Reads as 0.
8	CAP	RW	0h	Capture mode 0b = Compare mode 1b = Capture mode
7-5	OUTMOD	RW	0h	Output mode. Modes 2, 3, 6, and 7 are not useful for TAxCCR0 because EQUx = EQU0. 000b = OUT bit value 001b = Set 010b = Toggle/reset 011b = Set/reset 100b = Toggle 101b = Reset 110b = Toggle/set 111b = Reset/set
4	CCIE	RW	0h	Capture/compare interrupt enable. This bit enables the interrupt request of the corresponding CCIFG flag. 0b = Interrupt disabled 1b = Interrupt enabled
3	CCI	R	0h	Capture/compare input. The selected input signal can be read by this bit.
2	OUT	RW	0h	Output. For output mode 0, this bit directly controls the state of the output. 0b = Output low 1b = Output high

17.3.4 TAxCCRn Register

Timer_A Capture/Compare n Register

Figure 17-19. TAxCCRn Register

15	14	13	12	11	10	9	8
TAxCCRn							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)
7	6	5	4	3	2	1	0
TAxCCRn							
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Table 17-7. TAxCCRn Register Description

Bit	Field	Type	Reset	Description
15-0	TAxCCR0	RW	0h	Compare mode: TAxCCRn holds the data for the comparison to the timer value in the Timer_A Register, TAR. Capture mode: The Timer_A Register, TAR, is copied into the TAxCCRn register when a capture is performed.