MSP430 Clock System and Timer

College of Computer and Information Science,
Northeastern University

References:
Texas Instruments, “MSP430x1xx Family User’s Guide”
Texas Instruments, “MSP430x15x, MSP430x16x, MSP430x161x MIXED SIGNAL MICROCONTROLLER”,

Outline

- MSP430 basic clock module
- MSP430 Timer A
- Timer A examples
MSP430 Basic Clock Module

- **Clock Sources:**
  - **LFXT1CLK**: Low-frequency/high-frequency oscillator
  - **XT2CLK**: Optional high-frequency oscillator
  - **DCOCLK**: Internal digitally controlled oscillator (DCO)

- **Tmote Sky Configuration:**
  - **LFXT1CLK**: 32.768KHz crystal
  - **XT2CLK**: N/A
  - **DCOCLK**: Built-in DCO with configurable range from <100KHz to 4MHz
MSP430 Basic Clock Module

Clock Signals:

- **ACLK**: Auxiliary clock. The signal is sourced from LFXT1CLK with a divider of 1, 2, 4, or 8. (The calibration program for the serial link sets the divider to 4, but after the calibration it can be changed to any other values.) ACLK can be used as the clock signal for Timer A and Timer B.

- **MCLK**: Master clock. The signal can be sourced from LFXT1CLK, XT2CLK (if available), or DCOCLK with a divider of 1, 2, 4, or 8. MCLK is used by the CPU and system.

- **SMCLK**: Sub-main clock. The signal is sourced from either XT2CLK (if available), or DCOCLK with a divider of 1, 2, 4, or 8. SMCLK can be used as the clock signal for Timer A and Timer B.
# Clock System Registers

## DCOCTL, DCO Control Register

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCOx</td>
<td>MODx</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rw-0</td>
<td>rw-1</td>
<td>rw-1</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
</tr>
</tbody>
</table>

- **DCOx** Bits: DCO frequency select. These bits select which of the eight discrete DCO frequencies of the RSELx setting is selected.
- **MODx** Bits: Modulator selection. These bits define how often the $f_{DCO+1}$ frequency is used within a period of 32 DCOCLK cycles. During the remaining clock cycles $(32-MOD)$ the $f_{DCO}$ frequency is used. Not useable when DCOx=7.
### BCSCTL1, Basic Clock System Control Register 1

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>XT2OFF</td>
<td>rw-(1)</td>
</tr>
<tr>
<td>6</td>
<td>XTS</td>
<td>rw-(0)</td>
</tr>
<tr>
<td>5</td>
<td>DIVAx</td>
<td>rw-(0)</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>rw-(0)</td>
</tr>
<tr>
<td>3</td>
<td>XT5V</td>
<td>rw-0</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>rw-1</td>
</tr>
<tr>
<td>1</td>
<td>RSELx</td>
<td>rw-0</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td>rw-0</td>
</tr>
</tbody>
</table>

- **XT2OFF (Bit 7)**: Off. This bit turns off the XT2 oscillator.
  - 0: XT2 is on
  - 1: XT2 is off if it is not used for MCLK or SMCLK.

- **XTS (Bit 6)**: LFXT1 mode select.
  - 0: Low frequency mode
  - 1: High frequency mode

- **DIVAx (Bits 5-4)**: Divider for ACLK
  - 00: /1
  - 01: /2
  - 10: /4
  - 11: /8

- **XT5V (Bit 3)**: Unused. XT5V should always be reset.

- **RSELx (Bits 2-0)**: Resistor Select. The internal resistor is selected in eight different steps.
  - The value of the resistor defines the nominal frequency. The lowest nominal frequency is selected by setting RSELx=0.
BCSCTL2, Basic Clock System Control Register 2

<table>
<thead>
<tr>
<th>Bit 7-6</th>
<th>Bit 5-4</th>
<th>Bit 3</th>
<th>Bit 2-1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELMx</td>
<td>DIVMx</td>
<td>SELS</td>
<td>DIVSx</td>
<td>DCOR</td>
</tr>
<tr>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-0</td>
<td>rw-0</td>
</tr>
</tbody>
</table>

**SELMx** Bits
Select MCLK. These bits select the MCLK source.
- 00: DCOCLK
- 01: DCOCLK
- 10: XT2CLK when XT2 oscillator present on-chip. LFXT1CLK when XT2 oscillator not present on-chip.
- 11: LFXT1CLK

**DIVMx** Bits
Divider for MCLK
- 00: /1
- 01: /2
- 10: /4
- 11: /8

**SELS** Bit 3
Select SMCLK. This bit selects the SMCLK source.
- 0: DCOCLK
- 1: XT2CLK when XT2 oscillator present on-chip. LFXT1CLK when XT2 oscillator not present on-chip.

