Programming Notes MSP430F2618

# Core needed:

Clock

Timer

# Peripherals needed:

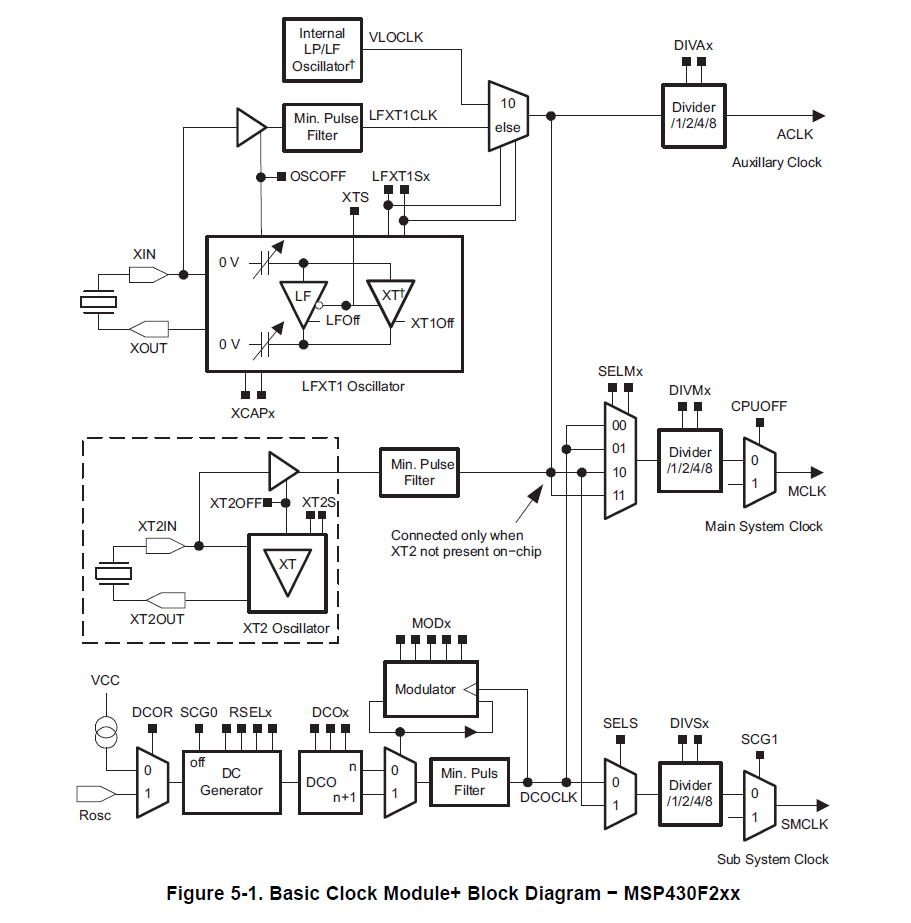
ADC

GPIO Interrupts

I2C

SPI

# Clock



2, 3 or 4 Clock sources available:

LFT1CLK: external low-/high-frequency oscillator. Can be used with 32768 crystals or standard crystals, resonators or external clock sources. Low frequency or 400kHz – 16Mhz.

XT2CLK: Optional. High frequency. Can be used with standard crystals, resonators or external clock sources. 400kHz – 16MHz.

DCOCLK: Internal Digitally Controlled Oscillator

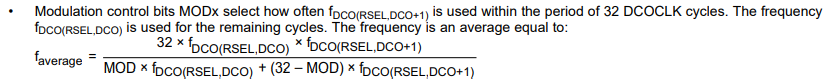
VLOCLK: Internal very low frequency oscillator. 12kHz

3 clock Signals available:

ACLK: Auxiliary clock. Software selectable as LFT1CLK, VLOCLK. Divided by 1, 2, 4 or 8

