Clock System

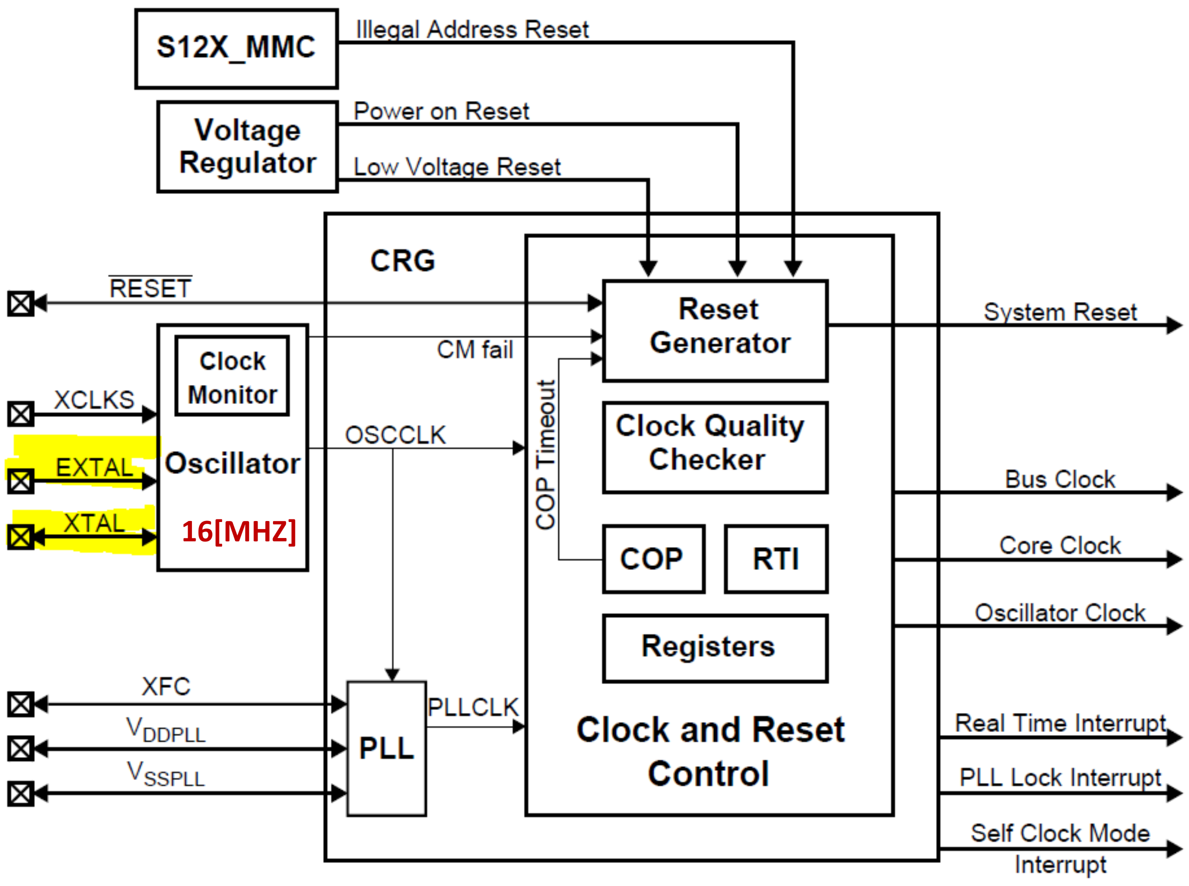


Figure 2-1. CRG Block Diagram

There are 2 possible ways of providing an oscillator input:

- **External Clock Input,** CMOS Compatible.
- **Crystal Oscillator**. This is the simplest approach and the one we use in our micro board.

XCLKS – This pin controls if wew either use a Loop Control Oscillator or Full Swing Oscillator / external clock.

