3 System Reset, Interrupts and Operating Modes

Topic		Page
3.1	System Reset & Initialization	3-3
3.2	Global Interrupt Structure	3-4
3.3	Interrupt Processing	3-8
3.4	Operating Modes	3-16
3.5	Low Power Modes	3-19
3.6	Basic Hints for Low Power Applications	3-21

3.1 System Reset & Initialization

The MSP430 has four possible reset sources: applying supply voltage to V_{CC} pin, a low input to the $\overline{\text{RST/NMI}}$ pin, a programmable watchdog timer time-out and a security key violation during write access to WDTCTL register.

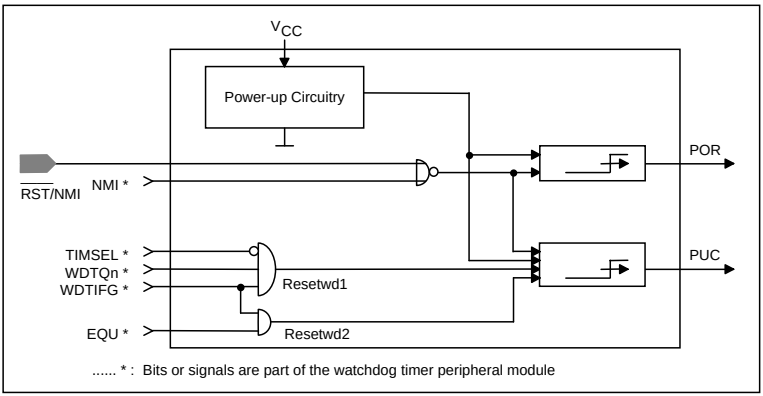


Figure 3.1: System Reset Functions

After the occurrence of a reset, the program can interrogate flags according to the reset source. The program can determine the source of reset in order to take appropriate action.

The MSP430 starts hardware initialization after applying V_{CC}:

- All I/O-pins are switched to the input direction
- The I/O-flags are cleared as described in the appropriate peripheral descriptions
- The address contained in the reset vector at word address 0FFFEh is placed into the Program Counter
  The CPU starts at the address contained in the power-up clear (PUC) vector.
- The status register SR is reset.
- All registers have to be initialized by the user's program (e.g., the Stack Pointer, the RAM, ...), except for PC and SR.
- Registers located in the peripherals are handled as described in the appropriate section.
- The frequency controlled system clock starts with the lowest frequency of the digital controlled oscillator. After the start of the crystal clock, the frequency is regulated to the target value.
  The $\overline{\text{RST/NMI}}$ pin is configured with the reset function after applying V_{CC}. It remains reset as long as the reset function is selected. When the pin is configured with the reset function, the MSP430 starts operation after the $\overline{\text{RST/NMI}}$ pin is pulled down to Gnd and released as follows:

- The address contained in the reset vector at word address 0FFFEh is placed into the Program Counter
- The CPU starts at the address contained in the reset vector after the release of the RST/NMI pin.
- The status register SR is reset.
- All registers have to be initialized by the user's program (e.g., the Stack Pointer, the RAM, ...), except for PC and SR.
- Registers located in the peripherals are handled as described in the appropriate section.
- The frequency controlled system clock starts with the lowest frequency of the DCO. After the start of the crystal clock the frequency is regulated to the target value.

3.2 Global Interrupt Structure

There are three types of interrupts:
- System reset
- Non-maskable interrupts
- Maskable interrupts

Sources causing a system reset are:
- Applying supply voltage @ POR, PUC
- 'low' on RST/NMI (if reset mode selected) @ POR, PUC
- Watchdog timer overflow (if watchdog mode selected) @ PUC
- Watchdog timer security key violation @ PUC
- (writing to WDTCTL with incorrect password)

A non-maskable interrupt can be generated by:
- Edge on RST/NMI-pin (if NMI mode selected)
- Oscillator fault

> **Note: Oscillator fault**
>
> The oscillator fault is maskable by an individual enable bit OFIE. It is not disabled during a general interrupt enable (GIE) reset.

Sources for maskable interrupts are:
- Watchdog timer overflow (if timer mode is selected)
- other modules having interrupt capability

MSP430 Interrupt Priority Scheme

The interrupt priority of the modules is defined by the arrangement of the modules in the connection chain: the nearer a module in the chain is towards the CPU/NMIRS, the higher is the priority.

