Câu chuyện bắt đầu từ một cậu bé,
và một ý tưởng
có thể
làm thay đổi thế giới...

PAY IT FORWARD

Đó là khi bạn giúp đỡ 3 người bạn không quen biết,
dù là bằng thời gian,
hay công sức,
hay kinh nghiệm,
hay kiến thức,
hay tiền bạc, ...
của mình.

Mà không chờ đợi một sự báo ân nào.

Chỉ cần mỗi người trong 3 người đó,
lại đem những gì mình có, mà người khác cần,
tiếp tục giúp đỡ thêm 3 người nữa.

Chính những người-giúp-đỡ, và người-được-giúp-đỡ,
sẽ là những người góp phần thay đổi thế giới...

Một thế giới sẻ chia kiến thức - và yêu thương ...

PAY IT FORWARD ...

Chúng tôi không sáng tạo ra câu nói này.

Pay it forward...

Hãy tri ân người giúp mình bằng cách giúp đỡ người khác
Cho đi không phải để nhận lại.

BASIC CLOCK MODULE
LOW POWER MODE

10/04/2013

MSP430G2553

payitforward.edu.vn

C8

payitforward.edu.vn

I/ BASIC CLOCK

1/ Introduction

2/ Internal Oscillators

3/ External Crystals

4/ Clock sources

5/ Clock Signals

6/ Choice of Oscillator

7/ Clock System Registers

II/ LOW POWER MODE

Chú ý: Phần Clock cho MSP430 tương đối khó hiểu. Tuy nhiên, người học có thể không cần hiểu hết các nội dung của bài này trước khi đọc bài tiếp theo. Phần kiến thức về Clock có thể được củng cố (đọc lại) dần dần khi tìm hiểu các module của MSP430.

BASIC CLOCK MODULE

1/ Introduction

Why are clocks important?

- Clocks are at the heart of any synchronous digital system

- The speed of instruction execution will depend on the clock.

- It presents many difficulties such as when the data to be processed from comes from different places at different speeds

BASIC CLOCK MODULE

1/ **Introduction**

How do we generate the clocks?

There are many ways to do so, but in the MSP430 (and many other microcontrollers) there are generally three types of clock sources:

- Internal Oscillators
- External Crystals
- External Oscillators

The implementation of these goals is largely based on the ability to select different clocks for different parts of the chip.

BASIC CLOCK MODULE

2/ **Internal Oscillators**:

- Internal Oscillators are usually an RC network with circuitry to try and improve the accuracy of the clock

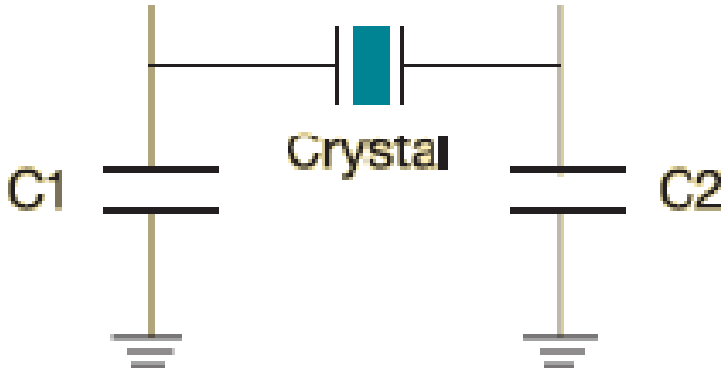
- The benefit of this type of oscillators is that their frequency

can be easily changed and they don’t occupy any more space on the PCB. On the MSP430, A fast Digitally Controller Oscillator (DCO) oscillator is avaialble,

BASIC CLOCK MODULE

3/ **External Crystals**:

- External Crystals add a large measure of accuracy to oscillators and should be used as much as possible unless cost and area considerations are more important.



BASIC CLOCK MODULE

4/ **Clock sources**

- The four clock sources are:

LFXT1CLK

XT2CLK

DCOCLK

VLOCLK

BASIC CLOCK MODULE

4/ **Clock sources**

- LFXT1CLK: Low-frequency/high-frequency oscillator that can be used with low-frequency watch crystals or external clock sources of 32768 Hz or with standard crystals, resonators, or external clocksources in the 400-kHz to 16-MHz range

