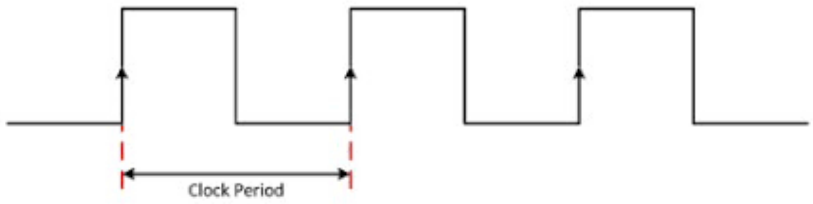
Lecture 27

Clock System & Timer Interrupts II & Low Power Modes

Last Time: The Clock System

A **clock signal** is a **square wave** whose edges

- trigger hardware throughout the device and
- synchronize different components of the MCU



Most MCU application need at least two clock signals

- **A fast clock to drive the CPU**
  - which can be started and stopped rapidly to conserve energy
  - does not need to be particularly accurate
- A **slower clock** that runs continuously **to monitor real time**
  - uses little power
  - is more accurate

Clock System of MSP430FR6989

Our MCU has several clock generators (oscillators) and clock signals

To learn more about them and to be able to configure and use them you have to read the user manual: slau367p.pdf

TEXAS INSTRUMENTS

Chapter 3

SLAU367P–October 2012–Revised April 2020

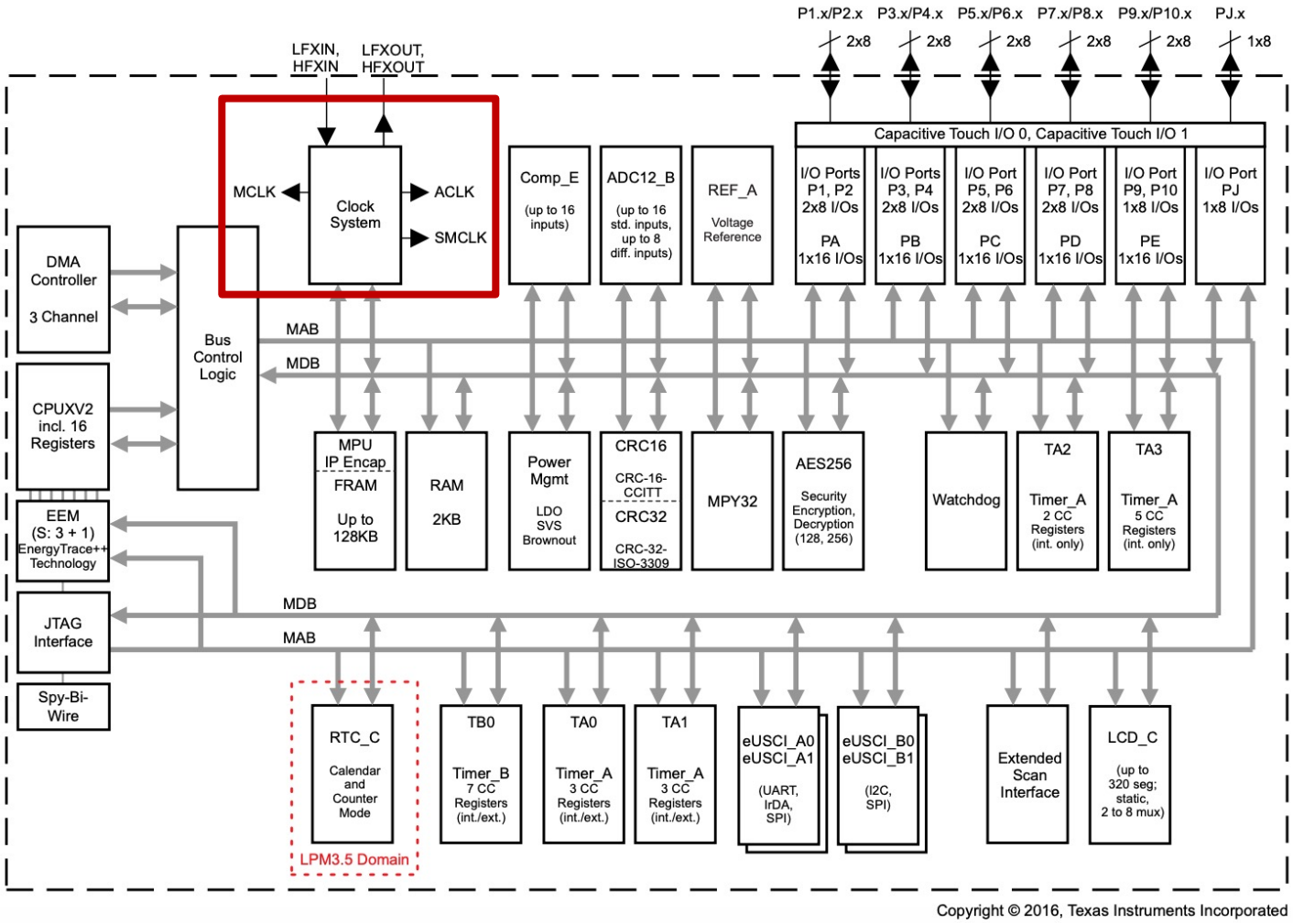
Clock System (CS) Module

This chapter describes the operation of the clock system, which is implemented in all devices.

