mTouch™ Sensing Solution Acquisition Methods Capacitive Voltage Divider

Author: *Burke Davison*
Microchip Technology Inc.

INTRODUCTION

Capacitive sensors are PIC® MCU pins connected to an area of conductive material through an optional series resistor. As the environment changes around the sensor, the capacitance of the conductive material relative to ground will change. While there are many methods for measuring the capacitance of the pin, most require special hardware or an advanced digital filtering system to achieve a clean signal.

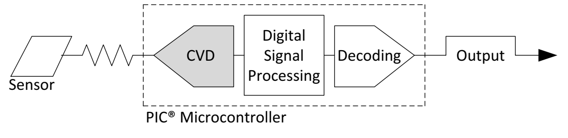
Microchip's differential Capacitive Voltage Divider (CVD) acquisition technique has been developed to require only an Analog-to-Digital Converter (ADC) and a minimal amount of digital processing overhead. This allows CVD to be implemented on the widest range of devices.

This application note will describe how the mTouch™ sensing solution CVD capacitive sensing method is implemented, analyze how its signal behaves in relation to changes in the environment, and define several optional performance enhancements to increase sensitivity and decrease noise on the output.

The code provided in this application note is for education purposes only. It is highly recommended to use the mTouch sensing Framework and Library provided in Microchip's Library of Applications (MLA, http://www.microchip.com/mla) for all real-world applications needing reliable noise rejection.

CVD is not the only technique available for measuring capacitance on a PIC device. Application notes on the Charge Time Measurement Unit alternative sensing method (AN1250, "*Microchip CTMU for Capacitive Touch Applications*"), as well as hardware and software design guidelines (AN1334, "*Techniques for Robust Touch Sensing Design*") are available on our web site at www.microchip.com/mTouch.

FIGURE 1: CVD SYSTEM OVERVIEW



BASIC CAPACITIVE TOUCH OVERVIEW

Capacitive sensors are most commonly created by placing an area of metal-fill on a printed circuit board, but can also be as simple as a piece of aluminum foil. This conductive pad is then connected to a PIC device through a thin trace and an optional series resistor. As shown in Figure 1, the PIC device will continuously poll the capacitance of the pad and watch for a significant shift to occur. The sensor is high-impedance during the measurement stage of the scan, and low-impedance in all other states.

The definition of "significant" depends on the level of noise. The shift must be appreciably higher than the noise level in the worst-case conditions. If the environment could change quickly for the application, the shift must be higher than the maximum possible change caused by the environment. So, the overall noise is a combination of high and low frequency disturbances, originating from a variety of possible sources.

For this reason, the quality of a capacitive sensor's signal should always be defined in terms of the signal-to-noise ratio (SNR) as defined in Equation 1, and not simply in terms of the signal change when activated.

EQUATION 1: SIGNAL-TO-NOISE RATIO

$$SNR = \frac{\mu}{\sigma}$$

μ is the amount of change when activated
σ is the standard deviation of the noise

This application note describes how the CVD measurement technique converts the sensor's capacitance to an integer value for digital processing. **However, Microchip does not recommend implementing CVD by hand!** The mTouch sensing Framework and mTouch sensing Library provided in the Microchip Library of Applications implements the scan automatically, and has been tested to provide a high level of noise immunity. It is highly recommended to use these resources rather than implementing the scan from scratch.

CAPACITIVE VOLTAGE DIVIDER OVERVIEW

CVD is a charge/voltage-based technique to measure relative capacitance on a pin using only the Analog-to-Digital Converter (ADC) module. Since its only requirement is a common PIC device peripheral, this technique can be implemented on the largest number of devices.

This technique performs a relative capacitive measurement based on the size of the internal ADC sample and hold capacitance. The electrical specifications of the PIC device will define the typical value of this capacitor; however, due to manufacturing tolerances this may vary by up to 20%. For this reason, it is not recommended to use CVD to produce an absolute measurement unless a calibration is performed and environmental conditions can be ensured not to change. Touch and proximity applications only require a relative measurement. This allows changes in the environment to be tracked and filtered out, and avoids the need for calibration.

Sensing Method Benefits

There are several reasons why the CVD technique performs well in real-world applications. These characteristics increase the reliability of the final touch decisions and minimize the cost of capacitive touch integration.

- Low Temperature Dependence

  A 1-3% signal offset change from -20°C to +60°C is typical. It is commonly removed in software by following slow changes in the value of the sensor.

- Low VDD Dependence

  The CVD waveform is not significantly dependent on VDD because both the sensor's charge and the ADC's positive reference use this same value. Because of this, low frequency changes in VDD are attenuated to a high degree. However, high-frequency disturbances in VDD may cause unwanted signal noise.

- Minimal Hardware Requirements

  An optional series resistor is recommended to reduce high-frequency noise on the signal. If noise is not a concern, no external components are necessary.

- Low-Frequency Noise Rejection

  Offsets caused by low-frequency noise will affect the two ADC samples of the CVD waveform in the same direction. However, increased capacitance will affect them in opposite directions. Subtracting the two samples will double the signal while simultaneously eliminating the noise offset.

Note:	The frequency range that is fundamentally rejected by the waveform will depend on the time delay between the two samples. The closer the samples are, the larger the bandwidth of noise rejection.

These benefits are not shared by every capacitive sensing technique, making CVD stand apart as an excellent choice for touch applications.

Theory of Operation

Assembly is the only recommended programming method for implementing CVD manually due to the importance of timing to the final SNR of the sensor. Microchip has provided libraries to implement the scan for you in the Microchip Libraries of Applications, and it is highly recommended to use this package rather than implementing the scan from scratch. This is available on our web site at www.microchip.com/mla. Assembly examples are also provided in this application note for several possible waveform configurations.

A capacitive sensor is connected to one of the PIC device's analog pins. An optional series resistor can be placed in the circuit to create a low-pass filter, attenuating high-frequency noise on the signal. The sampling will then be performed exclusively by manipulating the input/output ports and the ADC.

STEP 1: PRECHARGE THE CAPACITORS

Two capacitors are charged to opposite voltages. The first time this is performed is "Sample A". The second time (described in steps 4-6) is "Sample B". This is shown in Figure 4.

Sample A:

- External sensor discharged to VSS.
- Internal sensor charged to VDD.

Sample B:

- External sensor charged to VDD.
- Internal sensor discharged to VSS.

STEP 2: CONNECT CAPACITORS AND SETTLE

The two capacitors are connected in parallel and the charges are allowed to settle. As the external capacitance increases, so does its initial charge (Equation 2). The internal capacitance does not change, so its charge remains constant. This step is shown in Figure 5.

STEP 3: ADC CONVERSION

The final voltage on C_{hold} is determined by the size of the external capacitance in relation to the size of the internal capacitance (Equation 3).

STEPS 4-6: REVERSE THE PRECHARGE VOLTAGES AND REPEAT

The operation is then performed again, but this time the precharge voltages are reversed.

The difference between the two results is used as the current sensor reading. This is why the scanning technique is commonly called ‘differential CVD’. The complete waveform of the differential CVD sensing method is shown in Figure 2.

When a user approaches the capacitive sensor, the size of the external capacitance will increase with respect to the internal capacitance. This will result in a change in the settling voltages of the two samples.

For the Sample A, when the external sensor increases in capacitance and is discharged to VSS, the final settling voltage will decrease. For the Sample B, when the external sensor increases in capacitance and is charged to VDD, the final settling voltage will increase. Thus, as the external capacitance increases, the two settling points of the CVD waveform will diverge which causes a shift in the sensor reading.

Noise will cause a shift in the settled voltage based on the phase of the noise signal. For low-frequency noise, the phase will be approximately equivalent for both samples. Since the effect of the noise is roughly the same for both samples, we are able to significantly attenuate noise from our signal by taking the difference between the two settling points, causing the effect of the noise offset to cancel itself out. As the noise frequency increases, the quality of this noise rejection decreases and further filtering techniques become necessary. As the time difference between the two samples decreases, the ability to reject higher frequencies increases. This is one of the reasons why performing the scan in assembly (rather than C) can increase the noise performance of the sensing method.

FIGURE 2: DIFFERENTIAL CVD WAVEFORM

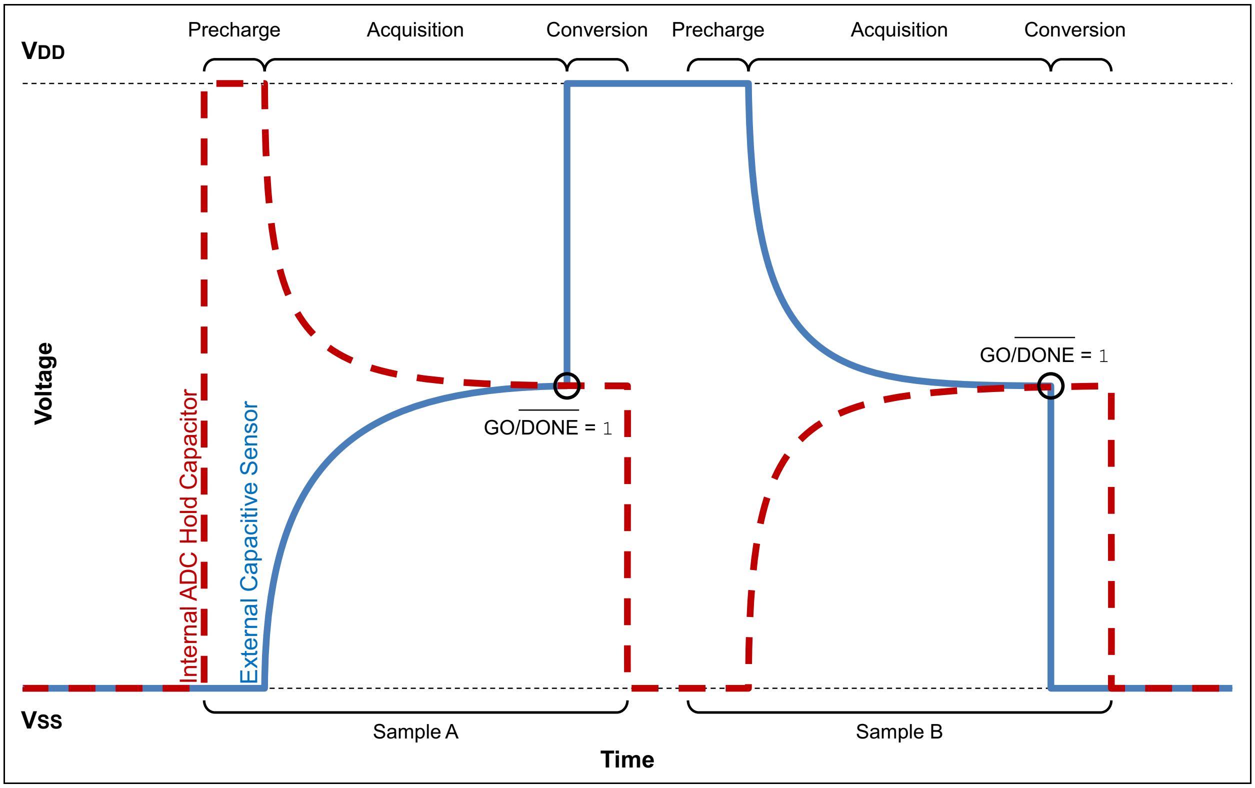
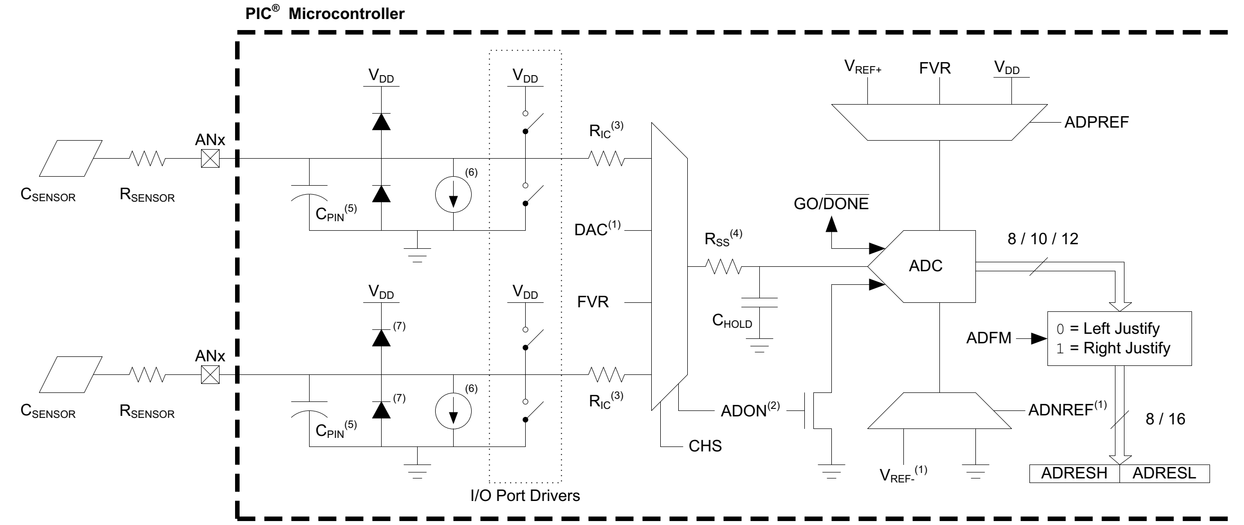


Figure 3 shows a generic overall system diagram for a capacitive touch design on a PIC device.

