POLITECNICO DI MILANO

FACOLTÀ DI INGEGNERIA DELL'INFORMAZIONE

Corso di Laurea Magistrale in Ingegneria Automazione

Implementation of the high resolution high dynamic Time to Digital Converter

Relatore: Prof. Angelo Geraci

Correlatore: Prof. Andrea Abba

Tesi di Laurea di:

LIU TIANYI

Matr. 784859

Anno Accademico 2013-2014

Summary

The requirement of precise time intervals measurement has grown up rapidly during recent decades and its application covers a wide range of applications. In many applications both high dynamic range as well as high precision are needed while the traditional time measure technologies such as delay line based TDC and TAC do not optimally address these requirements at the same time.

In this work I have designed and prototyped a time measurement system based on a new device from Texas Instruments. The THS788 is a four-channel timing measurement unit (TMU) aiming for fast and accurate time measurements. This TMU has 13 ps resolution and guarantees 8 ps on single-shot accuracy. It has a FSR as large as 7 seconds.

The development of the presented time measurement system can be divided in two parts; i.e. hardware and software. In the hardware phase a Print Circuit Board (PCB) which contains the TMU, analog front-end and other ancillary circuit has been designed. The sought precision has requested many efforts designing many parts of the PCB, first of all in order to guarantee the signal integrity at most. The analog front-end is principally composed by a Constant Fractional Discriminator (CFD) which is an effective approach to minimize time-walk caused by amplitude variation of arriving signal. In order to have a complete solution for the time measurement task, a programmable device is added to implement all necessary function. The TMU board are connected to a ML507 Virtex-5 FPGA development board through high speed plug. The architecture of the software of the system is designed in a flexible and highly modular structure for ease modification of the code.

Moreover, the thermal control is always a critical part of sampling systems since heat can cause several undesired effects such as bias and drift effects that make result less accurate and precise. The THS788 is a high energy consuming device that dissipates a lot of heat when running. In the presented system a dedicated

temperature control module is implemented based on a peltier solid-state active heat pump that precisely controls the temperature over a wide range.

At the end, the performance of the system has been tested and verified including accuracy, linearity, and thermal stabilization. The performance of the CFD has been also validated.

The thesis consists of six chapters. In the first chapter, the basic concept of time interval measurement, some applications of time measurement and most commonly used time measurement methods are introduced. In the second chapter, the top layer architecture of the designed time measurement system is illustrated. The third chapter focuses on the hardware design while the fourth chapter describes the software design. In the fifth chapter the performance including resolution, linearity and amplitude related time walk are tested and verified.

Abstract

The work presents a system for accurate measurement of time intervals. The proposed architecture is based on the THS788, which is a four-channel timing measurement unit released by the Texas Instruments and provides 13 ps of resolution and 7 seconds of dynamic range. During the hardware design phase, a Print Circuit Board (PCB) which contains the TMU and other necessary circuit has been designed and a Constant Fractional Discriminator (CFD) is implemented as the analog front-end. In order to optimize the performance of the CFD a new numerical exhaustive method is proposed and shown its feasibility. Sample reading and histogram generation functions have been implemented in the presented system software and it has been designed in a flexible and highly modular way for easy readability and portability. A GUI interface is implemented for data analysis and system control. The Region of Interest (ROI) function is integrated in the firmware. Moreover, in this time measurement system a dedicated temperature control module is implemented which is based on a peltier solid-state active heat pump that allows controlling the TMU's temperature very efficiently. The test shows this time measurement system can offer 7 seconds of full scan range with resolution of 13ps, 10ps accuracy and 1.5% LSB DNL of linearity performance.

Abstract

L’ attività svolta si accentra su un sistema per la misura accurata di intervalli di tempo . L' architettura proposta si basa sul THS788, che è un modulo di misura del tempo a quattro canali prodotto dalla Texas Instruments e fornisce 13 ps di risoluzione e 7 secondi di range dinamico. Durante la fase di progettazione hardware, è stata sviluppata una scheda (PCB) che contiene il TMU e altri circuiti necessari tra cui un CFD (Constant Fractional Discriminator) che è stato implementato come front-end analogico. Per ottimizzare la prestazione del CFD si propone un nuovo metodo esaustivo numerico e ne viene mostrata la fattibilità. Delle funzioni per la generazione di istogrammi e per la lettura dei campioni sono stati implementati nel software di sistema presentato che è stato progettato in modo flessibile e altamente modulare per una facile leggibilità e portabilità. Un'interfaccia GUI è stata implementata per l'analisi dei dati e per il controllo del sistema. La funzione della regione di interesse (ROI) è stata integrata nel firmware. Inoltre, in questo sistema di misura del tempo è stato implementato un modulo di controllo della temperatura dedicato. Tale modulo si basa su una pompa di calore attiva a stato solido Peltier che permette di controllare la temperatura del TMU in maniera molto efficiente. Il test mostra che questo sistema di misura del tempo è in grado di offrire 7 secondi di intervallo di scansione completa con una risoluzione di 13ps, 10ps di precisione e offre delle prestazioni di linearità differenziale (DNL) pari al 1,5 % del LSB.

Index

List of Figure

1 Introduction

1.1. Introduction to the Time Measurement

Time measurements occupies an important position in the world metrology. The unit of time interval, the second (s), can be measured with more resolution and less uncertainty than any other physical quantity.

There are two types of time measurement: time-of-day and time interval. The Time-of-day mainly concerns about the observation of time. The units including minute, hour, day and year are used to describe the measured time. This measurement describes how much time has passed since last synchronization point. A clock is used to measure the time precisely. It use some techniques to ensure that it is synchronized to the real time flowing, although the measurement always contains some error. If we use the time to label some events, such as a meeting or a train departing, these events are hoped to happen at certain time point. These labels are called time stamp.

Time interval is defined as the duration or elapsed time between two events. In some physical experiment these events are called START and STOP. The standard unit of time interval is the second (s), while in many applications a shorter time interval is involved such as milliseconds ($10^{-3}s$), microseconds (10^{-6}s), nanoseconds (10^{-6}s), and picoseconds ($10^{-12}s$). The time as well as its unit “second” is one of the seven basic physical quantities, which means their definitions don’t rely upon other quantities.

In history the “second” was defined based on the rotational speed of earth or as a fraction of the tropical year. Now as the realization of atomic timekeeping technology, the second is defined in a much more precise way. The current definition of the SI second is:

The duration of 9,192,631,770 periods of the radiation corresponding to the transition between two hyperfine levels of the ground state of the 133Cs atom [1].

Usually in high resolution timing measurement systems people are more interested in the time difference between two events. This time difference can be measured either between two correlated signals on two separate channels, or on two consecutive signals on the same channel. In common situation the time interval measurement is performed between two separated signals, START and STOP, which are defined as the beginning and the end of the interval, so as to simplify the timing system.

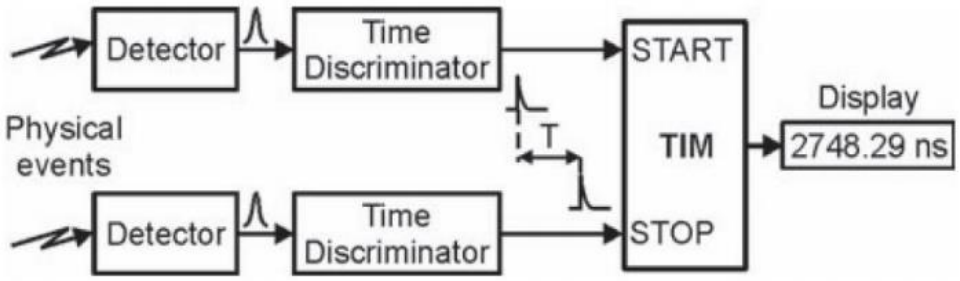


Figure 1.1 Working principle of the Time Interval measurement Meter

In the basic scenario illustrated in Figure 1.1, the time interval T is measured between the leading edges of two electrical pulses appear at the START and STOP channel of the time-interval meter TIM. These pulses usually are generated by a physical evens detector such as a laser scanner or photon detector. Then the pulses are processed by a time discriminators, which involves advanced methods and complicated electronic circuits to create the signal precisely time relative to the input pulse. The difficulty of determining the Time Interval between two pulses significantly arise as the demanding of accuracy and precision increase.

1.2. Applications

The requirement of precise time intervals measurement grows up rapidly during recent decades and its application covers a wide field from an instrument in aircraft to nuclear experiments. In this section, some well-known applications that based on time interval measurement is illustrated.

One of the most popular application is the Laser Ranging [2], where the TIM is used to measure the interval between the emission of a laser beam and the arrivals of its reflection. Figure 1.2 shows the working principle of this device. Since a light travels almost with a constant speed in air, the propagation time is proportional to the distance between laser source and obstacle. The distance can be measured indirectly by knowing the propagation time of the light. When a laser transmitter emits a series of light pulses, the target reflects them and the receiver senses the light reflection, which composites the START and STOP events for a TIM.

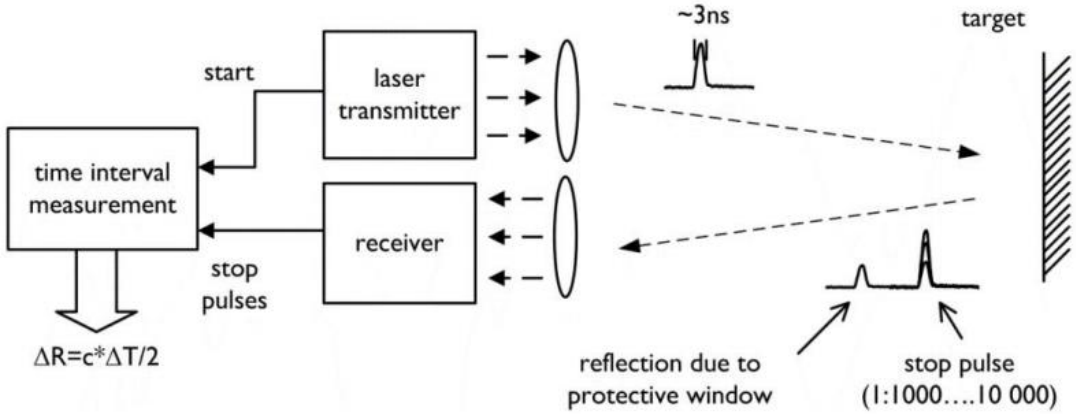


Figure 1.2 Working principle of the Laser Ranging

Due to fact that the speed of light is very high, which is $3 \times 10^8 \mathrm{m/s}$, a resolution of 1cm in distance requires a resolution of 66.67 ps in time. Therefor a high precision time interval measurement device is needed. Especially in some long range measurement application of laser ranging such as measuring the distant between the moon and earth, the full scan range of the TDC must at least several seconds while

the resolution must enough high to identify the possible small change of the distance.

Another famous application of time measurement is the time-resolved spectroscopy. In physics and physical chemistry, time-resolved spectroscopy is a way to study the dynamic processes in materials or chemical compounds by means of spectroscopic techniques. In particular in Fluorescence Spectroscopy a very short laser pulse is used to excite a system, and then another one is used to "read" the excited-state. With the help of pulsed lasers, it is possible to study processes that occur on time scales as short as 10^{-16} seconds. Figure 1.3 shows a Fluorescence Spectroscopy setup.

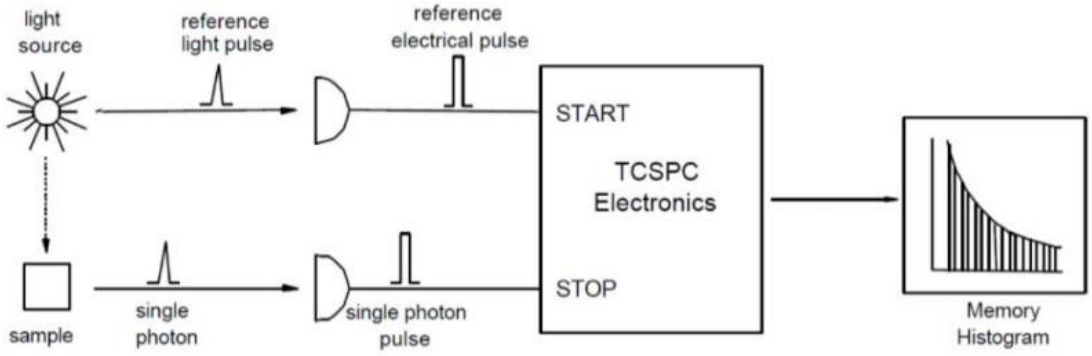


Figure 1.3 Block Diagram of a Fluorescence analysis by TCSPC technique

There are three commonly used technologies in time-resolved fluorescence spectroscopy: the stroboscopic technique (strobe), the time-correlated single photon counting technique (TCSPC) [3] and the frequency modulation or phase shift technique (phase). The first two are time domain technology while the last one is a frequency-domain.

The time-domain techniques, such as the strobe and TCSPC, are very close to each other in what they have measured and how the data is analyzed. The differences are mainly located in the hardware. Different detection electronics and pulsed light sources are used in two technology, although some light sources can be used in both technology. They are direct technologies which measure fluorescence decay curves

(i.e. fluorescence intensity as a function of time) directly and during the process of the experiment the physical mechanism can be seen by the researcher. Frequently, examining raw decay data and a proper fitting function can be thus selected.

These techniques are suitable for measurement both in case of instruments with two correlated channels and with one channel that refers to an internal clock. In the latter condition, the laser beams are generated synchronously with an internal clock, and the time of arrival of the reflected or fluorescence photons is measured.

1.3. Time measurement methods

In the instruments used for the applications described in the previous section, the measurement of the time interval turns out to be the crucial stage which determine the resolution of the entire measurement system. Thus particular interest must be taken in understanding the different methodologies that have been developed so far.

The time measurement systems can be divided into two main categories: the Time to Digital Converter (TDC) and the Time to Amplitude Converter (TAC). The first approach might involve the use of simple digital counter driven by a clock that have a limit in the minimum time resolution, which appears to be limited by the period of the clock used. If events to be discriminated are located temporally less than 100 picoseconds between each other, the clock frequencies required would be exceeding 10 GHz, which impractical for an ordinary digital circuits. While the counter based TDC can provide a very high dynamic range by extending the bit number of counter. Since the resolution required in these applications can be less than 10 picoseconds, thus leading to develop systems that are not limited by the clock frequency. In the next part, one pure digital circuit based on delay line and the second instead converts the time interval into a value of analog voltage which is then digitized by means of an ADC.

1.3.1. Time to Amplitude Converter

A Time to Amplitude Converters (TAC) is an analog system that converts a measured time interval into the amplitude of a voltage. Then the voltage signal can be

processed by an ADC so as to convert it into digital number for further process. Its principle is based on the integration of a constant current by the means of charging a conversion capacitor. When a capacitor is charged by constant current, the voltage across the capacitor will increase proportionally to the time, which can be expressed as:

$$U = \frac{I}{C}\int_0^t dt = \frac{I}{C}\cdot t = k \cdot t$$

Where I is the charging current, C is the capacitor size. Both of them determines the slope of ramp k between time and output voltage.

Figure 1.4 illustrates the block diagram of a modern TAC circuit [4].

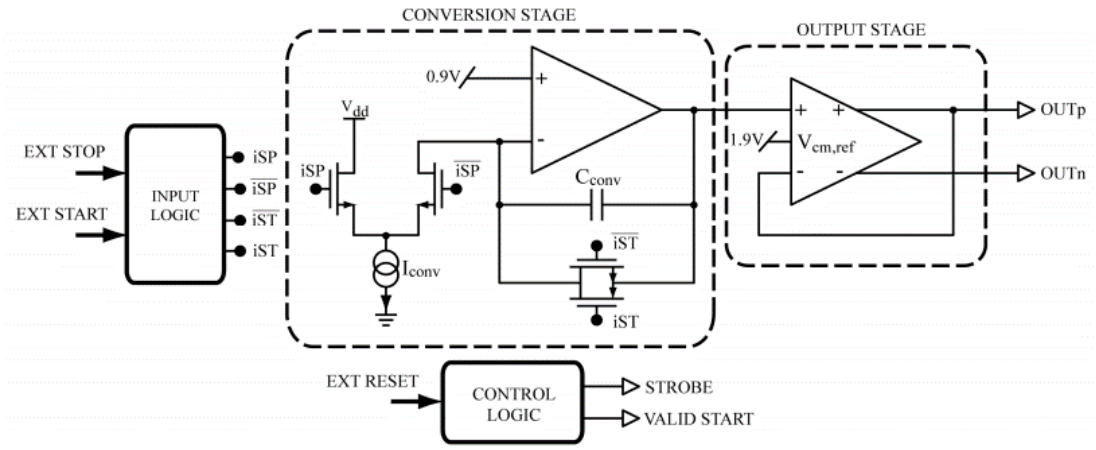


Figure 1.4 Block Diagram of a Modern TAC

Once a START event is detected, the conversion capacitor would be charged by a constant current until a STOP event arrive. In next stage the voltage across the capacitor is buffered and sampled by an external ADC if the data has to be processed by a digital system. When the convention has been finished, the conversion capacitor is discharged and the TAC is reset for next measurement. The current generator and the conversion capacitor constitute the core of the TAC.

The error of a TAC is mainly composed by two types:

- Random: Due to the electronic noise in the circuit, different results is obtained with the same time interval to be measured, which is also called jitter.
- Determined: Mismatch between components and the non-linearity of the converter cause systematic deviations of the actual conversion characteristic from the ideal. The error can be resolved by calibration of the system.

Since a TAC is an analog device, it is very sensitive to noise and environment, which will cause time jitter and offset in measurement. Meanwhile the external ADC also reduces the performance by creating nonlinearity and additional jitter.

The FSR (Full Scan Range) of the TAC can be changed by tuning the slope of ramp, which depends on the charging current and capacitor size as in the equation shown before. The time resolution of a TAC depends on the full scan range [5], which is determined by the slope of ramp k, if the resolution of an external ADC is fixed. The time resolution can be expressed as:

$$t_{LSB} = \frac{FSR}{2^n} = \frac{V_{MAX}}{2^n \cdot k}$$

Where n represents the bit number of the ADC, V_{MAX} is the maximum output voltage. On the other hand, the maximum measurable time interval is:

$$t_{MAX} = \frac{V_{MAX}}{k}$$

There is always a tradeoff between maximum measurable time and resolution. A high time resolution always accompanies with a short full scan range. If we want a 1s and 100ps resolution. A 33bit ADC has to be used, which will make the entire system extremely complex and not feasible.

1.3.2. Time to Digital Converter

The Time to Digital Converter (TDC) converts the measurement of the time interval directly to a digital value, which comes from the method that using delay lines to propagate a sampled signal from memory elements such as flip-flops [6]. One

possible implementation of this architecture is based on the Vernier Delay Line (VDL) [7] and an asynchronous output circuit which consists of two lines formed by the buffer having a different delay, depending on the chain of belonging. In particular, the buffer of the START line are designed to having a longer delay than those of the stop line. The typical structure is illustrated in the figure below:

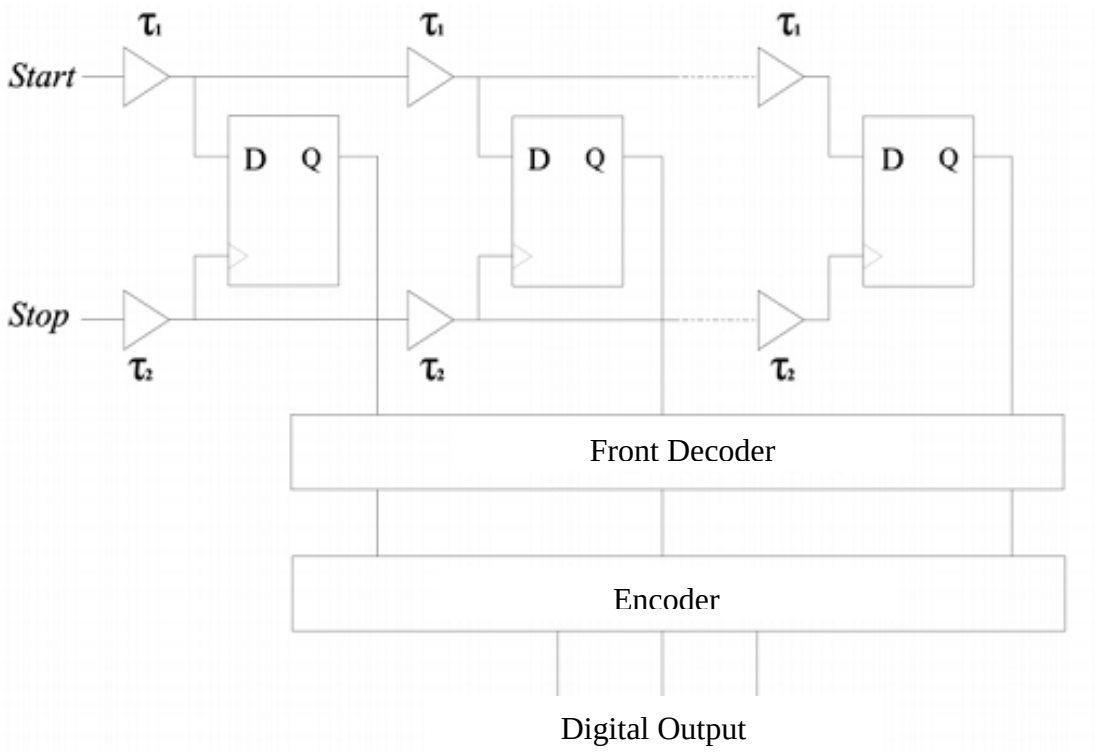


Figure 1.5 Block Diagram of a TDC based on Vernier Delay Line

The principle of measuring the time interval T between the START and STOP signal is: if from one side the two signals propagate along the chain, the time difference between the two pulses decreases at each stage of a constant amount equal to the difference between the delays of the buffers of the two lines. The stage of the delay line in which the STOP pulse reaches the Start provides the information on the value of the time interval to be measured with a resolution equal to the difference between the delays of the buffers of the two chains.

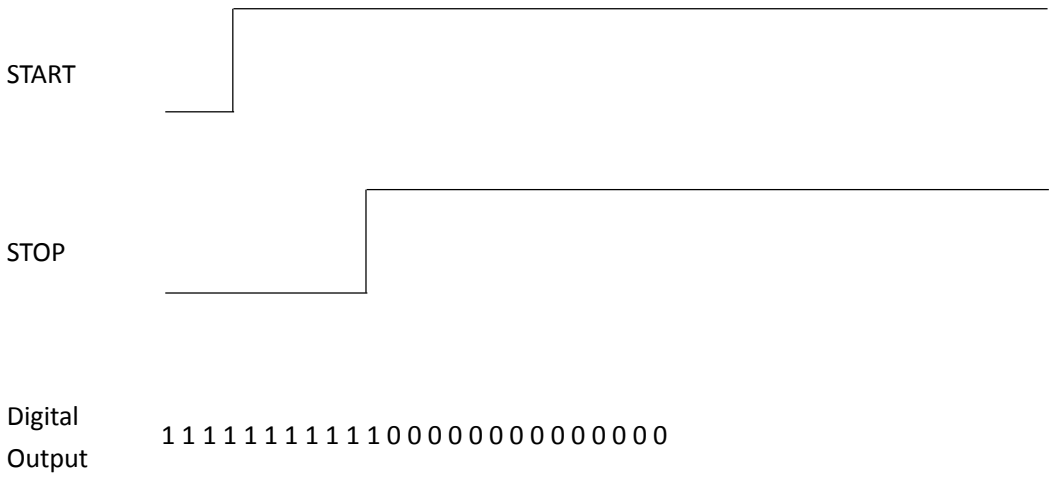


Figure 1.6 The relationship between the delay and output

The measured delay is then encoded where there is a transition of the digital output that changes from High (1) to Low (0). The output of the Flip-Flop Q at this point is processed by a detector and finally encoded by an encoder.

There is also the possibility of using more delay lines out of phase with each other, thus being able to obtain even higher resolution by means of interpolation techniques. For example you can put in parallel two of the TDCs and make the signal arrives at the first flip-flop of the second line with T_{delay} respect to the first delay line such that:

$$T_{delay} = \frac{\tau_d}{2}$$

If the circuit is modified properly, it is possible to obtain a resolution equals to T_{delay} (Figure 1.7)

However, some are detectable limits of these architectures, the first of which is due to the mismatch inevitably present between the delays of the individual blocks of the lines, which prevent the system to achieve high performance in terms of linearity of the conversion.

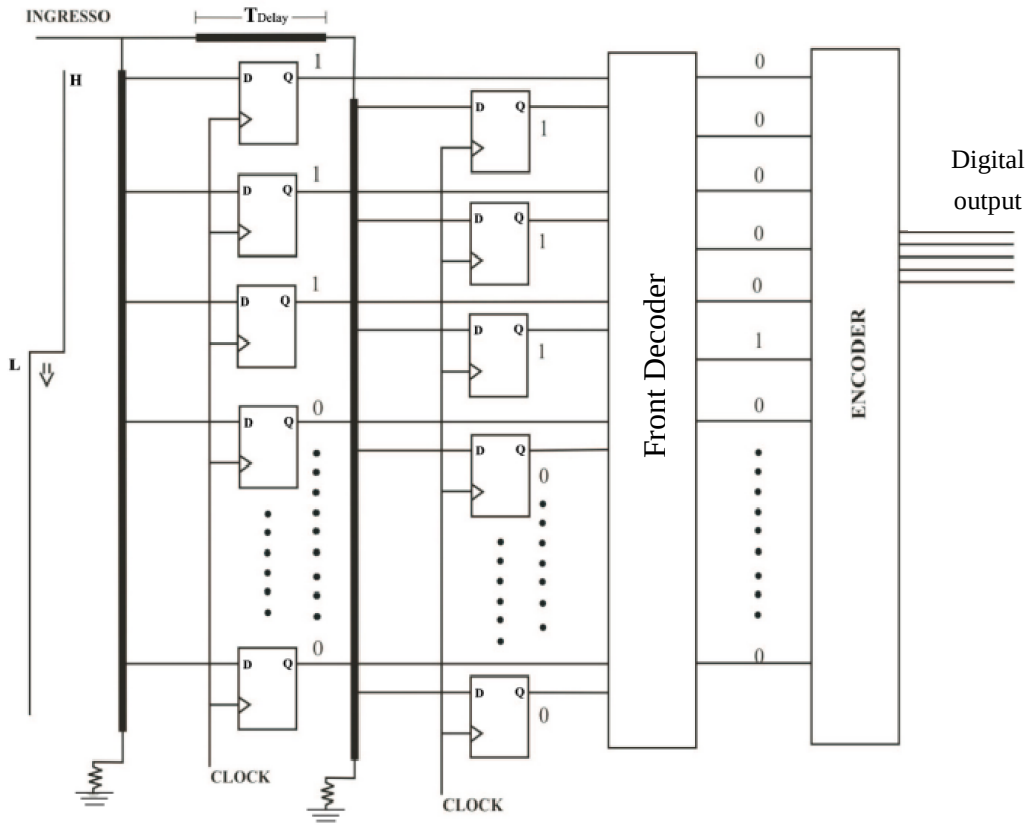


Figure 1.7 Interpolation techniques for higher resolution

Meanwhile the Full Scan Range of this type of TDC is limited by the number of Flip-Flop in series, or limited by the die-area in another words. Compared to the most advanced 4th generation of Intel Haswell processor, which only has integrated 17 billion transistors, in order to obtain a resolution of 100 ps and a FSR of 1s by using this architecture, at least 10 billion of Flip-flop in series is needed just for building the delay line and each Flip-flop has to be built with multiple transistors, which will increase the total number of transistors to more than 100 billion. The requirement of Flip-flop's number will increase lineally respect to the desired resolution and FSR. If a high resolution and high dynamic range is needed. Obviously we can't use this architecture in high dynamic range application.

2. System Overview

When a complicated system to be developed, the first step is determining the most appropriate fundamental techniques and evaluate its advantages, disadvantages, performance, possible complexity and feasibility based on the application requirements among a wide range of candidates. Once the core techniques has been decided, critical components that determines the performance and functionalities of the system are added to meet the performance and application requirements. Then some necessary supplementary components are involved so as to achieve a higher stability or ease to use. The architecture of the time measurement system is designed in this procedure.