Topic **Page**

Clock System of MSP430FR6989

Many more peripherals on our MCU



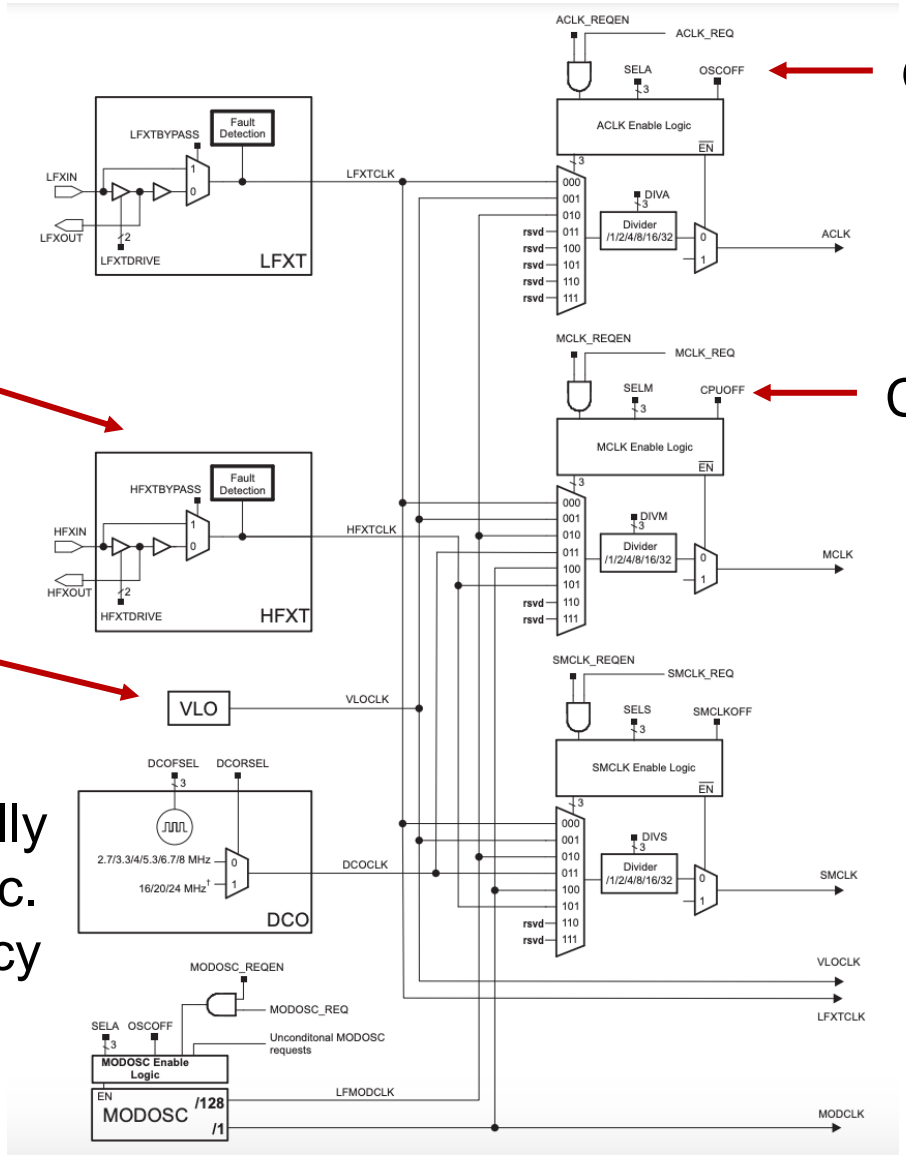
Clock System of MSP430FR6989

Low Freq. Crystal Osc.

High Freq. Crystal Osc.

VLO – Very Low Osc.