FIGURE 3: GENERIC PIC® DEVICE CAPACITIVE TOUCH SYSTEM DIAGRAM



NOTE
1: Not available on all PIC® microcontrollers.
2: When ADON = 0, all multiplexer inputs are disconnected.
3: Interconnect resistance of the pin. $R_{IC} \leq 1k\Omega$
4: Sampling switch resistance.
5: Input capacitance. $C_{PIN} \approx 5pF$
6: $I_{LEAKAGE}$ :: Refer to the Electrical Specifications chapter in the microcontroller's datasheet.
7: $V_T \approx 0.6V$

STEP-BY-STEP ANALYSIS

Precharge Stage

Both capacitors are charged to known, opposite voltage states, as defined in Equation 2. The internal ADC capacitance can be charged using either the drivers of an unused analog pin, another sensor's pin, or (if available as an ADC channel selection on the chosen PIC device) the Digital-to-Analog Converter.

EQUATION 2: PRECHARGE STAGE

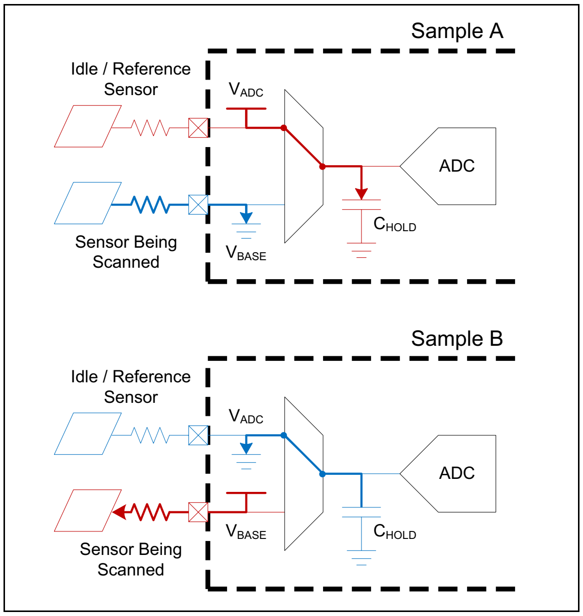
$$Q_{base} = C_{base}V_{base}$$
$$Q_{hold} = C_{hold}V_{hold}$$

Q_{base} is the total external charge.
Q_{hold} is the total internal charge.
V_{base} is the voltage provided to the external sensor during the Precharge stage.
V_{hold} is the voltage provided to the internal ADC hold capacitance during precharge.
C_{base} is the base external capacitance in the 'released' state
C_{hold} is the internal ADC capacitance for the sensor.

As shown in Figure 4, the value of V_{base} and VADC will alternate for Sample A and Sample B of the waveform. In both cases, one voltage value will be VSS and the other voltage will be VDD.

Note: Since VDD is usually considered constant for an application, most of the following math will divide by VDD to remove it from the equation. This is the same as assuming VDD is equal to '1V'.

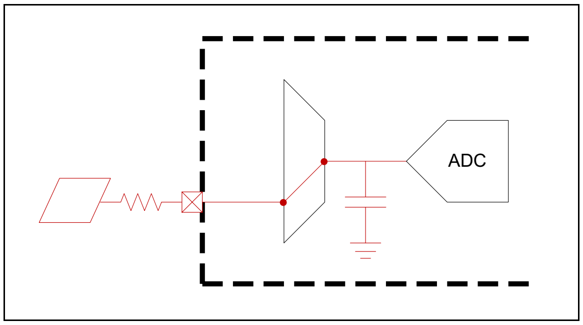
FIGURE 4: PRECHARGE STAGE, SAMPLE A AND SAMPLE B



Acquisition Stage

With the capacitors now charged to opposite voltage states, they are connected in parallel, as shown in Figure 5. This sums the charge across both capacitors as defined in Equation 3. The voltage across the capacitors will equalize by settling to a middle value based on the relationship of C_{hold} to the external capacitance.

FIGURE 5: ACQUISITION STAGE



Since the capacitors are now in parallel, we can combine their values to get the total capacitance on the circuit.

$$C_{total} = C_{hold} // C_{base} = C_{hold} + C_{base}$$

The total charge between the capacitors is the sum of the individual charges.

$$Q_{total} = C_{hold} V_{hold} + C_{base} V_{base}$$

EQUATION 3: ACQUISITION STAGE

$$V_{settle} = \frac{C_{hold} V_{hold} + C_{base} V_{base}}{C_{hold} + C_{base}}$$

Q_{total} is the total charge on both capacitors when connected during the Acquisition stage.

C_{total} is the total capacitance of the circuit during the Acquisition stage when both internal and external capacitors are connected and sharing charge.

V_{settle} is the final settling voltage during the Acquisition stage of a normal CVD waveform.

Differential Result

The above equation for the settling voltage is the generic form, true for both the first and second of the differential samples. The actual settling voltage for the first sample ('A') is calculated by substituting 0 for V_{base} and VDD for VADC. The second sample ('B') is calculated by substituting VDD for V_{base} and 0 for VADC.

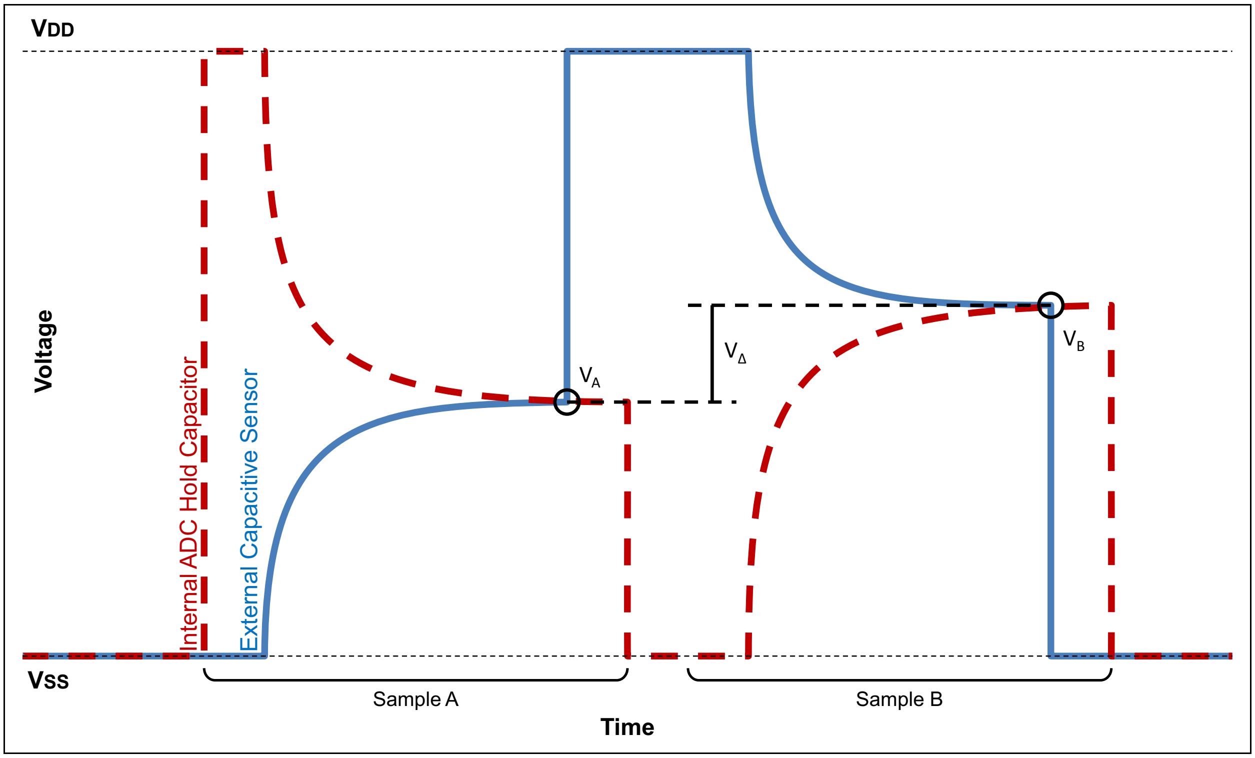
$$V_A = V_{settle}\{V_{hold} = V_{DD}, V_{base} = 0\}$$

$$V_B = V_{settle}\{V_{hold} = 0, V_{base} = V_{DD}\}$$

The reading for the sensor is then calculated by finding the difference between the two voltages. In practice, V_B is usually a higher value than V_A, so we adjust the order of the subtraction to generate a positive result.

$$V_{\Delta} = V_B - V_A$$

FIGURE 6: CVD DIFFERENTIAL RESULT



Note: In software, it's recommended to offset the value of V_B by 2^N (where N is the number of bits in the ADC) to further ensure a negative result is never achieved. For these calculations, this offset has been ignored to simplify the math.

$$V_{\Delta released} = V_{settle}(V_{hold} = 0, V_{base} = V_{DD}) - V_{settle}(V_{hold} = V_{DD}, V_{base} = 0)$$

$$\frac{V_{\Delta released}}{V_{DD}} = \frac{C_{base}}{C_{hold} + C_{base}} + \frac{C_{hold}}{C_{hold} + C_{base}}$$

EQUATION 4: DIFFERENTIAL RESULT

Difference in Voltage Between the Two CVD Settling Points

$$\frac{V_{\Delta released}}{V_{DD}} = \frac{C_{base} - C_{hold}}{C_{hold} + C_{base}}$$

Adding Finger Capacitance

Now, perform the same analysis but with an additional capacitor in the circuit: the user's finger. This has the effect of changing the external capacitance and the total capacitance.

$$C_{external, pressed} = C_{base} + C_{finger}$$
$$C_{total, pressed} = C_{hold} + C_{base} + C_{finger}$$

