UniversitàdelSannio-Benevento

LaboratoriodiElaborazionedeiSegnaliedelleInformazionidiMisura
Dipartimento di Ingegneria – Università del
Sannio 82100, Benevento

L.E.S.I.M.

N° 9175.USSN/2

IMPLEMENTATION AND TESTING OF A PHASE MEASUREMENT SYSTEM FOR SINE WAVE GENERATOR

María Cristina Rodríguez Álvarez.

Tutor U. Oviedo: Luis José Rodríguez Muñiz

Tutor U. of Sannio: Luca de Vito

RESUMEN
La influencia del sistema de medición en el mesurando tiene una importancia real. Puede haber algunos errores o efectos en la medida. Se querrá conseguir la mejor solución para obtener una función de transferencia sin efectos.
Este trabajo será sobre la medición de la fase de la onda sinusoidal, estudiando cómo es la respuesta en los rangos que van desde unos pocos mHz hasta 10MHz. En todo este rango será probando la estructura del prototipo que se está estudiando durante este informe. Las soluciones se comparan con la respuesta esperada.
El siguiente informe está organizado con una primera sección, la introducción, luego en el segundo paso se explica la estructura del sistema y cómo es el trabajo de cada aparato usado. En la tercera parte del escrito se observan los resultados donde serán comparados con los resultados esperados.

Los efectos del sistema de medición se eliminarán de las formas de onda obtenidas porque será necesario conocer la función de transferencia de los sistemas de medición.
El sistema de medición, su influencia, tiene gran importancia en la medida y por ello tiene que ser controlado y compensado. Se modelan como una caja negra y mediante un proceso de deconvolución (corresponde a una división de la magnitud del espectro de forma de onda por la respuesta de magnitud del sistema de medición y una sustracción de la forma de onda espectro de fase con el sistema de medición de la respuesta de fase) se describe la función de transferencia. Dos técnicas para la obtención de la función:
Frecuencia de barrido (swept frequency)
Método de pulso (pulse method)
El método de barrido de frecuencia se usará para medir la respuesta de fase de un sistema de medición. El sistema mide la señal de amplitud total en diferentes momentos generando una serie de pulsos.
Cada pulso tiene un retraso elegido por el operador, por lo que para cambiar la demora será posible medir la amplitud en un tiempo diverso. Para ello se necesita trabajar con un generador de retraso y selector de pulso.
El objetivo de este proyecto es implementar un sistema para medir la fase de las señales de onda sinusoidal.
La idea principal de este informe es medir una señal usando diferentes medidas de amplitud que están sincronizados con la señal de origen. Para medir la amplitud en diferentes momentos del tiempo, se considera una cantidad de trenes de impulsos y cada tren de pulsos tiene un retraso seleccionado por el operador.
El tren de pulsos se sincronizará con la fuente de señal, que tiene una tasa de repetición (la repetición no debe ser más de un pulso por ciclo de la onda sinusoidal). Para cada pulso, mide la amplitud en cada momento.
Para obtener este procedimiento de medición será necesario un generador de retraso y un selector de pulsos para generar los trenes de impulsos.
El sistema de medición de fase propuesto tiene como propósito medir la fase del seno onda suministrada por generadores de señal. Para implementar este método, serán necesarios:
El generador de señal bajo prueba, el generador de tren de pulsos, el generador de retardo, la muestra y el voltímetro.
[Los modelos matemáticos básicos para este trabajo están desarrollados en el punto 1.1.2]
Para la evaluación del rendimiento teórico del sistema de medición se usará el análisis de Monte Carlo, el cual considera el efecto de la fluctuación aleatoria que afecta a la señal del tren de pulsos y al ruido blanco Gaussiano aditivo que afecta a las señales de onda y al tren de pulso.
Este análisis consiste en que, mediante una simulación matemática compleja, aproxima el resultado de cálculos de los que no se puede obtener una solución exacta. Por ello se utiliza para la estimación con parámetros que muestran variabilidad.
Se tiene en cuenta todos los posibles escenarios que nos podemos encontrar en las señales que estamos evaluando donde se consiguen la estimación más aproximada de los resultados que obtendremos en la realidad.

Sistema de medición dentro del rango de 100mHz a 10MHz
El generador de señal que proporciona la onda sinusoidal de referencia y la señal de sincronización de 10MHz por Tektronix AWG 420. El reloj de referencia que se utiliza en este proyecto es de 10MHz.
Del generador de impulsos se obtiene el pulso para muestrear la onda sinusoidal (tanto la frecuencia, ancho y retardo serán seleccionados). FPGA (Field Programmable Gate Array) suministrará la fuente de impulsos y Agilent E8663B el encargado del generador de retraso.
FPGA es un dispositivo programable por bloques lógicos configurables. Es impulsado por un microcontrolador.
El generador de retardo es un equipo electrónico de prueba. Se utiliza para sincronizar eventos.
La muestra (Sample & Hold) es un dispositivo analógico que muestrea el voltaje de una señal. Su rendimiento es acumular la carga eléctrica en un condensador.
El voltímetro proporciona una medición de voltaje proporcional a la energía.
El funcionamiento de la arquitectura: la señal sinusoidal cambiará en un tren de pulsos, con un conocido ancho y frecuencia por la forma del pulso y el selector donde será posible elegir el número de pulsos por periodo que se enviará al láser.

El generador de impulsos tiene la intención de ofrecer un tren de pulsos sincronizado con la señal de referencia 10MHz. Cada pulso define un instante de la muestra. El tren de pulsos se genera para muestrear la onda sinusoidal. La señal r(t) es la salida de señal que recibe el generador de impulso y será la entrada en el generador de retraso.
Con todas las especificaciones que tendremos será posible crear la función con la que implementar el programa en el Xilinx con el fin de conseguir que el FPGA funcione como lo necesitamos para la obtención de la onda sinusoidal y el tren de pulsos.
El contador controlará las entradas de tal manera que
-Elegirá 16 bits (con esos 16 bit será posible un 100kHz).
-El ancho del pulso seleccionado de 1bit.
El microcontrolador envía el valor elegido en el contador. La información que se recibe en el FPGA cambiará en un tren de pulsos, con un ancho y frecuencia conocidos por la configuración del pulso.

El Tektronix AWG 420 se ha utilizado para proporcionar una señal sinusoidal de 10 MHz al generador de pulso. La secuencia de bits es la representación de la secuencia de acuerda con el período de tren de pulsos y la posición de tiempo del pulso. Las resoluciones de ancho de pulso y retardo de tiempo corresponden al periodo de reloj AWG.
El generador de señales funciona después de los siguientes estados:
Estado de secuencia de espera (espera a que se cargue una nueva secuencia en la memoria).
Estado de disparo (espera el evento para generar la secuencia de bits cargados).
Estado de generación de tren de pulsos (AWG genera el tren de pulsos continuamente hasta la parada el comando se recibe de la interfaz GPIB).
AWG420 es un generador de forma de onda arbitraria programable con 200 MS / s, DAC de 16 bits, dos canales arbitrarios, generador de forma de onda.
Muestra y retención: esta arquitectura se basa en seguir y mantener. Elemento utilizado: OPA615. Es un instrumento que se utiliza para la restauración rápida y precisa de DC, compensación, desalineación y supresión de la baja frecuencia de amplificadores de banda ancha.
Voltímetro: proporciona una medición de voltaje proporcional a la muestra salida. El voltímetro debería funcionar en DC o AC. Para mediciones de AC, la salida del muestreador es asociado con el rms. Un multímetro aumenta la velocidad de lectura y aumenta el rendimiento de la prueba. El tiempo de integración es el período de tiempo que el convertidor A / D mide la señal de entrada. Por las mediciones de CC o ohmios, el tiempo de integración determina la velocidad de medición, la precisión, dígitos máximos de resolución y la cantidad de rechazo de ruido en modo normal.