DCO – Digitally controlled Osc. High frequency



OSCOFF in SR

CPUOFF in SR

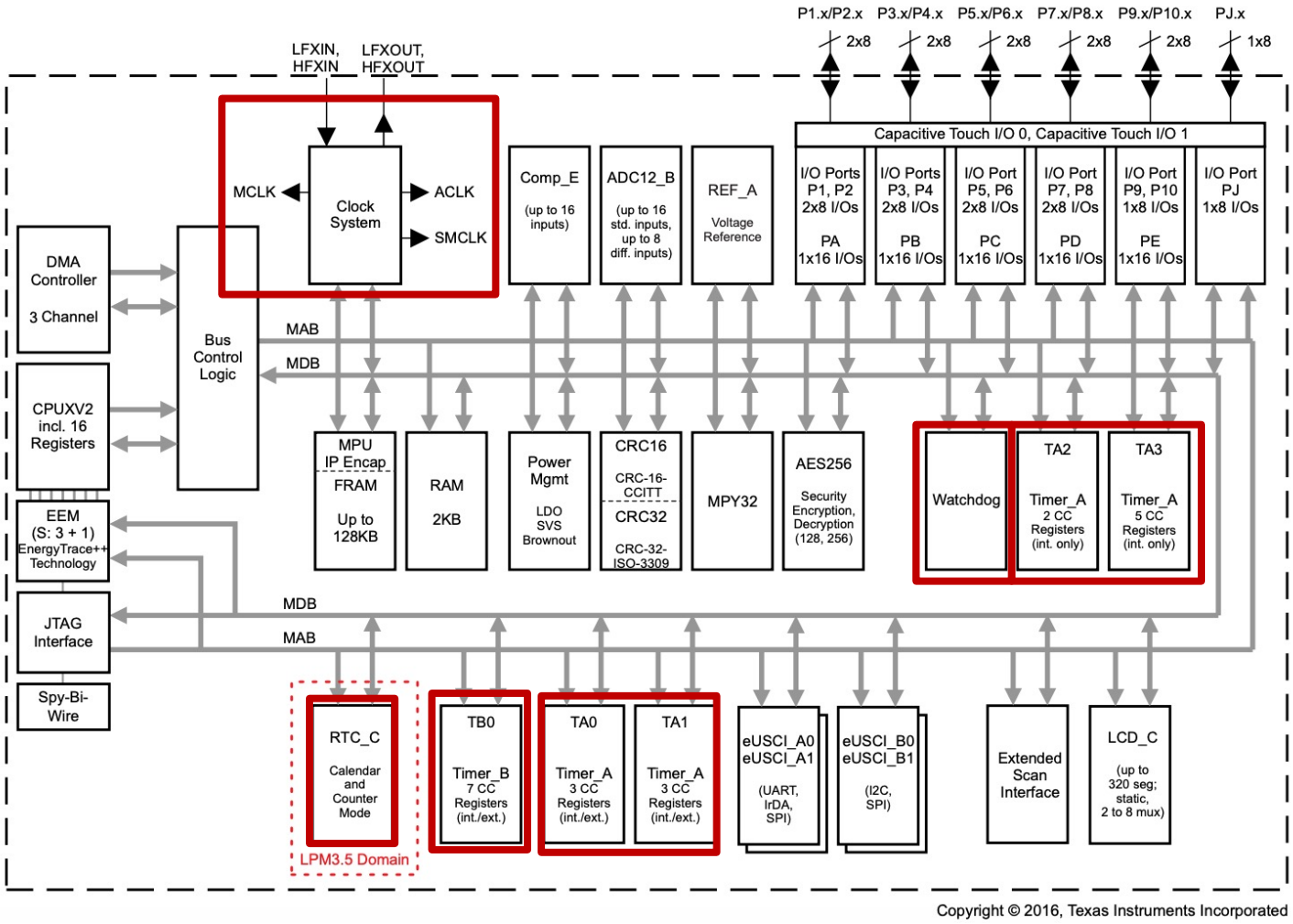
ACLK – Auxiliary clock
~ 32 kHz

MCLK – Master clock
Drives the CPU
~ 1 MHz

SMCLK – Subsystem Master clock

Timers in MSP430FR6989

The clock system feeds into multiple timers



Timer B

16-bit timers that can be used for different purposes

- **Interval timer** (programmable to be 8, 10, 12, or 16 bits)
- Supports capture/compare
- Supports Pulse Width Modulation (PWM)