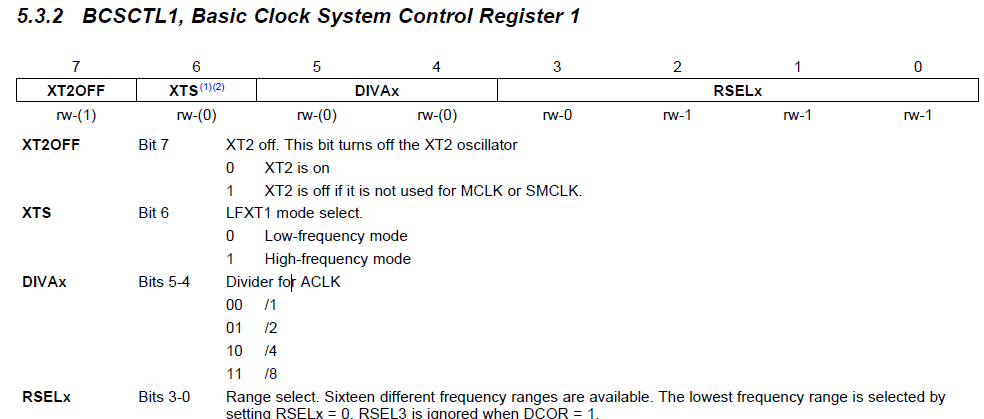
MCLK: Master clock. Software selectable as any. Divided by 1 ,2 ,4 or 8

SMCLK: Sub main clock. Software selectable as any. Divided by 1, 2, 4 or 8. Software selectable for individual peripheral modules.

MODx is used to fine adjust the frequency produced by the DCO. 

Set Main clock and Submain clock to DCO disabling LFTX1CLK and XT2CLK because no external Crystals or Resonators are available:

Basic Clock System Control Register 1 (BCSCTL1):

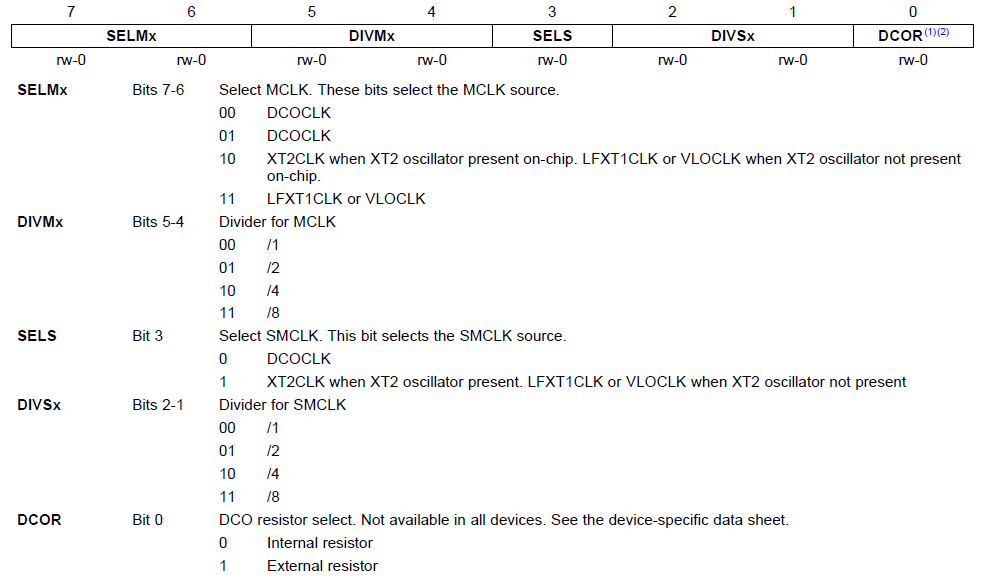


XT2OFF = 1

XTS = 0

RSELx -> See DCO configure later.