$\overline{\text{XCLKS}}$	Description
1	Loop controlled Pierce oscillator selected
0	Full swing Pierce oscillator/external clock selected

EXTERNAL CLOCK INPUT

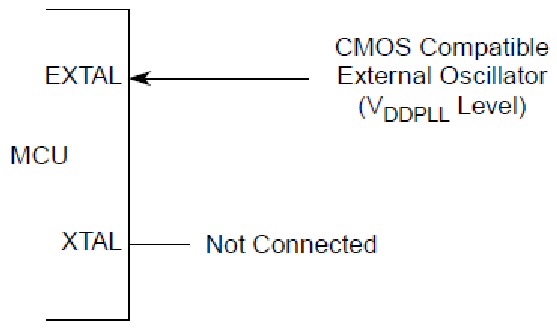


Figure 3-4. External Clock Connections ($\overline{\text{XCLKS}}$ = 0)

LOOP CONTROLLED OSCILLATOR

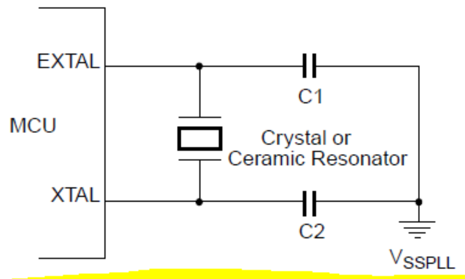
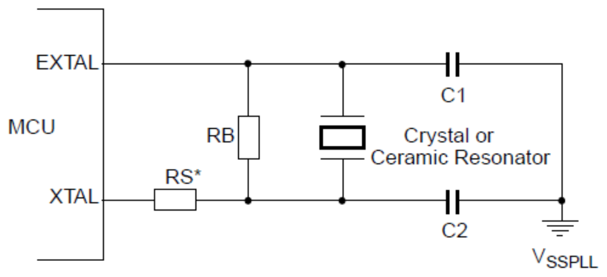


Figure 3-2. Loop Controlled Pierce Oscillator Connections ($\overline{\text{XCLKS}}$ = 1)

NOTE

Full swing Pierce circuit is not suited for overtone resonators and crystals without a careful component selection.



* R_S can be zero (shorted) when use with higher frequency crystals. Refer to manufacturer's data.

Figure 3-3. Full Swing Pierce Oscillator Connections ($\overline{\text{XCLKS}}$ = 0)

<u>XCLKS</u> pin is PE7 (Pin 36)

There are 2 ways of providing a clock to our system:

- Make **OSCCLK** the source of our bus clock in which case the clock will result in **OSCCLK / 2 .**

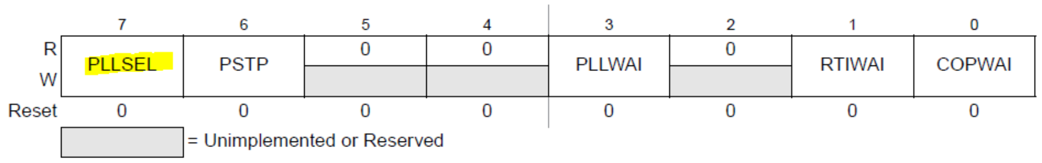
  Considering we are using a 16[MHZ] Crystal, the default bus clock is 8[MHZ]

- Make the **PLL** the source of our bus clock in which case the clock will result in **PLLCLK / 2 .**

When the PLL is selected, the PLLCLK can be determined according to the pollowing formula:

$$\text{PLLCLK} = 2\text{x}\text{OSCCLK}\text{x}\frac{(\text{SYNR}+1)}{(\text{REFDV}+1)}$$

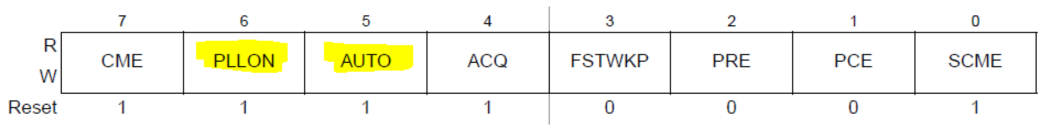
The registers to configure to use this option are:



	7	6	5	4	3	2	1	0
R	PLLSEL	PSTP	0	0	PLLWAI	0	RTIWAI	COPWAI
W			▨	▨		▨		
Reset	0	0	0	0	0	0	0	0

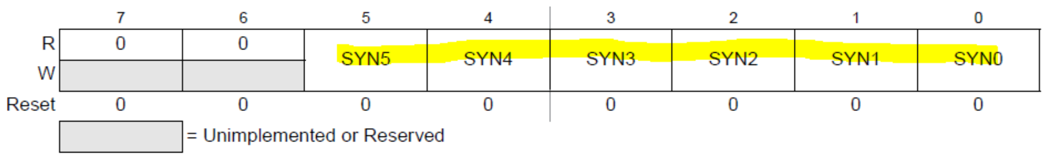
▨ = Unimplemented or Reserved

Figure 2-9. CRG Clock Select Register (CLKSEL)



	7	6	5	4	3	2	1	0
R/W	CME	PLLON	AUTO	ACQ	FSTWKP	PRE	PCE	SCME
Reset	1	1	1	1	0	0	0	1

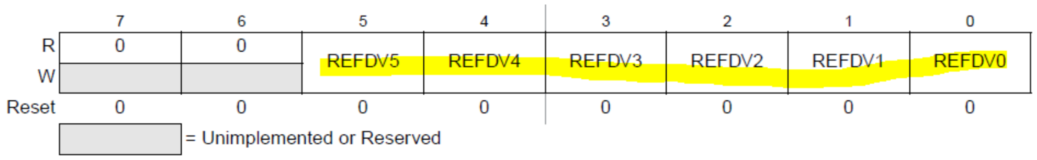
Figure 2-10. CRG PLL Control Register (PLLCTL)



	7	6	5	4	3	2	1	0
R	0	0	SYN5	SYN4	SYN3	SYN2	SYN1	SYN0
W	▨	▨						
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 2-4. CRG Synthesizer Register (SYNR)



	7	6	5	4	3	2	1	0
R	0	0	REFDV5	REFDV4	REFDV3	REFDV2	REFDV1	REFDV0
W	▨	▨						
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 2-5. CRG Reference Divider Register (REFDV)

It is also convinient to be able to check the actual clock output if an oscilloscope is available. To enable the clock output, we have to do it on this register:

	7	6	5	4	3	2	1	0	
R	NECLK	NCLKX2	0	0	0	0	EDIV1	EDIV0	
W			▨	▨	▨	▨			
Reset[1]	Mode Dependent	1	0	0	0	0	0	0	Mode
SS	0	1	0	0	0	0	0	0	Special Single-Chip
ES	1	1	0	0	0	0	0	0	Emulation Single-Chip
ST	0	1	0	0	0	0	0	0	Special Test
EX	0	1	0	0	0	0	0	0	Emulation Expanded
NS	1	1	0	0	0	0	0	0	Normal Single-Chip
NX	0	1	0	0	0	0	0	0	Normal Expanded

▨ = Unimplemented or Reserved

Figure 22-15. ECLK Control Register (ECLKCTL)

Table 22-17. Free-Running ECLK Clock Rate

EDIV[1:0]	Rate of Free-Running ECLK
00	ECLK = Bus clock rate
01	ECLK = Bus clock rate divided by 2
10	ECLK = Bus clock rate divided by 3
11	ECLK = Bus clock rate divided by 4

Configuring the clock to use the PLL: Steps

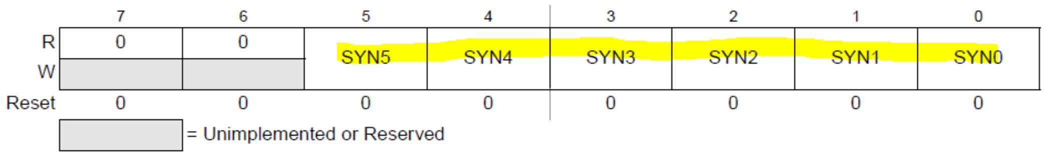
We are goint to configure our bus clock to 24[MHZ] as an example

$$BUS_{CK} = {}^{PLL_{CK}}/_{2}$$

So we nee to set the PLL to 48MHZ. We will use:

$$PLL_{CK} = \frac{2 \times 16[MHZ] \times 6}{4} = 48[MHZ]$$

1. Set the Synthesizer Register (SYNR)



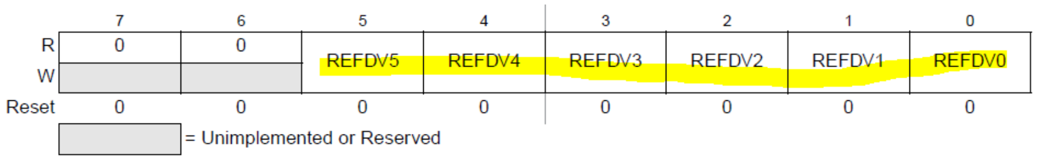
	7	6	5	4	3	2	1	0
R	0	0	SYN5	SYN4	SYN3	SYN2	SYN1	SYN0
W	▨	▨						
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 2-4. CRG Synthesizer Register (SYNR)

```
SYNR = 5; //0x5, set multiplier to: 5+1 = 6
```

2. Set the Reference Divider Register (REFDV)



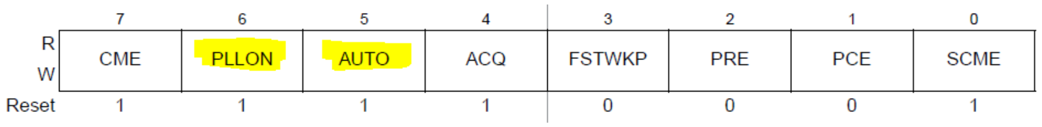
	7	6	5	4	3	2	1	0
R	0	0	REFDV5	REFDV4	REFDV3	REFDV2	REFDV1	REFDV0
W	▨	▨						
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 2-5. CRG Reference Divider Register (REFDV)

```
REFDV = 3; //0x3, set divider to: 3+1 = 4
```

3. Set the PLL Control Register (PLLCTL)



	7	6	5	4	3	2	1	0
R / W	CME	PLLON	AUTO	ACQ	FSTWKP	PRE	PCE	SCME
Reset	1	1	1	1	0	0	0	1

Figure 2-10. CRG PLL Control Register (PLLCTL)

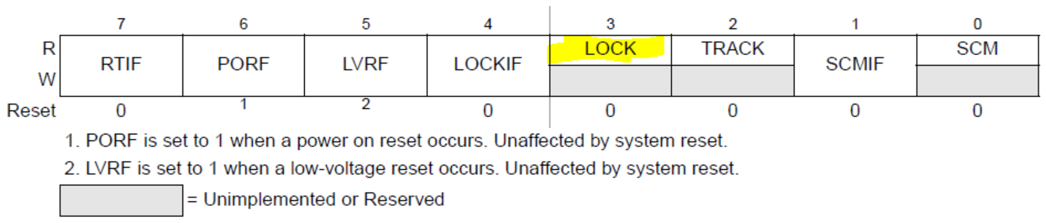
```
PLLCTL = PLLCTL_PLLON_MASK | PLLCTL_AUTO_MASK;//PLL ON and AUTO
```

Or

```
PLLCTL = 0b01100000;//PLL ON and AUTO
```

4. Now we have to wait for the PLL to lock into the prequency (very Important!)

This involves a check in the status register **(CRGFLG)**



	7	6	5	4	3	2	1	0
R	RTIF	PORF	LVRF	LOCKIF	LOCK	TRACK	SCMIF	SCM
W					▨	▨		▨
Reset	0	[1]	[2]	0	0	0	0	0

1. PORF is set to 1 when a power on reset occurs. Unaffected by system reset.
2. LVRF is set to 1 when a low-voltage reset occurs. Unaffected by system reset.