2.1. Introduction to TMU

A counter based TDC can provide a very high dynamic range by easily extending the bit number of counter but its resolution is limited by the clock frequency. Meanwhile the delay line based TDC can achieve a very high resolution while its dynamic range is relatively low due to the limitations of hardware resources. If these two methods can be combined, which use the counter to form the lower resolution part and delay line to provide high resolution part of the final result. Both high dynamics and high resolution can be achieved. This is the design ideas of the TMU.

THS788, which is released by the Texas Instruments, is a four-channel timing measurement unit (TMU) aiming for fast and accurate time measurements. This TMU

has 13 ps resolution (LSB) and it is able to provide 8 ps of single-shot accuracy. While it has a FSR as large as 7 seconds, which is much larger than most of the existing time measurement technology can provide at this level of resolution. Respect to developing a brand new time measurement technology, after evaluating THS788's performance and ease of use, it is a good start point of developing a new generation of time measurement system.

2.1.1. Working principle of THS788

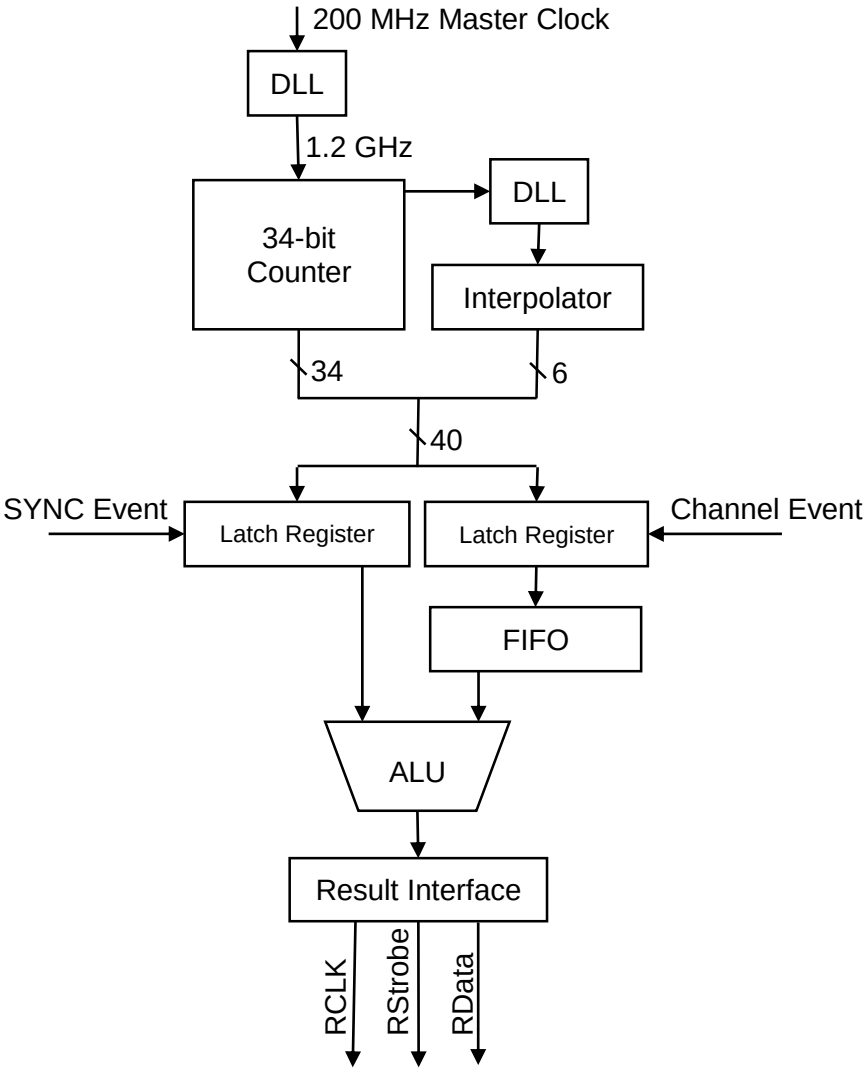


Figure 2.1 Block diagram of THS788

The figure 2.1 illustrates the working principle of the THS788. The core of the THS788 is a master synchronous counter which counts continuously at a rate of 1.2 GHz. This is the master timing generator for the whole TMU and defines the basic timing interval of 833 ps, which is further subdivided with Interpolator circuitry. The Interpolator subdivides the 1.2 GHz clock by 64 times and achieve a resolution of 13 ps. When a SYNC event, or START event, is detected, a sample of both counter and

Interpolator value is latched. Then if there is a STOP event on any channel, another sample will be latched into dedicated channel's FIFO as the timestamp of the STOP event. Their time interval is calculated by subtracting the counter value of START event and STOP event. Finally the result is sent to serial output interface.

In order to have a better understanding of the principle of counter based TDC, we place the output of a 3-bit counter in a circle (Figure 2.2) as and the time interval is simply the "distance" between two numbers on the circle. This distance can be calculated by subtracting two number directly.

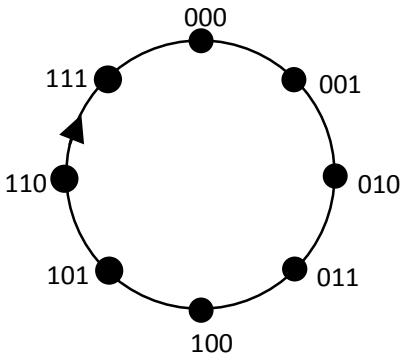


Figure 2.2 Output of a 3-bit counter

For example, if the time stamp of SYNC event is 111b and it of STOP event is 001b. Their subtraction is:

$$\begin{array}{r} 1001 \\ -\ \ 111 \\ \hline 010 \end{array}$$

Which is exactly the distance between two points. Another point where we have to pay attention is that, for example, if the time stamp of SYNC event is 000b and it of STOP event is 111b. Their subtraction is:

$$\begin{array}{r} 0111 \\ -\ 000 \\ \hline 111 \end{array}$$

You can both consider the number 111 as 7 or -1 as a complement number, or in other words, whether the START event happens before or after the STOP event. An ALU can only give a number but the chronological relationship between two events is hidden. Due to the fact that the THS788 performs subtraction operation just before the timestamp in a FIFO being transmitted but not when the timestamp being captured. As the transmission latency, if a new START event happens when a timestamp is waiting for being transmitted, the ALU will use the new timestamp of START event. We have to consider this result as a negative number since the STOP event happened before the new START event. In order to tolerant this situation the FSR of THS788 is defined as -7s to 7s but not 0 to 14s. As a compromise, the vast majority of the negative range in FSR is wasted since the transmission latency is usually very small. But if we consider that since the used negative range is small, it is possible to estimate the chronological relationship between two events in a more wise way and move the FSR to a more positive range. This operation would extend the effective FSR of TMU.

2.1.2. Delay-locked loop (DLL)

In the architecture of THS788 the delay-lock loop (DLL) consists a series connection of 12 identical and closely matched variable time-delay circuits. A single control voltage connects to each of the delay elements. The master 200-MHz clock connects to the input of the DLL. Because the period of 200 MHz is 5 ns, if the control voltage is adjusted to make the time delay of the DLL equal to 5 ns, the input and the output of the delay line is exactly phase matched. A phase detector connected to the input and the output of the delay line can sense this condition accurately, and a feedback loop with a low-offset-error amplifier is included in the clock multiplier to achieve this result. These are the second and third circuit blocks. With 12 equally spaced 200-MHz clock phases, select out six equally spaced 833-ps-wide pulses with AND gates and combine these pulses into a single 1200-MHz clock waveform with a

six-input OR gate.

The main difference of a DLL and a PLL (Phase-Locked loop) is how the phase delay is performed. A PLL use a VCO (Voltage Controlled Oscillator) to adjust the output frequency, therefore the phase is adjusted since a change in frequency will result an integral change of phase. A DLL use a VCDL (Voltage Controlled Delay Line) to achieve the same effect. In a DLL the phase of its last output is compared with the input clock then an error signal is generated and integrated to control the VCDL. The integrated error signal will increase until the phase difference comes to zero and a phase locked condition is achieved. Thus respect to PLL, a DLL is much easier to implement. Since there isn't signal generation part inside a DLL so the jitter is only influenced by the input signal. This property is very important in a time-sensitive application.

2.1.3. The Interpolator

The THS788 use an interpolator to achieve a 13ps LSB. The interpolator uses DLL techniques to subdivide the counter interval of 833 ps into 64 time intervals of 13 ps each. A large array of fast latches triggered by the hit latch captures the state of the 64 time intervals and logically determines 6 bits of timing data based on where the event occurred in the 833-ps clock interval. These 6 bits are latched and eventually passed to the FIFO, where they become the LSBs of the time-to-data conversion.

2.2. System layout

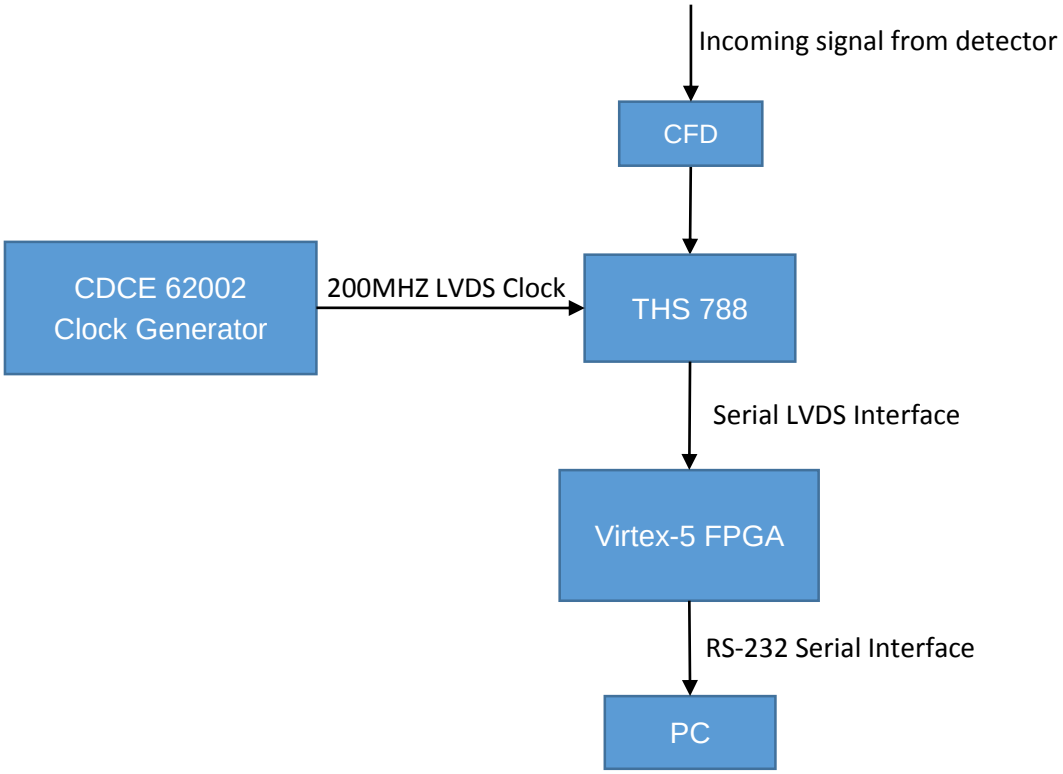


Figure 2.3 Block diagram of the system operation

2.2.1. System Architecture

The major part of the time measurement system's hardware consists of the TMU, clock generator, a Vitex-5 FPGA and CFD, whose block diagram is shown in figure 2.3. These components determines the performance and functionalities of the time measurement system directly. Although the TMU is a highly integrated circuit which contains a complete solution for time interval measurement, the time measurement system won't work with only TMU itself. Some necessary component in order to make the system operational must be included.

First of all is the selection of the clock generator, which is used to supply the 200MHz master clock for TMU. Due to the architecture of the THS788, which use the DLL technology, the jitter or skew of the clock source would have a direct influence in the final result. The clock generator has to be chosen carefully. In the datasheet of THS788 the jitter of master clock must be no more than 3 ps so as to maintain the specified performance of the TMU. The jitter and skew specification is the top priority when choosing the clock generator. In addition since the master clock is supplied in the form of LVDS signal, the LVDS protocol support is also an important specification to avoid unnecessary signal standard convention.

From many clock generators produced by various companies, the CDCE62002 produced by Texas Instruments is finally selected. The CDCE62002 is a high performance clock generator featuring low output jitter, which achieves jitter performance under 0.5 ps RMS specifically tailored for clocking data converters and high-speed digital signals. It has the capability of providing different combinations of output formats including LVPECL, LVDS and LVCMOS. Moreover it supports a high degree of configurability via a SPI interface, and programmable start up modes determined by on-chip EEPROM.

In the very start the analog front-end is designed as simple as a comparator, also called leading edge discriminator, which is triggered when the incoming signal transcends the threshold. But in practice this simple architecture can cause a serious time walk due to the fact that the amplitude of output signal of some detectors used in high energy physics and medical imaging is not fixed [8]. In leading edge discriminators signals of different amplitudes will cross the threshold at different times, thereby generating time walks [9]. This phenomenon is shown in figure 2.4.

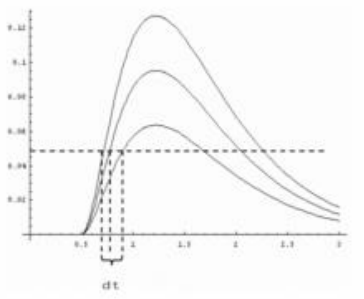


Figure 2.4 The time walk caused by amplitude variation

Usually this problem is solved by the Constant Fractional Discriminator (CFD) which is decided to be as the analog front end so as to achieve a better performance in amplitude-variation condition.

In order to have a complete solution for time measurement task, some programmable device must be added to the system to create some useful functions such as histogram generation, which is a powerful tool to analysis the distribution of the incoming signal. When determining which types of programmable device to be added, mainly between a micro-controller and a FPGA, the main consideration was the result interface of the TMU which is a non-standardized serial protocol and incompatible with most of the hardware peripheral of commercial micro-controller. Moreover the serial interface runs up to 300MHz, which is almost impossible or with high cost for a micro-controller to decode it in a software way. By considering these limitations, finally a FPGA is chosen to be involved in the time measurement system.

There are several key specifications when choosing the FPGA for this application. First, the register inside the FPGA must have the capability of running at 300MHz, which is frequency of the TMU serial interface. Second, the FPGA should have integrated LVDS receivers that can handle the LVDS signal from the TMU. Third, the FPGA should contain an enough amounts of storage elements for function implementations. Based on these requirements, the Xilinx FPGA architectures (Virtex-5 family) is chosen. All Virtex-5 family FPGAs have configurable high performance SelectIO drivers and receivers, supporting a wide variety of standard interfaces such as LVDS, LVCMOS and TTL. The robust feature set includes programmable control of output strength and slew rate, and on-chip termination using Digitally Controlled

Impedance (DCI). The DCI features an internal termination which is traditionally realized by a resistors to make the output and/or input impedance of receiver and driver matching to the impedance of the trace. In the case of controlled impedance drivers, DCI controls the driver impedance to match two reference resistors, or optionally, to match half the value of these reference resistors. DCI eliminates the need for external series termination resistors.

Apart from the major components, some supplementary parts is also necessary for the proper operation and system stability. Among them the most important part is the onboard device initializations, such as the TMU and clock generator, which is realized through their relatively low speed communication interface. Meanwhile a temperature control module is included in the system to minimize the temperature related performance influence. Considering these aspects, a micro-controller (MCU) is the best choice of handling these jobs. Respect to FPGA, a MCU provides a more flexible and simple way to implement the peripheral communication and controller where speed is not the critical issue.

The choosing of MCU is a tough task since there is great number of types of MCU. Their operation speed varies from some MHz up to several GHz while each of them includes different peripherals depends on the application they are designed. If we look back to our system architecture, it might be found that almost all types of signal including low/high speed, single ended/differential and even high current circuit appear. It is a very difficult task to arrange all these signal without any influence between them on a 4 layer PCB. An ordinary MCU usually offers little flexibility in its architecture when it is designed. This means that you have to find the corresponding pin of each peripheral in the datasheet and connect them to target device without any possibility of selection. In the PCB design phase, it is a common scene that in the same layer one trace has to cross another one in order to connect them to the target pins. Usually this is resolved by two vias which use the space in another layer. But in some case another side locates some critical high speed signal and it is hard to found a way without influence. In order to reduce the routing complicity as much as possible, flexibility is the top priority when choosing the MCU.

In the end the PSoC 3 (Programmable System-on-Chip) is chosen, which provides

microcontroller unit (MCU), memory, analog, and digital peripheral functions in a single chip. It gives a maximum flexibility when designing a mixed signal processing system. Thanks to its digital system interconnect (DSI) we can assign its peripheral to any pin of it, which makes routing much easier and less possibility of having crosstalk with other signal. It also provides functional flexibility through an array of small, fast, low-power UDBs. Although its 8051 CPU is not so powerful when compared to some new generations of MCU such as the STM32F4 series, it perfectly meets our requirement especially in the aspect of flexibility.

Thermal control is always a critical part of sampling system since heat can cause many undesired effect such as bias, drift inside the circuit and make the result less accurate and precise. The THS788 is a high energy consuming device that require 1.236A of current when running in maximum speed and resolution mode, which generate a lot of heat. In order to overcome this problem, the THS788 has an embedded temperature sensor locate in the core of the chip. It measures the core temperature directly and outputs an analog voltage proportional to the temperature. This sensor makes direct controlling of core temperature possible.

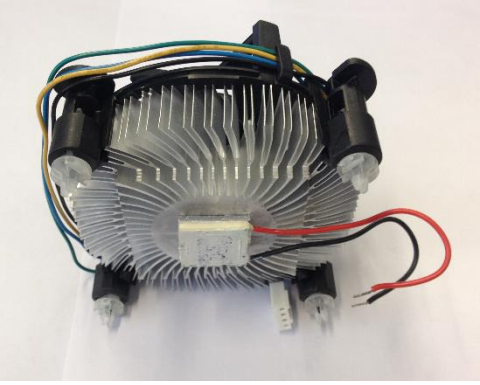


Figure 2.5 The thermal control module

Traditional heat dissipation method which use a metal radiator and electric fan. Although the temperature can be controlled by adjusting the speed of the fan, its specification of heat dissipation efficiency, response speed, and control range are unsatisfactory. In order to stabilized the system temperature over a wide range of

environment temperature, especially when the environment temperature is higher than set temperature, a peltier, which is a solid-state active heat pump and transfers heat from one side of the device to the other, is used. The figure 2.5 shows the configuration of the thermal control module. The cold side of the peltier is attached to the TMU and hot side is mounted on a cooling fan. The power of the peltier controls heat transferring rate between two sides. Since the heat sink rate of a cooling fan is proportional to the temperature difference of itself and ambient, when current is high, more heat is transferred from cold side to hot side, which creates a big temperature difference and enhance the heat sink rate of the cooling fan. The temperature regulation is realized by controlling the power of peltier through an H-bridge which controls the duty cycle of the peltier's current thus control the heat sinking rate of the cooling system. This method offers a very wide range of temperature set-point possibility as well as a very fast and stable temperature response respect to traditional cooling methods.

2.2.2. System Operation

The START and STOP signal are firstly processed by the CFD, which reduce the influence of amplitude variation. Then the signal is converted into differential LVDS signal and send to the corresponding port of TMU. The TMU generates the timestamp result based on the time difference between START and STOP signals. All results are transmitted through its high speed LVDS serial interface to Virtex-5 FPGA. User can manage the experiment through the serial interface by scripts or graphic user interface. When a timestamp data is received, it will be first processed by the ROI module to eliminate all uninterested samples. Then the sample will be passed to certain function blocks in the firmware of FPGA based on the experiment type selected. Once executions of the function blocks is terminated, it will upload the result through the serial interface to PC.

2.3. Region of Interest (ROI) function

In this architecture of time measurement system, a ROI (Region of Interest) function concerns of two aspects. The first works inside the FPGA to eliminate the

undesired samples during signal acquiring phase. In some applications the arrival time the sample can be approximately estimated in a defined range thus the ROI function can be used to eliminate any data out of this range if we only interested in the signal distribution in this range.

The second ROI function works in the GUI interface. It can significantly reduce the memory usage for TMU data display storage. Since both the resolution and dynamic range of the time measurement system are very high, when a lager time range is chosen, for example, the maximum range -7s to 7s, if there is no ROI function, it is necessary to use an array with a length of 2^{40} to store which bins have occurred while other are not. Even the smallest possible element size is chosen, it also needs $2^{40}\ \text{bytes} = 1024\text{GB}$ of memory which exceeds the memory of modern computer architecture can provide. Meanwhile since the resolution of a display screen is limited, it senseless to display in the highest resolution. The realization of ROI function is like a quantizer, the selected time range is evenly divided into a determined number of small slots whose upper bound, lower bound and step determined. The number of slots can be select for the most appropriate display effect. When a TMU data is received, it will be inserted into the proper slot where the bound constraints is meet. By using this techniques, the memory usage is significantly reduced while the level of display effect is kept.

3. Hardware Design

This chapter describes in detail the hardware design of each components in the time measurement system. The first section mainly describes the design notes of the PCB such as power supply system and signal Integrity. In the second section, the implementation of a Constant Fractional Discriminator (CFD) is discussed in detail and a new method of coefficients optimization based on numerical computation is proposed.

3.1. PCB Design

In the layout of PCB there are five main parts:

- Power supply
- Time Measurement Unit (TMU)
- Analog front-end
- Clock generator
- Peltier driver

The layout of the PCB is shown in figure 3.1 and 3.2.

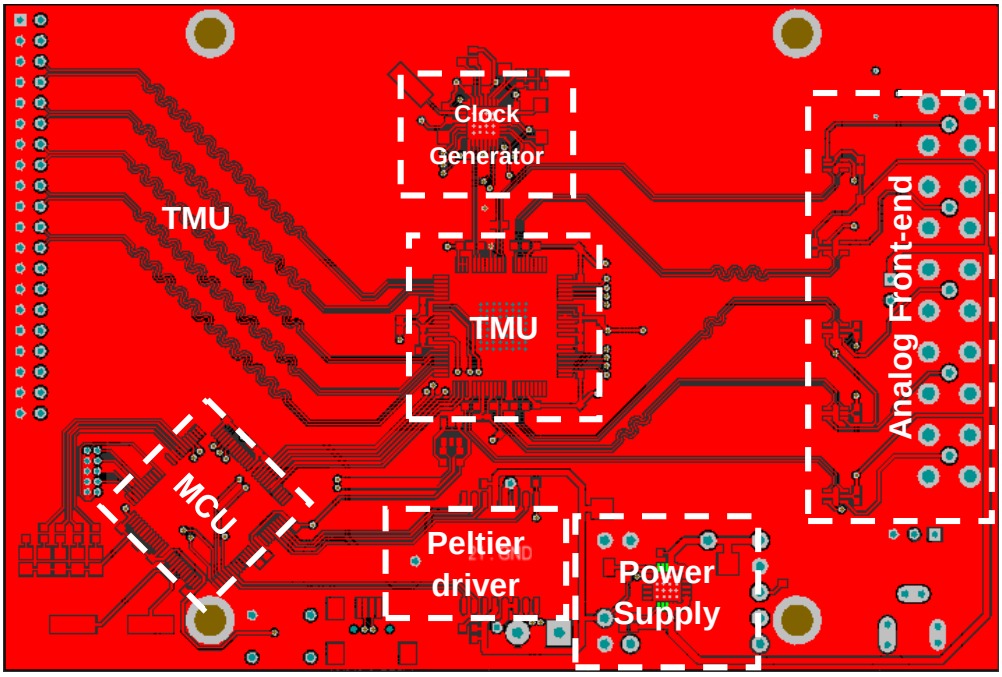


Figure 3.1 The top layer of the PCB

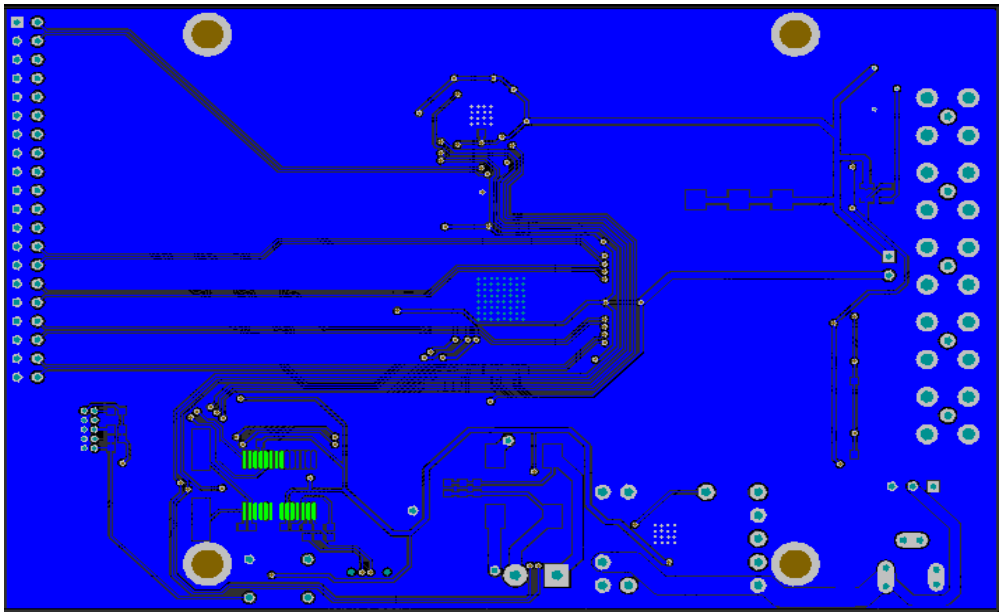


Figure 3.2 The bottom layer of the PCB

When designing this board, the most important designing objective is the noise reduction. The time measurement unit and analog front-end are noise-sensitive circuits whose performance will significant drops while the noise in electronic circuit increases. This PCB is a mixed signal system that includes almost all varieties of signal such as low/high speed signal, single end/differential signal and even high current circuit. Various methods and design rules are used to reduce the noise as much as possible.

The PCB consists of four layers, which are assigned as high speed signal layer, ground layer, power supply layer and mixed signal layer from top to bottom. The high speed signal layer contains all the high speed device including the clock generator, TMU and analog front-end. They are interconnected by Low-voltage differential signaling (LVDS). All low frequency signal, power regulator, and peltier driver are allocated to mixed signal layer. Between them are the ground plane and power plane. A complete ground plane and power supply plane are necessary when the noise has to be reduced. In a digital system there are several noise sources such as crosstalk between adjacent circuit traces and power supply coupling. When a digital circuit changes state, active devices such as transistors would generate a large current pulses flow through the ground circuit. If the power supply and ground traces have significant impedance, the voltage drop across them may create noise voltage pulses that disturb other parts of the circuit, which is also called ground bounce. The ground plane and power plane can greatly reduce the impedance as well as the ground bounce effect by reducing the impedance. Meanwhile a complete ground plane ensures all high speed signal have the shortest return path thus the inductive crosstalk is minimized. It also has a significant effect in reducing the crosstalk by separating the high speed signal layer and mixed signal layer to make them not influence each other.

Figure 3.3 Assignment of the PCB

3.1.1. Power supply

A low noise power supply scheme is the base of high performance time measurement system. The input voltage is 12V, which comes from an AC-DC adapter. This voltage is the rated value of the cooling fan and peltier. These two devices are powered directly by the AC-DC adapter since their demand of power quality is not significant. All other device such as the TMU, clock generator, and MCU require a 3.3V low noise power supply. A linear regulator should be used to obtain this objective. But due to the working principle of linear regulator, it is not adaptable to use a single regulator to transform 12V down to 3.3V due to the operation principle of linear regulator (Figure 3.4).