Basic Clock System Control Register 2 (BCSCTL2):

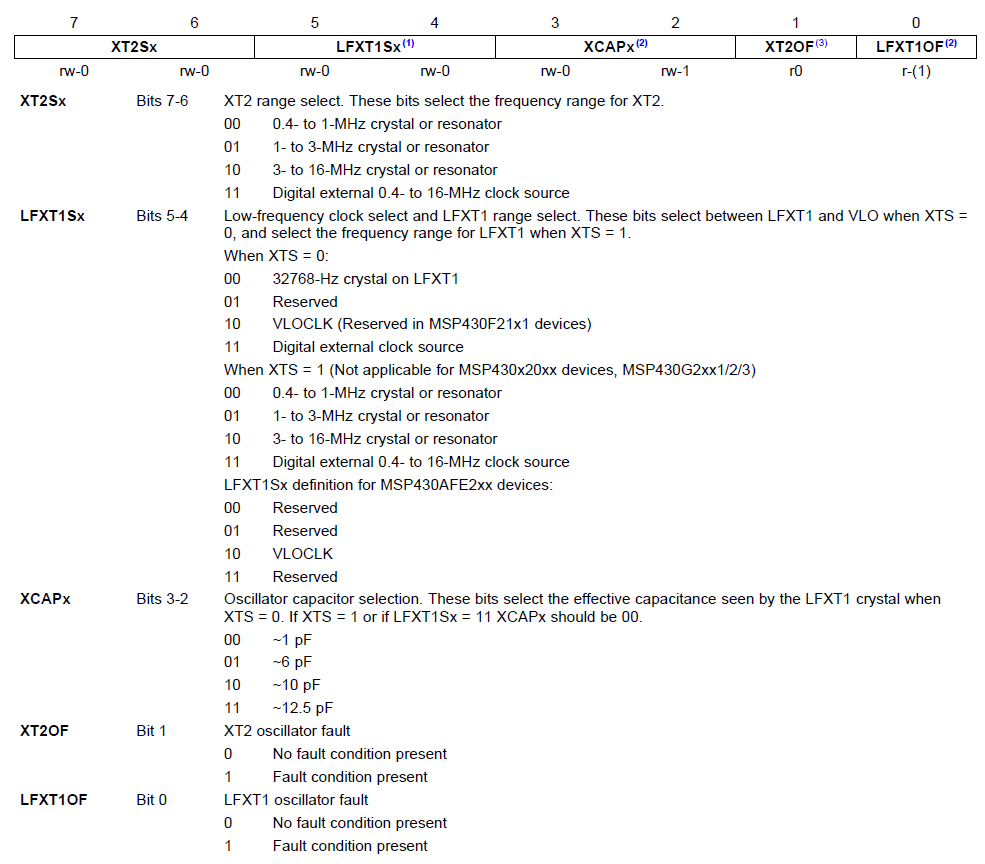


SELMx = 00

SELS = 0.

DCOR = 0

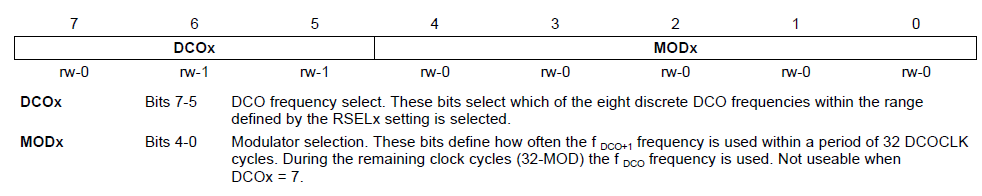
Basic Clock System Control Register 2 (BCSCTL2):



LFXT1Sx = 10

Programming the clock speed for the DCO

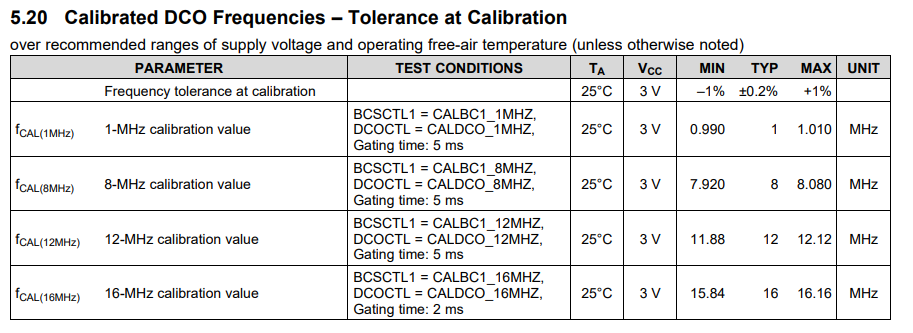
DCO Control Register (DCOCTRL):



DCO = 111

Also set RSLx to 1111

The DCO can also be calibrated by a pre-generated define



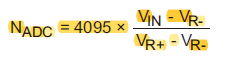
# ADC

The ADC core operates with an upper and a lower reference Voltage that can be defined by user.

