



Advanced Agricultural Electronics Networks Development

Lecture No. 6, Clock systems in the MSP 430

BAE 5030-352

Spring 2010

Dr. Marvin Stone

Biosystems and Agricultural Engineering
Oklahoma State University

Relationship between Frequency and Power

- CMOS – Logic gates

- Low power

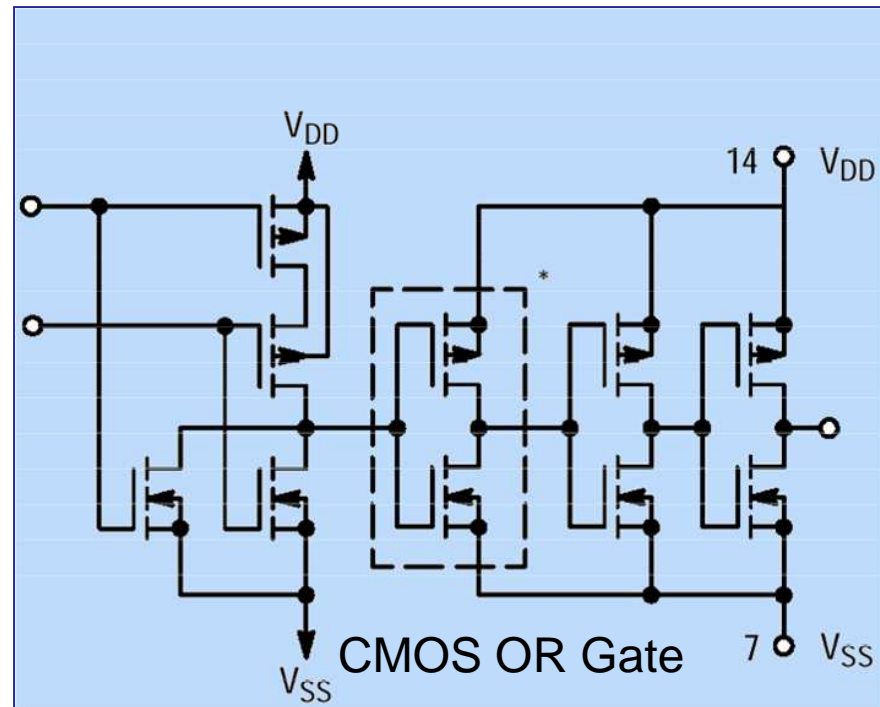
- When one gate of a pair is ON, the other is OFF
- Power flows only during switching and due to “leakage”
- Power demand is primarily proportional to frequency

- High input impedance

- Static Sensitive

- Inputs should set

- Floating input = chatter

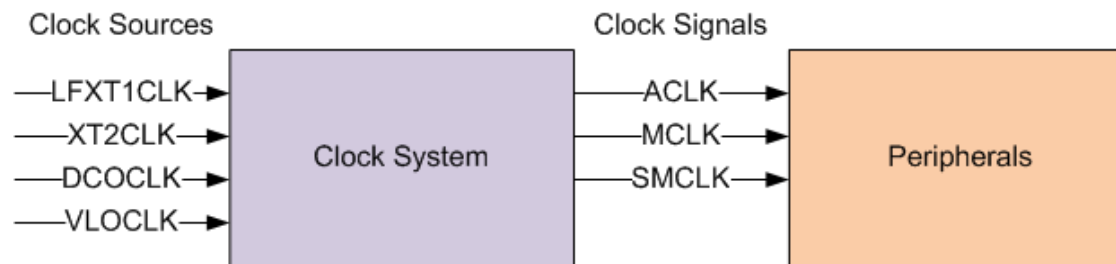


Using clock speed to reduce power demand

- Higher switching rates increase power demand
- Low power MCU characteristics
 - Ability to select different clock speeds for different parts of the chip
 - Ability to disable CPU and then wake-up later on interrupt
- Minimum power demand can be achieved by
 - Selecting choosing the minimum necessary clock speed for each module
 - “Sleeping” CPU and modules when not needed

