THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPY

EMBEDDED MEASUREMENT SYSTEMS

LARS E. BENGTSSON

UNIVERSITY OF
GOTHENBURG

Department of Physics
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2013

TITLE: Embedded Measurement Systems

LARS BENGTSSON
ISBN 978-91-628-8688-2
Internet-ID: http://hdl.handle.net/2077/32648

Department of Physics
University of Gothenburg
SE-412 96 Gothenburg, Sweden
Phone +46-(0)31-786 1000
www.physics.gu.se

Printed by Tryckeri Kompendiet
Gothenburg, Sweden 2013.

EMBEDDED MEASURMENT SYSTEMS

Lars Bengtsson
Department of Physics, University of Gothenburg

Abstract

The subject of Embedded Measurement Systems (EMS) is the merging of embedded systems and electrical measurement systems. This indicates that EMSs are hardware-software systems dedicated to measuring one or a few physical quantities. Applications are numerous; EMSs measure the temperature in refrigerators, freezers, irons, ovens and automobile combustion engines, they sense vibrations in tilt alarms and game consoles, they measure airflow in engines and ventilation systems, they measure shock impact in crash detectors and are used as shock and temperature loggers for transport goods, they measure air pressure in airplane cabins, humidity in air-conditioned environments, they measure liquid levels in fuel tanks, they detect smoke in fire alarms, they measure the viscosity of lubricant oil in engines, they measure the rotation speed of spinning wheels (in any engine), they measure torque in engines and are used as heart rate and ECG detectors in medicine etc.

The commercial demand for ever cheaper products and worldwide environmental legislations force vendors to continuously look for more cost-efficient and less power-consuming solutions for their embedded measurement systems. This thesis is concerned most of all with the implementation of cost-efficient/low-power measurement systems in embedded controllers. This includes some novel ideas in voltage, time and resistance measurements with embedded controllers and it will demonstrate how these quantities, analog in nature, can be measured accurately and precisely by inherently digital embedded controllers.

Keywords: Microcontroller, measurement system, direct sensor-to-controller, time-to-digital converter, phase detector, lock-in amplifier.

APPENDED PAPERS

This thesis is based on the work described in the following papers:

I. **Direct Analog-to-Microcontroller Interfacing**
Lars E. Bengtsson, Sens. Act. A., **179**, pp. 105-113, (2012).
doi: 10.1016/j.sna.2012.02.048

II. **Implementation of High-Resolution TDC in 8-bit Microcontrollers**
Lars E. Bengtsson, Rev. Sci. Instr., vol. **83**(4), no 045107, (2012).
doi: 10.1063/1.3700192

III. **A microcontroller-based lock-in amplifier for sub-milliohm resistance measurements**
Lars E. Bengtsson, Rev. Sci. Instr., vol. **83**(4), no 075103, (2012).
doi: 10.1063/1.4731683

Scientific publications, in the same research area, which are not included in this thesis:

Generation and measurement of pulses and delays with RISC-controllers
Lars E. Bengtsson, Meas. Sci. Technol. **8,** pp. 679-683 (1997).
doi: 10.1088/0957-0233/8/6/017

Analysis of Direct Sensor-to-Embedded Systems Interfacing: A comparison of Target's Performance
Lars E. Bengtsson, Int. J. Intell. Mech. Rob., 2(1), pp. 41-56, (Jan-Mar 2012).
doi: 10.4018/ijimr.2012010103

Lookup Table Optimization for Sensor Linearization in Small Embedded Systems
Lars E. Bengtsson, J. Sens. Tech., vol 2(4), pp 177-184, (2012).
doi: 10.4236/jst.2012.24025

Abbreviations

AA	Anti-Aliasing
ABS	Anti-lock Braking System
ADC	Analog-to-Digital Converter
AGC	Apollo Guidance Computer
AM	Amplitude Modulation
ARM	Advanced RISC Machine (previously: Acorn RISC Machine)
CAN	Controller Area Network
CPLD	Complex Programmable Logic Device
CPU	Central Processing Unit
CTMU	Charge Time Measurement Unit
DAC	Digital-to-Analog Converter
DAQ	Data Acquisition System
DMM	Digital Multi Meter
DR	Dynamic Range
DR	Dynamic Reserve
DSP	Digital Signal Processing
EIA	Electronic Industries Association
emf	electromagnetic force
EMS	Embedded Measurement System
ENOB	Equivalent Number Of Bits
ESR	Equivalent Series Resistance
FSS	Full Scale Span
FM	Frequency Modulation
FPGA	Field Programmable Gate Array
FSK	Frequency Shift Keying
ISR	Interrupt Service Routine
LIA	Lock-In Amplifier
LSB	Least Significant Bit
MEMS	MicroElectroMechanical Systems
MIM	Metal-Insulator-Metal
MIPS	Microprocessor without Interlocked Pipeline Stages
MCU	MicroController Unit

µC	MicroController
µP	MicroProcessor
MUX	Multiplexer
OSR	OverSampling Rate
pcb	printed circuit board
PCM	Pulse Code Modulation
pdf	probability density function
PLL	Phase-Locked Loop
ppm	parts per million
PPM	Pulse Proportion Modulation
PSD	Phase Sensitive Detector
PWM	Pulse Width Modulation
RAM	Random Access Memory
rms	root mean square
RT	Real-Time
RTOS	Real-Time Operating System
RTD	Resistive Thermal Device
SAR	Successive Approximation Register
S&H	Sample&Hold
ΣΔ	Sigma-Delta
SR	Slew-Rate
SNR	Signal-to-Noise Ratio
ST	Schmidt-Trigger
TDC	Time-to-Digital Converter
TDR	Time Domain Reflectometry
TIADC	Time-Interleaved ADC
VCO	Voltage Controlled Oscillator
VHDL	VHSIC Hardware Description Language
V/F	Voltage-to-Frequency (converter)
XLP	eXtreme LowPower

Contents

CONTENT

Chapter 1

Introduction

"Embedded Measurement Systems" is the merging of electrical measurement systems and embedded systems, i.e. how to use embedded digital systems like microcontrollers and programmable gate arrays (FPGAs) in automatic and computerized measurement systems. Microcontrollers have been used in measurement systems ever since the first microcontroller was introduced by Intel in 1976 (Askdefine, 2011) and are used extensively in measurement applications today. They are used to measure anything from the heart rate of athletes and water temperature in washing machines to the speed of air planes and depth of submarines. However, the worldwide increasing focus on environmental problems, energy consumption and cost reductions, forces designers of embedded measurement systems to continuously look for more cost-efficient and less energy consuming solutions.

The increasing demand for low-cost solutions also make embedded measurement systems interesting for "very-low SNR" (Signal-to-Noise Ratio) applications, since the cost of an embedded solution typically falls much below the cost of a commercial desktop instrument.

This thesis is mainly concerned with improved methods for embedded measurement systems in terms of cost and energy consumption and in terms of improved algorithms for signal recovery in noisy environments.

In this thesis I will most of all present some new (embedded) solutions for a few typical measurement applications that improve the overall system's performance in terms of cost, energy consumption and signal recovering capability.

The rest of this thesis is organized as follows: In Chapter 2 I will define some basic concepts. Chapter 3 will treat analog interfacing to embedded systems with

traditional analog-to-digital converters (ADCs). Chapter 4 treats “direct” interfacing of sensors to controllers, i.e. I will describe how sensors can be interfaced to controllers without using an ADC. Chapter 5 is concerned with “quasi” digital signals. It turns out that time can be measured more accurately than voltage and if we can transform the sensor signal into a time variation, we can measure it more accurately (and precise). Hence, chapter 5 describes in detail how time can be measured accurately in embedded systems. Chapter 6 describes the important phase locking technique and how it can be implemented in an embedded system and chapter 7 concludes this thesis.

Finally, appendix A describes the peer-reviewed papers that this thesis is based on.

Chapter 2

Definition of Concepts

2.1 Measure/Measurement

It is not exactly clarified where the word “measure” comes from originally. The oldest known candidates are the French word “mesure”, the Latin word “mensura” and the Greek word “metron” which are all translated to “measure”. The use of the English word “measure” has been traced back to the 13th century (Merriam-Webster Dictionary, 2011).

In this thesis, the verb “measure” is defined as the process of experimentally determining the magnitude of a physical quantity such a temperature or acceleration.

2.2 Measurement Systems

A “measurement system” is the hardware and/or software necessary to measure the magnitude of the quantity of interest. It may be mechanical, electrical or a hybrid. A mercury barometer is a mechanical measurement system, while a digital multi meter (DMM) is an electrical measurement system. An integrated MEMS-accelerometer (MicroElectroMechanical System) is an example of a hybrid system.

This thesis is concerned entirely with electrical measurement systems, and to be precise, it will focus mostly on those measurement systems that take more or less advantage of embedded systems, such as microcontrollers, CPLDs (Complex Programmable Logic Devices) or FPGAs (Field Programmable Gate Arrays) in the design of the measurement system.

Electrical measurement systems may vary considerably in design and several general models have been suggested (Bentley, 1995; Doeblin, 1990; Bengtsson, 2012a), but they mostly differ in sense of details. For example, Bentley (1995) suggests the following model:

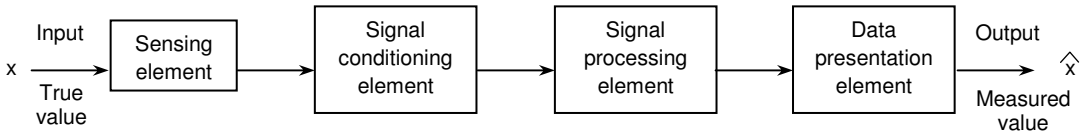


Fig 2.1 General measurement system structure (Bentley, 1995)

As indicated in figure 2.1, the measurement system's purpose is to produce an *estimator* $\hat{x}$, of the true value *x*. The estimator is characterized by its *accuracy* and its *precision*. If it is *accurate,* the average value of the measurements is close to the true value. If it is *precise*, the spread of the measured values is small. This is illustrated in figure 2.2 (Lean Six Sigma, 2011). Hence, the accuracy is $|x - \hat{x}|$ and the precision is proportional to the standard deviation of the samples, $\sigma_{\hat{x}}$.

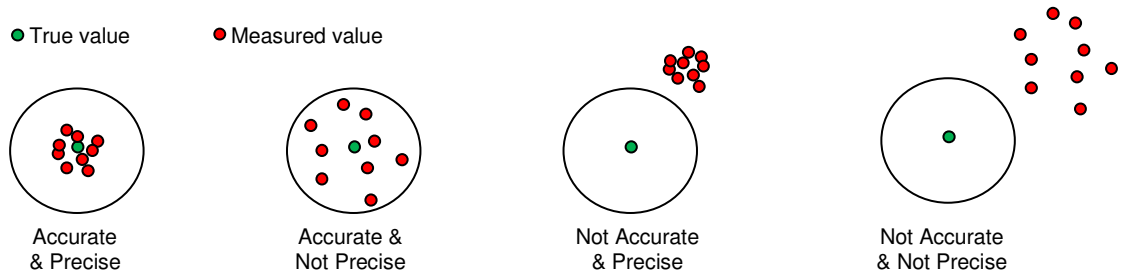


Fig 2.2 Accuracy vs precision (Lean Six Sigma, 2011)

2.3 Signal Conditioning

All measurement systems start with a sensor or a "sensing element", as indicated in figure 2.1. A sensor translates a variation in a physical quantity into a variation in some electrical quantity. For example, a Resistive Thermal Device (RTD), like a Pt-100 element, changes its resistance when the temperature changes. Sensors are *active* or *passive* depending on whether or not they need external power support or not. The electrical quantity that varies in a passive sensor is resistance, capacitance or inductance, while active sensors generate an emf (thermo couples), a charge (piezo crystals) or a current (PIN diodes). Active sensors transform physical energy into electrical energy.

The final destination of a measured estimator (a "sample") is almost always a computer. The analog sample is transformed into a "computer-friendly" digital format by an Analog-to-Digital Converter (ADC). This process is also called *quantization*. A typical ADC used in a measurement system, is only capable of transforming a *voltage* into digital form. If the electrical quantity of the sensor that

changes with the physical quantity is resistance (or anything else but voltage), the sensor cannot be directly connected to the ADC. The word "transducer" is often used as a synonym to "sensor", but "sensor" really refers to the device, while "transducer" refers to the principle involved (Unknown, 2007).

It is the *signal conditioning* electronics that adapt the sensor signal to the ADC (Bentley, 1995; Bengtsson, 2012a). In most systems, the signal conditioning also amplifies/attenuates the sensor signal in order to adapt the signal to the ADC's range. A typical ADC can handle voltages in the range of 0 – 5 volts. If the temperature range of interest is −10 to +80 °C, then the signal conditioning electronics should produce a voltage varying linearly from 0 to +5 volts when the temperature varies from −10 to +80 °C.

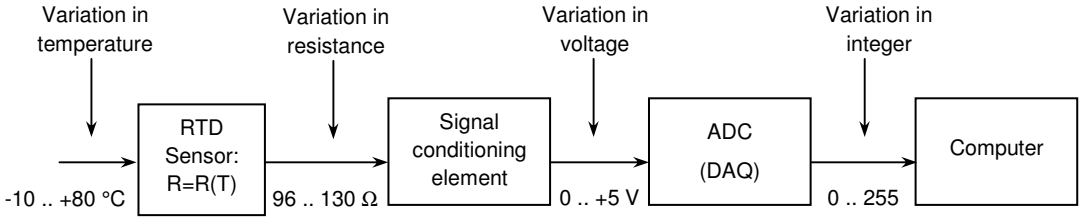


Fig 2.3 Signal conditioning

2.4 Signal Acquisition

If we take a closer look at the process of getting the physical quantity (the temperature in figure 2.3) into the computer, there are a few more steps than indicated in figure 2.3. In particular, the "ADC" block contains more than just an ADC.

First of all, before the signal may be sampled, it must be *filtered*. The rate at which the ADC converts values into digital form is the *sampling rate,* f_S, [S/s] and according to the *Nyquist sampling theorem* (Nyquist, 1928), all signals sampled must have a frequency (or "bandwidth") less than $f_S/2$, or *aliasing* will occur which will corrupt the signal (Proakis and Manolakis, 1992). The filter that prevents this from happening is called an *anti-aliasing* filter (AA) and is a low-pass filter with a cut-off frequency of $f_S/2$ (or less).

Secondly, the sensor signal is typically very small compared to the ADC's range and an *amplifier* is used to amplify the signal. By using the ADC's entire dynamic range, the resolution increases.

Next, the ADC needs some time to perform the conversion of the signal into a digital number. During this time, the ADC input signal should be kept constant. This

is the task of the *sample-and-hold* unit preceding the ADC. It consists of a switch, a capacitor and a voltage follower, see figure 2.4. The switch is closed during "sampling" while the capacitor is charged to the signal voltage level. When the switch opens, the capacitor and the voltage follower "holds" the signal level constant during the digital conversion.

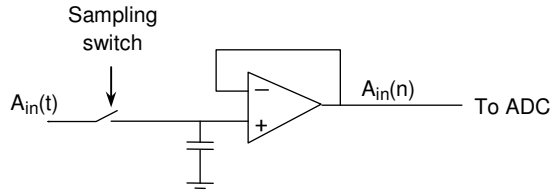


Fig 2.4 Sample-and-hold circuit (the signal is discretized by sampling)

The entire process of filtering, amplifying, sampling and quantization is a necessary part of any electrical measurement system and Carley (1987) introduced the name *Data Acquisition* for this chain of electronics (in most literature abbreviated to just *DAQ*).

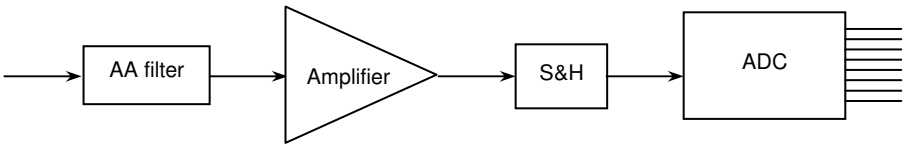


Fig 2.5 The data acquisition chain (according to Carley, 1987)

2.5 Microcontroller

A "microcontroller" (µC or MCU) is a single integrated chip that contains all necessary hardware in order to operate as a stand-alone computer (Wikipedia_1, 2011). Compared to a "microprocessor" (µP), that only contains the CPU, a µC contains both data (RAM) and program memory (flash) and some I/O hardware (counters, ADC, serial ports etc).

A µC is designed to operate in an embedded system, while a µP typically is the central unit in a general-purpose computer. µCs are more compact and cost effective while µPs are more flexible (Arnold, 2004). Architecture-wise, µP and µC designs have migrated into two characteristic implementations. µPs typically have a *von Neumann* architecture where data memory and program memory share the same data bus lines, while µCs are typically designed with *Harvard* architecture where data and program memory are physically separated in hardware.

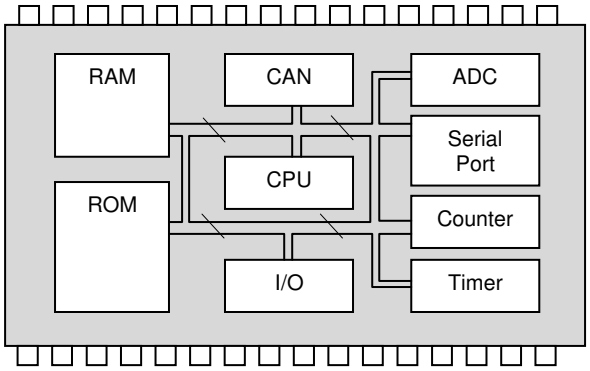


Fig 2.6 A microcontroller is a single-chip computer

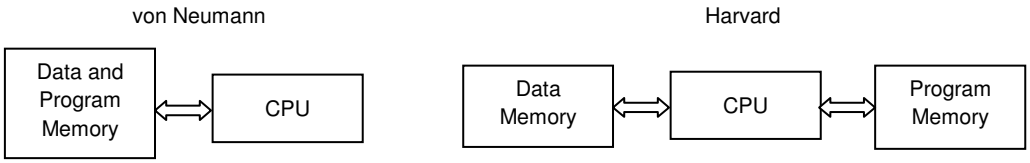


Fig 2.7 von Neumann vs Harvard architecture in computer design (Arnold, 2004)

The main advantage of the Harvard architecture is *speed*. The architecture allows *pre-fetching* of instructions; since the data and memory busses are separated, the next instruction can be fetched simultaneously as the previous instruction is still under execution. This technique is used to the extreme in modern *pipelined* 32-bit controller architectures such as MIPS and ARM. The advantage of the von Neumann architecture is *silicon area* (or *compactness*): since all memory shares the same physical hardware bus lines, the silicon area requirement is minimized at implementation.

The first microcontroller, the 8048, was designed by Intel and released in 1976 (Askdefine, 2011) and was designed to operate in the Korg Trident game console. Since then, several manufacturers have developed their own portfolio of 8-, 16- and 32-bit microcontrollers and shipped billions of microcontroller chips to vendors of digital electronics. One of the major suppliers of microcontrollers, Microchip, has shipped over 6 billion controllers by 2011 (Direct Industry, 2011) and STC ships over 100 million 8051 chips annually (Microcontroller, 2011). This is used for example in tire pressure monitors, where the controllers are necessarily powered by a battery that needs to last during the tire's entire life cycle, since the battery is not easily replaced (Lourens and Kell, 2004).

2.6 Embedded Systems

An "Embedded System" is a computer system where the software is encapsulated by the hardware it controls (Baskiyar, 2002) and should be able to run autonomously, without human interaction (Timmerman et al, 1998). Embedded systems are designed to do one task only (or a few) and for this reason the name "Dedicated systems" have been suggested by Timmerman et al (1998), but it's the name "Embedded" that is the most widely accepted name today.

The impact of embedded systems has been tremendous, much like the impact of the Otto engine, radio waves or the Wright brothers' flying machine. Embedded systems are used in a wide variety of applications, including anything from tooth brushes and athletes' heart rate monitors to satellites, space shuttles and nuclear plants. An average home in the developed world has two general-purpose computers but 20-30 embedded computers. A car alone may have 30-50 embedded systems. The automotive industry is one of the areas where the embedded technology has been widely embraced. 1/3 of the overall cost of producing a vehicle is spent on electronics and 1/3 of all semiconductors in a car/truck/bus are microcontrollers (Aroca and Caurin, 2009).

As for most of today's electronic inventions, they were originally driven by the American space and military programs. In particular, it was two simultaneous projects that were the driving forces in the development of the first embedded systems: The Apollo space project and the inter-continental Minuteman missile system (Wikipedia_2, 2011). The first "embedded" system is attributed to the Apollo Guidance Computer system (AGC) in 1961 and shortly after the Minuteman missiles were provided with a similar system. The computer in these systems was implemented using integrated logic gates only.

The Apollo/Minuteman projects are excellent examples of how space and military projects may spin off and provide technology for civil products. Before the Apollo/Minuteman projects started, the cost of an integrated NOR-gate was $1000/gate. The mass production of integrated circuits for the Apollo/Minuteman projects reduced the integrated NOR-gate price to only $3/gate in only a few years, which was an absolutely necessary condition in order for private vendors to start developing civil products based on integrated digital electronics.

Today, the development is not primarily driven by space and military projects. Entertainment industry, personal computers and private communication equipment have inherited the roll of being the leading driving force in electronics and computer industry. Game consoles, video games, animated film production and cell phones are examples of products that drive the development today. At the end of 2008 there were nearly 3 billion cell phones registered in the world, suggesting that 40 % of the world population uses cell phones[1]. In the developed world, the rate is close to 90 %

[1]At the end of 2011, 6 billion cell phones were registered (UN Telecom Agency, 2013)

(GS1 Mobile Com, 2008). (This is far more than the people who uses computers to access the Internet.)

Also, during the last decade, environmental considerations have had an enormous impact on the power consumption of embedded systems. For example, Microchip uses “nanoWatt XLP” (eXtreme Low Power) technology that consumes only 20 nA in sleep mode and can run on a single battery for more than 20 years (Microchip, 2011).

Fig 2.8 Launching of a Minuteman-III missile
(http://en.wikipedia.org /wiki/LGM30_ Minuteman)

2.7 Embedded Measurement Systems (EMS)

From the definitions above, we are now able to define an “Embedded Measurement System”, EMS. An EMS is a measurement system and hence, contains all of the electronics in figure 2.3, including the DAQ electronics in figure 2.5. However, in order to be “embedded”, there are a few more things to fulfill. First of all, the “computer” is a microcontroller (or an FPGA/CPLD). Secondly, all the electronics, except perhaps for the sensor element, should be contained on the same pcb (printed circuit board). It may, or may not, operate as a client, transferring data to a host via some communication link, where the host is typically a Windows PC. It may, or may not, be a node in a smart sensor network, like CAN, Ethernet or Zigbee.

Referring back to Baskiyar’s (2002) and Timmerman’s (1998) definitions of embedded systems, we define an embedded measurement system as follows:

An Embedded Measurement System (EMS) is an electrical measurement system consisting of a combination of hardware and software which creates a dedicated measurement system that performs specific, pre-defined measurements and which is encapsulated by the hardware it controls.

2.8 Interfacing Sensors to Microcontrollers

This thesis is mainly concerned with the problem of interfacing the (analog) sensing element to the (digital) microcontroller. A large number of interfacing techniques have been developed. The large variety of interfacing methods is motivated by the fact that different applications have different priorities. Depending on the application and the environment in which the EMS is intended to operate in, there are several different parameters to optimize the design on. These parameters include for example (BiPOM, 2006): cost, size, weight, power consumption, reliability, availability and manufacturability.

2.9 Software/firmware

In the embedded community, the programs that run the embedded device are sometimes referred to as “software” and sometimes referred to as “firmware”. Originally, “firmware” was used to refer to programs “not easily” changed, such as microcontrollers’ programs stored in flash memory. They can be updated, but not easily. Software, on the other hand, is programs that are written to run under an operating system (such as Windows) and can easily be changed. Firmware changes may require hardware changes (or, at least you need detailed knowledge about the hardware before you mess with the firmware). Software can typically be manipulated without any major knowledge about the hardware. In EMS, “firmware” would most likely be the correct term to use when referring to the programs running the devices. However, today most microcontrollers are equipped with program memory in flash technology that is readily re-programmed and the trend is that “firmware” and “software” are used interchangeably and many users use them as synonyms. In this thesis we will be using both expressions as synonyms.

Chapter 3

Interfacing analog signals

3.1 Introduction

In this chapter I will present the most common ways of interfacing a sensor/sensor signal to a μC; by using an *Analog-to-Digital Converter* (ADC).

The most obvious (and most used) way to interface an analog sensor signal (or the signal produced by the signal conditioning unit) is to simply use a μC with an integrated ADC. Every μC manufacturer provides controllers with an ADC interface.

ADCs come basically in one of three different designs; dual slopes (integrating), successive approximation (SAR) or flash (parallel). Dual slope ADCs have very high resolution but are too slow for most embedded measurement systems (EMS) (Bengtsson, 2012a). They are mainly used in digital multi meters (DMMs) where speed is not an issue. Flash ADCs are *very* fast, but they are the most component-intensive of all ADC designs and the silicon area they require for implementation is exponentially proportional to the resolution (number of bits), and the resolution of a realistic-size flash ADC is usually not good enough for EMSs or they require too large silicon area to be integrated on a μC chip. That leaves us with SAR (Successive Approximation Register) ADCs; almost all μCs with an ADC interface use a SAR implementation.

There are other kinds of ADCs too. First of all we have the ΣΔ ADCs ("sigma-delta"), and I will treat them in detail in this chapter. However, they are mostly implemented in digital signal processors (DSPs), but they can be implemented in typical non-dsp EMSs in a software/hardware combination with only a few external passive components (Peter et al, 1998; Soldera et al, 2005; STIMicroelectronics, 2008). They are gaining in popularity and are actually easy to implement in EMSs with only a few external passive components.

There are also “half flash” and “pipelined ADCs” (MAXIM, 2001a), but they are really just flash ADCs implemented in several sequential stages in order to reduce the number of required components (Bengtsson, 2012a).

Some ADCs need digital-to-analog converters (DACs), but DACs will not be covered in this thesis, see for example Socher (2012) and Wenn (2007) for a microcontroller implementation of a DAC.

3.2 ADC theory

3.2.1 The history of SAR ADCs

The SAR algorithm itself dates back to the 16th century and was presented as a solution to a popular mathematical problem: How to determine the weight of an object on a balancing scale using as few iterations as possible. In 1556, mathematician Tartaglia proposed a solution algorithm that would be implemented in millions of electronic chips more than 400 years later (Kester, 2004).

Tartaglia suggested that you use a number of counter-weights whose individual weights are “doubled”, see figure 3.1.

Tartaglia was able to prove that the fastest way to do the measurement was to start with the heaviest counter-weight, keep it if the scale doesn’t tip over, and then add/remove the weights in weight-order, the lightest one last. If the scale tips over for any new weight added, it is removed from the scale. This is exactly the SAR algorithm used in modern embedded systems to perform analog-to-digital conversion; the algorithm itself though, is by no means a “modern” invention.

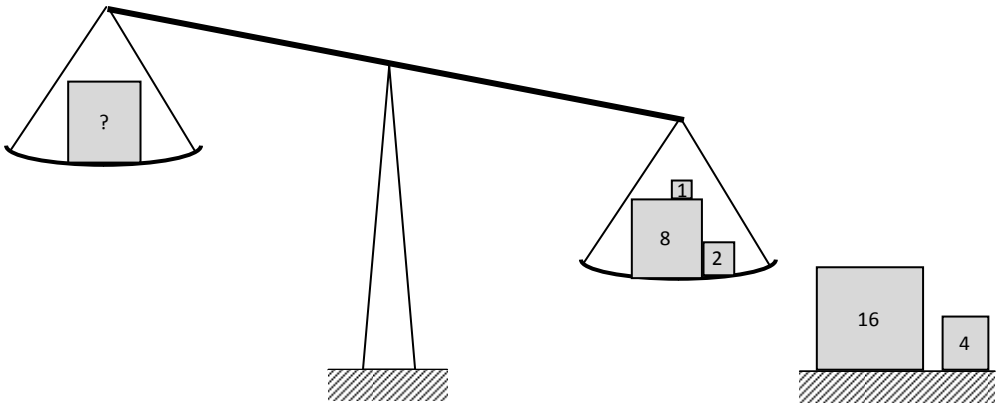


Fig 3.1 Tartaglia’s weighting algorithm

The pioneering development of SAR ADCs for use in electronic computer systems took place in the late 1940s to early 1950s. Schelleng (1946) and Goodall (1947) both developed SAR ADC prototypes for use in telephone and communication systems, but it was Bernard Gordon at Epsco Engineering who designed the first commercial SAR ADC with an R-2R DAC (see section 3.2.8) (Gordon and Talambiras, 1955). It was an 11-bit ADC sampling at 50 kS/s, build with vacuum tubes. It weighted 68 kilograms and consumed no less than 500 Watts, see figure 3.2.

From: Analogic Corporation
8 Centennial Drive
Peabody, MA 01960

http://www.analogic.com

Fig 3.2 Gordon's "DATRAK" (Gordon and Talambiras, 1955)

3.2.2 Quantization and quantization noise

First of all, by an "analog" signal, we really mean a time-continuous function where, unless otherwise indicated, the vertical axis is in volts [V]. Secondly, by "digital" we mean "integer". The sample-and-hold circuit *discretizes* the time-continuous signal. The range of the sample-and-hold produced voltages is still a continuum. The ADC *digitizes* these samples in the sense that it translates the sample into an integer.

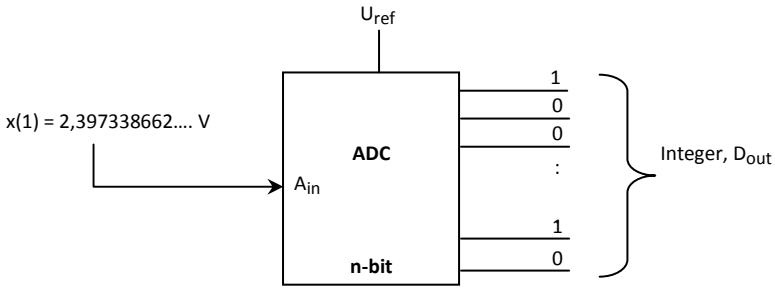


Fig 3.3 The ADC digitizes the samples (Bengtsson, 2012a)

An n-bit ADC with a reference voltage of U, has a voltage resolution of $\Delta U = U/2^n$ (= 19.53 mV for an 8-bit ADC with U = +5.00 volts). An ADC produces a correct rounding to the nearest integer of the following expression:

$$\frac{\text{sample}\left(A_{in}(n)\right)}{\Delta U}$$

Hence, the continuous sample value 2.397338662… V, will generate the integer

$$D_{out} = \left|\frac{2.397338662....}{0.01953...}\right| = 123 = 01111011_2$$

Notice that *all* samples in the range

$$123 \cdot \Delta U \pm \frac{1}{2} \cdot \Delta U = 2.40234375... \pm 0.009766.. \text{ V}$$

will result in exactly the same integer from the ADC. Hence, there will be a certain amount of uncertainty in the sample estimation calculated by the computer. This uncertainty is referred to as the *quantization uncertainty* and is an inherent property of any ADC. The quantization uncertainty is

$$\pm\frac{1}{2}\Delta U = \pm\frac{1}{2}\cdot\frac{U_{ref}}{2^n} = \pm\frac{U_{ref}}{2^{n+1}} = \pm\frac{1}{2}\text{LSB} \qquad (3.1)$$

where ”LSB” stands for Least Significant Bit. Expression (3.1) is referred to as the *quantization noise* and if the input swing is large enough, it may be considered to be a uniformly distributed stochastic variable.