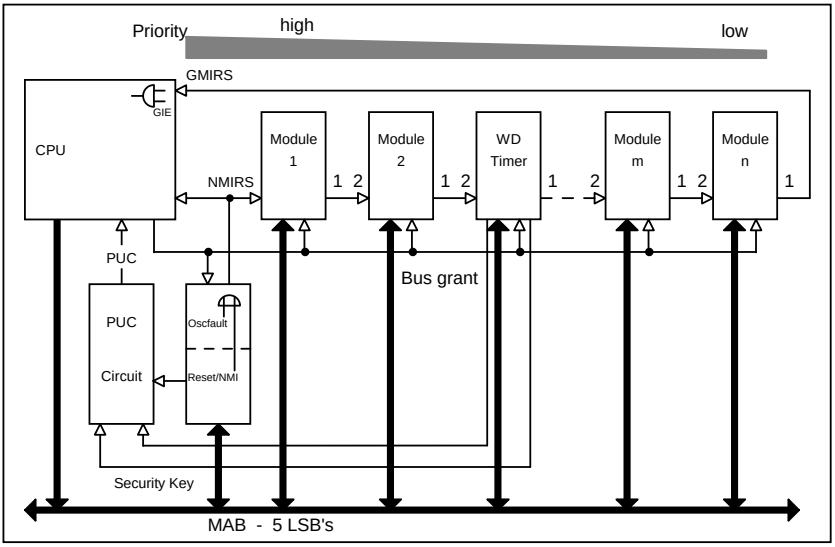


Figure 3.2: Interrupt Priority Scheme

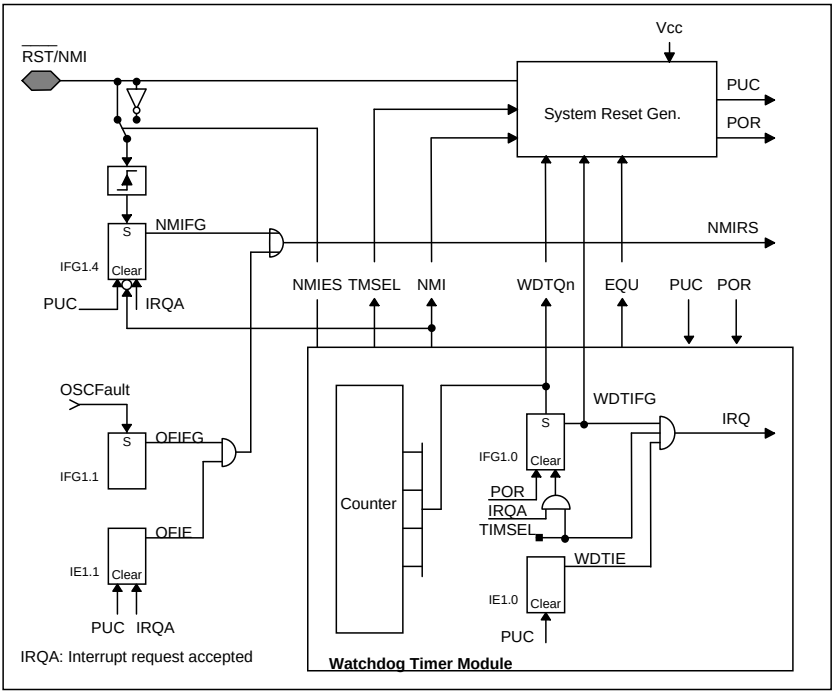


Figure 3.3: Reset/NMI-mode selection

Reset and NMI can be used only as alternatives, because they make use of the same input pin. The associated control bits are located in the Watchdog Timer Control register, and are also password protected.

WDTCTL
0120h

7							0
HOLD	**NMIES**	**NMI**	TMSEL	CNTCL	SSEL	IS1	IS0
rw-0	rw-0	rw-0	rw-0	(w)-0	rw-0	rw-0	rw-0

BIT 5: The NMI-Bit selects the function of the RST/NMI-input pin. It is cleared after PUC.

NMI = 0: The RST/NMI input works as Reset input.
As long as the RST/NMI-pin is held 'low', the internal PUC-signal is active (level sensitive).

NMI = 1: The ¯¯¯¯, RST/NMI input works as an edge-sensitive non-maskable
interrupt input.

BIT 6: This bit selects the activating edge of the ¯¯¯¯, RST/NMI input if NMI function is
selected. It is cleared after PUC.
NMIES = 0: A rising edge triggers a NMI-interrupt.
NMIES = 1: A falling edge triggers a NMI-interrupt.

Operation of global interrupt - Reset/NMI

If the Reset function is selected, the CPU is held in the reset state as long as the ¯¯¯¯, RST/NMI-pin is held 'low'. After the input has changed to high, the CPU starts program execution at the word address which is stored in word location 0FFFEh (Reset vector).

