PLL – Phase Locked Loop

By Dr. Min He

Outline

Clock in Microcontrollers

PLL Block Diagram

PLL Registers

PLL Initialization Steps

Clock in a Microcontroller

Normally, the execution speed of a microcontroller is determined by an external crystal.

TM4C123 has one internal oscillator and an external oscillator, both are 16MHz.

Most microcontrollers have a phase-locked-loop (PLL) that allows the software to adjust the execution speed of the microcontroller.

In TM4C123, by default the internal oscillator is used and the PLL is not initially active.

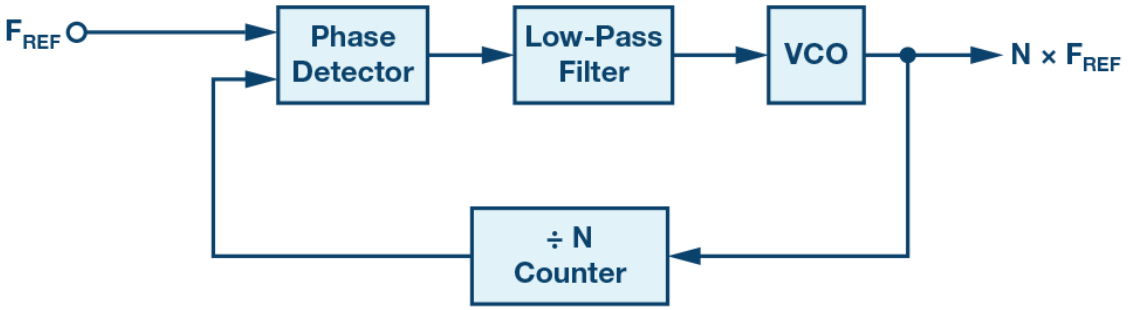
PLL – Phase Locked Loop

Most microcontrollers include a phase-lock-loop (PLL) that allows the software to adjust the execution speed of the computer.

The choice of frequency involves the tradeoff between software execution speed and electrical power.

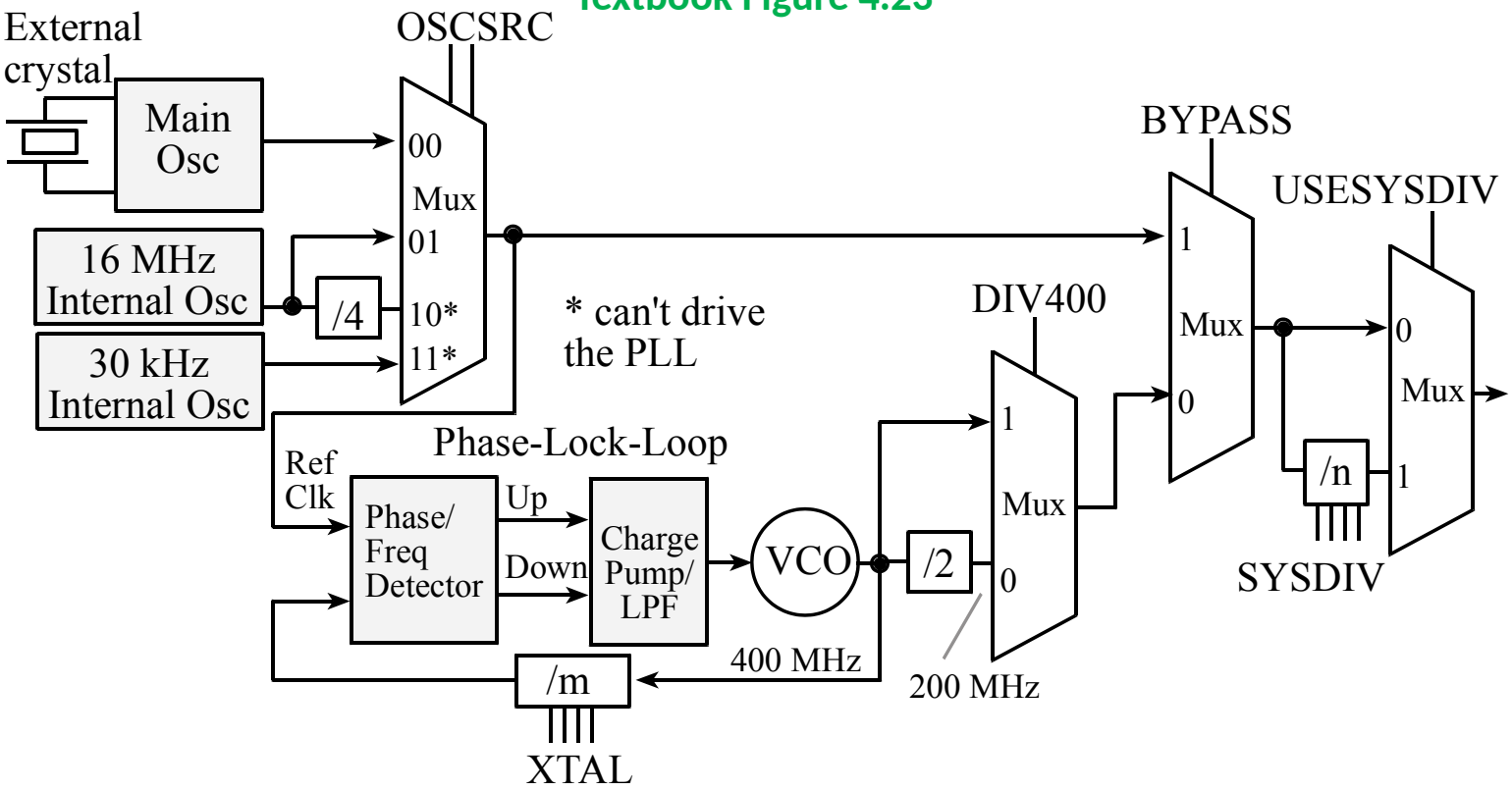
The internal oscillator with frequency 16MHz ±1% is significantly less precise than the crystal, but it requires less power and does not need an external crystal.

