The MSP430 Clock Module

Dr. Rhonda Kay Gaede

UAH

CPE/EE 421/521

The MSP430 Clock Module – Basic Clock Systems

MSP430 Clock System
- Low System Cost
- Low Power

- Variety of operating modes driven by ___________, ___________ selectable
- Support for the Burst Mode - when activated system starts and reacts rapidly
- Stability over _______ and __________
The MSP430 Clock Module – MSP430x1xx

- One DCO, digital controlled oscillator
  Generated ______
  frequency controlled by SW + HW
- One LF/XT oscillator
  LF: ______________
  XT: ______________
- Second LF/XT2 oscillator
- Clocks:
  ACLK _______ clock ACLK
  MCLK _______ clock MCLK
  SMCLK _______ system clock

The MSP430 Clock Module – Clock Sources

- LFXT1CLK: Low-frequency/high-frequency oscillator that can be used either with low-frequency 32,768-Hz watch crystals, or standard crystals, resonators, or external clock sources in the 450-kHz to 8-MHz range.
- XT2CLK: Optional high-frequency oscillator that can be used with standard crystals, resonators, or external clock sources in the 450-kHz to 8-MHz range.
- DCOCLK: Internal digitally controlled oscillator (DCO) with RC-type characteristics.
The MSP430 Clock Module – Available Clocks

- **ACLK**: Auxiliary clock. The ACLK is the buffered LFXT1CLK clock source divided by 1, 2, 4, or 8. ACLK is software selectable for ________________.

- **MCLK**: Master clock. MCLK is software selectable as LFXT1CLK, XT2CLK (if available), or DCOCLK. MCLK is divided by 1, 2, 4, or 8. MCLK is used by the ____________

- **SMCLK**: Sub-main clock. SMCLK is software selectable as LFXT1CLK, XT2CLK (if available), or DCOCLK. SMCLK is divided by 1, 2, 4, or 8. SMCLK is software selectable for individual peripheral modules.

The MSP430 Clock Module – Basic Clock Systems: Block Diagram
The MSP430 Clock Module – Basic Clock Systems: Characteristics

- DCOCLK Generated on-chip with 6μs start-up
- 32.768 KHz Watch Crystal - or - High Speed Crystal / Resonator to 8MHz
  - (our system is 4MHz/8MHz high Speed Crystal)
- Flexible clock distribution tree for CPU and peripherals
- Programmable open-loop DCO Clock with internal and external current source
The MSP430 Clock Module – Basic Operation

- After PUC (__________) MCLK and SCLK are sourced by DCOCLK (approx. 800KHz) and ACLK is sourced by LFXT1 in LF mode
- Status register control bits SCG0, SCG1, OSCOFF, and CPUOFF configure the MSP430 basic clock module and enable or disable portions of the basic clock module. The DCOCTL, BCSCTL1, and BCSCTL2 registers configure the __________
- The basic clock can be configured or reconfigured by software __________ during program execution

The MSP430 Clock Module – Low-Power Operation: An Example

- ______ can be configured to oscillate with a low-power 32,786-Hz watch crystal
- ______ can be configured to operate from the on-chip DCO that can be only activated when requested by interrupt-driven events.
- ______ can be configured to operate from either the watch crystal or the DCO, depending on peripheral requirements.
The MSP430 Clock Module – LFXT1 Oscillator

- **LF mode: XTS = 0**
  - 32,768-Hz watch crystal in LF mode. A watch crystal connects to XIN and XOUT without any __________

- **HF mode: XTS = 1**
  - The high-speed crystal or resonator connects to XIN and XOUT and requires ______________ on both terminals. These capacitors should be sized according to the crystal or resonator specifications.

- Software can __________ LFXT1 by setting OSCOFF, if this signal does not source SMCLK or MCLK

The MSP430 Clock Module – XT2 Oscillator

- Similar to LFXT1 in HF mode
- **XT2OFF** bit disables the XT2 oscillator if XT2CLK is not used for MCLK or SMCLK
The MSP430 Clock Module – Digitally-Controlled Oscillator DCO

- Integrated ring oscillator with RC-type characteristics.
  - frequency varies with __________, __________, and ________________.
- DCO frequency can be adjusted by __________ using the DCOx, MODx, and RSELx bits.
- The digital control of the oscillator allows frequency stabilization despite its RC-type characteristics.