Clock source can be ACLK, SMCLK or external; selected with `TBSSEL` bits

Four mode of operations selected by the `MC` bits

Table 26-1. Timer Modes

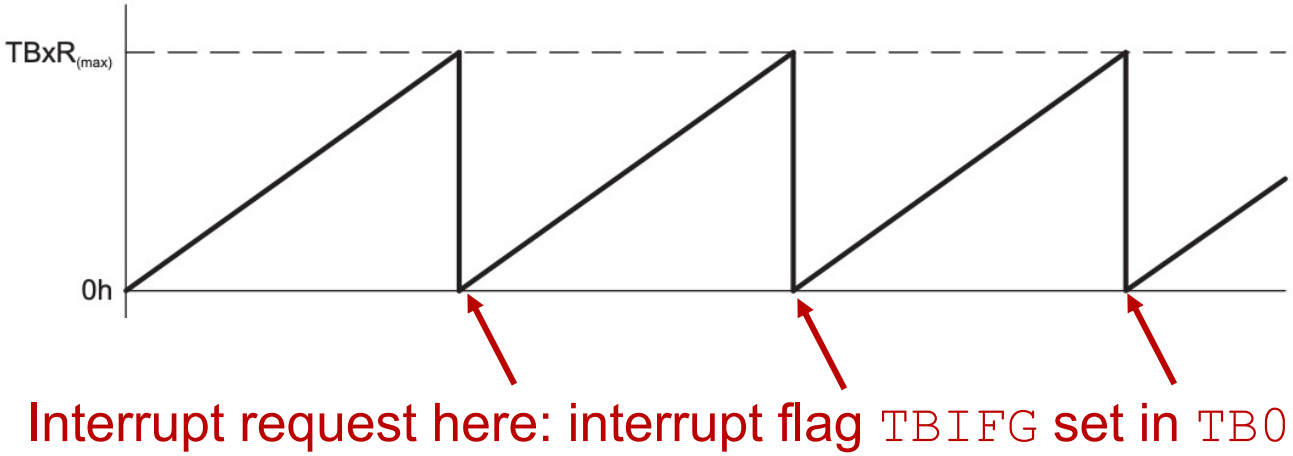
MC	Mode	Description
00	Stop	The timer is halted.
01	Up	The timer repeatedly counts from zero to the value of compare register TBxCL0.
10	Continuous	The timer repeatedly counts from zero to the value selected by the CNTL bits.
11	Up/down	The timer repeatedly counts from zero up to the value of TBxCL0 and then back down to zero.