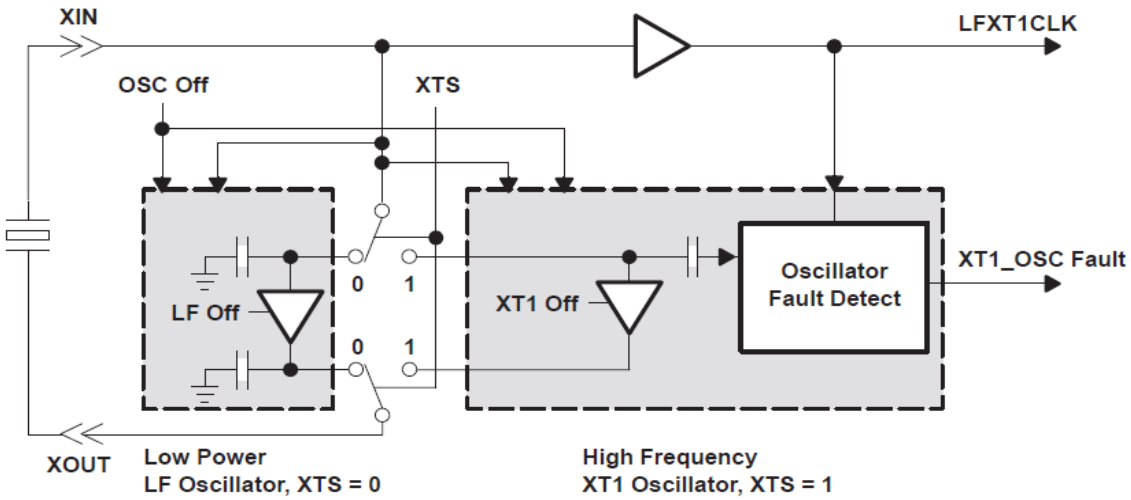


Figure 1. The LF/XT1 Oscillator

BASIC CLOCK MODULE

4/ **Clock sources**

- XT2CLK: Optional high-frequency oscillator that can be used with standard crystals, resonators, orexternal clock sources in the 400-kHz to 16-MHz range. MSP430G2xx3: LFXT1 does not support HF mode, XT2 is not present, ROSC is not supported.

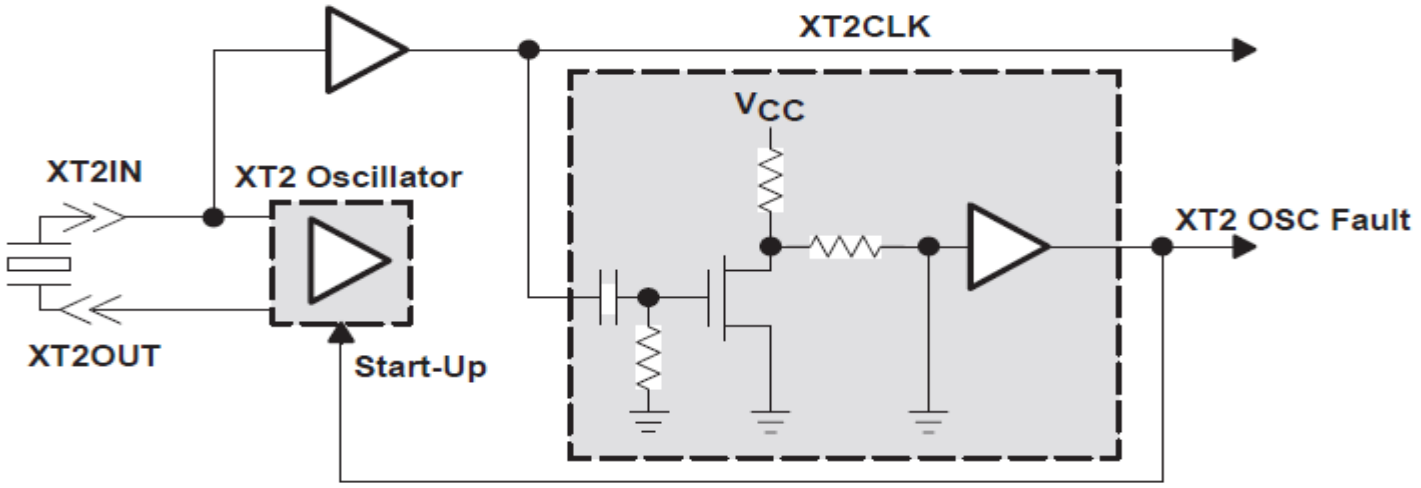


Figure 2. The XT2 Oscillator

BASIC CLOCK MODULE

4/ **Clock sources**

- DCOCLK: Internal digitally controlled oscillator (DCO)