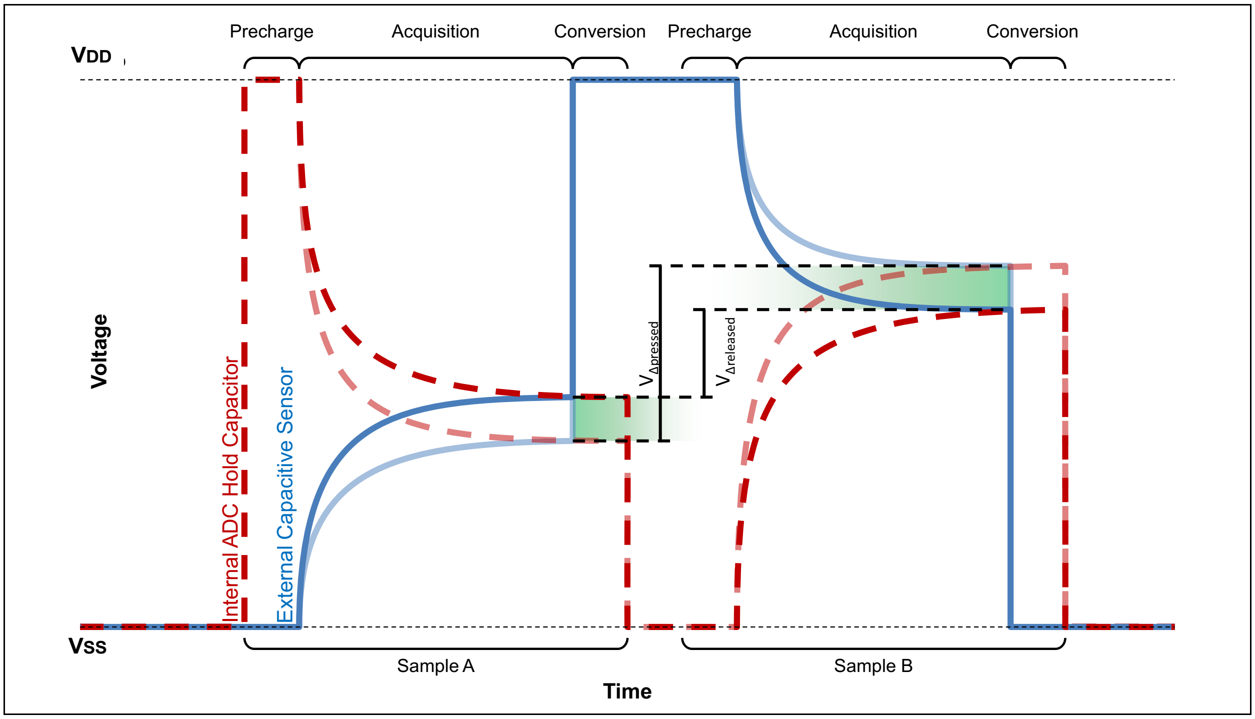
Calculate the general form equation for the CVD settling voltage when a finger is present on the sensor:

$$V_{settle, pressed} = \frac{(C_{base} + C_{finger})V_{base} + C_{hold}V_{hold}}{C_{hold} + C_{base} + C_{finger}}$$

Use the equation for V_Δ to calculate the settling point differential for the CVD waveform when a finger is present on the sensor:

$$\frac{V_{\Delta released}}{V_{DD}} = \frac{(C_{base} + C_{finger}) - C_{hold}}{C_{hold} + C_{base} + C_{finger}}$$

FIGURE 7: CVD SIGNAL – DIFFERENCE BETWEEN PRESSED AND RELEASED DIFFERENTIAL VALUES



Timing Considerations

PRECHARGE DELAY

Definition: the amount of time spent charging the internal and external capacitors.

This delay does not have a significant impact on the noise performance of the system. However, if the CVD implementation is using other sensors as the reference voltage source to the internal hold capacitor, and either the sensor or its reference has a large time constant, it's possible the default delay will not provide enough time for the external reference source to fully charge to VDD before exiting the Precharge stage.

If both capacitors are not charged completely, the sensors' signals will be corrupted. The circuit no longer has a known charge prior to entering the Acquisition stage. Instead, the charge is now dependent on the capacitance. This is an inoperable mode for the CVD scanning method and should be avoided. The precharge delay should be increased until there is no doubt that a full charge will occur before every sample.

Looking at the waveform on an oscilloscope will add capacitance to the sensor equivalent to that of a very heavy press. If the oscilloscope shows the sensor is not fully charging before entering the Acquisition stage, the precharge time should be increased.

To increase the precharge time in the code examples provided in this application note, add more `NOP`s to the line after the comment "Optional additional and/or variable delay" and before the Acquisition stage.

To increase the precharge time in other code libraries or frameworks, look for the advanced waveform settings and find the precharge (sometimes called the C_{hold}-Charge delay) setting.

ACQUISITION/SETTLING DELAY

Definition: the amount of time spent allowing the capacitors to equalize to a median voltage after being precharged to opposite states.

During the entire Acquisition stage of the CVD waveform until the ADC conversion has begun, the sensor will be set to an input (TRIS=`1`) which means it will have a high-impedance. Any low-impedance source near the sensor will be able to affect the settled charge by either discharging or charging it further. In other words, this is the noise susceptible period of our waveform. For this reason, the time allowed for this stage should be minimized.

There is a trade-off to be considered when deciding on the amount of settling delay time. If the settling delay is too small, the charge across both capacitors will not fully settle to an equalized value. This will reduce the amount of sensitivity when additional external capacitance is added to the circuit. However, if the settling delay is too large, noise will be able to couple in to the sensor and corrupt the final voltage. In general, the settling delay should be set to the minimum amount of time that still provides at least 90-95% of the fully-settled sensitivity. Specific applications may require a longer settling delay (for example, if there's a large external capacitance/resistance) or a shorter settling delay (for example, if noise in the design is a known problem).

DIFFERENTIAL DELAY

The amount of time between Sample A and Sample B.

The differential delay should be minimized in order to maximize the low-frequency noise rejection of the waveform.

Adding a small randomization to the time between the two samples will help attenuate noise in the lower kHz range but is not mandatory.

SAMPLING DELAY

Definition: the amount of time between Sample A and the next Sample A. In other words, the amount of time between CVD waveforms.

The sampling delay should be randomized to attenuate noise frequencies that are harmonics of the sampling rate of the sensor.

CVD WITH TWO ACQUISITION STAGES

The ideal settling voltage for a normal CVD waveform on both Sample A and Sample B is ½*VDD. This provides the largest separation between the settled voltage and the voltage rails, which maximizes its robustness against clipping due to noise. In order for the settling voltages to be near ½*VDD, the internal and external capacitors must be roughly equivalent.

If the internal capacitor is much larger than the external capacitor (or vice versa), the settling voltage will be proportionally dominated by the larger capacitor's starting value. This results in a less-than-ideal level of sensitivity.

To solve this problem, two Acquisition stages can be chained together to force the final voltage closer to ½*VDD.

If the external capacitor is much larger than the internal capacitor:

- Perform the normal CVD Precharge and Acquisition stages. (Equation 3)
- Maintain the voltage on the external capacitor while simultaneously re-charging the internal capacitor.
- Perform another Acquisition stage. (Equation 6)

When viewing the sensor's waveform on a oscilloscope, this process has the appearance of doubling the initial settled voltage from the first Acquisition stage as shown in Figure 8. For this reason, we call this the "Double-CVD Waveform."

$$V_{total}^{double} = C_{base}V_{settle}^{normal} + C_{hold}V_{hold}$$

$$V_{total}^{double} = \frac{C_{base}V_{settle}^{normal} + C_{hold}V_{hold}}{C_{base} + C_{hold}}$$

FIGURE 8: DOUBLE-CVD WAVEFORM

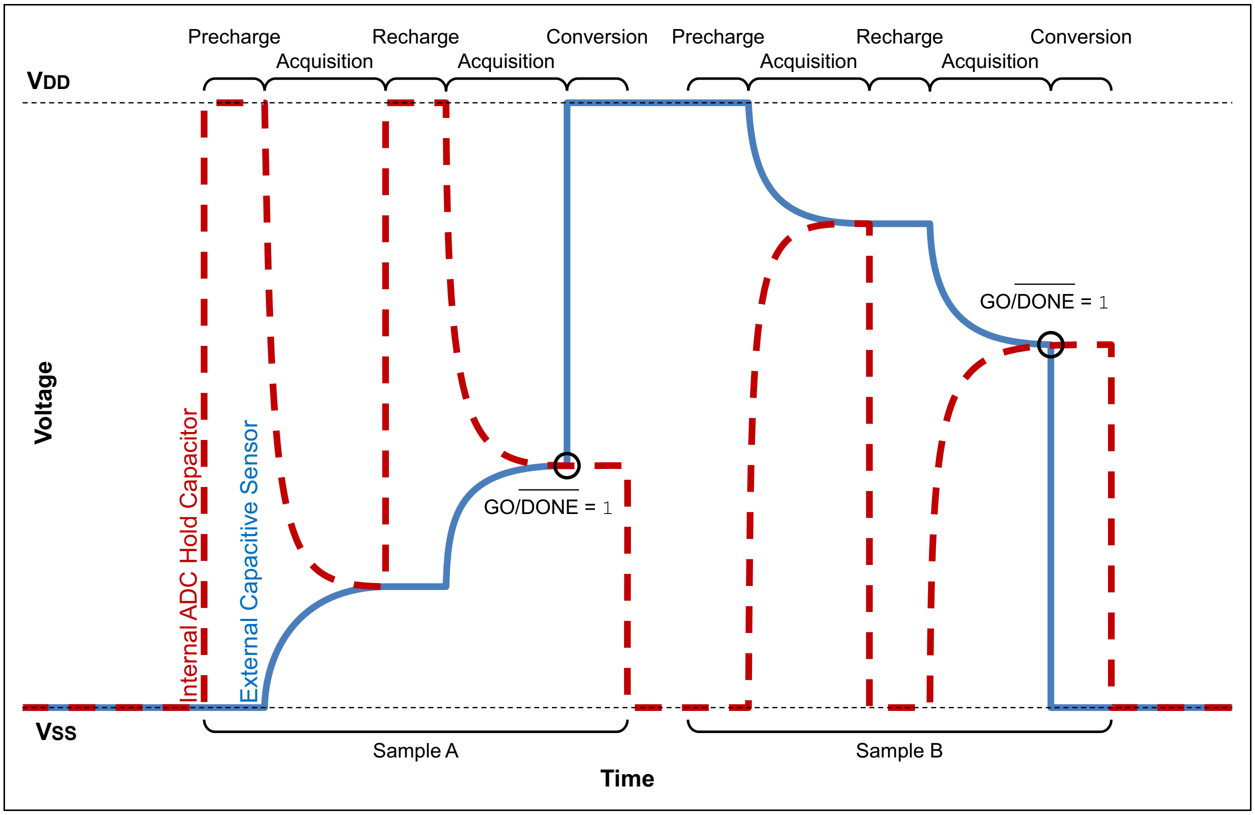
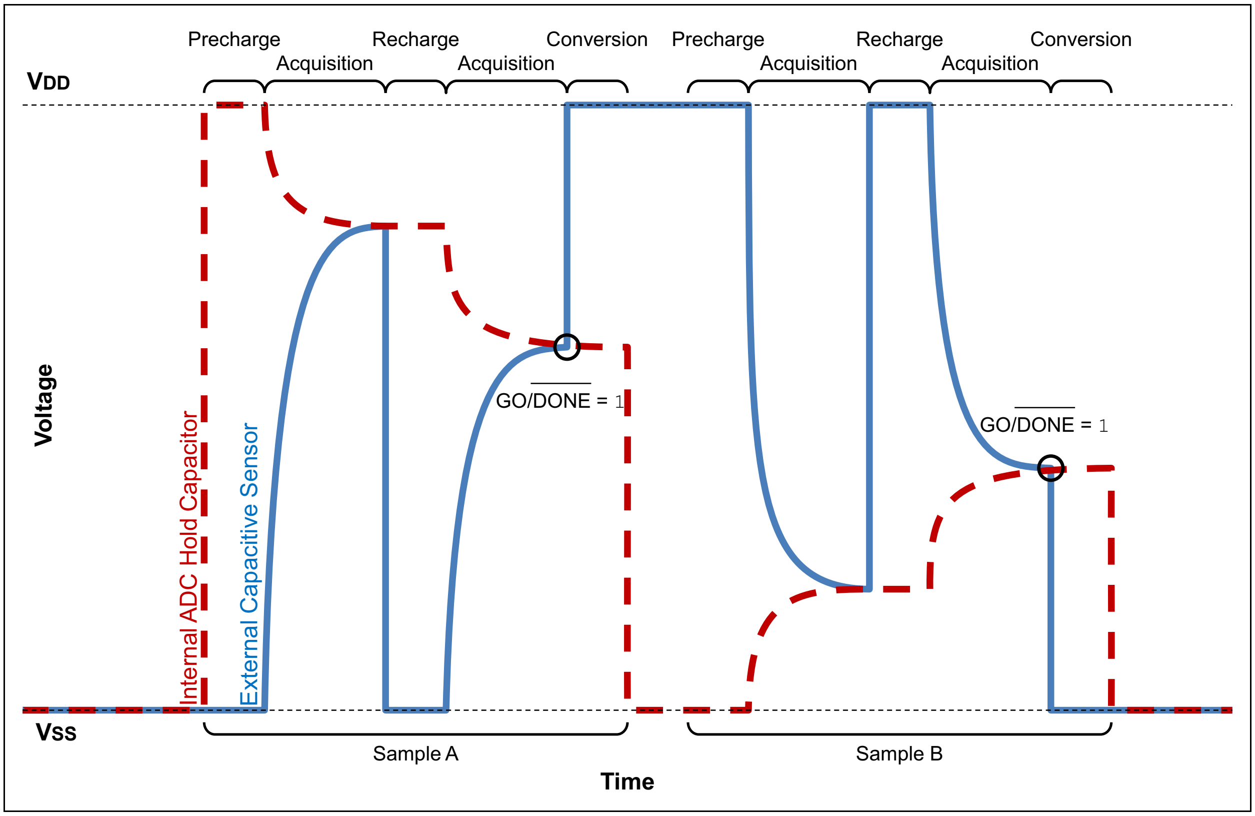