For full configuration information refer to the user manual **slau367p.pdf**

Working with Timer B0 Interrupts

We will use Timer B0 in continuous mode to trigger periodic interrupts
And toggle the red LED when an interrupt is generated ⇒ **Blinky version 3**

We configure Timer B0 to throw periodic interrupts



We write an interrupt service routine (ISR) to serve Timer B interrupts
Populate the interrupt vector table (IVT)

Timer B Interrupts

I have configured Timer B for you – if you want to use a different configuration (e.g., with a higher frequency), you have to read the user's manual (, figure out what the registers do, and set the values

```
; Configure Timer B0 to raise interrupts
            bis.w   #TBCLR, &TB0CTL
            bis.w   #TBSSEL__ACLK, &TB0CTL      ; connected to ACLK
            bis.w   #MC__CONTINUOUS, &TB0CTL    ; continuous mode
            bis.w   #TBIE, &TB0CTL              ; enable interrupts
```

You know how to write an ISR

How do we populate the IVT?

Check out **Table 6-4. Interrupt Sources, Flags, and Vectors** in the data sheet for MSP430FR6869

https://www.ti.com/lit/ds/symlink/msp430fr6989.pdf

Also available on Carmen: **SLAS789D.pdf**

Timer B Interrupts

Table 6-4. Interrupt Sources, Flags, and Vectors

INTERRUPT SOURCE	INTERRUPT FLAG	SYSTEM INTERRUPT	WORD ADDRESS	PRIORITY
System Reset Power up, Brownout, Supply Supervisor External Reset $\overline{RST}$ Watchdog time-out (watchdog mode) WDT, FRCTL MPU, CS, PMM password violation FRAM uncorrectable bit error detection MPU segment violation FRAM access time error Software POR, BOR	SVSHIFG PMMRSTIFG WDTIFG WDTPW, FRCTLPW, MPUPW, CSPW, PMMPW UBDIFG MPUSEGIIFG, MPUSEG1IFG, MPUSEG2IFG, MPUSEG3IFG ACCTEIFG PMMPORIFG, PMMBORIFG (SYSRSTIV) (1) (2)	Reset	0FFFEh	Highest
System NMI Vacant memory access JTAG mailbox FRAM bit error detection MPU segment violation	VMAIFG JMBINIFG, JMBOUTIFG CBDIFG, UBDIFG MPUSEGIIFG, MPUSEG1IFG, MPUSEG2IFG, MPUSEG3IFG (SYSSNIV) (1) (3)	(Non)maskable	0FFFCh	
User NMI External NMI Oscillator fault	NMIIFG, OFIFG (SYSUNIV) (1) (3)	(Non)maskable	0FFFAh	
Comparator_E	Comparator_E interrupt flags (CEIV) (1)	Maskable	0FFF8h	
Timer_B TB0	TB0CCR0.CCIFG	Maskable	0FFF6h	
Timer_B TB0	TB0CCR1.CCIFG to TB0CCR6.CCIFG, TB0CTL.TBIFG (TB0IV)(1)	Maskable	0FFF4h	
Watchdog timer (interval timer mode)	WDTIFG	Maskable	0FFF2h	

address for interrupt vector is **0xFFF4**

Recall: Populating the IVT

This is how we did it for Port 1, we'll do the same for Timer B

We start by finding the **word address for interrupt vector: 0xFFDA**

We locate the word address in the linker file "lnk_msp430fr6989.cmd"