The MSP430 Clock Module – Disabling DCO

- Software can disable DCO if not used for MCLK and SMCLK
The MSP430 Clock Module – Adjusting Frequency

- After a PUC, an internal resistor is selected for the DC generator \texttt{RSELx=4}, and \texttt{DCOx=3}, allowing the DCO to start at a _______ frequency.
- MCLK and SMCLK are sourced from DCOCLK. Because the CPU executes code from MCLK, which is sourced from the fast-starting DCO, code execution begins from PUC in less than 6\,\mu s.

The MSP430 Clock Module – Adjusting Frequency

- DCO frequency is determined by:
  - The current injected into the DC generator by either the internal or external resistor. \texttt{DCOR} bit selects the internal or external resistor.
  - The three \texttt{RSELx} bits select one of eight nominal frequency ranges for the DCO.
  - The three \texttt{DCOx} bits divide the DCO range into 8 frequency steps, separated by approx. 10\%.
  - The five \texttt{MODx} bits, switch between the frequency selected by the \texttt{DCOx} bits and the next higher frequency set by \texttt{DCOx+1}.
The MSP430 Clock Module – Basic Clock Control Registers: Overview

- Direct SW Control
- DCOCLK can be Set - Stabilized
- Stable DCOCLK over Temp/Vcc.

**BCSCTL1**

<table>
<thead>
<tr>
<th>XT2Off</th>
<th>XT5V</th>
<th>DIVA.1</th>
<th>DIVA.0</th>
<th>XT5V</th>
<th>Real.2</th>
<th>Real.1</th>
<th>Real.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>rw-1</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-1</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
</tr>
</tbody>
</table>

**DCOCTL**

<table>
<thead>
<tr>
<th>DCO.2</th>
<th>DCO.1</th>
<th>DCO.0</th>
<th>MOD.4</th>
<th>MOD.3</th>
<th>MOD.2</th>
<th>MOD.1</th>
<th>MOD.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
<td>rw-0</td>
</tr>
</tbody>
</table>

Selection of DCO nominal frequency

RSEL.x = Select DCO nominal frequency

DCO.x and MOD.x set exact DCOCLK

... select other clock tree options

---

The Basic Clock Module is configured using control registers DCOCTL, BCSCTL1, and BCSCTL2, and four bits from the CPU status register: SCG1, SCG0, OscOff, and CPUOFF.

User software can modify these control registers from their default condition at any time.

The Basic Clock Module control registers are located in the __________ peripheral map and should be accessed with __________ instructions.

<table>
<thead>
<tr>
<th>Register State</th>
<th>Short Form</th>
<th>Register Type</th>
<th>Address</th>
<th>Initial State</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCO control</td>
<td>DCOCTL</td>
<td>Read/write</td>
<td>056h</td>
<td>060h</td>
</tr>
<tr>
<td>Basic clock</td>
<td>BCSCTL1</td>
<td>Read/write</td>
<td>057h</td>
<td>084h</td>
</tr>
<tr>
<td>system control 1</td>
<td>BCSCTL2</td>
<td>Read/write</td>
<td>058h</td>
<td>reset</td>
</tr>
</tbody>
</table>
The MSP430 Clock Module –
Basic Clock Control Registers: Detail

• Digitally-Controlled Oscillator (DCO) Clock-Frequency Control
  DCOCTL is loaded with a value of 060h with a valid PUC condition

    0
   DCOCTL  DCO.2  DCO.1  DCO.0  MOD.4  MOD.3  MOD.2  MOD.1 MOD.0
  060H   0     1     1   0     0     0      0     0     0

  MOD.0 .. MOD.4: The MOD constant defines how often the discrete frequency \( f_{DCO+1} \) is used within a period of 32 DCOCLK cycles. During the remaining clock cycles \((32–MOD)\) the discrete frequency \( f_{DCO} \) is used.

  DCO.0 .. DCO.2: The DCO constant defines which one of the discrete frequencies is selected. The frequency is defined by the current injected into the dc generator. When the DCO constant is set to ________, no modulation is possible since the __________ feasible frequency has then been selected.

BCSCTL1 is affected by a valid PUC or POR condition.