Software: para programar y organizar los instrumentos de la manera que funciona como fue diseñado. MATLAB. El uso de Matlab conectará los dispositivos y enviará la información que necesita para la obtención de los resultados (código en Anexo). Todas las funciones están conectadas entre sí para conseguir que todo funcione tan bien como sea posible.

Resultados. Pruebas del sistema para varias frecuencias.
Se tendrán resultados para diferentes frecuencias: 100Hz, 10kHz, 100kHz, 1MHz, 10MHz. Con los resultados que se obtiene de cada test, su representación en los gráficos nos deja ver que las mediciones para los valores ‘intermedios’ de la frecuencia se obtiene una solución parecida al resultado esperado mientras que en las frecuencias de 100 y 10M Hz están siendo más afectados por el sistema.
Por lo que se concluirá que habrá que hacer cambios. Cambiar el FPGA, un contador que pueda trabajar a menor frecuencia. Cambiar la generación del pulso con una mejor resolución para poder trabajar en una amplia gama de frecuencias.

TABLE OF CONTENT

ABSTRACT

The influence of the measurement system on the measurand has a really importance. It can make some effects or errors on the measure. Finding the best solution for obtained the Transfer Function without effects.

This work will be about a measuring the phase of sine wave. It studies how is the response in the range of a few mHz up to 10MHz. On all this range it will be tested the particular structure architecture, is the prototype that it is studying during this report. The solutions will be compared with the response that was expected.

This report is organized with a first section, the introduction and after that was the second step where it is explained the structure of the system and how is work any part of it is formed and the codes of every instrument. In the third part it is written about the results where it will compared with the expected. In the last part is finished with a conclusion.

1 INTRODUCTION

The main purpose of this project consists to implement an automatic phase measurement system for sine wave.
Our research is organized as follows:

I: The implementation of a first prototype for the system that should work within the frequency range of 100 Hz up to 10 MHz.

II: The testing of the implemented prototype and the evaluation the repeatability and the systematic effects on the obtained phase measurements.

III: The comparison of the obtained results with a mathematical model, describing the entire system.

IV: The comparison of the system with the ones already available in literature.

1.1 DESCRIPTION

The effects of the measurement system will be remove from the wave forms obtained. For that, it will be necessary knows the Transfer Function of the measurement systems. By means of a deconvolution process it will be possible obtain the Transfer Function taking into consideration an uncertainty limit.
The measurement system, its influence, has great importance on the measured, for that reason, it has to be control and compensated for.
Each measurement system can be modeled as a black box (the input is the measurand while the output is the measured value).
When the transfer function is described by the box, the evaluation of the measurand is possible using the measured values and take out the influence of the measurement instrument (respecting the uncertainty limit). This will possible performing by a deconvolution process.
The deconvolution process corresponds to a division of the waveform spectrum magnitude by the magnitude response of the measurement system and to a subtraction of the waveform phase spectrum with the measurement system of the phase response.
There will be two techniques for obtain the Transfer Function: swept frequency (obtain a precise information about the spectral) and method and pulse method (get an estimation of the magnitude and phase system response). The frequency swept method will be used to measure the phase response of a measurement system. For using this technique, it is necessary to know the phase of the sinewave used for the frequency sweep.
The system measures the total amplitude signal in different times and so a serial group of pulses is generated. Every pulse has a delay chosen by the operator so for change the delay it

will be possible to measure the amplitude at diverse time. For this reason, it will necessary work with a delay generator and pulse selector.
The aim of this project is to implement a system for measuring the phase of sine wave signals.

The main idea of this report is measure a signal using different amplitude measurements that are synchronized with the source signal. Later, some amplitudes measures after, sine fitting methods can be executed to determinate the sine wave phase. So as to measure the it amplitude at different moments of the time, it is considered a number of pulse trains and each pulse train has a delay selected by the operator.
The pulse train will be synchronizing with the signal source and it has a repetition rate (the repetition must not be more than one pulse per cycle of the sine wave).
For each pulse, it measure the amplitude in each moment. In this way, the source signal is evaluated. In order to get this measurement procedure it will necessary a delay generator and a pulse selector for generating the pulse trains.

1.1.1 The proposed phase measurement system

The proposed phase measurement system has as purpose to measure the phase of the sine wave supplied by signal generators.
The plan is to utilize techniques for sine fit techniques according to the amplitude of sine wave that is sampled in different instants.

For implementing this method will necessary five fundamental elements (explain how is work in the point 2):
1_The signal generator under test
2_the pulse train generator
3_the delay generator
4_the sampler
5_the voltmeter

1.1.2 Mathematical model [1]

The sinusoidal signal provided by signal generator can be modeled as follows:
$y_{src}(t) = A \cdot sin(2\pi f_{src} t + \phi)$
where A, f_{src}, and Φ are the amplitude, the frequency, and the phase of the sinusoidal signal, respectively.
The reference signal synchronized with the signal generator clock is:
$r(t) = sin\left(2\pi f_r t - \theta_{src,r}\right)$
where f_r, is the frequency of the reference signal, usually 10 MHz, and $\theta_{src,r}$ is the phase delay between the sine wave and the reference signal.
The pulse train generator provides a pulse train having the same frequency of the reference signal:

[1]U. Sannio-benevento, Draft version 3 Project : 2014-NIST-MSE-01 Second year

$$p_{tg}(t) = p(t) * \sum_{n=-\infty}^{+\infty} \delta\left(t - nT_r - \frac{\theta_{src,r} + \theta_{tg}}{2\pi f_r}\right)$$

where $p(t)$ is the shape of the pulse, $T_r = {}^{1}/_{f_r}$ is the period of the generated signal, and θ_{tg} is the phase delay introduced by pulse train generator.

The signal provided by delay generator is given by:

$$p_{dg}(t, \Delta t_i) = p(t) * \sum_{n=-\infty}^{+\infty} \delta\left(t - nT_{dg} - \frac{\theta_{src,r} + \theta_{tg}}{2\pi f_r} - \frac{\theta_{dg}}{2\pi f_{dg}} - \Delta t_i\right)$$

where $T_{dg} = l \cdot T_r$, where $l \in \mathbb{N}^+$ is the pulse period of the delayed pulse train, θ_{dg} is the phase delay introduced by pulse delay generator

The sampler performs a correlation operation between the pulse train $p_{dg}(t, \Delta t_i)$, and the sine wave signal $y_{src}(t)$. For the m-th period of the pulse train signal, the sampler output can be modeled as follows:

$$y_{Sa}(\Delta t_i, m) = p(t) * h_{Sa}(t) \otimes y_{src}(t)|_{t=\Delta t_i + mT_{dg} + t'}$$

where $h_{Sa}(t)$ is the impulse response of the sampler, t' is equal to $\frac{\theta_{src,r}+\theta_{tg}}{2\pi f_r} + \frac{\theta_{dg}}{2\pi f_{dg}} + \frac{\theta_{Sa}}{2\pi f_{dg}}$, and θ_{Sa} is the phase delay introduced by the sampler.

The dc component of the signal provided by sampler is measured with a voltmeter, which provides a voltage measurement proportional to the amplitude of the sine wave at the sampling instant defined by pulse train:

$$y_v(\Delta t_i) = \frac{1}{K} \cdot \sum_{m=0}^{K-1} y_{Sa}(\Delta t_i, m)$$

where the integration time of voltmeter, t_{int}, is equal to $K \cdot T_{dg}$, $K \in \mathbb{N}^+$.