PLL can be used to generate stable output high frequency signals from a fixed low-frequency signal.



Phase-Locked-Loop

Textbook Figure 4.23

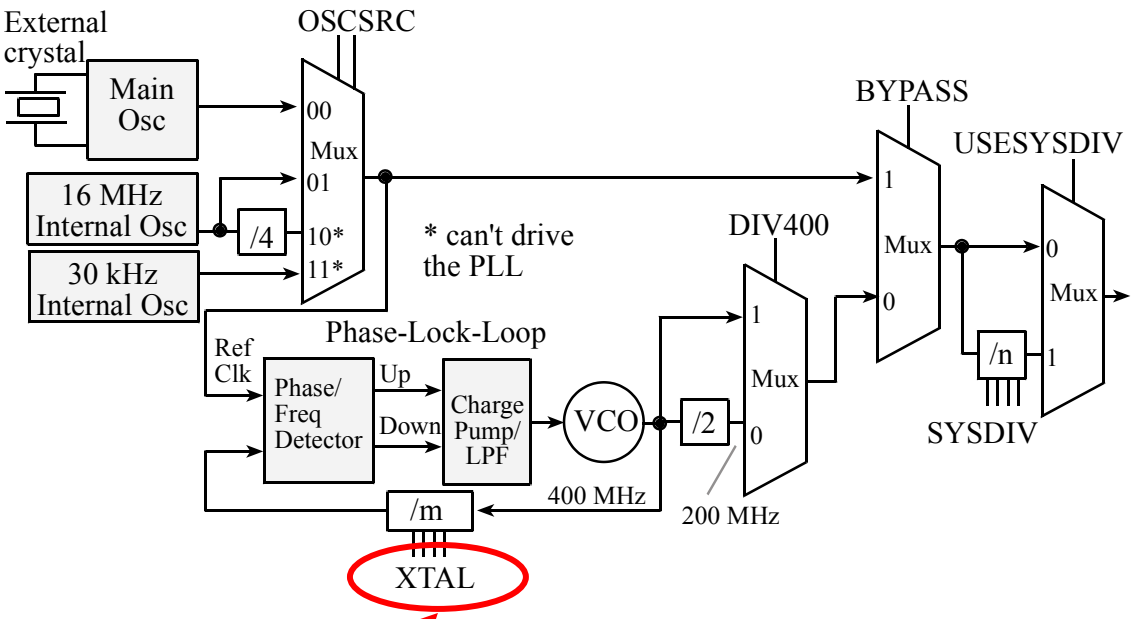


- **A phase-locked loop is a feedback system combining a voltage controlled oscillator (VCO) and a phase comparator so connected that the oscillator maintains a constant phase angle relative to a reference signal.** Charge pump is one of the important parts of PLL which **converts the phase or frequency difference information into a voltage, used to tune the VCO**.
- **The PLL allows us to speed up or slow down the clock, it can be set to a maximum of 80 MHz for Cortex M4.**

Run-Mode Clock Configuration

*Datasheet: page 254, Textbook Figure 4.23, Table 4.9b

- XTAL: bit 10 – 6, this field specifies the crystal value attached to the main oscillator.
- Ex: XTAL: 10101, select 16MHz.