    0
   BCSCTL1  XT2Off  XTS  DIVA.1  DIVA.0  XT5V  Rsel.0  Rsel.1 Rsel.2
  057h   1     0     0   0     0   1    0    0    0

  Rsel.0 to Rsel.2: 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 Rsel=0.
  XT5V: XT5V should always be reset.
  DIVA.1 – DIVA.0: The selected source for ACLK is divided by:

  \[ \text{DIVA} = 0: 1 \]
  \[ \text{DIVA} = 1: 2 \]
  \[ \text{DIVA} = 2: 4 \]
  \[ \text{DIVA} = 3: 8 \]
The MSP430 Clock Module – Basic Clock Control Registers: Detail

**XTS:** The LFXT1 oscillator operates with a low-frequency or with a high-frequency crystal:
- XTS = 0: The low-frequency oscillator is selected.
- XTS = 1: The high-frequency oscillator is selected.
The oscillator selection must meet the external crystal’s operating condition.

**XT2Off:** The XT2 oscillator is switched on or off:
- XT2Off = 0: the oscillator is on
- XT2Off = 1: the oscillator is off if it is not used for MCLK or SMCLK.

**BCSCTL2** is affected by a valid PUC or POR condition.

<table>
<thead>
<tr>
<th>BCSCTL2</th>
<th>SELM.1 SELM.0 DIVM.1 DIVM.0 SELS DIVS.1 DIVS.0 DCOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>058</td>
<td></td>
</tr>
</tbody>
</table>

**DCOR:** The DCOR bit selects the resistor for injecting current into the dc generator.
- DCOR = 0: Internal resistor on, the oscillator can operate. The ____________ mode is on.
- DCOR = 1: Internal resistor off, the current must be injected ____________ if the DCO output drives any clock using the DCOCLK.

**DIVS.1 – DIVS.0:** The selected source for SMCLK is divided by:
- DIVS = 0:1, DIVS = 1: 2, DIVS = 2: 4, DIVS = 3: 8
The MSP430 Clock Module – 
Basic Clock Control Registers: Detail

Bit3, SELS: Selects the source for generating SMCLK:
SELS = 0: Use the DCOCLK
SELS = 1: Use the XT2CLK signal (in three-oscillator systems) or
   LFXT1CLK signal (in two-oscillator systems)

Bit4, Bit5: The selected source for MCLK is divided by DIVM.0 .. DIVM.1
DIVM = 0: 1, DIVM = 1: 2, DIVM = 2: 4, DIVM = 3: 8

Bit6, Bit7: Selects the source for generating MCLK: SELM.0 .. SELM.1
SELM = 0: Use the DCOCLK
SELM = 1: Use the DCOCLK
SELM = 2: Use the XT2CLK (x13x and x14x devices) or
Use the LFXT1CLK (x11x(1) devices)
SELM = 3: Use the LFXT1CLK

The MSP430 Clock Module – 
Basic Clock Systems: Software FLL Idea

- Basic Clock DCO is an open loop - close with SW+HW
  - A reference frequency e.g. ACLK or 50/60Hz can be used to measure DCOCLK’s
  - Initialization or Periodic software set and stabilizes DCOCLK over reference clock
  - DCOCLK is programmable 100kHz - 5Mhz and stable over voltage and temperature
The MSP430 Clock Module – Basic Clock Systems: Software FLL Implementation

Example: Set DCOLCLK = 1228800, ACLK = 32768

- ACLK/4 captured on CCI2B, DCOLCLK is clock source for Timer_A
- Comparator2 HW captures SMCLK (1228800Hz) in one ACLK/4 (8192Hz) period
- Target Delta = 1228800/8192 = 150
- CCI2BInt ... ; Compute Delta
  cmp #150,Delta ; Delta = 1228800/8192
  jlo IncDCO ; JMP to IncDCO
  DecDCO dec &DCOCTL ; Decrease DCOLCLK
  reti
  IncDCO inc &DCOCTL ; Increase DCOLCLK
  reti

- Delta = 150

Capture/Compare

0
1
2
3

CCI2B

Target 1228800Hz DCOLCLK source for timer

Stable reference ACLK/4, 8192Hz source

The MSP430 Clock Module – Basic Clock Systems: DCO TAPS

- DCOCLK frequency control
  - nominal - injected current into DC generator
    1) internal resistors Rsel2, Rsel1 and Rsel0
    2) an external resistor at Rosc (P2.5/11x)
- Control bits DCO0 to DCO2 set fDCO tap
- Modulation bits MOD0 to MOD4 allow mixing of fDCO and fDCO+1 for precise frequency generation