**DIVSx** Bits
Divider for SMCLK
- 00: /1
- 01: /2
- 10: /4
- 11: /8

**DCOR** Bit 0
DCO resistor select
- 0: Internal resistor
- 1: External resistor
MSP430 Timer_A

- A 16-bit counter
- 4 modes of operation – Stop, Up, Continuous, Up/Down
- 3 capture/compare registers (CCRx)
- 2 interrupt vectors – TACCR0 and TAIIV
Modes of Operation: Up Mode

Diagram showing the relationship between Timer Clock, Timer, Set TAIFG, and Set TACCR0 CCIFG.
Modes of Operation: Continuous Mode
Modes of Operation:

Up/Down Mode
Timer_A Interrupt Vectors

- TACCR0 interrupt vector for CCIFG of CCR0
- TAIV interrupt vector for TAIFG and CCIFGs of CCR1, CCR2
## Timer_A Registers

- **TACTL, Timer_A Control Register (PART 1)**

<table>
<thead>
<tr>
<th></th>
<th>15</th>
<th>14</th>
<th>13</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Unused</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>r/w-(0)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Unused Bits**: 15-10

- **TASSELx Bits**
  - **Timer_A clock source select**
  - 00: TACLK
  - 01: ACLK
  - 10: SMCLK
  - 11: INCLK
<table>
<thead>
<tr>
<th>IDx</th>
<th>MCx</th>
<th>Unused</th>
<th>TACLRL</th>
<th>TAIE</th>
<th>TAIFG</th>
</tr>
</thead>
<tbody>
<tr>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>w-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
</tr>
</tbody>
</table>

**IDx**

Bits
- 7-6: Input divider. These bits select the divider for the input clock.
  - 00: /1
  - 01: /2
  - 10: /4
  - 11: /8

**MCx**

Bits
- 5-4: Mode control. Setting MCx = 00h when Timer_A is not in use conserves power.
  - 00: Stop mode: the timer is halted
  - 01: Up mode: the timer counts up to TACCR0
  - 10: Continuous mode: the timer counts up to 0FFFFFFh
  - 11: Up/down mode: the timer counts up to TACCR0 then down to 00000h

**Unused**

Bit 3: Unused

**TACLRL**

Bit 2: Timer_A clear. Setting this bit resets TAR, the TACLK divider, and the count direction. The TACLRL bit is automatically reset and is always read as zero.

**TAIE**

Bit 1: Timer_A interrupt enable. This bit enables the TAIFG interrupt request.
- 0: Interrupt disabled
- 1: Interrupt enabled

**TAIFG**

Bit 0: Timer_A interrupt flag
- 0: No interrupt pending
- 1: Interrupt pending
### TACCTLx, Capture/Compare Control Register

<table>
<thead>
<tr>
<th>Bit 15</th>
<th>Bit 14</th>
<th>Bit 13</th>
<th>Bit 12</th>
<th>Bit 11</th>
<th>Bit 10</th>
<th>Bit 9</th>
<th>Bit 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMx</td>
<td>CCISx</td>
<td>SCS</td>
<td>SCCI</td>
<td>Unused</td>
<td>CAP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>r-(0)</td>
<td>r-(0)</td>
<td>rw-(0)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTMODx</td>
<td>CCIE</td>
<td>CCI</td>
<td>OUT</td>
<td>COV</td>
<td>CCI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>r</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
<td>rw-(0)</td>
</tr>
</tbody>
</table>

**CAP** Bit 8  
Capture mode  
0  Compare mode  
1  Capture mode

**CCIE** Bit 4  
Capture/compare interrupt enable. This bit enables the interrupt request of the corresponding CCI FG flag.  
0  Interrupt disabled  
1  Interrupt enabled

**CCI FG** Bit 0  
Capture/compare interrupt flag  
0  No interrupt pending  
1  Interrupt pending
## TAIV, Timer_A Interrupt Vector Register

