2.

Three ways:

Method 1:

We set the **ADC0\_SSPRI\_R**register to 0x0123 to make sequencer 3 the highest priority

 we use software start (**EM3**=0x0)

The software writes an 8 (**SS3**) to the **ADC0\_PSSI\_R** to initiate a conversion on sequencer 3

Method 2:  
We perform the following steps to configure the ADC for software start on one channel. Program 14.1 shows a specific details for sampling PE4, which is channel 9. The function **ADC0\_InSeq3** will sample PE4 using software start and use busy-wait synchronization to wait for completion.

**Step 1.**We enable the port clock for the pin that we will be using for the ADC input.

**Step 2.**Make that pin an input by writing zero to the **DIR** register.

**Step 3.**Enable the alternative function on that pin by writing one to the **AFSEL** register.

**Step 4.**Disable the digital function on that pin by writing zero to the **DEN** register.

**Step 5.**Enable the analog function on that pin by writing one to the **AMSEL** register.

**Step 6.**We enable the ADC clock by setting bit 16 of the **SYSCTL\_RCGC0\_R** register.

**Step 7.**Bits 8 and 9 of the **SYSCTL\_RCGC0\_R** register specify the maximum sampling rate of the ADC. In this example, we will sample slower than 125 kHz, so the maximum sampling rate is set at 125 kHz. This will require less power and produce a longer sampling time, creating a more accurate conversion.

**Step 8.**We will set the priority of each of the four sequencers. In this case, we are using just one sequencer, so the priorities are irrelevant, except for the fact that no two sequencers should have the same priority.

**Step 9.**Before configuring the sequencer, we need to disable it. To disable sequencer 3, we write a 0 to bit 3 (**ASEN3**) in the **ADC\_ACTSS\_R**register. Disabling the sequencer during programming prevents erroneous execution if a trigger event were to occur during the configuration process.

**Step 10.**We configure the trigger event for the sample sequencer in the **ADC\_EMUX\_R**register. For this example, we write a 0000 to bits 15–12 (**EM3**) specifying software start mode for sequencer 3.

**Step 11.**Configure the corresponding input source in the **ADCSSMUXn**register. In this example, we write the channel number to bits 3–0 in the **ADC\_SSMUX3\_R** register. In this example, we sample channel 9, which is PE4.

**Step 12.**Configure the sample control bits in the corresponding nibble in the **ADC0SSCTLn**register. When programming the last nibble, ensure that the **END** bit is set. Failure to set the **END** bit causes unpredictable behavior. Sequencer 3 has only one sample, so we write a 0110 to the **ADC\_SSCTL3\_R** register. Bit 3 is the **TS0**bit, which we clear because we are not measuring temperature. Bit 2 is the **IE0** bit, which we set because we want to the **RIS** bit to be set when the sample is complete. Bit 1 is the **END0** bit, which is set because this is the last (and only) sample in the sequence. Bit 0 is the **D0** bit, which we clear because we do not wish to use differential mode.

**Step 13.**We enable the sample sequencer logic by writing a 1 to the corresponding **ASENn**. To enable sequencer 3, we write a 1 to bit 3 (**ASEN3**) in the **ADC\_ACTSS\_R**register.

Method 3:

**Step 1.**The ADC is started using the software trigger. The channel to sample was specified earlier in the initialization.

**Step 2.** The function waits for the ADC to complete by polling the RIS register bit 3.

**Step 3.**The 12-bit digital sample is read out of sequencer 3.

**Step 4.** The RIS bit is cleared by writing to the ISC register.

The ADC ISR runs when done and reads the sample.

3.Bypass capacitors help to dampen the AC or noise of the circuit. The key function of the bypass capacitor is to reduce the amount of ripple in a circuit.