If the NMI function is selected, an edge according to the NMIES-bit generates an unconditional interrupt, and program execution is resumed at the address which is stored in location 0FFFCh. The ¯¯¯¯, RST/NMI flag in the SFR (IFG1.4) is also set. It is automatically reset during interrupt request service. The ¯¯¯¯, RST/NMI pin should never be held permanently 'low'. When a situation happens that activates the PUC, the consecutive reset of the bits in WDTCTL register forces the reset function on ¯¯¯¯, RST/NMI pin. An continuous 'low' at ¯¯¯¯, RST/NMI pin results in a permanent reset and system hold.

Note: NMI edge select

When NMI mode is selected and the NMI edge select bit is changed, an NMI can be generated, depending on the actual level at ¯¯¯¯, RST/NMI pin.

When the NMI edge select bit is changed before selecting the NMI mode no NMI is generated.

Operation of global interrupt - Oscillator fault control

As described in the oscillator section, the FLL oscillator will continue to work even if the crystal is defective, but it will then run at the lowest possible frequency. The second limit is the highest possible frequency. Both cases are usually error conditions and must be detectable by the CPU. Therefore the oscillator fault signal can be enabled by SFR bit IE1.1 to generate an NMI interrupt. By testing the interrupt flag IFG1.1 in the SFR, the CPU can determine if the interrupt was caused by an oscillator fault.

Operation of global interrupt - Power-up-clear (PUC)

Three sources or events can initiate system reset:
- Power-up logic
- ¯¯¯¯,RST/NMI input

- Watchdog overflow.

Resets caused by $\overline{\text{RST}}$/NMI and the watchdog can be evaluated by software through testing the associated interrupt flag in SFR bit IFG1.0.

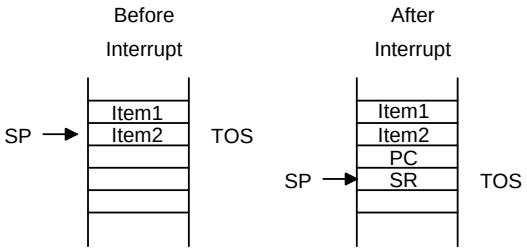
3.3 Interrupt Processing

The MSP430 programmable interrupt structure allows flexible on-chip and external interrupt configurations to meet real-time interrupt-driven system requirements. Interrupts may be initiated by the processor's operating conditions, such as watchdog overflow, peripheral modules or external events. Each interrupt source can be disabled individually by an interrupt enable bit or all interrupts are disabled by general interrupt enable bit GIE in the status register.

Whenever an interrupt is requested and the appropriate interrupt enable bit and the General Interrupt Enable Bit (GIE) is set, the interrupt service routine becomes active as follows:

- CPU active: The currently executed instruction is completed.
  CPU stopped: The low power modes are terminated.
- The Program Counter pointing to the next instruction is pushed onto the stack.
- The Status Register is pushed onto the stack.
- The interrupt with the highest priority is selected if multiple interrupts occurred during the last instruction and are pending for service.
- The appropriate interrupt requesting flag is reset automatically on single source flags. Multiple source flags remain set for servicing by software.
- The general interrupt enable bit GIE is reset;
  the CPUOff bit, the OscOff bit and the SCG1*) bit are cleared;
  the status bits V, N, Z and C are reset.
- The content of the appropriate interrupt vector is loaded into the Program Counter: The program continues with the interrupt handling routine at that address.

*) SCG0 is left unchanged, and FLL loop control remains in previous operating condition.



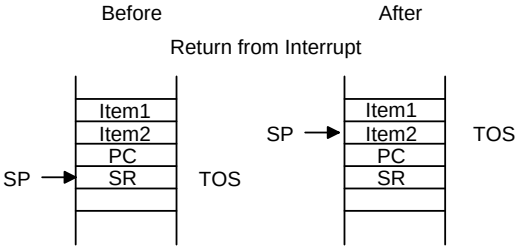
The interrupt latency is six cycles, starting with the acceptance of an interrupt request, and lasting until the start of execution of the first instruction of the appropriate interrupt service routine.

The interrupt handling routine terminates with the instruction:

RETI

which performs the following actions:

- The Status Register is popped from the stack.
  The interrupted software continues with exactly the same status as before the interrupt
  including OscOff, CPUOff and GIE bits.
  The GIE bit in the Status Register replaces the logical state present during interrupt service with the pushed state from TOS. It is set in any case, because it was set prior to accepting the interrupt.
- The Program Counter is popped from the stack.