<table>
<thead>
<tr>
<th>TAIV Contents</th>
<th>Interrupt Source</th>
<th>Interrupt Flag</th>
<th>Interrupt Priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>00h</td>
<td>No interrupt pending</td>
<td></td>
<td>–</td>
</tr>
<tr>
<td>02h</td>
<td>Capture/compare 1</td>
<td>TACCR1 CCIFG</td>
<td>Highest</td>
</tr>
<tr>
<td>04h</td>
<td>Capture/compare 2</td>
<td>TACCR2 CCIFG</td>
<td></td>
</tr>
<tr>
<td>06h</td>
<td>Reserved</td>
<td></td>
<td>–</td>
</tr>
<tr>
<td>08h</td>
<td>Reserved</td>
<td></td>
<td>–</td>
</tr>
<tr>
<td>0Ah</td>
<td>Timer overflow</td>
<td>TAIIFG</td>
<td></td>
</tr>
<tr>
<td>0Ch</td>
<td>Reserved</td>
<td></td>
<td>–</td>
</tr>
<tr>
<td>0Eh</td>
<td>Reserved</td>
<td></td>
<td>Lowest</td>
</tr>
</tbody>
</table>
Example 1

Continuous Mode

Output pin P5.4 (Red LED) with toggle rate = \( \frac{32768}{32768} = 1 \text{ Hz} \)

```c
#include "include/include.h"
#include "include/hardware.h"

int main ( void )
{
    WDTCTL = WDTPW + WDTHOLD; // Stop WDT
    P5DIR |= 0x10; // P5.4 output
    CCTL0 = CCIE; // CCR0 interrupt enabled
    CCR0 = 32768;
    TACTL = TASSEL_1 + MC_2; // ACLK, continuous mode
    eint(); // Enable the global interrupt
    LPM0; // Enter low power mode
}

// Timer_A TACCR0 interrupt vector handler
interrupt (TIMERA0_VECTOR) TimerA_procedure( void ){
    _BIS_SR(LPM0_bits + GIE);
    P5OUT ^= 0x10; // Toggle P5.4
    CCR0 += 32768; // Add offset to CCR0
}
```
Example 2

Up Mode

Output pin P5.4 (Red LED) with toggle rate = 32768/(32768) = 1 Hz

```c
#include "include/include.h"
#include "include/hardware.h"

int main ( void )
{
    WDTCTL = WDTPW + WDTHOLD;     // Stop WDT
    P5DIR |= 0x10;                 // P5.4 output
    CCTL0 = CCIE;                  // CCR0 interrupt enabled
    CCR0 = 32767;                  // Timer_A TACCR0 interrupt vector handler
    TACTL = TASSEL_1 + MC_1;      // ACLK, upmode
    _BIS_SR(LPM0_bits + GIE);     // Enable the global interrupt and enter LPM0
}

// Enable the global interrupt and enter LPM0

interrupt (TIMER0_VECTOR) TimerA_procedure ( void ){
    P5OUT ^= 0x10;                // Toggle P5.4
}
```
Example 3

Continuous Mode

Output pin P5.4 with toggle rate = \(32768/(16384)\) = 0.5 Hz

Output pin P5.5 with toggle rate = \(32768/(32768)\) = 1 Hz

Output pin P5.6 with toggle rate = \(32768/(65536)\) = 0.5 Hz

```c
#include "include/include.h"
#include "include/hardware.h"
int main ( void )
{
    WDTCTL = WDTPW + WDTHOLD;    // Stop WDT
    P5DIR |= 0x70;                // P5.4, P5.5, P5.6 in output mode
    CCTL0 = CCIE;                 // CCR0 interrupt enabled
    CCTL1 = CCIE;                 // CCR1 interrupt enabled
    CCTL2 = CCIE;                 // CCR2 interrupt enabled
    CCR0 = 0;
    CCR1 = 0;
    CCR2 = 0;
    TACTL = TASSEL_1 + MC_2+ TAIE; // ACLK, contmode, TAIE enabled
    _BIS_SR(LPM0_bits + GIE);     // Enable the global interrupt and enter LPM0
}
```
Example 3, continued

// Timer_A TACCR0 interrupt vector handler
interrupt (TIMERA0_VECTOR) TimerA0_procedure ( void ){
    P5OUT ^= 0x10; //on TACCR0 Toggle P5.4 (Red LED)
    CCR0 += 16384; // Add offset to CCR0
}

// Timer_A TAIV interrupt vector handler
interrupt (TIMERA1_VECTOR) TimerA1_procedure ( void ){
    switch ( TAIV )
    {
        case 2: P5OUT ^= 0x20; // on TACCR1 CCIFG Toggle P5.5 (Green LED)
            CCR1 += 32768; // Add offset to CCR1
            break;

        case 10: P5OUT ^= 0x40; // on Timer overflow TAIFG Toggle P5.6 (Blue LED)
            break;
    }
}