MSP430 Advanced Technical Conference 2006

Designing for Ultra-Low Power with MSP430

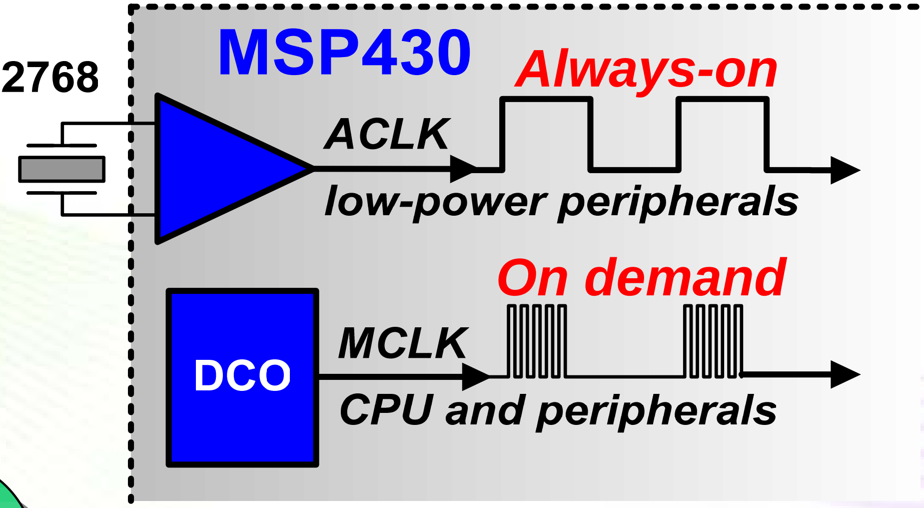
Christian Hernitscheck
MSP430 FAE Europe
Texas Instruments

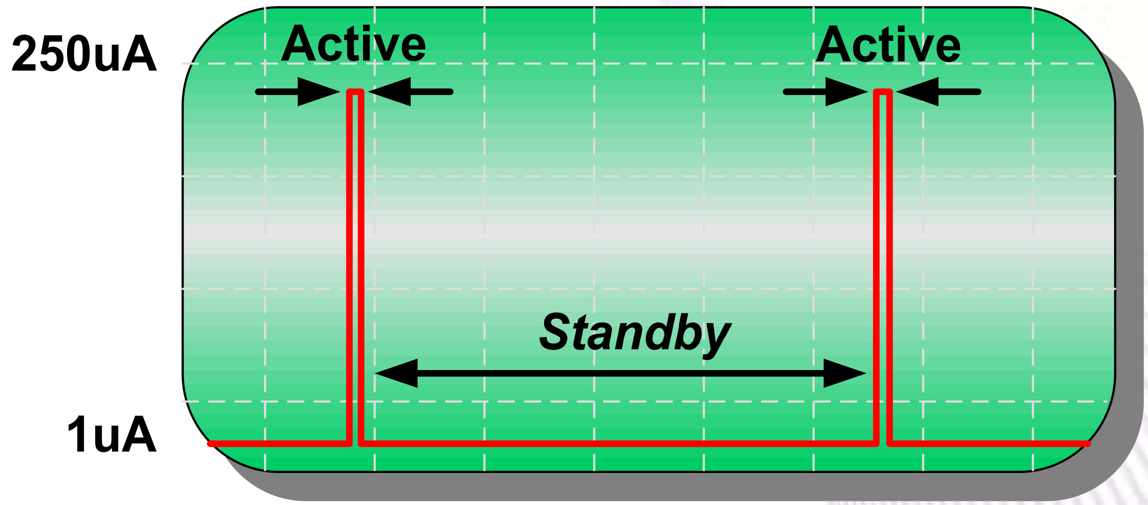
Agenda

- Introduction to Ultra-Low Power
- Looking for Ultra-Low Power Parts
- MSP430 – The Ultra-Low Power MCU
- Low-Power Efficient Coding Techniques
- Summary

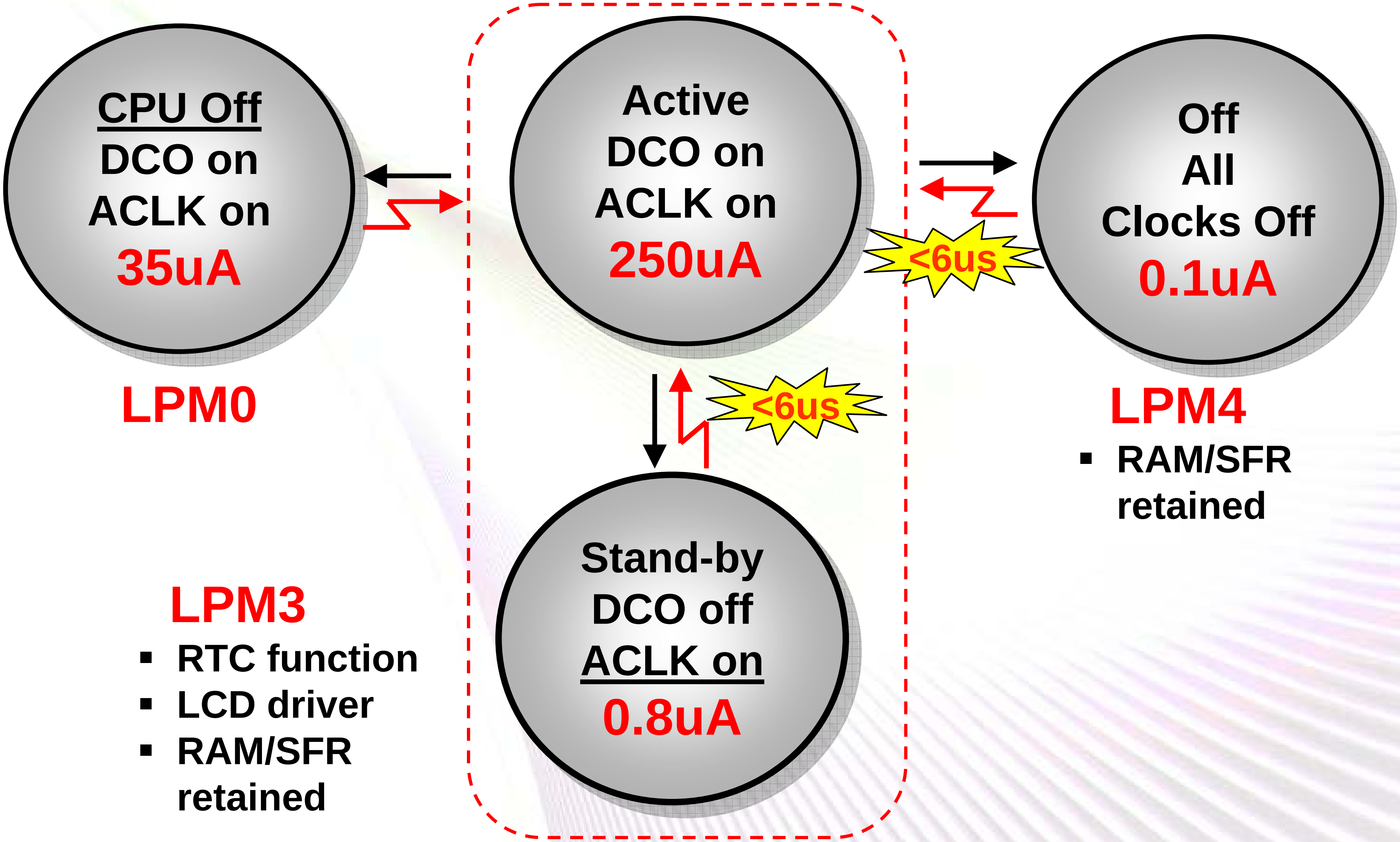
Achieving *Ultra-low Power*

- **Extended Ultra-low Power standby mode**
- **Minimum active duty cycle**
- **Performance on-demand**





Ultra-low Power Clock Control



Agenda

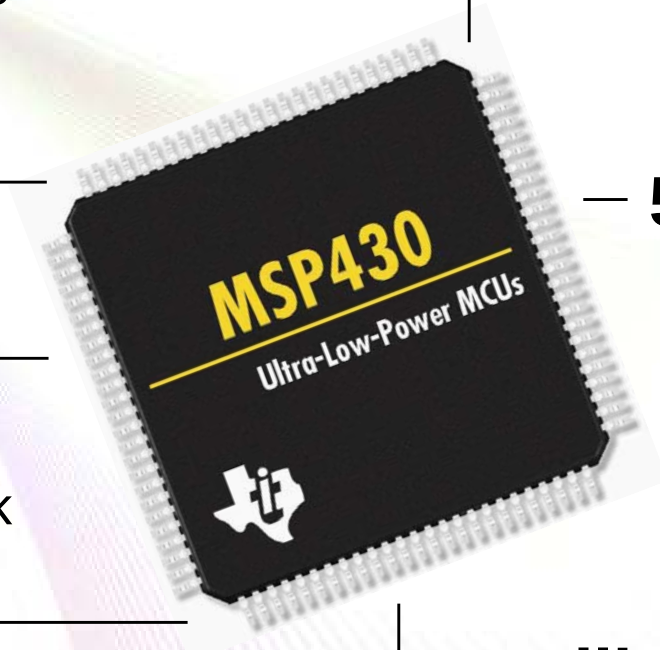
- Introduction to Ultra-Low Power
- Looking for Ultra-Low Power Parts
- MSP430 – The Ultra-Low Power MCU
- Low-Power Efficient Coding Techniques
- Summary