By performing voltmeter measurements at several time delays Δt_i, with $i = 0 \ldots N-1$, the following vector can be defined:

$$\bar{y}_v = [y_v(\Delta t_0) \quad .. \quad y_v(\Delta t_{N-1})]^T$$

In order to estimate the phase of the sine wave, the 3-parameters sine fitting is applied on the measured values $\bar{y}_v$.

The 3-parameters sine fitting method is based on the estimation of three coefficients describing a sine wave sampled at least in three known time instants and by knowing the sine wave frequency value, f_{SRC}. The estimated sine wave is written as follows:

$$y_{est} = S_1 \cdot \cos(2\pi f_{SRC} t) + S_2 \cdot \sin(2\pi f_{SRC} t) + S_3$$

where S_1, S_2, and S_3 are the coefficients estimated by means of sine fitting through the following formula:

$$\bar{S} = \bar{\bar{D}}^+ \cdot \bar{y}_v$$

where $\bar{\bar{D}}^+$ is the pseudo-inverse matrix of $\bar{\bar{D}}$,

$$\bar{\bar{D}} = \begin{bmatrix} cos(2\pi f_{SRC}\Delta t_0) & \sin(2\pi f_{SRC}\Delta t_0) & 1 \\ \vdots & \vdots & \vdots \\ cos(2\pi f_{SRC}\Delta t_{N-1}) & \sin(2\pi f_{SRC}\Delta t_{N-1}) & 1 \end{bmatrix}$$

From the estimated values $\bar{S}$, the phase of sine wave can be evaluated as follows:

$$\phi_{est} = atan\left(\frac{S_2}{S_1}\right)$$

1.2 MONTE CARLO ANALYSIS

In order to evaluate the theoretical performance of the proposed measurement system Monte Carlo analysis has been conducted. In particular, the Monte Carlo analysis considers the effect of random jitter affecting pulse train signal and additive White Gaussian noise affecting sine wave and pulse train signals.
The next three figure help it to make an idea about the limits of this project.

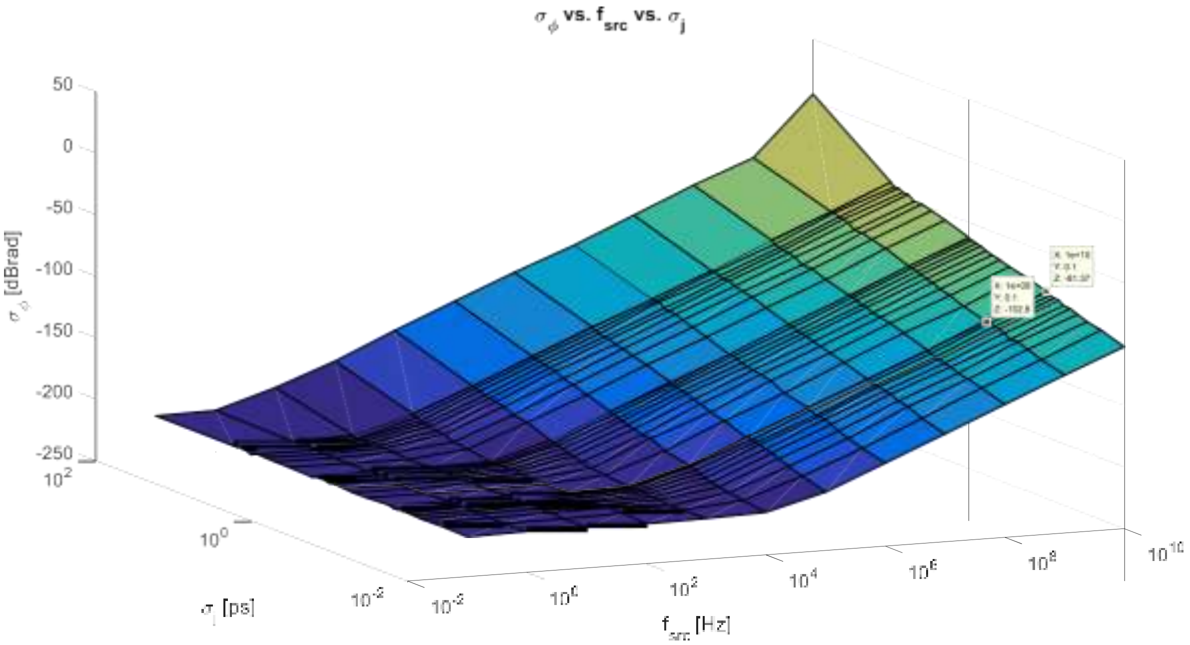

Fig. 1: Standard deviation, random jitter against frequency.

In the figure1, is shown the standard deviation between the random jitter and the frequency of the signal where it is not considered the additive White Gaussian noise.
The settings, which are using for this analysis, are written in the table1.

Sine wave amplitude	1V
Sine wave phase	0º
Number of time delays	10
K number of samples for the voltmeter measurements	1000
Number of iteration for Monte Carlo Analysis	100

table 1: Settings Monte Carlo Analysis.

For a frequency at 10 GHz the jitter: σj = 100 fs -> σφ = $10^{-3°}$
While at 100MHz there will be σj = 100 fs -> σφ = $10^{-4°}$
Of the graphic, it can obtain the value for 10 Mhz and 100 Hz of frequency. The results will be:
For 10 MHz a deviation of -79.0127.
In addition, for 100 Hz the deviation has -78.98 of value.

In the next figure (fig 2) it is possible see how, the additive White Gaussian Noise affects to the system, with the same that characteristics that are in the table1.

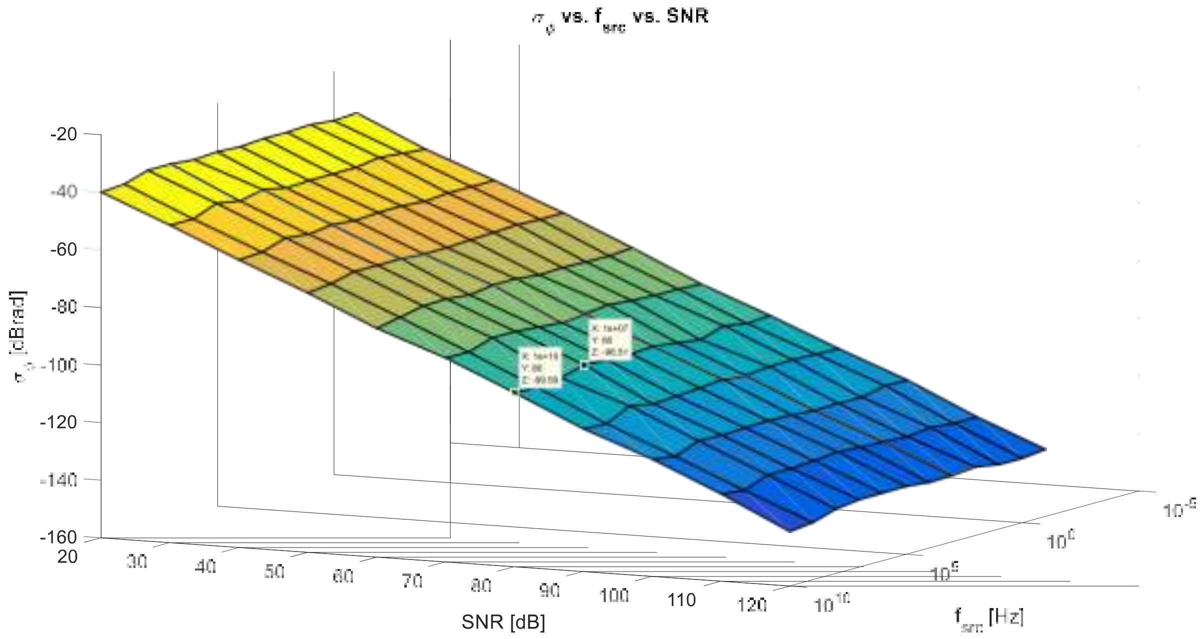

Fig. 2: SNR vs frequency vs deviation.