The return from an interrupt service routine with the RETI instruction takes five cycles.
Interrupt nesting is activated if the GIE-bit is set inside the interrupt handling routine.
The general interrupt enable bit GIE is located in the Status Register SR/R2 which is included in the CPU as follows:

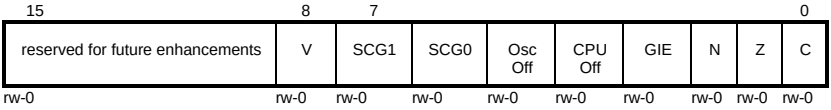


Figure 3.4: Status Register SR

Apart from the GIE bit, other sources of interrupt requests can be enabled/disabled individually or in groups. The interrupt enable flags are located together within two addresses of the special function register SFR. The program flow conditions on interrupt requests can be easily adjusted by extensive use of the interrupt enable masks. The hardware serves the highest priority within the empowered interrupt source.

3.3.1 Interrupt Control Bits in Special Function Registers SFRs

Most of the interrupt control bits, interrupt flags and interrupt enable bits are collected in SFRs under a few addresses. The Special Function Registers are located in the lower address range and are implemented in byte format. SFRs should be only accessed with byte instructions.

Address	7 0
000Fh	Not yet defined or implemented yet
000Eh	:
000Dh	:
000Ch	:
000Bh	:
000Ah	:
0009h	:
0008h	:
0007h	:
0006h	:
0005h	Module enable 2; ME2.x
0004h	Module enable 1; ME1.x
0003h	Interrupt flag reg. 2; IFG2.x
0002h	Interrupt flag reg. 1; IFG1.x
0001h	Interrupt enable 2; IE2.x
0000h	Interrupt enable 1; IE1.x

The various devices of the MSP430 Family support the SFRs with the correct logic and function within the individual modules. Each module interrupt source, except the non-maskable sources, can be individually enabled to access the interrupt function and the operation. Full software control of these configuration bits allows the application software to react to system requirements on interrupt enable mask.

Interrupt Enable 1 and 2

Bit position	Short form	Initial state*	Comment
IE1.0	WDTIE	reset	Watchdog Timer enable signal. Inactive if watchdog mode is selected.
IE1.1	OFIE	reset	Oscillator fault enable
IE1.2	P0IE.0	reset	Dedicated I/O P0.0
IE1.3	P0IE.1	reset	Dedicated I/O P0.1 or 8-bit Timer/Counter
IE1.4		reset	reserved, not defined yet
IE1.5		reset	reserved, not defined yet
IE1.6		reset	reserved, not defined yet
IE1.7		reset	reserved, not defined yet
IE2.0	URXIE	reset	USART receive enable
IE2.1	UTXRIE	reset	USART transmit enable
IE2.2	ADIE / TPIE	reset	ADC or Timer/Port enable signal (‘310 config.)

* Initial state is the logical state after PUC. For the WDTIFG see the appropriate comment.

Bit position	Short form	Initial state	Comment
IE2.3	TPIE	reset	Timer/Port (‘320,’330 config.)
IE2.4		reset	reserved, not defined yet
IE2.5		reset	reserved, not defined yet
IE2.6		reset	reserved, not defined yet
IE2.7	BTIE	reset	Basic Timer enable signal

3

Interrupt Flag Register 1 and 2

Bit position	Short form	Initial state	Comment
IFG1.0	WDTIFG	unchanged	Set on overflow or security key violation;
		or reset	Reset on VCC power-on or reset condition at ````, RST/NMI-pin
IFG1.1	OFIFG	set	Flag set on oscillator fault
IFG1.2	P0IFG.0	reset	Dedicated I/O P0.0
IFG1.3	P0IFG.1	reset	Dedicated I/O P0.1 or 8-bit Timer/Counter
IFG1.4	NMIIFG	reset	Signal at````, RST/NMI-pin
IFG1.5			reserved, not defined yet
IFG1.6			reserved, not defined yet
IFG1.7			reserved, not defined yet
IFG2.0	URXIFG		USART receive flag
IFG2.1	UTXIFG		USART transmitter ready
IFG2.2	ADIFG	reset	ADC, set on end-of-conversion
IFG2.3			reserved, not defined yet
IFG2.4			reserved, not defined yet
IFG2.5			reserved, not defined yet
IFG2.6			reserved, not defined yet
IFG2.7	BTIFG	unchanged	Basic Timer flag

Module enable 1and 2

Bit position	Short form	Initial state	Comment
ME1.0			reserved, not defined yet
ME1.1			reserved, not defined yet
ME1.2			reserved, not defined yet
ME1.3			reserved, not defined yet
ME1.4			reserved, not defined yet
ME1.5			reserved, not defined yet
ME1.6			reserved, not defined yet
ME1.7			reserved, not defined yet
ME2.0	URXE		USART receiver enable
ME2.1	UTXE		USART transmit enable
ME2.2			reserved, not defined yet
ME2.3			reserved, not defined yet
ME2.4			reserved, not defined yet
ME2.5			reserved, not defined yet
ME2.6			reserved, not defined yet
ME2.7			reserved, not defined yet