Example

<table>
<thead>
<tr>
<th>Selected</th>
<th>Frequency (kHz)</th>
<th>Cycle time (ns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>f3:</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>f4:</td>
<td>943</td>
<td>1060</td>
</tr>
<tr>
<td>f5:</td>
<td>1042</td>
<td>960</td>
</tr>
</tbody>
</table>

MOD=19

DCOCLK

Cycle time = ((32-MOD)*t1+MOD*t2)/32 = 1000.625 ns, selected frequency = 1 MHz.
The MSP430 Clock Module – Range (RSELx) and Steps (DCOx)

- The DCO temperature coefficient can be reduced by using an external resistor ROSC to source the current for the DC generator.
- ROSC also allows the DCO to operate at higher frequencies.
  - Internal resistor nominal value is approximately 200 kOhm => DCO to operate up to 5 MHz.
  - External ROSC of approximately 100 kOhm => the DCO can operate up to approximately 10 MHz.
The MSP430 Clock Module – DCO Modulator

• To produce an intermediate effective frequency between \( f_{DCO} \) and \( f_{DCO+1} \)

The modulator mixing formula is:

\[ t = (32 - MODx \times t_{DCO} + MODx \times t_{DCO+1} \]

The MSP430 Clock Module – Fail Safe Operation

• Basic module incorporates an __________ detection fail-safe feature.

• The oscillator fault detector is an analog circuit that monitors the LFXT1CLK (in HF mode) and the XT2CLK.

• An oscillator fault is detected when either clock signal is not present for approximately ________.
  – When an oscillator fault is detected, and when MCLK is sourced from either LFXT1 in HF mode or XT2, MCLK is automatically switched to the DCO for its clock source.
The MSP430 Clock Module – Fail Safe Operation

• When OFIFG is set and OFIE is set, an NMI interrupt is requested. The NMI interrupt service routine can test the OFIFG flag to determine if an oscillator fault occurred. The OFIFG flag must be cleared by software.

The MSP430 Clock Module – Synchronization of Clock Signals

• To avoid race conditions
  – The current clock cycle continues until the next rising edge.
  – The clock remains high until the next rising edge of the new clock.
  – The new clock source is selected and continues with a full high period.
The MSP430 Clock Module – Basic Clock Systems: Examples

- How to select the Crystal Clock

```c
void selectclock(void)
{
    IFG2=0; /* reset interrupt flag register 1 */
    IFG1=0; /* reset interrupt flag register 2 */
    BCSCTL1|=XTS; /*attach HF crystal (4MHz) to XIN/XOUT */
    do {
        /*wait in loop until crystal is stable*/
        IFG1&=~OFIFG;
    }while(OFIFG&IFG1);
    Delay();
    IFG1&=~OFIFG; /*Reset osc. fault flag again*/
}
```

- How to select a clock for MCLK

```c
BCSCTL2=SELM0+SELM1; /*Then set MCLK same as LFXT1CLK*/
TACTL=TASSEL0+TACLR+ID1; /*USE ACLK/4 AS TIMER_A INPUT CLOCK (1MHz) */
```

Adjusting the Basic Clock

The control registers of the Basic Clock are under full software control. If clock requirements other than those of the default from PUC are necessary, the Basic Clock can be configured or reconfigured by software at any time during program execution.

- ACLKGEN from LFXT1 crystal, resonator, or external-clock source and divided by 1, 2, 4, or 8. If no LFXTCLK clock signal is needed in the application, the OscOff bit should be set in the status register.
- SCLKGEN from LFXTCLK, DCOCLK, or XT2CLK (x13x and x14x only) and divided by 1, 2, 4, or 8. The SCG1 bit in the status register enables or disables SMCLK.
- MCLKGEN from LFXTCLK, DCOCLK, or XT2CLK (x13x and x14x only) and divided by 1, 2, 4, or 8. When set, the CPUOff bit in the status register enables or disables MCLK.
- DCOCLK frequency is adjusted using the RSEL, DCO, and MOD bits. The DCOCLK clock source is stopped when not used, and the dc generator can be disabled by the SCG0 bit in the status register (when set).
- The XT2 oscillator sources XT2CLK (x13x and x14x only) by clearing the XT2Off bit.
The MSP430 Clock Module –
Interrupt Service Routine Declaration