MSP430 Clock concept

- High-performance CPU used in very short bursts
 - Fast stabilization of DCO
 - Active and stable in less than 6 μ s
 - Low-frequency auxiliary clock
 - Ultralow-power stand-by mode
 - High-speed master clock
 - High performance signal processing
- Flexible clock distribution and divider system
 - provided to fine tune individual clock requirements
- Four clock sources selectable to drive three clock signals selectable to operate the CPU and peripherals



- On power-up clear (PUC)
 - MCLK and SMCLK are sourced from DCOCLK at ~1.1 MHz
 - ACLK is sourced from LFXT1CLK in LF mode

MSP430 Clock input sources

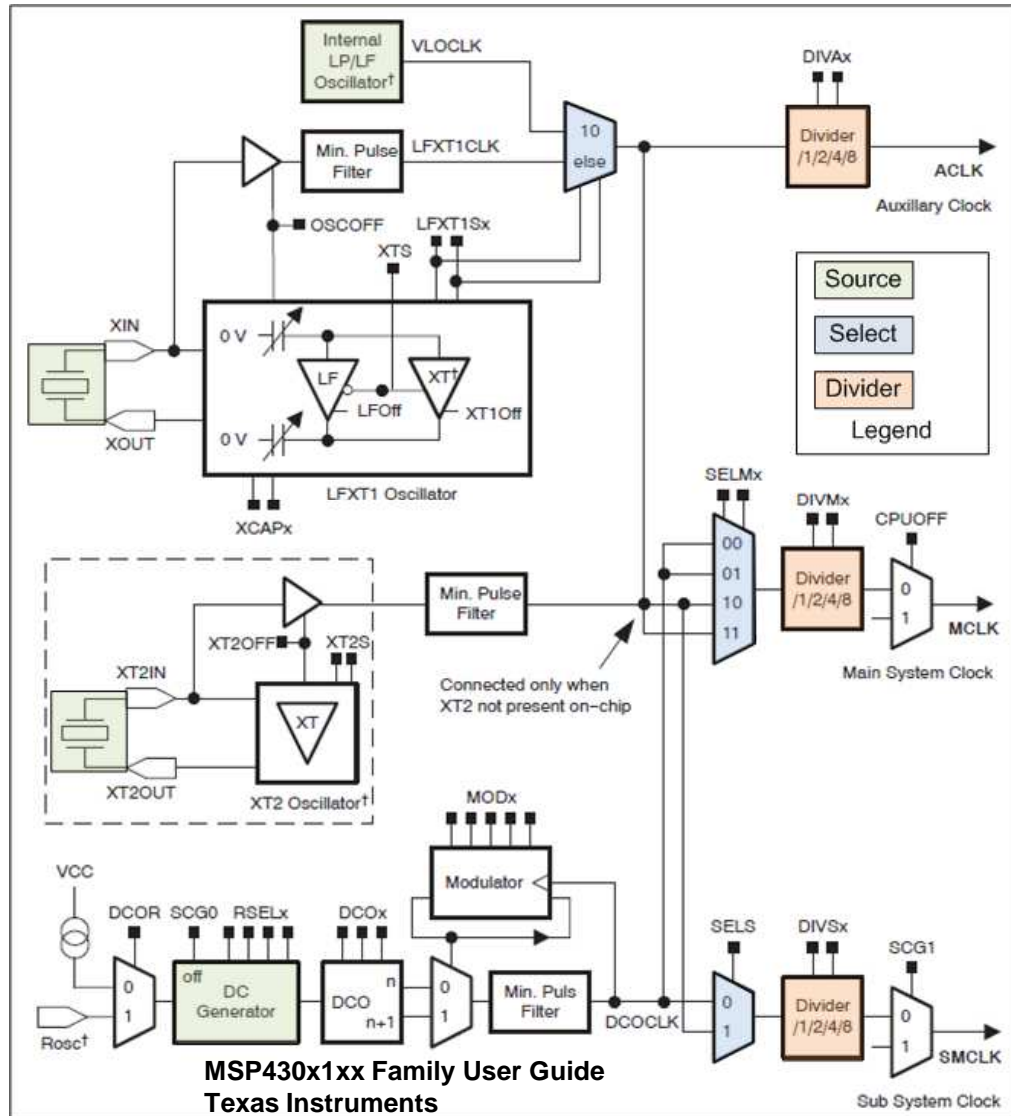
- LFX1CLK - Low-frequency/high-frequency oscillator
 - Low-frequency watch crystals (32768 Hz or 38400 Hz?)
 - Standard crystals (8-16 MHz depending on uC)
 - Resonators (450-kHz to 8-MHz)
- XT2CLK - Optional high-frequency oscillator
 - Standard crystals
 - Resonators
 - External clock sources (450-kHz to 8-MHz)
- DCOCLK - Internal digitally controlled oscillator
 - Uses internal components to produce RC-type characteristics.
- VLOCLK - Very low power, low frequency OSC
 - Internal oscillator, typical 12kHz