We will consider the ADC integer output to be an *estimator* of the sample:

$$\text{Computer estimate}: \ \hat{A}_{in} = D_{ut} \cdot \Delta U \qquad (3.2)$$

We will treat this as a *point estimation* of the input sample A_{in}. Then this is a stochastic variable with a uniform distribution, whose range is $(D_{out} \pm ½)\cdot\Delta U$. We can then use the following model in figure 3.4, where the ADC produces an estimate with some noise (corresponding to the quantization uncertainty).

We can also study the quantization noise by adding a digital-to-analog converter (DAC), see figure 3.5.

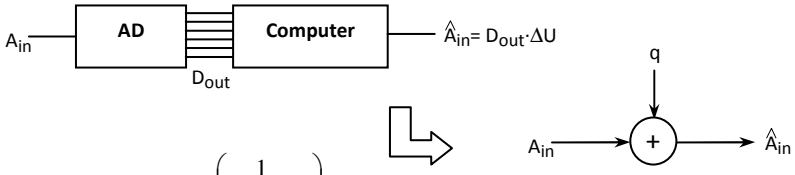


Fig 3.4 ADC model, $q \in U\left(0, \frac{1}{2}\Delta U\right)$

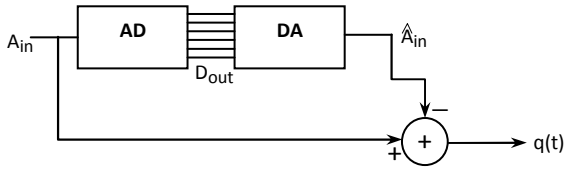


Fig 3.5 Quantization noise

In figure 3.6 we have plotted the quantization noise in the same diagram as a sinusoidal input signal $A_{in}(t)$ for an 8-bit ADC with a reference voltage of +5 volts. From figure 3.6 we can see that the size of the quantization noise will mark the limit of how small signals we're able to detect with an n-bit ADC with reference voltage U_{ref}. This thesis will later illustrate how the ADC resolution can be improved beyond this limit.

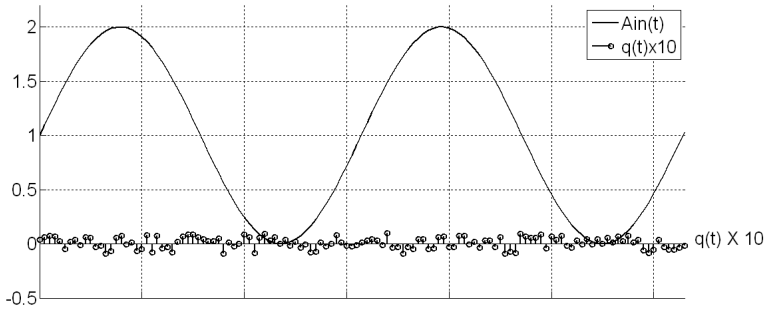


Fig 3.6 The quantization noise

3.2.3 Equivalent Number of Bits

The uncertainty of the output of an n-bit ADC is ±1/2 LSB = $\pm\Delta U/2$. For (continuous) signal levels >> q, the uncertainty can be represented by a uniformly distributed random variable, whose variance represents the noise power:

$$P_N = \sigma_q^2 = \frac{1}{\Delta U}\int_{-\Delta U/2}^{+\Delta U/2} x^2\,dx = \ldots = \frac{\Delta U^2}{12} \qquad (3.3)$$

The ADC's signal range is 0 – $2^n \Delta U$. A full scale sine wave would have an amplitude of $A = \frac{1}{2}2^n \Delta U = 2^{n-1}\Delta U$ and its power is

$$P_S = \left(\frac{A}{\sqrt{2}}\right)^2 = \frac{1}{2}A^2 = \frac{1}{2}\left(2^{n-1}\Delta U\right)^2 = \frac{1}{8}2^{2n}\Delta U^2 \qquad (3.4)$$

Hence, the signal-to-(quantization)noise ratio is

$$\text{SNR} = 10\cdot\log\frac{P_s}{P_N} = 10\cdot\log\left(2^{2n}\cdot\frac{3}{2}\right) = 20n\cdot\log 2 + 10\cdot\log(3/2) = 6.02n + 1.76 \text{ dB} \qquad (3.5)$$

For a non-perfect ADC, the SNR may be less than this. For example, from equation (3.1), we see that the reference voltage must be stable on a level less than ΔU or U_{ref} will contribute to less resolution. However, as we will see later, there are some tricks that will *increase* the SNR too.

Equation (3.5) is the SNR when we consider the quantization noise only. In a real system, other noise sources and signal distortions will decrease the SNR. This comes from converter nonlinearities, reference voltage noise, clock jitter, signal harmonics generated by the system's nonlinearities etc (Girard, 2011). All these contributions will degrade the system's *effective* resolution.

If we solve for n in (3.5) and replace the "q-based" SNR with the "real" SNR, we get the *Equivalent Number of Bits* (or *Effective* Number of Bits) (*ENOB*), of the system:

$$\text{ENOB} = \frac{\text{SNR} - 1.76}{6.02} \quad \text{bits} \qquad (3.6)$$

(ENOB is not necessarily an integer.)

3.2.4 Oversampling

In order to avoid aliasing, a signal with bandwidth f_b has to be sampled at a rate corresponding to twice the bandwidth of the signal (Nyquist, 1928; Shannon, 1949):

$$f_S > 2\cdot f_b \qquad (3.7)$$

In most digital measurement systems the signal is sampled faster than the Nyquist/Shannon limit in (3.7). The quotient $f_S/2f_b$ is called the *oversampling rate* (OSR):

$$\mathrm{OSR} = \frac{f_S}{2f_b} \qquad (3.8)$$

Apart from the obvious reason to get a better representation of the signal in time space, oversampling has other advantages too. In expression (3.3) we found that the quantization noise power is $\Delta U^2/12$. For a quantized signal sampled at f_S, all of its quantization noise is folded into the frequency range 0–$f_S/2$ (Hauser, 1991; Jarman, 1995). Hence, the spectral density of the quantization noise is

$$p = \frac{\Delta U^2/12}{f_S/2} = \frac{\Delta U^2}{12} \cdot \frac{2}{f_S} \qquad \left[\frac{\mathrm{W}}{\mathrm{Hz}}\right] \qquad (3.9)$$

The noise power in the frequency range of interest (i.e. the noise within the signal bandwidth) is $p \cdot f_b$:

$$p_b = p \cdot f_b = \frac{\Delta U^2}{12} \cdot \frac{2f_b}{f_S} = \frac{\Delta U^2}{12} \cdot \frac{1}{\mathrm{OSR}} \qquad (3.10)$$

and we see that the noise power within the signal bandwidth decreases with the oversampling rate. Hence, oversampling is advantageous from a signal-to-noise ratio point of view. This is true for *any* ADC. This is illustrated in figure 3.7.

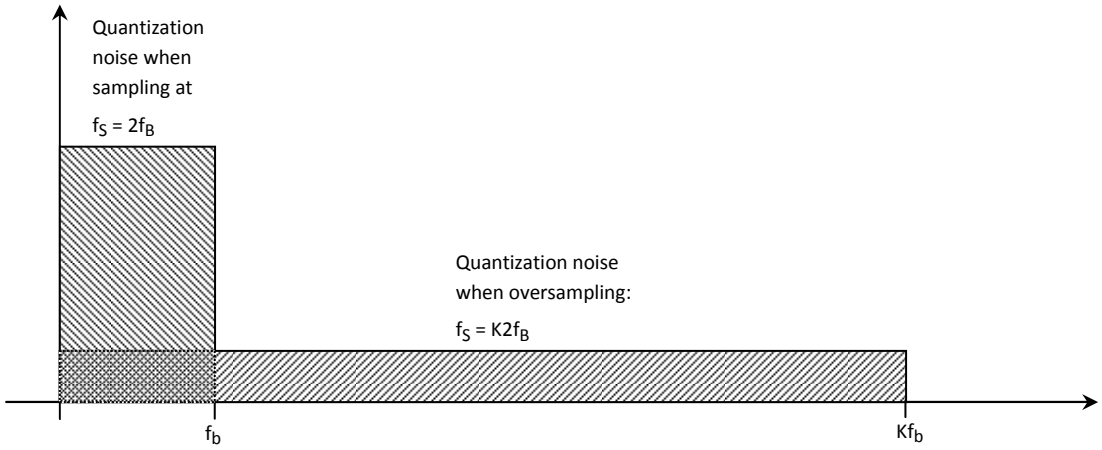


Fig 3.7 The quantization noise power is spread homogenously across the interval 0 – $f_S/2$ (Hauser, 1991)

According to Hauser (1991), the SNR of a full-scale sinusoidal oversampled by a factor of K is improved from expression (3.5) to

$$\text{SNR} = 6.02n + 1.76\ \text{dB} + 10 \cdot \log K \qquad (3.11)$$

or, if we express the oversampling rate in octaves L ($K = 2^L$), we get

$$\text{SNR} = 6.02(n + 0.5 \cdot L) + 1.76\ \text{dB} \qquad (3.12)$$

(3.12) indicates that an oversampling ADC produces an SNR that corresponds to the SNR of an $(n+0.5 \cdot L)$-bit conventional ADC sampling at the Nyquist rate. For example, an 8-bit ADC oversampling by a factor of 64 ($=2^6=4\times4\times4$) will only produce quantization noise corresponding to that of an 11-bit conventional, Nyquist-sampling ADC (i.e. quantization noise within the signal bandwidth).

Oversampling is advantageous also from another point of view; the anti-aliasing filter requirements are considerably relaxed. Consider the signal with amplitude spectrum as in figure 3.8.

If we sampled this signal with a sample rate of exactly $f_S = 2f_b$, the sampled signal's amplitude spectrum would be periodic with period $f = f_S$ (Bengtsson, 2012a; Proakis and Manolakis, 1992) as illustrated in figure 3.9.

Fig 3.8 An analog signal with bandwidth f_b

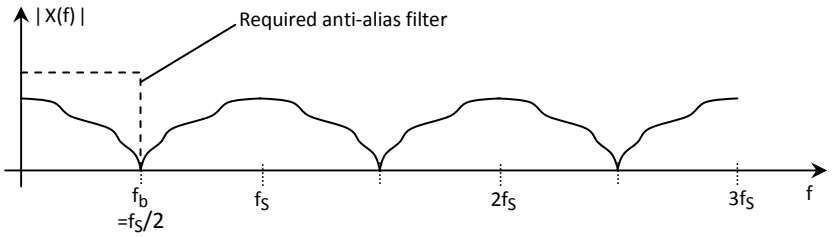


Fig 3.9 The periodic amplitude spectrum of a sampled signal

The anti-aliasing filter for this system should be a low-pass filter with a Bode diagram such that it allows all signals with frequencies up to f_b to pass without distortion and signals with frequencies above $f_b = f_S/2$ should be attenuated completely. From figure 3.9 it is obvious that this filter must have a very steep roll-off (infinitely steep, actually) and that indicates a very high-order filter. High roll-off filters typically have non-linear phase characteristics (Lynn and Fuerst, 1998; Svärdström, 1999) and hence signals within the bandwidth are most likely distorted in high-order filters. High-order filters are also complicated to design.

However, suppose instead that we *oversample* the signal. Then the sampled signal will have the spectrum in figure 3.10.

From figure 3.10 it is obvious that the anti-aliasing requirements can be relaxed considerably; the necessary roll-off is reduced and if the OSR is large enough, a first-order passive RC-filter may be all we need.

To summarize: Oversampling has so many advantages that it is used in all systems where it is possible (i.e. where the ADC and the signal bandwidth allow it).

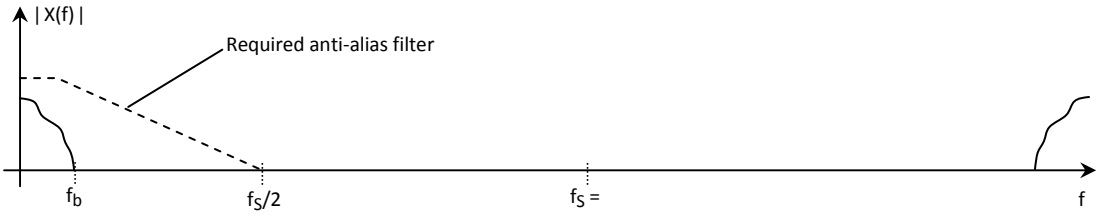


Fig 3.10 Spectrum of oversampled signal

3.2.5 Oversampling and averaging as a means for improving resolution

Assumptions: A 10-bit ADC samples a high-resolution temperature sensor at a rate of 5 S/s and has a reference voltage of U_{ref} = +5 volts; the resolution of 10 bits corresponds to a voltage resolution of 5/1024 = 4.88 mV. However, let's say that the particular application would really need a resolution of 1.0 mV.

A resolution of 1.0 mV corresponds to

$$\frac{5}{2^n} \le 1.0 \cdot 10^{-3} \qquad \Rightarrow \qquad n \ge \log_2 5000 = 12.2877..$$

i.e. we need a resolution of 13 bits. From equation (3.12) we see that this corresponds to an oversampling rate of

$$10 + 0.5 \cdot L = 13 \quad \Rightarrow \quad L = 6 \quad (\text{octaves}) \quad \Rightarrow K = 2^6 = 64 = \text{OSR}$$

Hence, if we oversample the 10-bit ADC by a factor of 64 (64·5 = 320 S/s), the in-band quantization noise equals that of a 13-bit ADC. However, the ADC still produces only a 10 bit result.

If we accumulate all 64 (2^6) 10-bit samples (with OSR = 64), we get a 10+6 = 16 bit number. Dividing this by 8 (2^3) (right-shift by three), we get our 13-bit samples which is the resolution we were looking for. This process is called *filtering and decimation*.

An excellent and instructional article about improving ADC resolution has been published by Silicon Laboratories (Silicon, 2009). In this work they state that for the above method to work, the overall noise distribution must be Gaussian (white) and have an rms that exceeds the ADC resolution (or is at least of the same order), which may not be true in "few-bit" ADCs (8-bits) where the resolution is poor enough to have all samples end up in the same histogram bin. I will comment more on that in the next section.

In the example above we increased the resolution from 10 to 13 bits, in other words, we inserted eight new levels into each quantization level in the 10-bit system.

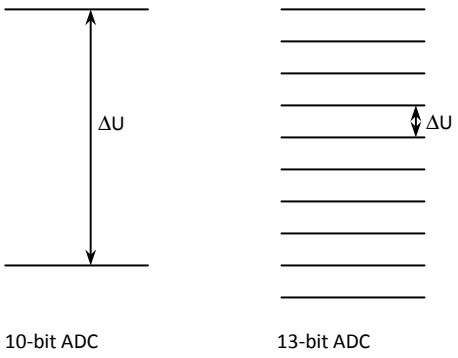


Fig 3.11 Interpolation

This is sometimes referred to as "interpolation" since we are able to read values between the original 10-bit levels.

It should also be emphasized that this resolution enhancement comes with a price; the oversampling and extra calculations required reduce the CPU throughput (the CPU bandwidth) (Silicon, 2009).

3.2.6 Dithering

Improving ENOB by oversampling and averaging only works when the ADC noise is (at least approximately) Gaussian (Candy and Themes, 1992; Lis, 1995). It may still work for non-Gaussian distributions but may be less efficient (Silicon, 2009).

In fact, some applications *intentionally inject* noise into the process to create the perfect noise distribution. This is called *dithering*[1]. For some more detailed references on the theory of dithering, see for example Kester (2006) or Schuchman (1964). Here I will only introduce the basic ideas of dithering by summarizing some parts of a work by Pohlmann (2005):

During the second World War, airplane bombers were controlled by mechanical "computers" and engineers were puzzled by the fact that the airplanes seemed to perform so much better when actually flying then what was indicated by test results in laboratory environments. They decided that this was due to the vibrations induced (by the airplane's engine) into the mechanical control system; this was beneficial since it helped reduce the errors caused by "sticky" moving parts. Because of the motor vibrations (the "noise"), the mechanical parts ran more smoothly without abrupt jerks.

This is the first known example of how a system's performance can be improved by noise injection (dithering). However, in retrospect, as pointed out by Pohlmann (2005), dithering has been used for decades by engineers; every time you tap on a mechanical meter in order to increase its accuracy, you do dithering!

To no surprise, dithering can indeed also be used to improve the performance of ADCs. The "stickiness" corresponds to the quantization levels and inserting the right noise into the system corresponds to tapping on the meter.

Suppose we have a 10-bit ADC with reference voltage +5 volts. An input DC voltage of 2.35700 volts will render a digital output number of 483 (round(2.357/0.0048828)) and we estimate the input voltage to be

$$483 \cdot \frac{5}{2^{10}} = 2.358398.. \text{ volts}$$

This corresponds to an error of 0.06%. If there is *no noise* in the system except for quantization noise, averaging multiple samples will not help to improve the accuracy of this estimation since the ADC would produce the number 483 every time.

However, if we inject some noise into the system, normally distributed with zero mean and "sufficient" variance, the ADC digital output number would vary randomly with the same pdf (probability density function) as the noise but with a

[1] "dither" = shiver, tremble; to act nervously or indecisively (Merrimam-Webster Dictionary)

mean that is centered on the true sample value.

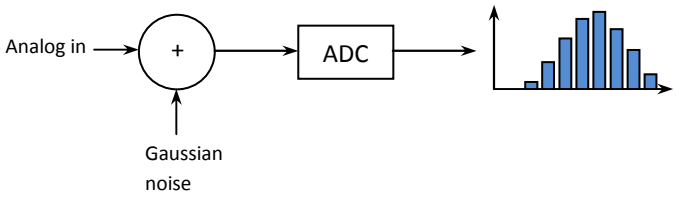


Fig 3.12 Adding noise can improve the accuracy

Suppose we add zero mean Gaussian noise with a standard deviation equal to twice the size of the quantization noise ($2 \cdot 5/2^{10} \approx 0.01$ V) and take 200 samples of this "dithered" signal. This can be simulated in MATLAB:

```
>> dith_signal=2.357+0.01*randn(1,200);          %generate 200 samples
```

The ADC would divide these numbers by the ADC's resolution and round the samples to the nearest integer:

```
>> delta=5/1024;                                 %ADC resolution
>> ADC_out=round(dith_signal/delta);             %ADC integers
>> hist(ADC_out)                                 %plot histogram
```

This would produce a histogram plot of the ADC's output integers, see figure 3.13.

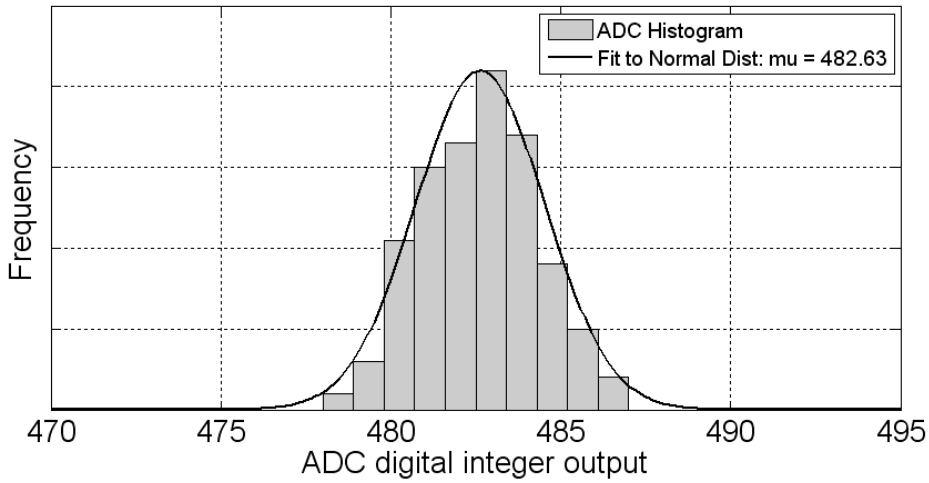


Fig 3.13 Histogram of dithered signal

Averaging these 200 samples and converting back to a voltage number gives us a new estimate of the input voltage level:

```
>> mean(ADC_out)*delta
ans =
  2.356542968750000
```

an error of only 0.02%.

3.3 ΣΔ ADCs

One of the first experiments reported in this thesis concerns the implementation of a ΣΔ ADC in a microcontroller and the theory of ΣΔ ADCs are covered in more detail.

3.3.1 Background

ΣΔ ADCs differ significantly from the other ADC techniques. The output of a conventional ADC like the SAR ADC, is always a numerical number (serial or parallel) representing the sample value. This is called *pulse code modulation* (PCM) (Beiss, 2007). A ΣΔ ADC can produce that too, but primarily it produces a *pulse proportion modulated* (PPM) signal (Beiss, 2007). This is a bitstream of 1s and 0s and the 1s' *density* is proportional to the sample level. A post-processing, digital averaging filter will sample this bitstream and convert it into a conventional PCM number, but primarily ΣΔ ADCs produce a bit stream whose density of 1s is proportional to the input signal level. The advantage of ΣΔ ADCs is that the resolution is increased significantly, but this is at the expense of conversion speed; they are typically much slower than SAR ADCs (Soldera et al, 2005).

A 1st order ΣΔ ADC is illustrated in figure 3.14 (Jarman, 1995). If we disregard the digital filter for now, it has four components; an analog summing circuit, an integrator, a comparator and a 1-bit DAC. The output of the comparator will be a stream of logic 1s and 0s that will be sampled by the digital filter. The 1-bit DAC produces either $+U_{ref}$ or $-U_{ref}$, depending on the comparator's output.

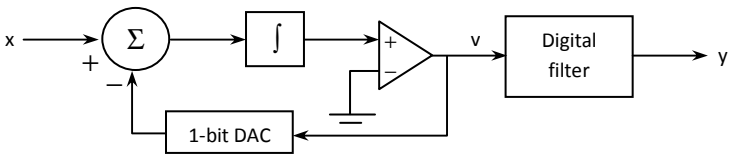


Fig 3.14 A first order ΣΔ ADC (Jarman, 1995)

The ΣΔ ADC has been described by several (Beiss, 2008; Hauser, 1991; Jarman, 1995; MAXIM, 2003; Paikin, 2003; UBICOM, 2000; Vu, 2005; Wender and Ihme, 2007) but my favorite is Kester (2009). The following description is mainly a summarizing of Kester's (2009) outlining:

The comparator's output is fed back to the input summing circuit via the 1-bit DAC. Even though the 1-bit DAC output is always only $+U_{ref}$ or $-U_{ref}$, the consequence of the negative feedback loop is that the *average* output of the DAC will equal U_{in} (= x). If the input signal level increases, so must the average value of the DAC output. Since the DAC output is generated by the 1s and 0s in the comparator's bitstream output, the density of 1s in the bitstream will increase as the input signal level increases; the density of 1s in the comparator's output will be proportional to the input signal level. (End of Kester's (2004) description.)

(For an interactive tutorial on ΣΔ ADCs, see Analog (2011a).)

Hence, all we need to do to produce a PCM ADC number is to sample the analog bitstream from the comparator in an averaging digital filter:

$$y_n = \frac{1}{N}(v_n + v_{n-1} + v_{n-2} + ...) \qquad (3.13)$$

I will talk more about this filter later. It turns out that it is not just a filter, it is also a *decimator*, that will be used to reduce the sampling speed. This is necessary in order to get a useful ADC resolution and it is possible because in ΣΔ ADCs, the signal is always *oversampled* (by maybe as much as a factor of 1000 or more).

3.3.2 Theory

Maybe the most "cunning" advantage of the ΣΔ ADC is its inherent property of *shaping* the quantization noise in such a way that it is "pushed" towards higher frequencies. In fact, the oversampling and the noise shaping in combination, has such a positive effect on the SNR that a ΣΔ ADC will be able to produce an SNR larger than indicated by expression (3.5) which is the theoretical limit of a conventional ADC sampling at the Nyquist rate. We've already seen that oversampling and dithering can do that too. Here we will show that the ΣΔ ADC can increase the SNR even more by noise shaping.

To understand the noise shaping, we need to return to figure 3.14 and re-draw it. First of all, we idealize the DAC component and replace it with a transfer function $H(s) = 1$. The integrator's transfer function is 1/s (Bengtsson and Karlström, 2007). We may consider the comparator to be a 1-bit ADC; the comparator output is a (rough) digitalized estimate of the input. We may model any ADC as an estimate of

the input plus some quantization noise (Kester, 2009); the ADC (i.e. the comparator) may be replaced by a circuit adding noise equal to the quantization noise of the ADC. This gives us the signal model in figure 3.15.

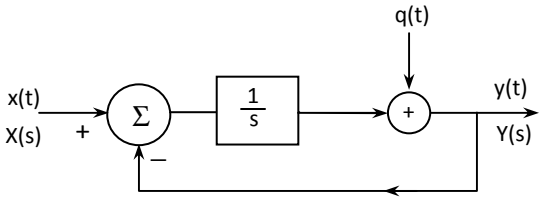


Fig 3.15 A simplified model of a first order ΣΔ ADC

This looks simple enough, but it is a *very* clever system! To see that, we need to figure out what it does both to the signal x and to the noise q. Let's start with the signal; to see what happens to the signal, we temporarily cancel the noise ($q(t) = 0$). That gives us figure 3.16.

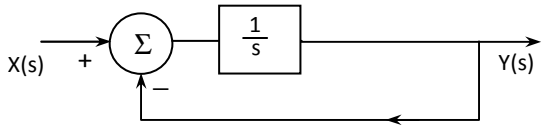


Fig 3.16 To see what happens to the signal, we cancel the noise

The system's transfer function is easily calculated:

$$Y(s)=\frac{1}{s}(X(s)-Y(s)) \qquad \Rightarrow \qquad sY(s)=X(s)-Y(s)$$

$$(1+s)Y(s)=X(s) \qquad \Rightarrow \qquad H(s)=\frac{Y(s)}{X(s)}=\frac{1}{s+1} \qquad (3.14)$$

From (3.14) we can see that as far as the input signal is concerned, the system is a first order low-pass filter.

In order to see how the system treats the noise, we cancel the input signal:

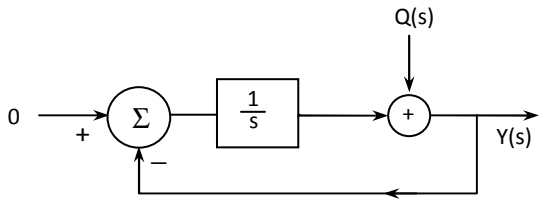


Fig 3.17 To see what happens to the noise, we cancel the signal

The system transfer function is now:

$$Y(s) = Q(s) - \frac{1}{s}Y(s) \qquad \Rightarrow \qquad sY(s) = sQ(s) - Y(s)$$

$$sQ(s) = Y(s)(s+1) \qquad \Rightarrow \qquad H(s) = \frac{Y(s)}{Q(s)} = \frac{s}{s+1} \qquad (3.15)$$

Look at (3.15). As far as the *quantization noise* is concerned, the system is a high-pass filter; the noise is pushed to higher frequencies! The system low-pass filters the signal and high-pass filters the noise. That means that an even larger part of the noise will be attenuated by a low-pass filter and that even less noise ends up within the signal's bandwidth. Now we need to go back to figure 3.7 and add the noise spectrum after it has passed the ΣΔ ADC. This is illustrated in figure 3.18.

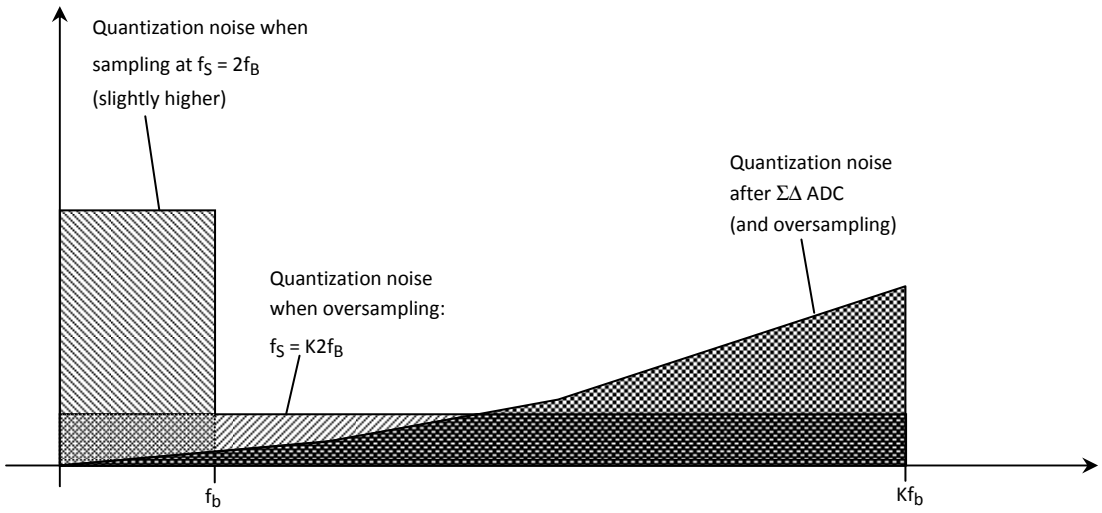


Fig 3.18 The noise is "shaped" by the ΣΔ ADC's high-pass property (Hauser, 1991)

Notice in figure 3.18 how the noise remaining within the signal bandwidth region has been reduced even more. Now we can understand why ΣΔ ADCs are able to offer SNR figures exceeding that of conventional ADCs represented by expression (3.5) According to Hauser (1991) the SNR of a first order ΣΔ ADC is

$$\text{SNR} = 6.02(n + 1.5 \cdot L) - 3.41\,\text{dB} \qquad (3.16)$$

We showed earlier that an 8-bit ADC with OSR = 64, produced quantization noise corresponding to that of an 11-bit Nyquist-sampling ADC. If we have an 8-bit ΣΔ

ADC, oversampling by a factor of 64 (2^6), according to (3.16) it will produce an SNR equal to

$$6.02(8+1.5\cdot 6)-3.41=98.93 \quad \text{dB}$$

Inserting this SNR value into (3.6) gives us an effective number of bits of 16.1! This is what oversampling in combination with the clever ΣΔ circuitry in figure 3.14 does to the ENOB.

3.3.3 Method

General purpose, low-bit microcontrollers almost always have SAR ADCs (if any ADC at all). ΣΔ ADCs are typically found in digital signal processing (DSP) controllers aimed at sound processing. You may think that the ΣΔ architecture in figure 3.14 would be hard to implement into a microcontroller due to the analog parts (analog summing, integrator, comparator) and that may be one of the reasons why they are usually not integrated on-chip in typical microcontrollers, but the main reason is that you don't need to integrate them on-chip; if only the controller is equipped with an analog comparator, the ΣΔ ADC is easily synthesized with only a few passive components (and some firmware, of course). This ΣΔ synthesizing is illustrated in figure 3.19.

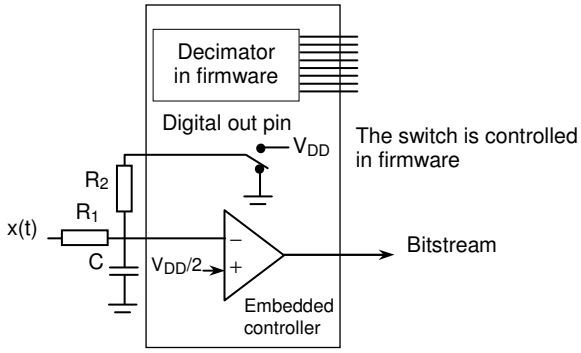


Fig 3.19 Implementing a ΣΔ ADC in embedded controllers with an analog comparator

There are several microcontrollers equipped with analog comparators from many manufacturers (Atmel, 2011; Freescale, 2009; Microchip, 2003). It is not clear who first suggested the ΣΔ ADC in figure 3.19, but one of the first was Peter et al (1998). Similar ideas have later been implemented and reported by several others (Soldera, 2005; STMicroelectronics, 2008; Weber and Windish, 2007). The capacitor *C*

corresponds to the integrator in figure 3.14 and the 1-bit DAC is replaced by firmware; the digital output pin is set/reset in firmware in accordance to the comparator's output (Peter et al, 1998). When the digital output pin is set, the capacitor is charged and the potential on the comparator's input increases. When it reaches the $V_{DD}/2$ level the comparator output will go low and the digital output pin will be reset. That will discharge the capacitor and when it is less than $V_{DD}/2$ the comparator will trip again and the digital output pin is set. This feedback system will force the comparator's negative input to equal $V_{DD}/2$ and the *lower x* is the more *ones* will be necessary. The density of *ones* on the digital output pin *increases* when *x decreases*; the density of 0s at the comparator output will be proportional to the input signal level.