Run-Mode Clock Configuration (RCC)

Base 0x400F.E000
Offset 0x060
Type RW, reset 0x078E.3AD1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved				ACG	SYSDIV				USESYSDIV	reserved	USEPWMDIV	PWMDIV			reserved
Type	RO	RO	RO	RO	RW	RW	RW	RW	RW	RW	RO	RW	RW	RW	RW	RO
Reset	0	0	0	0	0	1	1	1	1	0	0	0	1	1	1	0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved		PWRDN	reserved	BYPASS	XTAL					OSCSRC		reserved			MOSCDIS
Type	RO	RO	RW	RO	RW	RW	RW	RW	RW	RW	RW	RW	RO	RO	RO	RW
Reset	0	0	1	1	1	0	1	0	1	1	0	1	0	0	0	1

Table 4.9a. XTAL field used in the SYSCTL_RCC_R register of the TM4C123

XTAL	*Crystal Freq (MHz)*
0x0	Reserved
0x1	Reserved
0x2	Reserved
0x3	Reserved
0x4	3.579545 MHz
0x5	3.6864 MHz
0x6	4 MHz
0x7	4.096 MHz
0x8	4.9152 MHz
0x9	5 MHz
0xA	5.12 MHz
0xB	6 MHz (reset value)
0xC	6.144 MHz
0xD	7.3728 MHz
0xE	8 MHz
0xF	8.192 MHz

XTAL	*Crystal Freq (MHz)*
0x10	10.0 MHz
0x11	12.0 MHz
0x12	12.288 MHz
0x13	13.56 MHz
0x14	14.31818 MHz
0x15	16.0 MHz
0x16	16.384 MHz
0x17	18.0 MHz
0x18	20.0 MHz
0x19	24.0 MHz
0x1A	25.0 MHz
0x1B	Reserved
0x1C	Reserved
0x1D	Reserved
0x1E	Reserved
0x1F	Reserved

*Datasheet: page 1374, table 24-14

SYSCTL_RCC_R: XTAL: 10101, select 16MHz

SYSCTL_RCC2_R

*Datasheet: page 260

- USERCC2: bit 31, 0: The **RCC** register fields are used, and the fields in **RCC2** are ignored. 1: The **RCC2** register fields override the **RCC** register fields.
- DIV400: bit 30, 0: Use SYSDIV2 as is and apply to 200 MHz predivided PLL output. 1: Append the SYSDIV2LSB bit to the SYSDIV2 field to create a 7 bit divisor using the 400 MHz PLL output. On TM4C, the PLL operates at 400 MHz.
- PWRDN2: bit 13, 0: The PLL operates normally; 1: The PLL is powered down.
- BYPASS2: bit 11, 0: The system clock is the PLL output clock divided by the divisor specified by SYSDIV2; 1:The system clock is derived from the OSC source and divided by the divisor specified by SYSDIV2.
- OSCSRC2: bit 6-4, 0: Main oscillator.