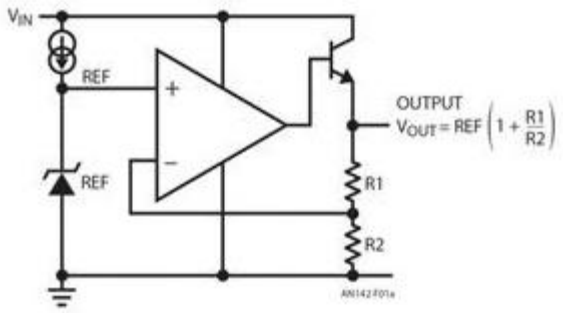


Figure 3.4 The operation principle of linear regulator

The linear regulator is formed by a negative feedback amplifier. The amplifier will try to adjust its output voltage which is the base voltage of a transistor thus changes the output current so as to keep the feedback voltage the same as the reference. It act like a variable resistor, continuously adjusting a voltage divider network to maintain a constant output voltage, and continually dissipating the difference between the input and regulated voltages as waste heat. For example, in this case when the output current is 1A, the regulator has to dissipate more than $(12 - 3.3) \times 1 = 8.7\text{W}$ of heat power. The regulator would overheat immediately and cause unpredictable damage to on board devices.

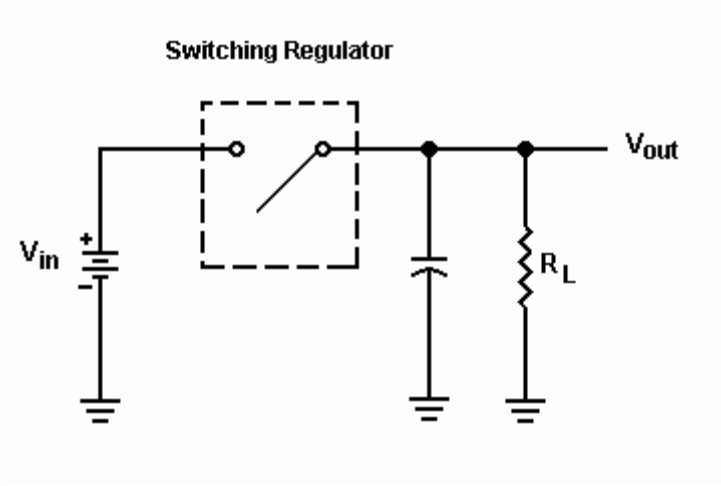


Figure 3.5 The operation principle of switching regulator

Although a switching regulator can provide a much higher power transform efficiency, its noise property is not acceptable because in order to regulate the output voltage a switching regulator needs to fast open and close its switch to change the duty cycle of the output voltage, which creates unavoidable noise in output. So the best way is the combination of these two solutions. This cascade configuration enrolls two regulators, a switching regulator and a linear regulator. The switching regulator works at the first stage, which transforms the input voltage down to 4V with high efficiency and low heat dissipation. The next stage is a linear regulator. It regulate the noisy 4V voltage into a clean 3.3V voltage and distribute it to the on board device. Since this time the voltage difference is only 0.7V, it generates little heat. With this cascade configuration, both transform efficiency and power quality are achieved.

Decoupling is also an important part in power supply scheme. The clock generator and TMU are high frequency switching devices, which generate a lot of high frequency noise to other part of circuits. Decoupling capacitor is assigned to each power pin of all devices so as to solve this problem. All decoupling capacitors are placed as close as possible to pins. When the board area is allowed, multiple decoupling capacitors are put in parallel to achieve a better decoupling effect by reducing the ESR (Equivalent Series Resistance) and ESL (Equivalent Series Inductance).

3.1.2. Signal Integrity

Signal Integrity has a significant influence in the performance of whole system. As signal frequency increases, we can no longer consider a circuit as a lumped circuit but a distributed circuit. This means that the mission is not as simple as connecting all device together. When in low frequency situation, only the resistance property of a generic conductor, for example a trace in the PCB, plays a major role. But in high frequency situation, the parasitic capacitance and inductance come into effect. The distributed parameter model must to be used to estimate the correct impedance of the conductor and the impedance control as well as crosstalk effect have to be considered when routing.

Reflection

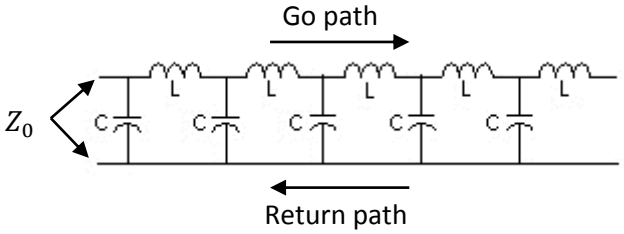


Figure 3.6 The transmission line model

If two wires are placed closely parallel to each other, one is the go path while the other is the return path, between them there is certain capacitance and inductance. These two values depend on many parameters such as the wire width, the clearance and the dielectric constant of the material between these two wires. When this pair of wire is applied with a high frequency signal, the voltage applied and the current in the wire define the characteristic impedance of this wire. Also we can use the formula $Z_0 = \sqrt{\frac{L}{C}}$ to calculate this characteristic impedance, where the L and R are defined as the inductance and capacitance per unit length of the so called transmission line.

In transmission line model, the signal is not transmitted to the destination immediately. The signal propagates through the go path to the destination and return

to the source with a delay. When a voltage generator is connected to a transmission line, it can only “see” the characteristic impedance of the transmission line because the signal hasn’t arrived at the destination. The voltage generator would generate a current in the transmission line depends on the characteristic impedance. Once the current has arrived at the destination and knows the termination resistor R_L immediately. When the current pass through R_L, there would be two situation:

- $R_L = Z_0$: In this situation there would be no reflection.
- $R_L \neq Z_0$: In this situation there would be a “reflection”. Some signal is bounced back to the source. The reflected voltage can be calculated as $\frac{R_L - Z_0}{R_L + Z_0}$.

There is an easy way to understand the reflection problem in the transmission line. Imagine that you are using a water pump to pour water in a water channel. The water wave would propagate through the channel at a certain speed which depends on the shape of the channel. In the end of the channel there is a hole to let water out. If the hole is too small, which means the leaking speed is smaller than the flow of water, the water would heap up and create a backward wave. Although these two process are not exactly the same, it is a good way to understand the reflection problem.

In the case of PCB, there are two types of transmission line: microstrip and stripline. The difference of these two types of transmission line is the number of return path. Microstrip has one only return path, which means this type of transmission line can only locate in the surface layer of a PCB. In the other hand, stripline has two return path, the upper layer and lower layer. The signal line is sandwiched between them. So this type of transmission line only exist in the internal layer of a PCB. Since in our case there are only 4 layer in this PCB, the microstrip is the only choice due to the assignment of layer.

Microstrip

Stripline

Crosstalk

The crosstalk is a phenomenon that a signal transmitted in a wire generates some undesired signal in other circuits. In time measurement system the crosstalk can create a problem that if the signal level changes in one channel, another channel would also be triggered due to the crosstalk effect thus cause some “fake” sample that pollute the results. The crosstalk has to be eliminated by all means.

There are two types of crosstalk: inductive (magnetic field coupling) and capacitive (electric field coupling). The principle of capacitive is, if two wires are placed close to each other, this structure is very close to a shunt capacitor. When a signal pass through one of them, it will generate a sudden change electric field and cause the electrons on the other side moves. Since the capacitance reduces with the distance. If these two wires are put enough far or isolated, the capacitive crosstalk would have little effect.

The inductive crosstalk share the same principle with mutual inductor. Every signal has a go path and return path. These two paths form a loop circuit. Once there is signal transmitted inside one loop. It will generate a magnetic field and couple to other signal loop. In order to reduce the inductive crosstalk, two methods have to be used. First, since the magnetic field intensity decreases with distance, the inductive crosstalk also decrease with distance. An enough distance between two wires is always necessary. Second, the loop area has to be reduced due to fact that electromotive force in victim generated by aggressor’s magnetic field is proportional to the loop area of victim. In the case of PCB routing, a common ground layer has to be used to reduce the signal loop area since in high frequency situation the return current always flow under the go path.

But as system integration level increase, the clearance between wires has to be decreased so that more wires can be put into the same area as before. The decreased clearance leads to a more serious crosstalk problem. This problem is overcome thanks to the differential signal technology. Unlike single ended signal, the differential signal use two wires to transmit one signal and the signal is defined as their difference. This means that if both of two wires suffer from one crosstalk source, there would be little influence since the common mode voltage is cancelled out during the differential process. Another advantage of differential signal is that it creates little crosstalk to other wire. The crosstalk always happens when there is level change of the signal. If

signals in two closed placed wire change in opposite direction with same amplitude, they create almost the same change of magnetic field and electric field but in opposite direction, which are cancelled out each other.

Although differential signal has many advantages, there are some critical rules have to be followed when routing in PCB so as to maximize its advantages. The distance ratio $\frac{d}{D}$ must be considered where d is the clearance between differential pair is and D is the distance between differential pair and another wires. Since both the electric field intensity and magnetic field intensity decreases with distance, if the distance ratio $\frac{d}{D}$ is high, when the differential pair is the victim, the aggressor would cause unbalanced crosstalk that can't be fully canceled out during differential process. On the other hand, when the differential pair is the aggressor, the crosstalk caused by the pair won't cancel each other in the victim. To sum up, the distance ratio $\frac{d}{D}$ must be kept as small as possible which means routing differential pair as close as possible and other wires as far as possible. The figure 3.7 shows how the high speed LVDS differential trace is routed in this PCB design to avoid the crosstalk effect. The distance between each pair are kept as large as possible.

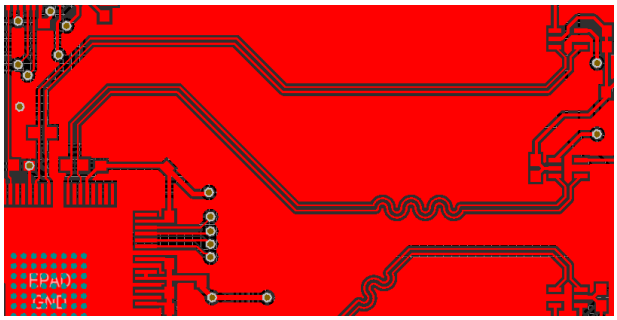


Figure 3.7 The routing of the LVDS trace in the PCB

3.1.3. LVDS

Low-voltage differential signaling, or LVDS, is a technical standard that specifies electrical characteristics of a differential, serial communication protocol. It supports a theoretical maximum signaling rate of 1.923 Gbps [10] while at the moment, commercial LVDS chipsets are specified for operation in the megabits-per-second range.

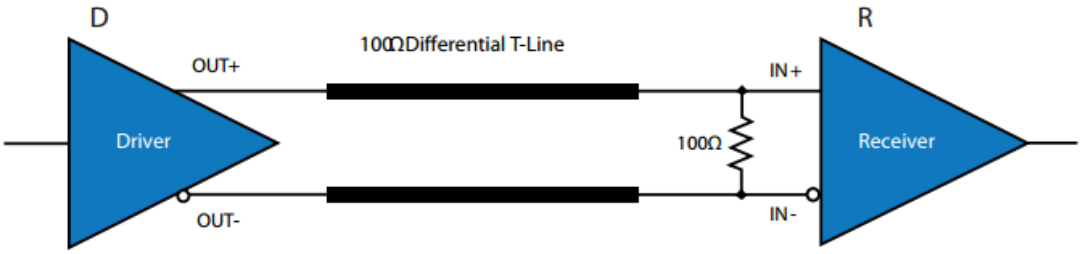


Figure 3.8 The standard LVDS signal structure

LVDS is a differential signaling system, which means that it transmits information as the difference voltages between pair of wires. The two wire voltages are compared at the receiver. In a typical implementation, a constant current of 3.5 mA is injected into the wires by the transmitter, with the direction of current determining the digital logic level. The current passes through a termination resistor of about 100 to 120 ohms, which should be the same as cable's characteristic impedance, at the receiving side,

and then returns in the opposite direction via the other wire. By using Ohm's law, the voltage difference across the resistor is therefore about 350 mV. The receiver senses the polarity of this voltage to determine the logic level.

Since the LVDS transmitter consumes a constant current, it places much less demand on the power supply decoupling and thus produces less interference in the power and ground lines of the transmitting circuit. This reduces or eliminates phenomena such as ground bounce which are typically seen in terminated single-ended transmission lines where high and low logic levels consume different currents, or in non-terminated transmission lines where a current appears abruptly during switching.

The low common-mode voltage of about 1.2V allows using LVDS with a wide range of integrated circuits with power supply voltages down to 2.5V or lower. The low differential voltage, about 350 mV, causes LVDS to consume very little power compared to other signaling technologies.

With these advantages, the TMU uses LVDS technology in its serial bus and time event input channel to have an enough fast signal transmission while the power dissipation is minimized. When routing the LVDS differential trace, the impedance as well as the termination resister must be calculated and chosen carefully so as to minimize the reflection. The termination resister must be put as close as possible to the TMU. The figure 3.9 shows an example of the placement of termination resister and LVDS trace.

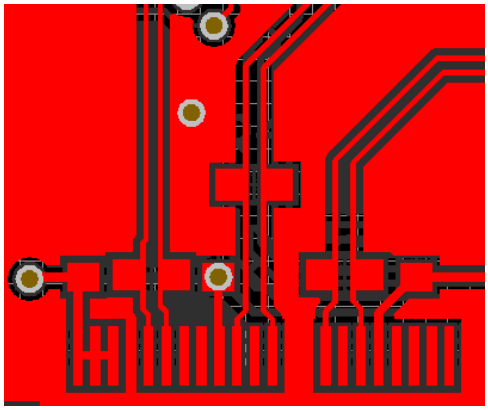


Figure 3.9 The allocation of onboard termination resister

3.1.4. Trace Length matching

When we turn on the switch of a lamp, it lights up immediately. This daily phenomenon make us thought that the electronic circuit usually works with no or little delay. The electrical signal propagates in a wire with the same order of light speed, which is hundred thousand kilometers per second. In low frequency signal situation, this delay is too small to be observed. But as the signal frequency becomes high, the signal delay in a wire or trace can no longer be ignored.

When designing a PCB, the difference in trace length will cause a difference in signal timing. In some application such as a high speed synchronous bus the circuit is very sensitive to the timing mismatch. As the signal frequency arise, the tolerance of timing error decreases significantly. For example, the synchronized serial interface of TMU runs at 300 MHz. The equivalent period is 3.33ns. 1 cm of length difference between data line and clock line would cause about 2% of its phase shift.

Also in time measurement circuit the length mismatch can lead to a constant offset in the result as the mismatched traces make detectors' signal arrive at TMU with a time offset. When routing these trace in this design, length of input signal trace for each channel must strictly matched with a tolerance less than 10 mil, which is about 1.5 ps of time difference. In Altium Designer, a famous PCB design tool, a feature called

“Interactive Length Tuning” can be used to generate redundancy trace with a preferred trace length. The result of length matching of the PCB is shown in Figure 3.10.

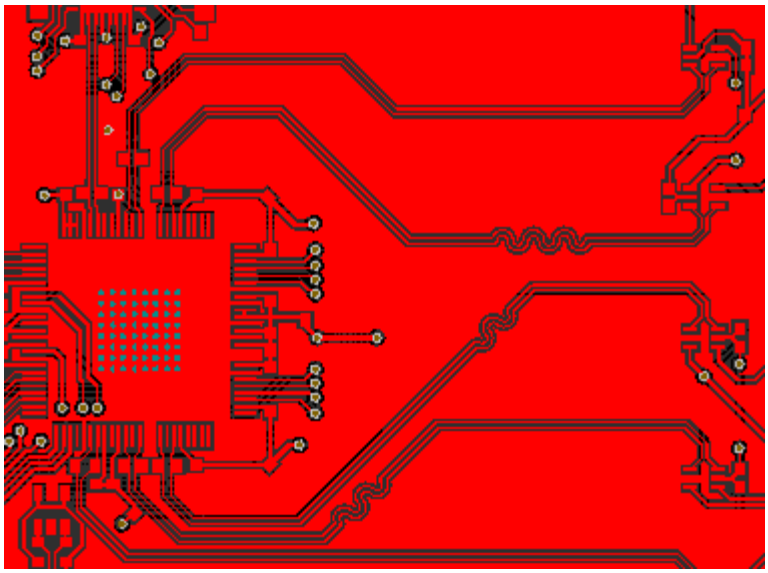


Figure 3.10 The result of length matching

And the length pairs is designed as close as possible to each other. Their length is shown in Figure 3.11.

EVENT_A	2182.62
EVENT_B	2183.099
EVENT_C	2183.228
EVENT_D	2183.236

Figure 3.11 The result of length matching

3.2. Constant Fractional Discriminator (CFD)

The accuracy in determining the time of arrival of a particle is a key issue in several modern applications such as time of flight techniques used both in high energy physics and medical imaging. But in practice the amplitude of output signal of these detectors is not fixed. The precision required involves the use of timing techniques to correct for amplitude variations of the signal and optimized to reduce noise effects. It is well known that in leading edge discriminators signals of different amplitudes will cross the threshold at different times, thereby generating time-walk. A Constant Fractional Discriminator (CFD) is an effective approach to minimize time-walk when the dynamic range is not too large so that the linearity of the system can be hold.

3.2.1. Working Principle and Implementation

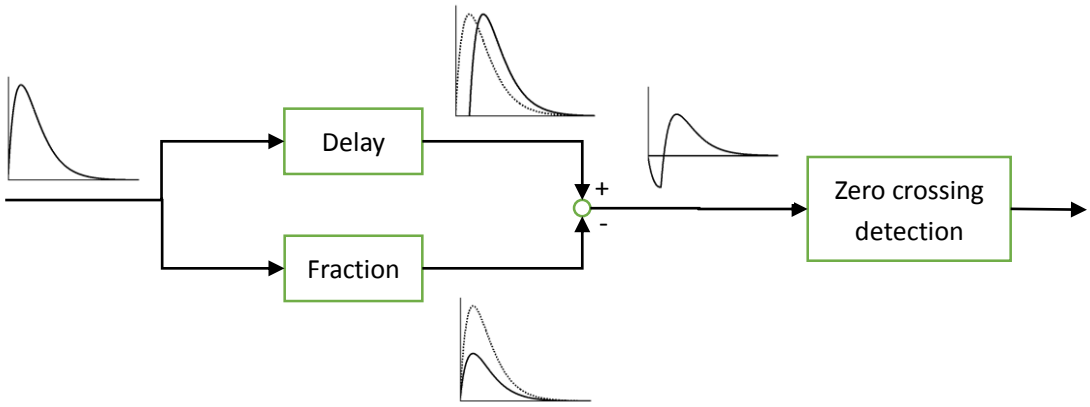


Figure 3.12 The working principle of CFD

The time walk problem comes from the leading edge discriminator trying to use a constant threshold to discriminate a variable amplitude signal. So if the threshold changes with the same ratio as signal amplitude, the trigger time will independent to the signal amplitude. The figure 3.12 illustrated the working principle of CFD. It detects the zero crossing of a bipolar waveform obtained by subtracting a fraction of the signal

(3.1)

to a delayed copy of it. The general transfer function of a CFD is given by eqn.3.1. Where f is the fraction coefficient.

$$V_{CFD} = y(t - t_d) - fy(t)$$

In a simple situation, it can be assumed that the input signal is a triangle wave, which can be expressed as:

$$y(t) = \frac{At}{T_p}$$

Thus the corresponding output of a CFD can be described as:

$$V_{CFD} = \frac{A(t - t_d)}{T_p} - f\frac{At}{T_p}$$

Since the trigger point is $V_{CFD} = 0$

$$\frac{A(t_z - t_d)}{T_p} = f\frac{At_z}{T_p}$$

$$t_z = \frac{t_d}{1 - f} \qquad (3.2)$$

So the trigger time is not affected by the amplitude of input signal as shown in equ.3.2.

However in practice detector signals are subject to statistical fluctuations that affect not only the amplitude, but also the signal shape. Usually the signal has this shape which can be described as equ.3.3. The trigger time is no longer independent to the signal shape. In other words, the performance depends greatly on the signal shape.

$$\mathrm{y}(t) = \frac{Ate^{-\frac{t}{\tau}}}{\tau^2} \qquad (3.3)$$

$$\frac{A(t_z - t_d)e^{-\frac{t_z - t_d}{\tau}}}{\tau^2} = f\frac{At_z e^{-\frac{t_z}{\tau}}}{\tau^2}$$

$$t_z = \frac{t_d}{1 - fe^{-\frac{t_d}{\tau}}}$$

To minimize the sensitivity of the CFD to the rise time fluctuations of the input signal a delay shorter than the peaking time must be used [11].

Since minimum level of the input signal is defined as zero. When there is no signal, both input pins of the zero-cross detector have the same inputs. In reality the comparator will be triggered by the noise and output fake event. There are two methods to solve this problem. The first method is implementing an arming condition discriminator [11] (figure 3.13), which use an AND gate as a switch.

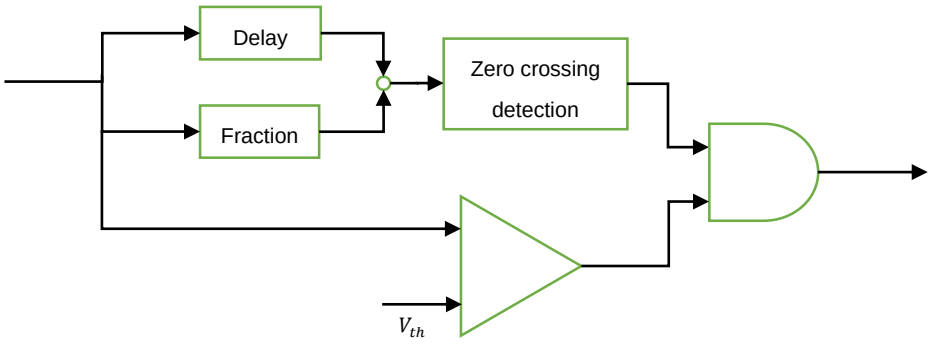


Figure 3.13 The CFD with arming condition discriminator

When there is no signal, the output of arming condition discriminator is low and as a result the output is disabled by the AND gate. Once the signal level has become higher than the threshold of the arming condition discriminator, one of the input of AND gate will be set by the discriminator priori to CFD since the CFD circuit has a larger delay than the arming condition discriminator circuit. In this situation the AND gate will be triggered as soon as the zero-cross detector is triggered with a constant propagation delay. This scheme is not applicable in our design because there is no commercial low jitter (less than 10 ps) AND gate available. This implementation usually seen in lower performance situation or ASIC application.

Another idea is changing the threshold of zero-cross detector from 0 to a little higher voltage with the cost of the time walk increasing. Since the trigger point of zero-cross detector is no longer at 0 but an offset voltage. The actual trigger time can be calculated as:

$$\frac{A(t-t_d)}{T_p} - f\frac{At}{T_p} = V_{offset}$$

$$t_z = \frac{t_d + V_{offset}\frac{T_p}{A}}{1-f}$$

The result shows the trigger time is no longer independent to the signal amplitude. The offset voltage owns a direct relationship with noise level so noise reduction is always the main topic in this designing task. In next section the optimization of reducing the time walk by tuning the t_d and f is discussed.

The trigger point of zero-cross detector can be changed in two ways. First way is applying an offset voltage to one of the two inputs of zero-cross detector by using the traditional AC coupling bias circuit. The other is utilizing the hysteresis feature of the zero-cross detector. The second method is chosen because it is more reliable and stable respect to the AC coupling bias circuit which is composed by some resisters and capacitors.

In the final design, which is shown in figure 3.14, the input signal is buffered by a wide-bandwidth buffer so as to make the input signal properly terminated and isolate the components in rear. The pulse attenuation is made by a voltage divider. Rather than using a very long trace, the delay part is realized by a six order RC low pass filter in series whose delay time can be changed more easily. The overall delay through the circuit can be estimated by nRC, where n is the number of cascaded stages. An ADCMP605 comparator from Analog Device is used as the zero-cross detector, which supports programmable hysteresis feature by changing a resister.

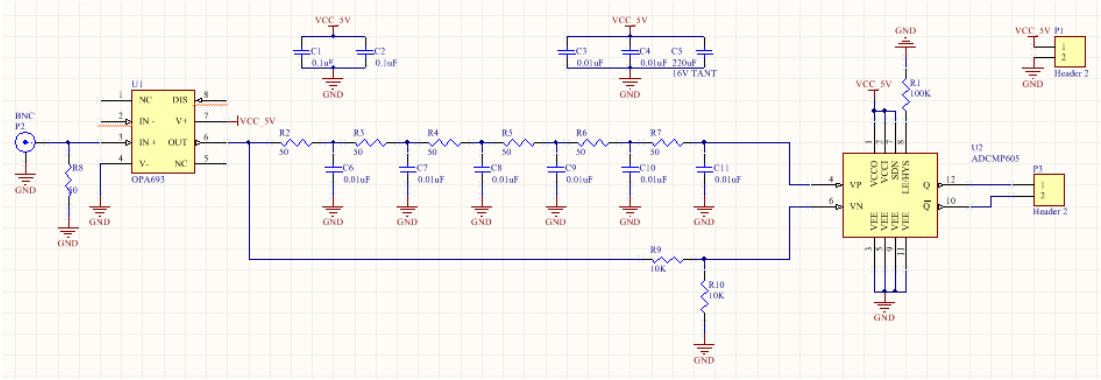


Figure 3.14 The schematics of the CFD

3.2.2. Performance Optimization

The performance of a CFD depends greatly on the matching of its coefficients with input signal, which means the coefficients have to be tuned based on the knowledge of input signal. Usually commercial CFD supports user tunable coefficients such as delay time t_d and fractional number f. But tuning of these coefficients depends heavily on one's experience and time costing. In previous studies [12], an analytical method is used to analysis the CFD and give out some qualitative conclusion. In this section, a new numerical method based on CFD simulation model is introduced to calculate the optimal coefficient set for arbitrary input signal shape.

The error of a CFD is mainly composited by three parts:

a) Noise
b) Time walk caused by offset voltage
c) Signal shape changing

The noise influence the jitter performance of the CFD by creating uncertainty when the output of the CFD crossing zero level. This influence is inversely proportional to the zero-crossing slop. By increasing the slop, the noise influence can be reduced.

Since the V_{offset} is very small (around some mV), the trajectory around the zero-crossing point can be approximately expressed by a proportional function as equ.3.4.

$$A\frac{dV}{dt}\Big|_{t=t_z} t_z = V_{offset} \tag{3.4}$$

And the corresponding time walk caused by amplitude change can be as:

$$\Delta t_z = \left(\frac{1}{A_{min}} - \frac{1}{A_{max}}\right)\frac{V_{offset}}{\frac{dV}{dt}\Big|_{t=t_z}} \tag{3.5}$$

where A_{min} and A_{max} mean the minimum and maximum value of amplitude factor.

From the equ.3.5, it can be easily concluded that the time walk decrease as the slop increase. Since the amplitude scale factor is determined by the signal source. While the offset voltage depends on the noise level in the system. In order to minimize the time walk, the zero-crossing slop should be maximized.

As what is stated before, the coefficients of the CFD, the delay time t_d and fractional number f, can be chosen carefully to maximize the zero-crossing slop. This goal can be achieved in an analytical way by finding the expression of the CFD's output and then the optimized t_d and f can be calculated by applying the constraints that the second derivative equals to zero and the corresponding time is the zero-crossing time. But this optimization problem is not as easy as it looks like. First, the input signal shape is usually difficult to be expressed by a mathematical formula or owns a very complex expression. Second, the CFD structure evolves the calculation of the signal after six order RC filter, which needs resolve a six order differential equation with a very complex excitation. Once the input signal change its shape, the whole procedure has to be executed again. The extremely complex calculation and low degrees of freedom lead to a conclusion that an analytical approach is not feasible.

But in another point of view, since the coefficients of a CFD is usually bounded and have a limited possible selection, which means an exhaustive method based on

numerical calculation can be implemented to find the optimal solution. The principle of this method is that perform the simulation of the CFD with every combination of t_d and f and the optimal coefficients will be selected through comparing their zero-crossing slop. There are several advantages of this approach. First, there is no necessary to find an analytical expression of the signal shape. A sample of signal is enough, which can be easily captured and recorded by an oscilloscope. Second, it is very easy to perform, just implement the structure of the CFD in the simulator, even the structure is very complex and highly non-linear, and the rest are handled by numerical solver.