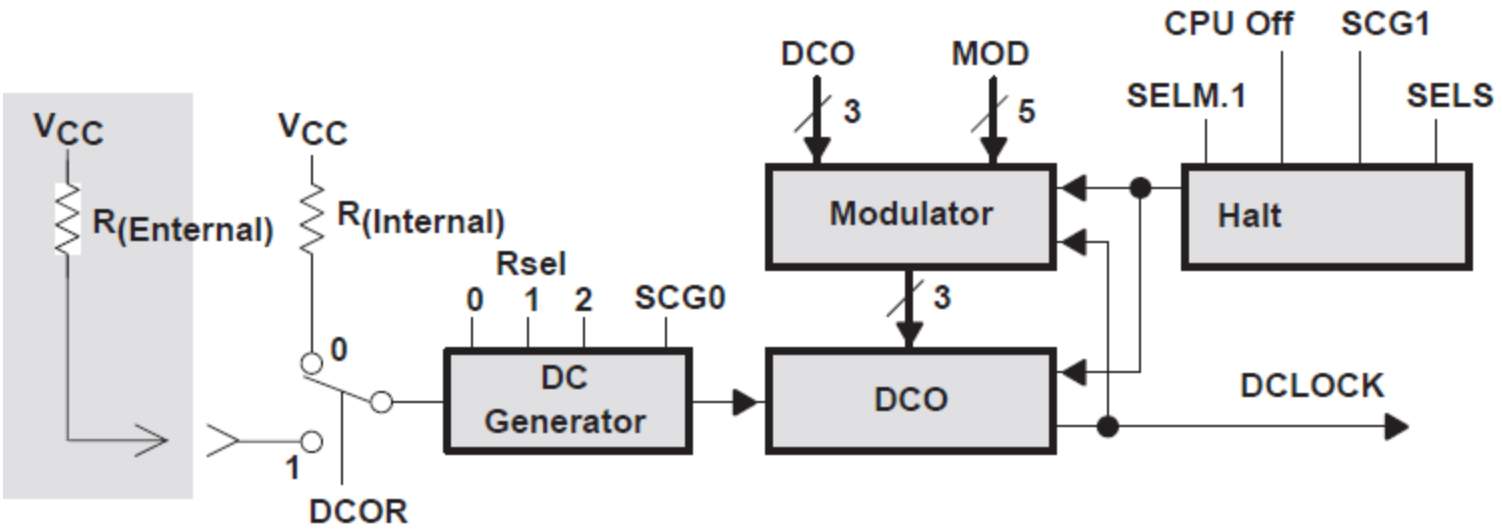
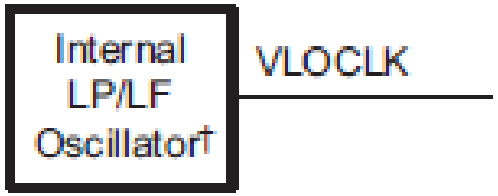


Figure 3. The DCO

BASIC CLOCK MODULE

4/ **Clock sources**

- VLOCLK: Internal very low power, low frequency oscillator with 12-kHz typical frequency



Summary: If you need more precision, use the external crystals at the expense of PCB space and some money. It is standard practice to use LFXT1 with a 32.768 kHz crystal, leaving XT2 to be used with a high frequency crystal.

BASIC CLOCK MODULE

5/ **Clock Signals**:

Three clock signals are available from the basic clock module+:

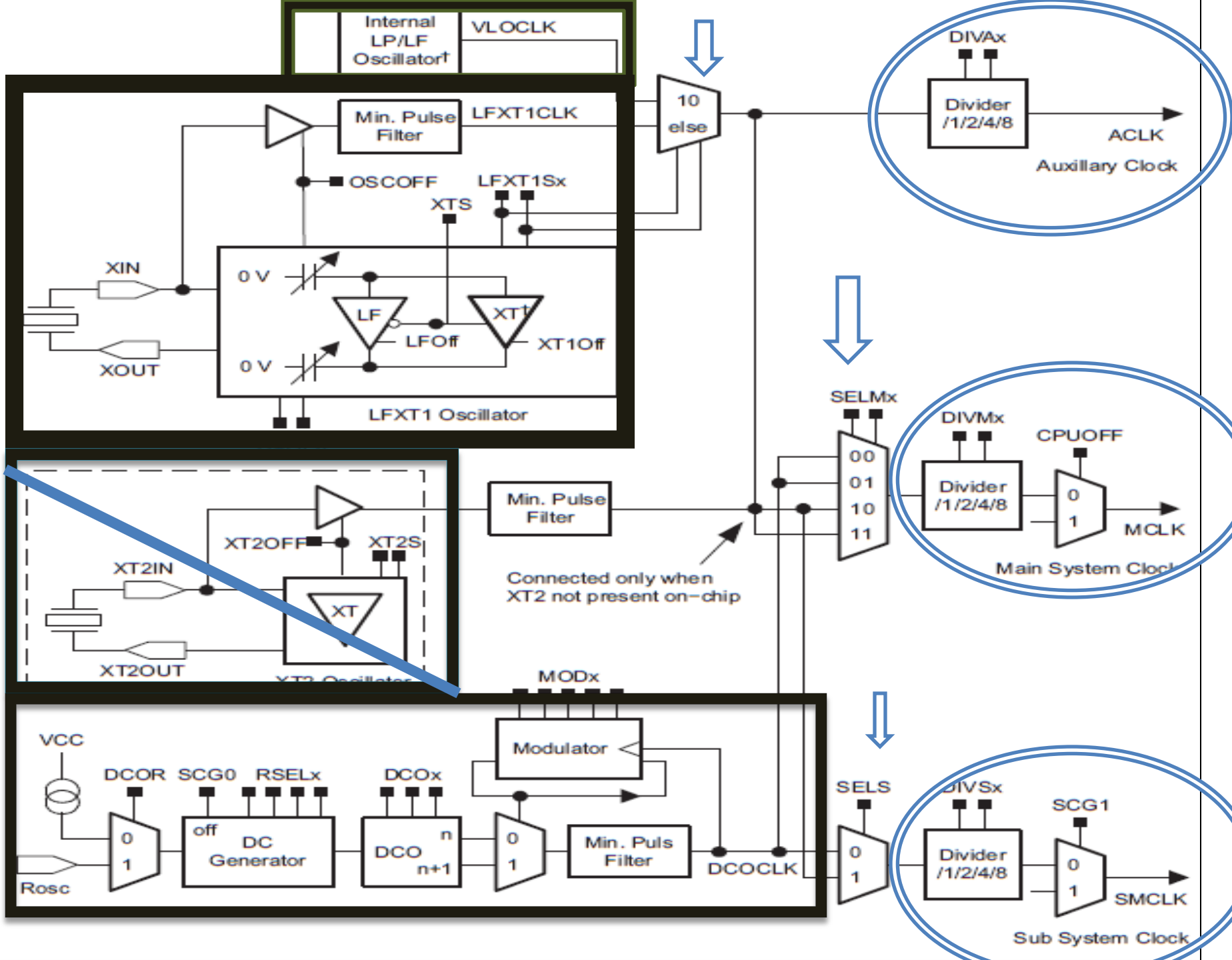
ACLK: Auxiliary clock. ACLK is software selectable as LFXT1CLK or VLOCLK. ACLK
is divided by 1, 2, 4, or 8. ACLK is software selectable for individual peripheral modules.

MCLK: Master clock. MCLK is software selectable as LFXT1CLK, VLOCLK, XT2CLK (if available on-chip), or DCOCLK. MCLK is divided by 1, 2, 4, or 8. MCLK is used by the
CPU and system.

BASIC CLOCK MODULE

5/ **Clock Signals**:

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


Internal LP/LF Oscillator†
VLOCLK
Min. Pulse Filter
LFXT1CLK
OSCOFF
LFXT1Sx
XTS
XIN
XOUT
0 V
0 V
LF
LFOff
XT1
XT1Off
LFXT1 Oscillator
10
else
DIVAx
Divider /1/2/4/8
ACLK
Auxillary Clock
SELMx
00
01
10
11
DIVMx
CPUOFF
Divider /1/2/4/8
0
1
MCLK
Main System Clock
XT2OFF
XT2S
XT2IN
XT
XT2OUT
Min. Pulse Filter
Connected only when XT2 not present on-chip
MODx
Modulator
VCC
DCOR
SCG0
RSELx
DCOx
off
DC Generator
DCO
n
n+1
Rosc
Min. Puls Filter
DCOCLK
SELS
DIVSx
SCG1
Divider /1/2/4/8
SMCLK
Sub System Clock

BASIC CLOCK MODULE

6/ **Choice of Oscillator:**

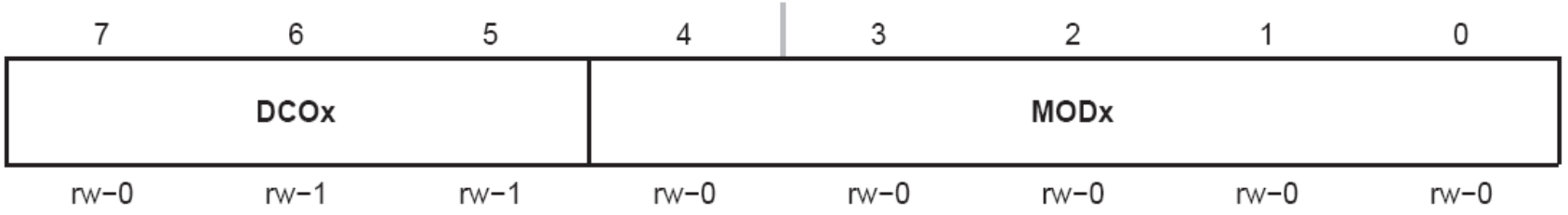
- DCO:
  - + It starts oscillating immediately, with no start-up delay.
  - + It has a relatively low operating current
  - + its output frequency will drift with supply voltage and temperature
- In applications requiring very stable or very precise clock frequencies, the XT1 or XT2 high frequency oscillators can be used.