Where for the 80dB of the SNR (Signal to noise ratio) it will be σφ = $10^{-4°}$at a frequency 10 GHz and for 10 MHz it will be a σφ = $10^{-4°}$
Using the last results it possible obtained the value the phase standard deviation at 100Hz by a extrapolate method.
It obtained a value for deviation of -96.5065dBrad-> $10^{-4°}$.

About the las figure (Fig.3), it is a graphic where the additive White Gaussian Noise is compare against the Random jitter. The characteristics using for this last analysis make the next Monte Carlo graphic it used again the settings that are in the table2.

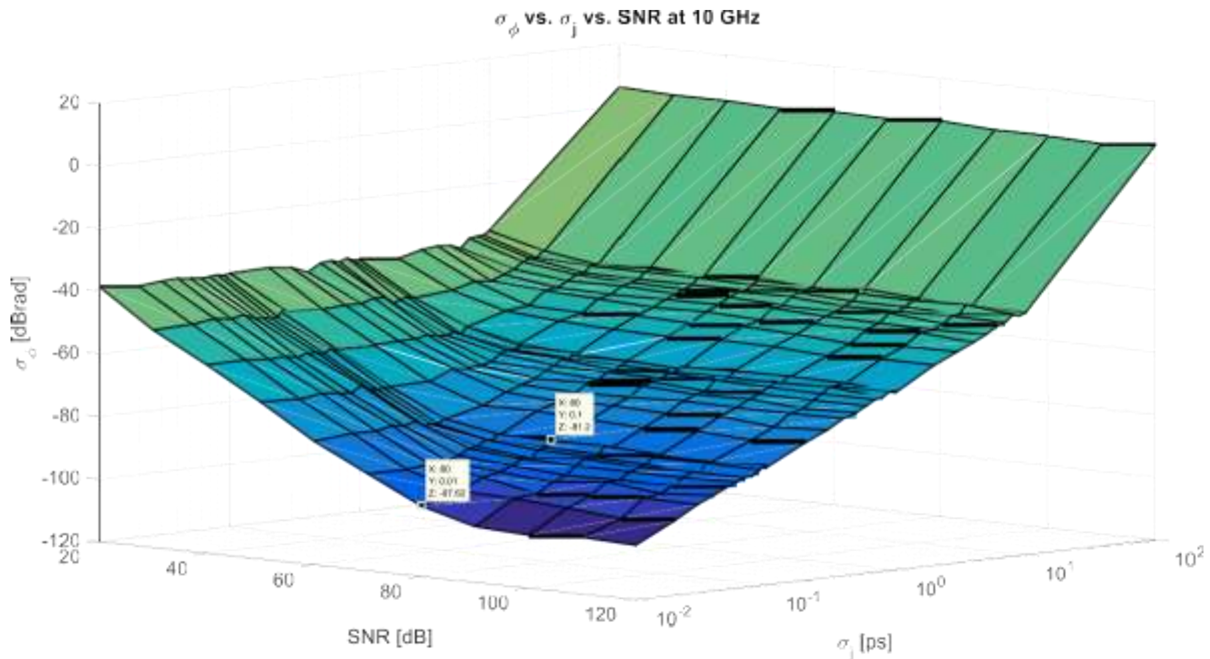

Fig. 3: Additive White Gaussian Noise and Random jitter.

Sine wave amplitude	1V
Sine wave phase	0º
Sine wave frequency	10GHz
Number of time delays	10
K number of samples for the voltmeter measurements	1000
Pulse train frequency	10MHz
Number of iteration for Monte Carlo Analysis	100

table 2: Settings for analysis additive White Gaussian Noise and Random jitter.

So in the figure3 at 10 GHz it is seen the different values of the deviation vs jitter vs SNR where, at 80 DB of SNR for example there will be σφ = 10 fs -> σφ = 10^{-4}° and for σφ = 100 fs -> σφ = 10^{-3}°.

In the first graphic (represent in the figure1) where it not additive the white Gaussian noise, is represent the random jitter against the frequency and the deviation. For the limits 100 Hz to 10Mhz the variation of the standard deviation it bigger than in the next graphic (additive white Gaussian Noise on pulse train and sine wave). So, in the first case, for the same value of the Random jitter (0.1 value) the standard deviation for this frequency values are around $10^{-2\circ}$ while in the second graphic the deviation, for the same SNR is the same (approximately) in all this point ($10^{-4\circ}$)

2 THE PROPOSED PHASE MEASUREMENT SYSTEM WITHIN THE RANGE OF 100 mHz UP TO 10 MHz

In this point, after knowing the limitations that will find on its characteristics it goes on the next step, starts with the electrical architecture that the design will be and continue with codes and tests.

2.1 HARDWARE

The architecture design used for this work where the phase measurement system working in the range of 100 mHz up to 10 MHz, that is showed in the figure4, works as follows:[2]

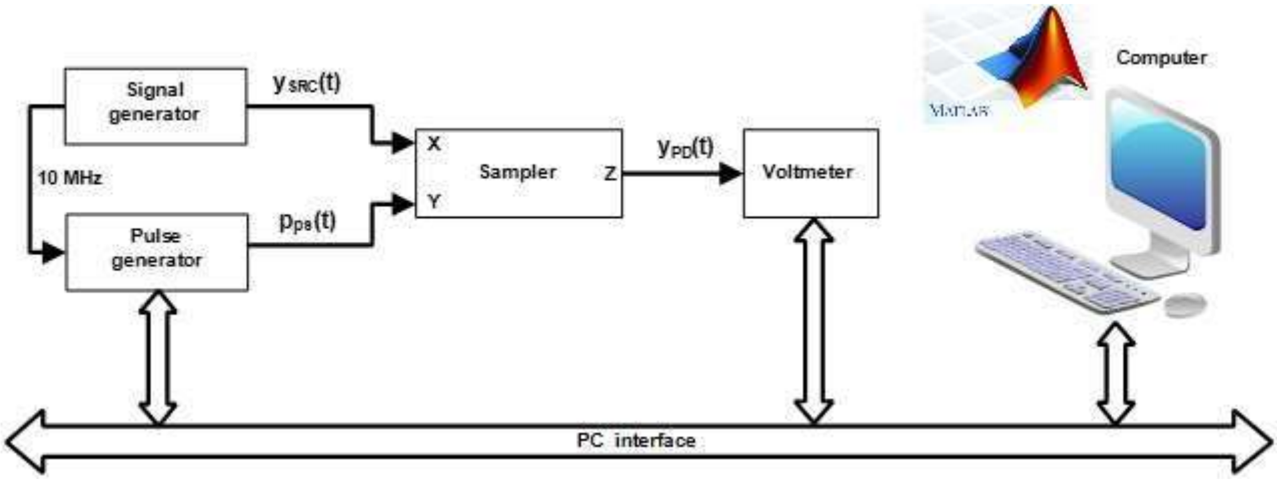

Fig. 4: architecture working in the range frequency 100 mHz to 10 MHz.