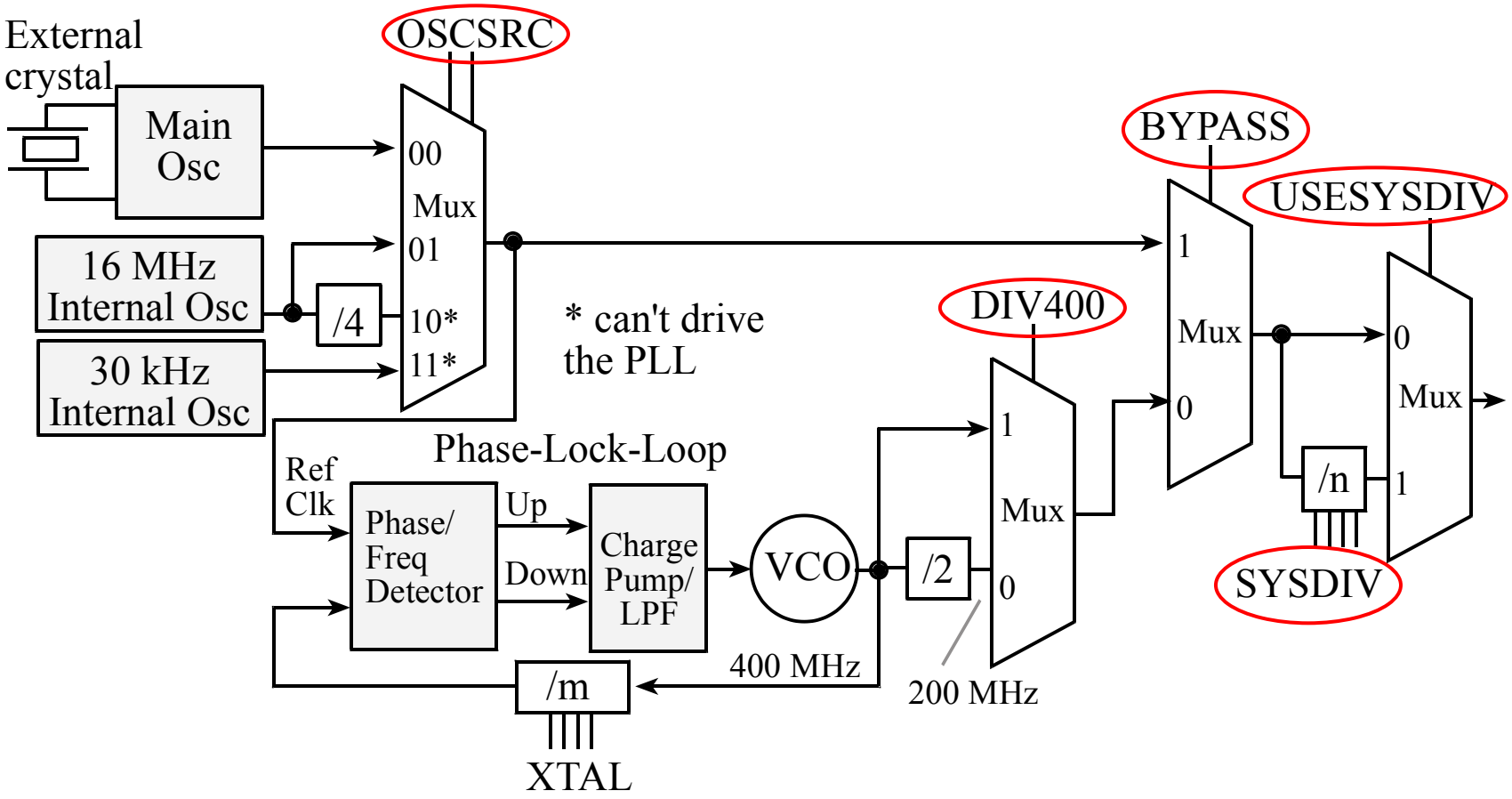
Run-Mode Clock Configuration 2 (RCC2)

Base 0x400F.E000
Offset 0x070
Type RW, reset 0x07C0.6810

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
USERCC2	DIV400	reserved	SYSDIV2						SYSDIV2LSB	reserved					
Type	RW	RW	RO	RW	RW	RW	RW	RW	RW	RW	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved	USBPWRDN	PWRDN2	reserved	BYPASS2	reserved				OSCSRC2			reserved			
Type	RO	RW	RW	RO	RW	RO	RO	RO	RO	RW	RW	RW	RO	RO	RO	RO
Reset	0	1	1	0	1	0	0	0	0	0	0	1	0	0	0	0

SYSCTL_RCC2_R Control Signals (Run-Mode Clock Configuration)



SYSCTL_RIS: Raw Interrupt Status (RIS)

*Datasheet: page 244

- This register indicates the status for system control raw interrupts.
- PLLLRIS: PLL Lock Raw Interrupt Status, bit 6,
  - 0: The PLL timer has not reached TREADY.
  - 1: The PLL timer has reached TREADY indicating that sufficient time has passed for the PLL to lock.