Ultra-Low Power Architecture

Multiple operating modes

- 0.1uA power down
- 0.7uA standby mode
- 250uA / 1MIPS

Zero-power BOR



50nA pin leakage

Modern CPU

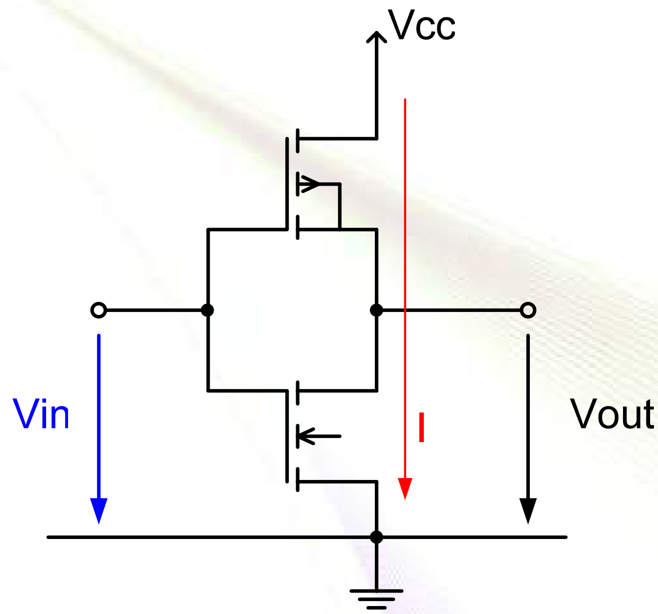
Minimum cycles per task

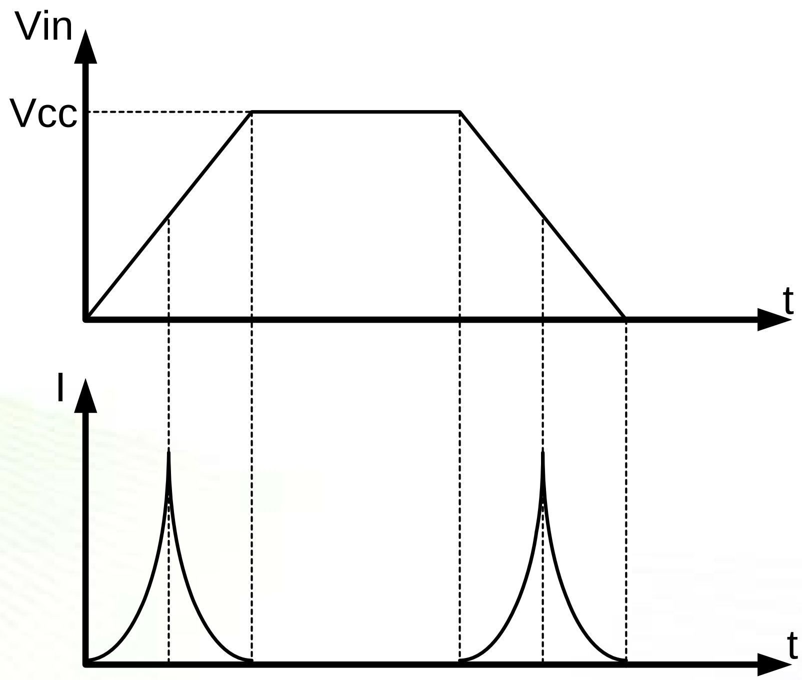
Intelligent peripherals

Instant-on stable high-speed clock

Power Consumption in CMOS Designs

- **CMOS Inverter:**





- **Power Consumption of a CMOS Inverter:**

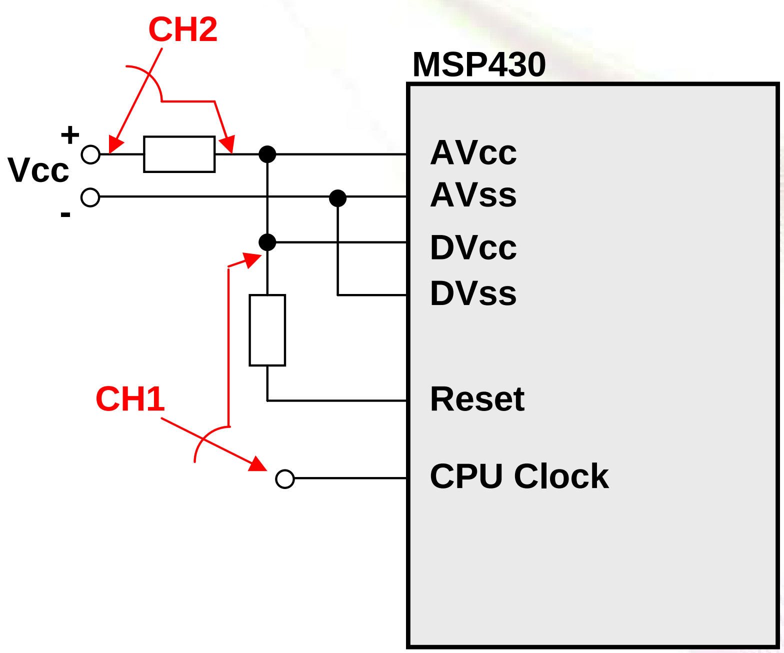
$$P = P_{stat} + P_Q + P_{dyn}$$

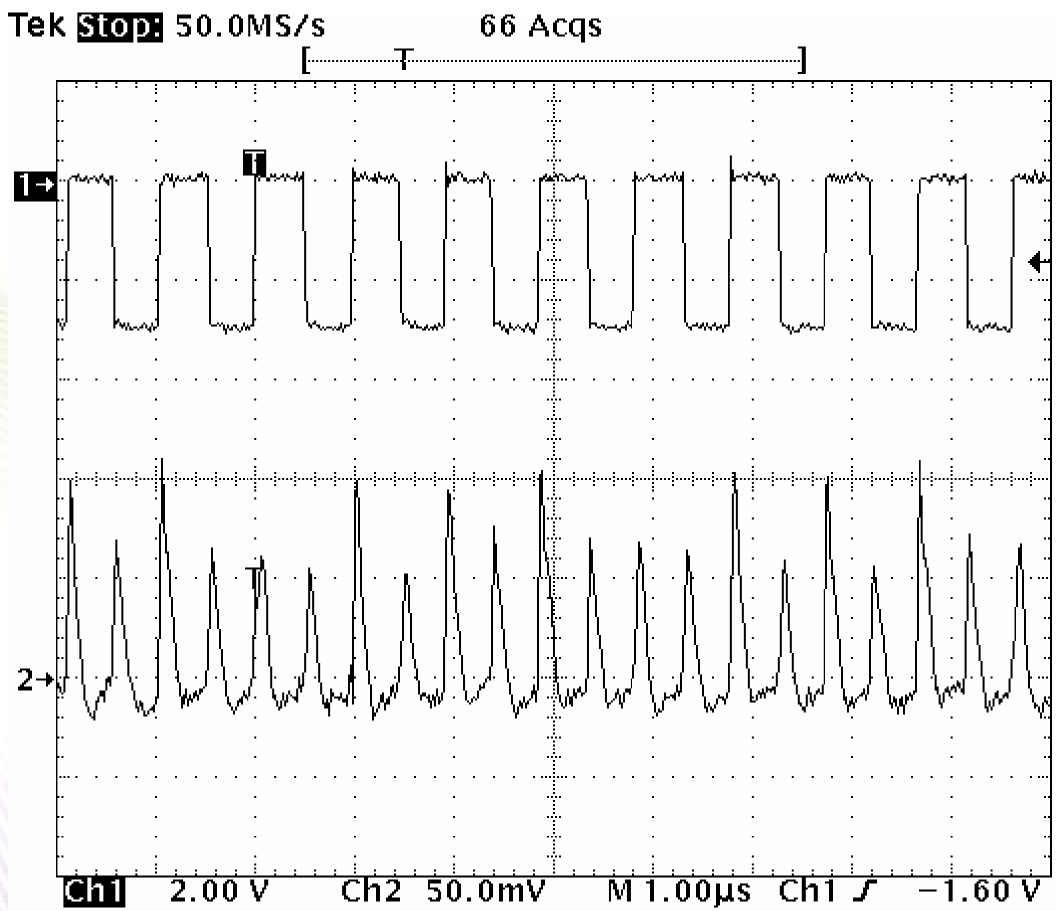
$$P_{stat} = Vcc * I_{LL}$$

$$P_Q = \beta / 12 * (Vcc - 2*U_{Tn})^3 * \tau / T$$

$$P_{dyn} = C_L * f * Vcc^2$$

MCU's Digital Supply Current





MSP430 Active Mode Supply Current

- **MSP430F2131 data sheet [slas439a]:**

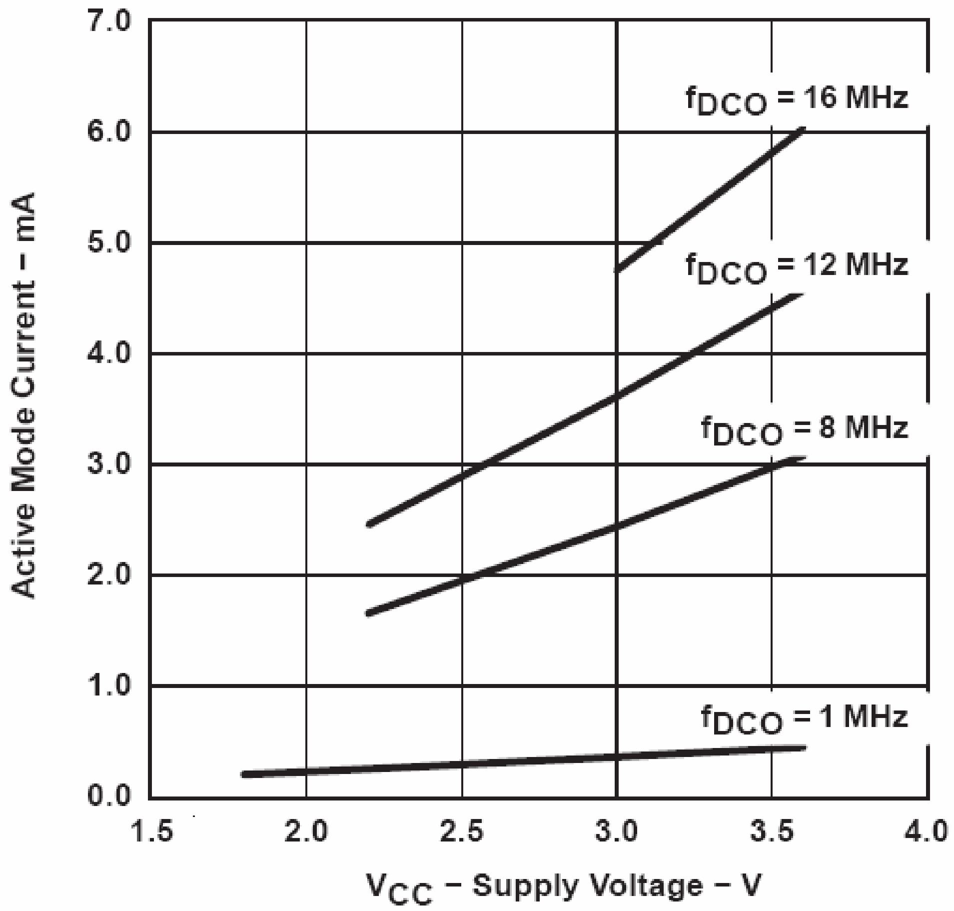


Figure 2. Active mode current vs V_{CC}, $T_A = 25°C$

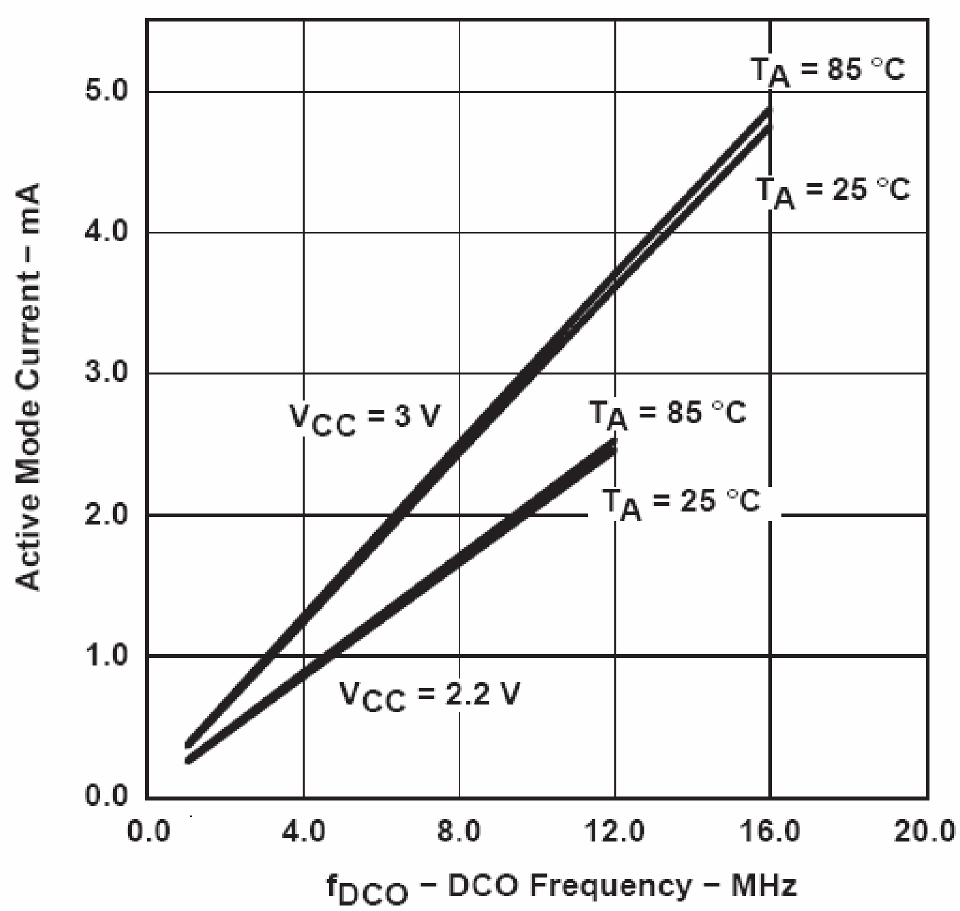


Figure 3. Active mode current vs DCO frequency

Device, Voltage, Temperature & Clock

MSP430F16x LPM3