The digital filter is implemented simply by counting the number of 0s during a fixed period of time. Figure 3.20 illustrates the software for an 8-bit ΣΔ ADC.

Notice in figure 3.20 that the "1-bit" sampling rate is determined by the loop time and that this sampling rate will be decimated by a factor of 256 in order to produce an 8-bit resolution result; this program samples, filters and decimates inherently.

The range of the ΣΔ ADC in figure 3.19 is determined by R_1 and R_2 (and V_{DD}, of course). The range is always centered round the comparator's reference voltage, which is $V_{DD}/2$ in figure 3.19. If we have a +5 V powered system, it is centered around +2.5 volts. The maximum and minimum readable input levels are, respectively (Peter et al, 1998):

$$U_{in,\max} = \left(1 + \frac{R_1}{R_2}\right) \cdot \frac{V_{DD}}{2} \qquad (3.17)$$

$$U_{in,\min} = \left(1 - \frac{R_1}{R_2}\right) \cdot \frac{V_{DD}}{2} \qquad (3.18)$$

The absolute values of R_1, R_2 and C must be chosen such that C is not saturated at any time for the particular sampling rate chosen.

For example, if we have a +5 volt powered microcontroller and want a range of 1-4 volt, we need $R_1/R_2 = 0.6$. If we only use standard EIA (Electronics Industries Association) resistor value from the E12 series, we use R_1 = 100k and R_2 = 180k. In figure 3.21 we have implemented a ΣΔ ADC in a PIC18F458, an 8-bit microcontroller from Microchip (Microchip, 2003).

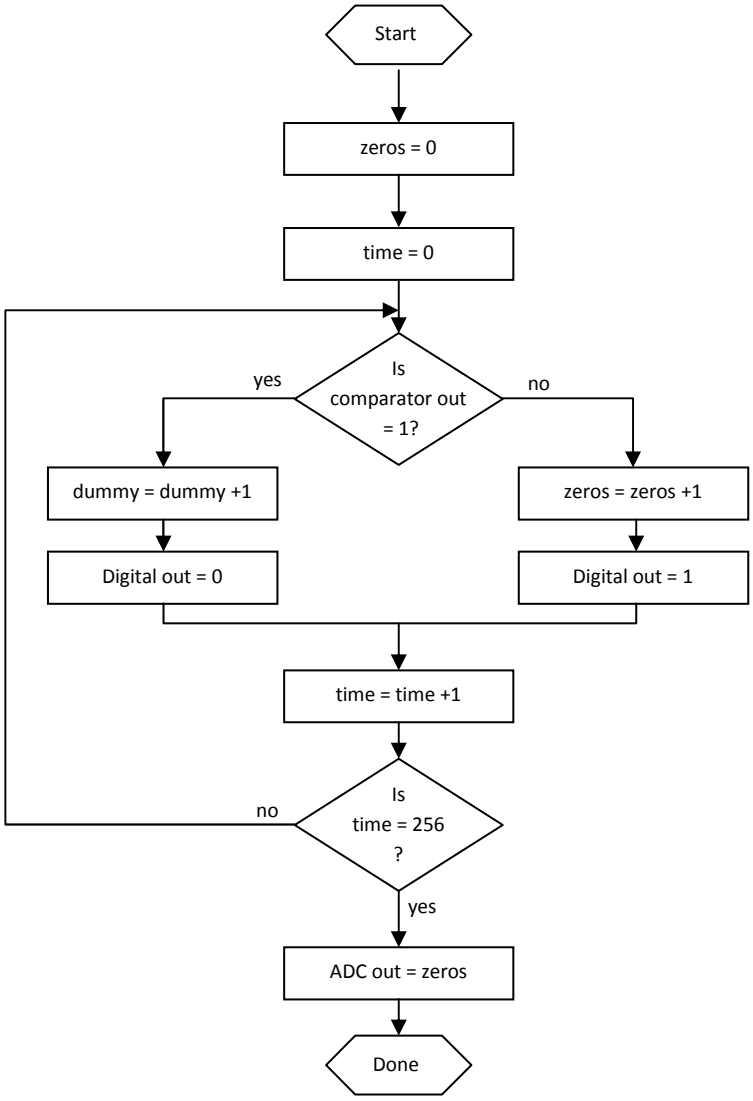


Fig 3.20 Flowchart of ΣΔ ADC for microcontroller implementation (incrementing the dummy only eats clock cycles but is necessary in order to make all program branches equally long (Peter et al, 1998)).

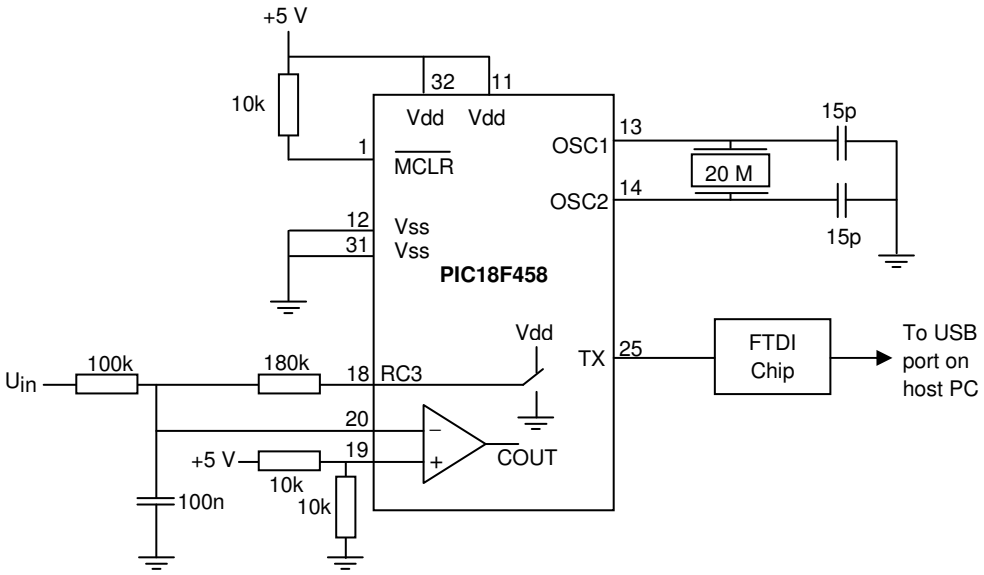


Fig 3.21 A detailed implementation of a ΣΔ ADC in a PIC18 microcontroller

The result is transmitted via an asynchronous serial link to a USB port of a host PC. The entire (non-generic) code is presented here:

```
////////////////////////////////////////////////////////////////////////////////
//
// September 14, 2011, by Lars Bengtsson
//
// Sigma-Delta ADC with analog comparator: v. 1.3
//
////////////////////////////////////////////////////////////////////////////////
#include <pic18.h>
#include <delay.c>

char dummy,zeros;
int i;

void main() {
        TRISC=0x80;                             //RC6 = TX
        SPBRG=129;                              //Initiate UART
        BRGH=1;
        SPEN=1;
        TXEN=1;
        CMCON=0x02;                             //Configure comparator
        while(1) {
                zeros=0;
                for (i=0;i<256;i++) {           //This for loop determines the 1-bit sample rate
                        if(C1OUT) {
                                RC3=0;
                                zeros++;
                        }
```

```
                else {
                        RC3=1;
                        dummy++;
                }
        }
        PORTB=zeros;
        while(!TRMT);                   //The following code is for the UART output
        TXREG=0x0A;
        while(!TRMT);
        TXREG=0x0D;
        while(!TRMT);
        TXREG=(zeros/100+0x30);
        zeros=zeros%100;
        while(!TRMT);
        TXREG=(zeros/10+0x30);
        while(!TRMT);
        TXREG=(zeros%10+0x30);
    }
}
```

3.3.4 Empiri

Figure 3.22 shows experimental results from the ADC's response.

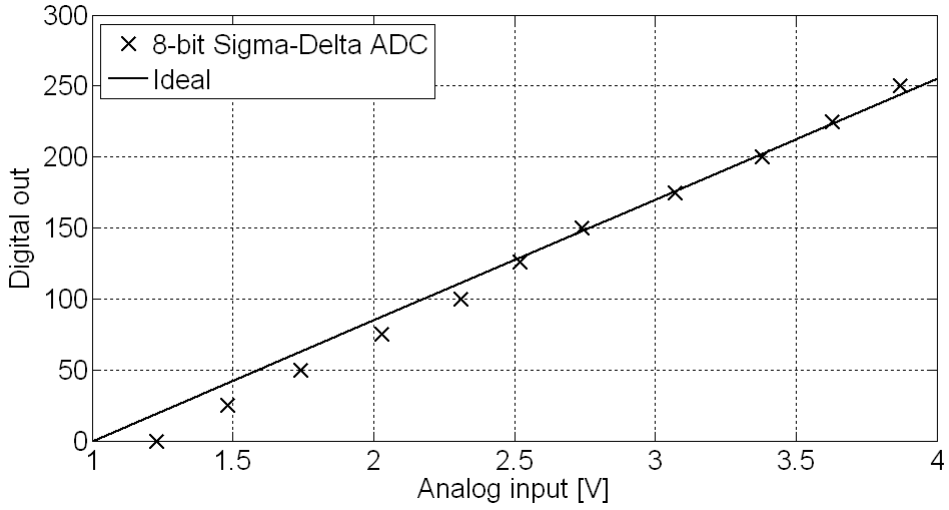


Fig 3.22 Characteristics of implemented ΣΔ ADC

It is also illustrative to perform a spectrum analysis on the bit stream itself. With the setup in figure 3.21, a low-frequency sinusoidal was applied to the input and the bitstream was recorded by using a digital oscilloscope (Tektronix TDS5032) and transferring the data to MATLAB. Figure 3.23 shows the spectrum of the bitstream.

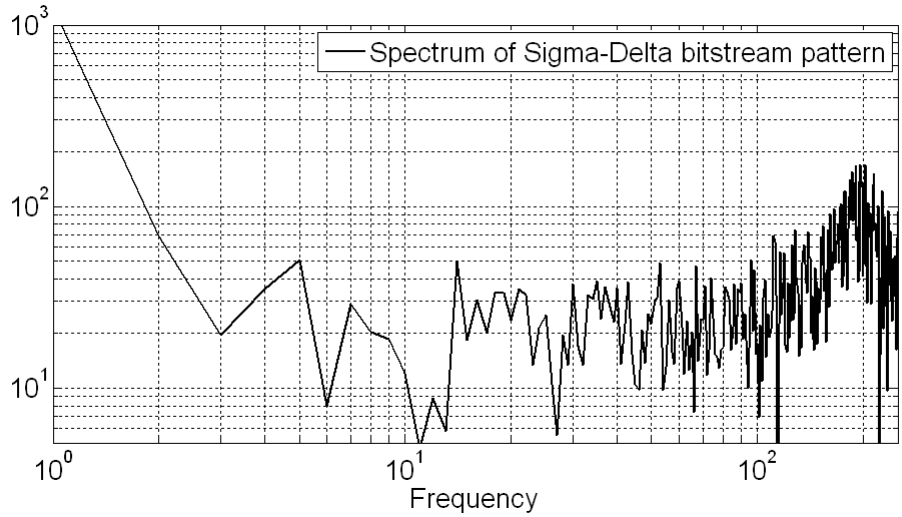


Fig 3.23 The noise is high-pass filtered by the ΣΔ design

With the same experimental setup, the ΣΔ ADC performance was compared with the performance of the embedded SAR ADC of the PIC18F458. A sinusoidal with frequency 135 Hz was sampled both with the ΣΔ ADC in figure 3.21 (OSR ≈ 100) and with the embedded SAR ADC (OSR: just slightly above the Nyquist rate). In figure 3.24 we can see the result, where also the experimental results of a second order Sigma-Delta ADC (Kester, 2008) is included.

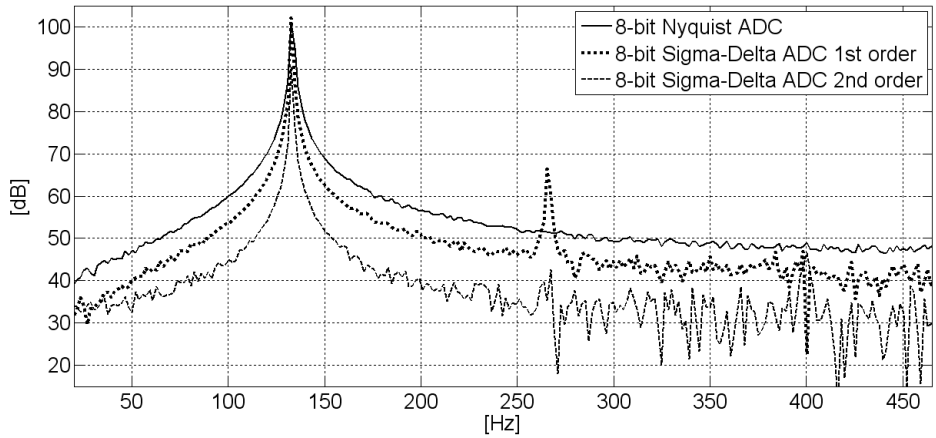


Fig 3.24 Performance comparison of ADCs

It is also illustrative to plot the input signal in the same diagram as the digital signal on the digital output pin (RC3 in figure 3.21), see figure 3.25.

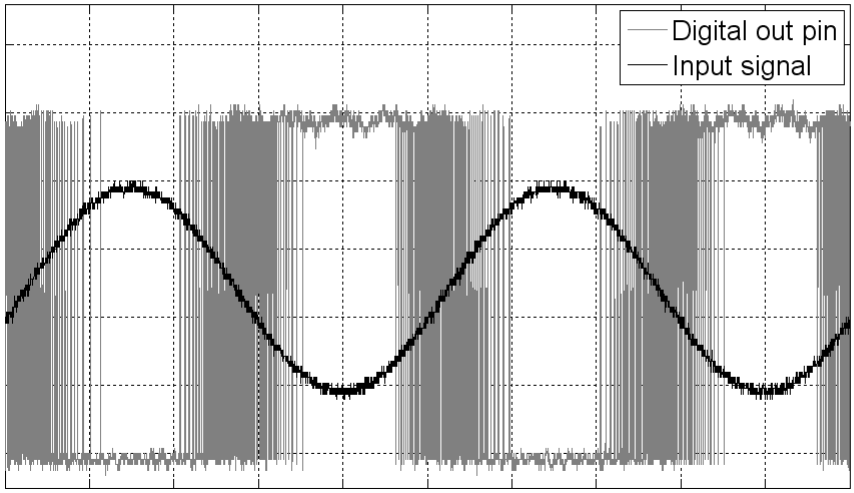


Fig 3.25 The bitstream density compared to the signal level

3.3.5 Discussion

From figure 3.22 we can see that the range of the ΣΔ ADC in figure 3.21 agrees very well with the expected range (but the linearity is not quite as good as has been reported by others (Soldera, 2005)).

Figure 3.23 clearly illustrates the ΣΔ ADC's ability of shaping the noise; notice in figure 3.23 how the noise has been shifted to higher frequencies. The noise suppression in figure 3.24 is less than predicted, but this is most likely due to the presence of other noise sources than quantization noise. (The extra peaks at 270 and 405 Hz are higher order harmonics from the signal generators. These peaks are not present in the SAR-Nyquist spectrum because it was not recorded at the same occasion and a different signal generator was used when recording that data.)

Notice in figure 3.25 how the bitstream density varies with the signal level. In figure 3.25 the density of 0s *increases* with the signal level as expected.

(When the data in figure 3.25 was recorded, the code segment transmitting data to the USB port of the host computer was removed in order to get an uninterrupted signal sequence.)

3.4 New interface proposal

3.4.1 Background

In this thesis I will suggest that sensors that generate analog voltage signals can also be "directly" interfaced to digital controllers with only a handful of passive components. I figure 3.19 and 3.21 we solved this by implementing a ΣΔ ADC, using two I/O pins and a few passive components. The disadvantage of that solution is that the digital controller must have an integrated comparator. This suggests a more advanced (more expensive) controller and excludes most CPLD/FPGA systems. The following proposed solution will solve that problem.

3.4.2 Method

Figure 3.26 below illustrates how analog voltage signals can be interfaced to any digital controller (i.e. even those not equipped with an internal comparator).

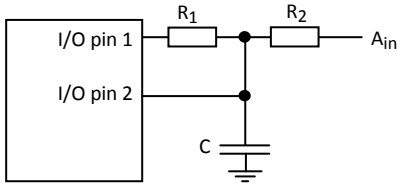


Fig 3.26 Simple analog-to-digital converter (Bengtsson, 2012b)

This circuit also requires that the embedded system's I/O-ports have tri-state capability. The capacitor is first charged to A_{in} by configuring the I/O-pins as High-Z inputs. Depending on the voltage level of A_{in}, I/O-pin 2 will reach a logic high level (V_{IH}) or not. If C is charged to a level higher than V_{IH}, then the capacitor is discharged through R_1 to V_{IL} (by configuring I/O-pin 1 as digital out, logic low) and the discharging time is proportional to A_{in}. If the charging of C does not reach the V_{IH} level on I/O-pin 2, we instead measure the time it takes to charge it all the way up to the V_{IH} level (i.e. from A_{in} to V_{IH}) by configuring I/O-pin 1 as digital out, logic high. In the latter case, the charging time is proportional to $V_{IH} - A_{in}$.

This provides all the hardware necessary for an n-bit ADC with a range of 0 – V_{DD} and a (theoretically) arbitrary resolution; the resolution is determined by the number of bits of the counting variable in firmware, the counter's speed and the R_1C time constant in figure 3.26.

3.4.3 Theory

When the capacitor has been charged to A_{in}, we have one of two possible situations; either the voltage potential on I/O-pin 2 is greater than (or equal to) V_{IH} (input high level) or it is not. In software we have a counting variable which is initiated to a start value N_0 and if $A_{in} \geq V_{IH}$, this variable is *incremented* until C is *discharged* to V_{IL}. If, on the other hand, $A_{in} < V_{IH}$, the counting variable is *decremented* until C is *charged* to V_{IH}. The charging/discharging is implemented by reconfiguring I/O-pin 1 to output and setting it high or low.

These two possible situations are illustrated in figures 3.27 and 3.28.

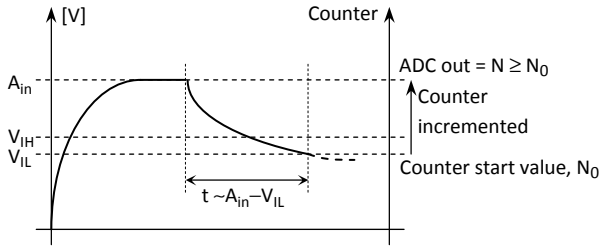


Fig. 3.27 Case 1: $A_{in} \geq V_{IH}$

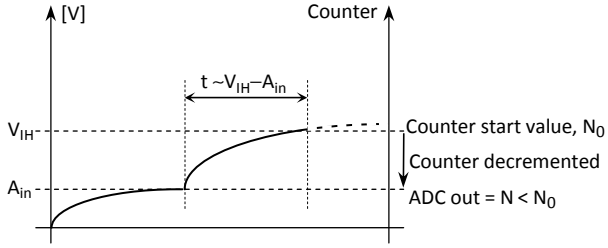


Fig. 3.28 Case 2: $A_{in} < V_{IH}$

This system is analyzed in detail in paper II appended to this thesis (Bengtsson, 2012b). In paper II, I propose that the discharging and charging times respectively, are given by expressions (3.19) and (3.20) below:

$$t_d = -\frac{1}{\frac{1}{C}\left(\frac{1}{R_1}+\frac{1}{R_2}\right)} \times \ln\left\{\frac{1}{R_1}\left(\frac{V_{IL}(R_1+R_2)}{A_{in}} - R_1\right)\right\} \qquad (3.19)$$

$$t_c = -\frac{1}{\frac{1}{C}\left(\frac{1}{R_1}+\frac{1}{R_2}\right)} \times \ln\left\{\frac{1}{R_1}\left(\frac{V_{IH}(R_1+R_2)-R_2V_{DD}-R_1A_{in}}{R_2(A_{in}-V_{DD})}\right)\right\} \qquad (3.20)$$

where t_d is the discharging when $A_{in} \geq V_{IH}$ and and t_c is the charging time when $A_{in} < V_{IH}$.

Figure 3.29 illustrates the firmware.

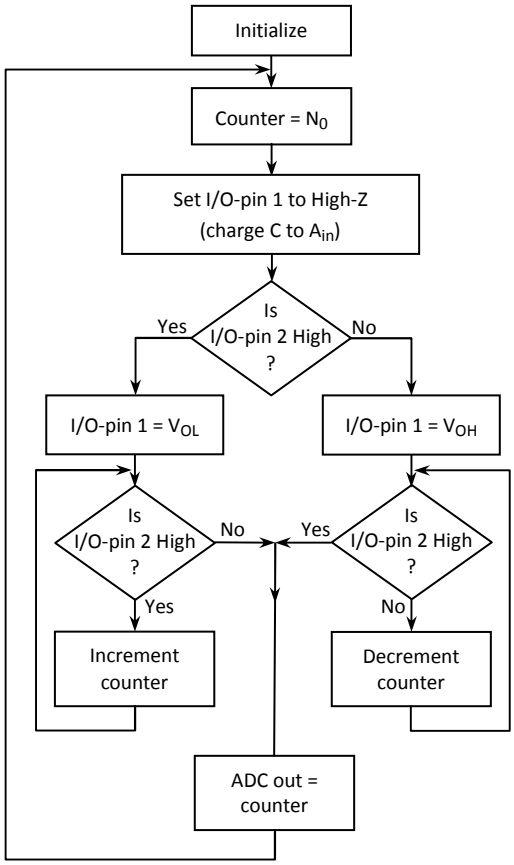


Fig. 3.29 The firmware flowchart

Notice in figure 3.29 that the counter is initiated to N_0 and incremented or decremented depending on whether $A_{in} \geq V_{IH}$ or not, and that the counter increment/ decrement rate corresponds to the loop time t_{loop} in figure 3.29. If N_c is the final count number during charging and N_d is the final count number during discharging then

$$N_c = N_0 - \frac{t_c}{t_{loop}} \qquad (3.21)$$

$$N_d = N_0 + \frac{t_d}{t_{loop}} \qquad (3.22)$$

This relates the counter value to the input analog voltage A_{in}.

3.4.4 Empiri

In figure 3.30 I have plotted the theoretically predicted response (N vs A_{in}) in the same diagram as experimental data for the case when R_1 = 2191 Ω, R_2 = 10023 Ω and C = 2.18 µF. The proposed design in figure 3.26 was implemented in a PIC18F458 microcontroller (Microchip, 2003) with the following parameter values; V_{IL} = 1.2730 V, V_{IH} = 1.2812 V and V_{DD} = 5.0280 V.

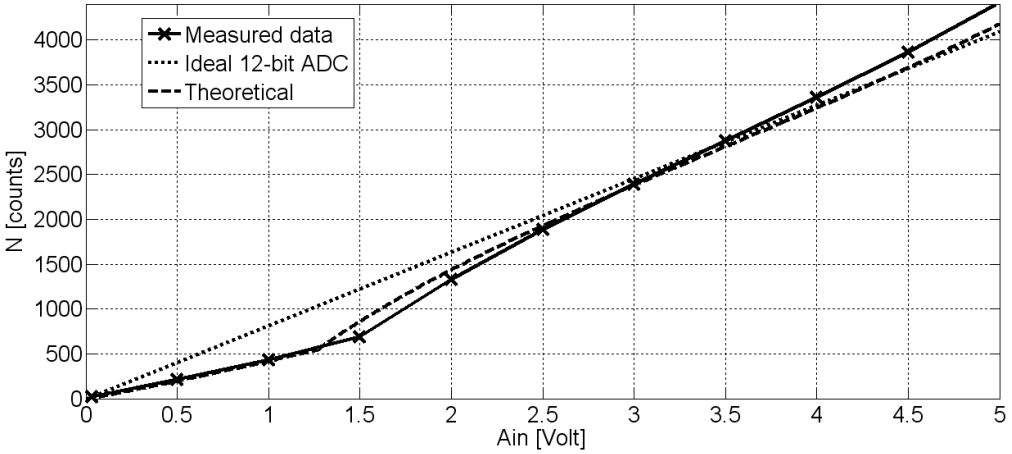


Fig. 3.30 R_1 = 2.2 kΩ, R_2 = 10 kΩ, C = 2.2 µF (Bengtsson, 2012b)

3.4.5 Discussion/Conclusions

The plots in figure 3.30 verify that experimental results are consistent with theoretical predictions. The relationship between the analog input and digital output deviates some from that of an ideal ADC, but is predicted in theory. The advantage of this solution (compared to the ΣΔ ADC in figure 3.19) is that it doesn't require an analog comparator; it can be used as an ADC interface for inherently digital targets

such as FPGAs/CPLDs. The disadvantage is that it is not quite linear and it does not possess the noise shaping property of the ΣΔ ADC in figure 3.19.

Paper II appended to this thesis contains a complete theoretical analysis of this design as well as design rules and an uncertainty analysis.

3.5 Conclusions

In this chapter I have demonstrated most of all that for “moderate” sampling rates, analog voltage and current signals can be interfaced to digital systems even if they do not have an on- (or off-) chip ADC. It requires only tri-state I/O-pins and a few passive components (R,C). The general idea is to let the input signal charge a capacitor and then measure the time it takes to discharge (or charge) it, alternatively measure the rate at which you must charge it in order to sustain a certain level.

Chapter 4

Direct Sensor-to-Controller Interfaces

4.1 Introduction

During the last decade, there's been a trend towards interfacing sensors directly to microcontrollers (Cox (1997); Merritt, (1999); Custodio et al (2001b); Jordana et al (2003); Reverter et al (2005); Lepkowski (2004); Reverter and Pallàs-Areny (2006)). The major advantage is that this reduces the overall cost of the acquisition system. The main idea is to transfer the analog signal variation caused by the sensor, into a *quasi* digital signal that can be measured by one of the controller's embedded timers. A variation in an analog voltage is transferred into an analog variation in either frequency, time duration or duty cycle ("quasi digital") (Reverter and Pallàs-Areny, (2005); Viorel, (2006)). I will treat quasi-digital signals in more detail in chapter 5.

Exactly how this direct sensor interfacing is implemented depends on the sensor, whether it is resistive, capacitive or part of a Wheatstone bridge or not. However, in all cases, the general idea is to time the charging (or preferably the *dis*charging) of a capacitor, similar to the idea of the ADC design in chapter 3. The key trick is the fact that embedded systems' digital I/O-ports can be set to *three* different states. When configured as output, they are either high (sourcing) or low (sinking) and when configured as inputs, they represent high-impedance inputs (HighZ). In microcontrollers this is configured in the I/O-ports' *data direction* registers (TRIS registers in Microchip controllers, (Microchip, 2003)). In VHDL programmed targets, like CPLDs/FPGAs, I/O pins declared as "std_logic" can have no less than eleven states (Hamblen et al, 2008; Pedroni, 2004); '1', '0' and nine more. Of the other nine states, typically only one is synthesizable; the 'High-Z' state. This makes CPLDs and FPGAs potential targets for direct sensor-to-embedded systems applications. We took advantage of this in chapter 3, both in the proposed ADC design in figure 3.26 and in the interfacing of current generating sensors in figure 3.31. Reverter (2012) is an excellent review article on this subject.

4.2 Resistive sensors

4.2.1 Background/theory

Typical resistive sensors are for example RTDs (Resistive Thermal Devices) like the Pt-1000 sensor.

Figure 4.1 illustrates the basic idea of how a resistive sensor is interfaced directly to a controller.

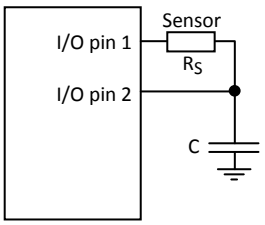


Fig. 4.1 Direct interfacing of resistive sensor

During the first stage, I/O pin 2 is configured as output and set high while I/O pin 1 is configured as input (high impedance). I/O pin 2 will charge the capacitor to V_{OH} (I/O pin output high level). During the second stage, the pins are reconfigured; pin 2 becomes high impedance and pin 1 is configured as an output pin and set low. Hence the capacitor will discharge through the sensing element's resistance R_S. The discharging continues until the voltage on pin 2 reaches the threshold level for input logic low (V_{IL}). Figure 4.2 shows the charge/discharge timing diagram on I/O pin 2.

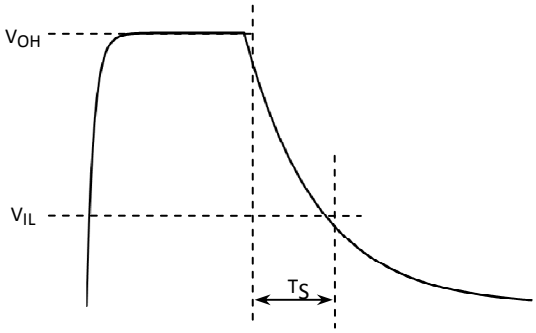


Fig. 4.2 Charging/Discharging at I/O-pin 2

An internal timer measures the discharging time T_S in figure 4.2, resulting in an integer N_S which is proportional to T_S. This number is proportional to the resistance of the sensing element (Merritt, 1999):

$$R_S = kN_S \qquad (4.1)$$

where the constant k depends on the trigger level threshold of the embedded timer, the capacitor C and also on the controller's speed and timer setup parameters (clock rate, pre-scale).

Cox (1997) was one of the first to suggest how to interface resistive sensors directly to microcontrollers, without the use of intermediate ADCs. He proposed an extra calibration resistor R_C and a "protection" resistor, as illustrated in figure 4.3.

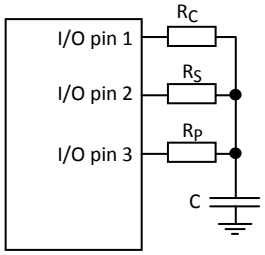


Fig. 4.3 The one-point calibration technique (Cox, 1997)

By timing the discharging of C both via the sensor R_S and via the calibration resistor R_C, the unknown sensor resistance R_S can be estimated as a quotient (Cox, 1997):

$$\hat{R}_S = \frac{T_S}{T_C} R_C = \frac{N_S}{N_C} R_C \qquad (4.2)$$

where N_S and N_C are the integer numbers produced by the controller's timer during the discharging through R_S and R_C, respectively.

Compared to Eq. (4.1), the expression is now independent of the capacitor C. Still, the *overall performance* is *not* independent of C (Jordana et al (2003); Reverter and Pallàs-Areny (2006)). The measurement time is of course increased, but the introduction of a calibration cycle removes/reduces all first order errors in offset, gain and temperature (Cox, 1997). Cox (1997) also suggested a protection resistor R_P, in order to protect the sinking I/O pin from ESD (Electro-Static Discharge) related breakdowns.

When it comes to selecting component values, Cox (1997) suggested that you should pick an R_C value that is one half of the maximum expected R_S, R_P should be a "small" resistor (100-200 ohms) and the charging capacitor should be

$$C = \frac{-T}{R_S \cdot \ln\left(1 - \frac{V_T}{V_R}\right)} \qquad (4.3)$$

where T is the charging time necessary to get the desired resolution and V_R and V_T are the start and stop levels of the charging process, respectively. Reverter et al (2005b), would later show that the R_P resistor is also necessary in order to reduce power supply interferences.