Interrupt Vector Addresses

The interrupt vectors and the power-up starting address are located in the ROM, using address range 0FFFFh - 0FFE0h. The vector contains the 16-bit address of the appropriate interrupt handler instruction sequence. The interrupt vectors are shown in decreasing priority order of priority:

Interrupt source	Interrupt flag	System Interrupt	Word Address	Priority
Power-up ext. Reset Watchdog	WDTIFG	Reset	0FFFEh	15, highest
NMI OSC. fault	NMIIFG OFIFG *	non-maskable (non-)maskable	0FFFCh	14
Dedicated I/O	P0IFG.0	maskable	0FFFAh	13
Dedicated I/O	P0IFG.1	maskable	0FFF8h	12
		maskable	0FFF6h	11
Watchdog timer	WDTIFG	maskable	0FFF4h	10
Timer_A	CCIFG0	maskable	0FFF2h	9
Timer_A	TAIFG **	maskable	0FFF0h	8
USART Receive	URXIFG	maskable	0FFEEh	7
USART Transmit	UTXIFG	maskable	0FFECh	6
ADC, Timer/Port [2]	ADCIFG	maskable	0FFEAh	5
Timer/Port [1]		maskable	0FFE8h	4
Port P2	P2IFG.07 *, **	maskable	0FFE6h	3
Port P1	P1IFG.07 *, **	maskable	0FFE4h	2
Basic Timer	BTIFG	maskable	0FFE2h	1
Port P0	P0IFG.27 *, **	maskable	0FFE0h	0, lowest

*) multiple source flags
**) Preliminary definition
1) Timer/Port vector in ‘320 and ‘330 configuration
2) Timer/Port vector in ‘310 configuration

Table 3.1: Interrupt sources, flags and vectors

3.3.2 External Interrupts

All eight bits of the entire ports P0, P1 and P2 are implemented for interrupt processing of external events. All individual I/O bits are programmable independently.

Any combinations of inputs, outputs and interrupt conditions are possible. This allows an easy adaptation to different I/O configurations.

Note: Minimum pulse width of external interrupt signals

All external interrupt signals should have a minimum pulse width of 1.5 MCLK to ensure stable interrupt acknowledgement, but shorter signals may also request an interrupt service

Port P0
Three separate vectors are allocated to the port P0 module. The signals on P0.0, P0.1 and the remaining port signals P0.2 to P0.7 are used as the three interrupt vector sources. The vector contained in the corresponding memory location is loaded into the Program Counter by an interrupt even.

The port P0 has 6 registers used for the control of the I/O-pins

- Input Register
- Output Register
- Direction Register
- Interrupt Flags: if — This register contains six flags, which contain information the I/O-pins are used as interrupt inputs:
  Bit = 0: No interrupt is pending
  Bit = 1: An interrupt is pending, due to a transition at the I/O-pin.
  Writing a zero to an Interrupt Flag resets it.
  Writing a one to an Interrupt Flag sets it. Device operation continues just the same way as if an interrupt event had occurred.
- Interrupt Edge Select: — This register contains a bit for each I/O-pin that selects which transition triggers the interrupt flag.
  Bit = 0: The interrupt flag is set with LO/HI transition
  Bit = 1: The interrupt flag is set with HI/LO transition
- Interrupt Enable: — This register contains six bits for the I/O-pins P0.2 to P0.7, to enable interrupt request on an interrupt event.
  Bit = 0: The interrupt request is disabled
  Bit = 1: The interrupt request is enabled

I/O-PIN interrupt handler for P0.2 to P0.7: Programming Example

```
; The I/O-PIN interrupt handler for P0.2 to P0.7starts here
;
IOINTR      PUSH        R5              ; Save R5
            MOV.B       &P0IFG,R5       ; Read interrupt flags
            BIC.B       R5,&P0IFG       ; Clear status flags with the read
                                        ; data
                                        ; Additional set bits are not cleared!
            EINT                        ; Allow interrupt nesting
;
; R5 contains information which I/O-pin(s) caused interrupt:
; the processing starts here.
;
            .........
            .........
            POP         R5              ; JOB done: restore R5
            RETI                        ; Return from interrupt
            .........
            .........
; Definition of interrupt vector table
            .sect       "IO27_vec",0FFE0h
            .WORD       IOINTR          ; I/O-Pin (2 To 7) Vector In ROM
            ;
            .sect       "RST_vec",0FFFEh ; Interrupt Vectors
            .WORD       RESET
```

Port P1, Port P2

The ports P1 and P2 are identical. A separate vector is allocated to the port P1 and port P2 module. The pins P1.0 to P0.7 and P2.0 to P2.7 are used as the interrupt sources. The vector contained in the corresponding memory location is loaded into the Program Counter by an interrupt event.