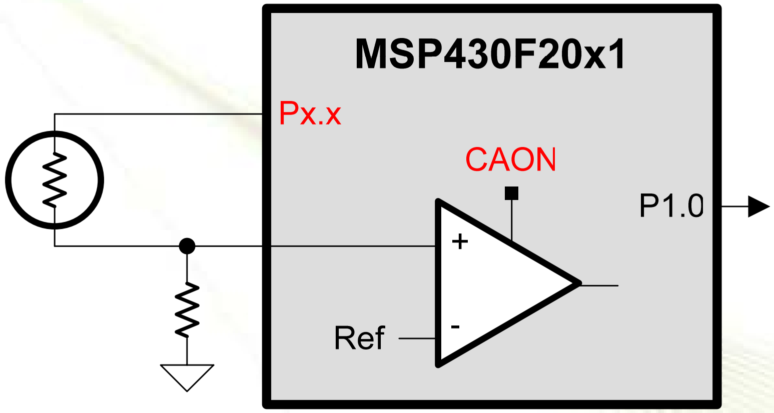
T_A = −40°C	V_{CC} = 2.2 V	1.1	1.6	µA
T_A = 25°C		1.1	1.6	
T_A = 85°C		2.2	3.0	
T_A = −40°C	V_{CC} = 3 V	2.2	2.8	
T_A = 25°C		2.0	2.6	
T_A = 85°C		3.0	4.3	

MSP430F20xx LPM3

LFXT1	T_A = −40°C	2.2 V	0.7	1.2	µA
	T_A = 25°C		0.7	1.0	
	T_A = 85°C		1.4	2.3	
	T_A = −40°C	3 V	0.9	1.2	
	T_A = 25°C		0.9	1.2	
	T_A = 85°C		1.6	2.8	
VLO	T_A = −40°C	2.2 V	0.4	0.7	µA
	T_A = 25°C		0.5	0.7	
	T_A = 85°C		1.0	1.6	
	T_A = −40°C	3 V	0.5	0.9	µA
	T_A = 25°C		0.6	0.9	
	T_A = 85°C		1.3	1.8	

- **Die size and # pins**
- **Family architectures and clock system**

Power Manage Internal Peripherals



Comparator_A

VCC	MIN	TYP	MAX	UNIT
2.2 V		25	40	µA
3 V		45	60	

```
P1OUT |= 0x02;                              // Power divider
CACTL1 = CARSEL + CAREF_2 + CAON;  // Comp_A on
if (CAOUT & CACTL2)
 P1OUT |= 0x01;                             // Fault
else
  P1OUT &= ~0x01;
P1OUT &= ~0x02;                             // de-power divider
CACTL1 = 0;                                 // Disable Comp_A
```