// Func. declaration
Interrupt[int_vector] void myISR (Void);

Interrupt[int_vector] void myISR (Void)
{
  // ISR code
}

EXAMPLE

Interrupt[TIMERA0_VECTOR] void myISR (Void);

Interrupt[TIMERA0_VECTOR] void myISR (Void)
{
  // ISR code
}

The MSP430 Clock Module –
Interrupt Vectors

/*************************************************************
* Interrupt Vectors (offset from 0xFFE0)
**************************************************************/
#define PORT2_VECTOR        1 * 2  /* 0xFFE2 Port 2 */
#define UART1TX_VECTOR      2 * 2  /* 0xFFE4 UART 1 Transmit */
#define UART1RX_VECTOR      3 * 2  /* 0xFFE6 UART 1 Receive */
#define PORT1_VECTOR        4 * 2  /* 0xFFE8 Port 1 */
#define TIMERA1_VECTOR      5 * 2  /* 0xFFEA Timer A CC1-2, TA */
#define TIMERA0_VECTOR      6 * 2  /* 0xFFEC Timer A CC0 */
#define ADC_VECTOR          7 * 2  /* 0xFFEE ADC */
#define UART0TX_VECTOR      8 * 2  /* 0xFFF0 UART 0 Transmit */
#define UART0RXVector       9 * 2  /* 0xFFF2 UART 0 Receive */
#define WDT_VECTOR          10 * 2  /* 0xFFF4 Watchdog Timer */
#define COMPARATORA_VECTOR  11 * 2  /* 0xFFF6 Comparator A */
#define TIMERB1_VECTOR      12 * 2  /* 0xFFF8 Timer B 1-7 */
#define TIMERB0_VECTOR      13 * 2  /* 0xFFF9 Timer B 0 */
#define NMI_VECTOR          14 * 2  /* 0xFFFc Non-maskable */
#define RESET_VECTOR        15 * 2  /* 0xFFFF Reset [Highest Pr.] */
The MSP430 Clock Module – Watchdog Timer: Overview

Watchdog/Timer two functions:
- SW Watchdog Mode
- Interval Timer Mode

Features of the Watchdog Timer include:
- Eight software-selectable time intervals
- Two operating modes: as watchdog or interval timer
- Expiration of the time interval in watchdog mode, which generates a system reset; or in timer mode, which generates an interrupt request
- Safeguards which ensure that writing to the WDT control register is only possible using a password
- Support of ultralow-power using the hold mode

The MSP430 Clock Module – Watchdog Timer: Block Diagram
The MSP430 Clock Module – Watchdog Timer: Registers

Watchdog Timer Counter

The watchdog-timer counter (WDTCNT) is a 16-bit up-counter that is not directly accessible by software. The WDTCNT is controlled through the watchdog-timer control register (WDTCTL), which is a 16-bit read/write register located at the low byte of word address 0120h. Any read or write access must be done using word instructions with no suffix or .w suffix. In both operating modes (watchdog or timer), it is only possible to write to WDTCTL using the correct password.

Watchdog Timer Control Register

<table>
<thead>
<tr>
<th>WDTCTL 0120h</th>
<th>MDB, HighByte</th>
<th>R/W</th>
<th>EQU</th>
<th>WDT 16-bit Control Register with Write Protection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Password Compare</td>
<td>600h</td>
<td>MDB, LowByte</td>
<td>R/W</td>
<td>HOLD</td>
</tr>
</tbody>
</table>

Bits 0, 1: Bits ISO and IS1 select one of four taps from the WDTCNT, as described in following table. Assuming f_{crystal} = 32,768 Hz and f_{System} = 1 MHz, the following intervals are possible:

<table>
<thead>
<tr>
<th>SSEL</th>
<th>IS1</th>
<th>IS0</th>
<th>Interval [ms]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.064 (t_{SMCLK} \times 2^6)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.5 (t_{SMCLK} \times 2^9)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1.9 (t_{ACLK} \times 2^6)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>8 (t_{SMCLK} \times 2^{13})</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>16.0 (t_{ACLK} \times 2^9)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>32 (t_{SMCLK} \times 2^{15}) ← Value after PUC (reset)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>250 (t_{ACLK} \times 2^{13})</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1000 (t_{ACLK} \times 2^{15})</td>
</tr>
</tbody>
</table>

Bit 2: The SSEL bit selects the clock source for WDTCNT. SSEL = 0: SMCLK, SSEL = 1: ACLK.