Each port P1 and P2 has 7 registers used for the control of the I/O-pins

- Input Register
- Output Register
- Direction Register
- Interrupt Flags: This register contains eight flags that contain information if the I/O-pins are used as interrupt inputs:
  Bit = 0: No interrupt is pending
  Bit = 1: An interrupt is pending due to a transition at the I/O-pin.
  Writing a zero to an Interrupt Flag resets it.
  Writing a one to an Interrupt Flag sets it. Device operation continues just the same way as if an interrupt event had occurred.

- Interrupt Edge Select: This register contains a bit for each I/O-pin that selects which transition triggers the interrupt flag.
  Bit = 0: The interrupt flag is set with LO/HI transition
  Bit = 1: The interrupt flag is set with HI/LO transition
- Interrupt Enable: This register contains eight bits for the I/O-pins P0.2 to P0.7 to enable interrupt request on an interrupt event.
  Bit = 0: The interrupt request is disabled
  Bit = 1: The interrupt request is enabled
- Function Select Register.

Note: How the interrupts on digital ports P0, P1 and P2 are handled

Only transitions (not static levels) cause interrupts.

The interrupt routine must reset the multiple used Interrupt Flags. Multiple interrupt flags are P0IFG.2 to P0IFG.7, P1IFG.0 to P0IFG.7 and P2IFG.0 to P0IFG.7. The single source flags P0IFG.0 and P0IFG.1 are reset when they are serviced.

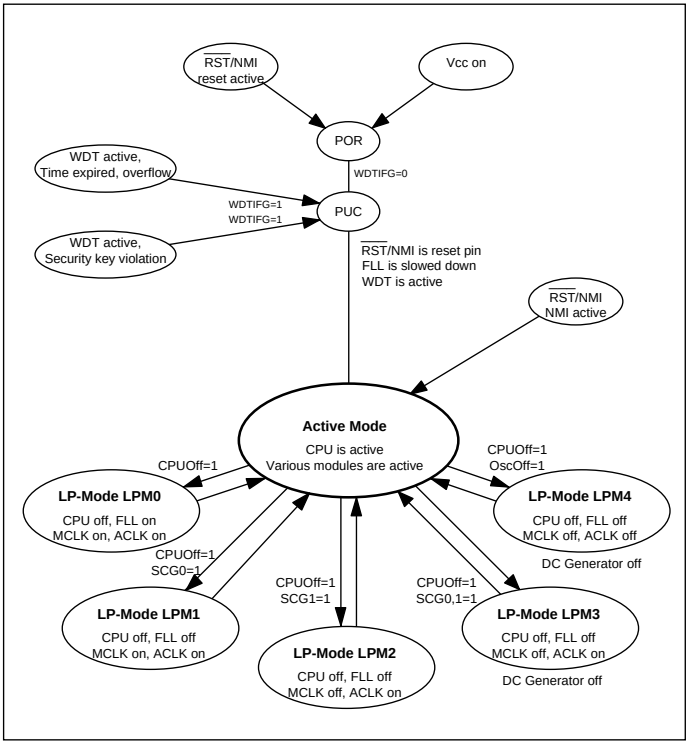
If an Interrupt Flag is still set (because the transition occurred during the interrupt routine) when the *RETI* instruction is executed, an interrupt occurs again after the *RETI* is completed. This ensures that each transition is seen by the software.

3.4 Operating Modes

The MSP430 operating modes support various requirements for ultra-low power and ultra-low energy consumption in an advanced manner. This is combined with an intelligent management of operations during the different module and CPU states. An interrupt event awakes the system from each of the various operating modes and the RETI instruction returns operation to the mode that was selected before the interrupt event.

The MSP430 Family has been developed for ultra-low power applications and uses different levels of operating modes.

Ultra-low power system design in CMOS technology takes account of three primary intentions:
- the desire for speed and data throughput conflicts with a design for ultra-low power
- minimize individual current consumption
- limit activity state to the minimum required.