A fully automatically coefficient optimization program is implemented based on this principle in the MATLAB, a powerful scientific computing tool. The simulation model of CFD is implemented in Simulink (Figure 3.15). Non like an electronic circuit simulation software that the simulation scheme is built based on electronic components , the R-C low pass filter is equivalently replaced by a $\frac{1}{Ts+1}$ transfer function block in Simulink, where T is the time constant of an R-C filter. The arming condition discriminator is also included to avoid fake triggering during zero input condition. A high pass filter is used to calculate the derivate value.

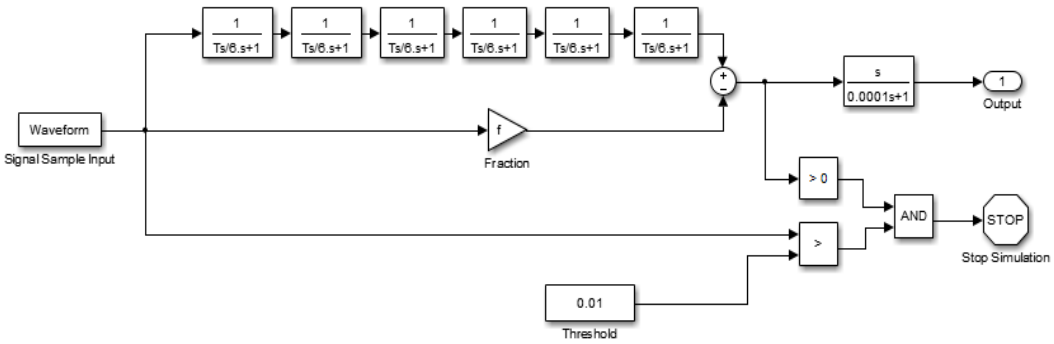


Figure 3.15 The simulation model

In order to record the zero-crossing slop, the simulation model is designed to stop the simulation immediately when the zero-crossing condition is detected, where the

slop is the last output of the simulation. The range and data number of t_d and f have to be defined to compose a group of combination set. During the routine, every combination set is input into simulation model, then the simulation model is run. Once the simulation has terminated, the last value of its output which represents the zero-crossing slop is recorded in a matrix. When the routine has finished, found the maximum value in this matrix and the corresponding row and column index represents the combination set. An example of result is shown in Figure 3.16, 3.17 and 3.18.

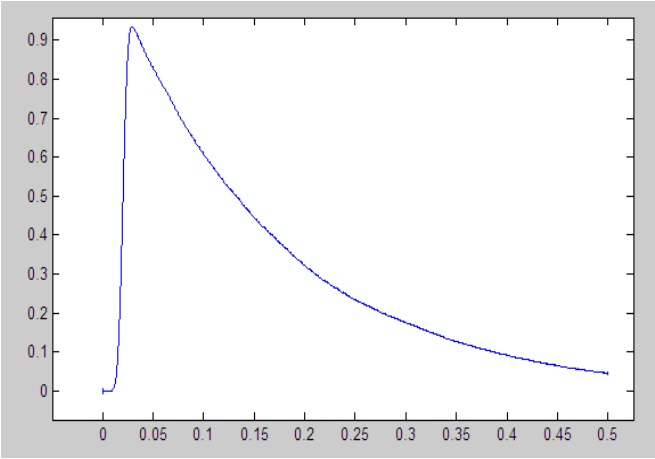


Figure 3.16 The signal sample generated captured by an oscilloscope

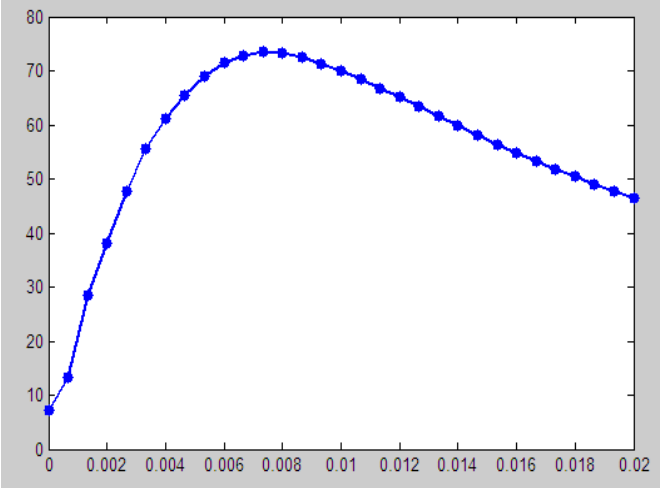


Figure 3.17 The maximum zero-crossing slop respect to t_d

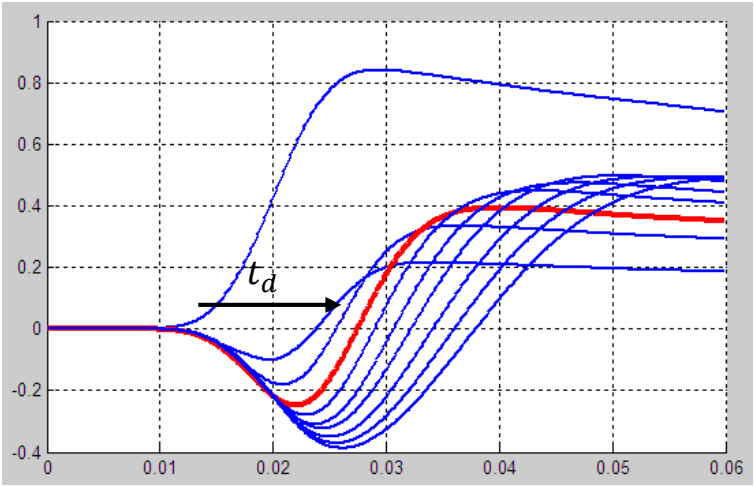


Figure 3.18 The response of CFD when t_d change from 0s to 0.02s.

The Figure 3.18 shows the result of the optimization where the f is set to obtain the maximum zero-crossing slop of each t_d. The red curve is the one with maximum zero-crossing slop

The example shows the feasibility of a numerical optimization method. By using this method, the optimization task has been significantly simplified respect to analytical method. But there are still a lot of problem to be solved such as the correctness of the Simulink respect to professional electronic circuit simulator, which can be solved by implementing more complicated simulation model since the basic principle of two types of simulation software are almost the same.

4. Software Design

This chapter illustrates the design of the software. The first part review the firmware design for FPGA and MCU. Since the FPGA plays a major role in the whole system, in the section of FPGA's firmware, from the micro architecture down to the VHDL implementation is introduced in detail to illustrate the organization and working principle. The firmware of the FPGA is modular designed to have a maximum flexibility for future function extension. In the section of MCU's firmware, the driver implementation for each onboard device as well as the TDC temperature control loop are illustrated. In the second part a Graphic User interface in Windows platform is introduced.

4.1. Firmware of FPGA

The FPGA plays a major role in the system. All system's features including:

- Synchronous Sampling
- Asynchronous Sampling
- Histogram Generation

are implemented inside the FPGA. Moreover it handles the TMU high speed serial interface as well as all the communication to PC, which is used to transfer the sampled

data and receive user instructions. The time measurement system supports a set of user instructions which can be used to control its behavior through the communication interface from PC. This instruction set will be introduced in detail in this section.

4.1.1. Firmware Architecture

The architecture of the firmware determine the quality and stability of the system. Respect to implementing everything in one VHDL file, which results a larger probability of having bugs and difficulty in maintaining, in this design each major part of the firmware is implemented in a VHDL file with defined port then all parts are “assembled” in the top layer VHDL file. The firmware is mainly consisted by 5 parts:

- Supervisor State Machine
- Clock Distributer
- RS-232 Serial Interface
- Function Blocks
- TMU Serial Interface Decoder

The top layer architecture of the firmware is shown in Figure 4.1.

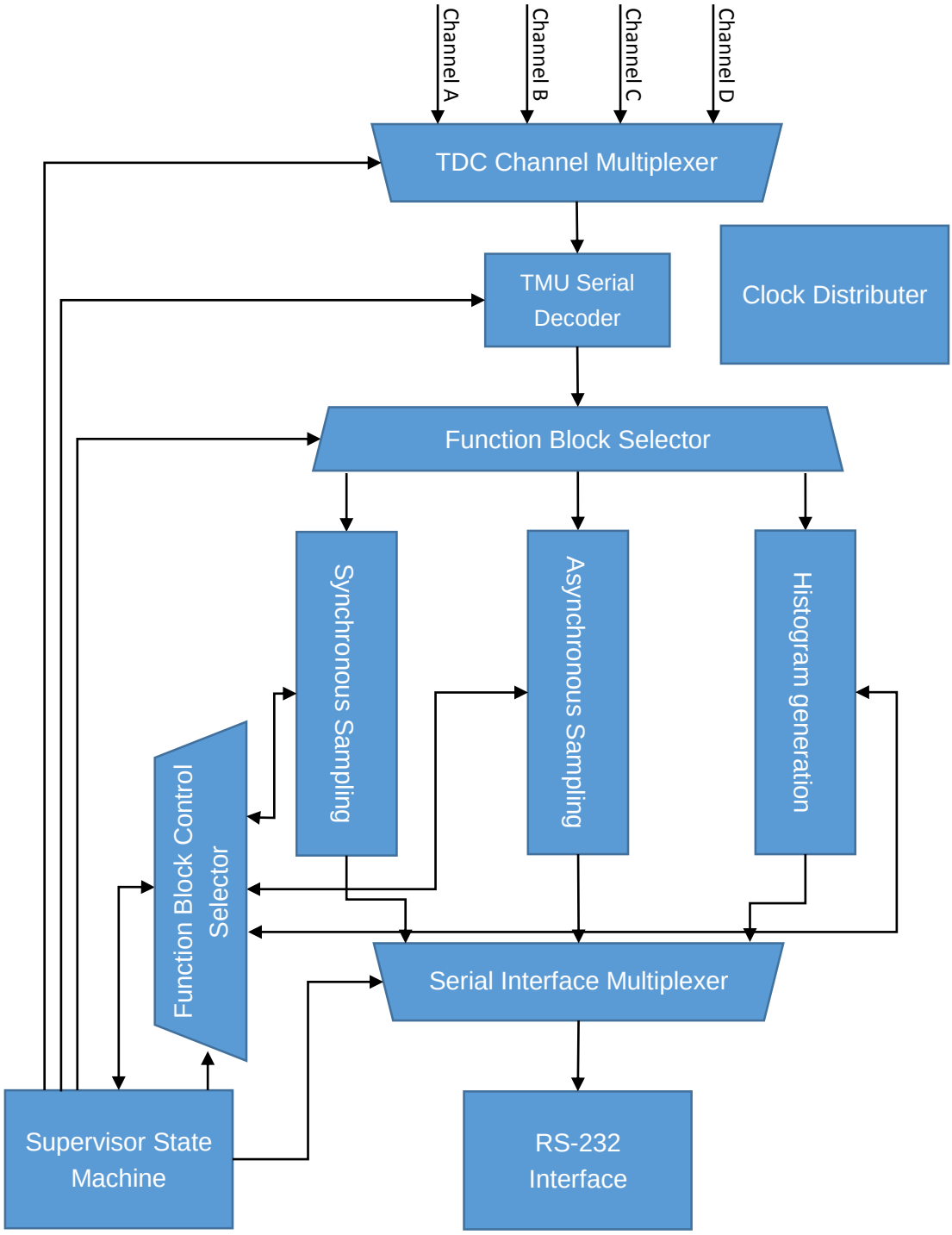


Figure 4.1 Block diagram of the firmware

Since most of the parts are composed by state machines, the Req-Ack mechanism is used to achieve the synchronization state between them rather than merge two state machine together. By using this architecture the I/O port of all function blocks can be standardized with two simple start/busy signal and the supervisor can control it without knowing the internal structure of a function. In addition to the Req-Ack mechanism, all signal path control value is designed to depend on the function identifier only, which is used to specify the function to be executed and supplied by user instruction. For example, when the function No.1 is requested to be executed, the supervisor just connect the input of function block No.1 to the desired TDC channel and the output to output interface. Then the supervisor will set the start pin of the function block and wait the execution to be finished. When a function is going to be added into the system, in the top layer VHDL file the design must be included and connect it to the corresponding port of Function Block Selector, Serial Interface Multiplexer and Function Block Control Selector. Then this function can be accessed directly by the supervisor state machine since the selectors and multiplexers are controlled directly by the value comes from the user instructions. By using this architecture the work of adding a function into the system is minimized.

Working Procedure

The Clock Distributer is used to generate the correct clock signal by dividing the input 300MHz clock. The supervisor state machine has a role of controlling the behavior of entire system and processing user instructions from serial interface. When certain function request instruction is received, it must configure the signal path, which is controlled by the TDC Channel Multiplexer, Serial Interface Multiplexer and Function Block Selector, so as to connect the desired function block to the TMU Serial Interface Decoder of defined channel and RS-232 Serial Interface. These control value are directly defined by the function identifier. Then it will set the start pin of the desired function block, which is the implementation of function, to start the execution. During the execution the busy pin of the function block will be set to halt the supervisor state machine in order to avoid the confliction. When the execution is finished, the

supervisor state machine will return to idle state and wait for new instruction.

The Req-Ack mechanism

The Req-Ack mechanism is used to synchronize each two state machines. When a synchronization between two state machines is required, for example, the TMU Serial interface decoder want to transfer a newly received data to supervisor, it will set the corresponding “require” signal to the supervisor. In the raising edge of supervisor’s clock, if the supervisor is raising edge triggered and it has been already in the state of waiting new data, this requirement will be detected. Then the supervisor will set the “Acknowledge” signal to tell decoder it has received the requirement and new data has been transferred. On receiving the acknowledgement the decoder would reset the “require” signal and elevates to next state. The corresponding example of state machine is shown below:

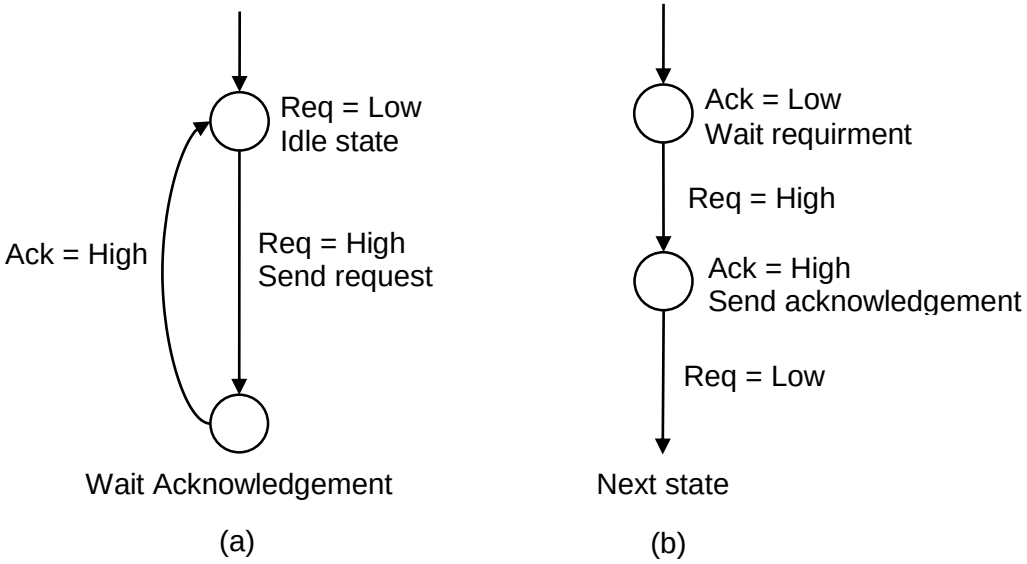


Figure 4.2 (a) The synchronization Requestor (b) The synchronization Recipients

Another usage of this mechanism is the implementation of waiting state. In the case of a master state machine want its slave state machine to perform some actions and the master has to wait until the execution finished, for example the supervisor state machine want the serial interface transfer a data through RS-232 interface. The

master will first set the “start” signal. In the trigger edge of serial interface it will receive this requirement and start the transmission procedure. During this procedure it will set a “Busy” signal until the transmission is done and return to idle mode. Once the supervisor has detected the busy signal is reset, it will proceed to next state. By using this mechanism, a giant state machine is divided into several sub-module which is easier to implement and have the possibility to make some of the sub-module being executed parallelly.

The corresponding example of state machine is shown in figure 4.3 (a) and (b).

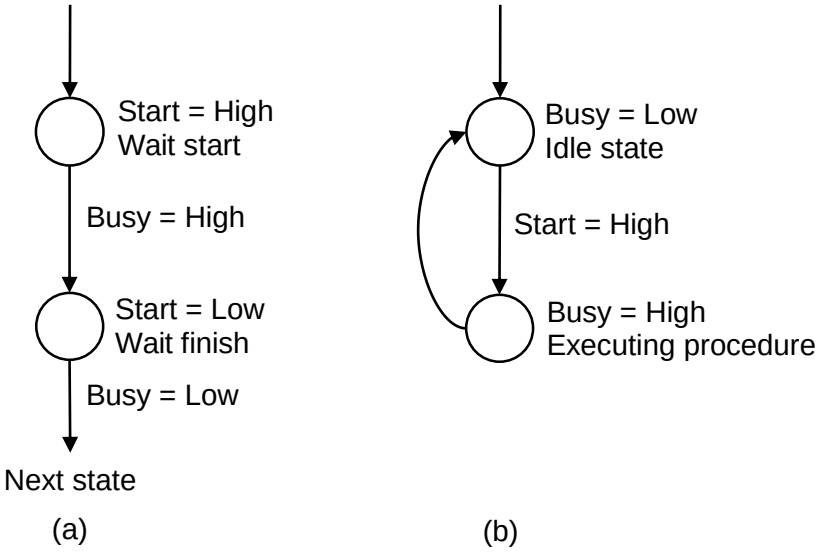


Figure 4.3 (a) The Requestor (b) The Executor

4.1.2. Clock Distributer

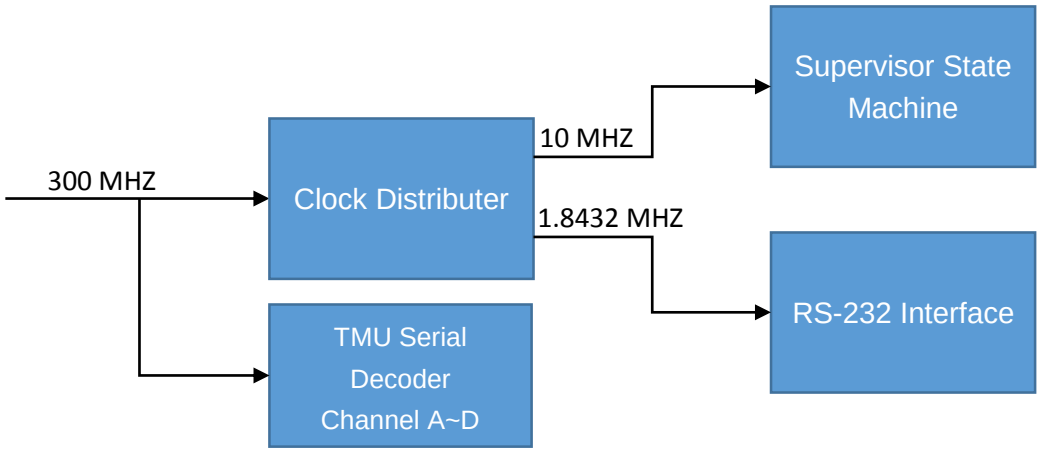


Figure 4.4 The Clock Distributer block

In the FPGA There are three parts work at 3 different clock domains. The TMU Serial interface decoder is clocked by the RCLK directly, which is 300 MHz. Then this clock is fetched into a clock distributer which contains 2 clock divider. The clock divider is based on the counter. When a defined clock cycles is reached, it will invert its out. The output frequency can be calculated as:

$$f_{out} = \frac{f_{in}}{(divide\ factor + 1) \cdot 2}$$

The first clock divider divide the 300 MHz clock by 30 thus get a 10 MHZ clock to drive the Central State Machine and function blocks. This clock assures that the system works at a faster rate than the throughput rate of the TMU serial interface, which is $\frac{300Mhz}{43} = 6.97Msamples/s.$ The second divider divide the 300 MHz clock by 163 as the requirement of RS-232 transceiver, which should be sixteen times of desired bund rate. The VHDL example code is shown as below.

```
Clock_Divider: PROCESS(Clock_In)
BEGIN
IF rising_edge(Clock_In) THEN --If the raising edge of the input
clock is detected
    IF Clock_Divider_Counter = Divide_Number THEN
    -- If the counter reaches the defined number, it will invert the
output and reset the counter
        Clock_Divider_Counter <= "0000000000000000";
            Output <= NOT Output;
    ELSE
        Clock_Divider_Counter <= Clock_Divider_Counter + 1;
    END IF;
END IF;
END PROCESS;
```

4.1.3. TMU Serial Decoder

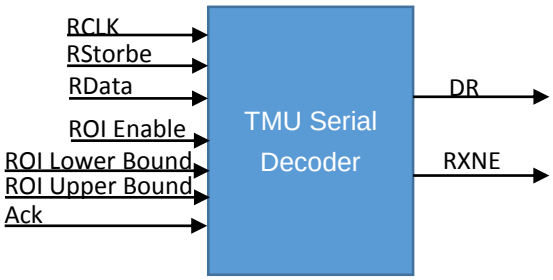


Figure 4.5 The TMU Serial Decoder Block

The port definition is shown in table 4.1:

Port Name	Direction	Description
RCLK	Input	The clock signal of the TMU serial interface.
RStrobe	Input	The strobe signal of the TMU serial interface.
RData	Input	The data signal of the TMU serial interface.
ROI Enable	Input	The ROI (Range of interest) function can be enable by setting this bit.

ROI Lower Bound	Input	The lower bound of the window of ROI function.
ROI Upper Bound	Input	The upper bound of the window of ROI function.
ACK	Input	The acknowledgment signal for the RXNE
DR	Output	The data buffer
RXNE	Output	This bit indicates that the data buffer is not empty

Table 4.1 The TMU Serial Decoder Block

The TMU Serial Decoder block is used to decode the serial data from TMU. The serial-results port consists of a clock signal (RCLK), four strobe signals (Rstrobex) and four data signals (Rdatax). The clock frequency is user selectable among 75, 150, and 300 MHz through the host programming interface. The Rstrobex signal indicates that a time-stamp data transfer is about to begin for the corresponding channel. The serial-result interface can be programmed to have a variable data-length format. Three register bits (Rlength0, Rlength1, and Rlength2), are used to program the required data transfer formats. Figure 4.6 shows a 16-bit result on the result interface.

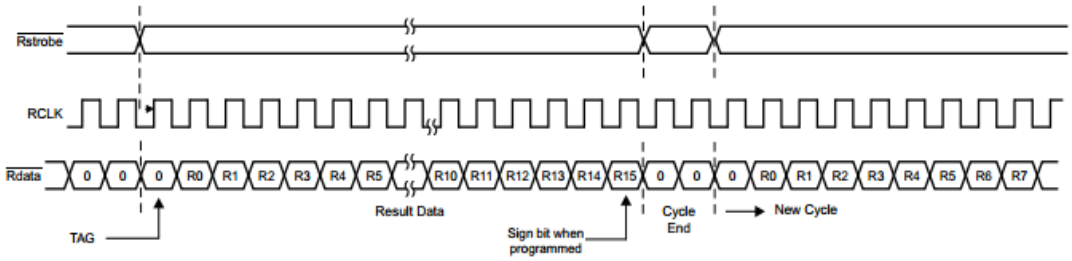


Figure 4.6 Result-Interface Operation

According to the serial protocol it takes 43 clock cycle to transfer 40 bits data. In the maximum throughput rate situation, the peak transfer speed per channel can be calculated as:

$$\frac{40}{43} \times 300{,}000{,}000Hz = 279069767bit/s = 33.26MB/s$$

The XILNX Virtex-5 supports an integrated LVDS receiver and digital controlled termination. In VHDL it is necessary to perform the instantiation procedure of the

integrated LVDS receiver so as to convert the differential signal into a single-ended signal inside the FPGA. The example instantiation VHDL code which is show below:

```
COMPONENT IBUFDS
GENERIC(DIFF_TERM:BOOLEAN;IOSTANDARD:STRING);
PORT(I,IB:IN STD_LOGIC;O:OUT STD_LOGIC);
END COMPONENT;
RCLK_Line :IBUFDS
GENERIC MAP (DIFF_TERM => TRUE,IOSTANDARD => "DEFAULT")
PORT MAP (O => TDC_Decoder_RCLK, I => RCLK_P, IB => RCLK_N);
```

The receiver is consisted by two processes. The first process consists a 40 bit shift register which is used to receive the TMU data and clocked by the RCLK signal. The shift register is enabled by the Rstrobe signal. The second process involves a state machine to handle the RXNE signal.

The ROI function is also implemented in this block. When the received data is out of the bound defined by the “ROI Upper Bound” and “ROI Lower Bound”, this data will be thrown and the RXNE signal will not be set. This function can be enable or disable by setting or resetting the “ROI Enable” port.

The VHDL code is show below:

```
Shift_Reg:PROCESS(RCLK)
BEGIN
IF rising_edge(RCLK) THEN
    IF Rstrobe = '0' THEN
        DR(38 downto 0) <= DR(39 downto 1);
        DR(39) <= RData;
    END IF;
END IF;
END PROCESS;
PROCESS(RCLK)
BEGIN
IF rising_edge(RCLK) THEN
```

```
CASE TDC_Decoder_State IS
    WHEN TDC_Decoder_Idle =>
--If the Rstrobe signal is driven low, which means the receive
procedure is undergoing, it will change the state to receiving state
       IF Rstrobe = '0' THEN
           TDC_Decoder_State <= TDC_Decoder_Receive;
       END IF;
    WHEN TDC_Decoder_Receive =>
--If the Rstrobe signal is driven High after being driven low, which
means the receive procedure is finished, it will set the RXNE signal
depends on the ROI function and return to idle state
       IF Rstrobe = '1' THEN
           TDC_Decoder_State <= TDC_Decoder_Idle;
           IF ROI_Enable = '1' THEN
               IF DR > ROI_Lower_Bound AND DR < ROI_Upper_Bound THEN
                   Data <= DR;
                   RXNE <= '1';
               END IF;
           ELSE
               Data <= DR;
```

4.1.4. RS-232 Serial Interface

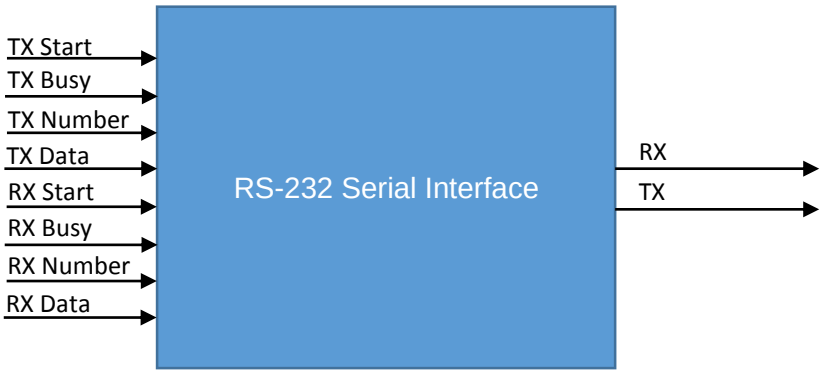


Figure 4.7 The RS-232 Serial Interface Block

The port definition is shown in Table 4.2:

Port Name	Direction	Description
TX/RX Start	Input	The start signal of the block. Together with the Busy signal they forms the Req-Ack mechanism.
TX/RX Busy	Input	It indicates that the serial interface is transmitting data or waiting data arrives.
TX/RX Number	Input	The desired number of bytes to be transmitted or received
TX/RX Data	Input	Data port of serial port
TX	Output	Transmit port of the serial interface
RX	Output	Receive port of the serial interface

Table 4.2 The port definition of the RS-232 Serial Interface Block

The core of the RS-232 serial transceiver uses a sub-modules of PicoBlaze, a free soft processor cores designed for Xilinx's FPGA, which supports a standard 8-bit, no-parity, 1-stop bit UART port with integral 16-byte FIFO buffers. The specialty of this soft serial transceiver is its low occupation of source which occupies only 40 slices of a FPGA.