MSP430 Clock output signals

- ACLK - Auxiliary clock
 - VLOCLK or buffered LFXT1CLK clock source divided by 1, 2, 4, or 8
 - Software selectable for individual peripheral modules
 - May be used for real-time clock self wake-up function
- MCLK - Master clock
 - Software selectable as LFXT1CLK, XT2CLK, VLOCLK or DCOCLK divided by 1, 2, 4, or 8
 - Used by CPU and system.
- SMCLK - Sub-main clock
 - Software selectable as LFXT1CLK, XT2CLK, VLOCLK or DCOCLK divided by 1, 2, 4, or 8
 - Software selectable for individual peripheral modules.

MSP 430 Clock sources

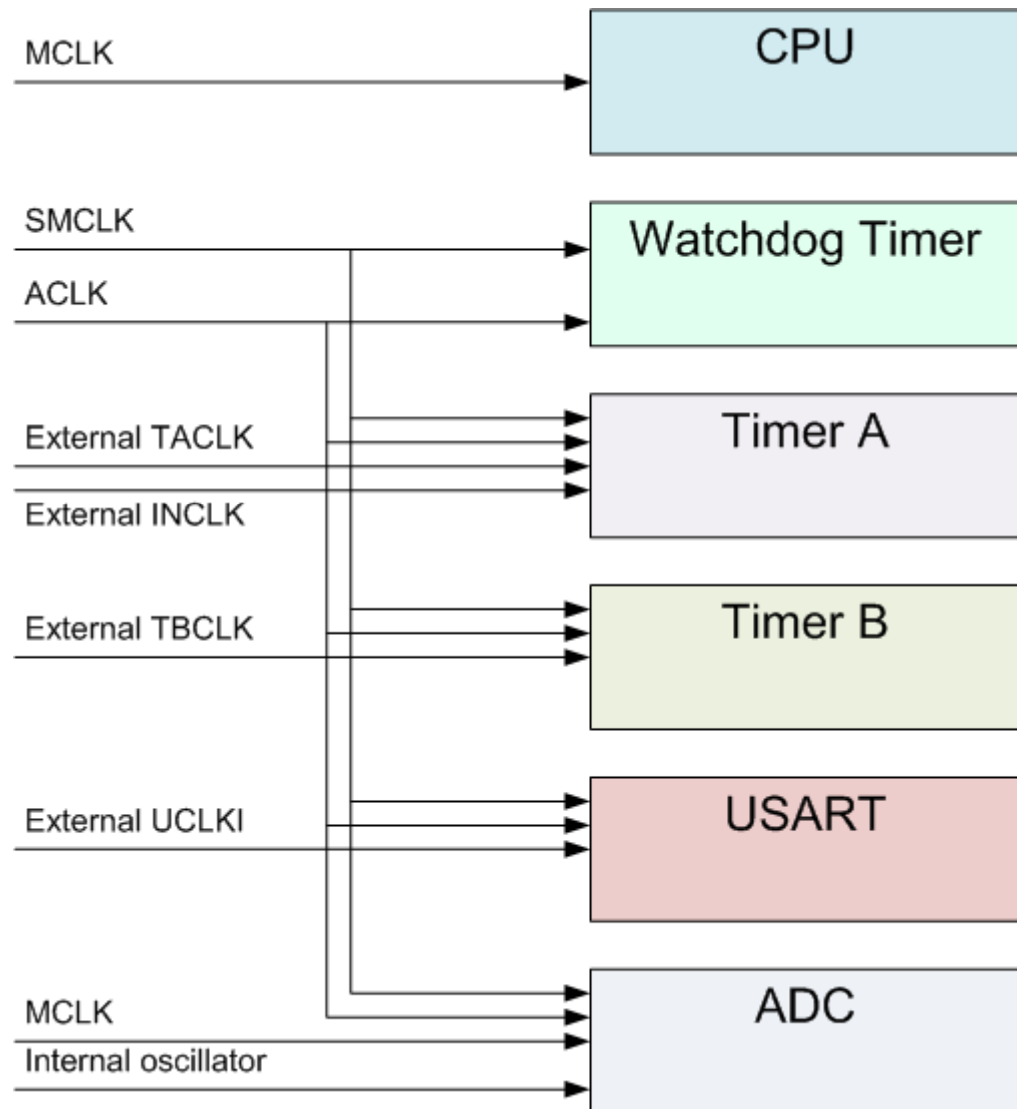
- Not all MSP430 MCUs have XTCLK available
- Systems can be built without a crystal using only the DCO
- Clock signals may be scaled and except for ACLK can be switched OFF