FIGURE 9: HALF-CVD WAVEFORM



If the internal capacitor is much larger than the external capacitor:

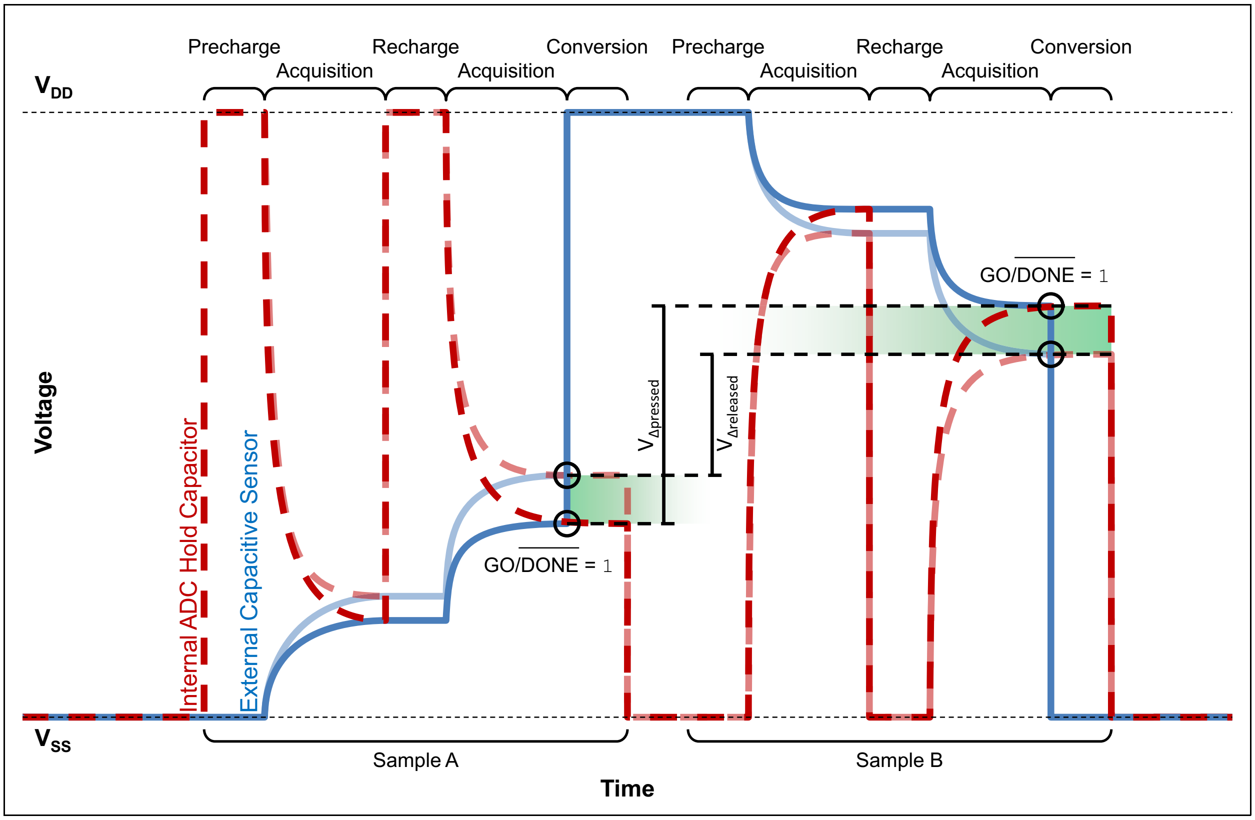
- Perform the normal CVD Precharge and Acquisition stages. (Equation 3)
- Maintain the voltage on the internal capacitor while simultaneously re-charging the external capacitor.
- Perform another Acquisition stage. (Equation 7)

When viewing the sensor's waveform on an oscilloscope, this process has the appearance of halving the initial settled voltage from the first Acquisition stage as shown in Figure 9. For this reason, we call this the "Half-CVD Waveform."

$$Q^{half}_{total} = C_{base}V_{base} + C_{hold}V^{normal}_{settle}$$

$$V^{half}_{total} = \frac{C_{base}V_{sensor} + C_{hold}V^{normal}_{settle}}{C_{base} + C_{hold}}$$

FIGURE 10: DOUBLE-CVD WAVEFORM DIFFERENTIAL RESULT



Substituting the 'normal' settling voltage with Equation 3 and simplifying, the final settling voltages for the double and half waveforms are:

EQUATION 6: SETTLING VOLTAGE FOR THE DOUBLE-CVD WAVEFORM

$$V_{total}^{double} = \frac{C_{hold}^2(V_{base} - V_{hold})}{(C_{base} + C_{hold})^2} + \frac{2C_{hold}(V_{hold} - V_{base})}{C_{base} + C_{hold}} + V_{base}$$

V_{total}^{double} is the final settling voltage for a Double-CVD waveform.

EQUATION 7: SETTLING VOLTAGE FOR THE HALF-CVD WAVEFORM

$$V_{total}^{half} = \frac{C_{hold}^2(V_{hold} - V_{base})}{(C_{base} + C_{hold})^2} + V_{base}$$

V_{settle}^{half} is the final settling voltage for a Half-CVD waveform.

Dual Acquisition-Stage Differential Results

The above equations for the settling voltages are the generic form, true for both the first and second of the differential samples. The actual settling voltage for the first sample (‘A’) is calculated by substituting 0 for V_{base} and VDD for VADC. The second sample (‘B’) is calculated by substituting VDD for V_{base} and 0 for VADC.

$$V_A = V_{settle}\{V_{hold} = VDD, V_{base} = 0\}$$

$$V_B = V_{settle}\{V_{hold} = 0, V_{base} = VDD\}$$

The reading for the sensor is then calculated by finding the difference between the two voltages. In practice, V_B is usually a higher value than V_A, so we adjust the order of the subtraction to generate a positive result. This math is illustrated in Figure 10

$$V_\Delta = V_B - V_A$$

EQUATION 8: DIFFERENCE IN VOLTAGE BETWEEN THE TWO DOUBLE-CVD SETTLING POINTS

$$\frac{V^{double}_{\Delta released}}{VDD} = \frac{C^2_{base} - 2C_{hold}C_{base} - C^2_{hold}}{(C_{base} + C_{hold})^2}$$

EQUATION 9: DIFFERENCE IN VOLTAGE BETWEEN THE TWO HALF-CVD SETTLING POINTS

$$\frac{V^{half}_{\Delta released}}{VDD} = \frac{C^2_{base} + 2C_{hold}C_{base} - C^2_{hold}}{(C_{base} + C_{hold})^2}$$

Adding Finger Capacitance

Now, perform the same analysis but with an additional capacitor in the circuit: the user’s finger. This has the effect of changing the external capacitance and the total capacitance.

$$C_{external, pressed} = C_{base} + C_{finger}$$

$$C_{total, pressed} = C_{hold} + C_{base} + C_{finger}$$

The difference in voltage between the two Double-CVD settling points when pressed:

$$\frac{V^{double}_{\Delta pressed}}{VDD} = \frac{(C_{base} + C_{finger})^2 - 2C_{hold}(C_{base} + C_{finger}) - C^2_{hold}}{(C_{hold} + C_{base} + C_{finger})^2}$$

The difference in voltage between the two Half-CVD settling points when pressed:

$$\frac{V^{half}_{\Delta pressed}}{VDD} = \frac{(C_{base} + C_{finger})^2 + 2C_{hold}(C_{base} + C_{finger}) - C^2_{hold}}{(C_{hold} + C_{base} + C_{finger})^2}$$

Finally, calculate the total CVD signal by subtracting the unpressed differential from the pressed differential for both types. This is the amount of change in the sensor reading due to the finger being added to the circuit. Equation 10 and Equation 11 contain the values we should design to maximize.

EQUATION 10: PRESSED DIFFERENTIAL VALUE MINUS RELEASED DIFFERENTIAL VALUE FOR BOTH THE DOUBLE-CVD SCANNING METHOD

$$\frac{CVD_{double}}{VDD} = \frac{2C_{hold}C_{finger}[2C_{base}(C_{hold} + C_{base} + C_{finger}) + C_{hold}C_{finger}]}{(C_{base} + C_{hold})^2(C_{finger} + C_{base} + C_{hold})^2}$$

EQUATION 11: PRESSED DIFFERENTIAL VALUE MINUS RELEASED DIFFERENTIAL VALUE FOR BOTH THE HALF-CVD SCANNING METHOD

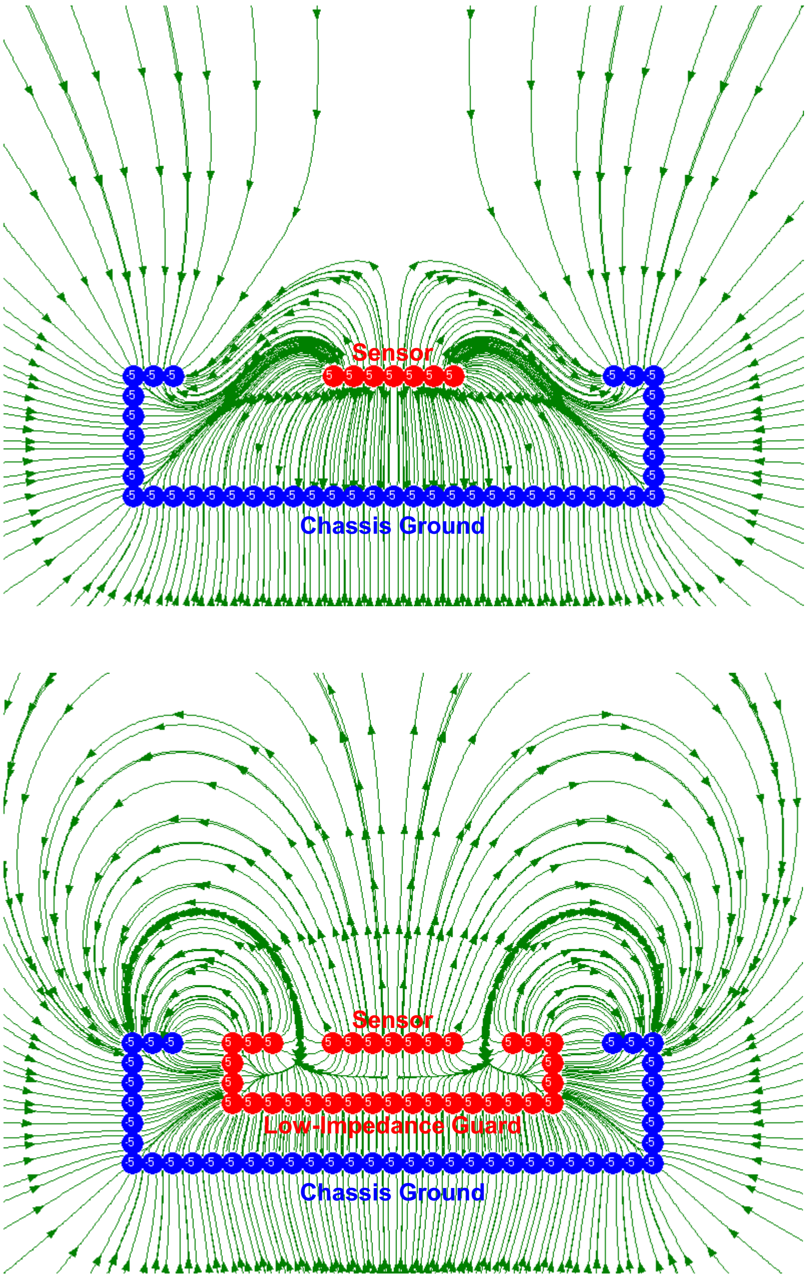
$$\frac{CVD_{half}}{VDD} = \frac{2C^2_{hold}C_{finger}(2C_{base} + 2C_{hold} + C_{finger})}{(C_{base} + C_{hold})^2(C_{finger} + C_{base} + C_{hold})^2}$$

CVD SCANNING WITH ACTIVE GUARD

The CVD scan is measuring total capacitance, so as the base capacitance of a sensor decreases, the change in signal caused by a user's finger will increase. Active guards are a way of minimizing the base capacitance by reducing the electric potential between the sensor and its surrounding environment.