The Serial Interface supports a variable length transfer/receive interface that allow sending and receiving a defined number data by only one request. When there is certain bytes of data to be received, the central state machine will set the desired number of bytes in the corresponding port and pending the receiving procedure to be completed. The procedure will terminate once the required amount of data is received and the received data is latched to the output port. The transmission procedure is similar, when the request of transmission is detected, it will transmit the defined number of data specified by the central state machine. This mechanism reduced the complicity of data transceiver in central state machine greatly.

The principle of state machine is not complicate. The stats machine contains five states and each of them is assigned a task of transmitting a byte locate in corresponding position of a 40 bits data port. Once a byte is transferred complete, it will elevate to next state automatically until the last byte is transferred. If a reduced number of byte is asked, for example 2 bytes, the state machine will jump to the forth state directly so that only last two transmission states will be executed. The data has

to be put in correct position since the transfer start from LSB of a 40bits data. The structure of receive state machine shares the same principle.

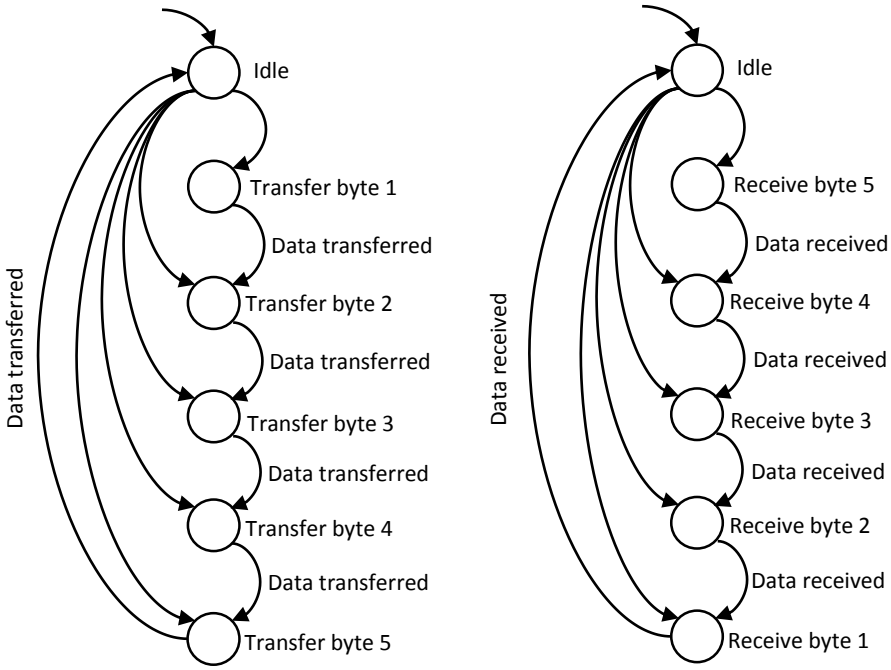


Figure 4.8 Block Diagram of the serial interface state machine

4.1.5. Function Blocks

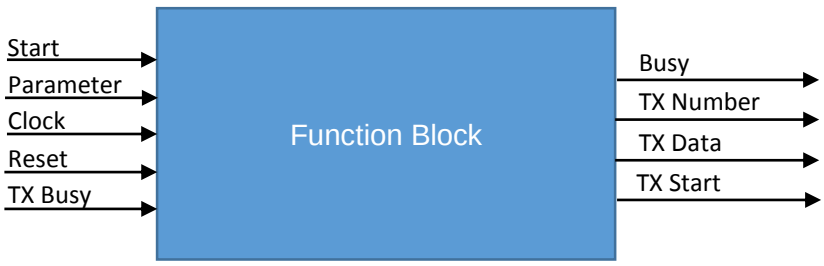


Figure 4.9 The generic function block

The port definition is shown in table 4.3.

Port Name	Direction	Description
Start	Input	The start signal of the function block. Together with the Busy signal they forms the Req-Ack mechanism.
Busy	Output	It indicates that the function block is running when set
TX Start	Output	The start signal of the serial port. Together with the TX Busy signal they forms the Req-Ack mechanism.
TX Busy	Input	It indicates that the serial port is running when set
TX Number	Output	The desired number of bytes to be transmitted
TX Data	Output	Data port of serial port
Parameter	Input	This parameter defines the total number to be collected.
Clock	Input	Clock input.
Reset	Input	Asynchronous reset of the function block. When set the function block will first run the RAM clear procedure then return idle state

Table 4.3 The port definition of the generic function block

There are three function blocks in the firmware which forms all functions of the time measurement system. The working principle of these blocks is introduced in detail in the next section.

Histogram generation

The histogram is a powerful tool to analysis the distribution of samples. Due to the limitation of serial port speed, the histogram generation function is implemented inside the FPGA by using the Block RAM of Virtex-5 so as to record all samples synchronously. The RAM is configured as 16bit wide address bus and 16bit wide word, where the address represents the data bin of the TMU results and its content represents the number of this data has been received. Every time a new TMU data is received, the corresponding content of RAM is read and add one then write back. Since the RAM is synchronized to the state machine, it is enough to use two states to finish the read and write operations while the Req-Ack mechanism is not required. After the desired number of samples to construct the histogram having received, it will

send a measurement finished command to the serial port then user can issue the “Upload Histogram” instruction to retire the content of RAM. This future can be executed parallel to other feature since the generation of histogram usually requires a great amount of data which is time costing. The diagram of the state machine is shown in figure 4.10.

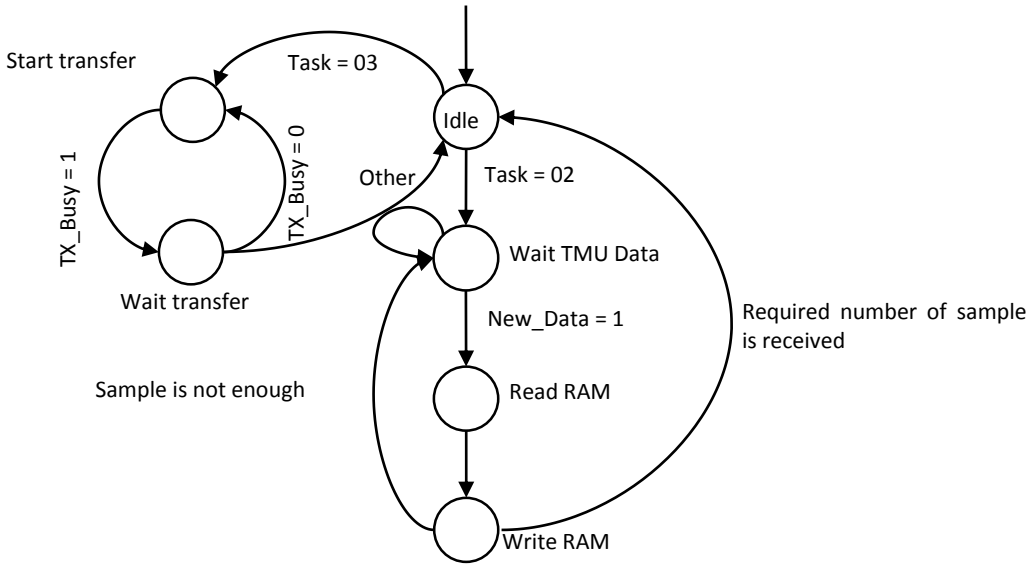


Figure 4.10 Block Diagram of the histogram generation state machine

Synchronous Data Reading

The synchronies Data Reading feature can be used to collect TMU data synchronously. Due to the fact that usually bit-rate of TMU is much higher than serial port, a FIFO is put between them to avoid data lose but the data amount is limited by the size of FIFO, which the depth is 65536. The FIFO is configured as a single port FIFO since there is no need of parallel read/write or different clock sources. Respect to the asynchronies reading, this feature partially solved the bit-rate mismatch problem with a cost of limited data number. All data is uploaded automatically right after the sampling is finished. The state machine is shown in figure 4.11.

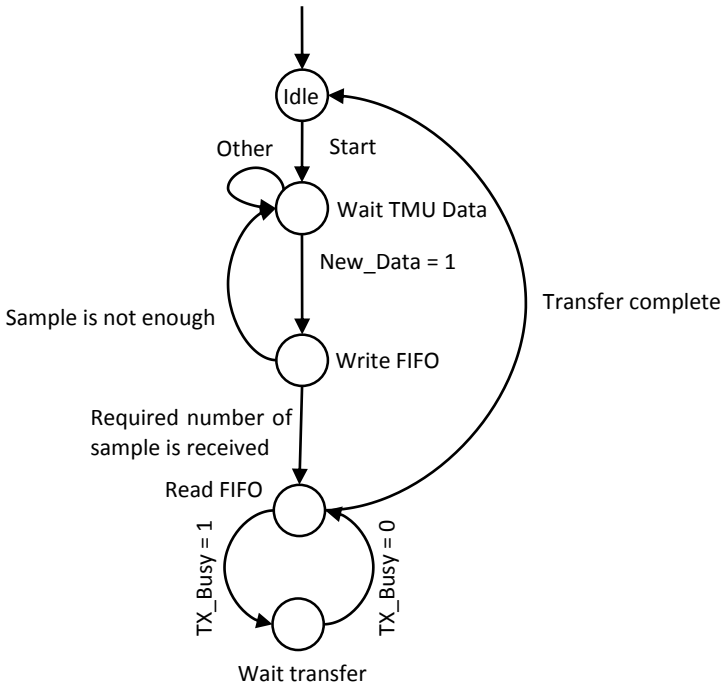


Figure 4.11 Block Diagram of the synchronous read state machine

Asynchronous Data Reading

This future can be used to gather a large amounts of data (max = 2^{32} samples) while the data is acquired asynchronously. The data-rate is limited by the speed of serial port. When the serial interface is busy transferring a data, all new data received by TMU decoder during this period will be abandoned until the transmission is finished. The architecture of the state machine is shown in figure 4.12.

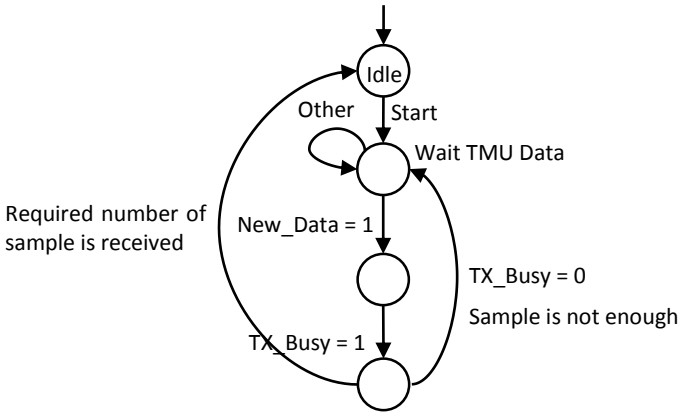


Figure 4.12 Block Diagram of the asynchronous read state machine

4.1.6. Multiplexer and Selector

There are two multiplexers and a selector in the firmware to build a flexible signal path. The example VHDL code is shown below:

```
Output <=  CH1 WHEN CH_Select = "00" ELSE
           CH2 WHEN CH_Select = "01" ELSE
           CH3 WHEN CH_Select = "10" ELSE
           CH4 WHEN CH_Select = "11";
```

```
PROCESS(Select_mux,Input)
BEGIN
CASE Select_mux IS
    WHEN "00" =>
        CH1 <= Input;
        CH2 <= (OTHERS => '0');
        CH3 <= (OTHERS => '0');
    WHEN "00" =>
        CH1 <= (OTHERS => '0');
        CH2 <= Input;
        CH3 <= (OTHERS => '0');
    WHEN "00" =>
        CH1 <= (OTHERS => '0');
        CH2 <= (OTHERS => '0');
        CH3 <= Input;
    WHEN OTHERS => NULL;
END CASE;
END PROCESS;
```

4.1.7. The Supervisor State Machine

The supervisor state machine has a role of controlling the behavior of entire system through control the channel selector and multiplexer as well as the execution of function block. Moreover it handles the processing of user instructions from serial interface.

User Instructions

The instruction enables user to control the FPGA through serial port to perform various desired measurement task. The user instructions is mainly composed by four types of instruction:

- 0x1F: The state synchronization instruction.
- 0xAA: The parameter upload request with ROI function disable
- 0xBB: The parameter upload request with ROI function disable
- One byte for specifying the measurement type and channel. The detail of function identifier and related mean of parameter are shown in the table 4.4.
- A 32bit integer function parameter.

Task	Type	Parameter
X1	Asynchronous Sampling	Total number of desired sample Maximum number is 2^{32}
X2	Histogram generation	Total number of desired sample Maximum number is 2^{32}
X3	Synchronous Sampling	Total number of desired sample Maximum number is 65536

Where x can be used to specify the desired channel

Table 4.4 Function identifier description

The instruction 0x1F, which in ASCII code corresponds to “?”, is used to synchronize the state machine and confirm the state machine is in “Idle” state. When a “?” is received, a 0xFF will be send back immediately. During next step, 0xAA or 0xBB can be sent to set the FPGA into parameter uploading mode. In the case a 0xAA is sent, a 0xAA would be returned from FPGA to tell the user the mode is entered successfully. Then a 5 byte parameter which composed by one byte of function identifier and 4 bytes of function parameter must be sent and FPGA will sent it back to confirm the received parameter is correct. In the case of bad instruction such as desired function or channel is Illegal, an “instruction error” messenger (00 00 00 00 00) will be sent. If the returned parameter is correct, the 0xFF can be sent to confirm or arbitrary byte to discard it and return to initial state. In the case a 0xBB is sent, which

asks the ROI function to be enabled, the only difference in the procedure is that three parameter rather than one parameter in the previous case have to be sent and confirmed. The three parameters are assigned as 1 byte of function identifier, 4 bytes of function parameter, 5 bytes of ROI Lower Bound and 5 bytes of ROI Upper Bound. The bound pair will be checked if the lower bound is higher than the upper bound.

The state machine is shown in next figure:

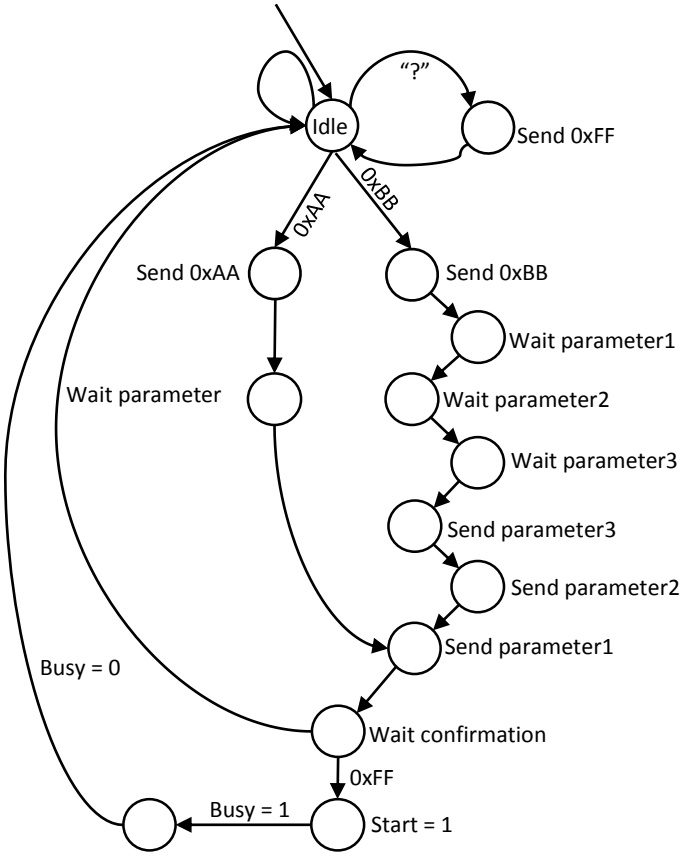


Figure 4.13 Block Diagram of the supervisor state machine

4.2. Firmware of MCU

The CY8C3866AXI MCU plays a role in configuring the clock generator and the TMU as well as monitoring their proper operation. Meanwhile the MCU is responsible

for the temperature control feature which is essential to the system's performance. The system architecture is shown is figure 4.14. The CDCE62002 clock generator is programmed through its standard SPI interface, which use the hardware SPI peripheral in PSoC 3 directly. The MCU communicate with THS788 time measurement unit through its host programming interface. Since this protocol isn't compatible with any existing hardware peripheral, it is emulated by software. A PWM waveform generator is used to control the H-Bridge so as to modulate the power and direction of heat transferring of the peltier.

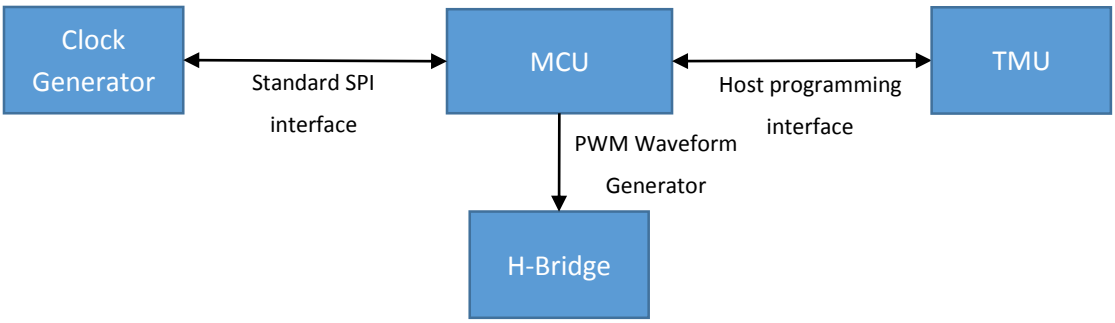


Figure 4.14 The system architecture

The develop environment is based on PSoC Creator, an official IDE released by Cypress. In the PSoC Creator both hardware and application firmware design of PSoC 3 systems can be implemented. Non like other MCU, the PSoC systems are designed with highly customable hardware witch supported by over 110 pre-verified, production-ready PSoC Components. PSoC Components are analog and digital "virtual chips" represented by an icon that users can drag-and-drop into a design and configure to suit a broad array of application requirements. Each component in the rich mixed-signal Cypress Component Catalog is configured with a Component Customizer and includes a full set of API libraries. The API library of each peripheral is generated automatically when the design is complied. The API is dynamically generated based on the configuration of corresponding peripheral. By using these API, the work load of driver developing is reduced significantly. Once the PSoC system has been configured, firmware can be written, compiled, and debugged within PSoC Creator.

In this section, the architecture of the firmware is introduced. The implementation

of driver for each device is described in detail from the communication protocol definition to the code which is fully commented in order to give a better understanding how the driver is implemented.

4.2.1. Firmware Architecture

The firmware is mainly composed by two parts: system initialization and temperature control loop. The system initialization function must configure the onboard device correctly so as to make the system operate properly. In the first step of initialization, all peripheral must be initialized by calling the initialization function of each peripheral which is included in the generated API library so as to enable the communication port to each onboard device. In the next step the initialization function of each device that implemented in their driver can be called to start the operation the system. The clock generator is initialized in first place since it has to supply the clock for THS788. Then the THS788 is initialized, which enables all its sampling channel as well as the internal temperature monitor which supplies the critical temperature data for the temperature control loop.

The temperature control loop is based on a PI controller which regulator the duty cycle of petiler as well as the heat dissipation rate. The PI controller itself is generated by RealTime Workshop in MATLAB. In the Simulink model, all I/O port name and time step must be configured properly, which is shown in figure X. Then the tool can generate the equivalent embedded C code from a Simulink model with tunable parameters by one clicking. The generated code can be used directly in the PSoC Creator by including the corresponding head file, which is shown in below:

```
#include "PI_Controller.h"
```

Once the head file is included, the function PI_Controller_initialize() which initialize all states of the model must be called before the model being executed. The input and output of the model are collected into two structures, which is PI_Controller_U and PI_Controller_Y. And their member is the same as the port defined in the Simulink model. For example, you can write the code:

```
PI_Controller_U.Set_Point = 25;
```

to set the set-point of the PI controller. Then the model can be executed by calling the function PI_Controller_Step(). As a result, the model is executed by a time step forward defined in the simulink based on the value in the input structure． After the execution, the output of the model can be read from PI_Controller_Y. The procedure of the execution of a generated model is shown in figure 4.15.

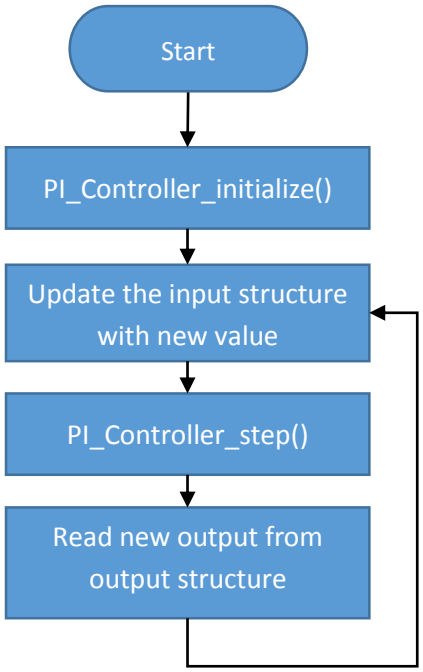


Figure 4.15 The procedure of the execution of a generated model

The loop time interval is precisely controlled and synchronous with a counter. The counter is clocked by a 1 MHz clock which is divided from 24 MHz master clock. The counter is configured as up count mode and reset itself every 20000 clock period, which corresponds to 20 ms or 50 Hz in frequency. When the counter reach its top, a terminal count flag will be set. When the loop has finished executing, it will wait this flag to be set so as to synchronize itself with the counter.

The Figure 4.16 shows the flow chart of the main loop.

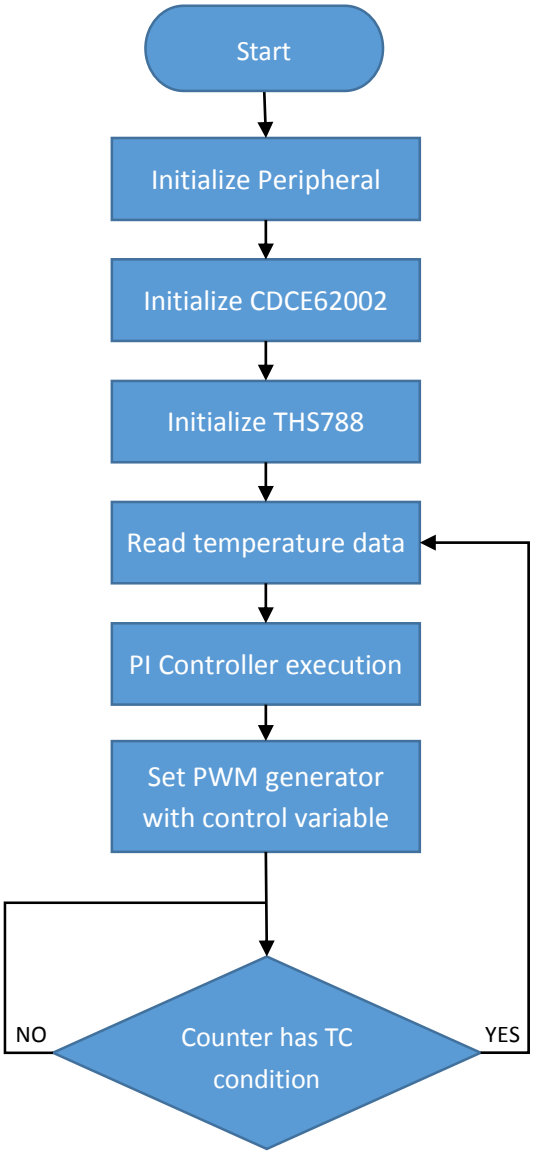


Figure 4.16 The flow chart of the main loop

4.2.2. Driver of CDCE62002

The configuration interface of CDCE62002 is a standard full-duplex Serial Peripheral Interface (SPI) interface which supports writing and reading to and from the device registers. It implements a low speed serial communications link in a

master/slave topology in which the CDCE62002 is a slave. The SPI consists of four signals:

SPI_CLK: Serial Clock (Output from Master) – the CDCE62002 and the master host clock data in and out on the rising edge of SPI_CLK. Data transitions therefore occur on the falling edge of the clock. (LVCMOS Input Buffer).

SPI_MOSI: Master Output Slave Input (LVCMOS Input Buffer).

SPI_MISO: Master Input Slave Output (Open Drain LVCMOS Buffer).

SPI_LE: Latch Enable (Output from Master). The falling edge of SPI_LE initiates a transfer. If SPI_LE is high, no data transfer can take place. (LVCMOS Input Buffer).

Writing to the CDCE62002

Figure 4.17 illustrates a Write to RAM operation. Notice that the latching of the first data bit in the data stream (Bit 0) occurs on the first rising edge of SPI_CLK after SPI_LE transitions from a high to a low. For the CDCE62002, data transitions occur on the falling edge of SPI_CLK. A rising edge on SPI_LE signals to the CDCE62002 that the transmission of the last bit in the stream (Bit 31) has occurred.

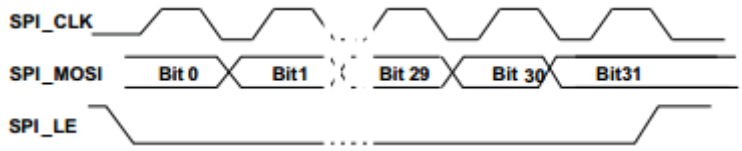


Figure 4.17 Write to RAM operation

Reading from the CDCE62002

Figure 4.18 shows how the CDCE62002 executes a Read Command. The SPI master first issues a Read Command to initiate a data transfer from the CDCE62002 back to the host (see Table X). This command specifies the address of the register of interest. By transitioning SPI_LE from a low to a high, the CDCE62002 resolves the address specified in the appropriate bits of the data field. The host drives SPI_LE low and the CDCE62002 presents the data present in the register specified in the Read Command on SPI_MISO.

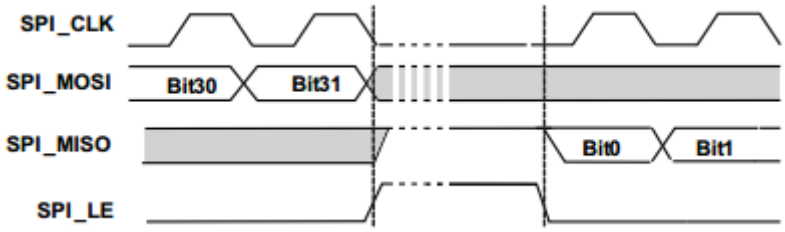


Figure 4.18 Read operation

Writing to EEPROM

After the CDCE62002 detects a power-up and completes a reset cycle, the device copies the contents of the on-chip EEPROM into the Device Registers. (SPI_LE signal has to be HIGH in order for the EEPROM to load correctly during the rising edge of Power_Down signal). The host issues a special commands shown in Figure 4.19 to copy the contents of Device Registers 0 and 1into EERPOM.

• Copy RAM to EEPROM – Unlock, Execution of this command can happen many times.

After the command is initiated, power must remain stable and the host must not access the CDCE62002 for at least 50 ms to allow the EEPROM to complete the write cycle and to avoid the possibility of EEPROM corruption.

			Data Field (28 Bits)																											Addr Field (4 Bits)			
Register	Operation	NVM	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	3	2	1	0
0	Write to RAM	Yes	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0	0	0	0
1	Write to RAM	Yes	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0	0	0	1
2	Status/Control	No	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0	0	1	0
Instruction	Read Command	No	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A	A	A	A	1	1	1	0
Instruction	RAM EEPROM	Unlock	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

Figure 4.19 CDCE62002 SPI Command Structure

SPI Consecutive Read/Write Cycles to the CDCE62002

Figure 4.20 Illustrates how two consecutive SPI cycles are performed between a SPI Master and the CDCE62002 SPI Slave.

