* **Datasheet TM4C123GH6PM: Pg. 796**
  + **RCGC0:** pg. 455
    - Bits 8-9: Maximum sampling rate
  + **AMSEL**: pg. 684
    - Bits 0-3: Activates analog function for the pin
  + **SSPRI**: pg. 838
    - Sets the priority of the ADC sequencer
  + **EMUX**: pg. 830
    - Bits 12-15: Specify how the ADC will be triggered
  + **ACTSS**: pg 818
    - Enables/Disables sequencer
  + **SSMUX3**: pg. 872
    - Configures what channel to sample
  + **SSCTL3**: pg. 873
    - Specifies the mode of the ADC sample
* [**Successive approximation:**](https://www.youtube.com/watch?v=mAtri-cYSvk)
  + The **speed** of successive approximation related linearly with its precision in bits
* **Data Acquisition System:** sensor data converted to voltage, sent to the ADC and converted into a digital equivalent which is either sent for processing, logged, or used for a function.
  + **TM4C ADC**
    - **Grader uses ADC1 and PD3** for volt meter and oscilloscope
    - **Labs use ADC0, sequencer 3, and PE2** for labs 14 and 15
  + **ADC Parameters:**
    - **Precision**: 12bits = 4096 alternatives (points of measurements in a plot).
    - **Range**: The min to maximum voltage (0v to 3v3).
    - **Resolution**: Smallest change in voltage that can be reliably detected.
      * Resolution = Range / Precision
      * Resolution = 3v3 – 0v / 4096
      * Resolution = 0.7mV
  + **Limitations in DAQ sampling:**
    - Amplitude
    - Amplitude range
    - Time quantization
    - Time interval
  + **Figure of merit:**
    - Precision (number of bits)
    - Speed (how fast we can sample)
    - Power (how mucy energy does it take to operate)
  + Cost of the ADC is a function of the number and quality of the internal components

**ADC Initialization Ritual:** Example for setting up PortE as an analog input

* + Steps 1 - 5: GPIO PortE port configuration
    - Clock
    - DIR
    - ALT
    - DEN
    - AMSELECT
  + Step 6: ADC Clock
    - RCGC0 (bit 16)
  + Step 7: ADC sample speed
    - RCGC0 (bit 8-9)
  + Step 8: ADC Sequencer Priority
    - SSPRI (bit 12-13)
  + Step 9: Disable ADC during configuration
    - ACTSS (bit 3)
  + Step 10: Select ADC trigger type
    - EMUX (bit 12-15)
  + Step 11: Select ADC channel (0-11)
    - SSMUX (bit 0 – 3)
  + Step 12: Configure the sample control bits
    - SSCTL (bit 0 – 3)
  + Step 13: Enable ADC after configuration
    - ACTSS (bit 3)

We perform the following steps to configure the ADC for software start on one channel. Program 14.1 shows specific details for sampling PE2, which is channel 1. The function **ADC0\_InSeq3** will sample PE2 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 1, which is PE2.

**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.

**void ADC0\_InitSWTriggerSeq3\_Ch9(void){ volatile unsigned long delay;**  
**SYSCTL\_RCGC2\_R |= 0x00000010;   // 1) activate clock for Port E**  
**delay = SYSCTL\_RCGC2\_R;         //    allow time for clock to stabilize**  
**GPIO\_PORTE\_DIR\_R &= ~0x04;      // 2) make PE2 input**  
**GPIO\_PORTE\_AFSEL\_R |= 0x04;     // 3) enable alternate function on PE2**  
**GPIO\_PORTE\_DEN\_R &= ~0x04;      // 4) disable digital I/O on PE2**  
**GPIO\_PORTE\_AMSEL\_R |= 0x04;     // 5) enable analog function on PE2**  
**SYSCTL\_RCGC0\_R |= 0x00010000;   // 6) activate ADC0**  
**delay = SYSCTL\_RCGC2\_R;**  
**SYSCTL\_RCGC0\_R &= ~0x00000300;  // 7) configure for 125K**  
**ADC0\_SSPRI\_R = 0x0123;          // 8) Sequencer 3 is highest priority**  
**ADC0\_ACTSS\_R &= ~0x0008;        // 9) disable sample sequencer 3**  
**ADC0\_EMUX\_R &= ~0xF000;         // 10) seq3 is software trigger**  
**ADC0\_SSMUX3\_R &= ~0x000F;       // 11) clear SS3 field**  
**ADC0\_SSMUX3\_R += 1;             //    set channel Ain1 (PE2)**  
**ADC0\_SSCTL3\_R = 0x0006;         // 12) no TS0 D0, yes IE0 END0**  
**ADC0\_ACTSS\_R |= 0x0008;         // 13) enable sample sequencer 3**  
**}**  
  
*Program 14.1. Initialization of the ADC using software start and busy-wait (C14\_ADCSWTrigger).*

**ADC Sample Capture Procedure:**