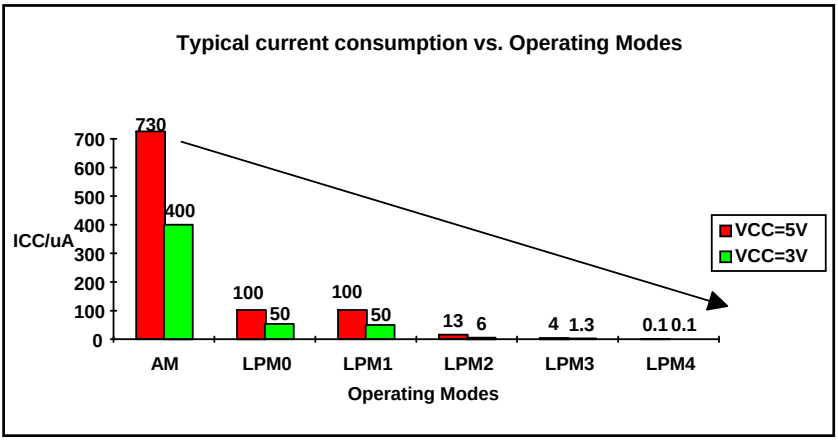


There are five operating modes which the software can configure:

- **Active Mode AM,**
with different combinations of active peripheral modules
- **Low Power Mode 0 LPM0,**
with CPUOff bit set, the CPU is disabled,
peripheral's operation is not halted by CPUOff,
ACLK and MCLK signal are active. Loop control for MCLK is active.
@ SCG1=0, SCG0=0, OSCOff=0, CPUOff=1
- **Low Power Mode 1 LPM1,**
with CPUOff bit set, the CPU is disabled,
peripheral's operation is not halted by CPUOff,
loop control (frequency-lock-loop) for MCLK is inactive,
ACLK and MCLK signal are active.
@ SCG1=0, SCG0=1, OSCOff=0, CPUOff=1
- **Low Power Mode 2 LPM2,**
with CPUOff bit set, the CPU is disabled,
peripheral's operation is not halted by CPUOff,

loop control for MCLK signal is inactive,
ACLK signal is active.
@ SCG1=1, SCG0=0, OSCOff=0, CPUOff=1

- **Low Power Mode 3 LPM3,**
  with CPUOff bit set, the CPU is disabled,
  peripheral's operation is not halted by CPUOff,
  Loop control for MCLK and MCLK signal are inactive,
  DC generator of the DCO (-> MCLK generator) is switched off.
  ACLK signal is active.
  @ SCG1=1, SCG0=1, OSCOff=0, CPUOff=1
- **Low Power Mode 4 LPM4,**
  with CPUOff bit set, the CPU is disabled,
  peripheral's operation is not halted by CPUOff,
  loop control for MCLK signal is inactive,
  DC generator of the DCO (-> MCLK generator) is switched off,
  ACLK signal is inactive; the crystal oscillator is stopped.
  @ SCG1=X, SCG0=X, OSCOff=1, CPUOff=1



Source: TI Data sheet SLASE07, January 1996 (MSP430C312/314)

The activity state of individual peripheral modules and the CPU can be controlled using the appropriate low power mode, and various options to stop operation of parts of peripheral modules, or to stop them completely. There are different ways to configurethe lowest potential current consumption, using the software on an application-specific basis. The special function registers include module enable bits that stop or enable the operational function of the specific peripheral module. All registers of the peripherals may be accessed even during disable mode. Other current saving functions can be implemented into peripherals that are accessed via the state of the register bits. An example is the enables/disable of the analog voltage generator in the LCD peripheral: this is turned on or off via one register bit. The most general bits that influence the

current consumption and support fast turn-on from low power operating modes are located in the status register SR. There are four bits that control the CPU and the system clock generator.

These four bits are very useful to support the request for discontinuous active mode AM, and to limit the time period of the full operating mode. The four bits are CPUOff, OscOff, SCG0 and SCG1. The major advantage of including the operating mode bits into the status register is that the present state of the operating condition is saved onto stack during an interrupt request service. As long as the stored status register information is not altered, the processor continues (after *RETI*) with the same operating mode as before the interrupt event. Another program flow may be selected by manipulation of the data stored on the stack or the stack pointer. The easy access of the stack and stack pointer with the instruction set allows individually optimized program structures.

3.5 Low Power Modes

The module enable bits in the SFRs enable the configuration of individual power consuming controller operation states. The users program defines the state of the peripheral modules to be active or inactive. The current consumption of disabled modules is decreased by the leakage current of all parts that can be disabled. The only active parts of a module are those which are mandatory to get it to the enable state or to pass interrupt requests to the CPU (e.g. external hardware interrupt).

In addition to the individual enable options, there are five more current saving modes possible: the CPU off mode (LPM0), and four operating configurations of the system clock generator. They are entered if one or more of the bits CPUOff, SCG1, SCG0, OscOff - located in the Status Register - are set. The reaction of the system clock generator module on the status of the bits SCG1, SCG0 and OscOff with its four low power modes are described in detail in the system clock generation section.