1_ Signal generator that give the reference sine wave and the 10 MHz synchronization signal by Tektronix AWG 420 is shows in the figure 5.
Tektronix AWG 420 is perfect for design or manufacturing test that need it generates an exactly analog and mixed signals. The reference clock that is used in this project is 10 MHz

[2] U. Sannio-benevento, *Draft version 2 Project : 2014-NIST-MSE-01 Second year*

Fig. 5: Tektronix AWG 420.

2_ Pulse generator, it generates a pulse for sampling the sine wave (it will possible select frequency, pulse width and time delay). In this architecture will use, for getting a pulse generator, FPGA -Field Programmable Gate Array- that will supply a pulse source and Agilent E8663B that will be the delay generator.

FPGA is a programmable device form by logic blocks, which it possible configured with the purpose that it will be adapted to the requirements. The logic blocks and the interconnection of FPGA is usually drive by microcontroller (that is shows in the figure 6, in this case with Virtex-II ML321). In this case the FPGA is join with microcontroller connected via USB to the PC
Delay generator (Agilent E86638) is a test electronic equipment that supplies specifics delays for tiggering that are require. The delay generator are used for synchronize events with a timing reference. In this case will be synchronize with 10 MHz of the signal generator.

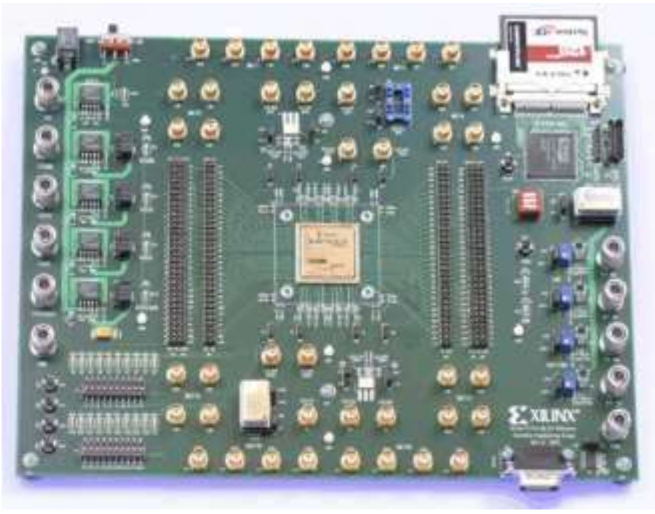

Fig. 6: FPGA board.

3_The sampler (sampling points defined by pulse train):

Sample & Hold is an analog device that samples the voltage of a signal. The input is compared to as internal voltage and stop when voltages are equal.
The performance of this devices is lay up the electric charge in a capacitor (makes the circuit inherently volatile) and it will be there, at least, one switching device (FET-Field Effect

Transistor) and one operational amplifier. The operational amplifier will be whose charges or discharges the capacitor.

4_ Voltmeter providing a voltage measurement proportional to the energy.

The purpose of the architecture, explaining before, will work in the range frequency of mHz up to 10 MHz so:
Signal generator (Tektronix AWG 420) supplying the reference sinusoidal signal and the sinusoidal signal testing. The sinusoidal signal will be change in a pulse train, with a known width and frequency by the pulse shaping (FPGA) and the selector where it will be possible to choosing the number of pulse per period to be send to the laser. The time delay will insert in the next step, by the Delay generator (Agilent E86638).

2.1.1 The pulse generator

For implement the architecture that it was explaining in the last point, first it is necessary have knowledge of how it is going to work and the specifications that the attachment have.
It will be start with the pulse generator. In this section, its intention is to offer a pulse train synchronized with the 10 MHz reference signal of generator. Each pulse defines the sampling instant of sampler. The pulse train's frequency has to be a sub-multiple of the sinusoidal frequency.
Here the pulse train is generated for sampling the sine wave. It is form by, as it said before, delay generator and the pulse shaper/selector.
Its response has the following evolution:

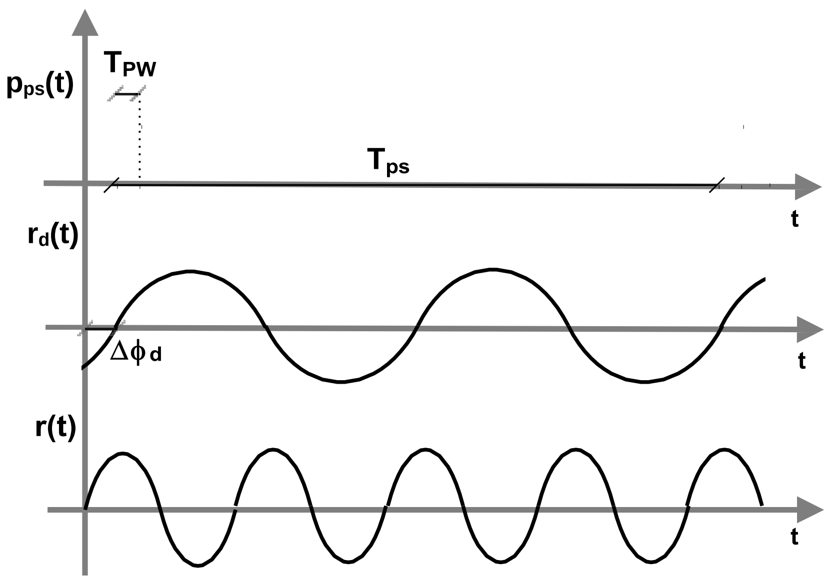

Fig. 7: response in each part of the pulse generator.

Each response that are shown in the figure7, answer with every output of each different parts of the pulse generator is formed. So, the signal r(t) is the signal output get it in the signal generator and it will be the input of the delay generator. The signal obtained in the delay rd(t) has be a delay respect to the input signal and it will possible select the frequency. On output's pulse shaper/selector will be the pulse train with a possibility to choose the pulse width.

- The specification's delay generator are in the next table (table3).

Frequency range	100 kHz – 3.2 GHz
Frequency resolution	0.001 Hz
Resolution, phase offset adjustable	0.1º
External reference frequency	10MHz ± 1.0 ppm

table 3: delay generator specifications.

- The specifications of the pulse shaper/selector to consider:
  The pulse width, for choosing, or equal to 10 ns or equal to the period of the input signal sine wave.
  The sampling pulse train period that it is possible chose must be higher or equal to the reference-delayed signal.

Therefore, considering the specifications, in the table4 are the configurations for changes for the pulse (frequency and width).

Frequency of the delayed reference signal f_{rd}	**Counting value**	**Sampling pulse train frequency f_{PS}**	**Pulse width T_{PW}**	
			Mux	
			0	1
100 kHz	10	10 kHz	10 µs	10 ns
1 MHz	10	100 kHz	1 µs	10 ns
10 MHz	10	1 MHz	100 ns	10 ns
1 MHz	100	10 kHz	1 µs	10 ns
10 MHz	1000	10 kHz	100 ns	10 ns
100 kHz	1000	100 Hz	10 µs	10 ns

table 4:pulse shaper/selector output.

Considering all that, specification it will be possible created a function, which for transfer the configured to the FPGA for obtain the purpose that the works requires.