Previous to Cox (1997), Bierl (1996) had suggested a two-point calibration technique, with one extra calibration resistor:

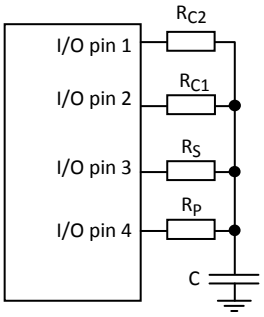


Fig. 4.4 The two-point calibration technique (Bierl, 1996)

In this case, the sensor's resistance is estimated from (Bierl, 1996)

$$\hat{R}_S = \frac{N_S - N_{C1}}{N_{C2} - N_{C1}} \cdot (R_{C2} - R_{C1}) + R_{C1} \qquad (4.4)$$

Reverter et al (2005a) would later analyze both the one- and two-point calibration technique and conclude that the accuracy of the R_S measurement in both cases depends on the I/O-ports' internal resistances R_i. However, in the one-point calibration case, the error is of the order of R_i, while in the two-point calibration case, the error is of the order of ΔR_{ij}, i.e. the *difference* in internal resistance of the I/O-pins. Hence, from an accuracy point of view, the two-point calibration method is preferred (but the measurement time increases and hence the system bandwidth is reduced). As far as precision (uncertainty) is concerned, Reverter et al (2005a) concluded that the one- and two-point calibration techniques yield similar results. The same work also suggested a "three-signal" calibration technique, which basically has the same performance as the two-point calibration technique but has better cost-performance ratio, since only one calibration resistor is required.

In 2001, Custodio et al (2001a) published a work that analyzed the importance of selecting the right calibration resistor in more detail. They analyzed in detail the influence of the microcontroller's internal parameters on the uncertainty of the discharging time measurement. The parameters included not only the current leakage of the I/O pins, but also the influence of non-ideal input/output resistances. Their model of the microcontroller's I/O pins showed that there are accuracy errors proportional to the I/O pin's output resistance. They also suggested gain errors if the two I/O pins' output parameters were not identical, and also quantified the non-linearity errors. One conclusion was that all these errors only existed when $R_C \neq R_S$ and suggested that R_C is chosen as close to the midrange value of R_S as possible (which agrees with Cox's (1997) suggestion, even if Cox's motivation was less detailed). The selection of C, suggested by Custodio et al (2001a), is that if you want an n-bit resolution representing the maximum and minimum values of R_S, then

$$C > \frac{2^n}{f\Delta R_S \cdot \ln\left(\frac{V_{OH} - V_{OL}}{V_{OH} - V_{IH}}\right)} \qquad (4.5)$$

where f = timer's counting rate, $\Delta R_S = R_{S,max} - R_{S,min}$, V_{OH} is the output voltage of an output pin set high, V_{OL} is the output voltage of an output pin set low and V_{IH}, is the high level threshold of an input pin.

Reverter et al (2003a) reported that the program code influences the power consumption of the microcontroller. Certain code instructions (jump and call) cause spike interferences in the power supply. Since the I/O pins' input trigger levels depend on the power supply, the trigger levels are code dependent. Results showed that jump/call instructions executed regularly in loops with a period of τ, could cause a quantization error of $\pm\tau$ (compared to the timer's inherent ±1 uncertainty). This was only observed in slow slewing signals and some solutions were suggested; the use of an external Schmitt-trigger (ST) and the necessity of a decoupling capacitor for the power supply voltage. (Later, Reverter and Pallàs-Areny (2004) and Reverter and Pallàs-Areny (2006), would suggest that the microcontroller should be put into low-power mode during the time measurement in order to reduce the firmware induced uncertainties.)

The need for decoupling capacitors in direct sensor-to-microcontroller circuits was further investigated by Jordana et al (2003). They concluded that a decoupling capacitor was necessary in order to reduce the trigger uncertainties caused by power supply fluctuations. The size of the decoupling capacitor was not critical; increasing the decoupling capacitor above 100 nF did not significantly improve their results. Jordana et al (2003) also investigated the influence of the charging/discharging

capacitor. While a large capacitor is attractive since it increases the charging time and therefore reduces the relative influence of the inherent quantization error of the timer, a small capacitor has the benefit of increasing the signal's slew rate and therefore reduces the uncertainties caused by internal trigger level noise. They tried to find a reasonable compromise between the error-uncertainty-time parameters influenced by the choice of C. They found that the standard deviation of the measured resistance increased with C but they found no distinct relationship between the relative error and C. They suggested C = 1.5 µF as a reasonable compromise between error-uncertainty-time considerations in the case of a Pt-1000 RTD.

Reverter et al (2003b) reported in 2003 the benefits of timing the discharging of the capacitor rather than the charging. The reason is that experimental results showed that the trigger level noise was smaller for the I/O-pin's lower threshold level (used at discharging) than for the higher threshold level (used at charging). This work also showed that the trigger level was more susceptible to signal noise than power supply noise. These results were later confirmed by Reverter et al (2004), where also the time measurement's dependence on the noise frequency was studied. The trigger uncertainty was reported to increase when sinusoidal noise was added to either the input signal or the power supply and was particularly severe when the noise frequency was close to that of the measured signal. Temperature variations and time drifts were reported not to affect the measurement.

The equivalent number of bits (ENOB) of the time-to-digital conversion involved in the discharging time measurement was thoroughly analyzed by Reverter and Pallás-Areny (2004) and they found that the ENOB was depending on the RC time constant only up to some particular time. For small RC values, the quantization uncertainties dominate and the ENOB improves with increasing RC values. However, the consequence of increasing RC is that the signal slew rate decreases at the trigger point and hence trigger level noise uncertainties increase. At some point, trigger level noise will dominate over quantization noise and from that point, increasing RC does not improve ENOB. Hence, there is an optimal measurement speed-ENOB combination for each sensor interface. In the reported experiment, this optimal combination was observed at $RC \approx$ 2-3 ms.

In Reverter et al (2005a) the influence of power supply interferences on the discharging time measurement was investigated. It was shown that power supply interferences did not influence the accuracy as much as the resolution and that these interferences are most severe at low frequencies. It was recommended that the analog and digital power supplies are separated.

4.2.2 Method and material

Taking all these aspects into account, I implemented a one-point calibration, direct sensor-to-controller circuit based on a PIC18 controller, see figure 4.5.

The circuit is intended for an RTD like a Pt-1000 but was simulated with discrete resistors ranging from 817.41 Ω to 2193.95 Ω, corresponding to a temperature range of −47 °C to +310 °C. Microcontroller data was transferred to a Windows PC via an asynchronous serial interface.

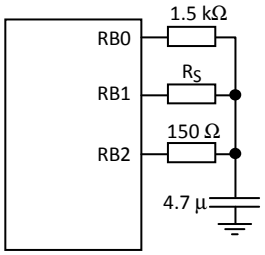


Fig. 4.5 PIC18-based direct sensor-to-controller

4.2.3 Empiri

Table 4.1 and the diagram in figure 4.6 illustrate the results and figure 4.7 illustrates a typical histogram over data sampled at a sensor resistance of 1502.12 ohms.

Table 4.1 PIC18F458 performance data

R_0 [Ω]	mean [µs]	std [µs]	$\hat{R}$ [Ω]	$\hat{R} - R_0$ [Ω]	Relative error
817.41	4895.7	5.63	823.37	+5.96	0.73%
998.17	5958.2	7.20	1002.07	+3.90	0.39%
1193.95	7109.6	8.31	1195.71	+1.76	0.15%
1502.12	8922.4	10.07	1500.59	−1.53	0.10%
1797.44	10658.5	12.68	1792.57	−4.87	0.27%
2193.95	12992.2	14.87	2185.06	−8.89	0.41%

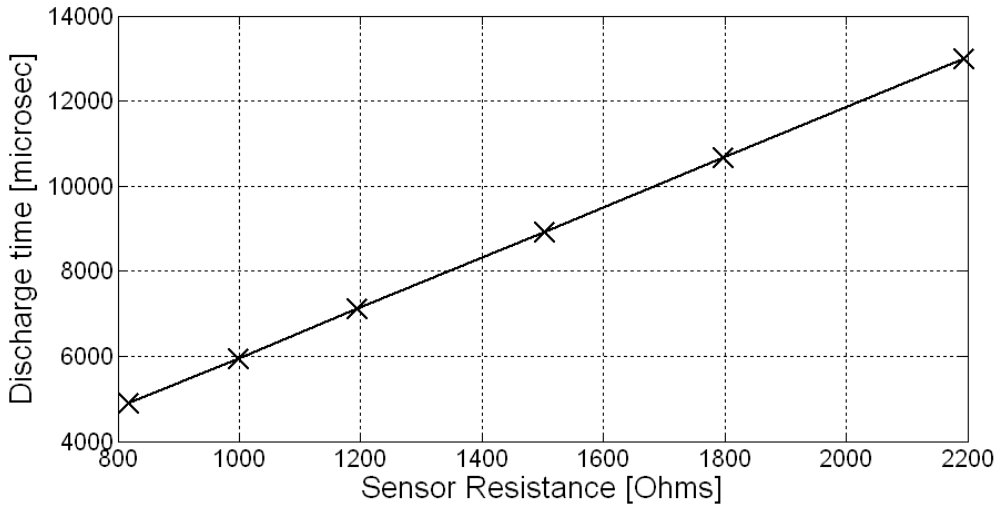


Fig. 4.6 Discharge time versus Sensor Resistance

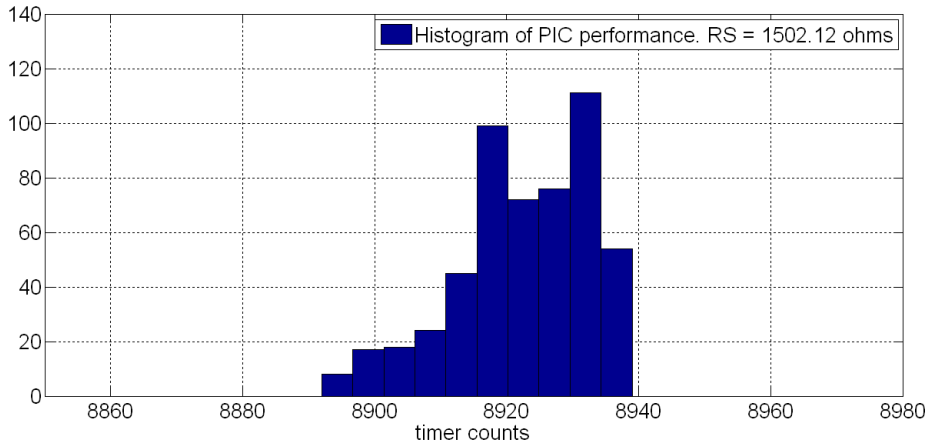


Fig. 4.7 Typical data distribution

4.2.4 Discussion/Conclusions

Notice in figure 4.6 the linear behavior and in table 4.1 the standard deviation and relative error numbers. Table 4.1 and figure 4.6 illustrate the usefulness and potential of the one-point calibration technique suggested in figure 4.1.

Notice in table 4.1 how the error is minimal when R_S is close to R_C. This agrees with the conclusions by Custodio et al (2001a) and Cox (1997). Figure 4.6 suggests a gradient of 55.8 µs/Ω, corresponding to 22.64 µs/°C for a Pt-1000 RTD. The PIC18F

design in figure 4.5 was clocked with a 4 MHz crystal and the internal timer measuring the discharging time was incremented at a rate of 1 MHz. Hence, 22.64 µs/°C corresponds to 22.64 counts/°C and that indicates a resolution of 0.044 °C/count.

4.3 Capacitive sensors

4.2.1 Background/theory

Capacitive sensors, for example humidity sensors (Humirel, 2011), pressure sensors (Chavan and Wise, 2001) and displacement sensors (Chu and Gianchandani, 2003) can also be directly interfaced to a digital system in a similar way as resistive sensors. Richey (1997) suggested the following solution for direct capacitive sensor-to-controller interfacing:

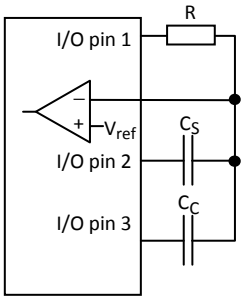


Fig. 4.8 Richey's (1997) capacitance meter

C_S is the sensor's capacitance (unknown) and C_C is a calibration capacitor (known). Each capacitor in figure 4.8 can be disconnected from the circuit by configuring I/O-pins 2 or 3 as digital inputs (High-Z). In step 1, C_C is disconnected and I/O-pin 1 and 2 are configured as outputs; I/O-pin 1 is set high and I/O-pin 2 is set low. This will charge the capacitor C_S and a timer is incremented until the comparator is tripped which stops the timer. The timer value, an integer N_S, is proportional to C_S. The procedure is repeated for C_C, producing an integer N_C. The sensor's capacitance C_S may then be estimated as (Richey, 1997):

$$C_S = \frac{N_S}{N_C} \cdot C_C \qquad (4.6)$$

Notice that Richey (1997) used a comparator to trigger the end-of-measurement. This calls for a more expensive controller but has the advantage of a reconfigurable reference voltage.

Richey's (1997) capacitance meter had a range of 1-999 pF with an average error of 3 %. The disadvantage of Richey's (1997) design in figure 4.8 is that it needs a comparator and times the charging instead of the discharging. As mentioned earlier, discharging should be timed since V_{IL} has less trigger noise than V_{IH} (Jordana et al, 2003).

By combining Richey's (1997) capacitance meter design in figure 4.8 and Cox's (1997) resistance meter design in figure 4.3, we get the direct capacitive sensor-to-controller circuit in figure 4.9.

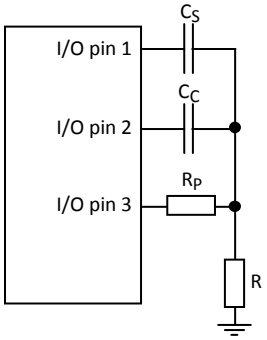


Fig. 4.9 Capacitance meter

First, one of the capacitors is disconnected (by configuring the corresponding I/O-pin as input). The other capacitor's I/O-pin is configured as output and set low; this capacitor is charged via R_P whose I/O-pin is configured as output and set high.

When the capacitor is fully charged, the I/O-pins are reconfigured; all I/O-pins are configured as inputs and the charged capacitor will discharge through R. The end-of measurement is now trigged when I/O-pin 3 reaches V_{IL}.

This has two advantages; no comparator is needed and it has better precision since the discharging is measured and V_{IL} has less trigger level noise than V_{IH}. The procedure is repeated for the other capacitor and equation (4.6) is still valid.

It is not clear who first suggested the Cox-Richey modified circuit in figure 4.9, but it was thoroughly analyzed by Reverter et al (2004) and they also suggested a "three-signal-calibration" technique in order to eliminate zero offset errors caused by stray capacitances. Reverter and Casas (2008) applied it to humidity sensors and designed a humidity sensor with an equivalent resolution of nine bits and a linearity of 0.1 % FSS (full scale span).

Gaitàn-Pitre et al (2009) suggested a two-point calibration technique in order to further reduce the effects of stray capacitance but also to reduce temperature and MCU related interferences. Reverter and Casas (2010a) analyzed the case where

"lossy" capacitive sensors are used (sensors where the parasitic conductance is not negligible) and Reverter and Casas (2010b) applied a similar technique to differential capacitive sensors.

4.4 Bridge sensors

4.4.1 Background/theory

There are plenty of sensors based on the "strain gauge-in-Wheatstone bridge" principle (Bengtsson, 2012a; Doeblin, 1990) but pressure sensors and load cells are the most common ones. (There seems to be a strain gauge-in-Wheatstone bridge based sensor for almost any physical quantity; flow, liquid level, acceleration, pressure, force, viscosity, density….). Two kinds of strain gauges can be used; metal strain gauges (constantan) and *piezo resistive* strain gauges. The resistance of any piece of material is given by

$$R = \rho \cdot \frac{L}{A} \qquad (4.7)$$

where L and A are the length and cross section of the object, respectively, and ρ is a material constant called the *resistivity*. When the object is stressed in any way, *all* of these parameters, ρ, L and A, will change. However, while changes in L and A are the major contributors to the resistance changes in metallic strain gauges, changes in ρ dominates in piezo resistive strain gauges. Piezo resistive materials are made of semi conductor material and can be implanted into silicon membranes. When the membrane suffers a strain, the piezo resistance changes and by connecting the piezo elements pair-wise in a Wheatstone bridge, a voltage proportional to the membrane strain is generated. This is the basic operating principle behind some of the most popular semiconductor pressure sensors, like for example the popular MPX-family from Motorola (Motorola, 2001).

Fig. 4.10 MPX10/D pressure sensor from Motorola

It is very easy to identify bridge type sensors; they almost always come with a four-wire connector, see figure 4.11. The end user typically applies power supply between two wires and the other two (= the bridge signal) are connected to a differential amplifier (an instrumentation amplifier).

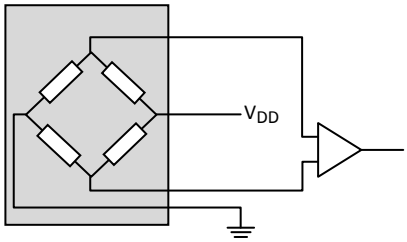


Fig 4.11 Typical bridge sensor interface

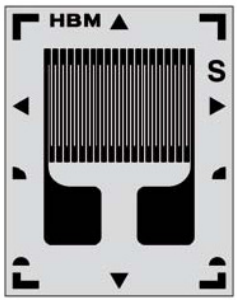


Fig 4.12 Strain gauge (from www.hbm.com)

Strain gauges in Wheatstone bridges are used in two fundamentally different ways. You either use discrete metallic gauges, as the one illustrated in figure 4.12. They are used for example in solid mechanic stress tests. In these applications it is up to the end user to apply the sensors properly to the test object in order to achieve correct temperature compensation (pair-wise, counter-acting) and connect the sensors in a Wheatstone bridge (Bengtsson, 2012a).

However, the typical resistance of a metallic strain gauge is only a few hundred ohms (HBM, 2006) and it has been reported that the resistance of this kind of sensors is too low to be directly connected to a microcontroller (Custodio et al, 2001b) since a microcontroller I/O-pin cannot source enough current.

The other major application is in integrated piezo gauges, like in the pressure sensor case with silicon membranes. These sensors have a resistance of a few kΩs and may be directly connected to a controller. Custodio et al (2001b) suggested the direct bridge-to-microcontroller interface in figure 4.13.

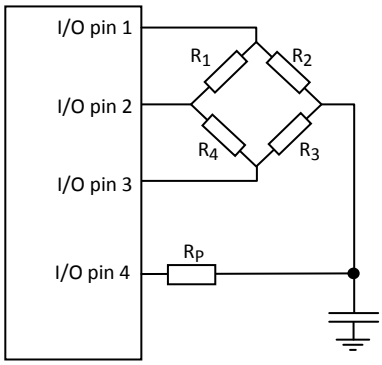


Fig 4.13 Direct bridge-to-controller interface
(Custodio et al, 2001b)

The idea is similar to the previous "direct" circuits, but we need to measure three charging times. First the capacitor is charged by setting I/O-pin 1 to V_{OH} and all the other I/O-pins are set to High-Z; the capacitor is charged through R_2 in parallel with R_1-R_3-R_4 until it reaches V_{IH} of I/O-pin 4 and this generates a charging time of t_{s1}. The capacitor is then discharged through R_P. The procedure is repeated for I/O-pins 2 and 3, generating charging times t_{s2} and t_{s3}, respectively.

Custodio et al (2001b) proved that the relative change of resistance x, in one of the gauges is

$$x = \frac{t_{s1} - t_{s3}}{t_{s2}} \qquad (4.8)$$

Custodio et al (2001b) claim that this technique cannot be applied to Whetstone bridges with metallic strain gauges since a typical microcontroller I/O-pin cannot source the interface.

This thesis claims otherwise; the problem is not that the microcontroller is incapable of sourcing the interface; the problem is that the relative change of resistance is too small.

4.4.2 Hypothesis

The problem of sourcing (or sinking) the bridge can be solved. Consider figure 4.13. The typical resistance of a metallic strain gauge is 350 Ω (HBM, 2006) at "rest". At any time, only one I/O-pin will source (or sink) the bridge current. The total resistance to ground for I/O-pin 1 (or pin 2 or pin 3) is $R_2//(R_1+R_3+R_4)$ = 350//1050 = 262.5 Ω. If $V_{OH} = V_{DD}$ = +5 volts, a single I/O-pin would need to source

$$\frac{5}{262.5} = 19 \text{ mA}$$

If we compare this to the specifications of a typical microcontroller, like Microchip's PIC18F458 (Microchip, 2003), we find that this is dangerously close to the specified "absolute maximum rating" of 25 mA for this controller. Exposing the device to such high levels may permanently damage the device and the functional operation of the device is not guaranteed.

Hence, we need another way to interface the bridge to the controller. This is easily solved, though; let each branch of the bridge be sourced from multiple pins as indicated in figure 4.14. In figure 4.14 each I/O-pin only needs to source 19/4 ≈ 5 mA which is well within the "safe zone" (Microchip, 2003). Hence, sourcing a Wheatstone bridge with metallic strain gauges doesn't have to be a problem.

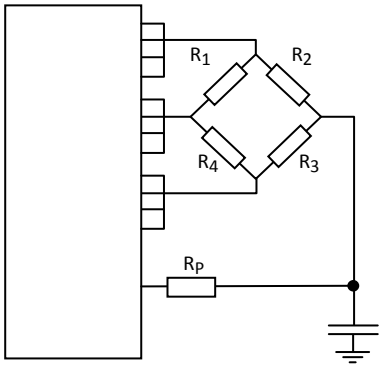


Fig 4.14 Multiple pin-sourcing will solve the sourcing problem

The problem is, however, that compared to piezo resistive sensors, the resistance change in a metallic strain gauge is too small to be detected. Consider figure 4.15. When the beam is exposed to a force *F*, there will be a strain ε in the beam (and of course in the strain gauges). This will cause a change of the gauge resistance by *dR* and the strain gauge factor *k* indicates the size of the change:

$$k = \frac{dR/R}{dL/L} = \frac{dR/R}{\varepsilon} \quad \Rightarrow \quad dR = kR\varepsilon \qquad (4.9)$$

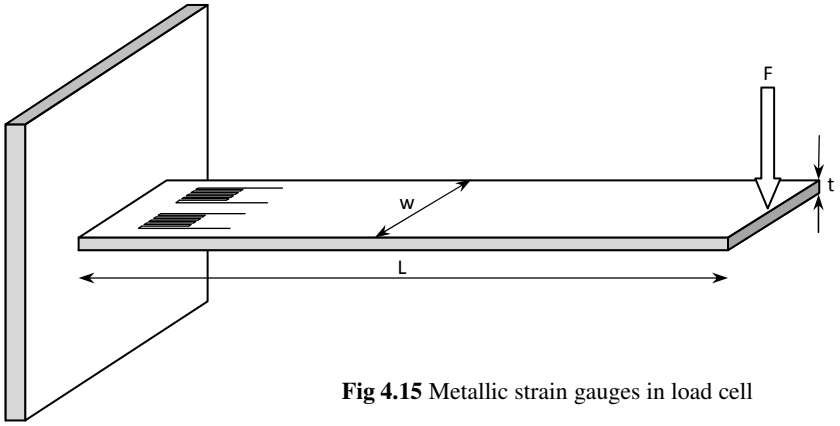


Fig 4.15 Metallic strain gauges in load cell

The strain is determined by the beam tension σ and the modulus of elasticity *E* of the beam material:

$$\varepsilon = \frac{\sigma}{E} \qquad (4.10)$$

The tension σ depends on the geometric dimensions of the beam, how the beam is fastened and the applied force *F*:

$$\sigma = \frac{6L}{wt^2} \cdot F \qquad (4.11)$$

Inserting (4.10) and (4.11) into (4.9) gives us

$$dR = k \cdot R \cdot \frac{1}{E} \cdot \frac{6L}{wt^2} \cdot F = \frac{6kRLF}{Ewt^2} \qquad (4.12)$$

We know that R = 350 Ω. Assuming a 300 mm long aluminum beam, width = 40 mm and thickness = 4 mm, a gauge constant of 2 (HBM, 2006) and a force of 1 N, we would get a resistance change of ≈30 mΩ.

4.4.3 Discussion

We need to find the time resolution required to detect this change in resistance. When an I/O-pin is set high (i.e. *all* I/O-pins sourcing *one* bridge branch in figure 4.14) the capacitor voltage increases exponentially:

$$u_c(t) = V_{OH}\left(1 - \mathrm{e}^{-t/R_{tot}C}\right)$$

This continuous until $u_c(t) = V_{IH}$. Assuming $V_{OH} = V_{DD}$ gives us

$$V_{DD}\left(1 - \mathrm{e}^{-t_c/R_{tot}C}\right) = V_{IH} \qquad \Rightarrow \qquad t_c = -R_{tot}C \cdot \ln\left(1 - \frac{V_{IH}}{V_{DD}}\right)$$

We already know R_{tot}:

$$R_{tot} = R_2 \,//\left(R_1 + R_3 + R_4\right) = R\,//\,3R = \frac{R \cdot 3R}{R + 3R} = \frac{3}{4}R$$

and the time it takes to charge C to V_{IH} is

$$t_c = -\frac{3}{4}RC \cdot \ln\left(1 - \frac{V_{IH}}{V_{DD}}\right)$$

Differentiating with respect to R gives us

$$dt_c = -\frac{3}{4}C \cdot \ln\left(1 - \frac{V_{IH}}{V_{DD}}\right) \cdot dR$$

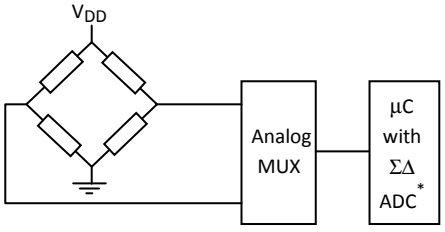
If we assume some typical parameter values (V_{IH} = +1.28 Volts, V_{DD} = +5 Volts, C = 1 μF) we can estimate the resolution required to detect a small change in resistance, dR = 30 mΩ:

$$dt_c = -\frac{3}{4}\cdot 10^{-6}\cdot \ln\left(1-\frac{1.28}{5}\right)\cdot 30\cdot 10^{-3} \approx 7\,\text{ns}$$

If we would clock a PIC18F458 controller with a 40 MHz crystal (= maximum clock rate, Microchip (2003)), the maximum updating speed of an internal timer would be 100 ns (the internal clock runs at a speed of one 4th of the external crystal speed). Hence, that would not be good enough to detect resistance changes in the 30 mΩ range. We could increase the capacitor value by a few orders of magnitude in order to get the desired time resolution, but that would require an electrolytic capacitor of several hundred μF or maybe even a few mF. (Remember that the 30 mΩ we deduced was based on a force of 1 N. Most applications need to be able to detect a lot smaller forces than that.)

However, large capacitor values will not solve our problems here. Electrolytes of this size have considerable ESR (Equivalent Series Resistance) that typically exceeds the mΩ resistance resolution we are looking for here (Adams, 2009). Even if the ESR is known, it would be hard to compensate for it since it depends on both frequency and temperature (Cespiva and Evans, 2009; Kwang-Woon et al, 2008). In order to measure a resistance change of the order of mΩ with the "direct" approaches in figures 4.13 or 4.14, not only do we need a very large capacitor; it also has to have a very low ESR that is frequency and temperature stable.

Finally, we should comment on Baker's work from 1999 (Baker, 1999). Baker used a different approach to Wheatstone bridge interfacing to controllers, as illustrated in figure 4.16.



*) The "direct" ΣΔ-type implemented as described in section 3.3.1.

Fig 4.16 Bridge interfacing by Baker (1999)

In this solution an analog MUX is used as a differential ended interface to the bridge voltage; the bridge potentials are measured separately by using the MUX to switch only one of them to the ADC. The ADC was implemented as a ΣΔ ADC as already described in section 3.3. With this approach a change in voltage is measured rather than a change in resistance.

Chapter 5

Time Measurements

5.1 Introduction/Background

In this context, a *quasi-digital* signal is defined as a binary signal carrying analog information in its variations in time. The analog information is in the "horizontal" direction (time) rather than in the vertical direction (voltage). The analog information is either in the distance between two positive (or negative) edges (= the period or frequency), the distance between a positive (negative) and a negative (positive) edge (= the pulse width) or it is in the quotient between the pulse width and the period (= the duty cycle) (Reverter and Pallàs-Areny, 2005; Viorel, 2006). This is illustrated in figure 5.1.

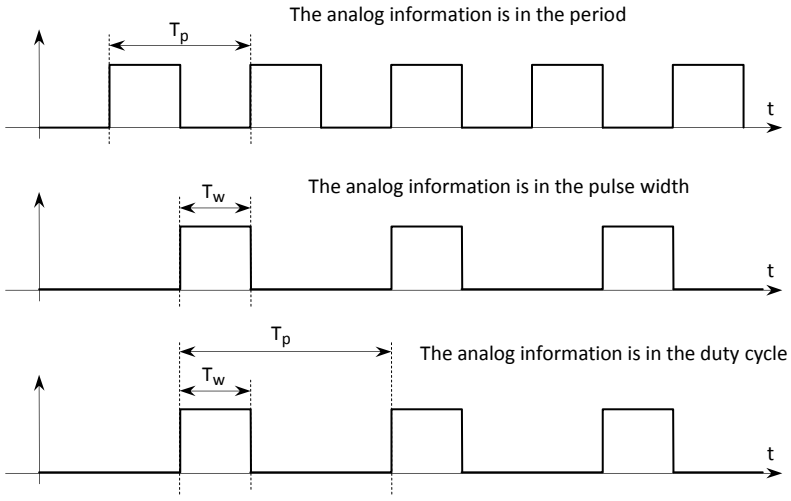


Fig. 5.1 The analog information is in the timing parameters

Hence, if an analog sensor's output can be translated into a quasi-digital signal, instead of an analog voltage, the ADC problem is transferred from one of measuring voltage into one of measuring time and measuring analog variations in time is much easier (and more cost-efficient) than measuring variations in an analog voltage. Also, performance parameters like dynamic range, and noise immunity, typically improve when you measure a variation in time instead of a variation in voltage (Pallàs-Areny and Webster, 2001; Bryant, 2002). In an integrated implementation, analog blocks are large-sized and power-consuming and don't scale very well into sub-micrometer technologies. The reason is that as CMOS transistors are scaled down, the system voltage has to decrease due to the reduction in transistor gate-oxide thickness, and this makes analog circuitry difficult to integrate (Roberts and Ali-Bakhshian, 2010). Embedded systems, like microcontrollers, typically have a number of built-in functions for high-resolution time measurements such as on-chip counters and timers, "input capture" (Microchip, 2003) and Charge Time Measurement Units (Bartling, 2009; Microchip, 2010; Yedamale and Bartling, 2011).

5.2 Time measurements in digital systems

5.2.1 Introduction

Regardless of whether the sensor's information is in the period/frequency or in the duty cycle, the issue of accurate time measurements in digital systems must be addressed. A digital time-measuring system is commonly referred to as a *Time-to-Digital Converter* (TDC). In general, a TDC quantize the time between a start and a stop signal and they are either analog or counter based (Henzler, 2010; Jovanovic and Stojcev, 2009; Webster, 2007).

Analog TDCs use the start and stop signals to generate a pulse with a length corresponding to the start and stop times. This signal is then converted into an analog voltage by an integrator and then an ADC digitizes this voltage, see figure 5.2.

The main disadvantage of the analog TDC is that it contains analog circuitry and that doesn't scale very well when area demands are sharpened by VLSI designers. All-digital circuitry scale much better than mixed-signal circuitry.

