MICROCONTROLLERS

Chapter 8

STM32 Peripherials - ADC

Dr. Saeed Ebadollahi

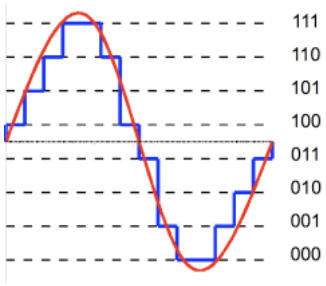
References:

- ARM® Cortex® M4 Cookbook – Mark Fischer – Packt publishing – 2016
- The Definitive Guide to ARM Cortex-M3 and Cortex-M4 Processors – Joseph Yiu – Newnes – 2014
- Discovering the STM32 Microcontroller - Geoffrey Brown - 2012

Introduction

Most signals that we encounter in the natural world are continuous; for example, we perceive sound produced by an orchestra as a continuum of intensities ranging from *pianissimo* (very soft) to *fortissimo* (very loud). Computers, on the other hand, work with binary quantities that are inherently discrete. The number of discrete values that can be represented depends on the number of bits that are used to represent the quantity (for example, 8 bits can represent 2^8 discrete values). Computers that are designed to interact with real-world phenomena (for example, sound, light, heat, and so on) need to overcome two problems. Firstly, they need to convert between its physical manifestation and a (continuous) electrical signal, and secondly, they need to convert between the signal's continuous and discrete representation. Returning to our sound example, solving the first problem requires a transducer to convert sound (pressure) waves to electrical signals and vice versa (that is, a microphone and loud speaker). Solving the second requires converting the analog (continuous) signal to a discrete form and vice versa. The device that is used to achieve this is called an **Analog-to-Digital converter** (**ADC**)—and conversely a **Digital-to-Analog converte**r (**DAC**).

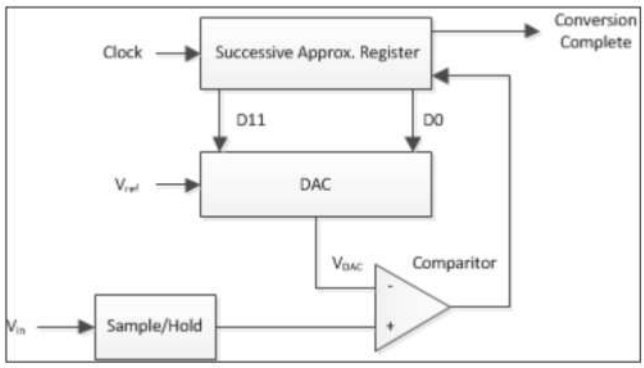
Introduction (Cont.)



How does it works ?

Analog-to-Digital conversion requires measuring (sampling) the signal at regular time intervals and converting each sample into a digital value. This raises the question, how often should we take the measurement? This fundamental question is addressed by signal processing theory. The short answer is that samples must be taken at least twice as frequently as the period of the highest-frequency component in the signal. However, the maximum number of samples that can be taken every second (that is, the maximum sampling frequency) is limited by the speed of conversion, and this, in turn, depends on the type of ADC. The STM32F407IG microcontroller includes a successive approximation ADC, which is fast enough for most audio applications (that is, signals having frequency components up to about 20 KHz). A block diagram of a successive approximation ADC is shown as follows:

How does it works ? (Cont.)