Integrated Analog Power Managing

ADC10

I_{ADC10}	Operating supply current into V_{CC} terminal (see Note 3)	$f_{ADC10CLK}$ = 5.0 MHz ADC10ON = 1, REFON = 0 ADC10SHT0=1, ADC10SHT1=0, ADC10DIV=0	V_{CC} = 2.2 V		0.52	1.05	mA
			V_{CC} = 3 V		0.6	1.2	
I_{REF+}	Reference operating supply current, reference buffer disabled (see Note 4)	$f_{ADC10CLK}$ = 5.0 MHz ADC10ON = 0, REFON = 1, REF2_5V = x; REFOUT = 0	V_{CC} = 2.2V/3 V		0.25	0.4	mA

DAC12

DAC12AMPx=2, DAC12IR=1, DAC12_xDAT=0800h , $V_{eREF+}=V_{REF+}= AV_{CC}$	2.2V/3V		50	110	µA
DAC12AMPx=5, DAC12IR=1, DAC12_xDAT=0800h, $V_{eREF+}=V_{REF+}= AV_{CC}$	2.2V/3V		200	440	
DAC12AMPx=7, DAC12IR=1, DAC12_xDAT=0800h, $V_{eREF+}=V_{REF+}= AV_{CC}$	2.2V/3V		700	1500	

OA

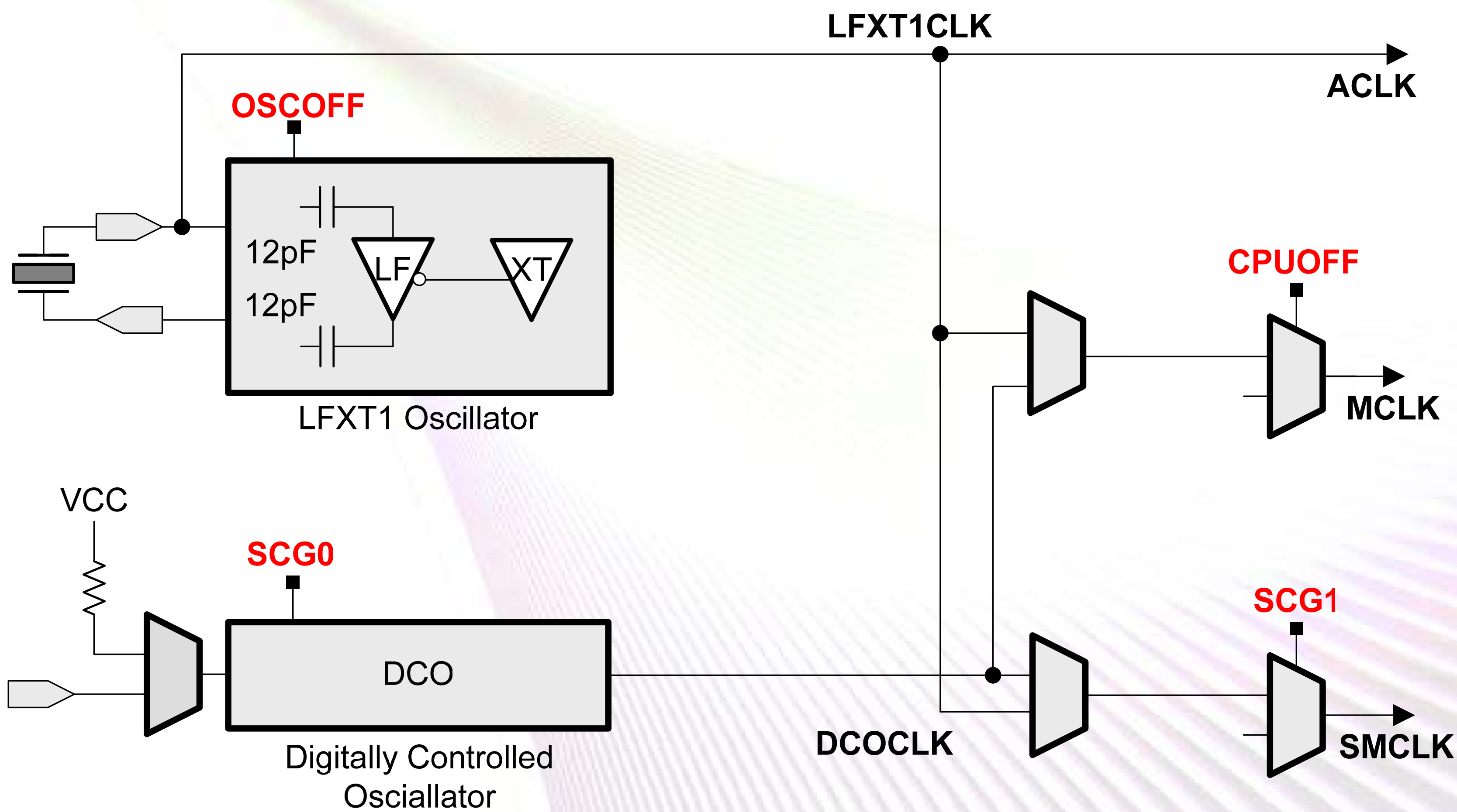
Fast Mode, RRIP OFF	2.2 V/3 V	180	290	
Medium Mode, RRIP OFF	2.2 V/3 V	110	190	
Slow Mode, RRIP OFF	2.2 V/3 V	50	80	µA
Fast Mode, RRIP ON	2.2 V/3 V	300	490	
Medium Mode, RRIP ON	2.2 V/3 V	190	350	
Slow Mode, RRIP ON	2.2 V/3 V	90	190	

Agenda

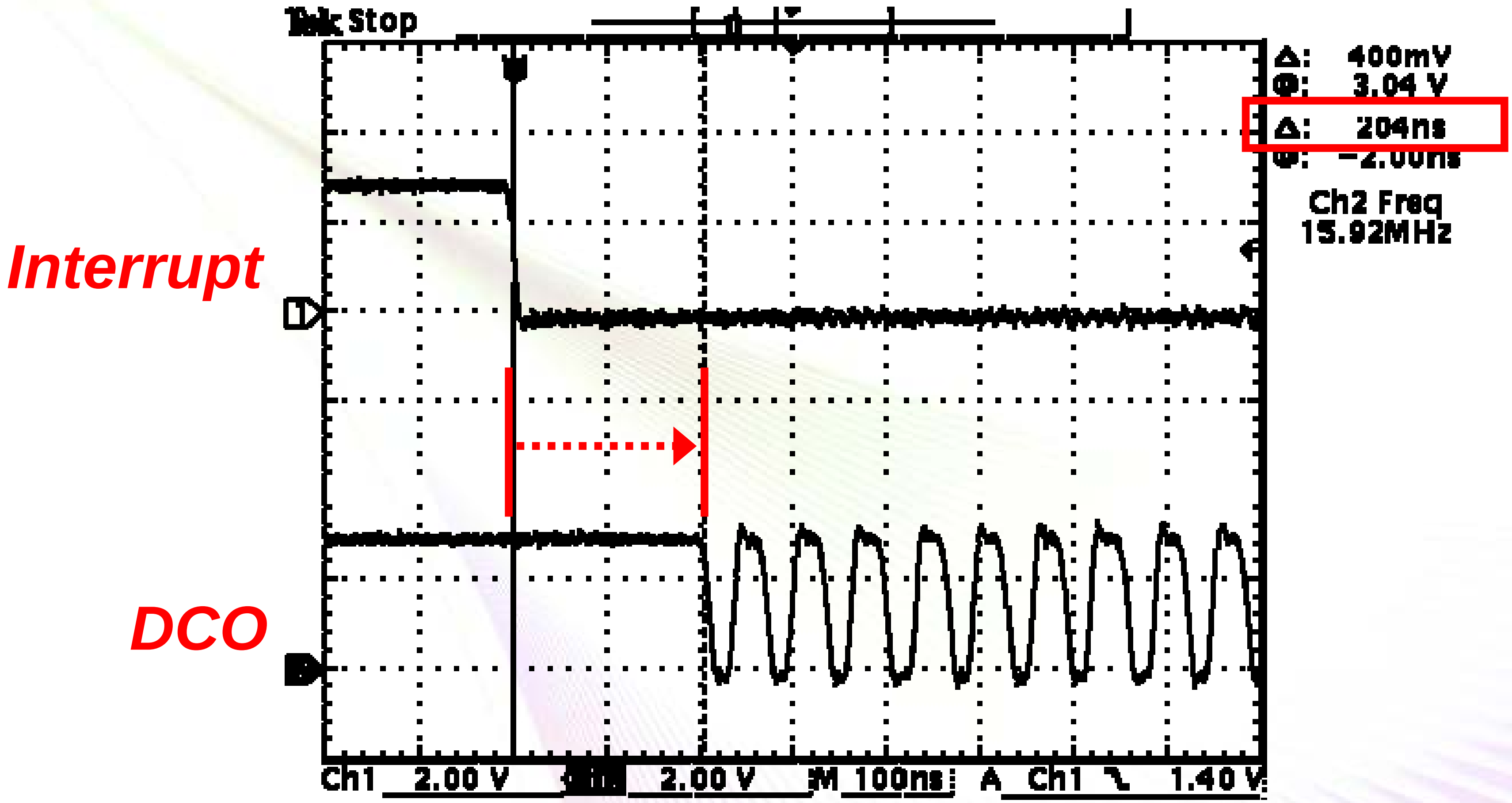
- Introduction to Ultra-Low Power
- Looking for Ultra-Low Power Parts
- MSP430 – The Ultra-Low Power MCU
- Low-Power Efficient Coding Techniques
- Summary