It was used the Xilinx program for getting the next configuration:

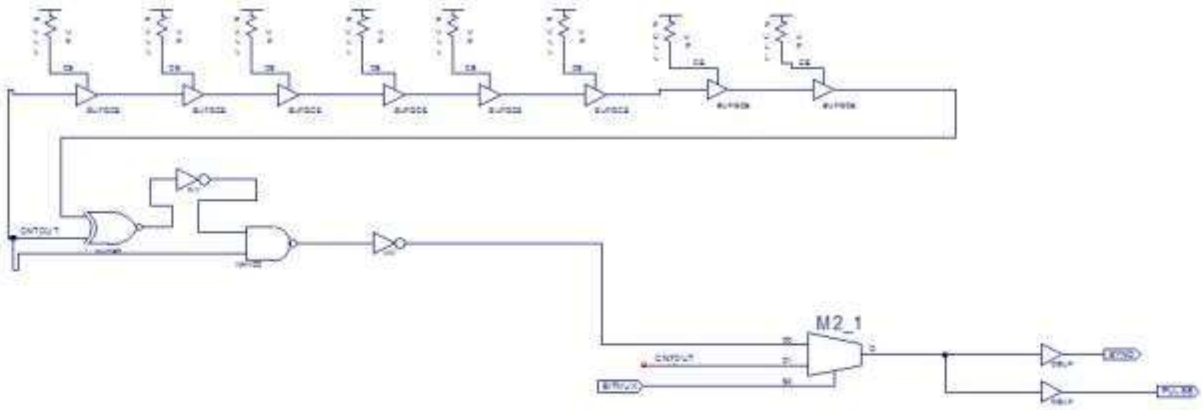

Fig. 8: pulse width generation and multiplexer for pulse width selection.

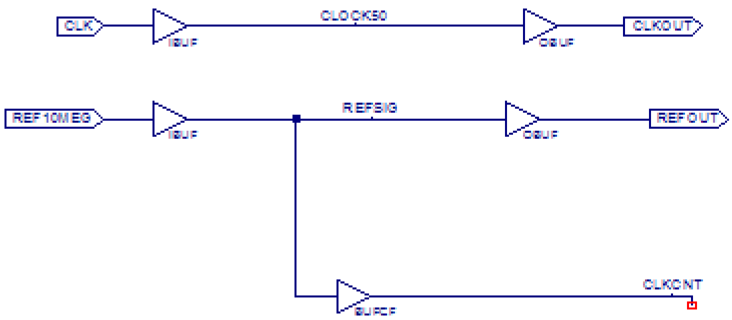

Fig. 9: internal clock.

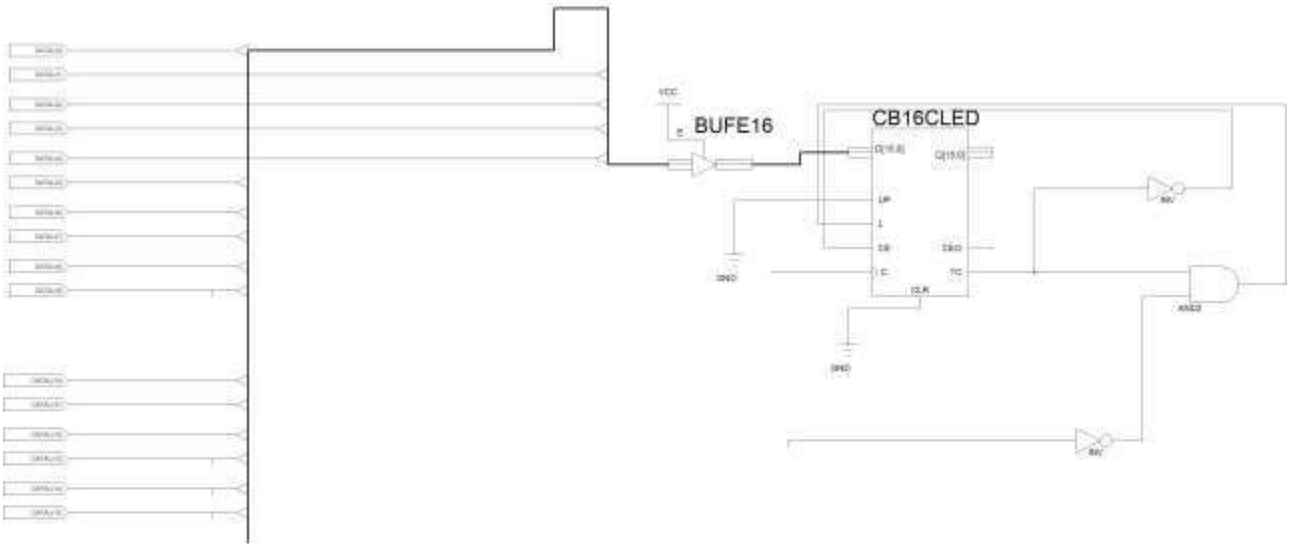

Fig. 10: digital counter, 16 bits.

In the last three figure (8,9 and 10) it represent the implementation the program in the Xilinx with the aim of getting that the FPGA works as we need for obtain the sine wave and train pulse.

The counter will control the inputs in the way of

1_ the counting value will be choosing on the 16 bits. With 16 bits, it will be possible use a 100kH

2_ the pulse width by selecting 1 bit.

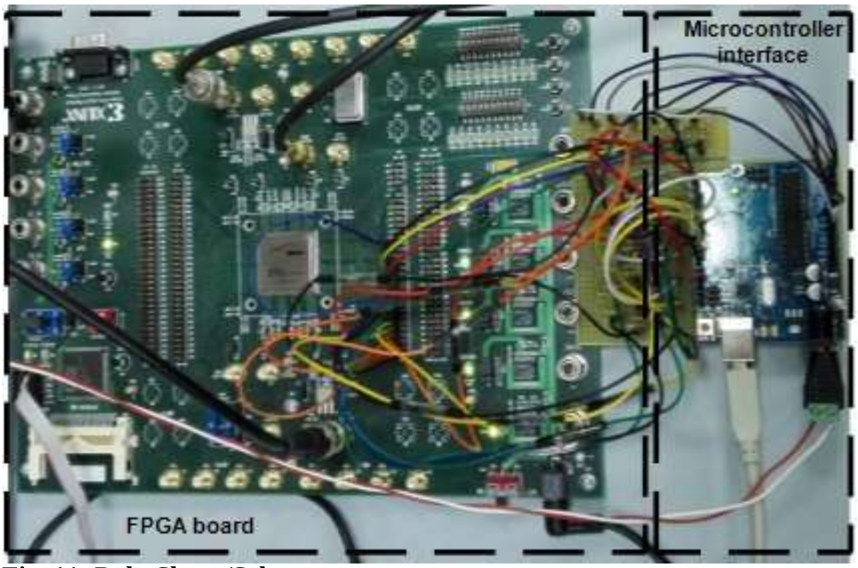

Fig. 11: PulseShape/Selector.

The last figure, Fig.11, shows how it is connected the FPGA board with the Microcontroller interface (Arduino ATEM 328D board). Microcontroller sends the value chosen on the counters. The information that it is received on the FPGA will be change in a pulse train, with a known width and frequency by the pulse shaping.

2.1.2 Signal generator

The Tektronix AWG 420 has been used to provide a sinusoidal signal of the 10 MHz to the pulse generator. The AWG should allow to select the time position of pulse in the sine wave period of each generated train signal. It necessary makes a trigger signal that it will possible synchronizing each other. Trigger input is connected with the sine wave signal.

For generate a pulse train a bit sequence is upload on the AWG. The bit sequence are the representation of the sequence according to the pulse train period and time position of pulse. The pulse width and time delay resolutions correspond to the AWG clock period.

The signal generator works following the next states:

1_waiting sequence state (waits for a new sequence to be uploaded on the memory).

2_waiting trigger state (waits the event for generating the uploaded bits sequence).

3_pulse train generation state (AWG generates the pulse train continuously until the stop command is received from GPIB interface).

AWG420 is a programmable Arbitrary Waveform Generator with 200 MS/s, 16-Bit DAC, Two-channel Arbitrary, Waveform Generator.