Successive approximation ADC

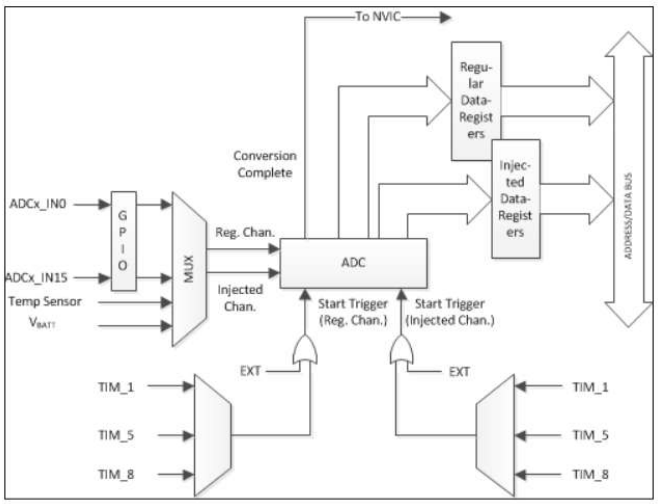
Signal Comparator

A single comparator is at the heart of the successive approximation ADC. This is simply a device that outputs a binary signal that depends on a comparison of V_{DAC} and V_{in}, where V_{DAC} represents the analog voltage corresponding to the output of the **Successive Approximation Register** (**SAR**). By testing the output of the comparator, an algorithm aims to update the SAR so as to find the value V_{DAC} that is closest to V_{in}. The successive approximation DAC achieves this by undertaking a search that aims to find V_{DAC} ($\leq V_{ref}$) in the fewest number of guesses. The time needed for the search depends on the value of the voltage, but the worst-case conversion time ultimately determines the maximum sampling frequency. The DAC is a much simpler analog circuit that uses a summing amplifier to add together the (weighted) digital outputs D0-D11. Hence, the DAC operates much faster than the ADC.

Sample and Hold Block

The purpose of the **Sample/Hold** block is to take a snapshot of the input voltage, and so, provide a stable signal for the ADC. The Sample/Hold block is not ideal and it takes some time (called the aperture time) to capture the input signal. The signal voltage stored by the Sample/Hold block also decays with time, but the Sample/Hold time can be adjusted to address these problems. A range of values can be specified in terms of a number of ADC clock cycles by writing to the two ADC **sample time registers** (SMPR1 and SMPR2). The time can be set for each channel using the following codes:

Our Microcontroller



Simplified STM32F4xxxx microcontroller ADC schematic

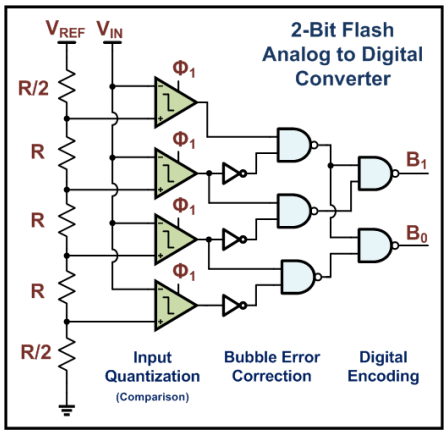
Different Types of ADCs

1. Successive Approximation ADC
2. Flash ADC
3. Sigma Delta ADC
4. Dual Slope ADC

Flash ADC

The earliest implementations consisted of a reference ladder of well matched resistors connected to a reference voltage. Each tap at the resistor ladder is used for one comparator, possibly preceded by an amplification stage, and thus generates a logical 0 or 1 depending on whether the measured voltage is above or below the reference voltage of the resistor tap. The reason to add an amplifier is twofold: it amplifies the voltage difference and therefore suppresses the comparator offset, and the kick-back noise of the comparator towards the reference ladder is also strongly suppressed. Typically designs from 4-bit up to 6-bit and sometimes 7-bit are produced.

Flash ADC

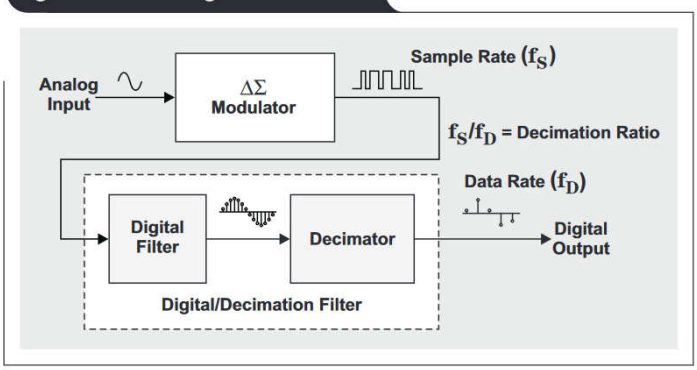


Sigma Delta ADC

Analog techniques have dominated signal processing for years, but digital techniques are slowly encroaching into this domain. The design of delta-sigma (ΔΣ) analog-to-digital converters (ADCs) is approximately three-quarters digital and one-quarter analog. ΔΣ ADCs are now ideal for converting analog signals over a wide range of frequencies, from DC to several megahertz. Basically, these converters consist of an oversampling modulator followed by a digital/decimation filter that together produce a high-resolution data-stream output. This two-part article will look closely at the ΔΣ ADC's core. Part 1 will explore the basic topology and function of the ΔΣ modulator, and Part 2 will explore the basic topology and function of the digital/decimation filter module.