payitforward.edu.vn

BASIC CLOCK MODULE

7/ Clock System Registers

DCOCTL, DCO Control Register



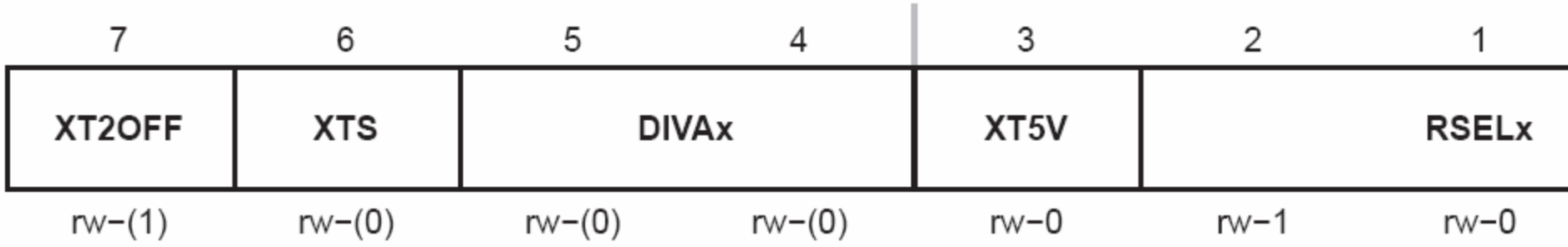
DCOx	Bits 7-5	DCO frequency select. These bits select which of the eight discrete DCO frequencies of the RSELx setting is selected.
MODx	Bits 4-0	Modulator selection. These bits define how often the f_{DCO+1} frequency is used within a period of 32 DCOCLK cycles. During the remaining clock cycles (32−MOD) the f_{DCO} frequency is used. Not useable when DCOx=7.

DCO Frequency

over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted)

PARAMETER		TEST CONDITIONS	V_{CC}	MIN	TYP	MAX	UNIT
V_{CC}	Supply voltage	RSELx < 14		1.8		3.6	V
		RSELx = 14		2.2		3.6	
		RSELx = 15		3		3.6	
$f_{DCO(0,0)}$	DCO frequency (0, 0)	RSELx = 0, DCOx = 0, MODx = 0	3 V	0.06		0.14	MHz
$f_{DCO(0,3)}$	DCO frequency (0, 3)	RSELx = 0, DCOx = 3, MODx = 0	3 V	0.07		0.17	MHz
$f_{DCO(1,3)}$	DCO frequency (1, 3)	RSELx = 1, DCOx = 3, MODx = 0	3 V		0.15		MHz
$f_{DCO(2,3)}$	DCO frequency (2, 3)	RSELx = 2, DCOx = 3, MODx = 0	3 V		0.21		MHz
$f_{DCO(3,3)}$	DCO frequency (3, 3)	RSELx = 3, DCOx = 3, MODx = 0	3 V		0.30		MHz
$f_{DCO(4,3)}$	DCO frequency (4, 3)	RSELx = 4, DCOx = 3, MODx = 0	3 V		0.41		MHz
$f_{DCO(5,3)}$	DCO frequency (5, 3)	RSELx = 5, DCOx = 3, MODx = 0	3 V		0.58		MHz
$f_{DCO(6,3)}$	DCO frequency (6, 3)	RSELx = 6, DCOx = 3, MODx = 0	3 V	0.54		1.06	MHz
$f_{DCO(7,3)}$	DCO frequency (7, 3)	RSELx = 7, DCOx = 3, MODx = 0	3 V	0.80		1.50	MHz
$f_{DCO(8,3)}$	DCO frequency (8, 3)	RSELx = 8, DCOx = 3, MODx = 0	3 V		1.6		MHz
$f_{DCO(9,3)}$	DCO frequency (9, 3)	RSELx = 9, DCOx = 3, MODx = 0	3 V		2.3		MHz
$f_{DCO(10,3)}$	DCO frequency (10, 3)	RSELx = 10, DCOx = 3, MODx = 0	3 V		3.4		MHz
$f_{DCO(11,3)}$	DCO frequency (11, 3)	RSELx = 11, DCOx = 3, MODx = 0	3 V		4.25		MHz
$f_{DCO(12,3)}$	DCO frequency (12, 3)	RSELx = 12, DCOx = 3, MODx = 0	3 V	4.30		7.30	MHz
$f_{DCO(13,3)}$	DCO frequency (13, 3)	RSELx = 13, DCOx = 3, MODx = 0	3 V	6.00	7.8	9.60	MHz
$f_{DCO(14,3)}$	DCO frequency (14, 3)	RSELx = 14, DCOx = 3, MODx = 0	3 V	8.60		13.9	MHz
$f_{DCO(15,3)}$	DCO frequency (15, 3)	RSELx = 15, DCOx = 3, MODx = 0	3 V	12.0		18.5	MHz
$f_{DCO(15,7)}$	DCO frequency (15, 7)	RSELx = 15, DCOx = 7, MODx = 0	3 V	16.0		26.0	MHz
S_{RSEL}	Frequency step between range RSEL and RSEL+1	$S_{RSEL} = f_{DCO(RSEL+1,DCO)}/f_{DCO(RSEL,DCO)}$	3 V		1.35		ratio
S_{DCO}	Frequency step between tap DCO and DCO+1	$S_{DCO} = f_{DCO(RSEL,DCO+1)}/f_{DCO(RSEL,DCO)}$	3 V		1.08		ratio
	Duty cycle	Measured at SMCLK output	3 V		50		%