If the Input Signal is equal or higher than the upper Voltage, the digital value is Full Scale (0x0FFF)

If the Input Signal is below the lower Voltage the digital value is low (0x0000)

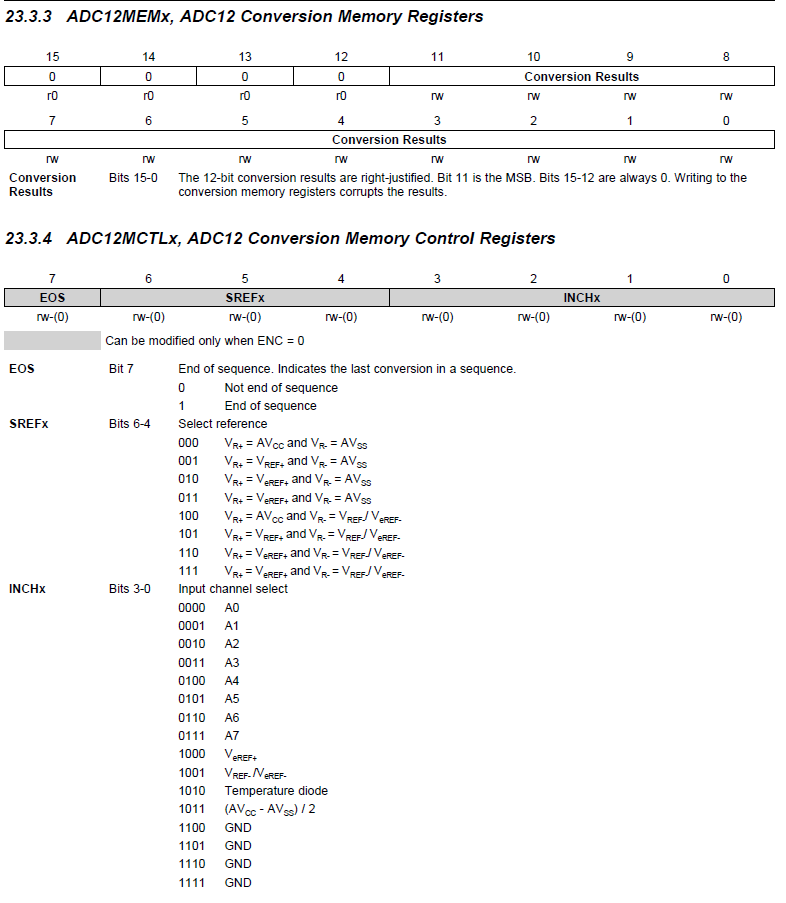
In between the Digital Value equals the calculated value NADC:

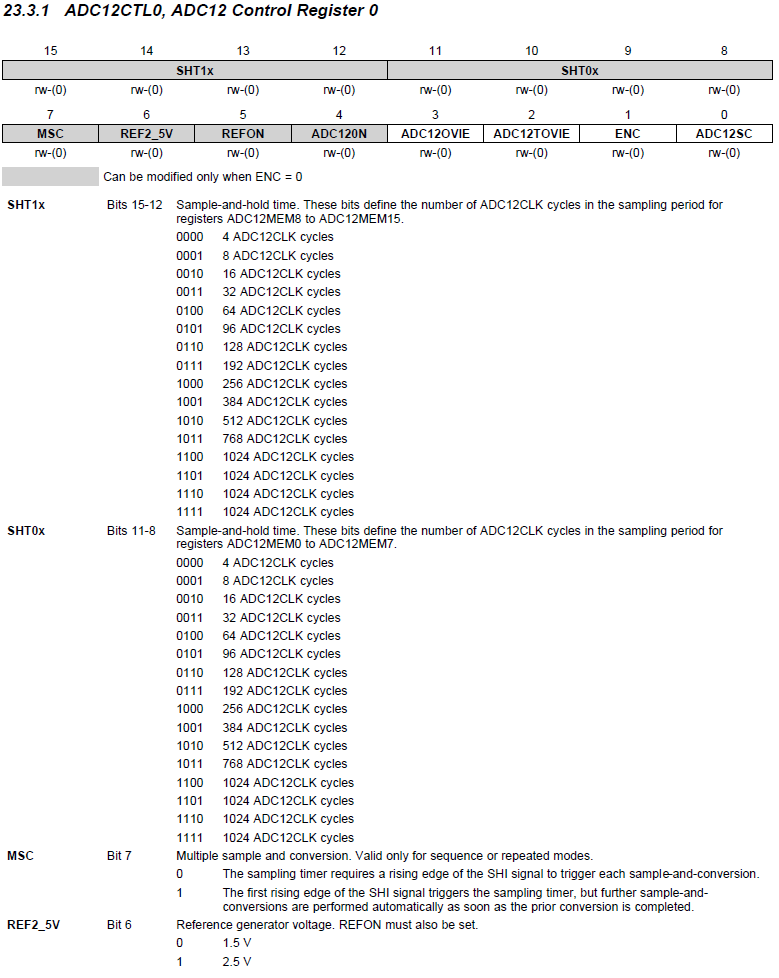


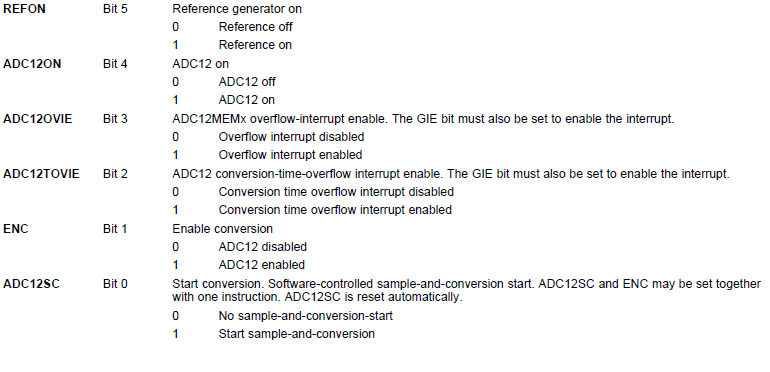
The ADC core is configured by only two registers: ADC12CTL0 and ADC12CTL1

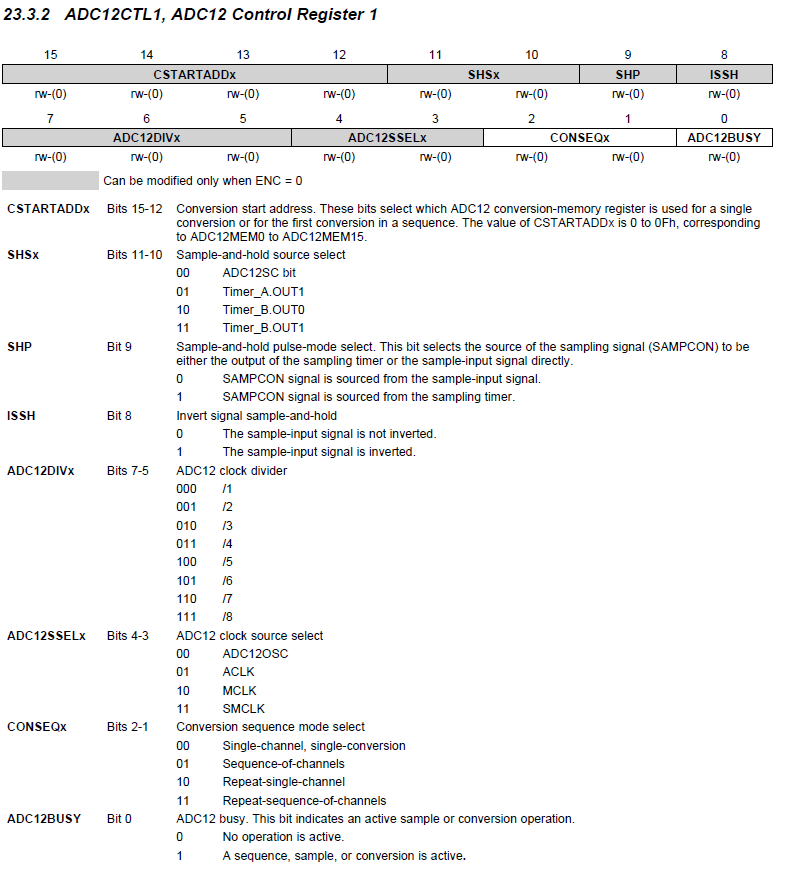
Its memory is stored in 16 registers therefore it can store up to 16-words of translation without any CPU intervention. The memory, that is to be saved, is controlled by another 16 Memory control registers. These decide what input channel is to be used and on what references the conversion has to be done.