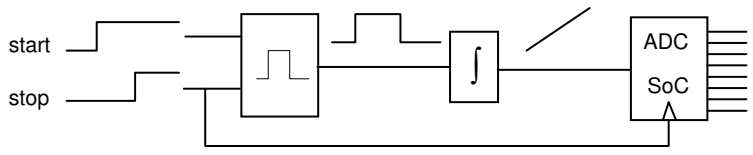


Fig. 5.2 Analog TDC (Henzler, 2010)

For that reason, TDCs are almost always *counter based*. This means that they in principle simply count the pulses from an oscillator during the start-stop interval.

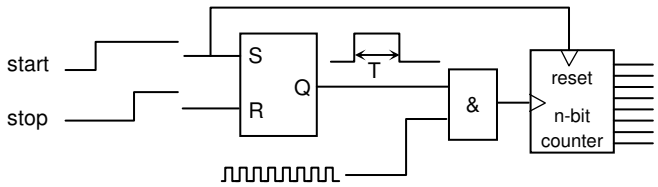


Fig. 5.3 Digital TDC

Notice that the start/stop signals and the oscillator clock pulses are asynchronous. Figure 5.4 illustrates a typical timing diagram.

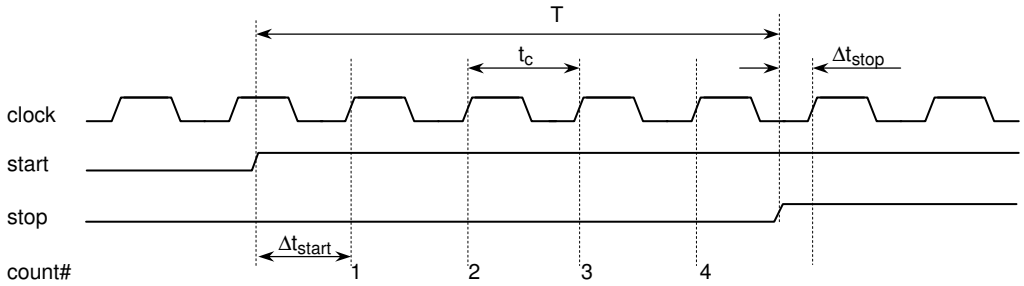


Fig. 5.4 Timing diagram of asynchronous counting TDC

If we assume that we have a positive-edge-triggered counter, we can see from figure 5.4 that the start/stop interval is

$$T = N \cdot t_c + \Delta t_{start} - \Delta t_{stop} \qquad (5.1)$$

Since both Δt_{start} and $\Delta t_{stop} \in [0, t_c]$ the inherent quantization error of a counting TDC is $\pm t_c$. Hence the quantization error scales with the reference oscillator's period. However, increasing the clock frequency raises two other issues; first of all the power consumption increases and secondly, there is a limit to the maximum oscillator frequency that can be implemented in CMOS technology (Henzler, 2010). In order to increase the resolution without increasing the power consumption and without changing from the well-established and inexpensive CMOS technology, other tricks have to be implemented. For example, if the signal is repetitive and if it is asynchronous to the clock signal, averaging a number (n) of time interval measurements will increase the resolution to $t_c / \sqrt{n}$ (Agilent, 2004a). However, this does require a repetitive signal and for 1-shot measurements more advanced tricks has to be implemented. Most of these tricks improve the resolution by interpolating

in between the clock cycle pulses (without increasing the clock frequency). These techniques are referred to as *Vernier* time measurements. The name refers to the inventor of the metric caliper, Pierre Vernier (1580-1637), which can indeed perform a mechanical interpolation between the millimeter markers of a ruler.

Fig. 5.5 A metric Vernier instrument. How can we transfer this idea into high-resolution digital time measurements?

Vernier time measurements can be implemented in several different ways, both in hardware and in software. In fact, according to Henzel (2010), the engine that performs this subdivision of the reference clock cycles is what distinguishes a TDC from a counter; the counter only provides the "coarse" quantization value while the TDC provides the interpolated "fine-structure". The following sub-sections will treat this in more detail.

5.2.2 Time-stretching

Suppose we use a counter to measure a time interval:

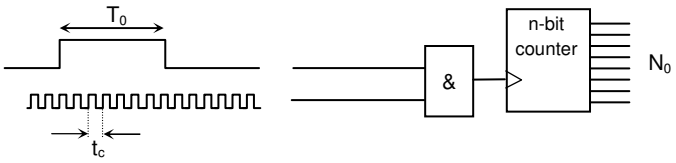


Fig. 5.6 A coarse, counter-based TDC

Then

$$\hat{T}_0 = (N_0 \pm 1) \cdot t_c \qquad (5.2)$$

The resolution is t_c and the uncertainty is $\pm t_c$. Next, suppose that we could *stretch* the time interval by a factor of k:

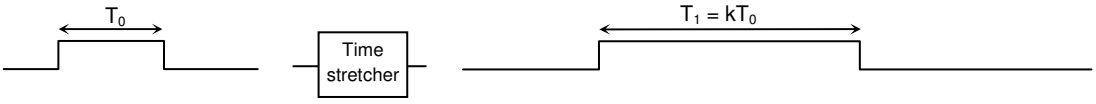


Fig. 5.7 If we can stretch the time interval, then we can improve the resolution

If we measure the stretched time interval T_1 with the same instrument as in figure 5.6, we get

$$\hat{T}_1 = N_1 t_c \pm t_c \qquad (5.3)$$

but since $T_0 = T_1/k$, then

$$\hat{T}_0 = \frac{1}{k}\hat{T}_1 = N_1 \cdot \frac{t_c}{k} \pm \frac{t_c}{k} \qquad (5.4)$$

Hence, if the time interval is stretched by a factor of *k*, both the resolution and the uncertainty improve by a factor of *k*.

There are several ways to stretch a time interval and figure 5.8 illustrates the basic method; charging and discharging a capacitor (Nutt, 1968; Ward, 2011).

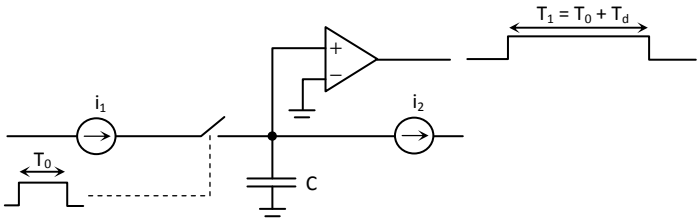


Fig. 5.8 Time stretching (Invented by Nutt, (1968))

In figure 5.8 it is assumed that $i_1 >> i_2$. The time stretching is a two-step process. In step one the switch is closed during a time equal to the time interval to be measured (stretched). During this time the capacitor is charged by a constant current $i_1 - i_2$. The comparator's output goes high immediately after the switch is closed. The switch opens when the time interval T_0 expires and at that time the voltage across the capacitor is

$$U_{final} = \frac{Q}{C} = \frac{1}{C}\int (i_1 - i_2)dt = \frac{1}{C}(i_1 - i_2)\cdot T_0 \qquad (5.5)$$

When the switch opens, the capacitor is discharged by the constant current i_2. The capacitor will be discharged after a time T_d:

$$\frac{1}{C} i_2 T_d = \frac{1}{C}(i_1 - i_2)T_0 \qquad \Rightarrow \qquad T_d = \frac{i_1 - i_2}{i_2}\cdot T_0 = \left(\frac{i_1}{i_2} - 1\right)\cdot T_0 \qquad (5.6)$$

The comparator's output will be high for a time

$$T_0 + T_d = T_0 + \left(\frac{i_1}{i_2} - 1\right) \cdot T_0 = \frac{i_1}{i_2} \cdot T_0 \qquad (5.7)$$

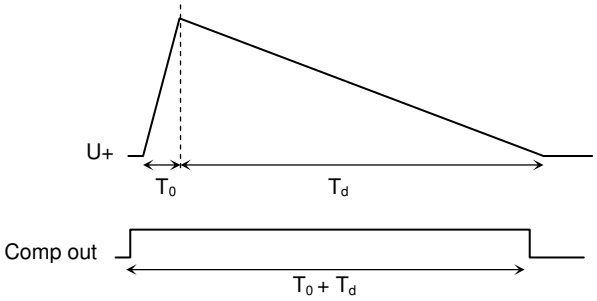


Fig. 5.9 Pulse stretching

Hence, if we compare the comparator's output to the input pulse, we can see that the time interval has been stretched by a factor of $k = i_1/i_2$. With this kind of pulse stretching, time interval resolutions < 10 ps have been reported (Kalisz et al, 1985). Räisänen et al (2000) designed a TDC with a 32 ps resolution by implementing a time stretching circuit on a 0.8 µm BiCMOS chip. This TDC was used as detector in a pulsed time-of-flight laser radar. With this TDC they reported a 4.5 mm precision over a range from 1.5 to 370 meters. See also Agilent (2004b).

5.2.3 Tapped delay lines

A very fast TDC technique is to use digital buffers as delay elements. This is illustrated in figure 5.10.

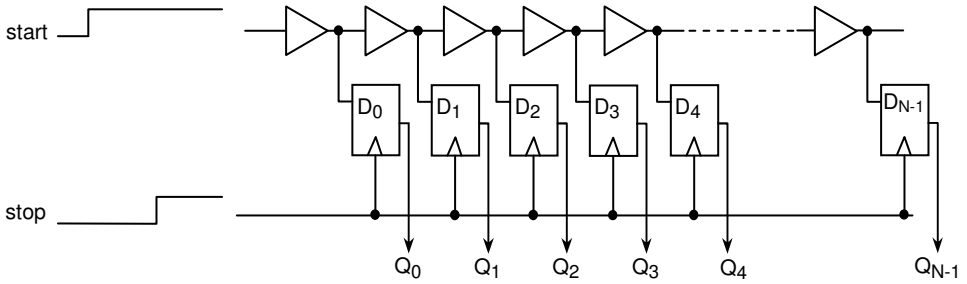


Fig. 5.10 A basic delay line for generation of clock cycle subdivisions

The start signal is connected to the first of an array of cascaded buffers. The start signal's high level will propagate through this chain of buffers at a speed corresponding to each buffer's gate delay τ_{delay}. The output of each buffer is the input to an edge-triggered flip-flop. The flip-flops are latched by the stop signal

arriving some time later. When the stop signal arrives and latches the flip-flops, some of them will have high (‘1’) inputs and some of them will have low (‘0’) inputs depending on how far the start signal’s high level has propagated. This is illustrated in figure 5.11.

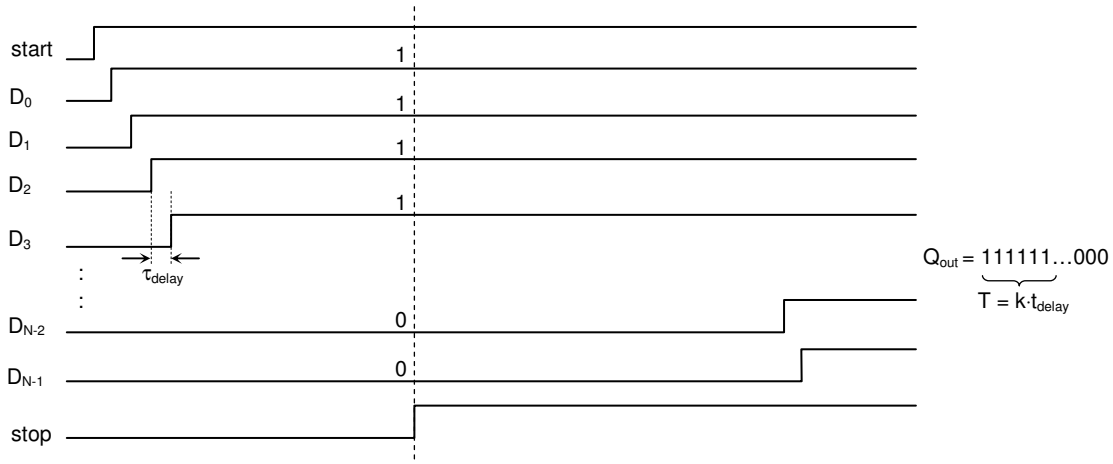


Fig. 5.11 The stop signal latches the flip-flops

When the stop signal latches the flip-flops, the flip-flops’ outputs will be a “thermometer” bit code representing how far the start signal’s positive edge has propagated. The resolution of this TDC is τ_{delay}. Notice that it doesn’t involve a counter. This means that its range is limited to the number of buffers (=N); the range is $N \cdot \tau_{delay}$. However, it is all-digital and therefore “scalable-friendly”. Since the time-to-digital conversion in this case is immediate, it is referred to as a “flash” TDC (representing the TDC correspondence to a flash ADC) (Levine and Roberts, 2004; Roberts and Ali-Bakhshian, 2010).

Figure 5.10 represents the basic tapped delay line. Modified variations of this basic structure have been suggested (Henzler, 2010), but they are all referred to as the “second-generation” TDCs since the time resolution of the design in figure 5.10 is limited to the gate delay τ_{delay} (~2 ns, Agilent, 2004b). In the third-generation TDCs, the time resolution is proportional to the *difference* in gate delays between two buffers. We will come back to that in section 5.2.5 after we have introduced the “vernier” principle.

5.2.4 The vernier principle

Even if all interpolation techniques could be referred to as “vernier” methods, the next one presented below is the one that is most often referred to as the “vernier method”. In this method interpolation between clock cycles are implemented by

employing two oscillators with slightly different frequencies, $f_1 = 1/T_1$ and $f_2 = 1/T_2$, respectively ($f_2 > f_1$) (Agilent, 2004a; Henzler, 2010; Jovanovic and Stojcev, 2009; Porat, 1973). There are two fundamentally different implementations of the vernier principle; whether you use a reference oscillator or not. We describe both methods here.

Figure 5.12 illustrates the timing diagram of the first method (not using a reference clock) (Porat, 1973). Oscillator 1, with frequency f_1 ($<f_2$), starts on the positive edge of the start signal. The second oscillator with frequency f_2 is triggered on the positive edge of the stop signal. Since $f_2 > f_1$, the pulses from the f_2-oscillator will eventually catch up with the pulses from the f_1-oscillator.

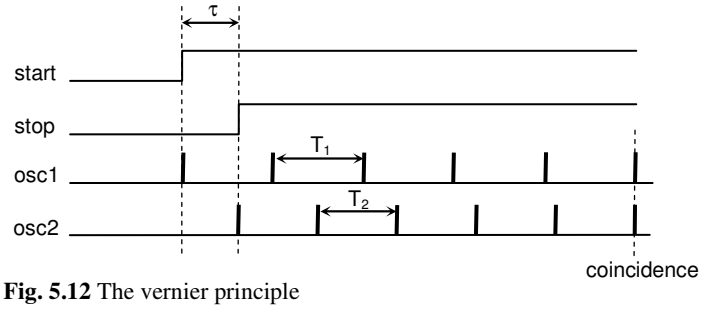


Fig. 5.12 The vernier principle

When this happens, both oscillators are stopped and notice that at this point both oscillators have generated exactly the same number of pulses, i.e. $N_1 = N_2 = N$. From figure 5.12 we get that

$$\tau = N_1 T_1 - N_2 T_2 = N(T_1 - T_2) = N \cdot \Delta T \qquad (5.8)$$

The time resolution depends on the time difference ΔT in the clock cycle periods.

The alternative approach is to use a reference clock that runs asynchronously to the vernier clocks (Agilent, 2004a; Webster, 2007), see figure 5.13.

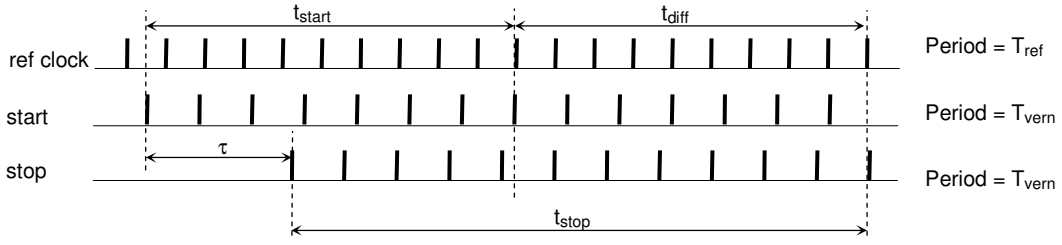


Fig. 5.13 The dual-vernier method (Agilent, 2004a)

If the reference clock cycle period is T_{ref}, we make the vernier clock's cycle period slightly longer; $T_{vern} = T_{ref}(1+1/N)$, where N is an integer that determines the overall time resolution. From figure 5.13 we can see that

$$\tau + t_{stop} = t_{start} + t_{diff} \quad \Rightarrow \quad \tau = t_{start} - t_{stop} + t_{diff}$$
$$\tau = n_1 \cdot T_{vern} - n_2 \cdot T_{vern} + n_0 \cdot T_{ref} \qquad (5.9)$$

where n_1 is the number of T_{vern}-pulses counted during t_{start}, and n_2 is the number of T_{vern}-pulses counted during t_{stop} and n_0 is the number of T_{ref}-pulses counted during t_{diff}.

If we insert $T_{vern} = T_{ref}(1+1/N)$, we get that

$$\tau = T_{ref}\left(n_1\left(1+\frac{1}{N}\right) - n_2\left(1+\frac{1}{N}\right) + n_0 \right) = T_{ref}\left(n_0 + (n_1 - n_2)\left(1+\frac{1}{N}\right)\right) =$$

$$= (n_0 + n_1 - n_2)\cdot T_{ref} + (n_1 - n_2)\cdot\frac{T_{ref}}{N} \qquad (5.10)$$

and we can see from (5.10) that this design allows for a time resolution corresponding to T_{ref}/N. We can easily translate a specified time resolution into a difference in clock cycle periods:

$$T_{vern} = T_{ref}\left(1+\frac{1}{N}\right) \qquad \Rightarrow \qquad T_{vern} - T_{ref} = \frac{T_{ref}}{N} = \Delta T \quad (5.11)$$

5.2.5 The third-generation TDCs: The vernier delay line

Above we had two oscillators with slightly different frequencies "racing" each other until the late starter caught up with the early starter. There is a delay-line version of this vernier principle, similar to the basic delay line in figure 5.10. This is illustrated in figure 5.14 and represents the third-generation TDCs since it can provide time resolutions of the order of the *difference* in gate delay between two buffers (Henzler, 2010).

In figure 5.14 the buffers in the top delay-line have a gate-delay of τ_1 that is slightly longer than the gate delay-time τ_2 of the buffers in the bottom delay-line; $\tau_1 > \tau_2$.

Hence, when the start and stop signals arrive, the stop signal will propagate faster through the delay-line than the start signal does. As long as the start signal is leading, 1s will be latched into each flip-flop when the stop signal (= latch signal) arrives. At

some point though, the stop signal will catch up and pass the start signal (since it propagates faster) and from that point on, 0s will be latched into each flip-flop. The "temperature" code formed by the flip-flops' output will represent the time difference between the arrivals of the start and stop signal and the resolution is now equal to the *difference* in the *gate-delays* between the buffers in the first delay-line and the second delay-line.

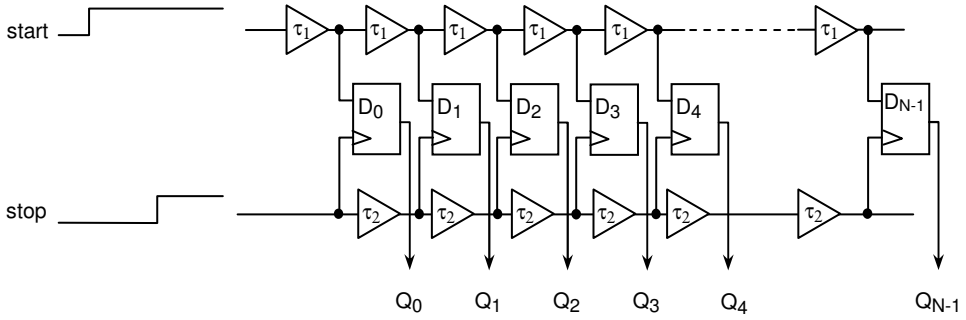


Fig. 5.14 A vernier delay-line (Henzler, 2010)

This has been implemented into embedded targets like FPGAs by Kalisz et al (1997) and Aloisio et al (2009).

5.2.6 Putting it together

If we want both the wide range offered by counter-based TDCs and the high-resolution offered by the tapped delay-line technique, we need to combine them somehow; we would like to use the counter integer value as the "coarse" value and one of the interpolation methods to read between the clock cycle periods in order to get a "fine" value. (Compare with the caliper in figure 5.5; it has both a coarse and a fine scale. That's what we would need to implement in a high-resolution wide-range TDC.)

For the time stretching technique, this problem was solved already by Nutt in his pioneering time stretching work (see Nutt, 1968). Here we will only illustrate how this is implemented using a tapped delay-line. The idea below is mainly from Porat (1973).

Suppose that we start with a counting TDC with N bits resolution and that we want to increase the resolution by M bits. That means that we add M "least significant" bits to the counter value. Then we need a delay line consisting of 2^M buffers. Assume that the clock cycle period of the reference clock is t_C. Then it is important that the gate delay of each buffer is exactly $t_c/2^M$ (which is easily achieved by adjusting the

reference clock cycle period). Hence, the clock cycle edge will propagate through the delay line in exactly the same time as the clock cycle period time; when one clock cycle edge leaves the delay line the next edge is just about to enter.

This is illustrated in figure 5.15.

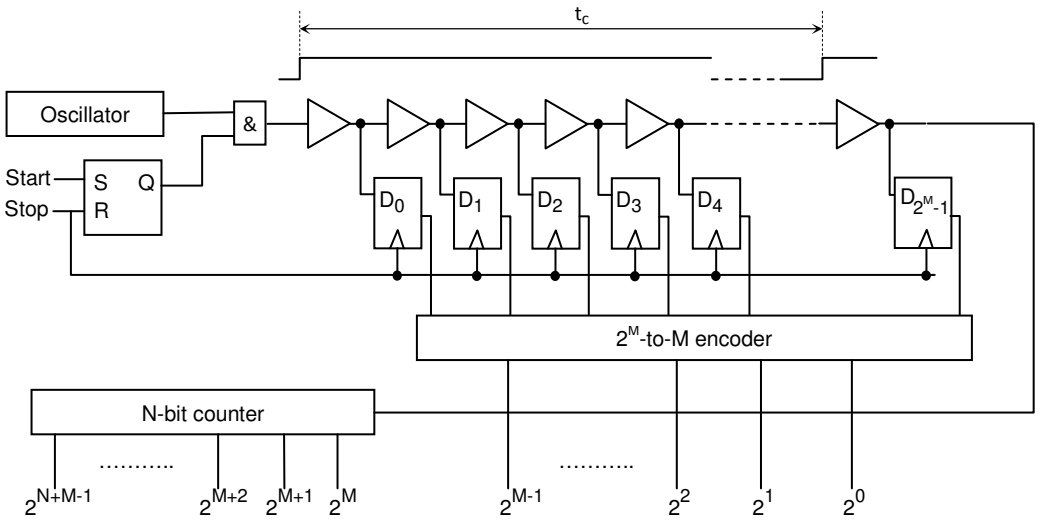


Fig. 5.15 High-resolution, wide-range TDC by Porat (1973)

5.2.7 Time measurement in embedded controllers

Even the simplest microcontrollers have (at least) one embedded timer/counter that can be used for immediate time measurements. No microcontroller with embedded interpolation mechanism has been found, though, but there are several works reported where both time stretching and delay lines have been implemented in FPGA-based designs (Aloisio et al, 2009; Kalisz et al 1997) and in CLPDs (Levine and Roberts, 2004).

The second generation time measurement units in microcontrollers included an *Input Capture* feature, where the content of a running timer is automatically latched into a "capture" register at the occurrence of an external "event", see figure 5.16. This is a great help in time measurements since external events are automatically time stamped by hardware.

Recently, Microchip released the next generation of embedded time measurement units; the CTMU (Charge Time Measurement Unit) (Microchip, 2010; Yedamale and Bartling, 2011). This is an analog TDC, very much like the one in figure 5.2.

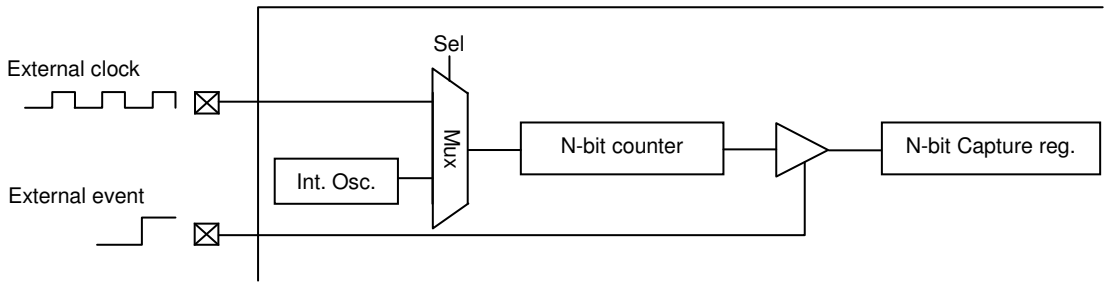


Fig. 5.16 Input Capture

The CTMU consists first of all of a very accurate current source and it is used with an internal ADC to measure either capacitance or time. The idea is that the sample&hold-capacitor of the ADC is charged for some time and the ADC measures the capacitor voltage. Since $Q = CU$ and $Q = It$ we have that $It = CU$. The current is provided by the very accurate and stable current source in the CTMU and the voltage U is measured by the ADC. So depending on whether we want to measure a capacitance or time, we solve for C or t, respectively. (When we measure a capacitance, the unknown capacitance is applied on an external ADC-pin and is hence connected in parallel with the internal S&H-capacitor). Figure 5.17 illustrates the CTMU.

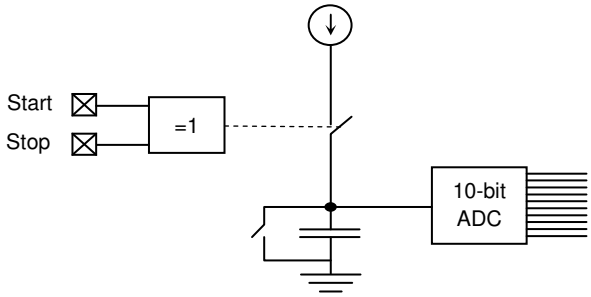


Fig. 5.17 Microchips CTMU (Microchip, 2010; Yedamale and Bartling, 2011)

When measuring time, two external pins are used to turn on and off the current source (which can be switched with < 1 ns accuracy) and the capacitor is charged only during the time interval between the start and stop pulses. The ADC output will be proportional to the charging time interval.

In 2009, Bartling demonstrated how the CTMU can be used in TDR applications (Time Domain Reflectometry) (Bartling, 2009).

5.3 Uncertainties in digital time measurements

5.3.1 Introduction

In this section I will discuss uncertainties in digital time (interval) measurements. I will start by discussing the uncertainty occurring in basic counting TDCs and then compare it with uncertainties occurring in microcontroller-based TDCs. It will show that the same uncertainties that occur in basic counting TDCs also occur in microcontroller TDCs, but in the microcontroller case, a few more uncertainties (might) occur.

5.3.2 Uncertainties in basic counting TDCs

Uncertainties in time interval measurements with a counting TDC come from several sources. In general, uncertainties come from (Agilent, 2004a; Reverter and Pallàs-Areny, 2006):

- ±1 count quantization error
- Trigger errors
- Time-base errors
- Systematic errors

Quantization error: The ±1 count uncertainty is an inherent property of any counting TDC. This uncertainty has a uniform distribution and therefore its standard uncertainty is (Bell, 2009; Bengtsson, 2012a)

$$\sigma_q = \frac{1}{\sqrt{3}} t_c \qquad (5.12)$$

where t_c is the cycle period of the reference clock. As mentioned before, if the reference clock is asynchronous to the start and stop signals, this contribution to the uncertainty can be reduced by averaging:

$$\sigma_1 = \frac{1}{\sqrt{N}} \sigma_q \qquad (5.13)$$

where N is the number of elements averaged.

Trigger error: This noise occurs when the start/stop signal does not trigger at the expected voltage level. This is typically due to noise, either in the input signal itself (σ_e, "external") or in the input amplifier (σ_i, "internal") or both (Reverter and Pallàs-Areny, 2006). The question is how noise, in "voltage", transfers into noise in "time". This depends on the signal's *slew-rate* (SR), see figure 5.18.

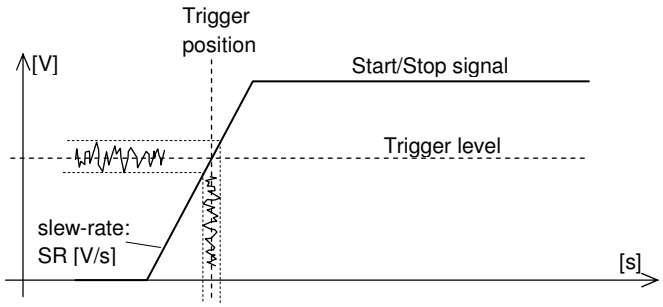


Fig. 5.18 The Start (or Stop) signal

From figure 5.18 we can see that if the total noise level (in Volts) is $\sqrt{\sigma_e^2 + \sigma_i^2}$, then this will cause an uncertainty in the start/stop time corresponding to

$$\sigma_t = \frac{\sqrt{\sigma_e^2 + \sigma_i^2}}{SR} \qquad (5.14)$$

Hence, it is important that both the start and stop signal have high slew-rates. As with quantization noise, the trigger noise may also be reduced by averaging as long as the noise sources have a random distribution.

(The same trigger error may occur in the clock signal, but if we assume that we count a "large" number of clock cycles, they will cancel.)

Time base error: This uncertainty is due to variations in the reference clock cycle period. If they are random, they will cancel if we count a large enough number of cycles. However, even if it is stable, they have a finite accuracy, but if we use crystal based oscillators they can be accurate down to a ppm level and need only be considered if we need 5-6 digits of precision (Agilent, 2004a).

Systematic errors: Typically caused by loading interconnecting cables. The signal propagation speed in a typical transmission cable (RG-58) is 20 cm/ns and it is important to match exactly the cable lengths (and cable types) of the start and stop signals (Agilent, 2004a). Also, the capacitive loading introduced by the interconnecting cables may change the shape of the input signal. If this is a problem, the use of Time Interval Probes should be considered (Agilent, 2004a).

5.3.3 Uncertainties in microcontroller-based TDCs

Reverter and Pallàs-Areny published a work in 2006 where they investigated the uncertainties in microcontroller-based time measurements (Reverter and Pallàs-Areny, 2006). According to this work, quantization uncertainties and trigger noise

are the main sources of error (as in counter-based TDCs). However, they found some significant differences compared to counter-based TDCs:

- In microcontroller-based TDCs, the quantization is not necessarily limited to the ±1 count inherently found in counter-based TDCs; it depends on the firmware algorithm and also on the hardware resources used.
- For microcontrollers, σ_i in (5.14) (internally generated threshold noise) is rarely (never?) specified in the data sheet.

Reverter and Pallàs-Areny suggested that both problems could be solved by putting the microcontroller in sleep mode while waiting for the start and stop signals. (This assumes that the controller has a sleep mode that allows for timers and interrupts to stay active during sleep mode, as for example in some Atmel/Microchip controllers which have an "Idle" mode (Atmel, 2011c; Microchip, 2009)). Most of this section is a review of their work (Reverter and Pallàs-Areny, 2006).

Apart from Microchip's CTMU that we presented in section 5.2.7, a microcontroller basically has three techniques for detecting external events: polling, interrupts and capture.

In polling, the event is detected by periodically checking the status of an I/O-pin in a firmware loop routine. Apart from wasting controller time (and power), the quantization error will depend on the length of the polling loop. If the polling loop executes p instruction cycles, the quantization error will be $\pm p \cdot t_c$, where t_c is the reference clock's cycle period. This is illustrated in figure 5.19.