BCSCTL1, Basic Clock System Control Register 1



XT2OFF	Bit 7	XT2 off. This bit turns off the XT2 oscillator 0 XT2 is on 1 XT2 is off if it is not used for MCLK or SMCLK.
XTS	Bit 6	LFXT1 mode select. 0 Low frequency mode 1 High frequency mode
DIVAx	Bits 5-4	Divider for ACLK 00 /1 01 /2 10 /4 11 /8
XT5V	Bit 3	Unused. XT5V should always be reset.
RSELx	Bits	Register Select. The internal register is selected in eight differ

BCSCTL2, Basic Clock System Control Register 2

SELMx	Bits	Select MCLK. These bits select the MCLK source.	
	7-6	00	DCOCLK
		01	DCOCLK
		10	XT2CLK when XT2 oscillator present on-chip. LFXT1CLK when XT2 oscillator not present on-chip.
		11	LFXT1CLK
DIVMx	BitS	Divider for MCLK	
	5-4	00	/1
		01	/2
		10	/4
		11	/8
SELS	Bit 3	Select SMCLK. This bit selects the SMCLK source.	
		0	DCOCLK
		1	XT2CLK when XT2 oscillator present on-chip. LFXT1CLK when XT2 oscillator not present on-chip.
DIVSx	BitS	Divider for SMCLK	
	2-1	00	/1
		01	/2
		10	/4

BASIC CLOCK MODULE

7/ Clock System Registers

-Default:

ACLK = LFXT1 , MCLK = SMCLK = default DCO

- ACLK = VL0, MCLK = VLO/8 ~1.5kHz, SMCLK = n/a

```
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
```

BASIC CLOCK MODULE

7/ Clock System Registers

ACLK = LFXT1 , MCLK = SMCLK = Selectable at 1 MHZ

```
if (CALBC1_1MHZ ==0xFF || CALDCO_1MHZ == 0xFF)
  {
    while(1);                          // If calibration constants erased
                                       // do not load, trap CPU!!
  }
 //1Mhz
  BCSCTL1 = CALBC1_1MHZ;               // Set range
  DCOCTL = CALDCO_1MHZ;                // Set DCO step +
modulation */
```