Enter interrupt routine
The interrupt routine is entered and processed if an enabled interrupt wakes-up the MSP430:
- The SR and PC are stored onto the stack, with the content present at the interrupt event.
- Subsequently the operation mode control bits OscOff, SCG1 and CPUOff are cleared automatically in the Status Register.

Return from interrupt
Two different ways back from interrupt service routine to continue flow of operation are practicable:

- Return with set low power mode bits
  When returning from the interrupt, the program counter points to the next instruction. The instruction pointed to is not executed, since the restored low power mode stops CPU activity.
- Return with reset low power mode bits
  When returning from the interrupt, the program continues at the address following the instruction which set the OscOff or CPUOff-bit in the Status Register.

3.5.1 Low Power Mode 0 and 1, LPM0 and LPM1

Low power mode 0 or mode 1 is selected if the appropriate bit CPUOff in the status register is set. Immediately after the bit is set the CPU stops operation, and the normal operation of the system core is stopped. The operation of the CPU is halted until any interrupt request or reset is effective. All internal bus activities are stopped. The system clock generator continues operation, and the clock signals MCLK and ACLK are active depending on the state of the other three bits, SCG0, SCG1 and OscOff in the status register. The SCG1 bit defines if the MCLK is controlled to be N*ACLK, or to run with the latest DCO control signals.

Those peripherals are active which are enabled and clocked with the MCLK or ACLK signal. All pins of I/O ports and the RAM/registers are unchanged. Wake-up is possible by all enabled interrupts.

```
; === Main program flow with switch to CPUOff Mode ========================
;
                BIS   #18h,SR   ; Enter LPM0 + enable general interrupt GIE.
                                ; The PC is incremented during execution of this in-
                                ; struction and points to the consecutive program step.
                .....           ; The program continues here if CPUOff bit is reset
                                ; during the interrupt service routine

; === Interrupt service routine ============================================
                .....
                .....
                RETI            ; RETI restores the same state of CPU before
                interrupt.
                                ; This is possible because control registers GIE,
                                ; CPUOff, OscOff, SGC1 and SCG0 are located in the
                                ; status register which is restored during execution of
                                ; return-from-interrupt.
```

3.5.2 Low Power Mode 2 and 3, LPM2 and LPM3

Low power mode 2 or mode 3 is selected if the appropriate bit CPUOff and SCG1 bit in the status register are set. Immediately after the bits are set, the CPU and MCLK are halted. The CPU and MCLK are halted until any interrupt request or reset is effective. All internal bus activities are stopped. The SCG1 bit defines if the MCLK is controlled to be N*ACLK or to run with the latest DCO control signals when the sytem returns to active mode.

Those peripherals are active that are enabled and clocked with the ACLK signal. Peripherals that are operating with the MCLK signal are inactive, because the MCLK signal is inactive. All pins of I/O ports and the RAM/registers are unchanged. Wake-up is possible by those enabled interrupts coming from system clock (MCLK) independent sources.

3.5.3 Low Power Mode 4, LPM4

All activities cease; only the RAM contents, Port and registers are maintained. Wake-up is only possible by enabled external interrupts.

Before activating LPM4, the software flow should consider the conditions that are applied to the system during the period of this low power mode. The two and most important figures that should be looked at are the environmental situation, with the influence at the DCO and the clocked operation conditions. The environmental situation defines whether the actual value of the frequency integrator should be held or corrected. A correction can be intended when ambient conditions would increase the system frequency drastically. When clocked operation is applied, it should be considered that the loop can lose control over the frequency if there remaining time slot is insufficient to hold the closed loop in the correct operating range.

The following example shows the entering of the low power mode 4 (OscOff):

```
BIS   #B8h,SR   ; Enter LPM4 + enable general interrupt GIE.
                ; The CPU must be switched of with LPMs.
                ; Additionally the DCO operation is enabled.
                ; When during the interrupt routine the LPM4 is going
                ; to be disrupted, DCO operation is prepared.
.....           ; The program continues here if OscOff bit is reset
.....           ; during the interrupt service routine.
.....           ; Otherwise it retains in OscOff mode
```

3.6 Basic Hints for Low Power Applications

There are some general basics principles which should be considered when the current consumption is a critical part of a system application:

- Tie unused FETI input to V_{SS}
- Switch-off the Analog Generator in the LCD+ module or an external one if convenient
- Do not tie the JTAG inputs TMS, TCK and TDI to V_{SS}
- Any CMOS input should have no floating node: tie all inputs to an appropriate voltage level
- Select the lowest possible operating frequency - for the core and for the individual peripheral module
- Select the weakest drive capability if an LCD is used, or switch it off
- Utilize the feature of interrupt driven software - the program starts execution rapidly.