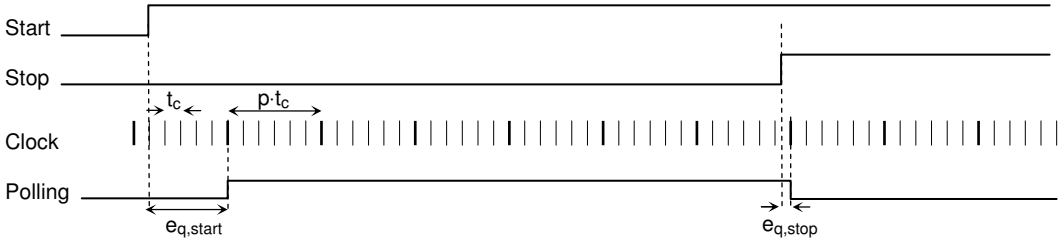


Fig. 5.19 Polling TDC (Reverter and Pallàs-Areny, 2006)

Also, the use of a polling loop indicates the use of jump/call instructions and they are known to be power-consuming and induce power-supply noise that causes threshold noise in the trigger level (Reverter and Pallàs-Areny, 2006). Power supply decoupling will be crucial.

If we use a (general-purpose) interrupt pin to detect the start and stop signals, the quantization error is (probably) reduced. Interrupts on general-purpose interrupt pins

are typically not served until the on-going instruction is completed. Even for typical 8-bit RISC controllers from Atmel/Microchip, instructions may need up to four instruction cycles to complete. If the maximum number of instruction cycles required by any executed instruction is m, the quantization error for the detection of the start and stop signals is $\pm m \cdot t_c$, see figure 5.20.

(There will be some delay between executing the ISR and reading the counter value, but that delay is equally long at both the start and the stop reading so they will cancel.)

Using the capture unit though, the reading of the running counter is taken care of by hardware and we are back to the $\pm t_c$ uncertainty characteristic of counter-based TDCs.

Fig. 5.20 Microcontroller-based TDC with general-purpose interrupt (Reverter and Pallàs-Areny, 2006)

In the work of Reverter and Pallàs-Areny (2006) they showed that the use of a general-purpose interrupt pin in combination with the sleep mode feature ("idle" mode) could also yield a $\pm t_c$ uncertainty.

To summarize, there are two ways to achieve the optimum $\pm t_c$ uncertainty in a microcontroller-based counting TDC; the use of a capture unit or using a general-purpose interrupt pin in combination with the sleep mode. The absolute uncertainty depends on the speed of the clock running the controller. For typical 8-bit controllers from Atmel and Microchip, the maximum (internal) speed is 32 MHz (PIC18FK-family and AVR XMEGA-family) which indicates a maximum time resolution of ~30 ns. However, with Microchip's CTMU device, time resolutions of <1 ns have been reported (Bartling, 2009).

5.4 Microcontroller-implementation of vernier TDC

This section is a discussion of how a vernier TCD could be implemented in a microcontroller and what results could be expected.

Implementing a vernier TDC in a microcontroller is straight forward in principle. Figure 5.21 illustrates the general idea.

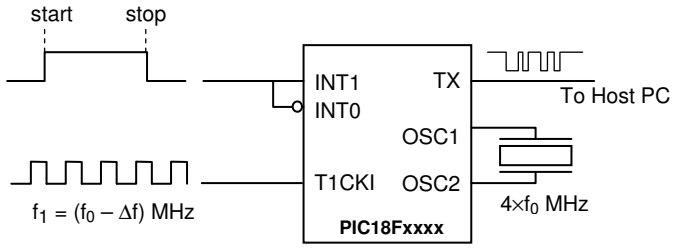


Fig. 5.21 Microcontroller TDC with vernier resolution

The positive edge of the start/stop signal triggers Timer1 (clocked by $f_1=f_0-\Delta f$) and the negative edge triggers Timer0, clocked by f_0. This would give a resolution of

$$\frac{1}{f_0-\Delta f}-\frac{1}{f_0} \quad (5.15)$$

The microcontroller's interrupt reaction time can be minimized if the microcontroller is waiting for interrupt in idle mode (Reverter and Pallàs-Areny, 2006).

Hence, an arbitrary resolution could be implemented in a microcontroller, however, there would be a problem with the precision due to the fact that in a typical microcontroller there would be an inherent uncertainty in the detection of the moment of coincidence of the timer registers. A typical microcontroller cannot compare two running timer registers in hardware and hence the moment of coincidence has to be detected in firmware, see figure 5.22.

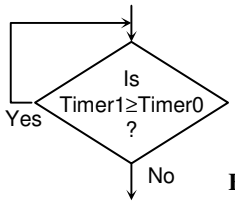


Fig. 5.22 Firmware, part 2

Notice the "≥" sign instead of a "=" sign; timers are updated asynchronously (at least Timer1) and due to loop overhead we will (probably) not be able to compare registers' content on every count. That means that we are likely to miss the moment of coincidence and therefore the "≥" instead of the "=".

The consequence is that there will be an uncertainty in the moment of coincidence detection. If the timer value is N when we detect the coincidence, and if the loop overhead in figure 5.22 is N_{loop}, the real moment of detection could be anywhere in

the interval [N–N_{loop}, N]. Hence, we subtract N_{loop} /2 from N to get an estimate of the moment of coincidence with a symmetric uncertainty interval:

$$\hat{N} = \left(N - N_{loop}/2\right) \pm N_{loop}/2 \qquad (5.16)$$

We insert this into (5.8) to estimate the start-stop interval:

$$\hat{\tau} = \hat{N} \cdot \Delta T = \left(N - N_{loop}/2\right) \cdot \Delta T \pm N_{loop}/2 \cdot \Delta T \qquad (5.17)$$

This is illustrated in figure 5.23. So, due to the fact that the timer registers have to be compared in firmware, we get a 1-shot uncertainty of $\pm N_{loop} \times \Delta T/2$. However, if the signal is repetitive this can easily be improved by averaging a number of samples. This will work since the $\pm N_{loop} \times \Delta T/2$ uncertainty is random.

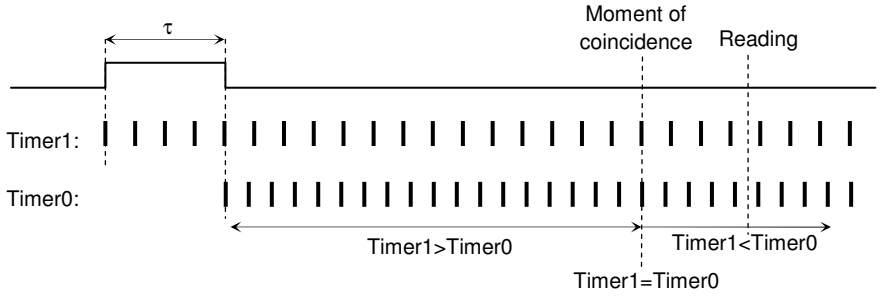


Fig. 5.23 Reading of the timer registers

For example, if we use a 20 MHz crystal in figure 5.21 (f_0 = 5.00000 MHz) and f_1 = 4.95000 MHz, we would have a theoretical time resolution of 1/4.95 – 1/5.00 = 2.02 ns. (Compare that to the corresponding counter-based resolution of 1/5 = 200 ns.) Using a typical C compiler (for example HI-TECH's PICC-18), the firmware loop overhead in figure 5.22 would be 30 ics (instruction cycles) and we would have an uncertainty of

$$\pm \frac{N_{loop} \times \Delta T}{2} = \pm \frac{30 \times 2.02\ \text{ns}}{2} \approx \pm 30\,\text{ns}$$

Hence, we have a resolution of 2.02 ns, but the one-shot precision would only be 30 ns.

There are some things that could be done to improve the precision. First of all, if both timers ran much slower than the system clock, registers could be compared on every count (by activating timer pre-scalers). However, if we use a pre-scale factor of

k, then the time resolution is reduced by a factor of *k*, and we would have to decrease the frequency difference by a factor of *k* in order to get the same resolution. The problem with that is that in order for this to work very stable and accurate clock sources are required, i.e. crystal based, and they don't come in arbitrary frequencies.

Another possibility would be to decrease the loop overhead in figure 5.22. One possibility would be to use an advanced microcontroller with a DMA controller (Direct Memory Access) and via a DMA channel transfer the content of one timer register to the period register of the other timer; registers' content would then be compared in hardware and would (probably) require less firmware overhead.

5.5 Implementation of a High-Resolution TDC in an 8-bit Microcontroller: time stretching

5.5.1 Introduction

Due to the fact that there is no immediate way to compare two "running" timer registers in hardware in a microcontroller the comparison has to be performed in firmware and the discussion in the previous section indicated that this method will most likely suffer from poor precision.

5.5.2 Method

This thesis will now suggest another way to implement high-resolution TDCs in 8-bit microcontrollers. With only a slight modification of the "direct-voltage" and "direct-sensor" methods introduced in chapters 3 and 4, we can implement a microcontroller-based time-stretcher that will increase the time resolution as described in section 5.2.2. The time-stretching method suggested in this thesis is illustrated in figure 5.24.

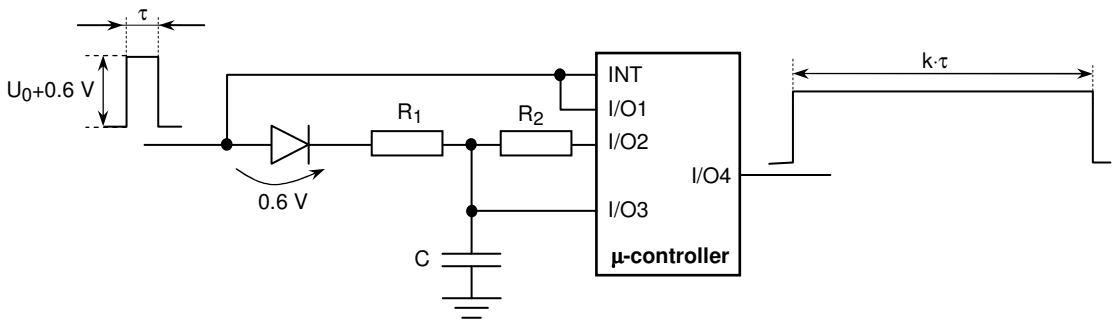


Fig 5.24 A microcontroller-based time-stretcher (Bengtsson, 2012c)

The design in figure 5.24 will stretch the incoming pulse as follows: in "idle" mode, digital output-pin 4 is set to logic low and I/O-pins 1-3 are all configured as inputs, the interrupt pin is enabled for external interrput (on positive edge) and the controller is put to sleep (low-power mode). When the rising edge of the incoming pulse arrives, the controller wakes up from hibernation and just waits for the negative edge of the incoming pulse (by polling I/O-pin 1). During this time the capacitor C is charged via resistor R_1 to a voltage proportional to the pulse width τ. When the negative edge of the incoming pulse is detected, two things happens: I/O-pin 2 is reconfigured to output and set low and I/O-pin 4 is set high. The capacitor will start discharging through resistor R_2 and when the voltage across the capacitor has reached the input logic low threshold of I/O-pin 3, V_{IL}, I/O-pin 4 is reset. If the time constant $R_2C >> R_1C$ the width of the pulse on I/O-pin 4 will be a prolonged version of the incoming pulse.

Notice the finesse with the diode; that enables the incoming pulse to charge the capacitor and prohibit charge leaking back to the input source during discharging.

In order to find the time-stretching constant of this design, we need to analyse the system in more detail.

First of all, there will be a voltage drop across the diode (≈ 0.6 if a silicon diode is used). We assume that the voltage potential *after* the diode is U_0 volts during charging (the input pulse height is U_0 + 0.6 volts). During time τ, the capacitor is charged to a voltage U_τ:

$$U_C(t) = U_0 \cdot \left(1 - e^{-t/R_1C}\right)$$

$$U_\tau = U_0 \cdot \left(1 - e^{-\tau/R_1C}\right) \qquad (5.18)$$

It is important that we choose the charging time constant R_1C so that $U_\tau > V_{IH}$ for τ_{min}. To have some safety marginal, we set $U_\tau > 1.1 \cdot V_{IH}$. We must also make sure that the capacitor is not saturated for τ_{max}. Our safety marginal here is that we demand that for τ_{max}, $U_\tau < 0.8 \cdot V_{DD}$:

$$1.1 \cdot V_{IH} < U_\tau < 0.8 \cdot V_{DD}$$

$$1.1 \cdot V_{IH} < U_0 \cdot \left(1 - e^{-\tau/R_1C}\right) < 0.8 \cdot V_{DD}$$

$$\frac{1.1 \cdot V_{IH}}{U_0} < 1 - e^{-\tau/R_1C} < \frac{0.8 \cdot V_{DD}}{U_0}$$

:

$$-\frac{\tau}{\ln\left(1-0.8\cdot\frac{V_{DD}}{U_0}\right)} < R_1C < -\frac{\tau}{\ln\left(1-1.1\cdot\frac{V_{IH}}{U_0}\right)} \qquad (5.19)$$

The experimental results to be presented later were acquired using a PIC18F4580 microcontroller from Microchip and the following parameter values were used (measured):

Table 5.1 Measured parameter values in time-stretching design

Parameter	Measured value	Uncertainty	Comment	Method/Equipment
R_1	99.71 Ω	0.029 Ω	-	Phillips DMM, PM2534
R_2	8.078 kΩ	0.0029 kΩ	-	Phillips DMM, PM2534
C	207 nF	2.9 nF		HP4261A LCR
V_{IH}	1.25384 V	0.000029 V	Port RB3	Phillips DMM, PM2534
V_{IL}	1.23825 V	0.000029 V	Port RB3	Phillips DMM, PM2534
U_0	3.16 V	0.029 V	-	Tektronix, TDS2012B
V_{DD}	5.05761 V	0.000029 V	-	Phillips DMM, PM2534

Inserting these measured values into (5.19) gives us

$$0.22\tau < R_1C < 2.52\tau \qquad (5.20)$$

$$0.40\cdot R_1C < \tau < 4.54\cdot R_1C \qquad (5.21)$$

In our design example below, the time constant R_1C was dimensioned in order to fit an input time interval range of approximately 10 µs – 100 µs. Inserting the R_1C – values from table 5.1 into (5.21) gives us:

$$8.26\cdot10^{-6} < \tau < 93.7\cdot10^{-6}$$

As long as τ is within this interval, the capacitor will be charged to a voltage in the "allowed" range $1.1\cdot V_{IH} - 0.8\cdot V_{DD}$. This is our first design rule.

In stage 2, the capacitor is discharged through R_2:

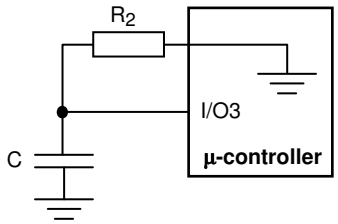


Fig 5.25 Discharging

Here we need to find the time t_d it takes to discharge the capacitor to V_{IL} (of I/O-pin 3):

$$U_C(t) = U_\tau \cdot \mathrm{e}^{-t/R_2C} \quad \Rightarrow \quad t_d = -R_2C \cdot \ln\left(\frac{V_{IL}}{U_\tau}\right)$$

$$t_d = -R_2C \cdot \ln\left(\frac{V_{IL}}{U_0 \cdot \left(1-\mathrm{e}^{-\tau/R_1C}\right)}\right) \qquad (5.22)$$

Hence, the time-stretching factor of this design is

$$k = \frac{t_d}{\tau} = -\frac{R_2C}{\tau}\ln\left(\frac{V_{IL}}{U_0\left(1-\mathrm{e}^{-\tau/R_1C}\right)}\right) \qquad (5.23)$$

In figure 5.26 I have plotted the stretch factor versus τ.

5.5.3 Empiri/Discussion

As expected the stretch factor is not constant and as illustrated in figure 5.26, the theoretical prediction in (5.23) agrees well with experimental data. The reason that the experimental data and the theoretical predictions don't agree exactly, lies in our simplyfied model of the system; for example the diode's capacitance has been neglected. Anyway, the deviation from the theoretical model is unimportant; the time stretching effect is obvious and since it is non-linear, a LUT table would have to be implemented to translate a stretched time interval to input pulse width. For a more detailed description, see paper III appended to this thesis (Bengtsson, 2012c).

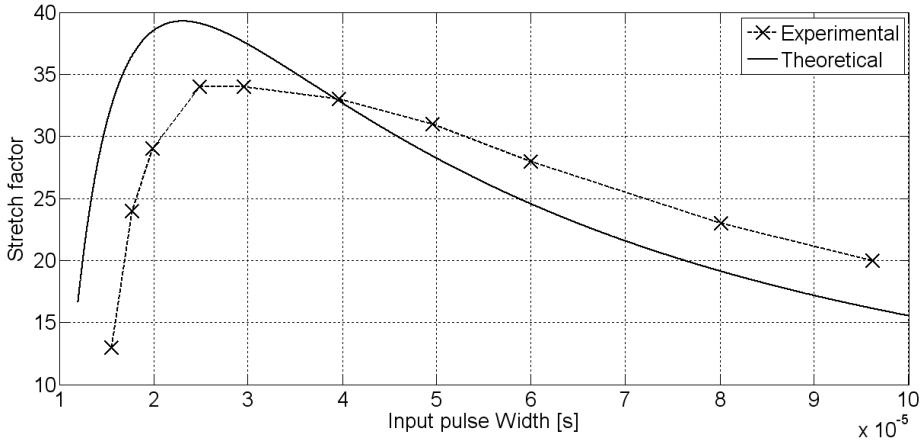


Fig 5.26 Stretch factor vs τ (Bengtsson, 2012c)

To illustrate the power of the proposed method, let's look at an example. If our PIC controller is clocked by a 20 MHz clock, the internal clock rate is 20/4 = 5 MHz. Measuring time with this controller using basic counter technique would yield a maximum time resolution of 200 ns; an input pulse with a duration of 80.15 μs will generates 400 pulses, and we will be able to determine the input time interval as follows:

$$\tau = 400 \cdot 200 \pm 200 \text{ ns} = 80.0 \pm 0.2 \text{ micro seconds}$$

However, if we stretch it as proposed in figure 5.24, the pulse will be stretched by a factor of 23, lasting a total of 1.84345 ms. This will generate 9217 counts. This corresponds to a time

$$\tau' = 9217 \cdot 200 \pm 200 \text{ ns} = 1843.4 \pm 0.2 \text{ micro seconds}$$

and we find the τ-estimate by diving by the stretch factor 23:

$$\hat{\tau} = \frac{1843.4}{23} \pm \frac{0.2}{23} = 80.1478 \pm 0.0087 \text{ micro seconds}$$

Notice also that a count of 9217 corresponds to a 16-bit counter (actually, a 15-bit counter would do too); we have demonstrated how to achieve a 10 ns resolution with an 8-bit controller working at 5 MHz. All it needs is a 16-bit counter/timer.

In paper III appended to this thesis, this design has been used to achieve pulse stretching factors of the order of 1000-2000 and sub-nano second time resolution.

Chapter 6

Digital lock-in amplifiers

6.1 Introduction

Lock-in amplifiers (LIAs) are advanced instruments that are used to attenuate noise in single-frequency ac signals. A LIA behaves like an extremely selective ac voltmeter, i.e. it has the capability of only measuring the rms of one particular signal frequency even in the presence of other larger signals (Bengtsson, 2012a; Scott, 2002; Skoog and Leary, 1992). It can be modeled as a very narrow band-pass filter followed by an rms voltmeter.

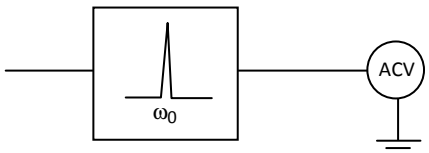


Fig. 6.1 Model of LIA

However, LIAs are capable of extreme selectivity corresponding to filter Q values of the order of 10^6. Such band-pass filters cannot be designed with common filter design techniques. An active analog band-pass filter can have a Q value of perhaps 100 (Scott, 2002) and this is not good enough in LIAs. And even if we could design filters with high enough Q values, it still wouldn't help. A filter Q value of 10^6 would require a ppm level stability of the signal frequency. This kind of frequency stabilities are rare in real applications and the slightest drift in signal frequency (or filter parameters) would make the signal "slip" off the filter's resonance frequency (Bengtsson, 2012a).

The solution is to somehow "lock" the filter's resonance frequency to the signal's frequency. Remember we're looking for a *single* frequency; we *know* what frequency

we are looking for. We only want to know its rms value (in the presence of many other, larger signals).

The solution is to take a signal, having exactly the frequency we're looking for (= the *reference* signal) and multiply it with the measurement signal:

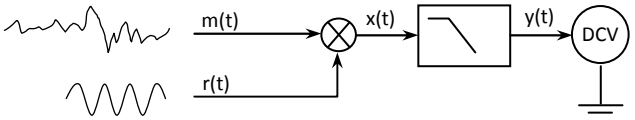


Fig. 6.2 Multiply and filter

If $m(t) = \sum_{i=0,1,2,3...} A_i \cos\omega_i t$ and we want to find the rms of, say $A_0 \cos\omega_0 t$ only, we choose $r(t) = B \cdot \cos\omega_0 t$. Since multiplying two cosines produces two new cosines, with the sum and difference frequency, respectively, the signal $x(t)$ is

$$x(t) = B \cdot \sum_i A_i \cos\omega_i t \cdot \cos\omega_0 t = \frac{B}{2} \sum_i \left(A_i \left(\cos(\omega_i - \omega_0)t + \cos(\omega_i + \omega_0)t \right) \right) =$$

$$= \frac{A_0 B}{2} + \frac{A_0 B}{2} \cos(2\omega_0 t) + \frac{B}{2} \sum_{i \neq 0} \left(A_i \left(\cos(\omega_i - \omega_0)t + \cos(\omega_i + \omega_0)t \right) \right) \qquad (6.1)$$

since cos0 = 1. Hence, the signal $x(t)$ consists of a DC component proportional to the rms value we're looking for and some ac components. All we need to do to isolate the DC component is to low-pass filter $x(t)$. As opposed to high-Q resonance filters, narrow-band low-pass filters are easily implemented (Sedra and Smith, 1990); it is only a matter of choosing the filter's time constant and you can do that almost arbitrarily. Hence $y(t)$ is really a DC signal equal to $A_0B/2$ that we can measure with a DC voltmeter. (Notice also that if we choose the amplitude of the reference signal (B) to be $= \sqrt{2}$, then the DC voltmeter reading will correspond exactly to the rms of the signal we're looking for.)

Notice how we realize the "high-Q resonance filter + AC voltmeter" combination in figure 6.1; we "multiply and low-pass filter".

Due to the extreme selectivity potential of LIAs, they are used in any applications suffering from poor SNR. The only catch is that you must know what frequency you're looking for. This is typically achieved by actively *exciting* the sample. In laser experiments the laser is typically pulsed and the signal triggering the laser is also the reference signal to the LIA. According to Scott (2002), typical applications include low-level optical experiments, acoustical and cross-talk measurements, electron

spectroscopy, neurological research, feedback control of lasers, complex impedance measurements, optical pyrometry, superconducting squid measurements and hot wire anemometry. In this thesis we will later design a digital milliohm meter by implementing a LIA in an embedded microcontroller design.

However, even if the LIA model in figure 6.2 looks straightforward, it is much more complicated when you look into the details. We will do that in this thesis, but before we do that we need to define some new concepts. That includes the concepts of *phase-locked loops* (PLLs), *phase sensitive devices* (PSDs) and *voltage controlled oscillators* (VCO).

6.2 Theory

6.2.1 VCO

A VCO is an oscillator whose frequency can be tuned by an analog voltage.

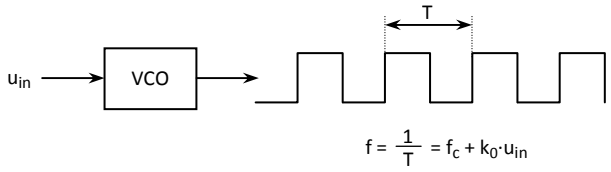


Fig. 6.3 A voltage controlled oscillator

The VCO output depends on the input voltage. If the input voltage is zero, the VCO generates an output signal with frequency f_c (= the "free-running frequency"). In figure 6.3, k_0 is the *sensitivity* of the VCO with unit [Hz/V]. Hence, the frequency of the VCO output signal can be greater than or less than f_c, depending on the sign of u_{in}.

We also need to know the relationship between phase and frequency. By definition, frequency is the rate at which the phase is changed, and hence we can get the phase by integrating the frequency:

$$\varphi(t) = k_1 \int f(t) dt \quad (6.2)$$

VCOs are sometimes referred to as V/F converters (voltage-to-frequency converters) and the output signal isn't always a square wave as indicated in figure 6.3; it can be sinusoidal or triangular.

6.2.2 PSD

A phase sensitive detector (PSD) is a device that produces a signal output proportional to the phase difference between two signals, see figure 6.4. In the simplest case, you can design a PSD by simply integrating the output signal from an XOR gate, see figure 6.5. That will certainly produce an analog voltage proportional to the phase difference between the two input signals.

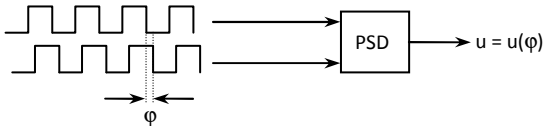


Fig. 6.4 A phase sensitive device

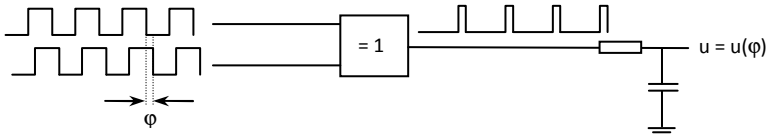


Fig. 6.5 A simple PSD (Abramovitch, 2002; Roon, 1995)

This would only produce a positive signal and we wouldn't be able to see the difference between positive and negative phase shifts. For analog signals, the circuit in figure 6.6 is commonly used.

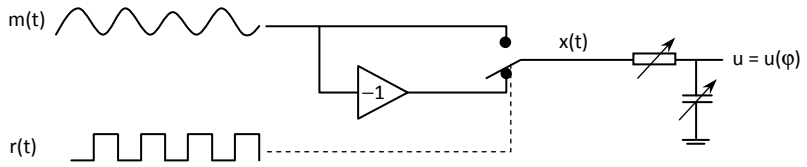


Fig. 6.6 Analog PSD (Stoltenberg and Vilches, 1995)

When the square wave is "on" the switch is in the upper position and when it is "off" it is in the lower position (and the signal is phase-shifted 180°). If $m(t)$ and $r(t)$ are "in-phase", this circuit will rectify $m(t)$ (and the RC-filter will smooth it), see figure 6.7.

When the phase difference $\varphi = 90°$, the signal $x(t)$ in figure 6.6 will be smoothed to zero as illustrated in figure 6.8 and when $\varphi = 180°$, the signals are "out of phase" and will be smoothed to −1, see figure 6.9.

Obviously, the signal u is a function of the phase difference between $m(t)$ and $r(t)$; $u = f(\varphi)$. This dependence is illustrated in figure 6.10. Hence, the circuit in figure 6.6 will produce a zero output when the reference signal is 90° out of phase relative $r(t)$.

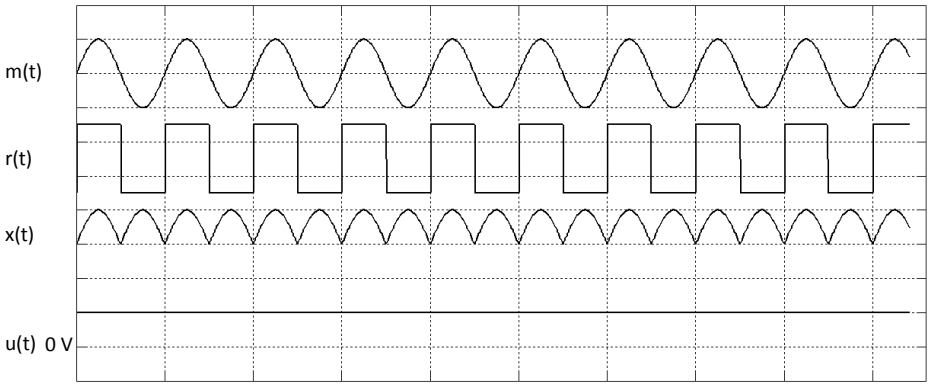


Fig. 6.7 $\varphi = 0°$

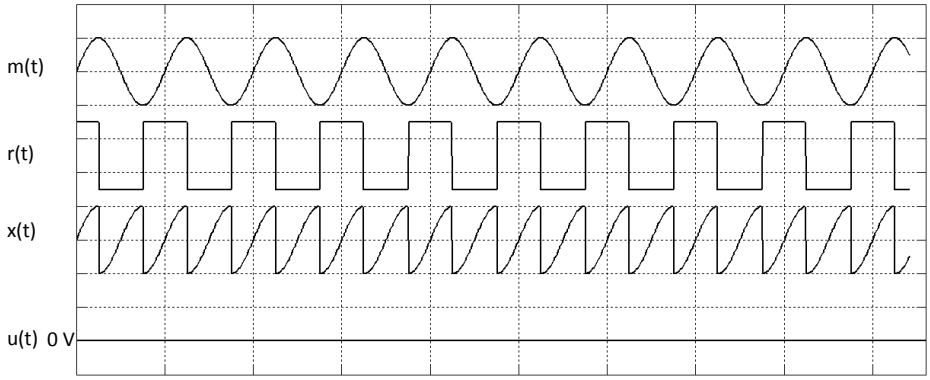


Fig. 6.8 $\varphi = 90°$

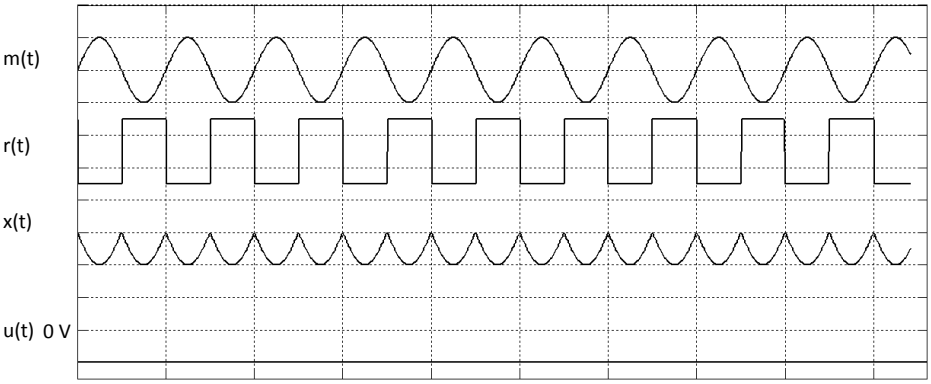


Fig. 6.9 $\varphi = 180°$ (See also Fisher, 2003)

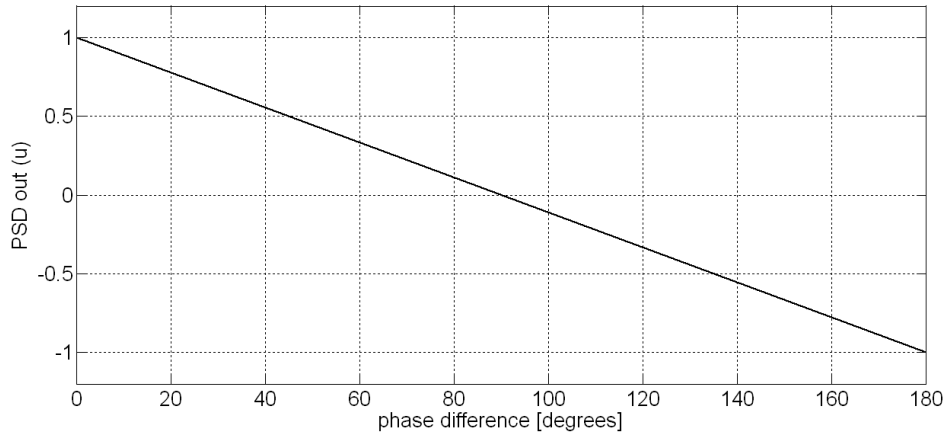


Fig. 6.10 The smoothed PSD output versus φ

If we want it to produce a zero output for a phase difference of $\varphi = 0°$, then we need to shift the reference signal by 90°. Electronic circuits that provide tunable phase shifts have standard solutions; see for example Horowitz and Hill (1990, pp 77-78).