BASIC CLOCK MODULE

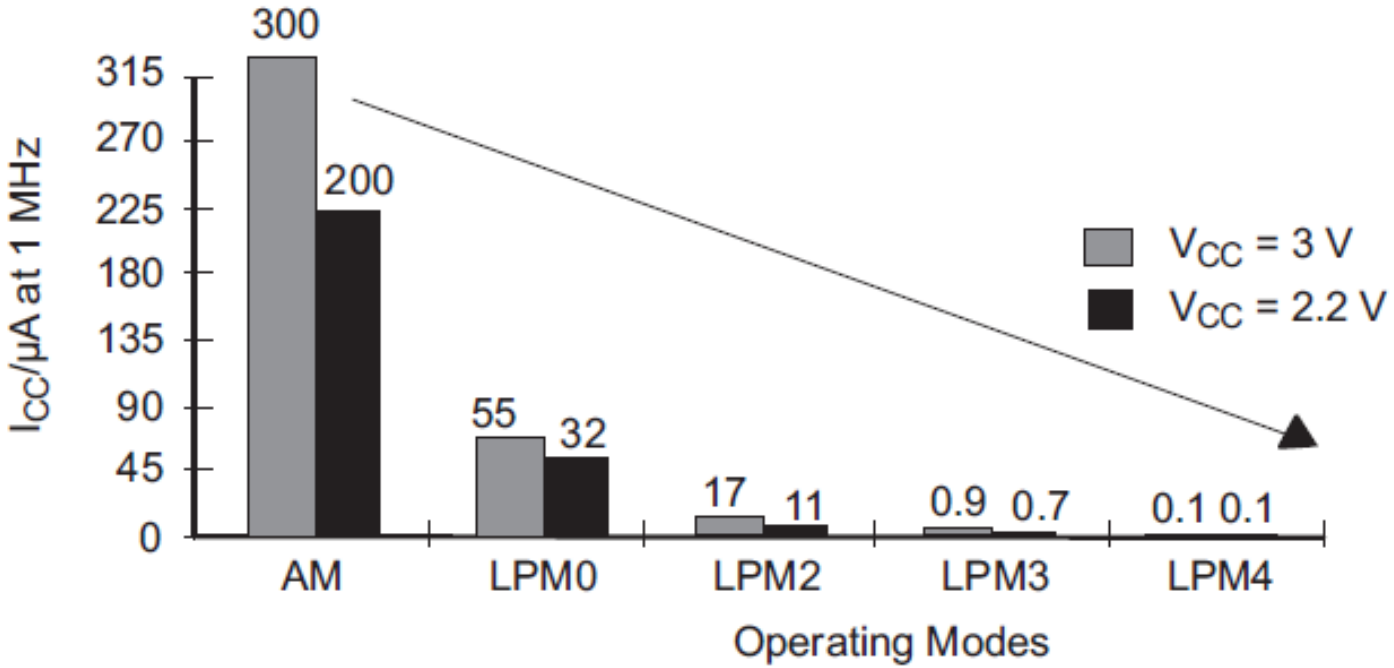
7/ Clock System Registers

ACLK = LFXT1 , MCLK = SMCLK = Selectable at 1 MHZ

```
if (CALBC1_1MHZ ==0xFF || CALDCO_1MHZ == 0xFF)
  {
    while(1);                          // If calibration constants erased
                                       // do not load, trap CPU!!
  }
 //1Mhz
  BCSCTL1 = CALBC1_1MHZ;               // Set range
  DCOCTL = CALDCO_1MHZ;                // Set DCO step +
modulation */
```