2.1.3 Sample and Hold

Sampler architecture is based on track and hold. It has chosen track and hold is the OPA615 by Texas Instruments. It an instrument using for fast and precise DC restoration, offset clamping and low frequency hum suppression of wideband amplifiers. It is included an Operational Trans conductance Amplifier (OTA) with high-impedance current source and precise sampling comparator.

2.1.4 Voltmeter

The objective on this instrument is give a voltage measurement proportional to the sampler output. The voltmeter should works in DC or AC measurement depends on the sampler typology and have a selectable integration time. For AC measurements, the sampler output is associated with the rms.
A multimeter is increased reading rate results in increased test throughput is clear. Strongly affecting throughput as well when changing function, range, reading speed or interfacing mode. Integration time is the period of time that the A/D converter measures the input signal. For DC or ohms measurements, the integration time determines the measurement speed, accuracy, maximum digits of resolution and the amount of normal mode noise rejection.

2.2 SOFTWARE

In order to program and organized the instruments in the way that works as it was design. The Matlab will control them.
Using the Matlab will connect the devices and will send the information that its need it for obtain the results. For example with function, SIN_GENERATION to the Tektronix AWG will send a sine wave with a frequency and amplitude chosen.

For example there is the function, SI_FIT_3_PARAMETERS that give values using in the sin fit method three parameters (this one was used for obtained the graphics that there are in the last point).

All of the functions are connected each other for getting that the board, devices and sine works as well as possible.

APPENDIX

CODE LeCraySDA600A_init()
(Connected device object, osc)

```
function deviceObj = LeCroySDA6000A_init()
     % Create a TCPIP object.
    interfaceObj = instrfind('Type', 'tcpip', 'RemoteHost',
'192.168.124.110', 'RemotePort', 1861, 'Tag', '');

    % Create the TCPIP object if it does not exist
    % otherwise use the object that was found.
    if isempty(interfaceObj)
        interfaceObj = tcpip('192.168.124.102', 1861);
    else
        fclose(interfaceObj);
        interfaceObj = interfaceObj(1);
    end

    % Create a device object.
    deviceObj = icdevice('lecroy_8600a.mdd', interfaceObj);
        % Connect device object to hardware.
    connect(deviceObj);

    set(interfaceObj, 'Name', 'TCPIP-192.168.124.102');
end
```

CODE SIN_GENERATION

```
function [Nsamp, Nk] = sin_generation(fsin,fp,Asin,phi)
    %fsamp = 10e3;
    fsamp = 200e6;
    Nper = fsin/fp;
    ch_sin = 2;
    filename_sin = 'sinNIST.wfm';
    t = 0:1/fsamp:Nper/fsin-1/fsamp;
    Nsamp = length(t);

    if Nsamp > 256
        ysrc = sin(2*pi*t*fsin + phi/180*pi);
        length(ysrc);
        tekAWGDownload(0,6,ysrc,filename_sin)
        tekAWGSet(0,6,ch_sin,Asin,filename_sin,0,1)
        Nk = 1;
    else
        t = 0:1/fsamp:10*Nper/fsin-1/fsamp;
        ysrc = sin(2*pi*t*fsin + phi/180*pi);
```

```
        tekAWGDownload(0,6,ysrc,filename_sin)
        tekAWGSet(0,6,ch_sin,Asin,filename_sin,0,1)
        Nk = 10;
    end
end
```

CODE tekAWDownload
(Charging in the device)

```
function tekAWGDownload(dev,tekAWG,x,filename)

    N=length(x);

    x=round((x/max(x))*32767);
    data_h = floor(x/256);
    data_l = x-data_h*256;

    data(2:3:3*N)=data_h;
    data(1:3:3*N)=data_l;
    data(3:3:3*N)=[0 ones(1,N-1)]; %[1 zeros(1,N-1)];

    I=find(data>0);
    data=data+128;

    digits=fix(log10(3*N))+1;

    header = ['MAGIC 2003' 13 10];
    body = ['#' num2str(digits) num2str(3*N)];

    body(length(body)+1:length(body)+3*N)=data;
    trailer = ['CLOCK 2.0E+8' 13 10];
    %trailer = ['CLOCK 10.0E+3' 13 10];
    file = [header body trailer];

    block_data = ['#' num2str(fix(log10(length(file)))+1)
num2str(length(file)) file];

    cmd = ['MMEM:DATA "' filename '",' block_data];

    g=gpib('ni',dev,tekAWG);

    set(g,'OutputBufferSize',4*N);

    fopen(g);

    fprintf(g,'MMEM:MSIS MAIN');
    fprintf(g,'MMEM:CDIR "/"');

    fwrite(g,cmd,'uint8');

    fclose(g);
    delete(g);
end
```

CODE tekAWGset
(Executed the program)

```
function tekAWGSet(dev,tekAWG,channel,A,filename,off,filter)
    g=gpib('ni',dev,tekAWG);
    fopen(g);
    fprintf(g,['SOUR' num2str(channel) ':VOLT:LEV:IMM:AMPL '
num2str(A)]);
    fprintf(g,['SOUR' num2str(channel) ':VOLT:LEV:IMM:OFFS '
num2str(off)]);

    if filter == 1
        fprintf(g,['OUTP' num2str(channel) ':FILT:LPAS:FREQ 50MHZ']);
    else
        fprintf(g,['OUTP' num2str(channel) ':FILT:LPAS:FREQ INF']);
    end

    fprintf(g,[':SOUR' num2str(channel) ':FUNC:USER "' filename
'"']);
    fprintf(g,['OUTP' num2str(channel) ':STAT ON']);
    fprintf(g,['AWGC:RUN']);
    fclose(g);
    delete(g);
end
```

CODE PULSE_SHAPER
(Create the size of the pulse depending of PW. In there, it will necessary two function more)

```
function fpulse=pulse_shaper(fp,PW)

    fpulse=1/PW;  %constante, Tpulse==PW
    pulse_freuquency(fpulse);

    Nc=(fpulse/fp)-1 % fp<fpulse

    set_FPGA_driver(Nc,1);  %0 o 1 (0_10ns)
end
```

CODE Pulse_frequency

```
function pulse_freuquency(fpulse)
    dev = 0;
    add =19;
    g = gpib('ni',dev,add);
    fopen(g);
    fprintf(g,[':FREQ:FIXed ',num2str(fpulse)]);
    fprintf(g,':POWer:LEVel 10.0DBM');
    fprintf(g,':OUTPut:STATe ON');
    fclose(g);
end
```

CODE Set_FPGA_driver

```
function set_FPGA_driver(count,mux)
    s = serial('COM4','BaudRate',9600);
    fopen(s);
    pause(2);
    fprintf(s,'C%c',char(count));
    pause(0.5);
    fprintf(s,'M%d',mux);
    pause(0.5);
    fclose(s);
end
```

CODE Phase_Measurement

```
function [phi, v_used, dT_used] = phase_measurement(fsin,fpulse,PW,Ndelay)
%% f is the frequency of the sinusoidal signal
    cc =1;
    for ii=1:Ndelay
        if ii==1
            dT_imp(ii)=0;
        else
            dT_i = delay_generation(fpulse,1/(fsin)/Ndelay);
            dT_imp(ii)=dT_imp(ii-1)+dT_i;
        end
        pause(1)
        v_used(cc) = Fluke8508A_read(0, 18);
        dT_used(cc) = dT_imp(ii);
        cc = cc+1;
        pause(0.5)
    end

    delay_generation(fpulse,1/(fsin)/Ndelay);
    pause(2)
    [A, phi, off]= sin_fit_3_parameters(v_used,dT_used,fsin,PW);

    t = 0:1/(fsin*100):1/(fsin)-1/(fsin*100);
    figure
    plot(dT_used,v_used,'*')
    grid on
    hold on
    vrec = A*sin(2*pi*fsin*t+phi*pi/180)+off;
    plot(t,vrec,'r')

end
```