lnk_msp430fr6989.cmd | main.asm | main.asm | *main.asm | m

```
103     INT34                   : origin = 0xFFD4, length = 0x0002
104     INT35                   : origin = 0xFFD6, length = 0x0002
105     INT36                   : origin = 0xFFD8, length = 0x0002
106     INT37                   : origin = 0xFFDA, length = 0x0002
107     INT38                   : origin = 0xFFDC, length = 0x0002
108     INT39                   : origin = 0xFFDE, length = 0x0002
```

We add the **label of the ISR** to the Interrupt Vectors (at the end of *.asm)

```
;-----------------------------------
; Interrupt Vectors
;-----------------------------------
            .sect   ".int37"
            .short  P1_ISR

            .sect   ".reset"
            .short  RESET
```

Identifies address 00xFFDA

Label of ISR

Recap: Operation Modes

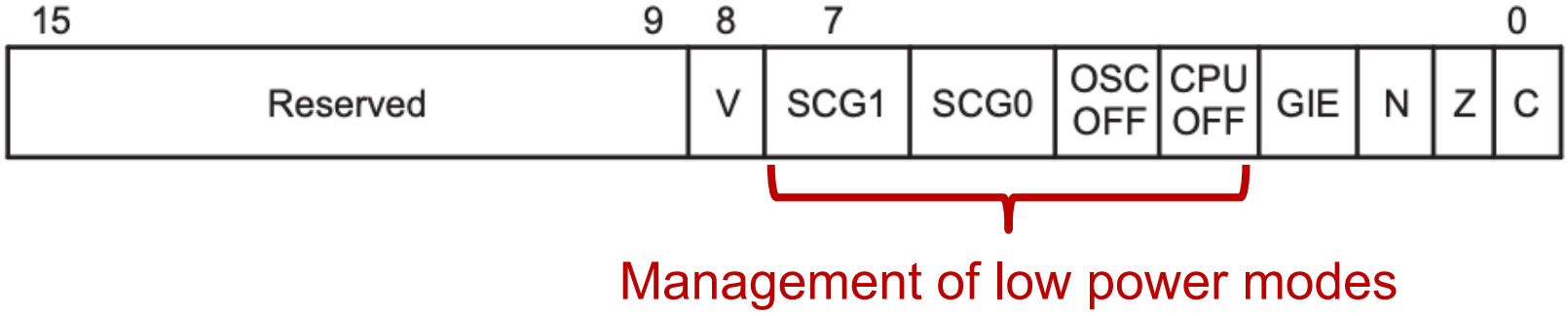
The MSP430 family has five low power modes LPM0 through LPM4

+ two additional modes LPM3.5 or LPM4.5 which disable the Power Management Module

The operating mode is chosen based on different needs

- Power consumption – Some modes designed to provide ultra-low power
- Speed and data throughput
- Minimization of individual peripheral current consumption