MSP430x11x/12x Basic Clock

R2/SR:	Reserved	V	SCG1	SCG0	OSC OFF	CPU OFF	GIE	N	Z	C



Performance on Demand



- **Immediate-stable clock start for reaction to events**

Low Power Mode Configuration

R2/SR:

Reserved	V	SCG1	SCG0	OSC OFF	CPU OFF	GIE	N	Z	C

	SCG1	SCG0	OSC OFF	CPU OFF	
Active Mode	0	0	0	0	~ 250uA
LPM0	0	0	0	1	~ 35uA
LPM3	1	1	0	1	~ 0.8uA
LPM4	1	1	1	1	~ 0.1uA

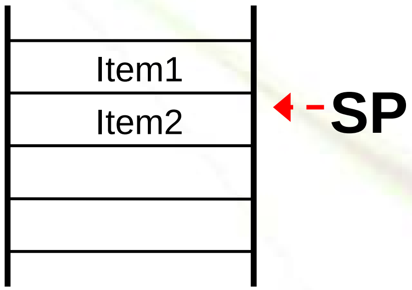
- **Assembler Code Example:**

```
bis.w    #CPUOFF,SR      ; LPM0
```

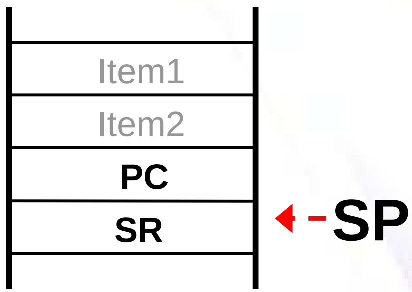
- **C Code Example:**

```
_BIS_SR (CPUOFF);        // LPM0
```

Interrupt Processing

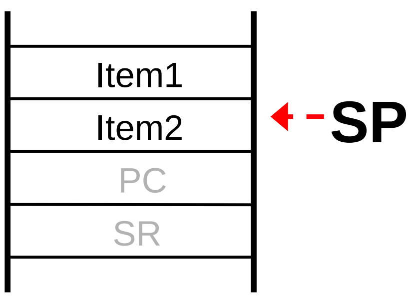


Prior to ISR



ISR hardware - automatically

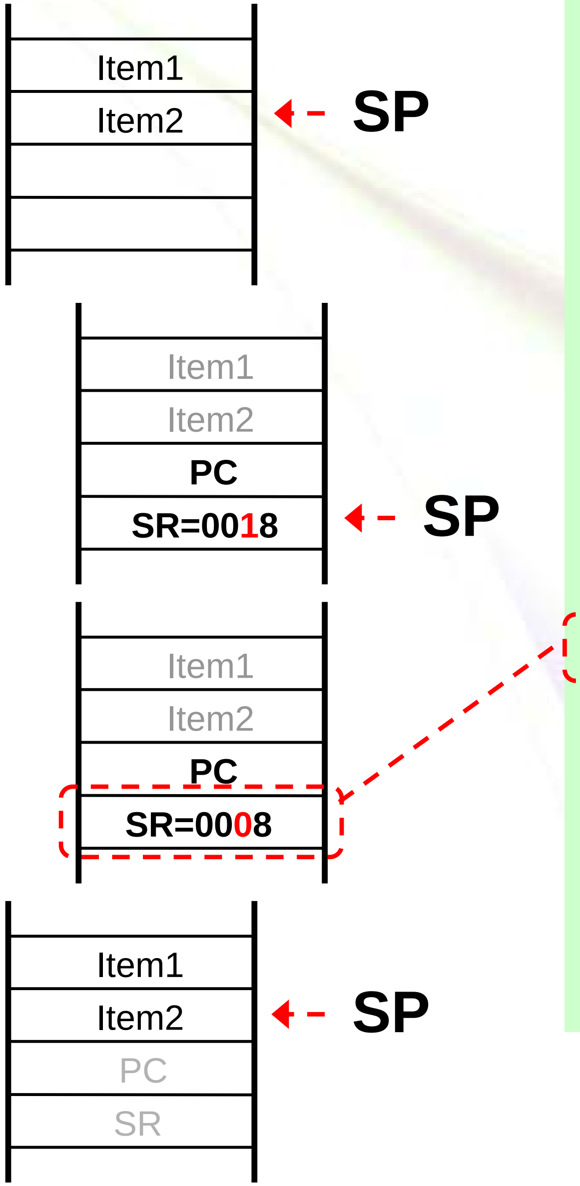
- **PC** pushed
- **SR** pushed
- Interrupt vector moved to PC
- **SR** is cleared
- IFG flag cleared on single source flags



`reti` - automatically

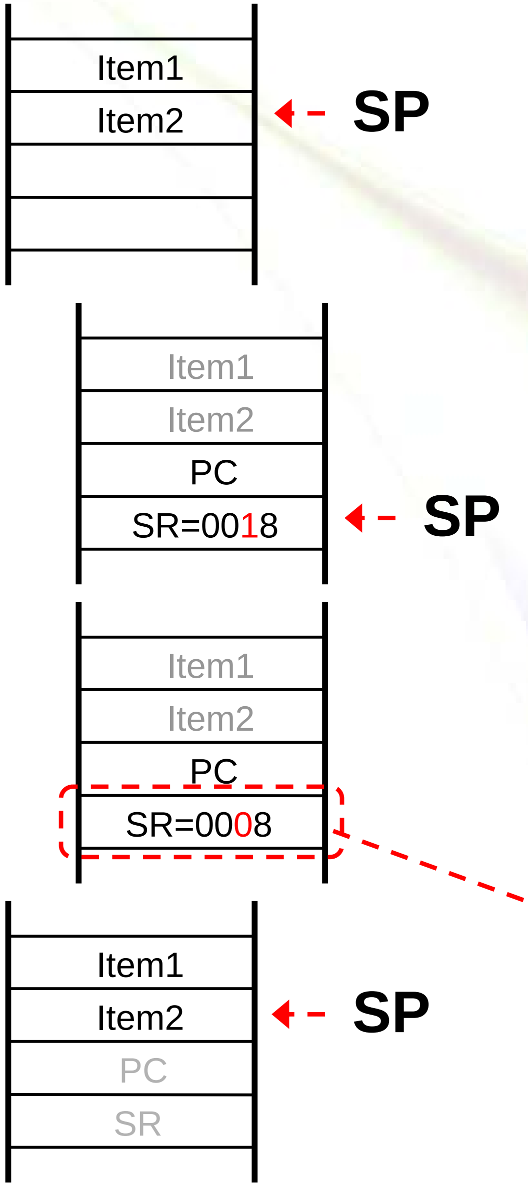
- SR popped - *original*
- PC popped

Low Power Modes In Assembler




```
      ORG      0F000h
RESET         mov.w    #300h,SP
              mov.w
#WDT_MDLY_32,&WDTCTL
              bis.b    #WDTIE,&IE1
              bis.b    #01h,&P1DIR

Mainloop      bis.w    #CPUOFF+GIE,SR
              xor.b    #01h,&P1OUT
              jmp      Mainloop

WDT_ISR       bic.w    #CPUOFF,0(SP)
              reti

              ORG      0FFFEh
              DW       RESET
              ORG      0FFF4h
              DW       WDT_ISR
```