When designing the guard for an application, the best solution is to encircle the sensor and its trace completely. If this is not possible, particular care should be taken at any point where a low-impedance source comes near to the sensor. Communication lines, motor drive lines, ground planes, and power planes are examples of traces that should be kept away from capacitive sensors. The guard trace should be placed between the sensor and these sources. However, you do not want to place the guard between the user and the sensor, as this will shield the sensor from the effect of the approaching finger.

FIGURE 11: ELECTRIC FIELD LINES WITH AND WITHOUT A GUARD TRACE



Guard Trace Design Guidelines

When designing a guard, here are some guidelines to keep in mind:

One guard trace can be used for all sensors. Sensors are scanned sequentially, so the guard can be actively driven for the sensor being currently scanned without affecting the others.

Any power planes or low-impedance traces should be guarded from the sensor.

Around the sensor's pad, the guard's trace should be about 1 mm thick and separated from the sensor by 2-3 mm.

Following the sensor's trace back to the PIC device's pin, the guard's trace can be same thickness as the sensor's trace: 0.1-0.3 mm. The separation of the guard trace from the sensor trace can be as small as 0.5 mm.

The standard method for guarding involves using a unity-gain amplifier to buffer the sensor's waveform onto a surrounding trace. This is more expensive than necessary since there are two available options for driving the guard signal that do not require external components.

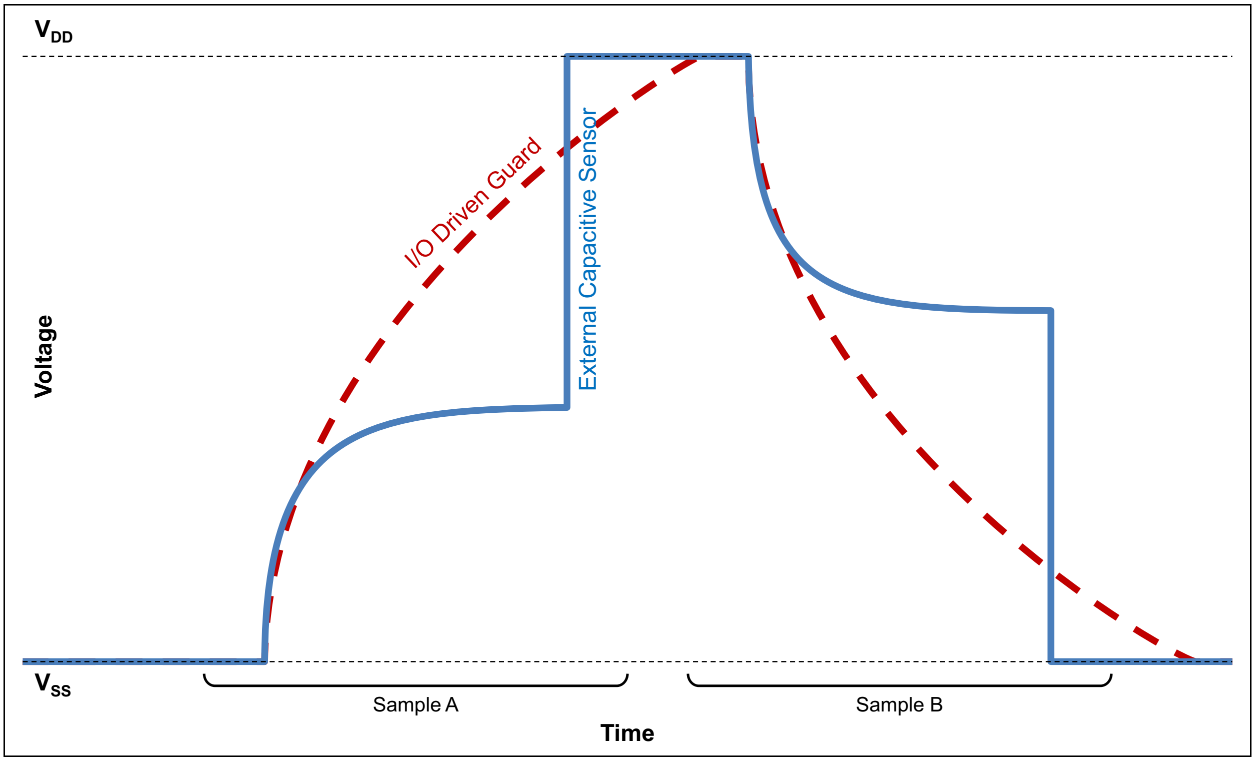
Guarding with Any I/O Pin

Any I/O pin can be used to drive the CVD guard signal, as shown in Figure 12. The waveform is not perfectly matched, so the efficiency of the guard will be decreased. However, testing has shown this method provides 50-70% of the benefits of a perfectly matched guard and is definitely worth the cost of a single pin.

A series resistor can be added to the guard to increase its time constant when charged and discharged. The better the guard waveform can be designed to match the sensor's waveform, the less electric potential is seen by the sensor and the better its sensitivity.

This implementation is slightly inefficient due to a one-instruction cycle timing difference between the output of the guard and the connection of the two capacitors. (See the example code implementation in the appendix.) This time difference could be overcome by using a hardware timer and a PWM output to synchronize the guard's change in potential with the connection of the two capacitors. Due to the tuning required for this configuration based on FOSC, no example code is provided.

FIGURE 12: I/O DRIVEN GUARD WAVEFORM

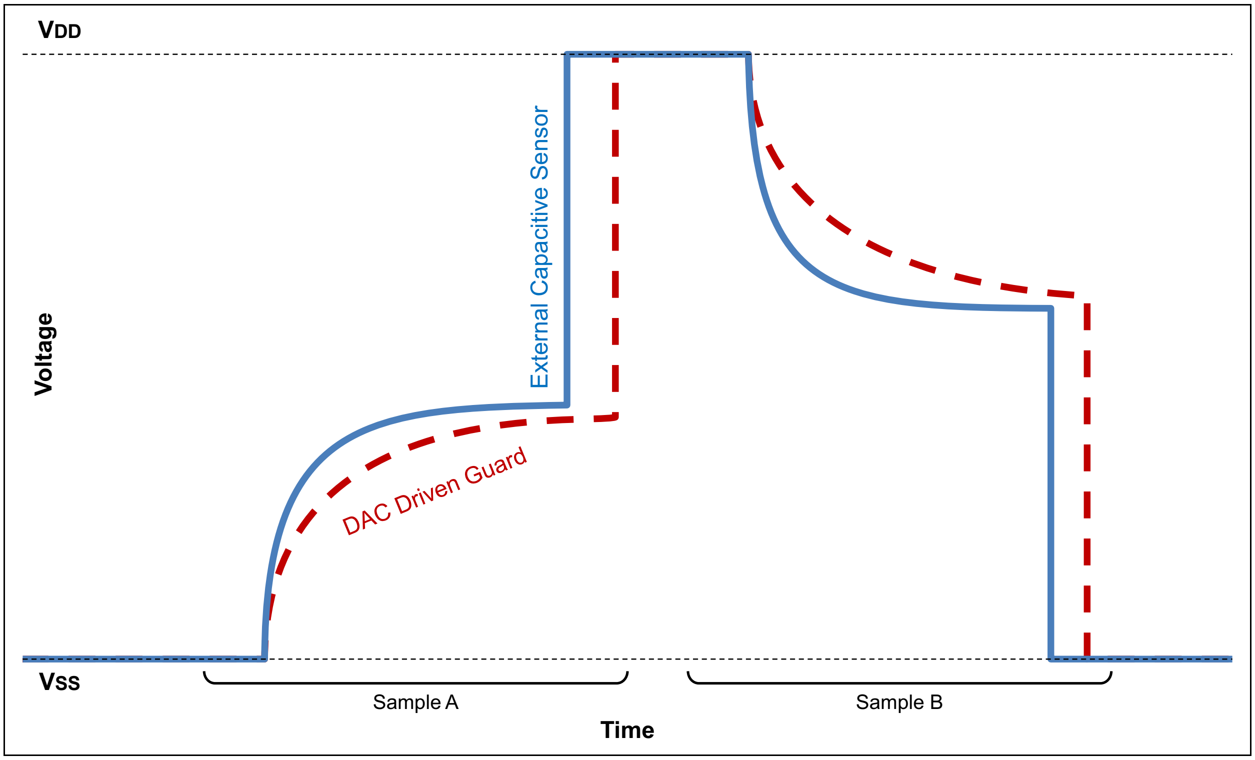


Guarding with the DACOUT Pin

If the DACOUT pin is used to drive the guard signal as shown in Figure 13, the sensor's waveform can be much more closely matched by choosing the settling value of the DAC during the two Acquisition stages. This method provides 70-90% of the benefits of a perfectly matched guard. If the DACOUT pin is available, this is the recommended method to use. If DACOUT is not available, using an I/O driven guard is the next best option.

The benefits of the guard could be further enhanced by using the individual CVD settling values for Sample A and Sample B to constantly tune the DAC to the closest matching output voltage for each. For a 10-bit ADC and a 5-bit DAC, this can be achieved by simply right-shifting the ADC result by 5 bits and using it as the DACOUT settling voltage on the next sample.

FIGURE 13: DACOUT-DRIVEN GUARD WAVEFORM



CVD SCANNING WITH MUTUAL DRIVER

The CVD sensing method was designed to measure relative changes in capacitance; however, high-impedance traces like capacitive sensors will also be affected by nearby low-impedance sources such as ground/power planes, antennas, and high-frequency digital traces.

The purpose of an active-guard trace is to minimize the electric potential of the environment as seen by the sensor. As discussed in the guard section of this application note, this increases the sensitivity of the sensor to capacitive changes.

The purpose of a mutually-driven trace, however, is to detect a change in coupling between the high-impedance capacitive sensor and the low-impedance mutual signal.

Mutual Drive Usage Scenarios

There are two main situations where a mutual drive signal is advantageous to a design:

- If a piece of metal has the possibility of physically touching the sensor, driving the metal with a mutual signal will eliminate glitches on the sensor's reading when the short occurs.

  (For example, the metal layer on a metal-over-capacitive system. This is not required, but beneficial if the metal layer can short to the sensor.)
- If the target being detected is isolated from the sensor's ground reference, placing a mutual drive near the sensor will allow the application to detect changes in the permittivity between the sensor and the mutual drive.

Note:	More information about Microchip's Metal-Over-Capacitive designs can be found in application note AN1325, "*mTouch™ Metal-Over-Cap Technology*", which can be found on our web site at www.microchip.com/mTouch.

Theory of Operation

The mutual drive signal is a square wave that is driven low during all of Sample A, and high during all of Sample B. (So it is in-phase with the external sensor's starting voltage for each sample.) This is shown in Figure 14.

Two effects will be measured by the capacitive sensor when performing any CVD waveform:

1. The capacitance of the sensor will be measured as normal due to the typical CVD waveform physics.

   Sample A will decrease in value when the capacitance increases.

   Sample B will increase in value when the capacitance increases.
2. The coupling of the sensor to the mutual drive will also be measured.