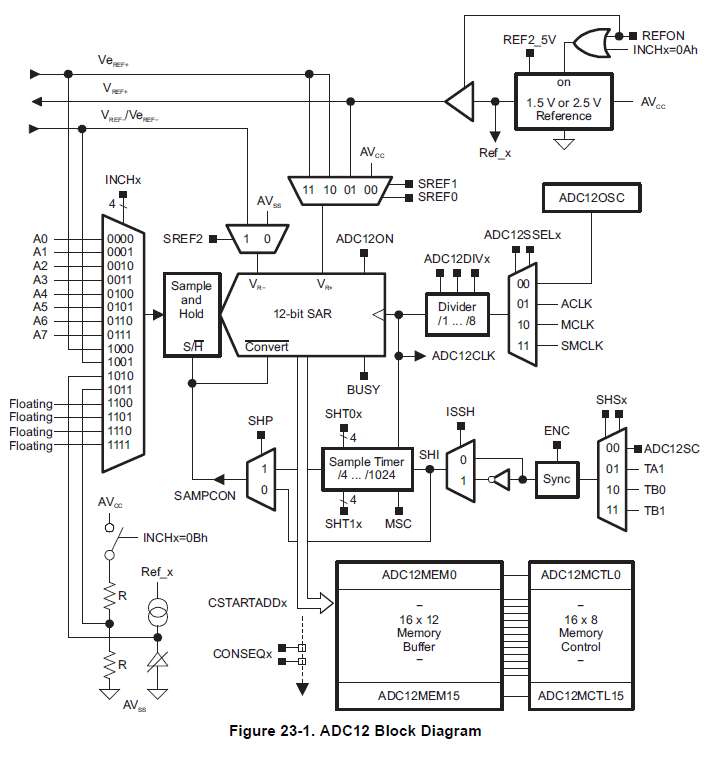
## Memory Register



Control Register





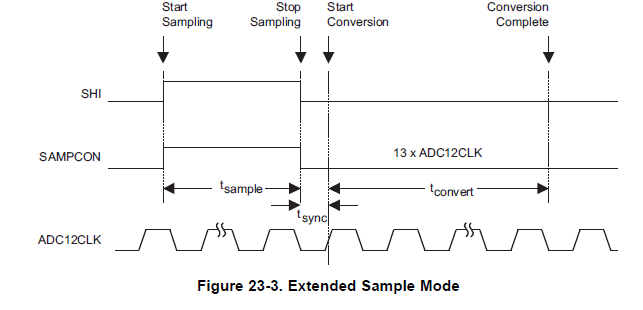


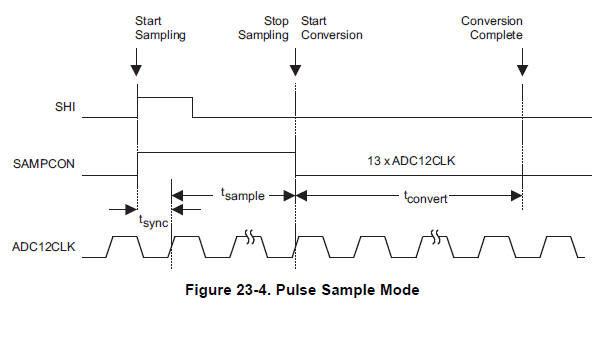
## Register explanations

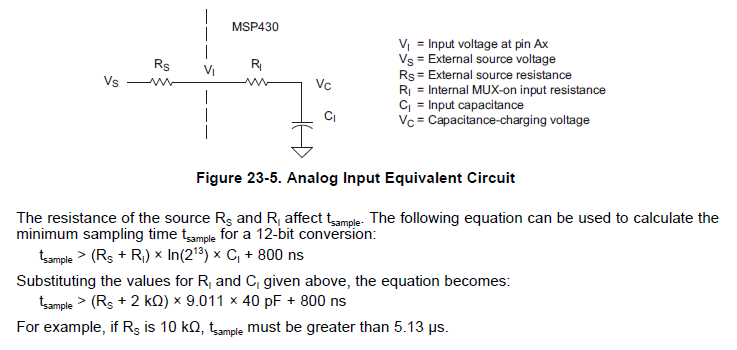
### SHP

This Bit decides between Extended Sample Mode and Pulse Sample Mode. In the Expanded Sample Mode, the conversion takes place as long as SHI is high. In the Pulse Sample Mode a pulse on SHI will trigger a Timer that will hold the sample go signal SAMPCON high as long as by the user defined in the SHT0 and SHT1 register. These two register will hold SAMPCON high for 4-1024 times the ADC12CLK. The 4.