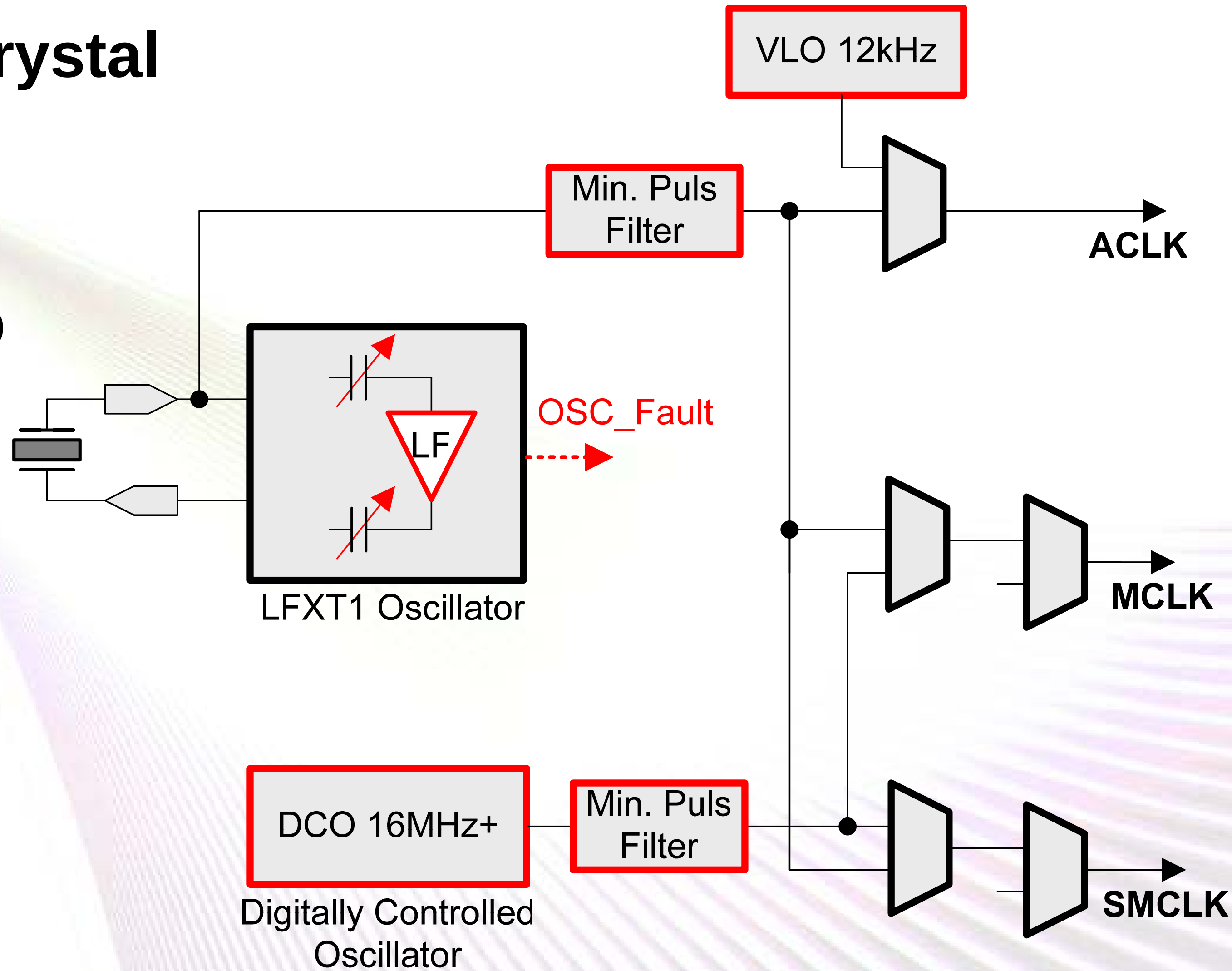
Low Power Modes In C



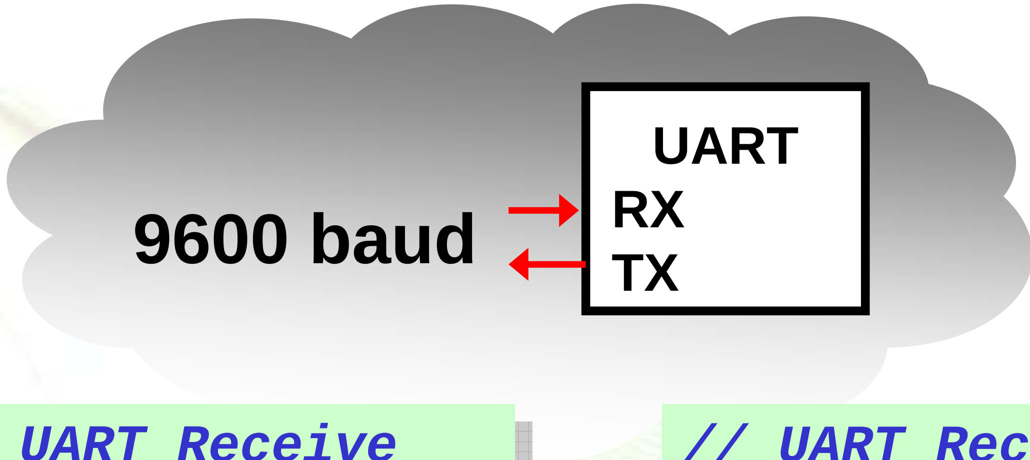

```
void main(void)
{
  WDTCTL = WDT_MDLY_32;
  IE1 |= WDTIE;
  P1DIR |= 0x01;

  for (;;)
  {
    _BIS_SR(CPUOFF + GIE);
    P1OUT ^= 0x01;
  }
}
#pragma vector=WDT_VECTOR
__interrupt void
watchdog_timer(void)
{
    _BIC_SR_IRQ(CPUOFF);
}
```


2xx *Basic Clock Module*+ with VLO Clock

- **VLO provides crystal alternative**
- **Lower power**
- **< 500 nano-amp**



Interrupts Control Program Flow



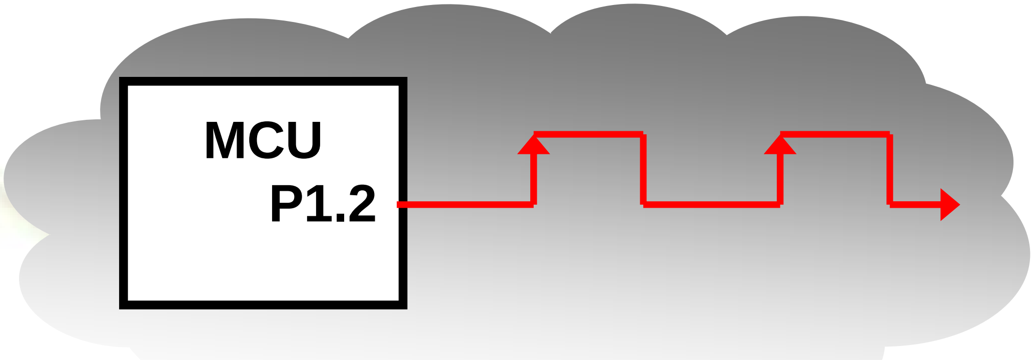

```
// Polling UART Receive
for (;;)
{
 while (!(IFG2&URXIFG0));
 TXBUF0 = RXBUF0;
}
```

100% CPU Load

```
// UART Receive Interrupt
#pragma vector=UART_VECTOR
__interrupt void rx (void)
{
 TXBUF0 = RXBUF0;
}
```

0.1% CPU Load

Software Functions >> Peripherals




```
// Endless Loop
for (;;)
{
 P1OUT |= 0x04;  // Set
 delay1();
 P1OUT &= ~0x04; // Reset
 delay2();
}
```

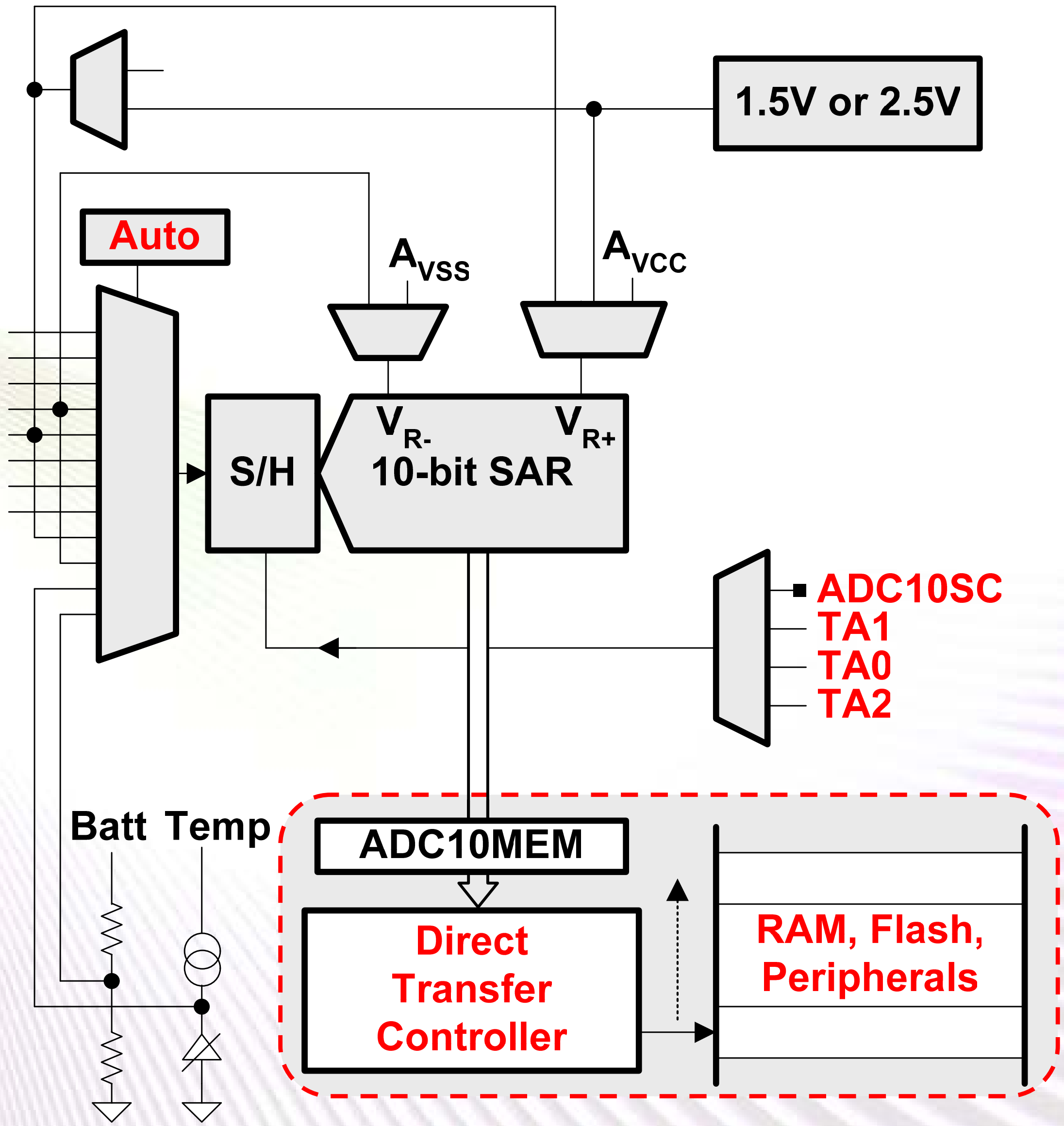
100% CPU Load

```
// Setup output unit
 CCTL1 = OUTMOD0_1;
 _BIS_SR(CPUOFF);
```