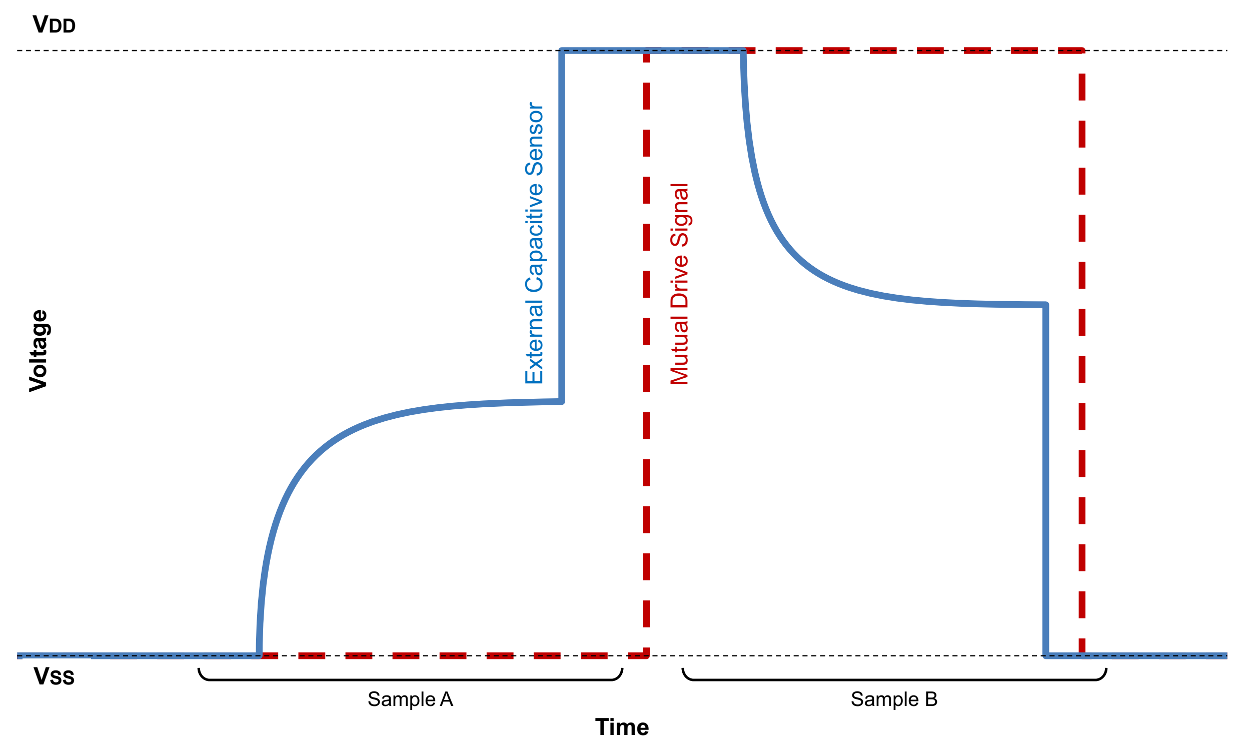
   Sample A will decrease in value when the coupling increases.

   Sample B will increase in value when the coupling increases.

Without the mutual drive, increased coupling between that trace and the sensor would cause conflicts between the two effects taking place.

For example, if the trace were driven to VSS at all times, then increased coupling would cause Sample B to decrease its settling voltage. At the same time, increased capacitance will cause Sample B to increase its settling voltage. Since these two effects tend to happen at the same time, there is a conflict between which direction to shift and sensitivity is lost. There will also be strange behaviors on the sensor's signal when one effect overpowers the other.

FIGURE 14: I/O-DRIVEN MUTUAL COUPLING WAVEFORM



As the coupling between the mutual drive trace and the capacitive sensor increases, the direction of shift caused by the increased coupling will match the direction of shift caused by increased capacitance. This will maximize the level of sensitivity on the sensor and avoids conflicting influences.

HARDWARE CVD

Some select PIC® devices contain an advanced ADC module capable of generating the CVD waveform in hardware.

Examples include:

- PIC16(L)F1512/3
  - 17 analog channels
- PIC12LF1552
  - 4 analog channels

The hardware module allows program space to be saved in your application by not having to implement a dedicated scan routine for each sensor. It also reduces the required execution time of the mTouch sensing service routine, which increases the number of other functions the application can perform. Because the scan is performed in hardware, the CPU is free to execute other algorithms, while the hardware manages the scanning method.

Features

The hardware CVD module adds functionality to the typical 10-bit ADC available on PIC16 enhanced core devices.

- Highly configurable automatic differential CVD waveform generation on any analog channel
- Two sets of ADC result registers for storing the Sample A and Sample B results
- Software adjustable internal ADC capacitance from +0pF to +28pF in increments of 4pF
- Automatic conversion trigger based on timers and/or the CCP module
- Optional pin connection to the internal ADC bus for external visibility
- Configurable precharge and acquisition waveform timing
- Synchronous guard ring drive outputs using one or two pins

Module Configuration

AADCON0 must enable the ADC module.

AADCON1 must set the conversion clock according to the current FOSC value. See the device's data sheet for more information on how to do this for your application. The recommended positive reference is always VDD.

AADCON2 determines if there are any triggers that cause the GO/$\overline{\text{DONE}}$ bit of the ADC to be set automatically. If so, it is recommended to enable GIE, PEIE, and ADIE and use the ADIF flag within the Interrupt Service Routine to process the result. The ADIF flag must be cleared manually.

AADCON3 manipulates the CVD waveform. ADEPPOL and ADIPPOL should be set to opposite values to ensure the Precharge stage works correctly. The waveforms in this application note correspond to ADEPPOL = 0 and ADIPPOL = 1, though the opposite would also work. ADDSEN should be set high to enable the second sample. ADIPEN should be set high to invert the second sample and achieve a differential waveform rather than two copies of 'Sample A'.

AADPRE must contain a value larger than '0'. It should be set to ensure more than enough time to fully charge both external and internal capacitors. See the Timing Considerations section of this application note for more information.

AADACQ must contain a value larger than '0'. It should be set to the minimum value allowing both capacitors to settle completely. See the Timing Considerations section of this application note for more information.

AADGRD can optionally be used to enable a synchronous guard on one or two pins. See the device's data sheet for more information on how to do this for your application.

AADCAP can be used to increase the internal ADC hold capacitance. This is beneficial only when the external sensor has a capacitance that is larger than the internal capacitor. (The external capacitor is larger than the internal capacitor when the Sample A settling voltage is smaller than VDD/2.) The value of AADCAP should ideally be adjusted until the settling voltage matches VDD/2.

Theory of Operation

Due to the automatic nature of this process, C code may be used to drive the waveform when using this module. Assembly is the only recommended programming method for implementing CVD manually.

After the module is configured and the GO/$\overline{\text{DONE}}$ bit is set, the ADC module will execute the CVD state machine and perform two conversions. The GO/$\overline{\text{DONE}}$ bit is cleared and the ADIF bit is set only after both conversions are complete.

The results of both conversions are stored in matching sets of result registers: AADRES0 and AADRES1. (There are four total 8-bit registers where the values are placed: AADRES0L, AADRES0H, AADRES1L, and AADRES1H.)

EXAMPLE 1: HARDWARE CVD MODULE INITIALIZATION AND USAGE

```
void InitHardwareADC(void)
{

    AADCON0bits.ADON    = 1;

    // Right aligned, Vref = Vdd
    AADCON1bits.ADFM    = 1;
    AADCON1bits.ADCS    = 0b101;
    AADCON1bits.ADPREF  = 0b00;

    // No trigger selected
    ADCON2bits.TRIGSEL  = 0b000;

    // Sample A: External Vss, Internal Vdd
    AADCON3bits.ADEPPOL = 0;
    AADCON3bits.ADIPPOL = 1;
    AADCON3bits.ADDSEN  = 1;
    AADCON3bits.ADIPEN  = 1;
    AADCON3bits.ADOLEN  = 0;
    AADCON3bits.ADOEN   = 0;
    AADCON3bits.ADOOEN  = 0;

    AADPRE   = PRECHARGE_DELAY;
    AADACQ   = SETTLING_DELAY;

    // Single-Guard-Pin enabled
    AADGRDbits.GRDAOE   = 1;
    AADGRDbits.GRDBOE   = 0;
    AADGRDbits.GRDPOL   = 0;

    // 4pf additional ADC capacitance
    ADCAPbits.ADDCAP    = 0b00001;

}

void ServiceHardwareADC(void)
{

    AADCON0bits.CHS = 0b00000;
    AADCON0bits.GO  = 1;

    while(AADCON0bits.GO);
    differentialResult  = AADRES1 | 0x0400;
    differentialResult -= AADRES0;

}
```

SUMMARY

The capacitive voltage division (CVD) sensing method is a powerful, low-cost technique for sensing relative changes in capacitance.

The basic scan configuration works well for a majority of applications and requires the fewest components of any solution on the market. Advanced features, such as guard traces and mutual coupling drives, allow any system to achieve a quality signal-to-noise ratio as long as smart design decisions are made.

For more information about Microchip’s mTouch™ sensing techniques and the recommended hardware design guidelines to maximize noise robustness, visit our web site at www.microchip.com/mTouch.

APPENDIX A – CODE EXAMPLES

Example: DAC as Reference

This code example implements the CVD waveform using the DAC as the reference voltage source for the internal hold capacitor, rather than a separate analog channel. In single-sensor applications, this prevents the scan from dedicating a pin as the reference.

SAMPLE A:

```
#define    PIC_DACCON0_VDD         0xC0
#define    PIC_DACCON1_VDD         0x1F
#define    PIC_ADCON0_SELECT_DAC   0x79
#define    PIC_ADCON0_SELECT_AN0   0x01

#define    SENSOR_LAT              LATA
#define    SENSOR_TRIS             TRISA
#define    SENSOR_PIN              0

; Initialize the DAC for 'VDD' Reference
BANKSEL      DACCON0
movlw        PIC_DACCON0_VDD
movwf        DACCON0
movlw        PIC_DACCON1_VDD
movwf        DACCON1

; Precharge ADC Capacitor
BANKSEL      ADCON0
movlw        PIC_ADCON0_SELECT_DAC
movwf        ADCON0

movlw LOW    SENSOR_LAT
movwf        FSR1L
movlw HIGH   SENSOR_LAT
movwf        FSR1H
movlw LOW    SENSOR_TRIS
movwf        FSR0L
movlw HIGH   SENSOR_TRIS
movwf        FSR0H

; Optional additional and/or variable delay
NOP

; Acquisition Stage
movlw        PIC_ADCON0_SELECT_AN0
bsf          INDF0, SENSOR_PIN    ; (TRIS)
movwf        ADCON0

; Acquisition / Settling Delay
NOP
NOP
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf          ADCON0, 1

; (Recommended) 0.5 TAD Delay
NOP

; Precharge Sensor for Sample B
bsf          INDF1, SENSOR_PIN    ; (LAT)
bcf          INDF0, SENSOR_PIN    ; (TRIS)

btfsc        ADCON0, 1
goto         $-1

; Result stored in ADRESL and ADRESH
```

The differences between Sample A and Sample B have been bolded and colored in red.

SAMPLE B:

```
#define    PIC_DACCON0_VSS         0x00
#define    PIC_DACCON1_VSS         0x00
#define    PIC_ADCON0_SELECT_DAC   0x79
#define    PIC_ADCON0_SELECT_AN0   0x01

#define    SENSOR_LAT              LATA
#define    SENSOR_TRIS             TRISA
#define    SENSOR_PIN              0

; Initialize the DAC for 'VSS' Reference
BANKSEL      DACCON0
movlw        PIC_DACCON0_VSS
movwf        DACCON0
movlw        PIC_DACCON1_VSS
movwf        DACCON1

; Precharge ADC Capacitor
BANKSEL      ADCON0
movlw        PIC_ADCON0_SELECT_DAC
movwf        ADCON0

movlw LOW    SENSOR_LAT
movwf        FSR1L
movlw HIGH   SENSOR_LAT
movwf        FSR1H
movlw LOW    SENSOR_TRIS
movwf        FSR0L
movlw HIGH   SENSOR_TRIS
movwf        FSR0H

; Optional additional and/or variable delay
NOP

; Acquisition Stage
movlw        PIC_ADCON0_SELECT_AN0
bsf          INDF0, SENSOR_PIN    ; (TRIS)
movwf        ADCON0

; Acquisition / Settling Delay
NOP
NOP
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf          ADCON0, 1

; (Recommended) 0.5 TAD Delay
NOP

; Precharge Sensor for next Sample A
bcf          INDF1, SENSOR_PIN    ; (LAT)
bcf          INDF0, SENSOR_PIN    ; (TRIS)

btfsc        ADCON0, 1
goto         $-1

; Result stored in ADRESL and ADRESH
```

Example: Sensor as Reference

This code example implements the CVD waveform using another analog channel as the reference voltage source for the internal hold capacitor. This is the lowest power scanning option since the DAC is not required.