MSP430 Peripherals have selectable clocks

In addition to the control of the clock signal source dividers, each peripheral typically has a 1,2,4,8 divider selection

Some parts may not have all of the clock functions and some have special clock options



Strategy for setting up clocks

- Determine the needed processor speed
 - This will set the clock used for “bursts” if needed
- Identify the necessary peripherals and their requirements
 - Determine the clock speeds for each peripheral
 - Requirements for timing, A/D clocking, and serial communications speed must be accommodated
 - Resolve the speeds to suit the shared clocks
- Develop code to initialize and select the clocks
 - Select the appropriate sources and divided speeds for the sources
 - Select the proper clock signals for each peripheral

eZ430 Clocks

- The only clocks available are:
 - DCOCLK
 - VLOCLK

Clock Setup Demo (test8)

```
#include <msp430.h>

void main(void)
{
    WDTCTL = WDTPW + WDTHOLD;           // Stop WDT
    BCSCTL2 |= DIVS_3;                 // SMCLK = DCO div 8
    TACTL = TASSEL_2+MC_2+TAIE+ID0;    // SMCLK selected, continuous timer mode,
                                        // Timer interrupt enabled, SMCLK div by 2

    P1DIR |= 0x01;                    // P1.0 output

    _BIS_SR(LPM0_bits + GIE);         // Enter LPM0 with general interrupt enable
}

// Timer_A2 Interrupt Vector (TAIV) handler
#pragma vector=TIMERA1_VECTOR        // Select to react to the timer A interrupt
__interrupt void Timer_A(void)
{
    switch( TAIV )                    //Test the Timer A return value, TAIV
    {
        case TAIV_TACCR1: break;      // CCR1 case not handled
        case TAIV_TAIFG: P1OUT ^= 0x01; // Timer A overflow case
        break;
    }
}
```

Example – TI slac080G x2xx examples - msp430x20x3_1_vlo.c

```
void main(void)
{
    volatile unsigned int i;           // Volatile to prevent removal
    WDTCTL = WDTPW + WDTHOLD;         // Stop watchdog timer
    BCSCTL3 |= LFXT1S_2;              // LFXT1 = VLO
    IFG1 &= ~OFIFG;                  // Clear OSCFault flag
    __bis_SR_register(SCG1 + SCG0);   // Stop DCO
    BCSCTL2 |= SELM_3 + DIVM_3;       // MCLK = LFXT1/8
    P1DIR = 0xFF;                     // All P1.x outputs
    P1OUT = 0;                         // All P1.x reset
    P2DIR = 0xFF;                     // All P2.x outputs
    P2OUT = 0;                         // All P2.x reset

    for (;;)
    {
        P1OUT |= 0x01;                // P1.0 set
        for (i = 10; i > 0; i--);     // Delay 1x
        P1OUT &= ~0x01;               // P1.0 reset
        for (i = 1000; i > 0; i--);   // Delay 100x
    }
}
```