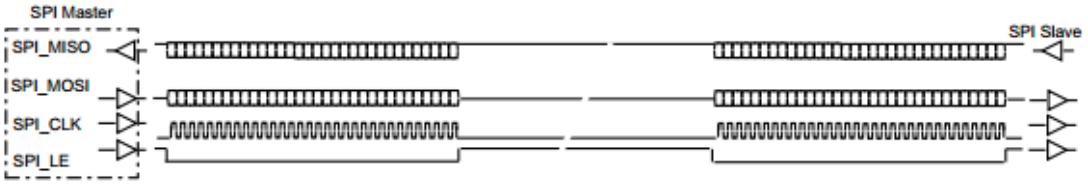


Figure 4.20 SPI Consecutive Read/Write Cycles

Peripheral Configuration

As what is introduced before, all peripherals to be used in a PSoC system must be included and configured into the design during the hardware design phase. In the Cypress Component Catalog there is already a per-defined standard SPI peripheral that can be used directly, which is shown in figure 4.21.

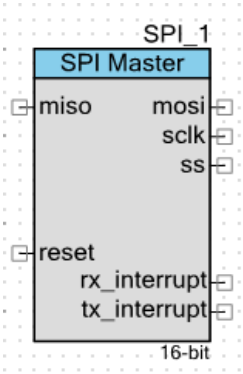


Figure 4.21 The SPI peripheral

Determining the coefficients of a SPI interface based on the protocol definition of the slave device is a critical part in configuring the SPI, otherwise the bus won't work. There are three basic SPI protocol coefficients: CPOL, CPHA and shift direction. Which are shown in figure 4.22. The CPOL determines the polarity when clock is in idle mode. The CPHA decides which edge of the clock is used for data sampling or latching. The shift direction controls whether LSB or MSB would be shifted out first.

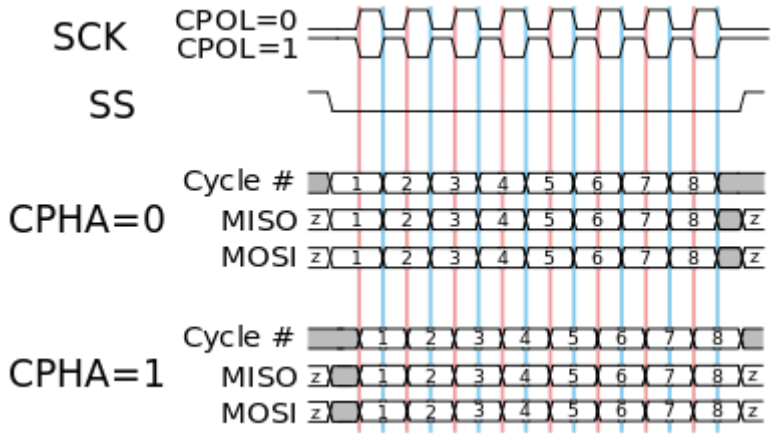


Figure 4.22 SPI protocol coefficients

Referring to the protocol definition of CDCE 62002, the coefficients is set to CPOL = 0, CPHA = 0 and shift direction = LSB First, which is shown in figure 4.23.

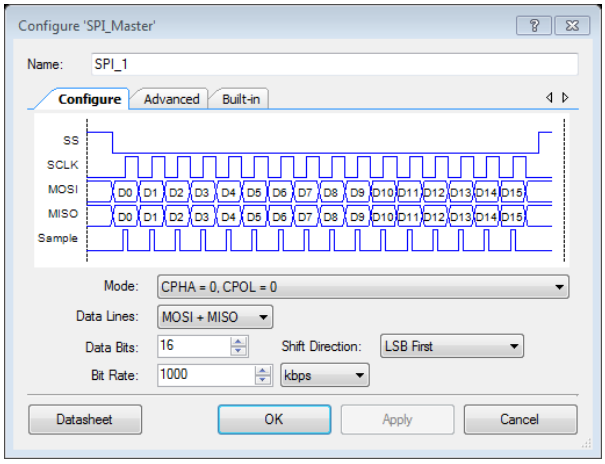


Figure 4.23 SPI peripheral settings

Computing the output frequency

Figure 4.24 presents the block diagram of the CDCE62002 synthesizer highlighting the clock path for a single output. It also identifies the following regions containing dividers comprising the complete clock path:

- R: Is the Reference divider values.

- O: The output divider value (see Output Block for more details)
- I: The input divider value (see Synthesizer Block for more details)
- P: The Prescaler divider value (see Synthesizer Block of more details)
- F: The cumulative divider value of all dividers falling within the feedback divider

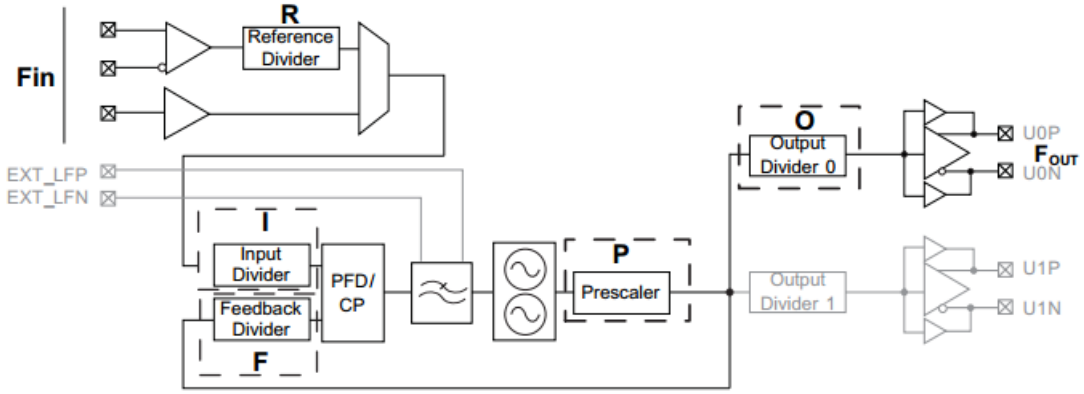


Figure 4.24 CDCE62002 Clock Path

With respect to Figure 4.24, any output frequency generated by the CDCE62002 relates to the input frequency connected to the Synthesizer Block by the following equation:

$$F_{OUT} = F_{IN} \frac{F}{R \cdot I \cdot O}$$

Equation X holds true subject to the following constraints:

$$1.750\ \text{GHz} < O \cdot P \cdot F_{OUT} < 2.356\ \text{GHz}$$

And the comparison frequency F_{COMP}:

$$40\ \text{KHz} < F_{COMP} < 40\ \text{MHz}$$

Where:

$$F_{COMP} = \frac{F_{IN}}{R \cdot I}$$

Due to the requirement of TMU, a 200 MHz is needed. With the 25MHz input, the

coefficients is selected as:

R = 1

I = 1

F = 40

P = 2

O = 5

$$F_{OUT} = 25MHz\frac{40}{1 \cdot 1 \cdot 5} = 200MHz$$

And the constraints are check:

$$O \cdot P \cdot F_{OUT} = 2000MHz$$

$$F_{COMP} = \frac{25MHz}{1 \cdot 1} = 25MHz$$

The calculation of coefficients and the corresponding register configuration can be done automatically by the GUI software released by Texas Instruments which is shown in figure 4.25.

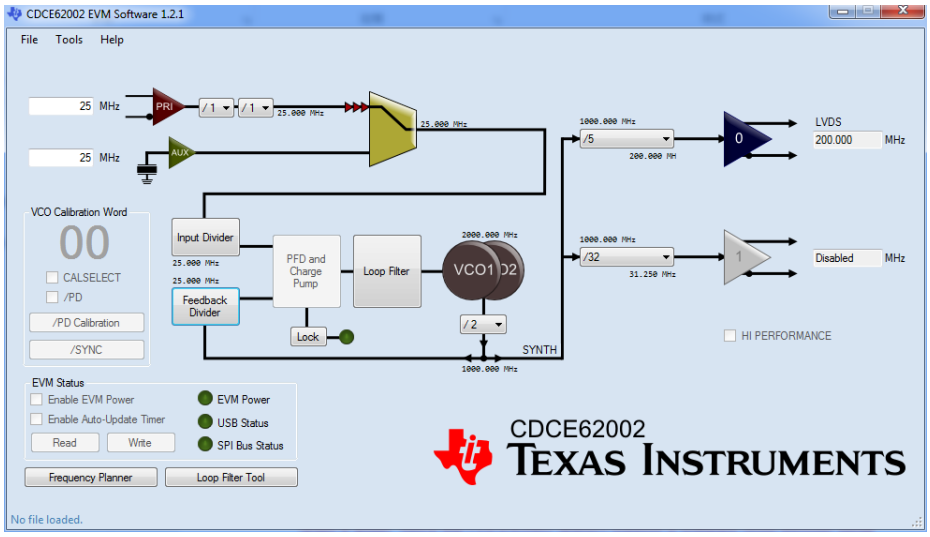


Figure 4.25 The GUI software

Implementation of the initialization function

The driver includes three Low-level functions: Reset, Read from and write to a CDCE62002's register. The device initialization function is implemented based on these three basic function.

Since the CDCE62002 doesn't support a software reset through serial interface, the reset function is realized by pulling up the reset pin of CDCE62002 and pull down it again.

The fundamental SPI write and read function supports writing a 16 bit data into SPI bus meanwhile receiving 16bit from the bus. The functions that write and read data from the data buffer of SPI have already implemented in the API library generated by PSoC Creator. The transmission is started automatically when certain data is loaded into the buffer.

```
uint16 spi_transfer(uint16 Data){
uint16 buffer;
SPI_1_WriteTxData(Data);
       // Load the data into buffer and start the transmission
while(0u == (SPI_1_TX_STATUS_REG & SPI_1_STS_SPI_DONE));
       //Check the state register to wait the transmission done
buffer = SPI_1_ReadRxData();
       //Reitre the received data from data buffer
return buffer;
}
```

The CDCE62002 read and write function supports read or write a 32 bit command into CDCE62002. Due to the fact that the SPI interface of PSoC 3 only supports 16bit word transfer, two SPI transfer command are put in series so as to transfer a 32 bit long data. The code of register read and write function is shown below:

```
void CDCE62002_Write(uint32 Data)
{
CDCE62002_CS_LOW();          //Pull down the chip select pin
```

```
uint32 CDCE62002_Read(uint16 Reg_Address)
{
uint32 ReadBuffer;

CDCE62002_CS_LOW();  //Pull down the chip select pin
spi_transfer(0x000E|Reg_Address);
                     //Transfer the register address to be read
spi_transfer(0x0000);
CDCE62002_CS_HIGH(); //Pull up the chip select pin

CDCE62002_CS_LOW();  //Pull up the chip select pin again to start
                     //the read sequence
ReadBuffer = (uint32)spi_transfer(0x0000);
                     //The SPI master must generate the SPI clock
                     //by sending any data to slave device
                     //The 16bit in LSB is first retired
ReadBuffer |= ((uint32)spi_transfer(0x0000) << 16);
                     //The 16bit in MSB is then received
CDCE62002_CS_HIGH(); //Pull up the chip select pin
```

After the data being written into the RAM of CDCE62002, a “write to EEPROM” command (0x00000001F) must be sent to CDCE62002 to make it store all data in the EEPROM. The CDCE62002 must be reset so as to reinitialize the VCO calibration procedure based on the new clock divider coefficient.

In the initialization procedure when the board is power up, the MCU will first read all register file from the CDCE62002 and compare them to the desired value. If they are the same, the MCU will then check the correct operation of the clock generator by looking at the value of “PLL Lock” pin, where the high value means the clock generator is working properly. In the case of mismatch of the register file between the clock generator and MCU, the MCU will update them and restart the initialization procedure.

The flow chart of the initialization procedure is shown below:

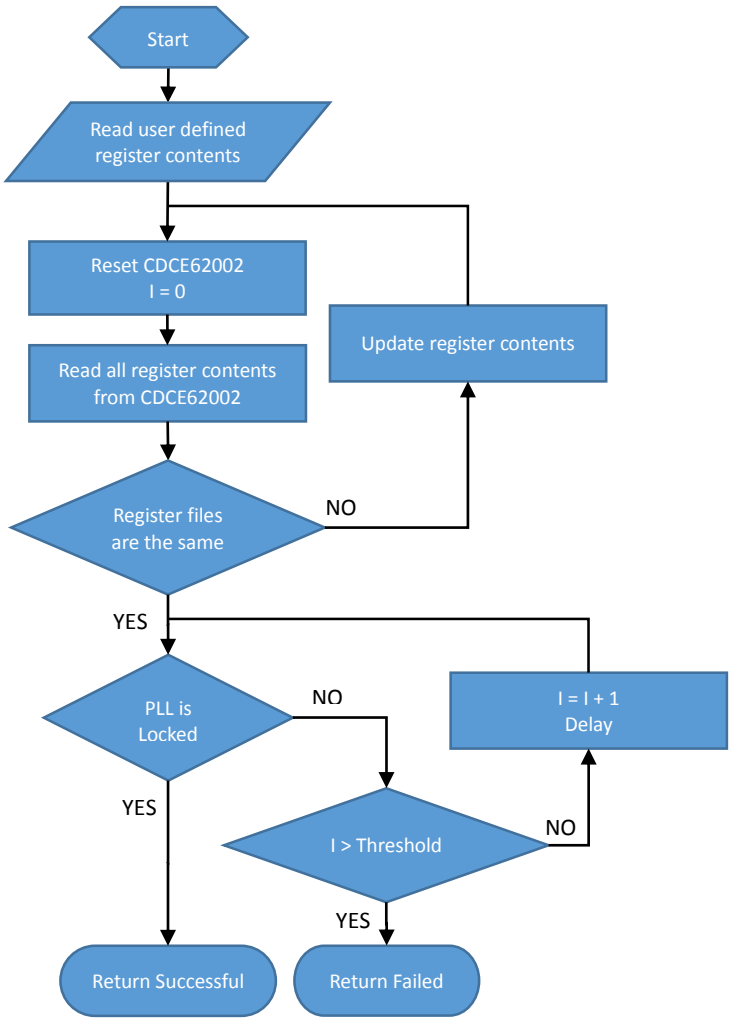


Figure 4.26 The initialization procedure of CDCE62002

4.2.3. Driver of THS788

The TMU host interface is very close to a 3-wire SPI interface but its protocol is not fully compatible with it. The interface operates at speeds of up to 50 MHz. Register addresses are 8 bits long. Data words are 16 bits wide, enabling more-efficient interface transactions. The serial bus implementation uses three LVCMOS signals: HCLK, Hstrobe, and Hdata. The HCLK and Hstrobe signals are inputs only, and the Hdata signal is bidirectional. The HCLK signal is not required to run continuously. Thus,

the host processor may disable the clock by setting it to a low state after the completion of any required register accesses.

When data is transferred into the device, Hdata is configured as an input bus, and data is latched on a rising edge of HCLK. When data is transferred out of the part, Hdata is configured as an output bus, and data is updated on the falling edge of HCLK. Hstrobe is the control signal that identifies the beginning of a host bus transaction. Hstrobe must remain low for the duration of the transaction, and must go high for at least two clock cycles before another transaction can begin.

Read Operations

Reading the THS788 registers via the host interface requires the following sequence:

The host controller initiates a read cycle by setting the host strobe signal, Hstrobe, to a low state. The serial Hdata sequence starts with a high R/W bit, followed by (either 1 or 0) for parallel-write bit and 8 bits of address, with most-significant bit (A7) first. The host controller should put the Hdata signal in the high-impedance state beginning at the falling edge of HCLK pulse 10. The THS788 allows one clock cycle, (r) for the host to reverse the data-channel direction and begins driving the Hdata line on the falling edge of HCLK pulse 12. The data is read beginning with the most-significant bit (D15) and ending with the least-significant bit (D0). The host must drive Hstrobe to a high state for a minimum of two HCLK periods beginning at the falling edge of HCLK pulse 27 to indicate the completion of the read cycle. Figure 4.27 shows the timing diagram of the read operation.

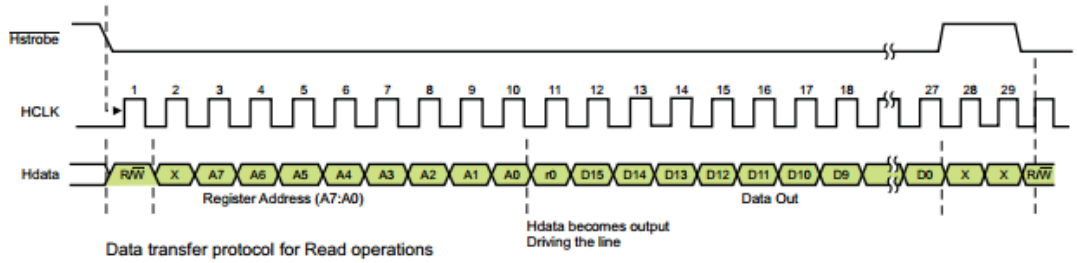


Figure 4.27 Read Operation of the host interface

Write Operations

Writing into the THS788 registers via the host interface requires the following sequence:

After the Hstrobe line is pulled low (start condition), the R/W bit is set low, followed by a 0 for the parallel-write bit (single-register write), then the memory address (A7–A0) followed by the data (D15:D0) to be programmed. The next clock cycle (w) is required to allow data to be latched and stored at the destination address (or addresses in the case of a parallel write), followed by at least two dummy clock cycles during which the Hstrobe is high, indicating the completion of the write cycle. Figure 4.28 and Figure 4.29 show timing diagrams of write operations.

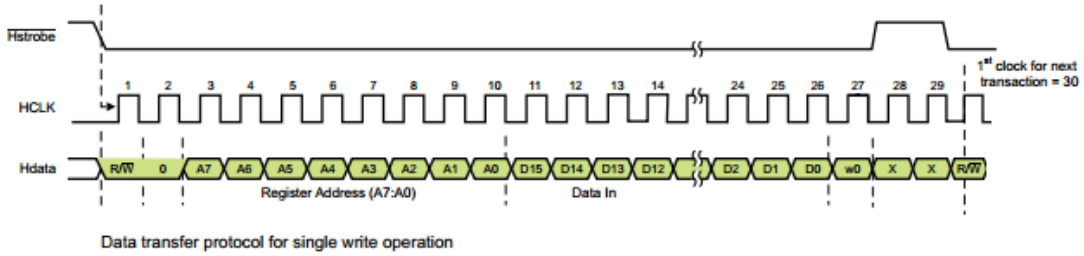


Figure 4.28 Write Operation of the host interface

Write Operations to Multiple Destinations

This operation would configure the same type of register for all four channel in a single write operation. The only difference respect to normal single write operation is that the parallel-load bit is set to 1.

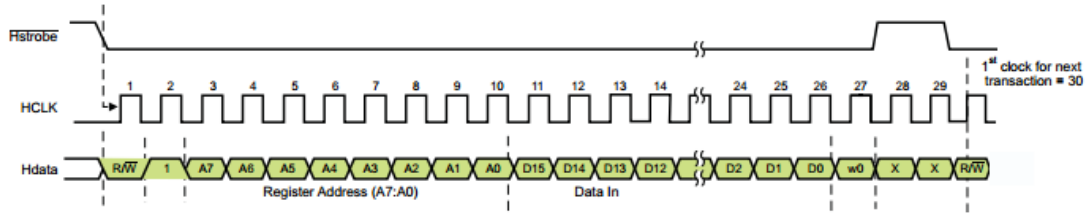


Figure 4.29 Write Operations to Multiple Destinations

Core temperature sensor

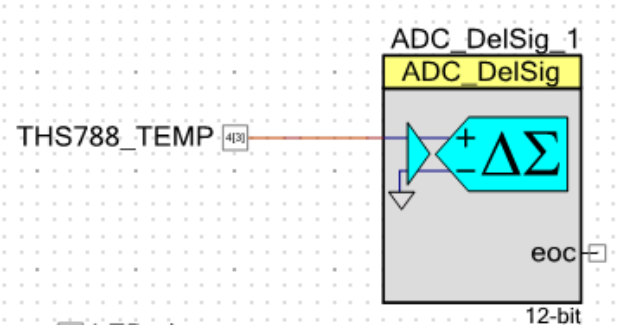


Figure 4.30 The per-defined sigma-delta ADC

The output of the temperature sensor is an analog voltage proportional to the temperature. An ADC (Analog-to-Digital Converter) must be included in the hardware design to sample this signal (shown in Figure 4.30), which use the per-defined sigma-delta ADC in Cypress Component Catalog directly. The ADC is configured as single-ended mode with zero reference to ground and full scan range to 2.048v, which use an internal precise voltage reference (shown in Figure 4.31). The resolution is chosen as 12bit so as to have a LSB of 500uV, where the corresponding read precision of 0.1℃.

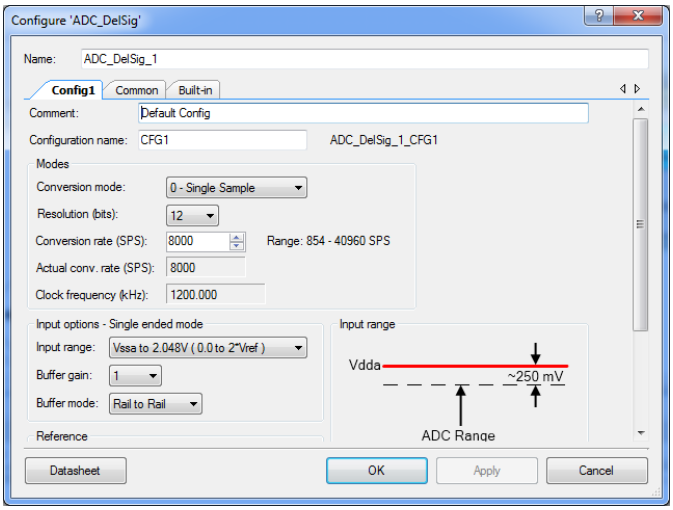


Figure 4.31 The configuration of the ADC

Library Implementation

The library supports three functions: Reset, register write/read and temperature reading. The register reading and writing functions are written fully in software which by write or read the level of the corresponding GPIO. The core of reading and writing functions which generate clock signal and read/write data are written respect to the timing specification of the bus that the data must be latched in falling-edge and read in raising-edge at the side of MCU. The implementation of clock generation function is shown below:

```
void send_one_bit(uint8 Data){
THS788_Bit_Write(Data&0x01);//Set the output pin
//When several send function is put in series, the output is
//latched after the falling edge of previous function execution
//This configuration assures the clock line is low in idle state
THS788_SCLK_HIGH();//Set the clock pin
THS788_SCLK_LOW();//Reset the clock pin
}

uint8 receive_one_bit(void){
uint8 ReadBuffer;

THS788_SCLK_HIGH();//Set the clock pin
ReadBuffer = THS788_Bit_Read();// Read the data at raising edge
THS788_SCLK_LOW();//Reset the clock pin
return ReadBuffer;
}
```

Based on the fundamental bit read and write function, the host interface register read and write function can be easily implemented. The code is shown below:

```
uint16 THS788_Read(uint8 Address)
{
int8 i;uint16 ReadBuffer = 0;
/*|R/W|1/0|A7|A6|A5|A4|A3|A2|A1|A0*/
```

```
THS788_Data_Output_Mode();  //Change the data pin to output mode
THS788_CS_LOW();            //Pull down the chip select pin
send_one_bit(1);            //|R/W|,1 means reading
send_one_bit(0);            //|1/0|,0 means Single register access
for(i = 7;i >= 0;i--){
send_one_bit(Address >> i); //|A7|A6|A5|A4|A3|A2|A1|A0|
}

THS788_Data_Input_Mode();   //Change the data pin to input mode
receive_one_bit();          //|r|, Redundancy bit
/*|D15|D14|D13|D12|D11|D10|D9|D8|D7|D6|D5|D4|D3|D2|D1|D0|*/
for(i = 15;i >= 0;i--){
ReadBuffer |= (uint16)(receive_one_bit()&0x01) << i;
}

THS788_CS_HIGH();           //Pull up the chip select pin
/*Send two additional clock as the requirement in datasheet*/
receive_one_bit();
receive_one_bit();

return ReadBuffer;
}
```

The RESET function is performed by set the reset pin of the THS788 followed by a reset.

The temperature reading is based on read the ADC that sampling the output of the analog temperature sensor in the core of THS788. Then the ADC value is converted into real temperature by following formula:

$$T_{REAL} = T_{ADC} \times 0.01 - 273$$

4.2.4. Driver of NCV7729

This H-Bridge is used to modulate the duty cycle of the pertiler. The integrated switches in this H-Bridge can be controlled by input signals (INx) as well as via the SPI

Interface. Mode selection is performed via the SPI configuration register. In this case the control is performed through input signals since it would be easier to be regulated by a PWM waveform generator. The device provides two enable inputs: EN = active high and DIS/SF = active low. The default setting for DIS/SF is to operate as an enable input. All control inputs provide internal pull-up (IN1, IN2, DIS/SF) or pull-down (EN) to ensure defined functionality in case of open pin conditions. Bridge control logic is shown in Table 4.5 demonstrates all Operational Modes.

Operation Mode	EN	DIS	IN1	IN2	OUT1	OUT2
Forward	H	L	H	L	H	L
Reverse	H	L	L	H	L	H
Free−wheeling low	H	L	L	L	L	L
Free−wheeling high	H	L	H	H	H	H
Disable via DIS/SF	X	H	X	X	Z	Z
Disable via EN	L	X	X	X	Z	Z

Where H = High, L = Low, Z = High impedance, X = don’t care.

Table 4.5 The truth table of the bridge control logic

PSoC Peripheral Configuration

In order to regulate the duty cycle of the signal, a pwm generator is included in the design (shown in figure 4.32). When one input of H-Bridge is regulated by the PWM generator while the other pin keeps LOW state, the duty cycle of pertler is modulated. The PWM generator is clocked by the 24 MHz directly and the period is set to 255. This setting gives output a 94.1 KHz, 255 level adjustable PWM signal which makes the current of pertler enough smooth.

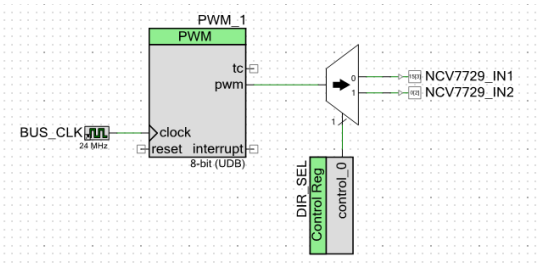


Figure 4.32 The PWM Peripheral

The direction control is achieved by a two channel de-multiplexer. When one of two output channel is selected through the channel select pin, the other output pin will stay low as what is stated in the table 4.6 of de-multiplexer component.

Output Select	Input	Output[0]	Output[1]
0	0	0	0
0	1	1	0
1	0	0	0
1	1	0	1

Table 4.6 The truth table of the de-multiplexer

The channel select pin is connected to a register so as to be configured in the main loop. When the bit is set high, the H Bridge will drive the peltier in heat direction and vice versa.

4.3. Graphic User interface (GUI)

A Graphic User interface (GUI) is designed to help the user perform their desired measurement or analysis more easily using the time measurement system. The GUI is written in C# which supports a fast GUI development under the windows platform. In the software, all settings of the TDC can be easily configured in a dialog. A User can start a measurement by configuring the TDC and supplying all necessary experiment parameters then the data plotting function will display the acquired data based on the selected experiment type with cooperation with FPGA using the protocol introduced before.