LOW POWER MODE

II/ LOW POWER MODE



Typical Current Consumption of 'F21x1 Devices vs Operating Modes

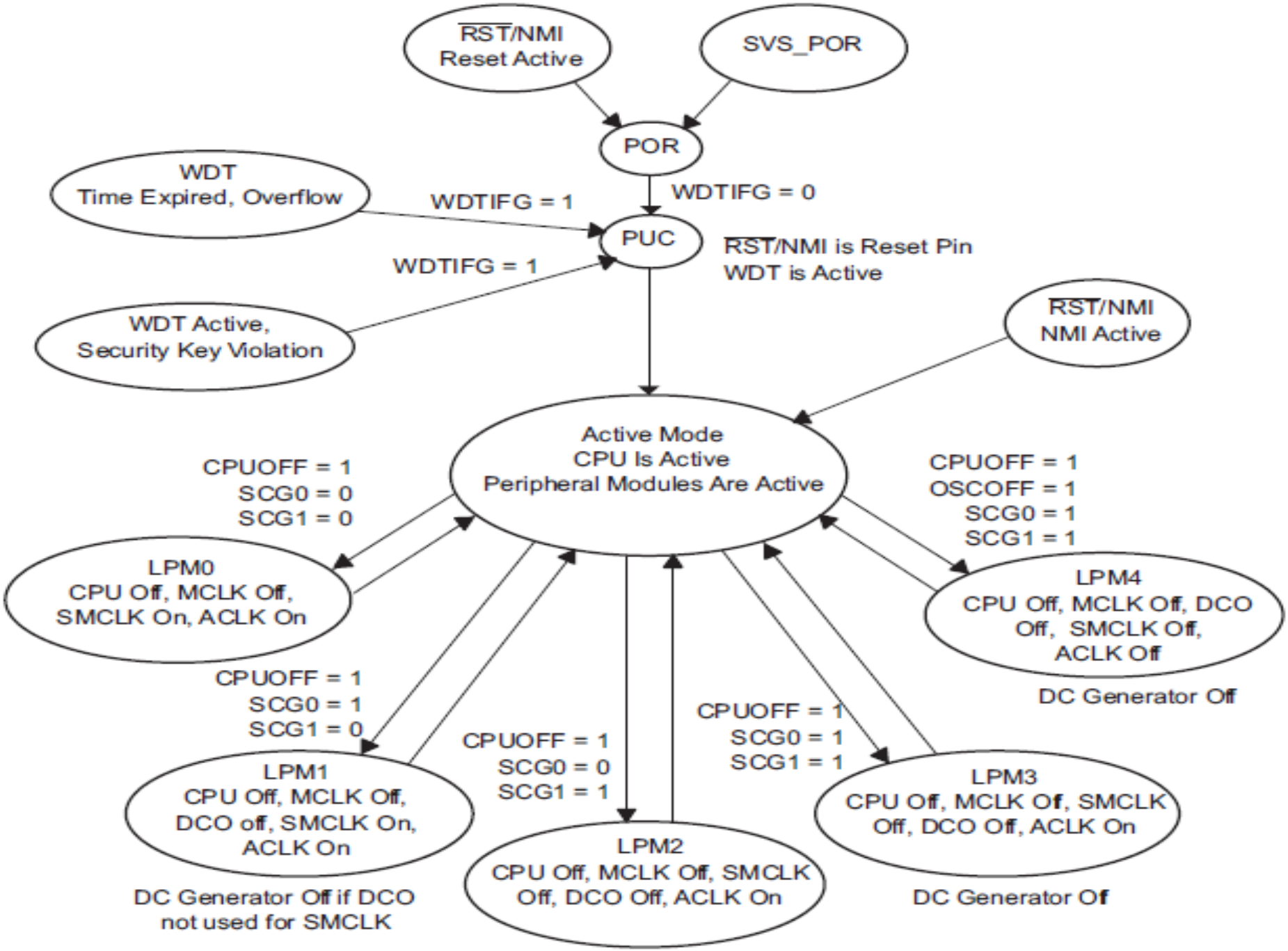


Figure 2-9. Operating Modes For Basic Clock System

payitforward.edu.vn

LOW POWER MODE

II/ LOW POWER MODE

Table 2-2. Operating Modes For Basic Clock System

SCG1	SCG0	OSCOFF	CPUOFF	Mode	CPU and Clocks Status
0	0	0	0	Active	CPU is active, all enabled clocks are active
0	0	0	1	LPM0	CPU, MCLK are disabled, SMCLK, ACLK are active
0	1	0	1	LPM1	CPU, MCLK are disabled. DCO and DC generator are disabled if the DCO is not used for SMCLK. ACLK is active.
1	0	0	1	LPM2	CPU, MCLK, SMCLK, DCO are disabled. DC generator remains enabled. ACLK is active.
1	1	0	1	LPM3	CPU, MCLK, SMCLK, DCO are disabled. DC generator disabled. ACLK is active.
1	1	1	1	LPM4	CPU and all clocks disabled

LOW POWER MODE

II/ LOW POWER MODE

1/Low-Power Mode 0 and 1 (LPM0 and LPM1):

All I/O port pins and RAM/registers are unchanged. Wake up is possible through all enabled interrupts.

2/ Low-Power Modes 2 and 3 (LPM2 and LPM3):

All I/O port pins and the RAM/registers are unchanged. Wake up is possible by enabled interrupts coming from active peripherals or **RST/NMI**

LOW POWER MODE

II/ LOW POWER MODE

3/ Low-Power Mode 4 (LPM4)

All activities cease; only the RAM contents, I/O ports, and registers are maintained. Wake up is only possible by enabled external interrupts

payitforward.edu.vn

LOW POWER MODE

II/ LOW POWER MODE

Using C

```
__bis_SR_register(CPUOFF + GIE);        // LPM0, ADC10_ISR will force exit

// . . .

// ADC10 interrupt service routine
#pragma vector=ADC10_VECTOR
__interrupt void ADC10_ISR(void) {
  __bic_SR_register_on_exit(CPUOFF);    // Clear CPUOFF bit from 0(SR)
}
```

In main routine
(enter LPM mode)

In ISR (exit LPM when returning to main program).

LOW POWER MODE

II/ LOW POWER MODE

```
 BCSCTL1 |= DIVA_2;                    // ACLK/4
  WDTCTL = WDT_ADLY_1000;              // WDT 1s/4 interval timer
  IE1 |= WDTIE;                        // Enable WDT interrupt
While(1)
{
            .
            .
            _BIS_SR(LPM3_bits + GIE);          // Enter LPM3
}
#pragma vector=WDT_VECTOR
__interrupt void watchdog_timer (void)
{
   _BIC_SR_IRQ(LPM3_bits);             // Clear LPM3 bits from 0(SR)
}
```

LOW POWER MODE

II/ LOW POWER MODE

```
WDTCTL = WDTPW + WDTHOLD + WDTNMI + WDTNMIES;
IE1 |= NMIIE;                          // Enable NMI
_BIS_SR(LPM0_bits);                    // Enter LPM0

#pragma vector=NMI_VECTOR
__interrupt void nmi_ (void)
{
IFG1 &= ~NMIIFG;                       // Reclear NMI flag in case
bounce
  IE1 |= NMIIE;                        // Enable NMI
}
```

payitforward.edu.vn

Thank for watching

PAY IT FORWARD

payitforward.edu.vn