SAMPLE A:

```
#define   SENSOR_LAT              LATA
#define   SENSOR_TRIS             TRISA
#define   SENSOR_PIN              0
#define   REFERENCE_LAT           LATA
#define   REFERENCE_PIN           1

#define   PIC_ADCON0_SELECT_AN0   0x01
#define   PIC_ADCON0_SELECT_AN1   0x05

; Initialize AN1 as the Reference Source
BANKSEL     REFERENCE_LAT
bsf         REFERENCE_LAT, REFERENCE_PIN

; Precharge ADC Capacitor
BANKSEL     ADCON0
movlw       PIC_ADCON0_SELECT_AN1
movwf       ADCON0

movlw LOW   SENSOR_LAT
movwf       FSR1L
movlw HIGH  SENSOR_LAT
movwf       FSR1H
movlw LOW   SENSOR_TRIS
movwf       FSR0L
movlw HIGH  SENSOR_TRIS
movwf       FSR0H

; Optional additional and/or variable delay

NOP

; Acquisition Stage
movlw       PIC_ADCON0_SELECT_AN0
bsf         INDF0, SENSOR_PIN    ; (TRIS)
movwf       ADCON0

; Acquisition / Settling Delay
NOP
NOP
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf         ADCON0 ,1

; (Recommended) 0.5 TAD Delay
NOP

; Precharge Sensor for Sample B
bsf         INDF1, SENSOR_PIN    ; (LAT)
bcf         INDF0, SENSOR_PIN    ; (TRIS)

btfsc       ADCON0, 1
goto        $-1

; Result stored in ADRESL and ADRESH
```

The differences between Sample A and Sample B have been bolded and colored in red.

SAMPLE B:

```
#define   SENSOR_LAT              LATA
#define   SENSOR_TRIS             TRISA
#define   SENSOR_PIN              0
#define   REFERENCE_LAT           LATA
#define   REFERENCE_PIN           1

#define   PIC_ADCON0_SELECT_AN0   0x01
#define   PIC_ADCON0_SELECT_AN1   0x05

; Initialize AN1 as the Reference Source
BANKSEL     REFERENCE_LAT
bcf         REFERENCE_LAT, REFERENCE_PIN

; Precharge ADC Capacitor
BANKSEL     ADCON0
movlw       PIC_ADCON0_SELECT_AN1
movwf       ADCON0

movlw LOW   SENSOR_LAT
movwf       FSR1L
movlw HIGH  SENSOR_LAT
movwf       FSR1H
movlw LOW   SENSOR_TRIS
movwf       FSR0L
movlw HIGH  SENSOR_TRIS
movwf       FSR0H

; Optional additional and/or variable delay

NOP

; Acquisition Stage
movlw       PIC_ADCON0_SELECT_AN0
bsf         INDF0, SENSOR_PIN    ; (TRIS)
movwf       ADCON0

; Acquisition / Settling Delay
NOP
NOP
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf         ADCON0 ,1

; (Recommended) 0.5 TAD Delay
NOP

; Precharge Sensor for Sample A
bcf         INDF1, SENSOR_PIN    ; (LAT)
bcf         INDF0, SENSOR_PIN    ; (TRIS)

btfsc       ADCON0, 1
goto        $-1

; Result stored in ADRESL and ADRESH
```

Example: Double-CVD Waveform with DAC as Reference

This code example implements the Double-CVD waveform using the DAC as the reference voltage source for the internal ADC hold capacitance.

SAMPLE A:

```
#define   PIC_DACCON0_VDD        0xC0
#define   PIC_DACCON1_VDD        0x1F
#define   PIC_ADCON0_SELECT_DAC  0x79
#define   PIC_ADCON0_SELECT_AN0  0x01
#define   SENSOR_LAT             LATA
#define   SENSOR_TRIS            TRISA
#define   SENSOR_PIN             0

; Initialize the DAC for 'VDD' Reference
BANKSEL     DACCON0
movlw       PIC_DACCON0_VDD
movwf       DACCON0
movlw       PIC_DACCON1_VDD
movwf       DACCON1

; Precharge ADC Capacitor
BANKSEL     ADCON0
movlw       PIC_ADCON0_SELECT_DAC
movwf       ADCON0

movlw LOW   SENSOR_LAT
movwf       FSR1L
movlw HIGH  SENSOR_LAT
movwf       FSR1H
movlw LOW   SENSOR_TRIS
movwf       FSR0L
movlw HIGH  SENSOR_TRIS
movwf       FSR0H

; First Acquisition Stage
movlw       PIC_ADCON0_SELECT_AN0
bsf         INDF0, SENSOR_PIN   ; (TRIS)
movwf       ADCON0
NOP
NOP         ; Acquisition / Settling Delay
NOP
; Recharge the internal ADC capacitance
movlw       PIC_ADCON0_SELECT_DAC
movwf       ADCON0
NOP         ; Optional, variable delay
; Second Acquisition Stage
movlw       PIC_ADCON0_SELECT_AN0
movwf       ADCON0
NOP
NOP         ; Acquisition / Settling Delay
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf         ADCON0, 1

; (Recommended) 0.5 TAD Delay
NOP

; Precharge Sensor for Sample B
bsf         INDF1, SENSOR_PIN   ; (LAT)
bcf         INDF0, SENSOR_PIN   ; (TRIS)

btfsc       ADCON0, 1
goto        $-1

; Result stored in ADRESL and ADRESH
```

SAMPLE B:

```
#define   PIC_DACCON0_VSS        0x00
#define   PIC_DACCON1_VSS        0x00
#define   PIC_ADCON0_SELECT_DAC  0x79
#define   PIC_ADCON0_SELECT_AN0  0x01
#define   SENSOR_LAT             LATA
#define   SENSOR_TRIS            TRISA
#define   SENSOR_PIN             0

; Initialize the DAC for 'VSS' Reference
BANKSEL     DACCON0
movlw       PIC_DACCON0_VSS
movwf       DACCON0
movlw       PIC_DACCON1_VSS
movwf       DACCON1

; Precharge ADC Capacitor
BANKSEL     ADCON0
movlw       PIC_ADCON0_SELECT_DAC
movwf       ADCON0

movlw LOW   SENSOR_LAT
movwf       FSR1L
movlw HIGH  SENSOR_LAT
movwf       FSR1H
movlw LOW   SENSOR_TRIS
movwf       FSR0L
movlw HIGH  SENSOR_TRIS
movwf       FSR0H

; First Acquisition Stage
movlw       PIC_ADCON0_SELECT_AN0
bsf         INDF0, SENSOR_PIN   ; (TRIS)
movwf       ADCON0
NOP
NOP         ; Acquisition / Settling Delay
NOP
; Recharge the internal ADC capacitance
movlw       PIC_ADCON0_SELECT_DAC
movwf       ADCON0
NOP         ; Optional, variable delay
; Second Acquisition Stage
movlw       PIC_ADCON0_SELECT_AN0
movwf       ADCON0
NOP
NOP         ; Acquisition / Settling Delay
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf         ADCON0, 1

; (Recommended) 0.5 TAD Delay
NOP

; Precharge Sensor for Sample A
bcf         INDF1, SENSOR_PIN   ; (LAT)
bcf         INDF0, SENSOR_PIN   ; (TRIS)

btfsc       ADCON0, 1
goto        $-1

; Result stored in ADRESL and ADRESH
```

Example: Half-CVD Waveform with DAC as Reference

This method requires an unimplemented ADC channel to store the ADC's charge.

This code example implements the Half-CVD waveform using the DAC as the reference voltage source for the internal ADC hold capacitance.

SAMPLE A:

```
#define    PIC_DACCON0_VDD        0xC0
#define    PIC_DACCON1_VDD        0x1F
#define    PIC_ADCON0_SELECT_DAC  0x79
#define    PIC_ADCON0_SELECT_AN0  0x01
#define    PIC_ADCON0_UNIMP_CH    0xF1
#define    SENSOR_LAT             LATA
#define    SENSOR_TRIS            TRISA
#define    SENSOR_PIN             0

; Initialize the DAC for 'VDD' Reference
BANKSEL      DACCON0
movlw        PIC_DACCON0_VDD
movwf        DACCON0
movlw        PIC_DACCON1_VDD
movwf        DACCON1

; Precharge ADC Capacitor
BANKSEL      ADCON0
movlw        PIC_ADCON0_SELECT_DAC
movwf        ADCON0

movlw LOW    SENSOR_LAT
movwf        FSR1L
movlw HIGH   SENSOR_LAT
movwf        FSR1H
movlw LOW    SENSOR_TRIS
movwf        FSR0L
movlw HIGH   SENSOR_TRIS
movwf        FSR0H

; First Acquisition Stage
movlw        PIC_ADCON0_SELECT_AN0
bsf          INDF0, SENSOR_PIN   ; (TRIS)
movwf        ADCON0
NOP
NOP          ; Acquisition / Settling Delay
NOP
; Recharge the external capacitance
movlw        PIC_ADCON0_UNIMP_CH
movwf        ADCON0
bcf          INDF0, SENSOR_PIN   ; (TRIS)
NOP          ; Optional, variable delay
; Second Acquisition Stage
movlw        PIC_ADCON0_SELECT_AN0
bsf          INDF0, SENSOR_PIN   ; (TRIS)
movwf        ADCON0
NOP
NOP          ; Acquisition / Settling Delay
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf          ADCON0, 1
NOP          ; (Recommended) 0.5 TAD Delay

; Precharge Sensor for Sample B
bsf          INDF1, SENSOR_PIN   ; (LAT)
bcf          INDF0, SENSOR_PIN   ; (TRIS)

btfsc        ADCON0, 1
goto         $-1
; Result stored in ADRESL and ADRESH
```

SAMPLE B:

```
#define    PIC_DACCON0_VSS        0x00
#define    PIC_DACCON1_VSS        0x00
#define    PIC_ADCON0_SELECT_DAC  0x79
#define    PIC_ADCON0_SELECT_AN0  0x01
#define    PIC_ADCON0_UNIMP_CH    0xF1
#define    SENSOR_LAT             LATA
#define    SENSOR_TRIS            TRISA
#define    SENSOR_PIN             0

; Initialize the DAC for 'VSS' Reference
BANKSEL      DACCON0
movlw        PIC_DACCON0_VSS
movwf        DACCON0
movlw        PIC_DACCON1_VSS
movwf        DACCON1

; Precharge ADC Capacitor
BANKSEL      ADCON0
movlw        PIC_ADCON0_SELECT_DAC
movwf        ADCON0

movlw LOW    SENSOR_LAT
movwf        FSR1L
movlw HIGH   SENSOR_LAT
movwf        FSR1H
movlw LOW    SENSOR_TRIS
movwf        FSR0L
movlw HIGH   SENSOR_TRIS
movwf        FSR0H

; First Acquisition Stage
movlw        PIC_ADCON0_SELECT_AN0
bsf          INDF0, SENSOR_PIN   ; (TRIS)
movwf        ADCON0
NOP
NOP          ; Acquisition / Settling Delay
NOP
; Recharge the external capacitance
movlw        PIC_ADCON0_UNIMP_CH
movwf        ADCON0
bcf          INDF0, SENSOR_PIN   ; (TRIS)
NOP          ; Optional, variable delay
; Second Acquisition Stage
movlw        PIC_ADCON0_SELECT_AN0
bsf          INDF0, SENSOR_PIN   ; (TRIS)
movwf        ADCON0
NOP
NOP          ; Acquisition / Settling Delay
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf          ADCON0, 1
NOP          ; (Recommended) 0.5 TAD Delay

; Precharge Sensor for Sample B
bcf          INDF1, SENSOR_PIN   ; (LAT)
bcf          INDF0, SENSOR_PIN   ; (TRIS)

btfsc        ADCON0, 1
goto         $-1
; Result stored in ADRESL and ADRESH
```

EXAMPLE: I/O Driven Guard

This code example implements the CVD waveform using another sensor as the reference voltage source for the internal hold capacitor and a dedicated I/O pin for the guard signal.