The analog PSD in figure 6.6 is easily implemented in hardware with a single OP amp and an analog switch (i.e. a semiconductor switch like AD7512DI (Analog, 2010c)), see figure 6.11. This thesis will later suggest a microcontroller implementation of the analog PSD in figure 6.6 that does not require an analog switch (Bengtsson, 2012d).

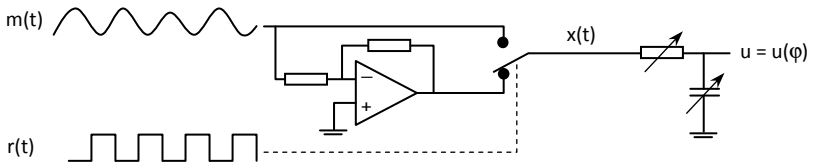


Fig. 6.11 Analog PSD (Wolfson, 1991)

We also need to consider what happens when there is a *frequency* difference Δf, between $m(t)$ and $r(t)$. The answer is that φ will not be a constant but will vary periodically with period $1/\Delta f$ (Stoltenberg and Vilches, 1995). Signals with a frequency deviating with Δf from the frequency of $r(t)$ will appear as ac-fluctuations at the output and depending on the size of Δf and the time constant of the low-pass filter, they will be more or less attenuated by the filter. Figure 6.12 illustrates the situation when the frequency of $m(t)$ deviates from that of $r(t)$ by 10%.

It is interesting to see how easily a PSD is implemented in software; during a time

corresponding to the period $T_0 = 1/f_r$ of the reference signal, the measurement signal is sampled twice, separated by a time corresponding to π radians. If the frequencies of the reference signal and the measurement signal agree, then the two samples will have different signs and if we subtract them the signal will be enhanced (Momo et al, 1981), see figure 6.13.

The first sample is $m(kT_0)$ and the second sample is $m((k+1/2)T_0)$ and if we add a large number of $m(kT_0) - m(k+1/2)T_0$ values, any signal component in $m(t)$ that has a frequency not equal to $1/T_0$ will be averaged out while the signal with frequency $1/T_0$ is enhanced.

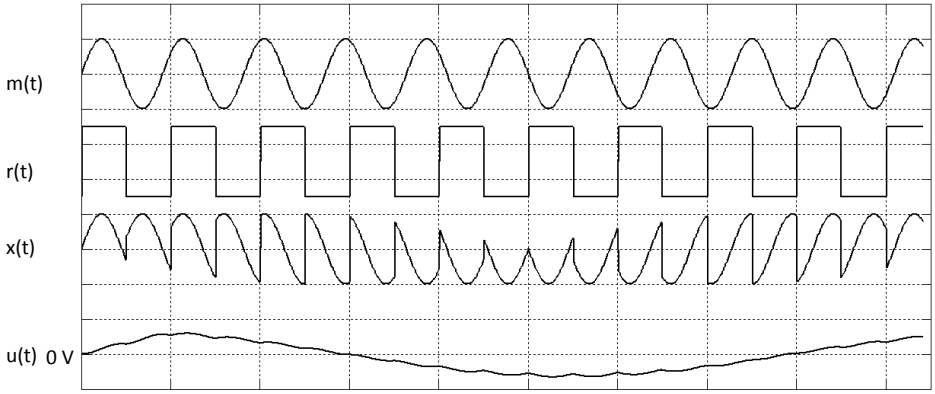


Fig. 6.12 If frequency of $m(t) \neq$ the frequency of $r(t)$, then $u(t)$ will fluctuate with frequency Δf

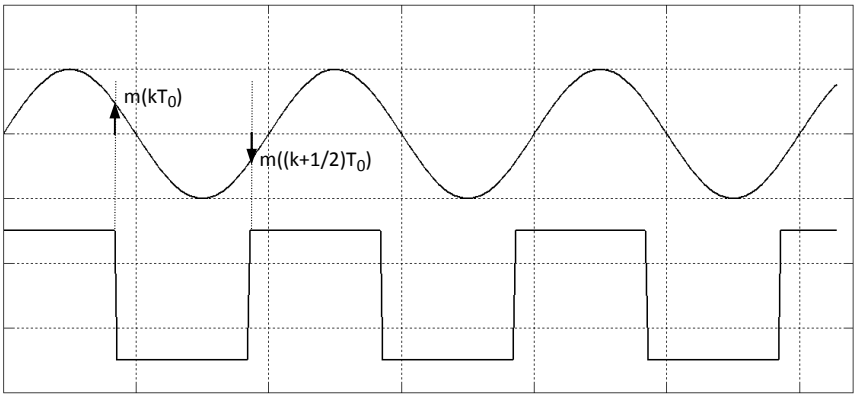


Fig. 6.13 Software PSD; sample at $T_S = 0.5T_0$ and subtract pair-wise (Momo et al, 1981)

From figure 6.13 we can also see that this simple software algorithm not only discriminates any signal with a frequency $\neq 1/T_0$, the magnitude of $m(kT_0) - m((k+1/2)T_0$ also depends on the phase difference between $m(t)$ and $r(t)$; it is indeed a PSD.

Notice also that the mixer circuit in figure 6.2 is also a PSD. If there is a phase difference between $m(t)$ and $r(t)$, expression (6.1) changes to

$$x(t) = \frac{A_0 B}{2}\cos\varphi + \frac{A_0 B}{2}\cos(2\omega_0 t + \varphi) + \\ + \frac{B}{2}\sum_{i\neq 0}\left(A_i\left(\cos((\omega_i - \omega_0)t + \varphi) + \cos((\omega_i + \omega_0)t + \varphi)\right)\right) \quad (6.3)$$

and $y(t) = (A_0 B \cdot \cos\varphi)/2$, and we obviously have a phase sensitive device. According to Scott (2002) though, analog multipliers suffer from gain instability, non-linearity, distortion and offset drift and for that reason, real world PSDs are typically implemented with some kind of switching circuit, like the one in figure 6.11.

6.2.3 PLL

The expression (6.1) worked out nicely because the reference signal was a perfect cosine. Most of the time it is not perfectly stable. It must be periodic but its shape (amplitude, harmonics and phase) may change (with time or from one application to another). Whatever it looks like, the reference signal fed into the multiplier must always be constant or the output reading cannot be used for absolute measurements. Also, if the reference signal is supposed to drive an analog semiconductor switch, like in figure 6.11, it has to be a "solid reliable" digital signal.

This problem is solved by first feeding the raw reference signal into a *phase-locked loop* (PLL). In principle, a PLL produces a constant-shaped signal with the same period as the input signal, independent of the input signal's shape (exactly what we're looking for!). As a matter of fact, most PLLs produce a constant-shaped square wave signal with a period equal to the input signal; if the input signal's period changes, so does the square wave's period.

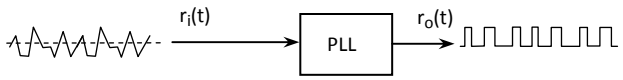


Fig. 6.14 A PLL will produce a "clean" replica of the input signal

A phase-locked loop is easily designed once you understand the operating

principles of VCOs and PSDs. The basic PLL principle is illustrated in figure 6.15. In figure 6.15 the PSD has its own low-pass filter (see figure 6.6 and 6.11) and the "extra" low-pass filter in figure 6.15 is optional (but will improve the settling time (Scott, 2002)) and the (DC) amplifier may or may not be necessary depending on the inherent amplification of the PSD. In the PLL circuit in figure 6.15, the VCO will lock at a frequency equal to that of the input signal frequency.

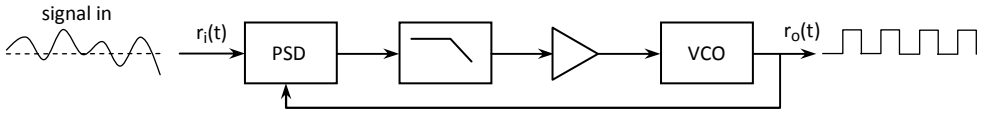


Fig. 6.15 The PLL principle (Roon, 1991)

If there is a difference in phase between signals $r_i(t)$ and $r_o(t)$ ($= \varphi_i - \varphi_o$), a control signal $\neq 0$ is generated to the VCO and this will either speed up or slow down $r_o(t)$ until the phases are aligned. Notice also that for the phases to agree, the frequencies must agree and hence not only the signal phases are locked; the frequency of $r_o(t)$ is locked to the frequency of $r_i(t)$ (Rota, 2005).

We conclude by mentioning that in many applications a $\div N$ ("divide-by-N") circuit is inserted in the feedback path. This will force the VCO to generate a signal with a frequency N times that of $r_i(t)$. This is typically used in frequency synthesizers (Roon, 1991).

The PLL circuit was first suggested by a French scientist in 1932 (Bellescise, 1932) and is now used in a wide range of applications like AM and FM demodulators, FSK decoders, motor speed control, robotics, radio transmitters and receivers etc (Roon, 1991). Cell phones and satellite TVs would not exist without them (Roon, 1991). In this thesis we will apply them to signal recovery in measurement systems only.

6.2.4 Hardware

We can now expand our lock-in amplifier in figure 6.2 to include some more details, see figure 6.16 (mostly from Scott (2002)). Notice in figure 6.16 how the external oscillator signal excites the experiment and at the same time is used as the reference signal. In order to get a stable, constant reference signal to the lock-in phase detector, the external oscillator is injected into a phase-locked loop circuit. The PLL output signal can be phase-shifted arbitrarily in order to match the phase of the experiment signal. Notice also that the experiment signal is pre-amplified and "pre-filtered" before it reaches the phase detector.

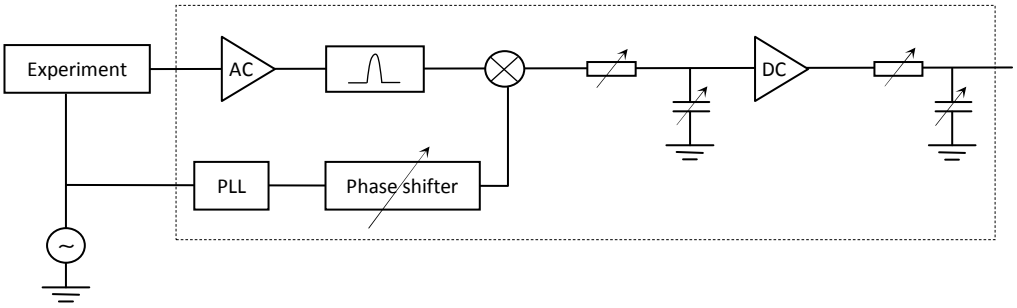


Fig. 6.16 Single-Phase LIA (Scott, 2002)

From expression (6.3) we know that the output from the first low-pass filter is $(A_0 B \cdot \cos\varphi)/2$ or $A_0 \cos\varphi$ if $B = 2$. Even with the tunable phase-shifter in figure 6.16, it can be difficult to the find the optimal setting of the reference signal's phase shift and advanced LIAs solve this by implementing a second PSD.

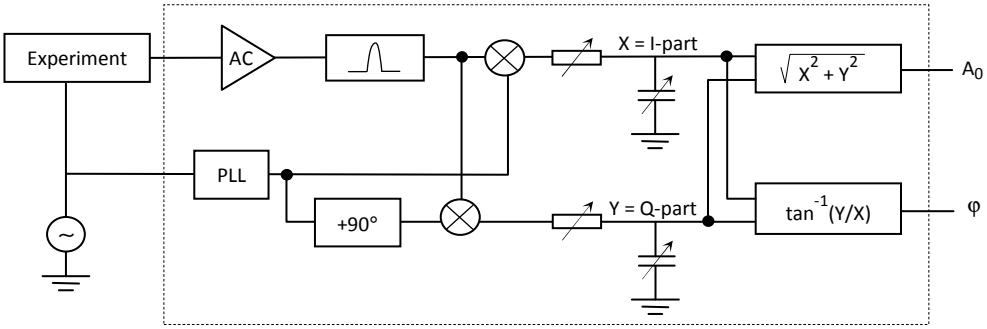


Fig. 6.17 Dual-Phase LIA (Stanford, 2001)

In a LIA with dual phase detectors the problem of aligning the reference signal's phase with the experiment signal's phase is eliminated. The first PSD produces a signal *X* proportional to the part of the experiment signal that is "in-phase" with the reference signal ($I = A_0 \cos\varphi$). In the second PSD the reference signal is first phase shifted 90° (= a *Q*uarter of a period) and the second PSD now produces a signal proportional to the part of the experiment signal being 90° out of phase (= the "Q-part": $Q = A_0 \sin\varphi$). The square root of the sum of the square of these two signals will eliminate the output signal's dependence on the phase difference between the experiment signal and the reference signal.

Notice also that by taking the inverse tangent of Q/I, we can actually measure the phase difference. For this reason, LIAs with dual phase detectors are sometimes referred to as "vector voltmeters" since they can indeed measure both the magnitude

and the phase shift of an ac signal (Scott, 2002).

Li et al (2011) presented in 2011 a firmware algorithm based on figure 6.13, that could also recover both the in-phase and the quadrature signals; by sampling twice as fast as indicated in figure 6.13 (samples $\pi/2$ apart), they showed that the I-part corresponds to $(x(1)-x(3))/4$ and the quadrature part corresponds to $(x(0)-x(2))/4$.

6.3 Method and Material

I will here demonstrate the use of digital LIAs by designing a digital submilliohm meter with phase lock-in detection.

6.3.1 Firmware

Firmware implementations of LIAs are based on the software PSD described in section 6.2.2 and figure 6.13. Implementations of firmware LIAs based on this principle has been reported by Dorrington and Künnemeyer (2002) and Momo et al (1981) for example. Wang (1990) demonstrated the power of an improved software PSD technique; instead of taking just two samples separated by π radians as indicated in figure 6.13, Wang sampled continuously. If the reference signal was positive (or "high") all samples were added to a software variable and if the reference signal was negative (or "low") all samples were subtracted from the software variable. The LIA output was the value of the software variable divided by the number of samples during one reference signal period. This efficiently amplifies only signals with the right phase and frequency. Yiding et al (2007) even claimed that the firmware LIA method would produce less error than the analog mixing LIA.

6.3.2 Simulating a binary switch

In order to use the firmware algorithm suggested in figure 6.13, we need to excite the experiment in such a way that it produces a bipolar signal. By sampling twice and subtracting the samples, any noise not having a frequency that equals the exciting frequency, will be cancelled by the subtraction.

Figure 6.18 suggests how we can create a bipolar alternating current using a microcontroller by taking advantage of the fact that digital output pins can both source and sink current.

Notice that I once more take advantage of the fact that the digital I/O-pins are reconfigurable; we configure both as outputs but alternate their output levels to high and low in order to excite the experiment with an alternating bipolar current.

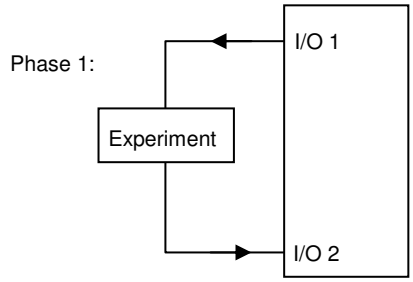


Fig. 6.18a I/O1 is OUT-HIGH, I/O2 is OUT-LOW

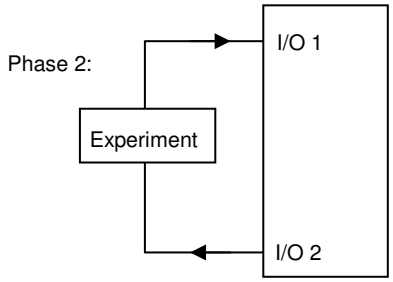


Fig. 6.18b I/O1 is OUT-LOW, I/O2 is OUT-HIGH

6.3.3 Digital milliohm meter

Resistance is typically measured by taking advantage of Ohm's law; a known current is injected into the resistor and the voltage drop across the resistor is measured with a voltmeter. In order to measure resistances in the milliohm range, special care must be taken to avoid fundamental errors. If the distance between the resistor and the instrument is "large" (indicating long wires), the 4-wire method must be used in order to eliminate the wiring resistance (Bengtsson, 2012a). In embedded measurement systems, this is typically not a problem since the resistor (i.e. the resistive sensor) and the controller are typically on the same pcb, close to each other and connected with wide copper wires.

A bigger problem is that when measuring very small resistances the voltage drop across the resistor will be very small. A 10 mA current will cause a 100 μV drop across a 10 mΩ resistor. We could increase the current but some samples may not allow the increased heat loss due to the increased current and in many applications we are simply forced to deal with very low signal levels (Feng et al, 1999; Wong, 2011). Signals in the sub-millivolt range are also of the same size as typical noise sources such as Johnson noise and 1/f-noise (Stanford, 2006). We could (and will) amplify the signal, but a general non-selective amplifier will amplify the noise as well as the signal. In order to separate the resistance signal from the noise we need to measure the resistance signal with phase lock-in technique. This thesis now suggests the embedded measurement system in figure 6.19 for the measurement of sub-milliohm resistances.

R_X is the unknown resistance (of the order of milliohm). I/O-pins 1 and 2 are configured as outputs and will excite the sample resistance as described in section 6.3.2, see figure 6.18. The R_0-resistors are of the order of 100-200 ohms and are necessary in order to limit the current required to excite the experiment; the maximum current that can be sourced by a typical microcontroller I/O-pin is

typically of the order of 10 mA. A high-gain instrument amplifier is used to amplify the mV-signal across the milliohm resistor R_X. The amplifier output is a bipolar signal and R_1, R_2 and R_3 is a bipolar-to-unipolar converter (Greggio, 2011).

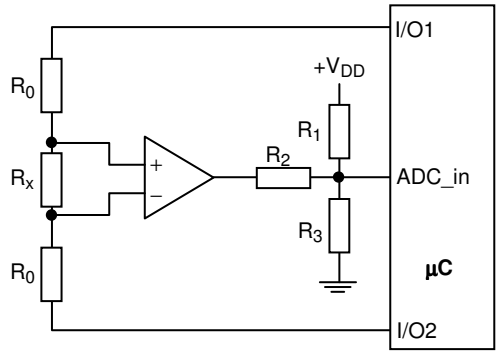


Fig. 6.19 A digital LIA for submilliohm measurement (Bengtsson, 2012d)

During phase 1 I/O-pin 1 sources the exciting current and I/O-pin 2 sinks it. The microcontroller samples the ADC input. During phase 2, the exciting current is reversed and sample number two is acquired. These samples are subtracted in order to discriminate out of phase-signals and by averaging a large number of these subtracted samples, any signal that doesn't have a frequency that agrees with the exciting reference signal's frequency will be discriminated. The rate at which the switching of the I/O-pins logic levels occurs corresponds to the lock-in frequency.

6.4 Empiri

Figure 6.20 illustrates the results of one such implementation in a PIC18F4580 microcontroller.

The system was calibrated by using a copper wire with different lengths (diameter 0.598 mm). The instrumentation amplifier used was an INA128 (Burr-Brown, 1995) configured for maximum amplification (×10.000). In figure 6.20, "counts" refers to the ADC produced integer (10-bits).

6.5 Discussion

Notice the resolution indicated in figure 6.20; 17.98 counts/mΩ translates to 55.6 µΩ/count. (This was actually even better than the theoretically predicted resolution of 72 µΩ/count and the reason for the deviation was that the amplification of the instrumentation amplifier was unsymmetrical; it was 20-30% larger for the negative samples than for the positive samples. (See paper IV appended to this thesis for a detailed description of this design and a detailed analysis of the system and the calibration data, (Bengtsson, 2012d).)

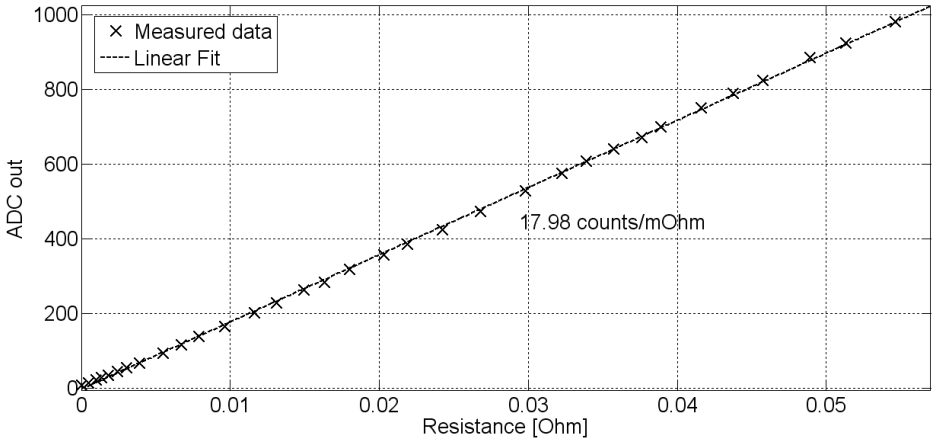


Fig. 6.20 Milliohm meter with digital lock-in; calibration data (Bengtsson, 2012d)

When measuring resistances of the order of milliohm, not only must you be careful to eliminate the wiring resistance, you must also make sure to control the *contact* resistances. There is always a contact resistance between two "not perfectly smooth" surfaces (Kister, 1998) which may very well be as high as 1 Ω. Oxide coatings of contact surfaces may also cause contact resistance (Timsit, 2012). All these problems are eliminated if a high-impedance instrumentation amplifier is used; the circuit in figure 6.19 is a 4-terminal Kelvin probe (Wolf and Smith, 2004). Another problem that must be considered in microohm/microvolt measurements is the interference of thermo emfs in connections between different wire materials; this effect is eliminated here due to the fact that we subtract samples which cancel any thermo emfs.

6.6 Conclusions

This work has demonstrated how sub-milliohm measurements can be performed with a simple 8-bit microcontroller and a PSD firmware algorithm. The resolution is of the same order as a commercial desktop LIA (Stanford, 2001) would produce and with only minor changes to the firmware algorithm, both the I- and the Q-part can be detected (Li et al, 2011) which would make the design useful also for complex impedance measurements (or for the general case where the detection electronics introduce a phase shift).

Chapter 7

Conclusions

The subject of “embedded measurement systems” is an interdisciplinary mixture of physics, electrical measurement technique and microcomputer technology. This work has focused on microcontroller design of electronic measurement systems and focus has been on designing cost-efficient solutions. That indicates using general-purpose, inexpensive, 8-bit microcontrollers and to provide solutions that can be implemented also in inherently digital targets with no, or a minimum of, on-chip integrated analog blocks of circuitry. For most of the designs presented in this thesis, a “direct” sensor/signal approach has been used to solve the measurement problem. This technique typically only requires a few bidirectional I/O-pins on the controller and a few passive components (like some resistors or a capacitor). That approach makes the measurement system solutions very cost-efficient. Most of the work in this thesis has been focused on microcontrollers, but the inherent digital nature of the proposed ideas, makes them suitable also for FPGAs/CPLDs.

This work has defined the concept of an *embedded measurement system* as a system confined to one embedded controller or one single pcb that is dedicated to measure one or just a few physical quantities. The concept of an embedded measurement system is a merging of *embedded systems* and *measurement systems* and the main purpose is to demonstrate that expensive and power-consuming desktop instruments can often be replaced with an inexpensive, low-power embedded measurement system. This has been demonstrated repeatedly in this work.

Chapter 8

Acknowledgements

I would first of all like to thank **Hans Odelius** for all the pcbs and without whose expertise in pcb design and electronics this work would not have been possible.

I would also like to thank Professor **Dag Hanstorp** for helping me with the manuscript and docent **Mattias Goksör** for making it all possible.

Appendix A

Summary of appended papers

PAPER I: *Sens. Actuators A: Phys (2012), doi: 10.1016/j.sna.2012.02.048.* This paper describes how analog voltage signals can be interfaced directly to an inherently digital controller using a few passive components only. It is based on the "direct sensor-to-controller" technique introduced by Cox (1997), Richey (1997), Baker (1999) and Bierl (1996) in the mid 90th. A capacitor is charged to the input analog voltage level and if that exceeds the input logic high threshold of an I/O-pin, the capacitor is discharged and the discharging time is proportional to the analog voltage. If the input voltage cannot charge the capacitor to the input logic high threshold of an I/O-pin, the capacitor is instead charged to that level. The charging time depends on the difference between the capacitor's voltage level and the input logic high threshold. So, whether or not the input voltage reaches the input logic high threshold of the digital I/O-pin, the input voltage level can be determined from either the charging or the discharging times. The proposed design take advantage of the fact that digital I/O-pins on most embedded systems are bidirectional and can be either high-impedance (input) or low-impedance (output) high/low.

PAPER II: *Rev. Sci. Instrum., **83**, 045107 (2012), doi: 10.1063/1.3700192*. This work describes how a high-resolution TDC based on time-stretching can be implemented in a microcontroller design. Again, I take advantage of the bidirectional property of digital I/O-pins to design a time-stretcher; the length of the input pulse is stretched by collecting the pulse charge on a capacitor and then discharges it slowly through a high-resistance resistor. A diode make sure

that the discharging current is directed through the resistor only (and not back through the original pulse source). Time stretching factors of 1000-2000 is demonstrated and pulse width measurements with sub-nano resolution are reported.

PAPER III: *Rev. Sci. Instrum., **83**, 075103 (2012), doi: 10.1063/1.4731683.* In this paper a digital lock-in amplifier is implemented in a microcontroller and applied to resistance measurements of the order or micro-ohms. The resistance under test is connected to two digital output pins of the controller and by alternating their outputs high and low, the current direction through the resistor is alternating. The rate at which the current is alternating corresponds to the lock-in amplifier's reference signal. The lock-in detection is implemented in firmware by a sampling technique suggested by Momo et al in 1981; I take two samples separated by a time corresponding to a phase angle π. By subtracting these samples and accumulating a lot of pair-wise subtracted samples, any signal with a frequency not equal to the reference frequency is efficiently cancelled. A copper wire with a 0.6 mm diameter was used to calibrate the system and changes in the length of the wire of single millimeters were detectable; a resistance resolution of 56 $\mu\Omega$/count was reported.

References

Abramowitch, D., 2002. “Phase-Locked Loops: A Control Centric Tutorial”, *Proceedings of the 2002 ACC*, [online] (Updated September 13, 2008). Available at <http://dabramovitch.com/ pubs/pll_tutorial.pdf> [Accessed November 29, 2011].

Adams, T. M., 2002. “G104-A2LA Guide for Estimation of Measurement Uncertainty in Testing”, [online] (Updated February 29, 2008) Available at <http://www.a2la.org/guidance/ est_mu_testing.pdf> [Accessed January 2, 2012].

Adams, D., 2009. “Introduction to capacitors”. CapSite. [online] (Updated: September 15, 2009) Available at <http://my.execpc.com/~endlr/esr.html> [Accessed November 17, 2011].

Agilent, 2004a. “Fundamentals of Time Interval Measurements”, Agilent Technologies, Application Note 200-3, [online] (Updated October 15, 2004). Available at <http://www.leapsecond.com/pdf/an200-3.pdf> [Accessed December 16, 2011].

Agilent, 2004b. “Fundamentals of Electronic Counters”, Agilent Technologies, Application Note 200, [online] (Updated October 15, 2004). Available at <http://www.leapsecond.com/pdf/ an200.pdf> [Accessed December 23, 2011].

Aloisio, A., Branchini, P., Cicalese, R., Giodano R., Izzo, V., Loffredo, S. and Lomoro R., 2009. “High-Resolution Time-to-Digital Converter in Field-Programmable Gate Array”, [online] (Updated August 6, 2009). Available at <http://cdsweb.cern.ch/record/1158663/files/ p383.pdf> [Accessed December 23, 2011].

Analog, 2010a. “Low Cost, Precision IC Temperature Transducer”. Analog Devices Inc., [online] (Updated September 16, 2010). <Available at https://www1.elfa.se/data1/wwwroot/assets/ datasheets/st646844_e.pdf> [Accessed November 20, 2011].

Analog, 2010b. “DI CMOS Protected Analog Switches”, Analog Device Inc., [online] (Updated April 15, 2010) Available at <https://www1.elfa.se/data1/wwwroot/assets/datasheets/ di641839-641840_e.pdf> [Accessed November 28, 2011].

Analog, 2011a. “Interactive Design Tools: Sigma-Delta Analog-to-Digital Converters: Sigma-Delta ADC Tutorial”. Analog Devices Inc. [online] (Updated: dynamic) Available at <http://designtools.analog.com/dt/sdtutorial/sdtutorial.html> [Accessed September 14, 2011].

Analog, 2011b. ”ADC592: Current Output – Precision IC Temperature Transducer”. Analog Devices Inc., [online] (Updated: dynamic) Available at <http://www.analog.com/en/mems-sensors/analog-temperature-sensors/ad592/products/product.html> [Accessed November 20, 2011].

Analog, 2011c. “Precision ±2 g Dual Axis, PWM Output Accelerometer ADXL212”, Analog Devices Inc., [online] (Updated June 6, 2011) Available at <http://www.analog.com/static/ imported-files/data_sheets/ADXL212.pdf> [Accessed December 8, 2011].

Arnold K., 2004. “Embedded Controller Hardware Design”, Elsevier Science, Embedded Technology series, ISBN: 1-878707-52-3, Burlington MA (2004).

Aroca, R.V. and Caurin, G.A.P, 2009. “A Real Time Operating Systems (RTOS) Comparison”. In: *Workshop de Sistemas Operacionais (WSO) – Congresso da Sociedade Brasileira de Computação (CSBC)*, 2009. Bento Gonçalves, Brazil 22-23 July 2009. [online] (Updated 23 May 2009). Available at <http://sites. google.com/site/rafaelaroca/artigos > [Accessed April 8, 2011].

Askdefine, 2011. ”Define microcontroller”, [online] (Updated April 17, 2011). Available at <http://microcontroller.askdefine.com/> [Accessed May 23, 2011].

Atmel, 2011a. “ATtiny 4/5/9/10 Preliminary Summary”, Atmel Inc., [online] (Updated January 28, 2011) Available at <http://www.atmel.com/dyn/resources/prod_documents/8127S.pdf> [Accessed September 13, 2011].

Atmel, 2011b. “AVR401: 8-bit precision AD-converter””, Atmel Inc., [online] (Updated January 28, 2011) Available at <http://www.atmel.com/dyn/resources/prod_documents/ doc0953.pdf> [Accessed December 23, 2011].

Atmel, 2011c. “8-bit AVR® XMEGA A Microcontroller”. Atmel Corp. [online] (Updated July 13, 2011). Available at <http://www.atmel.com/dyn/resources/prod_documents/doc8077.pdf>]Accessed December 25, 2011].

Baker C.B., 1999. “Building a 10-bit Bridge Sensing Circuit using the PIC16C6XX and MCP601 Operational Amplifier”. Microchip Technology Inc., Application Note AN717, Chandler, Arizona, 1999.

Barr, M., 2003. “Special Report: Choosing RTOS”. Embedded Systems Programming, January 2003.

Bartling, J., 2009. “Low-cost, High-Resolution Time-Measurement Application”, The ECN Daily, [onine] (Updated July 16, 2011). Available at <http://www.ecnmag.com/Articles/2009/07/ Low-Cost,-High-Resolution-Time-Measurement-Application/> [Accessed March 14, 2012].