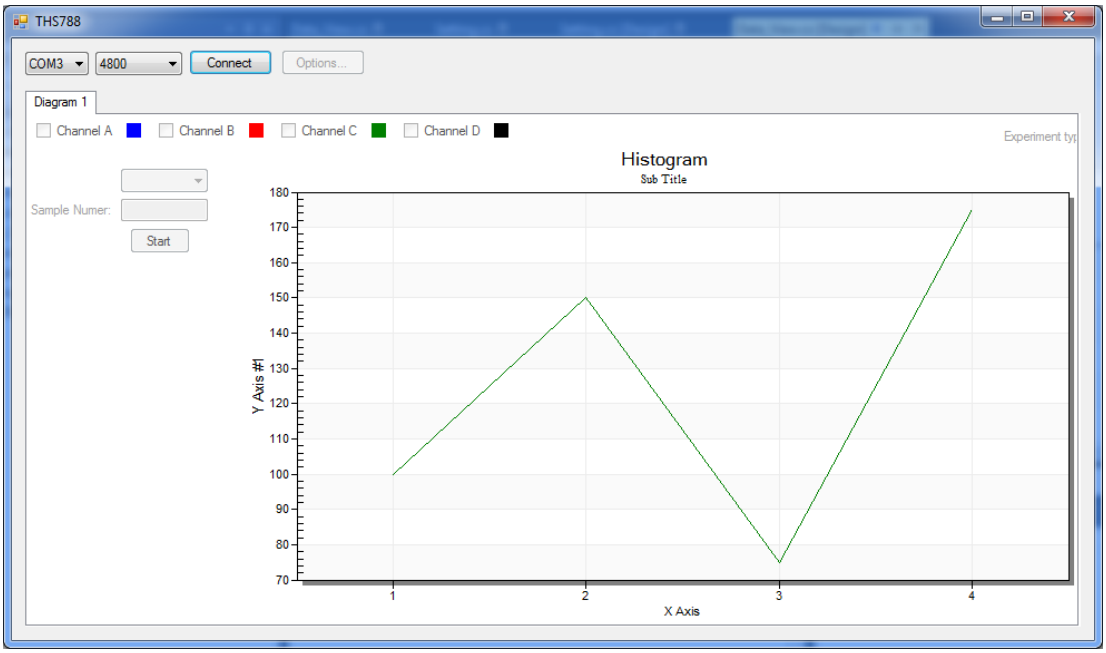


Figure 4.33 The main inerface

Figure 4.33 shows the main window. The upper left part is the serial port configuration. The correct serial port parameters including the port number and baud rate must be chosen for proper function of the system. Once the connect button is clicked, the software will issue an inquire command to check connectivity to the FPGA. If the connection is established, the option and experiment management button is

enabled for further command.

The software is mainly composed by three parts: communication, Options and data plotting.

Communication

This part process the requirement from user and handle the communication protocol between the software and time measurement system. The communication is design into two layer, the algorithm layer and driver layer, which enables for future upgrade to other communication port such as USB so as to achieve a higher performance.

The driver layer mainly consist the port initialization as well as the fundamental transmitting and receiving function. These functions are hardware related and the algorithm layer will call these functions directly for initialization and transmitting data without knowing what the physical port it is. Currently the communication is realized through serial port in PC, which is connected to the FPGA through a USB-to-Serial convertor. In Visual C# there is already an easy-to-use toolbox of serial port implemented. It only needs several lines of code to be initialized. The example code of initialization is shown below:

```
private void Open_Port(){
/*Load the port name and baudrate from selection box*/
SerialPort.PortName = Comport_Selection_ComboBox.Text;
SerialPort.BaudRate = int.Parse(Baudrate_Selection_ComboBox.Text);
/*Try opening the port, if there is any error, the error message will */
/*be shown in a pop up dialog*/
try
{
    SerialPort.Open();
}
catch (Exception ex)
{
    MessageBox.Show(ex.Message);
}
}
```

The receiving and sending function supports transfer or receive one byte from the serial port. These functions use the library function of the serial tool box directly. The example code of the read/write functions is shown below:

```
private void Send_Byte(Byte Data){
Byte[] TxData;

    TxData = new Byte[1];
    TxData[0] = Data;
    SerialPort.Write(TxData, 0, 1);
}

private Byte Receive_Byte()
{
    return (Byte)SerialPort.ReadByte();
}
```

The receiving and sending function is allocated to a standalone thread where a state machine is implemented to handle the request sending and data receiving. This

is the algorithm layer. The evaluation of the state machine is based on the user command and the state report from the FPGA. Also it have the job of handling the exception such as a data receiving error or requirement confliction. When a data receiving error is detected, for example, the desired number of data is not shown before the timeout event occur, the state machine will try to inquire the data again from the FPGA. It will also generate the state report to main interface where display the current state of the measurement.

Options

This part of GUI implements an easy and fast way to configure the TDC and each channel. Every configuration available inside the TDC is realized in this dialog with a more user-friendly way. When the OK button is clicked, all configuration will be converted into corresponding bit field of each register and send to TDC. A brief introduction of this dialog is shown below.

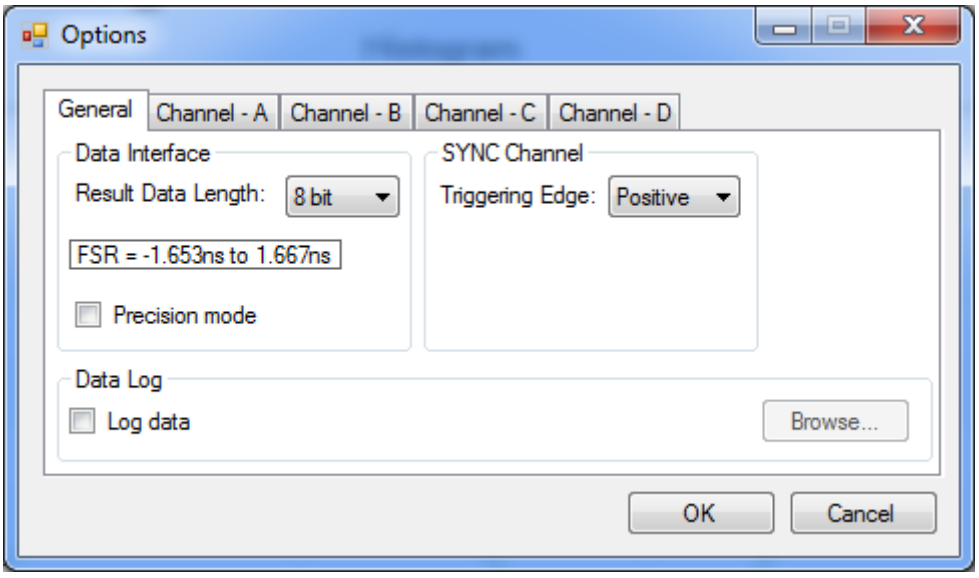


Figure 4.34 The general tab

The dialog consists five tabs. In the general tab locates the common setting of the TDC and software. It supports to choose the result data length and selecting the trigger edge of SYNC channel. The corresponding full scan range of each result data length

configuration is calculated and shown in the interface.

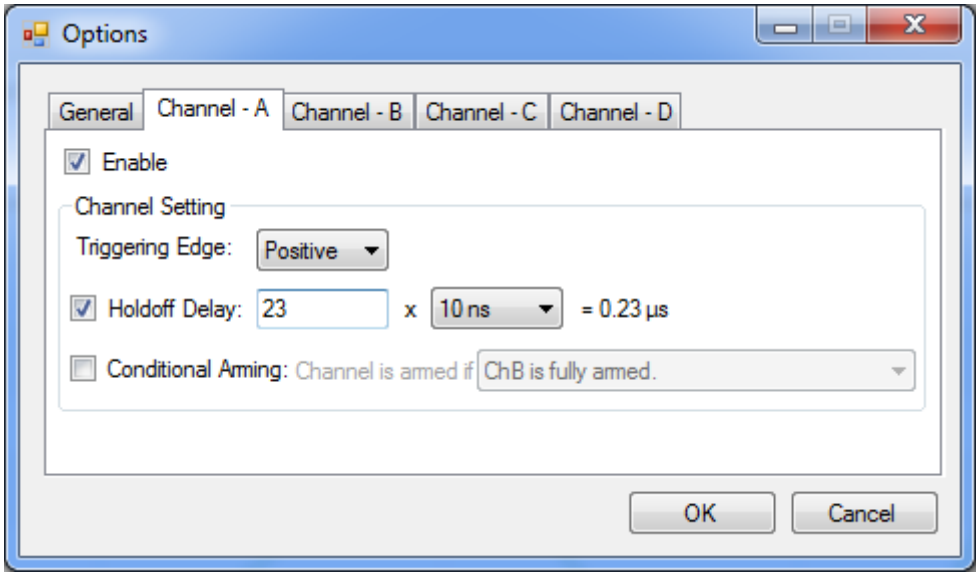


Figure 4.35 The channel setting tab

In this tab collects all each channel related options. The holdoff delay will be automatically calculated if any configuration is applied, which makes the setting of holdoff delay much easier.

Data plotting

This part will plot the received data such as a histogram or a series of sample. The core of this feature uses a toolbox called ProEssentials from Gigasoft. This toolbox supports a fast and easy to use diagram generation for engineering, manufacturing, financial, and handling larger data-sets. It has integrated many special effects for data plotting and background shading inside the toolbox which can be used directly by setting the value of corresponding property in code.

Experiment Control:

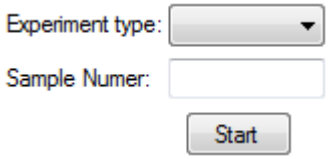


Figure 4.36 Experiment Control

In this part of GUI you can choose the desired feature and specify the corresponding parameters like how many samples to be collected. The parameters will be checked dynamically respect to the selected feature. Once the start button is pushed, the specified test is executed and the data will be shown in the data plotting area once the procedure has been finished.

5. Test & Conclusion

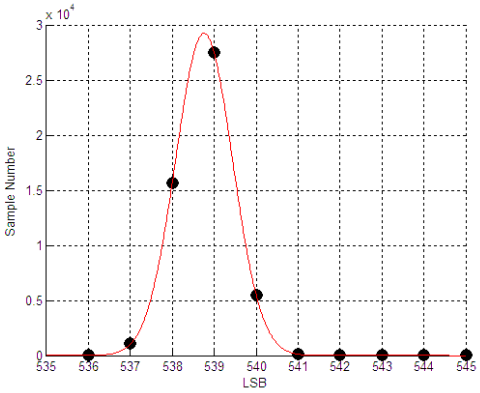
In this chapter a great amount of data is collected to test and verify the performance of the time measurement system including its accuracy and linearity. In the linearity test a new deterministic method which use two correlated waveform to scan the test range of a time measurement system is illustrated. In the last section the performance of the CFD is tested. Based on the test data, the designed time measurement system shows obvious advantages when compared to other types of time interval measurement solutions.

5.1. Accuracy Test

The concept of accuracy is usually confused with the resolution in the specification of a measurement device. The resolution of TDC is determined by the number of bits it uses to digitize an input time interval. For a 16-bit TDC the total voltage range is represented by 2^{16} discrete digital values or output codes. Therefore the absolute minimum level that a system can measure is represented by 1 bit or 1/65536 of the FSR. The accuracy of the time digital converter determines the distribution of the digital output for a given time interval. It is possible that a measurement device supports a very high resolution but its output involves more uncertainty which results in a low accuracy.

The accuracy specification of 8 ps one-sigma means that samples meet normal distribution with a variance of 8 ps. In order to create the distribution of the sample, huge amounts of samples with a fixed input must be collected. In order to have an enough “clean” signal, the time interval to be measured is generated by two coax cables with different length. The signal transmitted by the shorter cable will arrive at the TMU before the signal transmitted by the longer cable with a time interval proportional to their length difference. This method guarantees that the signal to be measured have the lowest noise and jitter. Then the results are fitted with the Gaussian function by the curve fitting tool in MATLAB thus the mean and variance value are calculated in a more precisely way. Every channel is tested with a short and long delay, each configuration is also test with two different temperature set point so as to test the thermal stability. 50000 samples are collected in every test. All results is shown in the Figure 5.1 to 5.4.

Results of Channel A

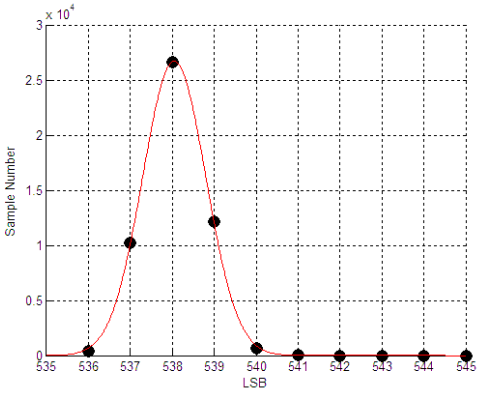


(a)

Temperature = 25

Mean = 538.8 LSB

Variance = 0.68 LSB = 8.84 ps

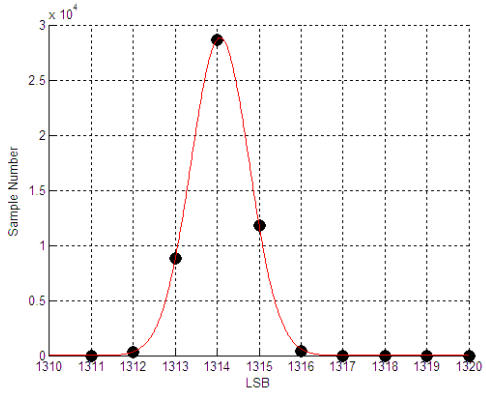


(b)

Temperature = 50

Mean = 538.0 LSB

Variance = 0.75 LSB = 9.79 ps

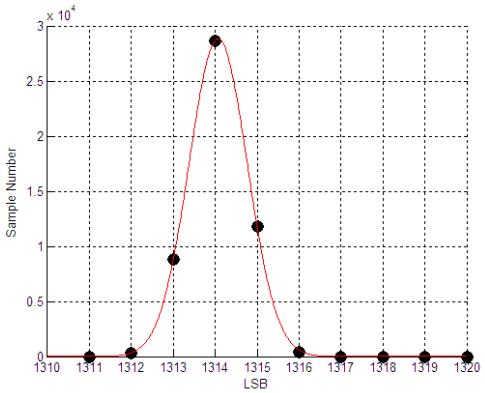


(c)

Temperature = 25

Mean = 1314.1 LSB

Variance = 0.69 LSB = 9.03 ps



(d)

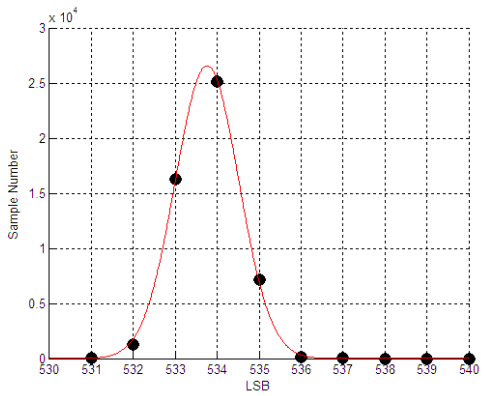
Temperature = 50

Mean = 1313.7 LSB

Variance = 0.75 LSB = 9.78 ps

Figure 5.1 (a) (b) (c) (d) Fitted distribution of Channel A

Results of Channel B

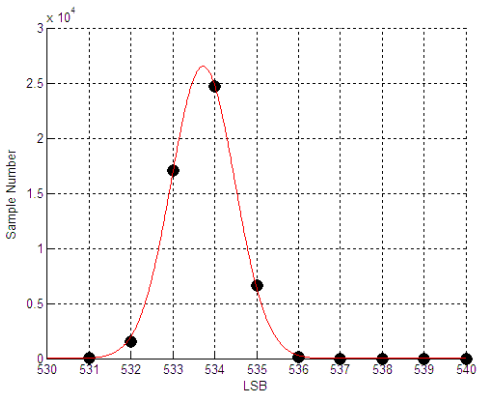


(a)

Temperature = 25

Mean = 533.8

Variance = 0.76 LSB = 9.86 ps

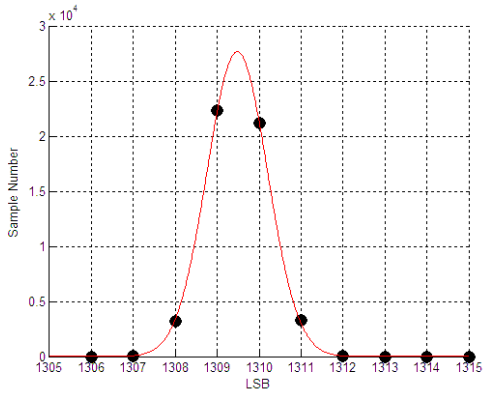


(b)

Temperature = 50

Mean = 533.7

Variance = 0.76 LSB = 9.87 ps

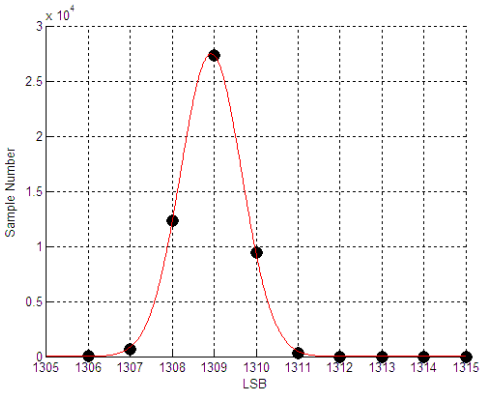


(c)

Temperature = 25

Mean = 1309.5

Variance = 0.72 LSB = 9.38 ps



(d)

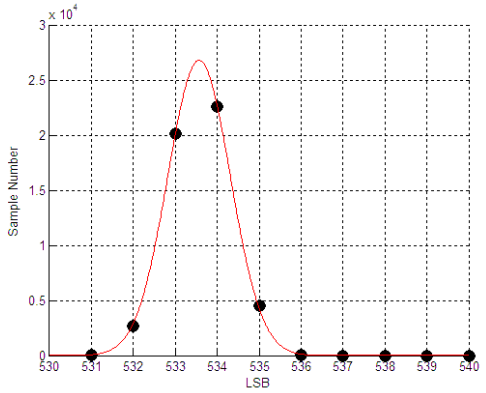
Temperature = 50

Mean = 1308.9

Variance = 0.73 LSB = 9.50 ps

Figure 5.2 (a) (b) (c) (d) Fitted distribution of Channel B

Results of Channel C

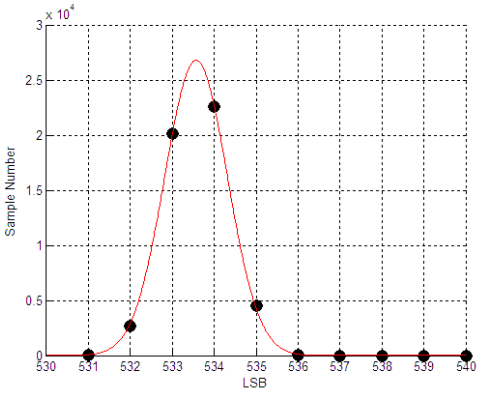


(a)

Temperature = 25

Mean = 533.6

Variance = 0.75 LSB = 9.72 ps

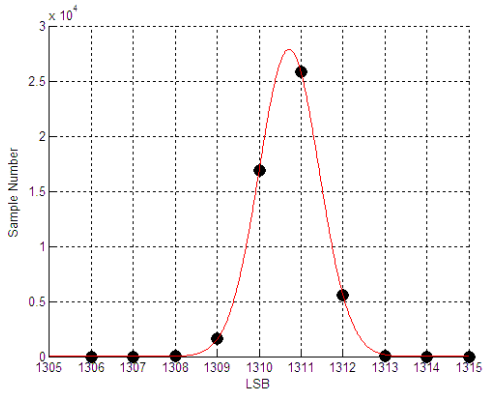


(b)

Temperature = 50

Mean = 534.5

Variance = 0.80 LSB = 10.37 ps

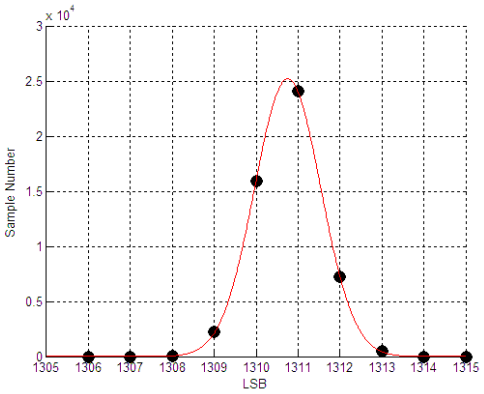


(c)

Temperature = 25

Mean = 1310.7

Variance = 0.72 LSB = 9.30 ps



(d)

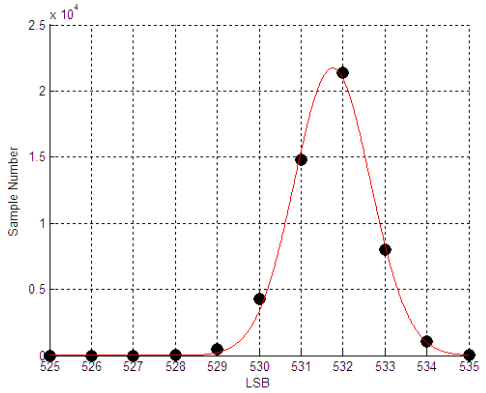
Temperature = 50

Mean = 1310.8

Variance = 0.79 LSB = 10.26 ps

Figure 5.3 (a) (b) (c) (d) Fitted distribution of Channel C

Results of Channel D

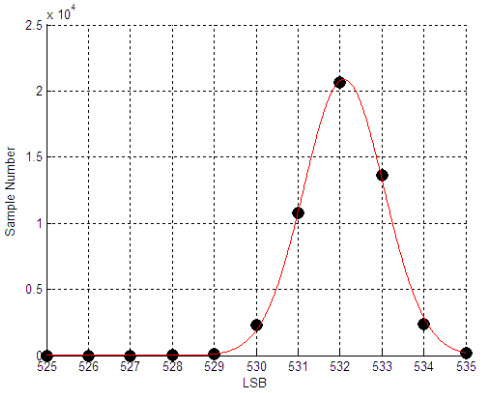


(a)

Temperature = 25

Mean = 531.8

Variance = 0.91 LSB = 11.78 ps

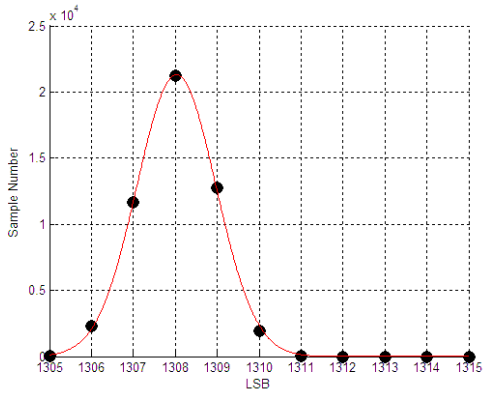


(b)

Temperature = 50

Mean = 532.1

Variance = 0.96 LSB = 12.43 ps

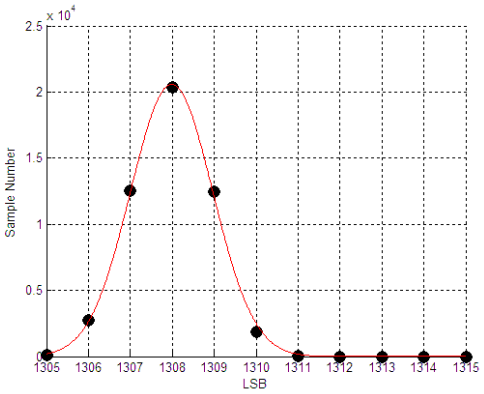


(c)

Temperature = 25

Mean = 1308.0

Variance = 0.94 LSB = 12.23 ps



(d)

Temperature = 50

Mean = 1308.0

Variance = 0.98 LSB = 12.75 ps

Figure 5.4 (a) (b) (c) (d) Fitted distribution of Channel D

Data Analysis and Conclusion

The test result is summarized in Table 5.1 for analyzing.

Channel	Type	Temperature	Mean	Accuracy
Channel A	Short	25	538.8	8.84
Channel A	Short	50	538.0	9.79
Channel A	Long	25	1314.1	9.03
Channel A	Long	50	1313.7	9.78
Channel B	Short	25	533.8	9.86
Channel B	Short	50	533.7	9.87
Channel B	Long	25	1309.5	9.38
Channel B	Long	50	1308.9	9.50
Channel C	Short	25	533.6	9.72
Channel C	Short	50	534.5	10.37
Channel C	Long	25	1310.7	9.30
Channel C	Long	50	1310.8	10.26

Channel D	Short	25	531.8	11.78
Channel D	Short	50	532.0	12.16
Channel D	Long	25	1308.0	12.23
Channel D	Long	50	1308.5	12.04

Table 5.1 Results of accuracy test

The test shows the importance of the thermal control for this time measurement system. When the temperature is set to a higher value, the accuracy of every channel is decreased.

Although the input trace is perfectly matched, there are still some offset between each channels. It should be caused by the internal structure of the THS788 since both in the short cable test and long cable test the mismatch between each two channels is the same which can be easily removed by calibration.

From the table it can be found that the accuracy of the channel D is 3 ps lower than other three channels. The cause may explained by the PCB design. In the PCB layout, the signal trace of channel D is the closest one to the CDCE62002 clock generator. The layout is shown in Figure 5.5. There are some unexpected cross talk between the clock and signal trace of channel D. Other than the channel D, the rest three channel can achieve the accuracy no more than 10 ps.

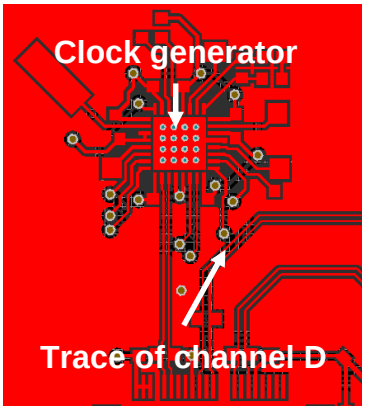


Figure 5.5 layout around trace of channel D

5.2. Linearity Test

Like an ADC the time measurement system also suffers from the non-linearity issue which makes its output drift away from perfect value. The linearity, which is usually expressed in Differential Non-Linearity (DNL) and Integral Non-linearity (INL), is an essential specification of the time measurement system.

In ideal situation, all width of slots in the FSR of a TDC should be the same, which means all bins owns the same probability to be selected. But in practice due to many factors some slots are bigger than others and create a deviation from the ideal time step size. This non-ideality is expressed by DNL specification [13]. In ideal the probability of a code being selected in the FSR is:

$$P_{code}^{ideal} = \frac{1}{total\ bins}$$

This probability can be estimated in a straightforward way by collecting huge amounts of samples and calculated the ration between the count of occurrence for each code and the total sample number. The estimated probability approaches the ideal probability as the samples number goes to infinity.

The estimated probability of occurrence for a code for an "ideal" converter is therefore:

$$\hat{P}_{code}^{ideal}[n] = \frac{Ideal\_Counts[n]}{total\ samples}$$

Then the DNL can be expressed in the normalized derivation between the ideal probability and actual probability:

$$DNL[n] = \frac{\hat{P}_{code}^{actual}[n] - \hat{P}_{code}^{ideal}[n]}{\hat{P}_{code}^{ideal}[n]} = \frac{measured\_Counts[n] - Ideal\_Counts[n]}{Ideal\_Counts[n]}$$

DNL errors can accumulate over a series of codes and cause a total deviation from the ideal transfer curve. All the DNL errors from the previous codes contribute to

a total deviation from the ideal curve at code n. This accumulated DNL errors form the Integral Non-linearity or INL, which can be calculated as:

$$INL[n] = \sum_{k=1}^{n} DNL[k]$$

In the linearity test of an ADC, which make an ADC constantly measure a triangular wave from 0 to FSR and check the histogram, which should be a flat platform in ideal situation. But when testing a Time Measurement Unit, it is not possible to apply a continuous triangular wave the same as the one in the test of an ADC since the time interval measurement is discrete by itself. Usually a stochastic testing method [14], which is also called Code Density Test, is used to perform this experiment by generating evenly distributed random time interval, when the amount of measurement is enough large, the measurement should evenly distributed in all bins of the TMU. But there are several disadvantages of this method. First it is not easy to find a random waveform generator that support a resolution smaller than the resolution of a TMU, which in this case is 13 ps. If the resolution of the signal to be measured is higher than the TMU, not all the bins of the TMU is tested, which means that the linearity test is performed at the resolution level of the waveform generator.