▨ = Unimplemented or Reserved

Figure 2-7. CRG Flags Register (CRGFLG)

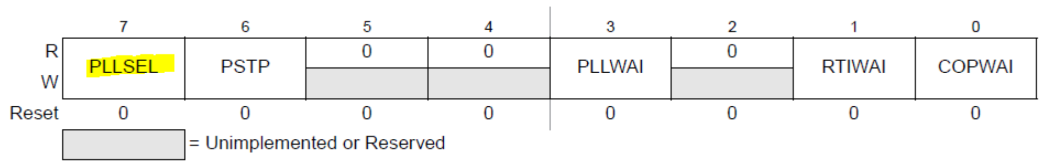
```
while(!(CRGFLG & CRGFLG_LOCK_MASK));//Wait for PLL to lock
```

Or

```
while(!(CRGFLG & 0b00001000));//Wait for PLL to lock
```

5. Select the PLL as clock source: Clock Select Register (CLKSEL)

This cannot be done before the PLL has locked into the prequency, as the system cannot get a clock source that is not yet stable



	7	6	5	4	3	2	1	0
R	PLLSEL	PSTP	0	0	PLLWAI	0	RTIWAI	COPWAI
W			▨	▨		▨		
Reset	0	0	0	0	0	0	0	0

▨ = Unimplemented or Reserved

Figure 2-9. CRG Clock Select Register (CLKSEL)

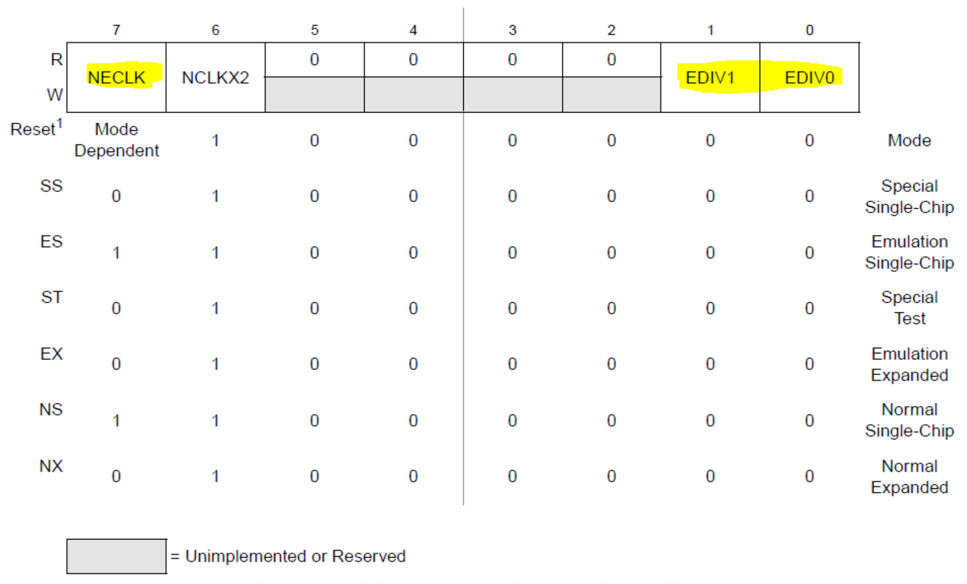
```
CLKSEL |= CLKSEL_PLLSEL_MASK;  //Select PLL as clock source
```

Or

```
CLKSEL |= 0b10000000;// Select PLL as clock source
```

6. Optional: Activate the clock output: ECLK Control Register (ECLKCTL)

If the clock output is activated, it is convenient to use a divider so the clock output is not the same as the bus clock, maybe for measuring such signal with an oscilloscope or any other means.



	7	6	5	4	3	2	1	0	
R	NECLK	NCLKX2	0	0	0	0	EDIV1	EDIV0	
W			▨	▨	▨	▨			
Reset[1]	Mode Dependent	1	0	0	0	0	0	0	Mode
SS	0	1	0	0	0	0	0	0	Special Single-Chip
ES	1	1	0	0	0	0	0	0	Emulation Single-Chip
ST	0	1	0	0	0	0	0	0	Special Test
EX	0	1	0	0	0	0	0	0	Emulation Expanded
NS	1	1	0	0	0	0	0	0	Normal Single-Chip
NX	0	1	0	0	0	0	0	0	Normal Expanded

▨ = Unimplemented or Reserved

Figure 22-15. ECLK Control Register (ECLKCTL)

```
  ECLKCTL &= ~ECLKCTL_NECLK_MASK; //Activate clock output
  ECLKCTL |= ECLKCTL_EDIV1_MASK | ECLKCTL_EDIV0_MASK;//clock divide by 4
```