Sigma Delta ADC

Figure 1. Block diagram of ΔΣ ADC



Sigma Delta ADC

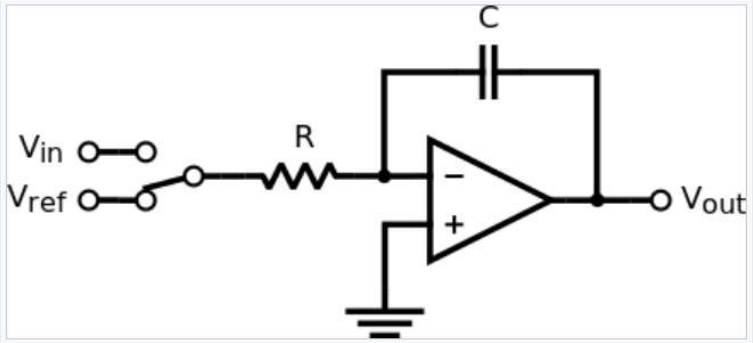
Visit this link for more information •

http://www.ti.com/lit/an/slyt423a/slyt423a.pdf

Dual Slope ADC

The basic integrating ADC circuit consists of an integrator, a switch to select between the voltage to be measured and the reference voltage, a timer that determines how long to integrate the unknown and measures how long the reference integration took, a comparator to detect zero crossing, and a controller. Depending on the implementation, a switch may also be present in parallel with the integrator capacitor to allow the integrator to be reset . Inputs to the controller include a clock (used to measure time) and the output of a comparator used to detect when the integrator's output reaches zero.

Dual Slope ADC



Basic integrator of a dual-slope integrating ADC. The comparator, the timer, and the controller are not shown.

Getting Started with ADC in Cube

Example 1:

- Write a code to read A1 pin voltage on your discovery board.
- First connect it to GND and then connect it to VDD and read numbers !
- Use debugger to read outputs.

Example 2:

- Write a code to read LDR
- Which kind of circuit we have to made for it ?
- Write a code to read potentiometer

DMA

DMA is implemented in processors with dedicated hardware devices. These devices share the memory bus and peripheral buses with the processor (CPU) as illustrated in Figure 12. In this diagram, the DMA device reads from memory over the memory bus and writes to a peripheral over the peripheral bus. The situation with the STM32 is somewhat more complicated because there are multiple peripheral buses, but the principle is the same.

DMA (Cont.)

is 750,000 transfers/second while the memory bus of the STM32 Value line device can support 24,000,000/5 RAM transfers/second (each transfer takes 5 bus cycles). Thus, our block transfer consumes roughly 15% of the memory bus cycles. The STM32 architecture guarantees that the CPU will not be starved. Furthermore, only a fraction of STM32 instructions directly access RAM memory – the rest simply pull instructions from FLASH which uses a different bus.

DMA (Cont.)

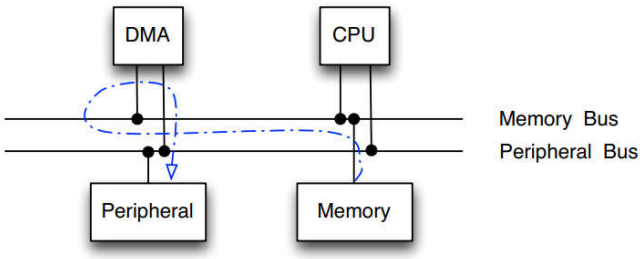


Figure 12.2: DMA Transfer

Example 3:

Use ADC DMA to read both of LDR and Potantiometer•