Raw Interrupt Status (RIS)

Base 0x400F.E000
Offset 0x050
Type RO, reset 0x0000.0000

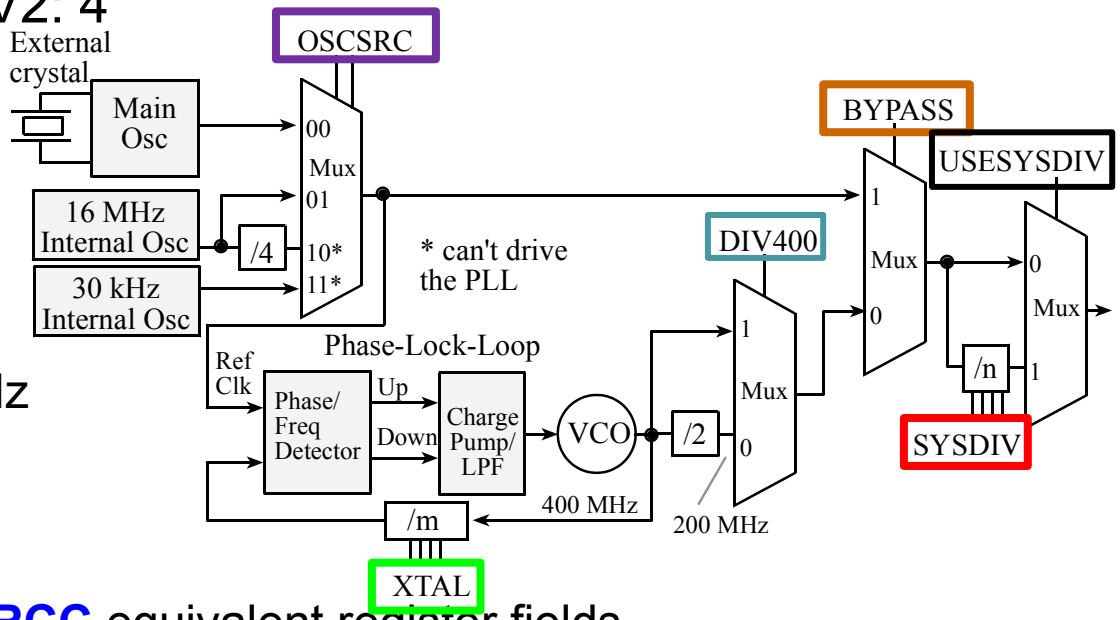
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	reserved															
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	reserved				BOR0RIS	VDDARIS	reserved	MOSCPUPRIS	USBPLLLRIS	PLLLRIS	reserved		MOFRIS	reserved	BOR1RIS	reserved
Type	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO	RO
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4.9b. Main clock registers for TM4C123

SYSCTL_RCC_R: XTAL: 10101, select 16MHz
SYSCTL_RCC2_R: SYSDIV2: 4
400MHz/(4+1)=80MHz

Example SYSDIV2 (n) values:
n=4 gives 400/(4+1) = 80 MHz
n=7 gives 400/(7+1) = 50MHz
n=9 gives 400/(9+1) = 40MHz
n=15 gives 400/(15+1) = 25MHz



- Use both **RCC** and **RCC2** for more options.
- **RCC2** register overrides the **RCC** equivalent register fields.
- **Important:** Write the **RCC** register prior to writing the **RCC2** register

Address	26-23	22	13	11	10-6	5-4	Name
$400FE060	SYSDIV	USESYSDIV	PWRDN	BYPASS	XTAL	OSCSRC	SYSCTL_RCC_R

	31	30	28-22	13	11	6-4	
$400FE070	USERCC2	DIV400	SYSDIV2	PWRDN2	BYPASS2	OSCSRC2	SYSCTL_RCC2_R

Program 4.6 PLL_Init()

0) Use RCC2 for more options.
1) The first step is to set BYPASS2 (bit 11) to bypass PLL.
2) The second step is to specify the crystal frequency in the four XTAL bits using the code in Table 4.9a. The OSCSRC2 bits are cleared to select the main oscillator as the oscillator clock source.
3) The third step is to clear PWRDN2 (bit 13) to put PLL to normal mode.
4) The fourth step is to configure and enable the clock divider using the 7-bit SYSDIV2 field(bits 22 to 28). If the 7-bit SYSDIV2 is **n**, then the clock will be divided by **n**+1. To get the desired 80 MHz from the 400 MHz PLL, we need to divide by 5. So, we place a 4 into the SYSDIV2 field.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
			1	1	1	1	1	1	1																						

5) The fifth step is to wait for the PLL to stabilize by waiting for PLLRIS (bit 6) in the **SYSCTL_RIS_R** to become high.
6) The last step is to activate the PLL by clearing the BYPASS2 bit.

Reading Materials

- Textbook: Chapter 4, Section 4.3
- TM4C123 Datasheet Chapter 5, Section 5.2.5, 5.5
- Example Project: SysTickPLL