SAMPLE A:

```
#define   SENSOR_LAT                 LATA
#define   SENSOR_TRIS                TRISA
#define   SENSOR_PIN                 0
#define   REFERENCE_LAT              LATA
#define   REFERENCE_PIN              1
#define   GUARD_LAT                  LATA
#define   GUARD_PIN                  2
#define   PIC_ADCON0_SELECT_AN0      0x01
#define   PIC_ADCON0_SELECT_AN1      0x05

; Initialize AN1 as the Reference Source
BANKSEL     REFERENCE_LAT
bsf         REFERENCE_LAT, REFERENCE_PIN

; Precharge ADC Capacitor
BANKSEL     ADCON0
movlw       PIC_ADCON0_SELECT_AN1
movwf       ADCON0

movlw LOW   GUARD_LAT
movwf       FSR1L
movlw HIGH  GUARD_LAT
movwf       FSR1H
movlw LOW   SENSOR_TRIS
movwf       FSR0L
movlw HIGH  SENSOR_TRIS
movwf       FSR0H

; Optional additional and/or variable delay
NOP

; Acquisition Stage
movlw       PIC_ADCON0_SELECT_AN0
bsf         INDF1, GUARD_PIN    ; Guard
bsf         INDF0, SENSOR_PIN   ; TRIS
movwf       ADCON0

; Acquisition / Settling Delay
NOP
NOP
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf         ADCON0 ,1

; (Recommended) 0.5 TAD Delay
NOP

; Precharge Sensor for Sample B
BANKSEL     SENSOR_LAT
bsf         SENSOR_LAT, SENSOR_PIN
bcf         INDF0, SENSOR_PIN   ; (TRIS)

BANKSEL     ADCON0
btfsc       ADCON0, 1
goto        $-1

; Result stored in ADRESL and ADRESH
```

The differences between Sample A and Sample B have been bolded and colored in red. Notice that the steps remain the same except the initialization of the voltage references and the guard drive.

SAMPLE B:

```
#define   SENSOR_LAT                 LATA
#define   SENSOR_TRIS                TRISA
#define   SENSOR_PIN                 0
#define   REFERENCE_LAT              LATA
#define   REFERENCE_PIN              1
#define   GUARD_LAT                  LATA
#define   GUARD_PIN                  2
#define   PIC_ADCON0_SELECT_AN0      0x01
#define   PIC_ADCON0_SELECT_AN1      0x05

; Initialize AN1 as the Reference Source
BANKSEL     REFERENCE_LAT
bcf         REFERENCE_LAT, REFERENCE_PIN

; Precharge ADC Capacitor
BANKSEL     ADCON0
movlw       PIC_ADCON0_SELECT_AN1
movwf       ADCON0

movlw LOW   GUARD_LAT
movwf       FSR1L
movlw HIGH  GUARD_LAT
movwf       FSR1H
movlw LOW   SENSOR_TRIS
movwf       FSR0L
movlw HIGH SENSOR_TRIS
movwf       FSR0H

; Optional additional and/or variable delay
NOP

; Acquisition Stage
movlw       PIC_ADCON0_SELECT_AN0
bcf         INDF1, GUARD_PIN    ; Guard
bsf         INDF0, SENSOR_PIN   ; (TRIS)
movwf       ADCON0

; Acquisition / Settling Delay
NOP
NOP
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf         ADCON0 ,1

; (Recommended) 0.5 TAD Delay
NOP

; Precharge Sensor for Sample A
BANKSEL     SENSOR_LAT
bcf         SENSOR_LAT, SENSOR_PIN
bcf         INDF0, SENSOR_PIN   ; (TRIS)

BANKSEL     ADCON0
btfsc       ADCON0, 1
goto        $-1

; Result stored in ADRESL and ADRESH
```

EXAMPLE: DACOUT Driven Guard

SAMPLE A:

```
#define   PIC_DACCON0_VDD        0xC0
#define   PIC_DACCON1_VDD        0x1F
#define   PIC_DACCON1_SETTLE_A   0x0E
#define   PIC_ADCON0_SELECT_DAC  0x79
#define   PIC_ADCON0_SELECT_AN1  0x05
#define   SENSOR_LAT             LATA
#define   SENSOR_TRIS            TRISA
#define   SENSOR_PIN             1
#define   DACOUT_LAT             LATA
#define   DACOUT_PIN             0

; Initialize the DAC for 'VDD' Reference
BANKSEL     DACCON0
movlw       PIC_DACCON0_VDD
movwf       DACCON0
movlw       PIC_DACCON1_VDD
movwf       DACCON1

; Precharge ADC Capacitor
movlw LOW   ADCON0
movwf       FSR1L
movlw HIGH  ADCON0
movwf       FSR1H

movlw       PIC_ADCON0_SELECT_DAC
movwf       INDF1

movlw LOW   SENSOR_TRIS
movwf       FSR0L
movlw HIGH  SENSOR_TRIS
movwf       FSR0H

NOP         ; Optional delay

; Acquisition Stage
BANKSEL     DACCON0
movlw       PIC_ADCON0_SELECT_AN0
bsf         INDF0, SENSOR_PIN   ; (TRIS)
movwf       INDF1
movlw       PIC_DACCON1_SETTLE_A
movwf       DACCON1         ; Set value
bsf         DACCON0, DACOE  ; Enable DACOUT

NOP         ; Acquisition / Settling Delay
NOP
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf         INDF1, 1

; (Recommended) 0.5 TAD Delay
NOP

; Precharge Sensor for Sample B
BANKSEL     SENSOR_LAT
bsf         SENSOR_LAT, SENSOR_PIN
bcf         INDF0, SENSOR_PIN   ; (TRIS)
BANKSEL     DACCON0
bcf         DACCON0, DACOE ; Disable DACOUT
BANKSEL     DACOUT_LAT
bsf         DACOUT_LAT, DACOUT_PIN

btfsc       INDF1, 1
goto        $-1

; Result stored in ADRESL and ADRESH
```

This code example implements the CVD waveform using the DAC as the reference voltage source for the internal hold capacitor and, simultaneously, as the DACOUT guard driver.

SAMPLE B:

```
#define   PIC_DACCON0_VSS        0x00
#define   PIC_DACCON1_VSS        0x00
#define   PIC_DACCON1_SETTLE_B   0x10

; Initialize the DAC for 'VSS' Reference
BANKSEL     DACCON0
movlw       PIC_DACCON0_VSS
movwf       DACCON0
movlw       PIC_DACCON1_VSS
movwf       DACCON1

; Precharge ADC Capacitor
movlw LOW   ADCON0
movwf       FSR1L
movlw HIGH  ADCON0
movwf       FSR1H

movlw       PIC_ADCON0_SELECT_DAC
movwf       INDF1

movlw LOW   SENSOR_TRIS
movwf       FSR0L
movlw HIGH  SENSOR_TRIS
movwf       FSR0H

NOP         ; Optional delay

; Acquisition Stage
BANKSEL     DACCON0
movlw       PIC_ADCON0_SELECT_AN0
bsf         INDF0, SENSOR_PIN   ; (TRIS)
movwf       INDF1
movlw       PIC_DACCON1_SETTLE_B
movwf       DACCON1         ; Set value
bsf         DACCON0, DACOE  ; Enable DACOUT

NOP         ; Acquisition / Settling Delay
NOP
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf         INDF1, 1

; (Recommended) 0.5 TAD Delay
NOP

; Precharge Sensor for Sample A
BANKSEL     SENSOR_LAT
bcf         SENSOR_LAT, SENSOR_PIN
bcf         INDF0, SENSOR_PIN   ; (TRIS)
BANKSEL     DACCON0
bcf         DACCON0, DACOE ; Disable DACOUT
BANKSEL     DACOUT_LAT
bcf         DACOUT_LAT, DACOUT_PIN

btfsc       INDF1, 1
goto        $-1

; Result stored in ADRESL and ADRESH
```

Example: CVD Sampling Using a Sensor as the Reference and a Mutual Drive

SAMPLE A:

```
#define   SENSOR_LAT              LATA
#define   SENSOR_TRIS             TRISA
#define   SENSOR_PIN              0
#define   REFERENCE_LAT           LATA
#define   REFERENCE_PIN           1
#define   MUTUAL_LAT              LATA
#define   MUTUAL_PIN              2
#define   PIC_ADCON0_SELECT_AN0   0x01
#define   PIC_ADCON0_SELECT_AN1   0x05

; Initialize AN1 as the Reference Source
BANKSEL     REFERENCE_LAT
bsf         REFERENCE_LAT, REFERENCE_PIN

; Precharge ADC Capacitor
BANKSEL     ADCON0
movlw       PIC_ADCON0_SELECT_AN1
movwf       ADCON0

movlw LOW   SENSOR_LAT
movwf       FSR1L
movlw HIGH  SENSOR_LAT
movwf       FSR1H
movlw LOW   SENSOR_TRIS
movwf       FSR0L
movlw HIGH  SENSOR_TRIS
movwf       FSR0H

; Optional additional and/or variable delay
NOP

; Acquisition Stage
movlw       PIC_ADCON0_SELECT_AN0
bsf         INDF0, SENSOR_PIN   ; (TRIS)
movwf       ADCON0

; Acquisition / Settling Delay
NOP
NOP
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf         ADCON0 ,1

; (Recommended) 0.5 TAD Delay
NOP

; Precharge Sensor for Sample B
bsf         INDF1, SENSOR_PIN   ; (LAT)
bcf         INDF0, SENSOR_PIN   ; (TRIS)

; Prepare Mutual Drive for Sample B
BANKSEL     MUTUAL_LAT
bsf         MUTUAL_LAT, MUTUAL_PIN

BANKSEL     ADCON0
btfsc       ADCON0, 1
goto        $-1
; Result stored in ADRESL and ADRESH
```

The differences between Sample A and Sample B have been bolded and colored in red. Notice that the steps remain the same except the initialization of the voltage references and the mutual drive direction.

SAMPLE B:

```
#define   SENSOR_LAT              LATA
#define   SENSOR_TRIS             TRISA
#define   SENSOR_PIN              0
#define   REFERENCE_LAT           LATA
#define   REFERENCE_PIN           1
#define   MUTUAL_LAT              LATA
#define   MUTUAL_PIN              2
#define   PIC_ADCON0_SELECT_AN0   0x01
#define   PIC_ADCON0_SELECT_AN1   0x05

; Initialize AN1 as the Reference Source
BANKSEL     REFERENCE_LAT
bcf         REFERENCE_LAT, REFERENCE_PIN

; Precharge ADC Capacitor
BANKSEL     ADCON0
movlw       PIC_ADCON0_SELECT_AN1
movwf       ADCON0

movlw LOW   SENSOR_LAT
movwf       FSR1L
movlw HIGH  SENSOR_LAT
movwf       FSR1H
movlw LOW   SENSOR_TRIS
movwf       FSR0L
movlw HIGH  SENSOR_TRIS
movwf       FSR0H

; Optional additional and/or variable delay
NOP

; Acquisition Stage
movlw       PIC_ADCON0_SELECT_AN0
bsf         INDF0, SENSOR_PIN   ; (TRIS)
movwf       ADCON0

; Acquisition / Settling Delay
NOP
NOP
NOP

; Perform the ADC Conversion (GO/nDONE = 1)
bsf         ADCON0 ,1

; (Recommended) 0.5 TAD Delay
NOP

; Precharge Sensor for Sample A
bcf         INDF1, SENSOR_PIN   ; (LAT)
bcf         INDF0, SENSOR_PIN   ; (TRIS)

; Prepare Mutual Drive for Sample A
BANKSEL     MUTUAL_LAT
bcf         MUTUAL_LAT, MUTUAL_PIN

BANKSEL     ADCON0
btfsc       ADCON0, 1
goto        $-1
; Result stored in ADRESL and ADRESH
```

NOTES:

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION, INCLUDING BUT NOT LIMITED TO ITS CONDITION, QUALITY, PERFORMANCE, MERCHANTABILITY OR FITNESS FOR PURPOSE**.** Microchip disclaims all liability arising from this information and its use. Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights.

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
═ ISO/TS 16949 ═

Trademarks

♻ Printed on recycled paper.

ISBN: 9781620766729

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Hangzhou
Tel: 86-571-2819-3187
Fax: 86-571-2819-3189

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Osaka
Tel: 81-66-152-7160
Fax: 81-66-152-9310

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-5778-366
Fax: 886-3-5770-955

Taiwan - Kaohsiung
Tel: 886-7-213-7828
Fax: 886-7-330-9305

Taiwan - Taipei
Tel: 886-2-2508-8600
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

10/26/12