The low-power modes are configured in the SR



Operation Modes

Table 1-2. Operation Modes

SCG1 (1)	SCG0	OSCOFF (1)	CPUOFF (1)	Mode	CPU and Clocks Status (2)
0	0	0	0	Active	CPU, MCLK are active. ACLK is active. SMCLK optionally active (SMCLKOFF = 0). DCO is enabled if sources ACLK, MCLK, or SMCLK (SMCLKOFF = 0). DCO bias is enabled if DCO is enabled or DCO sources MCLK or SMCLK (SMCLKOFF = 0).
0	0	0	1	LPM0	CPU, MCLK are disabled. ACLK is active. SMCLK optionally active (SMCLKOFF = 0). DCO is enabled if sources ACLK or SMCLK (SMCLKOFF = 0). DCO bias is enabled if DCO is enabled or DCO sources MCLK or SMCLK (SMCLKOFF = 0).
0	1	0	1	LPM1	CPU, MCLK are disabled. ACLK is active. SMCLK optionally active (SMCLKOFF = 0). DCO is enabled if sources ACLK or SMCLK (SMCLKOFF = 0). DCO bias is enabled if DCO is enabled or DCO sources MCLK or SMCLK (SMCLKOFF = 0).
1	0	0	1	LPM2	CPU, MCLK are disabled. ACLK is active. SMCLK is disabled.
1	1	0	1	LPM3	CPU, MCLK are disabled. ACLK is active. SMCLK is disabled.
1	1	1	1	LPM4	CPU and all clocks are disabled.
1	1	1	1	LPM3.5	When PMMREGOFF = 1, regulator is disabled. No memory retention. In this mode, RTC operation is possible when configured properly. See the *RTC* module for further details.
1	1	1	1	LPM4.5	When PMMREGOFF = 1, regulator is disabled. No memory retention. In this mode, all clock sources are disabled; that is, no RTC operation is possible.

(1) This bit is automatically reset when exiting low-power modes. See Section 1.4.2 for details.

(2) The low-power modes and, hence, the system clocks can be affected by the clock request system. See the Clock System chapter for details.

Low Power Mode LPM3

So far, we have always used **active mode**

- MCU fully awake: CPU, all clocks, all modules active

Starting today we will use **LPM3**

- CPU, MCLK, SMCLK, DCO are disabled – ACLK still active

How? **Easy!**

```
; Enable  general interrupts and LPM3
            nop
            bis.w   #GIE|LPM3, SR       ; Enable general interrupts and LPM3
            nop
```

Execution of this line will immediately put the MCU into LPM3

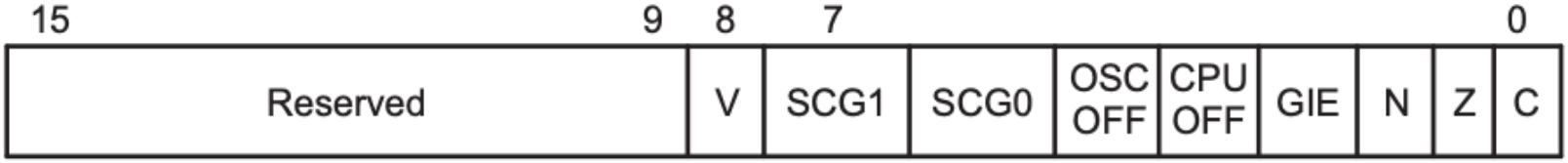
Subsequent instructions will not be executed

No need for an infinite loop!

Low Power Modes

Other low power modes are

```
#define LPM0                    (CPUOFF)
#define LPM1                    (SCG0+CPUOFF)
#define LPM2                    (SCG1+CPUOFF)
#define LPM3                    (SCG1+SCG0+CPUOFF)
#define LPM4                    (SCG1+SCG0+OSCOFF+CPUOFF)
```



Recall MSP430 interrupt handling

With an IRQ

- `SR` is pushed onto stack – current mode of operation is saved
- `SR` is cleared – MCU put into fully active mode

  CPU wakes up
- with `reti` the `SR` is restored from stack – mode of operation is restored

Timer B Interrupt: Blinky v. 3

Task: Make Timer B blink the red LED while the CPU is in LPM3

```
; Enable  general interrupts and LPM3
            nop
            bis.w   #GIE|LPM3, SR       ; Enable general interrupts and LPM3
            nop
```

- Download Lecture_26.asm from Carmen
- Timer B is in continuous mode and throws periodic interrupts

- **Interrupt flag is bit TBIFG in 16-bit register `TB0CTL` – zero not letter ”O”**

- **Timer_B_ISR toggles the red LED (Pin P1.0)**

- **Address for interrupt vector of Timer B is 0xFFF4**
  Find the section by locating the address 0xFFF4 in the linker file “lnk_msp430fr6989.cmd”