In the test a deterministic method [15] is used. The principle of this method is that, if we consider two correlated square wave with a constant period difference Δt, at certain instance the two corresponding edges, for example two leading edges, have a time difference T and the following edge pairs would have a difference $T + \Delta t$, $T + 2 \cdot \Delta t, \ldots$, $T + k \cdot \Delta t$. If we choose one of them as START and the other one as STOP event, the TMU will generate a time stamp series with the increment of Δt in ideal situation. It can be easily concluded that all bins of the TMU in the test range are test if the Δt is set to a number smaller than the resolution of the TMU. Through limiting the total number of measurement, which should be the integer multiples of total number of bins in the test range, it can be assured that all bins in the test range have the same chance to be tested. The test range is limited by the smallest period of two square wave. By using this method, the resolution and test range of the experiment can be determinate precisely but the demanding of waveform generator is very low,

which needs only two square wave generators with enough high frequency resolution.

Frequency calculation

Due to the limitation, the least 16 bit of 40 bit is tested. As what has introduced before, it is necessary to extend the period of square wave in order to leave some space, which is at least 200ns in the waveform so as to avoid the START event of next edge influence the current measurement in the FIFO. Then the measurement out of the test range is eliminated the FPGA.

$$T_1 > 13ps \cdot 2^{16} + 200ns \approx 1.1\mu s$$

$$f_1 = \frac{1}{T_1} < 949KHz$$

In order to have a better result in the calculation of f_2, the f_1 is chosen as 500KHz. Since the resolution of TMU is 8ps, the increment step is chosen as half LSB.

$$\frac{1}{f_2} - \frac{1}{f_1} = \frac{LSB}{2}$$

Where the LSB of TMU can be expressed precisely as:

$$LSB = \frac{1}{1.2 \times 10^9 \times 2^6}$$

The f_2 results in about 499998.3724Hz, which is rounded to the effective number of the frequency supported by the Waveform Generator to be used.

Alignment of two waveform generators

Due to the limitation of available instruments in the lab, two identical on channel waveform generators are used to generate the required correlated waveforms. The correlation of two waveforms must be strictly hold therefore all bins in the test range is assigned with equal chance to be accessed. The correlation can be verified by changing the two waveform generators into the same desired frequency and check the output in the oscilloscope. In perfect situation the outputs should have the same

frequency and synchronized but in practice it is difficult to ensure their outputs are fully correlated since two waveform generator use their own frequency generator, which creates frequency offsets and random drift in the output and makes the increment step drift away from perfect value. In order to solve this problem, a cable is used to connect the reference frequency output of a waveform generator to the reference frequency input of the other one so as to make reference frequency signal fully synchronized and cancel all the uncorrelated influence.

In next step it is necessary to align the two waveform generator to the desired increment step as close as possible. The alignment is performed by measure the slop of ramp of a series of samples. The synchronous sample reading feature is used to collect a series of samples and the data is imported into MATLAB. Then the curve fitting tool is used to find out the slop. If the slop is far away from 0.5, it is necessary to tune the frequency of one of two waveform generators until the slop is enough close to 0.5. In the test the f_2 is changed to 499998.3726Hz so as to achieve a perfect 0.5 ramp of slop, which is 0.0002Hz from the perfect value. This can be explained by the imperfect reference clock frequency for the waveform generator. The verification results is shown in figure 5.6.

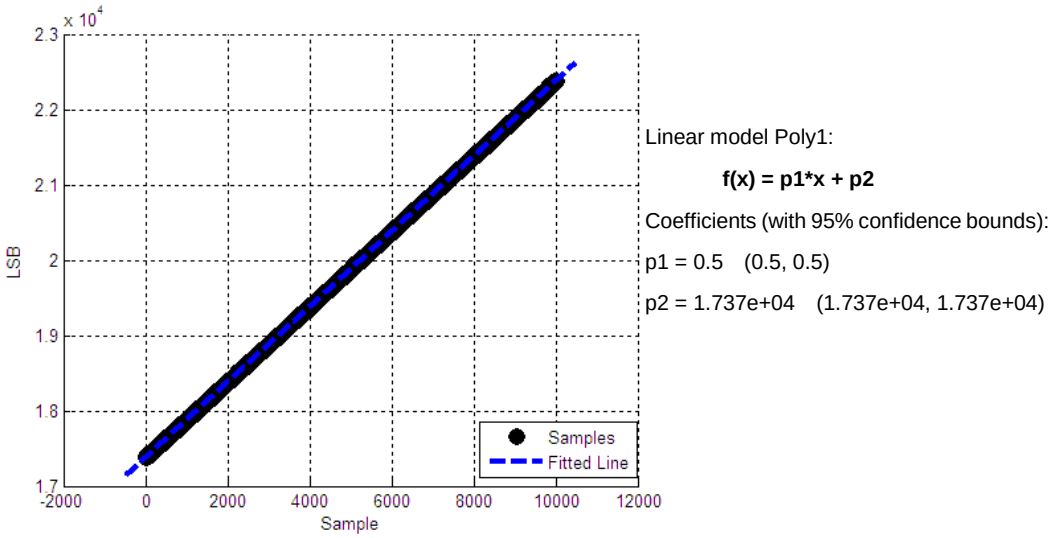


Figure 5.6 Fitted result for alignment

Test Result

Since the dynamic range of this TMU is very high, it will take several days or weeks to collect enough samples to construct the histogram if all the bins want to be covered during the test. The 2^{16} bins in LSB are chosen to be test to evaluate the linearity specification of the TMU which only needs several hours to collect the enough samples. Every histograms were generated inside the FPGA with 655360000 of samples by using the histogram generation feature, which assigns each data bin of TMU with 10000 samples in ideal situation. Then the result are imported into MATLAB for the calculation of DNL and INL. The raw test results of each channel are shown in figure 5.7 (a) ~ (d).

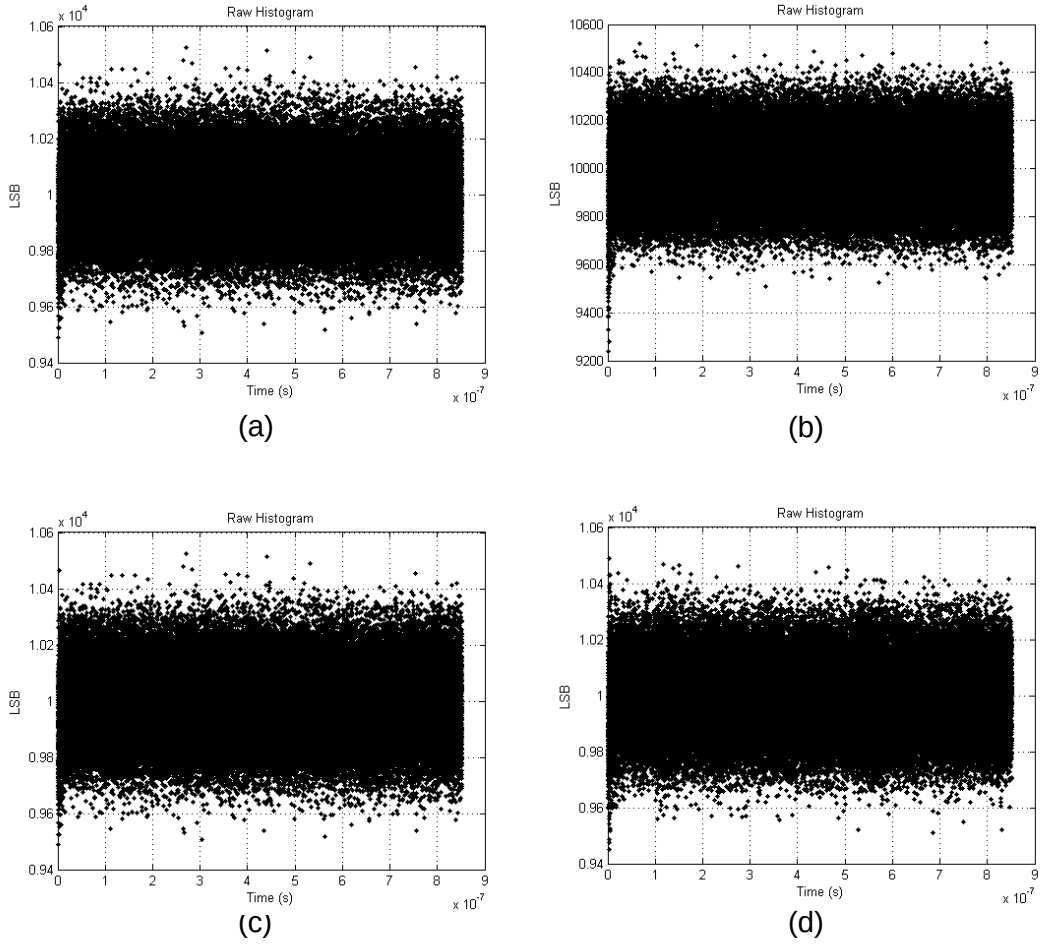
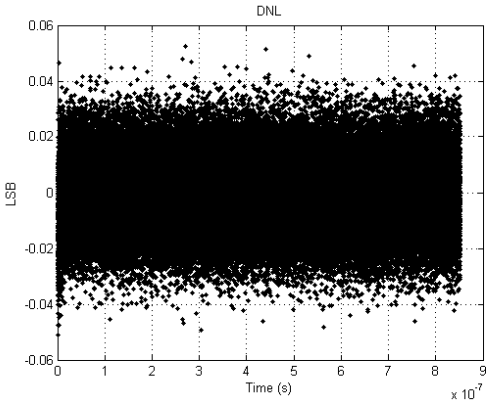


Figure 5.7 (a) (b) (c) (d) Raw histogram data of Channel A to D

The DNL results of each channel are calculated with the formula introduced before, which is shown in Figure 5.8 (a) to (d):

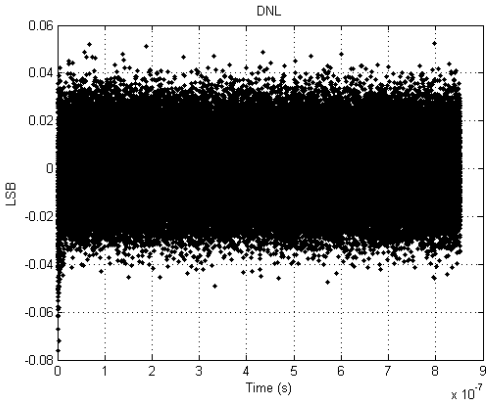


(a)

Maximum DNL = 0.053

Minimum DNL = -0.051

$\sigma_{DNL} = 1.3\%\ \mathrm{LSB}_{RMS}$

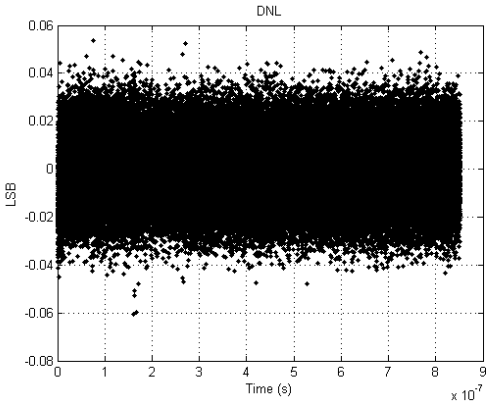


(b)

Maximum DNL = 0.052

Minimum DNL = -0.076

$\sigma_{DNL} = 1.5\%\ \mathrm{LSB}_{RMS}$

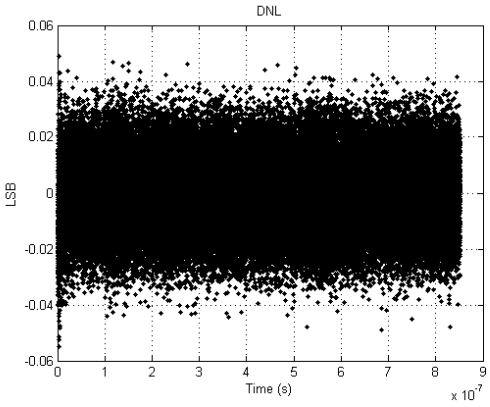


(c)

Maximum DNL = 0.053

Minimum DNL = -0.060

$\sigma_{DNL} = 1.4\%\ \mathrm{LSB}_{RMS}$



(d)

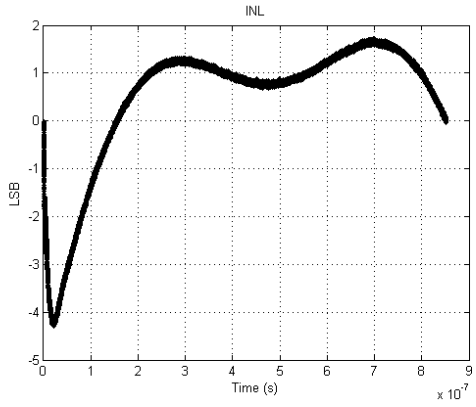
Maximum DNL = 0.053

Minimum DNL = -0.051

$\sigma_{DNL} = 1.3\%\ \mathrm{LSB}_{RMS}$

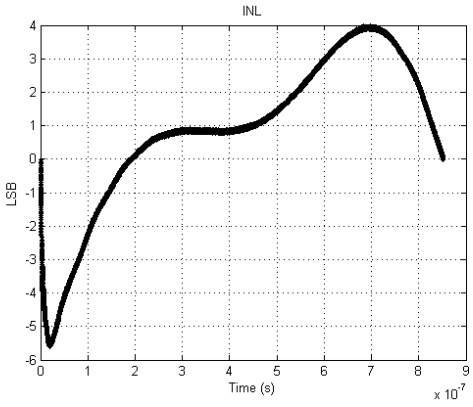
Figure 5.8 (a) (b) (c) (d) DNL results of Channel A to D

Then the INL is calculated by accumulating the DNL data. The results of each channel is shown in Figure 5.9 (a) ~ (d).



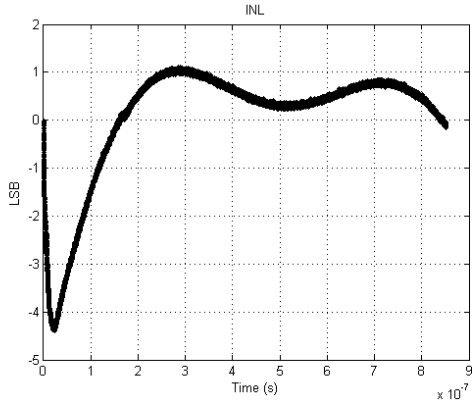
(a)

Maximum INL = 1.716
Minimum INL = -4.291



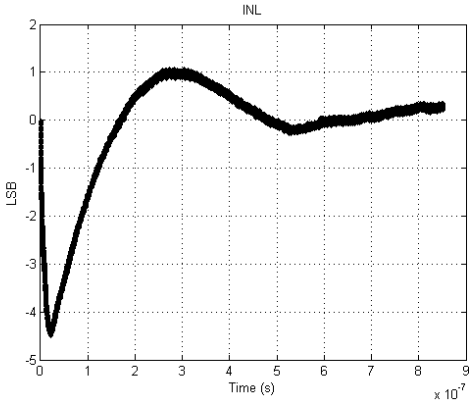
(b)

Maximum INL = 3.982
Minimum INL = -5.592



(c)

Maximum INL = 1.118
Minimum INL = -4.389



(d)

Maximum INL = 1.074
Minimum INL = -4.493

Figure 5.9 (a) (b) (c) (d) INL results of Channel A to D

Data Analysis and Conclusion

In the linearity test the TMU shows an excellent DNL performance in the test. It have a RMS of 1.5% LSB. While the INL of the TMU was significantly high in the start part of the range, which means that the in this range much less samples are recorded. Respect to the principle of the TMU, the internal counter could do nothing with the INL since when the STOP signal arrives, the state of the counter is totally random. The cause may relays in the data latching mechanism of the TMU, when STOP signal arrives closed to the START signal, the internal bus may busy processing the START signal and the closed arrived STOP signal is delayed. Since this is a commercial TMU whose internal architecture is unknown, it is not possible to find out which part of the TMU causes this problem. But this problem can be resolved by the data calibration by using the INL data.

5.3. CFD Test

The purpose of the CFD test is evaluating its capability of compensating the time walk caused by amplitude variation. In the test the same techniques as what is in accuracy test which use two cables with different length to create a lowest noise time interval signal but the amplitude of the signal will be changed during the test so as to obtain the relationship between time walk and signal amplitude.

Figure 5.10 The DT5800D digital detector emulator

The DT5800D digital detector emulator (Figure 5.10) was chosen as the signal generator. It is a powerful instrument purposed for emulating the signal from real detectors in order to avoid using a real detector during development. It support a wide range of feature such as energy spectrum emulation and Poisson distribution pulse generation. In this test the main reason of choosing this advanced instrument is that the signal shape of waveform generator must not change or change as little as possible during the variation of amplitude to eliminate the time walk caused by signal shape change while the available waveform generator in our lab gives out a bad result. The signal shape is chosen as a commonly seen RC-CR signal whose shape, raising and falling time can be independently set inside the GUI interface on PC.

Two tests are performed to verify performance of CFD. The first signal for test is set to a raising time at 50 ns and falling time at 5us. Thus the parameters for capacitors and resisters in delay line were chosen as 600 ohm and 10 pF, which resulted in a

total delay of 36ns. During the test the amplitude started from 2.1V and it was decreased at a step of 200mv until 100mv. The result is shown in figure 5.11.

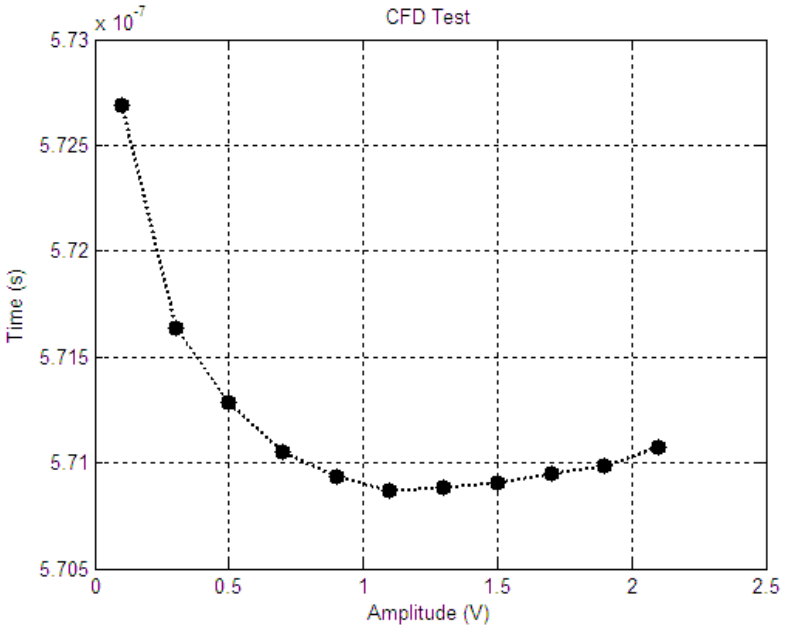


Figure 5.11 Result of first test

In the second test the raising time of the signal is set to 20 ns in order to see the responses of the CFD in high speed signal. The parameters for resisters and capacitors in delay line are changed to 60ohms and 30pf to create a 10.8 ns delay. The result is shown below:

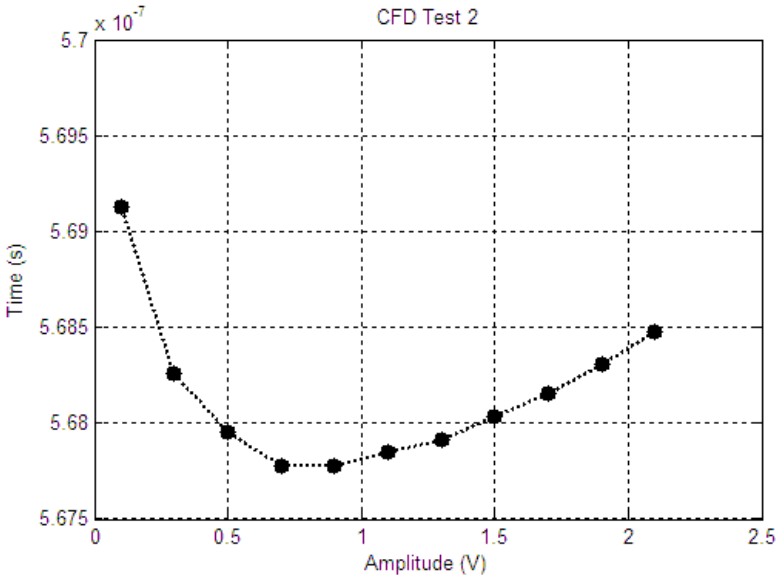


Figure 5.12 Result of second test

For a better comparison, the two results are plotted in the same figure as what is shown below:

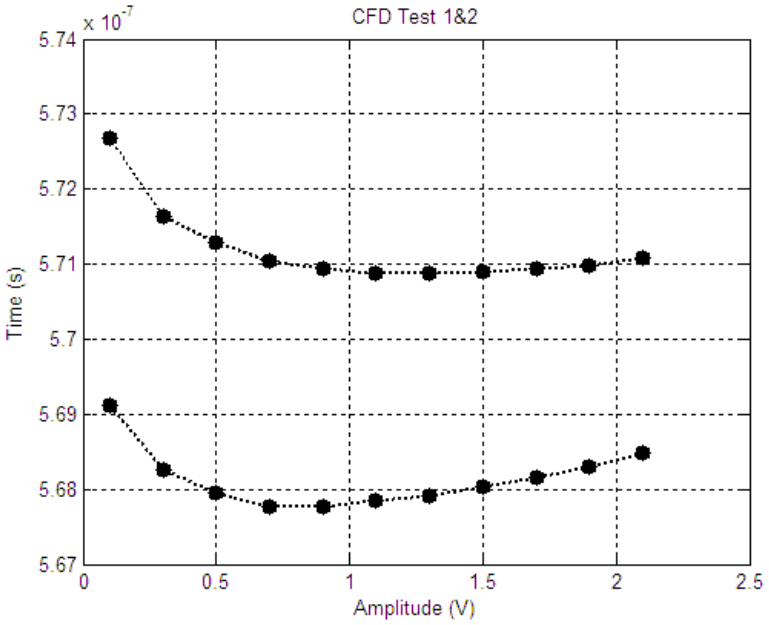


Figure 5.13 Result of both first and second test

Data Analysis and Conclusion

In the first test, a lower speed signal is applied. When the amplitude is in the range of 1~2V this CFD can keep the time walk lease than 15 LSB, which is around 0.2ns. But when the amplitude keep decreasing, the time walk increased significantly. This is because the time walk will increase as the decrease of amplitude if there is offset voltage in the zero-crossing detector, which is bring into the system by the hysteresis feature of the comparator.

$$t_z = \frac{1}{\mathrm{A}} \cdot \frac{V_{offset}}{\left.\frac{dV}{dt}\right|_{t=t_z}}$$

In order to prove this assume, in the second test a signal with faster raising time was used. As a result, the ramp of slop changes its sign from negative to positive. This means that, there are two factors influencing in the CFD circuit, one of the factor is the offset voltage which will cause the trigger time increase as amplitude decrease. While the second factor may relays in the signal shape change caused by imperfectness of the signal generator, which decrease the trigger time as amplitude decrease. In the first test, it can be concluded that these effects are well compensated with each other accidently. Once the signal slop is increased, which decrease the voltage offset caused time walk, the influence caused by signal shape change comes into effect and changes the sign of the ramp.

The test proves the analysis in the chapter 3, the CFD is very sensitive to the quality of input signal, which is the signal shape and baseline offset. But in practice the output of a detector is not perfect. If the specification of time walk less than 1 LSB, the imperfectness of input signal must be corrected. It is necessary to implement signal shaper and baseline restorer before the CFD to form the incoming signal into a perfect shape. In summary, as the analog front-end a CFD is not enough. The analog front-end of the time measurement system needs more study.

5.4. Conclusion

This thesis describe the development of a time measurement system based on a commercial time measurement unit. The test verified that this architecture of time measurement system can offer a resolution of 13ps with accuracy no more than 10ps and 1.5% LSB DNL_{RMS} of linearity performance. In recent publication [16] the similar specification is achieved by an ASIC TDC. When comparing their development cost, ease of use and range of applications, the designed time measurement system shows obvious advantages. The cost of ASIC fabrication is very high while in traditional time measurement solution they have to change their architecture and hardware design to adopt themselves to a specified applications, which offers little degrees of freedom. Although a TDC can be implemented inside a FPGA which use not much of hardware resource and achieve a very high resolution, the linearity is always a big issue due to the architecture of the FPGA, which is at the level of 10% LSB DNL or even more. Moreover the software architecture of this system has been deigned in a way that new application oriented function can be easily integrated into the system. This high dynamic, high precision time measurement system can be seen as a general purposed time interval meter which can be adopt to almost every time interval measurement applications.

6. Reference

[1] Lombard, Michael A. "Fundamentals of time and frequency." The Mechatronics Handbook, -2 Volume Set (2002).

[2] Kostamovaara, Juha, Sami Kurtti, and Jussi-Pekka Jansson. "A receiver–TDC chip set for accurate pulsed time-of-flight laser ranging."

[3] Edinburgh Photonics, "What is TCSPC?" Edinburgh Instruments Ltd, 2012.

[4] Crotti, M.; Rech, I.; Ghioni, M., "Monolithic Time-to-Amplitude converter for TCSPC applications with 45 ps time resolution," Ph.D. Research in Microelectronics and Electronics (PRIME), 2011.

[5] Becker, Wolfgang. "The bh TCSPC handbook." Becker & Hickl GmbH, (2008).

[6] Koch, K.; Hardel, H.; Schulze, R.; Badura, E.; Hoffmann, J., "A new TAC based multi channel front-end electronics for TOF experiments with very high time resolution," Nuclear Science Symposium Conference Record, 2004.

[7] Dudek, Piotr, Stanislaw Szczepanski, and John V. Hatfield. "A high-resolution CMOS time-to-digital converter utilizing a Vernier delay line." Solid-State Circuits, IEEE Journal of 35.2 (2000): 240-247.

[8] Manjeshwar, Ravindra M., Yiping Shao, and Floris P. Jansen. "Image quality improvements with time-of-flight positron emission tomography for molecular imaging." Acoustics, Speech, and Signal Processing, 2005. Proceedings.(ICASSP'05). IEEE International Conference on. Vol. 5. IEEE, 2005.

[9] Spieler, Helmuth. "Fast timing methods for semiconductor detectors." Nuclear Science, IEEE Transactions on 29.3 (1982): 1142-1158.

[10] Goldie, John. "An Overview of LVDS Technology." (1998).

[11] Garbolino, S., S. Martoiu, and A. Rivetti. "Implementation of Constant-Fraction-Discriminators (CFD) in sub-micron CMOS technologies." Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), 2011 IEEE. IEEE, 2011.

[12] Binkley, David M. "Optimization of scintillation-detector timing systems using Monte Carlo analysis." Nuclear Science, IEEE Transactions on 41.1 (1994): 386-393.

[13] Kuyel, Turker. "Linearity testing issues of analog to digital converters." Test Conference, 1999. Proceedings. International. IEEE, 1999.

[14] Samarah, Amer, and Anthony Chan Carusone. "A digital phase-locked loop with calibrated coarse and stochastic fine TDC." Solid-State Circuits, IEEE Journal of 48.8 (2013): 1829-1841.

[15] Yousif, Abdel S., and James W. Haslett. "A fine resolution TDC architecture for next generation PET imaging." Nuclear Science, IEEE Transactions on 54.5 (2007): 1574-1582.

[16] Tamborini, D., et al. "TDC with 1.5% DNL based on a single-stage vernier delay-loop fine interpolation." Time-to-Digital Converters (NoMe TDC), 2013 IEEE Nordic-Mediterranean Workshop on. IEEE, 2013.