Bit 3: Counter clear bit. In both operating modes, writing a 1 to this bit restarts the WDTCNT at 00000h. The value read is not defined.

Bit 4: The TMSEL bit selects the operating mode: watchdog or timer. TMSEL = 0: Watchdog, TMSEL = 1: Interval-timer.
The MSP430 Clock Module – Watchdog Timer: Registers

Bit 5: The NMI bit selects the function of the RST/NMI input pin. It is cleared by the PUC signal.

- NMI = 0: The RST/NMI input works as reset input.
  - As long as the RST/NMI pin is held low, the internal signal is active (level sensitive).
- NMI = 1: The RST/NMI input works as an edge-sensitive non-maskable interrupt input.

Bit 6: If the NMI function is selected, this bit selects the activating edge of the RST/NMI input. It is cleared by the PUC signal.

- NMIES = 0: A rising edge triggers an NMI interrupt.
- NMIES = 1: A falling edge triggers an NMI interrupt.

CAUTION: Changing the NMIES bit with software can generate an NMI interrupt.

Bit 7: This bit stops the operation of the watchdog counter. The clock multiplexer is disabled and the counter stops incrementing. It holds the last value until the hold bit is reset and the operation continues. It is cleared by the PUC signal.

- HOLD = 0: The WDT is fully active.
- HOLD = 1: The clock multiplexer and counter are stopped.

The MSP430 Clock Module – Watchdog Timer: Interrupt Function

- The Watchdog Timer (WDT) uses two bits in the SFRs for interrupt control.
  - The WDT interrupt flag (WDTIFG) (located in IFG1.0, initial state is reset)
  - The WDT interrupt enable (WDTIE) (located in IE1.0, initial state is reset)

  - When using the watchdog mode, the WDTIFG flag is used by the reset interrupt service routine to determine if the watchdog caused the device to reset. If the flag is set, then the Watchdog Timer initiated the reset condition (either by timing out or by a security key violation). If the flag is cleared, then the PUC was caused by a different source. See chapter 3 for more details on the PUC and POR signals.

  - When using the Watchdog Timer in interval-timer mode, the WDTIFG flag is set after the selected time interval and a watchdog interval-timer interrupt is requested. The interrupt vector address in interval-timer mode is different from that in watchdog mode. In interval-timer mode, the WDTIFG flag is reset automatically when the interrupt is serviced.

  - The WDTIE bit is used to enable or disable the interrupt from the Watchdog Timer when it is being used in interval-timer mode. Also, the GIE bit enables or disables the interrupt from the Watchdog Timer when it is being used in interval-timer mode.
The MSP430 Clock Module – Watchdog Timer: Timer Mode

- Setting WDTCTL register bit TMSEL to 1 selects the timer mode. This mode provides periodic interrupts at the selected time interval. A time interval can also be initiated by writing a 1 to bit CNTCL in the WDTCTL register.

- When the WDT is configured to operate in timer mode, the WDTIFG flag is set after the selected time interval, and it requests a standard interrupt service. The WDT interrupt flag is a single-source interrupt flag and is automatically reset when it is serviced. The enable bit remains unchanged. In interval-timer mode, the WDT interrupt-enable bit and the GIE bit must be set to allow the WDT to request an interrupt. The interrupt vector address in timer mode is different from that in watchdog mode.

The MSP430 Clock Module - Watchdog Timer: Examples

- **How to select timer mode**
  
  ```c
  /* WDT is clocked by fACLK (assumed 32Khz) */
  WDTCL=WDT_ADLY_250; // WDT 250MS/4 INTERVAL TIMER
  IE1 |=WDTIE;        // ENABLE WDT INTERRUPT
  ```

- **How to stop watchdog timer**

  ```c
  WDTCTL=WDT_PW + WDTHOLD; // stop watchdog timer
  ```

- **Assembly programming**

  ```assembly
  WDT_key .equ 05A00h       ; Key to access WDT
  WDTStop mov #(WDT_key+80h),&WDTCTL; Hold Watchdog
  WDT250 mov #(WDT_key+1Dh),&WDTCTL; WDT, 250ms Interval
  ```