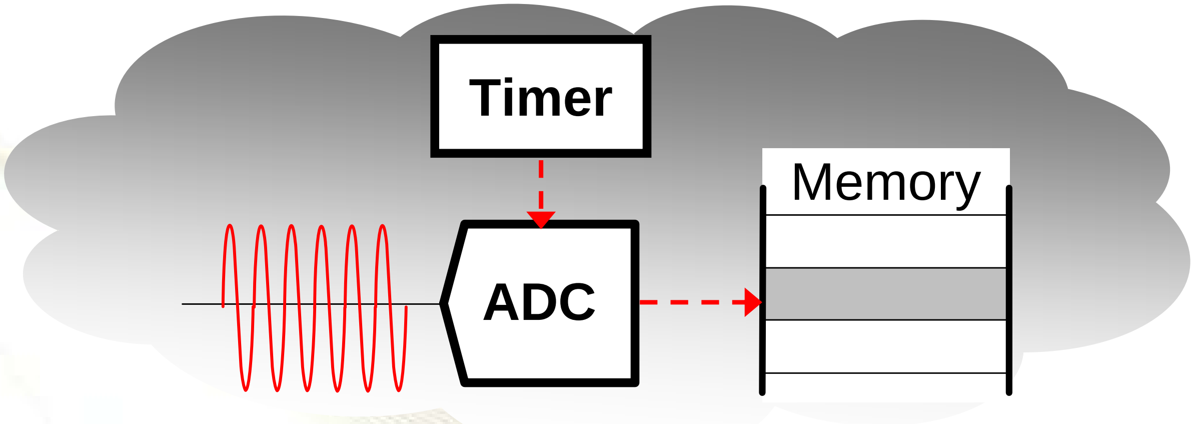
Zero CPU Load

MSP430 ADC10

- 10-bit ADC
- 200ksps+
- Autoscan
- Single
  Sequence
  Repeat-single
  Repeat-sequence
- Internal/external reference
- TA SOC triggers
- Direct transfer controller (DTC)

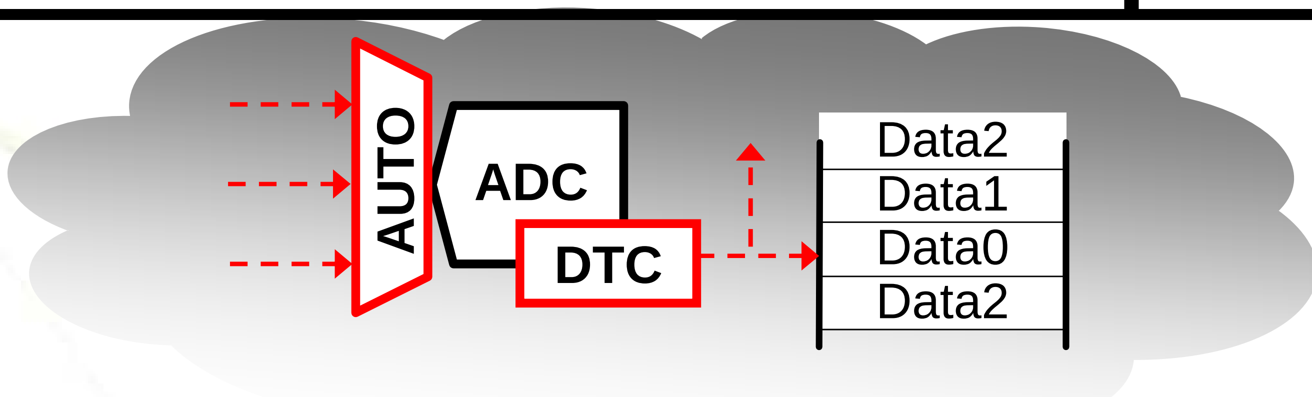


Is Timer-Triggered ADC Important?




```
// Interrupt                                       CPU cycles
; MSP430 ISR to start conversion                       6
BIS  #ADC12SC,&ADC12CTL0 ; Start conversion            5
RETI                     ; Return                      5
                         ;                            16
```

Why Is Autoscan + DTC Important?



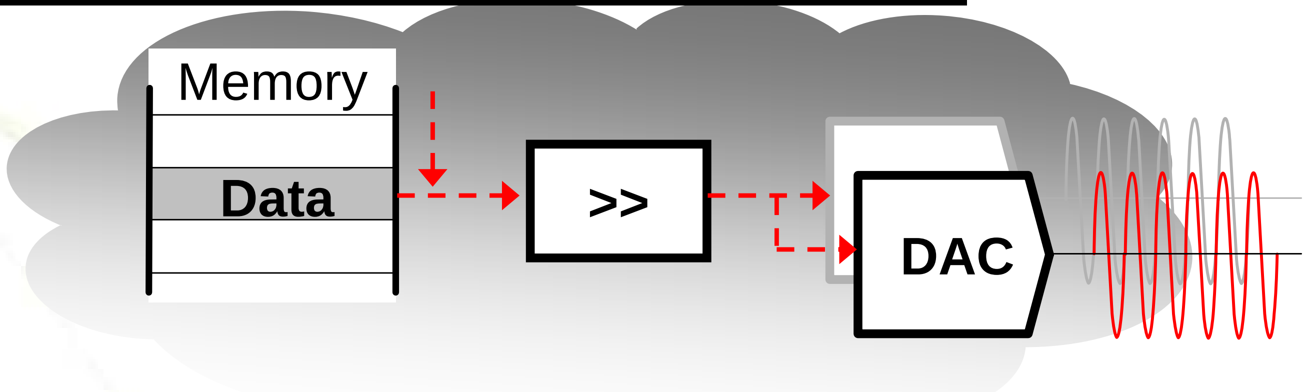

```
// Software
Res[pRes++] = ADC10MEM;
ADC10CTL0 &= ~ENC;
if (pRes < NR_CONV)
 {
 CurrINCH++;
 if (CurrINCH == 3)
   CurrINCH = 0;
 ADC10CTL1 &= ~INCH_3;
 ADC10CTL1 |= CurrINCH;
 ADC10CTL0 |= ENC+ADC10SC;
 }
```

```
// Autoscan + DTC
 _BIS_SR(CPUOFF);
```

70 cycles/Sample

Fully Automatic

Why Is DMA Important?



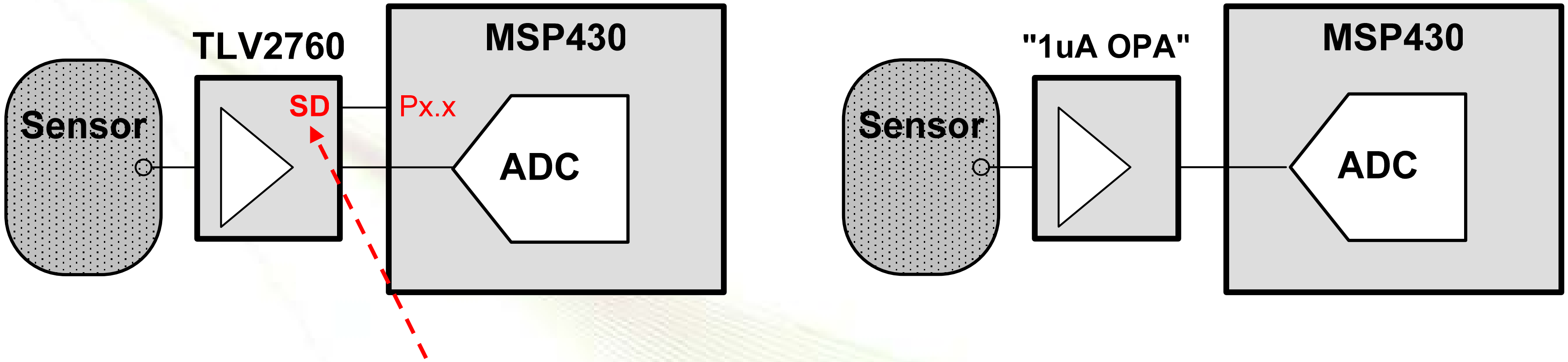

```
// Interrupt                                          CPU cycles  DMA clocks
; MSP430 ISR for one output waveform                       6          0
MOV   @R5+,&DAC12_0DAT ; Update DAC0                       5          2
AND   #1F,R5           ; Modulo pointer                    2          0
RETI                   ; Return                            5          0
                       ;                                  18          2

; MSP430 ISR for two output waveforms                      6          0
MOV   @R5+,&DAC12_0DAT ; Update DAC0                       5          2
MOV   @R5+,&DAC12_1DAT ; Update DAC1                       5          2
AND   #3F,R5           ; Modulo pointer                    2          0
RETI                   ; Return                            5          0
                       ;                                  23          4
```

Low-Power Peripheral Features

- ADC10 reference buffer automatically controlled
- ADC10, ADC12, SD16 cores automatically controlled
- Auto-scan ADC modes
- Timer-triggered data conversion
- I2C and USCI modules automatically enable clock
- DAC and OA have speed vs. power settings

What can I do without the CPU?