Baskiyar, S., 2002. “A Survey of Real-time Operating Systems”. In: *NPDPA-2002, Networks, Parallel and Distributed Processing and Applications*, Tsukuba, Japan 2-4 October 2002. [online] (Updated 13 October 2006). Available at: <http:// www.ece.stevenstech.edu/ ~ymeng/ courses/CPE555/papers/ rtos_paper.pdf> [Accessed 12 April 2011].

Beiss U., 2007. “An introduction to Delta Sigma Converters.” [online] (Updated June 10, 2008). Available at <http://www.beis.de/Elektronik/DeltaSigma/DeltaSigma.html> [Accessed April 19, 2011].

Bell, S., 2009. “A Beginner’s Guide to Uncertainty of Measurement”, [online] (Updated July 13, 2009) Available at <http://www.wmo.int/pages/prog/gcos/documents/gruanmanuals/ UK_NPL/mgpg11.pdf> Accessed [December 25, 2011].

Bellescize de, H, 1932. L’Onde Electrique **11**, pp. 230-240, (1932).

Bengtsson L. and Karlström B., ”Transformer – från jω till Wavelets”, Studentlitteratur, ISBN 978-91-44-04701-0, Lund, Sweden, 2007.

Bengtsson L., 2012a. “Elektriska mätsystem och mätmetoder”, 3rd ed., ISBN 978-91-44-08068-0, Studentlitteratur, Lund, 2012.

Bengtsson L., 2012b, Sens. Act. A., **179** (2012), pp. 105-113.

Bengtsson, L., 2012c, Rev. Sci. Instr., Vol. **83**(4), Art. no 045107, (2012). doi: 10.1063/1.3700192.

Bengtsson, L., 2012d, Rev. Sci. Instr., Vol. **83**, Art. no 075103, (2012). doi: 10.1063/1.4731683.

Bentley J.P., 1995. ”Measurement Systems”, 3rd ed., ISBN 0-582-23779-3. Harlow: Prentice-Hall.

Bierl L., 1996. “Precise Measurements with the MSP430”. Application Report, Texas Instruments, 1996.

BiPOM, 2006. “Microcontroller to Sensor Interfacing Techniques”, BiPOM Electronics, Inc., [online] (Updated November 23, 2009). Available at <http://www.bipom.com/documents/ lectures/Microcontroller%20to% 20Sensor%20Interfacing%20Techniques.pdf> [Accessed June 2, 2001].

Bryant, J., 2002. “Ask the Application Engineer”, Analog Devices, Application Note AN-361, [online] (Updated October 9, 2002) Available at <http://www.analog.com/static/imported-files/application_notes/84860375AN361.pdf> [Accessed December 12, 2011].

Burr-Brown, 1995. “INA128/INA129: Precision, Low Power Instrumentation Amplifiers”. [online] (Updated January 6, 2012) Available at <http://www.ti.com/lit/ds/symlink/ina128.pdf> [Accessed February 18, 2012].

Candy, J.C and G.C. Temes, 1992. “Oversampling Methods for A/D and D/A Conversion”. In IEEE Press, “Oversampling Delta-Sigma Data Converters: Theory, Design and Simulation”, pp. 1-29, 1992.

Carley L.R., 1987. IEEE Transitions on Circuits and Systems, vol. **34**, pp. 83-91 (Jan 1987).

Cespiva, D. and Evans D.A., 2009. “Characterization of an Evans Tantalum Hybrid Capacitor THQA2016502 – 5 mF / 16 Volts.” [online] (Updated: September 14, 2009) Available at <http://www.evanscap.com/pdf/THQA2016502-Characterization-paper.pdf> [Accessed November 17, 2011].

Chavan, A.V. and Wise, K.D., 2001. Journal of MicroElectroMechanical Systems, vol. **10**, no. 4, (December 2001).

Chu L.L. and Gianchandani Y.B., 2003. Journal of Micromechanics and Microengineering, **13**, pp. 279-285, (2003).

Cox D., 1997. “Implementing Ohmmeter/Temperature Sensor”. Microchip Technology Inc., Application Note AN512, Chandler, Arizona, 1997.

Custodio, A., Pallàs-Arreny R. and Bragós R., 2001a. IEEE Transaction on Instrumentation and Measurement, vol. **50**, no 6, pp. 1644-1647, (December 2001).

Custodio, A., Bragós, R. and Pallàs-Areny, R., 2001b. “A Novel Sensor-Bridge-to-Microcontroller Interface.” In: *IEEE Instrumentation and Measurement Technology Conference*, Budapest, Hungary, May 21-23, 2001.

Dieter P., Baker B.C., Butler D. and Darmawaskita H., 1998. ”Make a Delta-Sigma Converter Using a Microcontroller’s Analog Comparator Module”. Microchip Inc., Application Note AN700. [online] (Updated 14 June, 2009). Available at <http://ww1.microchip.com/downloads/en/ AppNotes/00700a.pdf> [Accessed April 26, 2011]

Direct Industry, 2011. ”The Microchip advantage”, [online] (Updated May 23, 2011). Available at <http://pdf.directindustry.com/pdf/microchip-technology/8-bit-picmicrocontrollers/23455-72917-_2.html> [Accessed May 25, 2011].

Doeblin E.O., 1990. ”Measurement Systems – Applications and Design, 4th ed”, ISBN 0-07-100697-4, International Edition. New York:McGraw-Hill.

Dorrington, A.A. and Künnemeyer R., 2002. “A simple microcontroller based digital lock-in amplifier for the detection of low level optical signals”. *Proc. of the First IEEE International Workshop on electronic Design, Test and Applications* (DELTA’02). [online] (Updated May 31, 2011) Available at <http://researchcommons.waikato.ac.nz/bitstream/10289/3215/1/A%20simple%20microcontroller.pdf> [Accessed November 30, 2011].

Feng, X. G., Zelakiewicz, S. and Gramila, T.J., Rev. Sci. Instr., **70**, No. 5, pp 2365-2371, (May 1999).

Fisher, E.H., 2003. “The Evolution of the modern Lock-in Amplifier”, DL Instruments Technical Notes IAN35, [online] (Updated December 23, 2003). Available at <http://www.dlinstruments.com/technotes/index.html> [Accessed November 28, 2003].

Freescale, 2009. “MC9S08JM60/MC9S08JM32 Data Sheet”. Freescale Semiconductors Inc., Rev. 3, 1/2009. [online] (Updated January 23, 2009) Available at <http://www.freescale.com/ files/microcontrollers/doc/data_sheet/MC9S08JM60.pdf> [Accessed September 13, 2009).

Gaitàn-Pitre, J.E., Gasulla, M. and Pallàs-Areny, R., 2009. IEEE Transactions on Instrumentation and Measurement, vol. **58**, no. 9, pp. 2931-2937, (September 2009).

Girard T., 2011. "Understanding effective bits", Application Note AN95091, Signatec, 2011. [online] (Updated February 4, 2011) Available at <http://www.fastcomtec.com/fwww/ datashee/tra/effbits.pdf> [Accessed May 13 2011].

Goodall, W.M., 1947. "Telephony by Pulse Code Modulation", Bell Systems Technical Journal, Vol. 26, pp. 395-409, July 1947.

Goodman, D.J., 1969. Bell Systems Technical Journal, vol. **48**, pp. 321-343 (Feb 1969).

Gordon B.M. and Talambiras R.P., 1955. "Signal Conversion Apparatus", U.S. Patent 3.108,266, filed July 22, 1955, issued October 22, 1963.

Greggio, D., 2011. Microchip Forums, Microchip Inc., [online] (Updated December17, 2011) Available at <http://www.microchip.com/forums/m621050-print.aspx> [Accessed February 9, 2012].

GS1 Mobile Com, 2008. "Mobile Commerce: opportunities and challenges". [online] (Updated May 6, 2009). Available at: <http://www.gs1.org/docs/mobile/GS1_ Mobile_Com_ Whitepaper.pdf> [Accessed May 24, 2011].

Hamblen, J.O., Hall T.S. and Furman M.D., 2004. "Rapid Prototyping of Digital Systems". SOPC ed., ISBN 978-0-387-72670-0, Springer, New York, USA, 2008.

Hauser, M.W., 1991. Journal of Audio Engineering Society, vol. **39**, No. 1(2), pp. 3-26, (January/February, 1991).

HBM, 2006. "C Series Strain Gauges." [online] (Updated February 10, 2006) Available at <http://www.hbm.com/en/menu/products/strain-gages-accessories/strain-gages-for-stress-analysis/documentation/categorie/strain-gages-universal-foil-strain-gages/product/c-series/backPID/strain-gages-for-stress-analysis/?cHash=0d2d548e560a39995508edb1fb309b04&parent=364&pg=3> [Accessed October 8, 2011].

Henzler, S., 2010. "Time-to-Digital Converters", Springer Series in Advanced Microelectronics 29, DOI 10.1007/978-90-481-8628-0__2, 2010.

HI-TECH, 2011. "HI-TECH C Tools for the PIC18 MCU Family." [online] (Updated September 30, 2011) Available at <http://ww1.microchip.com/downloads/en/DeviceDoc/PICC_18_9_80_manual.pdf> [Accessed January 2, 2012].

Horowitz, P. and Hill, W., 1990. "The Art of Electronics", 2nd ed. Cambridge University Press, ISBN 0-521-37095-7, Cambridge, Massachusetts, USA, 1990.

Humirel, 2011. "Relative Humidity Sensor, HS 100 / HS 1101", Humirel [online] (Updated: Dynamic) Available at <http://pdf1.alldatasheet.com/datasheet-pdf/view/47866/HUMIREL/ HS1101.html> [Accessed October 2, 2011]

Jarman D., 1995. "A Brief Introduction to Sigma Delta Conversion", Intersil Application Note AN9504, May 1995. [online] (Updated November 1, 2002). Available at <http://www.intersil.com/data/an/an9504.pdf> (Accessed April 21 2011).

Jordana J., Reverter F. and Pallàs-Areny R., 2003. "Uncertainty in resistance measurements based on microcontrollers with embedded time counters." In: *IMTC 2003 - Instrumentation and Measurement Technology Conference*, Vail Colorado USA, May 20-22, 2003.

Jovanovic, G.S. and Stojcev, M.K., 2009. Scientific Publications of the state University of Novi Pazar. Ser. A: Appl. Math. Inform. And Mech. vol **1**, pp. 11-20 (2009).

Kalisz, J., Pawlowski, M. and Pelka, R., 1985. J. Phys. E: Sci. Instrum. vol **18**, pp. 444-452, (1985).

Kalisz, J., Szplet, R., Pasierbinski, J. and Poniecki, A., 1997. IEEE Transactions on Instrumentation and Measurement, vol. **46**, no.1, pp. 55-55, (February 1997).

Kang J. and Sung W., 1997. "Fixed-Point C Compiler for TMS320C50 Digital Signal Processor", [online] (Updated March 19, 1997). Available at <https://nats-www.informatik.uni-hamburg.de/intern/proceedings/1997/ICASSP/pdf/scan/ic970707.pdf> [Accessed March 1, 2012].

Kester W., 2004. "The Data Conversion Handbook.", Analog Devices Inc. ISBN 0-916550-27-3, [online] (Updated May 1, 2007), Available at <http://www.analog.com/library/ analogDialogue/archives/39-06/data_conversion_handbook.html> [Accessed September 9, 2011].

Kester W., 2006. "ADC Input Noise: The Good, The Bad and The Ugly. Is No Noise Good Noise?". Analog Devices Inc., Analog Dialogue 40-02, February, 2006. [online] (Updated January 30, 2006). Available at <http://www.analog.com/library/analogDialogue/archives/40-02/adc_noise.pdf> [Accessed September 19, 2001].

Kester W., 2009. "ADC Architectures III: Sigma-Delta ADC Basics". Analog Devices, Application note MT22, Rev. A, 10/08, 2008. [online] (Updated February 10, 2009). Available at <http://www.analog.com/static/imported-files/tutorials/MT-022.pdf> [Accessed May 1, 2011].

Kister, J., 1998. "Introduction to the physics of contact resistance", Probe Technology, SouthWest Test Workshop, San Diego, 1998. [online] (Updated December 23, 2009) Available at<http://www.swtest.org/swtw_library/1998proc/PDF/S01_kister.PDF> [Accessed February 15, 2012).

Kornecki, A.J. and Sorton, E., 2003. Systemics, Cybernetics and Informatics, **1**(6), pp. 5-10 (2003).

Kuglestadt, T., 2000. "The operation of the SAR-ADC based on charge redistribution." Analog Application Journal, SLT176, February 2000, Texas Instruments Inc. [online] (Updated August 6, 2011), Available at <http://www.ti.com/lit/an/slyt176/slyt176.pdf>, [Accessed September 8, 2011].

Kwang-Woon L., Myungchul K. and Jangho Y., 2008. IEEE Trans. Ind. Appl., vol. **44**, no. 5, pp. 1606-1613, (September/October 2008).

Lattice, 2001. "Adding a sign bit a unipolar ADC", [online] (Updated November 12, 2008) Available at <http://www.msc-ge.com/download/lattice/files/cs1005.pdf> [Accessed February 9, 2012].

Lean Six Sigma, 2011. "Measurement System Analysis", [online] (Updated May 23, 2011). Available at <http://www.moresteam.com/toolbox/t403.cfm> [Accessed May 23, 2011].

Lepkowski J., 2004. "Temperature Measurement Circuits for Embedded Applications." Microchip Technology Inc., Application Note AN929, Chandler, Arizona, 2004.

Levine, P.M. and Roberts, G.W., 2004. "A high-resolution flash time-to-digital converter and calibration scheme". In: *ITC International Test Conference 2004*. [online] (Updated May 16, 2006). Available at <http://www.itcprogramdev.org/itc2004proc/papers/pdfs/0040_2.pdf> [Accessed December19, 2011].

Li, G., Zhou, M., He, F. and Lin, L., 2011. Rev. Sci. Instr., **82**, pp. 095106-1 - 095106-6, (2011).

Lis, J., 1995. "Noise Histogram Analysis". Cirrus Logic Application Note AN37. [online] (Updated April 2, 2009) Available at <http://www.cirrus.com/en/pubs/appNote/an37.pdf> [Accessed September 19, 2011].

Lourens R. and Kell C., 2004. "Tire Pressure Monitoring (TPM) System. Microchip Application note AN238. [online] (Updated August 13, 2008), Available at <http://ww1.microchip.com/ downloads/en/AppNotes/00238b.pdf> (Accessed August 30, 2011).

Lynn P.A. and Fuerst W., 1998. "Introductory Digital Signal Processing", John Wiley & Sons, ISBN 0471 97631 8, Chichester, England, 1998.

MAXIM, 2001a, "Understanding Pipelined ADCs", Application Note 1023, [online] (Updated October 6, 2007). Available at <http://pdfserv.maxim-ic.com/en/an/AN1023.pdf> [Accessed June 2, 2011).

MAXIM, 2001b. “Understanding SAR ADCs”. Application Note 1080. [online] (Updated March 22, 2011) Available at <http://pdfserv.maxim-ic.com/en/an/AN1080.pdf> [Accessed September 8, 2011].

MAXIM, 2003. “Demystifying Delta-Sigma ADCs”, Application Note 1870. [online] (Updated Mars 2003, 2011). Available at <http://www.maxim-ic.com/app-notes/index.mvp/id/1870> [Accessed May 6, 2011].

MAXIM, 2011. “Delta-Sigma ADCs Replacing Integrating ADCs for Panel Meters.” Application Note 2102, [online] (Updated April 5, 2011), Available at <http://pdfserv.maxim-ic.com/en/an/AN2102.pdf> [Accessed September 22, 2011]

Merriam-Webster Dictionary, 2011. [online] (Updated May 23, 2011). Available at <http://www.merriam-webster.com/dictionary/measure> [Accessed May 23, 2011].

Merritt B., 1999. “MPS430 Based Digital Thermometer.” Texas Instruments, Application Report SLAA038, 1999.

Microchip, 2003. “PIC18FXX8 Data Sheet”. Microchip Inc., DS41159C, Tuscon, Arizona, 2003. [online] (Updated: dynamic) Available at <http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010301> [Accessed Mars 12, 2005].

Microchip, 2009. “PIC18F2480/2580/4480/4580 Data Sheet”. Microchip Inc., DS39637D, Tuscon, Arizona, 2009. [online] (Updated: November 16, 2009) Available at <http://ww1.microchip.com/downloads/en/DeviceDoc/39637d.pdf> [Accessed December 27, 2011].

Microchip, 2010. “Charge Time Measurement Unit (CTMU)”, Microchip Techn. Inc., [online] (Updated June 1, 2010). Available at <http://ww1.microchip.com/downloads/en/ DeviceDoc/70635A.pdf> [Accessed March 14, 2012].

Microchip, 2011. “PIC microcontrollers with XLP are battery friendly”. [online] Available at: <http://www.microchip.com/en_us/technology/xlp/> [Accessed May 24, 2011].

Microcontroller, 2011. ”STC 8051 Microcontroller”, [online] (Updated May 23, 2011). Available at <http://microcontroller.com/STC_8051_Microcon trollers.htm> [Accessed May 25, 2011].

Momo, F., Ranieri, G.A., Sotgui, A. and Terenzi, M., 1981. J. Phys. E: Sci. Instrum., Vol **14**, pp. 1253-1256 (1981).

Motorola, 2001. “10 kPa Uncompensated Silicon Pressure Sensors.” Freescale Semiconductor Inc., [online] (Updated April 16, 2010) Available at <https://www1.elfa.se/data1/wwwroot/ assets/datasheets/tw647372_e.pdf> [Accessed October 8, 2011].

Nastase, A.S., 2009. “Design a Bipolar to Unipolar Converter to drive an ADC”, MasteringElectronicDesign.com, [online] (Updated October 26, 2009) Available at <http://masteringelectronicsdesign.com/design-a-bipolar-to-unipolar-converter/> [Accessed February 9, 2012].

Nutt R., 1968. Rev. Sci. Instrum. **39**, pp. 1342-1345, (1968).

Nyquist H., 1928. AIEE Trans., vol. **47**, pp. 617-644, (1928).

Paikin A., 2003. “Delta-Sigma ADC”, National Semiconductor, [online] (Updated April 15, 2003). Available at <http://www.hitequest.com/Kiss/DeltaSigma.htm> (Accessed April 19, 2011).

Pallàs-Areny, R. and Webster, J.G., 2001. “Sensors and signal conditioning”, 2nd ed., John Wiley & Sons, New York, 2001.

Pedroni V.A., 2004. “Circuit Design with VHDL”, MIT Press, ISBN 0-262-16224-5, Cambridge, Massachusetts, USA, 2004.

Pereira J.M.D., Girão P.M.B.S. and Postolache O., 2001. IEEE Instrumentation & Measurement Magazine, pp. 26-39, (December 2001).

Peter D., Baker B.C. and Butler D., 1998."Make a Delta-Sigma Converter Using a Microcontroller's Analog Comparator Module", Microchip Application Note, AN700, 1998. [online] (Updated 14 June 2009). Available at <http://www.microchip.com/stellent/idcplg? IdcService=SS_GET_PAGE&nodeId= 1824&appnote=en011642> [Accessed 26 April 2011]

Pohlmann, K.C., 2005. "Principles of digital audio". 4th ed., McGraw-Hill, ISBN 0-07-134819-0. New York (USA), 2005.

Porat D.I., 1973. IEEE Transactions on Nuclear Science, vol. **20**(5), pp. 36-51, (October 1973).

Proakis J.G. and Manolakis D.M., 1992. "Digital Signal Processing – Principles, algorithms and Applications". New York: Macmillan Publishing Company.

Reverter F., Jordana J. and Pallàs-Areny R., 2003a. "Program-Dependent Uncertainty in Period-to-Code Converters Based on Counters Embedded in Microcontrollers", In: *IMTC 2003 – Instrumentation and Measurement Technology Conference*, Vail Colorado USA, May 20-22, 2003.

Reverter F., Jordana J. and Pallàs-Areny R, 2003b. "Internal trigger errors in microcontroller-based measurement", In: *Proceedings of XVII IMEKO World Congress*, June 22-27, Dubrovnik, Croatia, 2003.

Reverter F. and Pallàs-Areny R., 2004. Measurement Science and Technology, **15**, pp.2157-2162, (2004).

Reverter F., Gasulla M. and Pallàs-Areny R., 2004. "A Low-cost Microcontroller Interface for Low-Value Capacitive Sensors." In: *IMTC 2004 – Instrumentation and Measurement Technology Conference,* Como, Italy, 18-20 May, 2004.

Reverter F. and Pallàs-Areny R., 2005. "Direct Sensor-to-Microcontroller Interface Circuits", ISBN 978-8426713803, Barcelona, Spain: Maracombo, 2005.

Reverter F., Jordana J., Gasulla M. and Pallàs-Areny R., 2005a. Elsevier: Sensors and Actuators A*,* **121**, pp 78-87, (2005).

Reverter F., Gasulla M. and Pallàs-Areny R., 2005b. "Analysis of Power Supply Interference Effects on Direct Sensor-to-Microcontroller Interfaces", In: *IMTC 2005 – Instrumentation and Measurement Technology Conference*, Ottawa, Canada, May 17-19, 2005.

Reverter F. and Pallàs-Areny R., 2006. Elsevier: Sensors and Actuators A, **127**, pp 74-79, (2006).

Reverter F and Casas Ò., 2008. Sensors and Actuators A. **143**, pp. 315-322 (2008).

Reverter, F. and Casas, Ò., 2010a. Measurement Science and Technology, **21**, doi: 10.1088/0957-0233/6/065203, (2010).

Reverter, F. and Casas, Ò., 2010b. IEEE Transactions on Instrumentation and Measurement, vol. **59**, no 10, pp. 2763-2769 (October 2010).

Reverter, F., 2012. J. Low Power Electron. Appl., vol **2**, no 4, pp 265-281, 2012.

Richey R., 1997. "Resistance and Capacitance Meter Using a PIC16C622", Microchip technology Inc., Application Note AN611, Chandler Arizona, 1997.

Roberts, G.W. and Ali-Bakhshian, M., 2010. IEEE Transactions on Circuits and Systems-II: Express Briefs, vol. **57**, no. 3, pp. 153-157, (March 2010).

Roon von, T., 1991. "Phase Locked Loops", [online] (Updated November 18, 2010) Available at <http://www.sentex.net/~mec1995/gadgets/pll/pll.html> [Accessed November 28, 2011].

Rota, H.R., 2005. "Phase-locked Loop", [online] (Updated December 28, 2005) Available at <seit.unsw.adfa.edu.au/staff/sites/hrp/teaching/Electronics4/docs/PLL/pllJune03.pdf> [Accessed November 27, 2011]

Räisänen-Ruotsalainen, E., Rahkonen, T. and Kostamovaara, J., 2000. IEEE Journal of Solid-State Circuits, vol. **35**, no. 10, pp. 1507-1510, (October 2000).

Schelleng J.C., 1946, “Code Modulation Communication Systems”, U.S. Patent 2,453,461, filed June 19 1946, issued November 9, 1948.

Schuchman, L., 1964. IEEE Transactions on Communication Technology, vol. **12**, issue 4, pp. 162-165, (1964).

Scott, J.L., 2002. “Introduction to lock-in amplifiers.” DL Instruments Technical Notes IAN47, [online] (Updated August 12, 2002) Available at <http://www.dlinstruments.com/ technotes/index.html> [Accessed November 24, 2011]

Sedra, A.S. and Smith K.C., 1990. “Microelectronic Circuits”, 3rd ed, (International Edition), Saunders College Publishing, Philadelphia, USA, 1990.

Shannon C.E., 1949. Proc. Institute of Radio Engineers, vol. **37**, no. 1, pp. 10-21, (January, 1949).

Silicon Laboratories, 2009. “Improving ADC Resolution by Oversampling and Averaging”. Silicon Laboratories, Application Note 118, Rev. 1.2, 12/03, [online] (Updated March 26, 2009) Available at <http://www.silabs.com/Support%20Documents/TechnicalDocs/an118.pdf> [Accessed September 19, 2011].

Skansholm J. and Bilting U., 2000. “Vägen till C”, Studentlitteratur, ISBN 91-44-01468, Lund, 2000.

Skoog D.A. and Leary J.J., 1992. “Principles of Instrumental Analysis”, 4th ed, pp.52-53, (International Edition) Saunders College Publishing, Philadelphia, USA, 1992.

Smith B.D., 1953 “Coding by Feedback Methods," *Proceedings of the I. R. E.*, Vol. 41, August 1953, pp. 1053-1058.

Socher, G., 2012., “A digital DC power supply”, [online] (Updated: Dynamic) Available at <http://linuxfocus.org/English/June2005/article379.shtml> [Accessed February 10, 2012).

Soldera, J.D.B, Espindola, M. and Olmos, 2005. A. “Implementing a 10-bit Sigma-Delta Analog-to-Digital Converter Using the HC9S08Rx MCU Family Analog Comparator”, Freescale Semiconductor, Application Note AN2688, Rev. 0.1, 2005.

Stanford, 2001. “Model SR530 Lock-In Amplifier”, Standford Research Systems, [online] (Updated March 30, 2006). Available at <http://www.thinksrs.com/downloads/PDFs/Manuals/ SR530m.pdf> [Accessed December 1, 2011].

Stanford, 2006. ”About Lock-In Amplifiers”, Stanford Research Systems, Application Note #3, [online] (Updated March 30, 2006) Available at <http://www.thinksrs.com/downloads/ PDFs/ApplicationNotes/ AboutLIAs.pdf> [Accessed February 15, 2012]

STMicroelectronics, 2008. “Implementation of sigma-delta ADC with ST7FLUE05/09”, STMicroelectronics Application Note 1827, 2008. [online] (Updated May 15, 2012). Available at <http://www.st.com/internet/com/TECHNICAL_RESOURCES/TECHNICAL_ LITERATURE/APPLICATION_NOTE/CD00010935.pdf> [Accessed July 4, 2012].

Stoltenberg, J. and Vilches O.E., 1997. “Introduction to the Phase Sensitive (Lock-In) Detector”, [online] (Updated July 1, 2010), Available at <http://advancedlab.org/mediawiki/index.php/ Appendix_D:_the_Phase_Sensitive_%28Lock-In%29_Detector> [Accessed November 28, 2011].

Svärdström, A., 1999. “Signaler och system”, Studentlitteratur, ISBN 91-44-00811-2, Lund, Sweden, 1999.

Timmerman, M., van Beneden, B. and Uhres, L., 1998. ”RTOS Evaluations Kick Off!”. Real-Time Magazine, 98-3. [online] (Updated 13 September 2000). Available at <http://www. omimo.be/magazine/ 98q3/1998q3_p006.pdf> [Accessed 11 April 2011].

Timsit, R.S., 2012. “Electrical Contact Resistance: Review of Elementary Concepts”, [online] (Updated February 2, 2012) Available at <http://www.connectorsupplier.com/tech_updates_Timron_Electrical ContactResistance_1-23-07.htm> [Accessed February 2, 2012].

UBICOM, 2000. “Sigma Delta ADC Virtual Peripheral”, [online] (Updated October 18, 2001) Available at <http://www.piclist.com/images/com/ubicom/www/http/pdf_files/e2/ an2.pdf> [Accessed May 2, 2011].

Unknown, 2007. “Sensors, Excitation and Linearization”, [online] (Updated October 6 13, 2007). Available at <http://media.wiley.com/product_data/excerpt/33/07803601/0780360133-2.pdf> [Accessed February 15, 2012].

UN Telecom, 2013. [online] (Updated Dynamic). Available at < http://www.huffingtonpost.com/2012/10/11/cell-phones-world-subscribers-six-billion_n_1957173.html > [Accessed Mars 22, 2013].

Viorel, C.P., 2006. “Microcontroller based Measurements: how to take out the best we can of them”, In: *Proceedings of the 8th WSEAS Int. Conf. on Mathematical Methods and Computational Techniques in Electrical Engineering*, Bucharest, Hungary, October 16-17, 2006.

Vogel. C and Johansson, H., 2006. “Time-Interleaved Analog-To-Digital Converters: Status and Future Directions.” *Proceedings of the 2006 IEEE International Symposium on Circuits and Systems (ISCAS 2006)*, Kos, Greece, May 21-24, 2006, pp.3386-3389.

Vu P., “Benefits of Sigma-Delta ADCs”, Test Edge Inc., [online] (Updated November 4, 2005) Available at <http://www.testedgeinc.com/docs/sigma_delta_adc.pdf> [Accessed May 2, 2011].

Wang, X., 1990. Rev. Sci. Instrum., **61** (7), pp. 1999-2001 (July 1990).

Ward, J., 2011. “Time Interval Measurement Literature Review”, [online] (Updated March 29, 2011). Available at <http://www.rrsg.ee.uct.ac.za/members/jon/activities/timcs.pdf> [Accessed December 19, 2011].

Weber P. and Windish C., 2007. “Build a complete industrial-ADC interface using a microcontroller and a sigma-delta modulator”, Electronic Design Strategy News, July 5, 2007, pp. 63-66. [online] (Updated August 1, 2007) Available at <http://www.edn-europe.com/buildacompleteindustrialadcinterfaceusingamicro controllerandasigmadeltamodulator+article+1668+Europe.html> [Accessed May 5, 2011].

Webster, J.G., 2007 (ed). “Time Interval Measurement”, Wiley Encyclopedia of Electrical and Electronics Engineering. [online] (Updated 13 July, 2007). Available at <http://onlinelibrary.wiley.com/ doi/10.1002/047134608X.W3989.pub2/pdf> [Accessed December 2011].

Wender R. and Ihme D., 2007. “How to Design a 16-bit Sigma-Delta Analog to Digital Converter”, Application Notes, Triad Semiconductors. [online] (Updated January 25, 2007) Available at <http://www.triadsemi.com/2007/01/25/how-to-design-a-16-bit-sigma-delta-analog-to-digital-converter/> [Accessed April 19, 2011].

Wenn, D., 2007. ”Implementing Digital Lock-In Amplifiers Using the dsPIC® DSC”, Microchip Technology Inc., Application Note AN115, [online] (Updated October 20, 2007) Available at <http://ww1.microchip.com/downloads/en/AppNotes/01115A.pdf> [Accessed February 10, 2012].

Wikipedia_1, 2011. ”Microcontroller”, [online] (Updated May 16, 2011). Available at <http://en.wikipedia.org/wiki/Embedded_system> [Accessed May 23, 2011].

Wikipedia_2, 2011. ”Embedded Systems”, [online] (Updated May 22, 2011). Available at <http://en.wikipedia.org/wiki/Embedded_system> [Accessed May 23, 2011].

Wolf, S. and Smith, R.F.M., 2004. “Electronic Instrumentation Laboratories”, Pearson-Prentice Hall, 2nd ed., ISBN 0-13-042182-0, Upper Saddle River, New Jersey, USA (2004).

Wolfson, R., 1991. Am. J. Phys., **59**, pp 569-572 (1991).

Wong, K.D., 2011. “Accurate Milliohm Measurement” [online] (Updated August 14, 2011). Available at <http://www.kerrywong.com/2011/08/14/accurate-milliohm-measurement/> [Accessed February 15, 2012].

Yedamale, P. and Bartling, J., 2011. “See what you can do with the CTMU”, Microchip Techn. Inc., AN1375. [online] (Updated May 6, 2011). Available at <http://ww1.microchip.com/downloads/en/ appnotes/01375a.pdf> [Accessed March 14, 2012].

Yiding, W., Yunhong, W. and Shi Z., 2007. “Errors analysis of spectrum inversion methods”. In: *Proc. of 2007 IEEE International Conference on Signal Processing and Communications* (ICSPC 2007), November 24-27, Dubai, United Arab Emirates, 2007. [online] (Updated: Dynamic) Available at <http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4728304> [Accessed December 1, 2011].