-Bit-SHT0 register controls sampling time for the ADC12MCTL 0-7 register and the 4-Bit-STH1 register for the ADC12MCTL 8-15 register.

SAMPCON = Sample Conversion

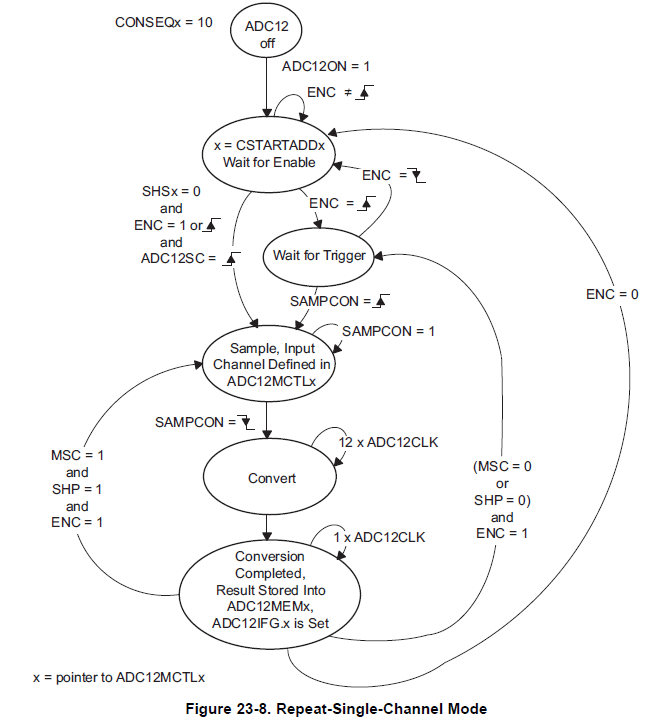






### Programming

For my use, the ADC will be initialized in the startup of the device and then will be triggered all 100m seconds by a Timer. When the conversion is complete, an interrupt will occur and the Controller can read the value and dim or brighten the LEDs.



ADC12CTL0

SHT0 = 256 ADC12CLK cycles (0x8) = 16us

ADC12ON = 1

ENC = 1

ADC12CTL1

SHSx = TIMER\_A (01)

SHP = 1

ADC12SSEL = MCLK (10)

CONSEQ = 10

ADC12MCTL0

Standard values

ADC12IE

ADC12IE0 = 1