Power Manage External Devices



0.01uA = Shutdown

20uA = Active

0.06uA = Average

1uA = Quiescent

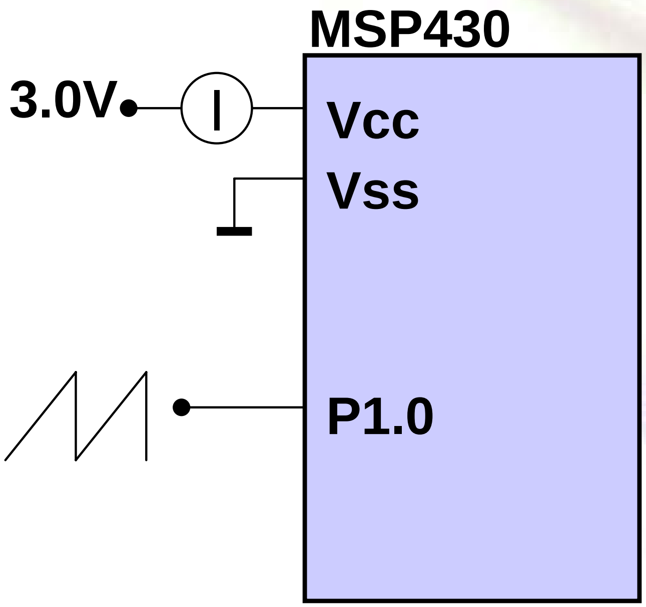
1uA = Active

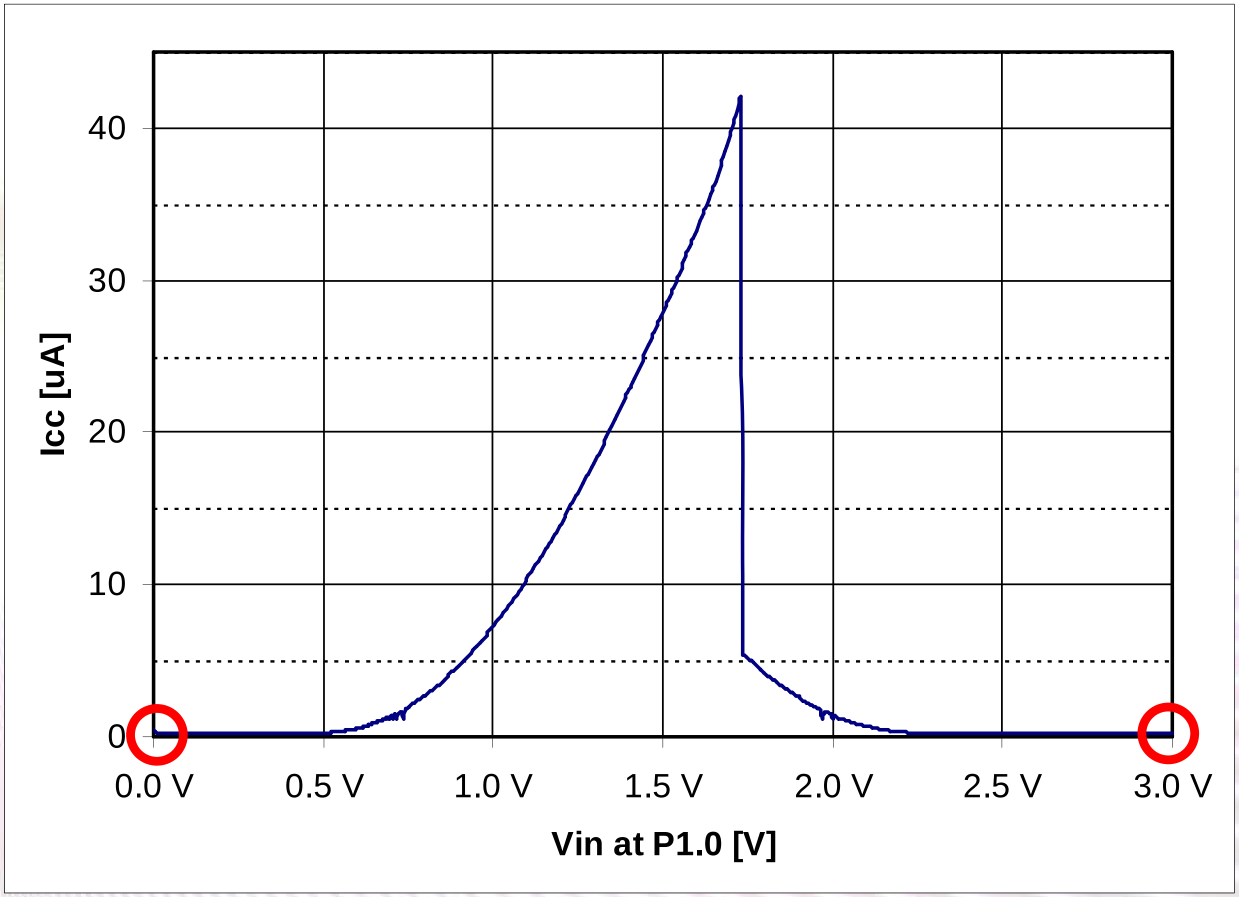
1uA = Average

- **OPA with shutdown can be 20x lower total power**

How To Terminate Unused Pins?

- **Floating inputs cause additional current consumption!**





- **Please see last page of chapter 2 in User's Guide**

Agenda

- Introduction to Ultra-Low Power
- Looking for Ultra-Low Power Parts
- MSP430 – The Ultra-Low Power MCU
- Low-Power Efficient Coding Techniques
- Summary

Bytes, Words & CPU Registers

```
; 16-bit addition                                    Code/Cycles
5405               add.w   R4,R5                      ; 1/1
529202000202       add.w   &0200,&0202                ; 3/6

; 8-bit addition
5445               add.b   R4,R5                      ; 1/1
52D202000202       add.b   &0200,&0202                ; 3/6
```

- **Use CPU registers for calculations and dedicated variables**
- **Same code size for word or byte**
- **Use word operations when possible**

Effect Of The Constant Generator

```
D3E20021         bis.b   #002h,&P1OUT        ; With CG

D0F200100021     bis.b   #010h,&P1OUT        ; Without CG
```

- **Immediate values 0xFFFF, 0, 1, 2, 4, 8 generated in hardware**
- **Reduces code size and cycles**

Completely Automatic!

Interrupt Vector Generator

15													0	
0	0	0	0	0	0	0	0	0	0	0	0	Interrupt vector	0	TAIV

- TAIV is used to efficiently decode the TIMER_A1 interrupt vector for all other interrupt sources
- Contents is either 0, 2, 4, or 10
- Reading TAIV returns and clears the highest-priority pending interrupt
- Add TAIV to the PC and use a jump-table for TAIV de-mux
- Using TAIV instead of IFG polling greatly reduces interrupt overhead

C Coding Tips

- **Use local variable as much as possible. Local variables use CPU registers whereas global variables use RAM.**
- **Use bit mask instead of bitfields for unsigned int and unsigned char.**
- **Use unsigned data types where possible**
- **Use pointers to access structures and unions**
- **Use "`static const`" class to avoid run-time copying of structures, unions, and arrays.**
- **Avoid modulo**
- **Count down "`for`" loops**
- **Get to know your C code and its disassembly!**

Agenda

- Introduction to Ultra-Low Power
- Looking for Ultra-Low Power Parts
- MSP430 – The Ultra-Low Power MCU
- Low-Power Efficient Coding Techniques
- Summary

Principles For ULP Applications

- **Maximize the time in standby (LPM3)**
- **Use interrupts to control program flow**
- **Replace software functions with peripheral hardware**
- **Power manage internal peripherals**
- **Power manage external devices**
- **Device choice can make a difference**
- **Effective code is a must. Every unnecessary instruction executed is a portion of the battery wasted that will never return.**

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
RFID	www.ti-rfid.com	Telephony	www.ti.com/telephony
Low Power Wireless	www.ti.com/lpw	Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265