CODE Delay_generation
(get it the value dT)

```
function dT_imp = delay_generation(fpulse,fsin,dT)
    if fsin==10e6
        step=1;
        phi_step = 3.6;
    elseif fsin==1e6
        step = 1;
        phi_step = 36;
    elseif fsin ==10e3
```

```
            step = 2;
            phi_step = 180;
        elseif fsin ==100e3
            step = 1;
            phi_step = 36;
        end
        dev = 0;
        add =19;
        g = gpib('ni',dev,add);
        phi_im=0;
        fopen(g);
        for i=1:step
            fprintf(g,':PHAS:REF');
            fprintf(g,[':PHAS ', num2str(round(phi_step*10)/10),'DEG']);
            phi_im=phi_im+phi_step;
            pause(0.1);
        end
        dT_imp = phi_im/360/fpulse;
        fclose(g);
    end
```

CODE Phase_osc

```
function phi = phase_osc(ch1,ch2,fsin,deviceObj)
    fosc = 10e9;
    data = LeCroySDA6000A([ch1,ch2],deviceObj,fsin);
    Ns = 1/fsin*fosc;
    ysin=data(1:Ns,1);
    ypulse=data(1:Ns,2);
    ym = abs(ysin);
    [v_sine,t_sine] = min(ym);
    vpp = (max(ypulse)-min(ypulse))/2;
    Iu = find(ypulse>vpp);
    tpulse = Iu(1);
    %figure
    %plot(ysin)
    %hold on
    %plot(ypulse,'r')
    phi=(t_sine-tpulse)*100*10^-12*fsin*360;
end
```

CODE Fluke8508A_read

```
function vm = Fluke8508A_read(dev, add)

    g = gpib('ni',dev,add);
    fopen(g);

    fprintf(g,['*TRG']);
    fprintf(g,['RDG?']);
    vm = str2double(char(fread(g,80))');
    fclose(g);
    delete(g);
end
```

CODE Sin_fit_3_parameters

```
function [A, phi, off] = sin_fit_3_parameters(x,dt,f,PW)
% sin_fit_3_parameters(x,phi,f,fp,PW,fsim) - this function returns
the estimated
% values of A, phi and off of the sinusoidal using the sin fit method
with
% three parameters:
% - x is the amplitude vector of the sinusoidal signal;
% - phi is the vector of the delay times considered in the
%   delay generation step;
% - f is the frequency of the signal;
% - PW is the pulse width time;
    dt = dt;
    M = length(x);
    D = [cos(2*pi*f*dt(1:M))' sin(2*pi*f*dt(1:M))' ones(M,1)];
    s = D \ x';
    A = sqrt(s(1)^2+s(2)^2);
    phi = (pi/2-atan2(s(2),s(1)))*180/pi;
    off = s(3);
end
```

3 RESULTS

3.1 TEST BENCH FOR TRACK AND HOLD CIRCUIT

TEST AT 100 HZ: For this test, the sine wave frequency has been conducted considering a pulse width of 10µs and with a frequency of 100 Hz in the pulse train. In the figure12 shows the measured phases respect to the reference phases measured with universal counter.

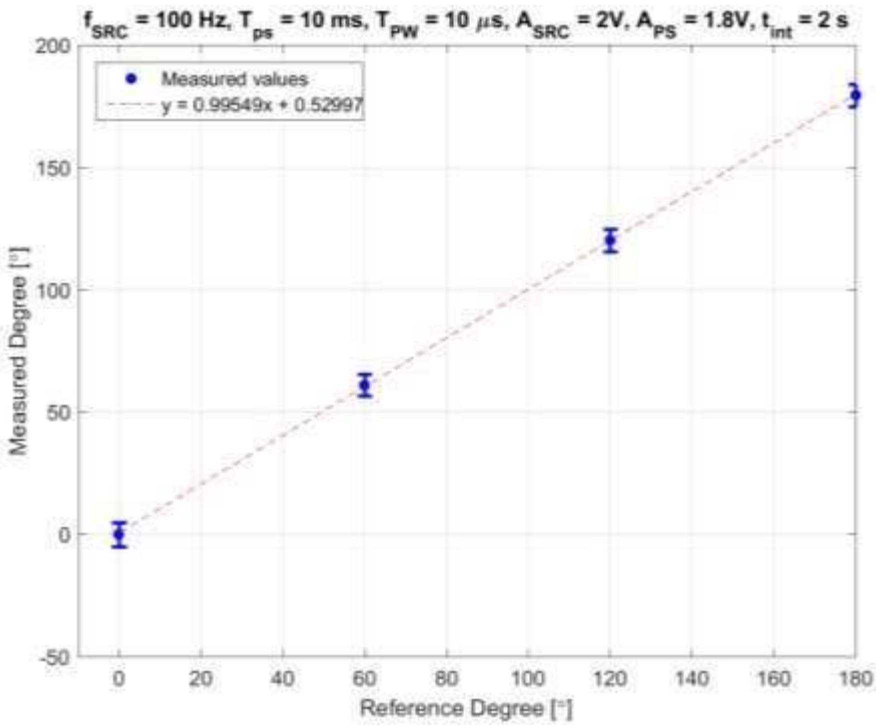

Fig. 12: testing at 100 Hz

TEST AT 10 KHz: For that, test the pulse width considering of 10 ns and a pulse train frequency of 10 kHz. The next figure, figure13, shows the measured phases respect to the reference phases measured with universal counter.

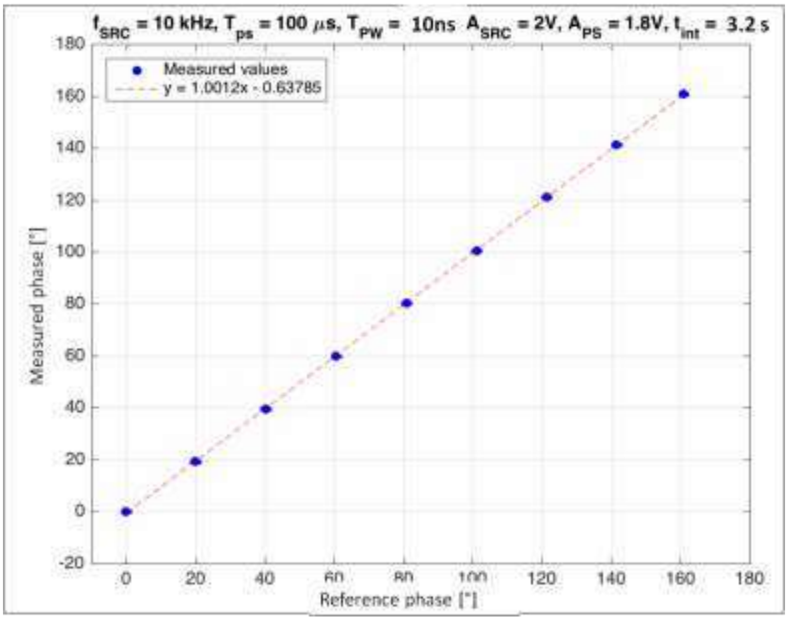

Fig.13: testing at 10 KHz

TEST AT 100 KHz: in this case, it has been considering a pulse width of 1µs and a pulse train frequency of 100 kHz. The figure 14 reports the measured phases.

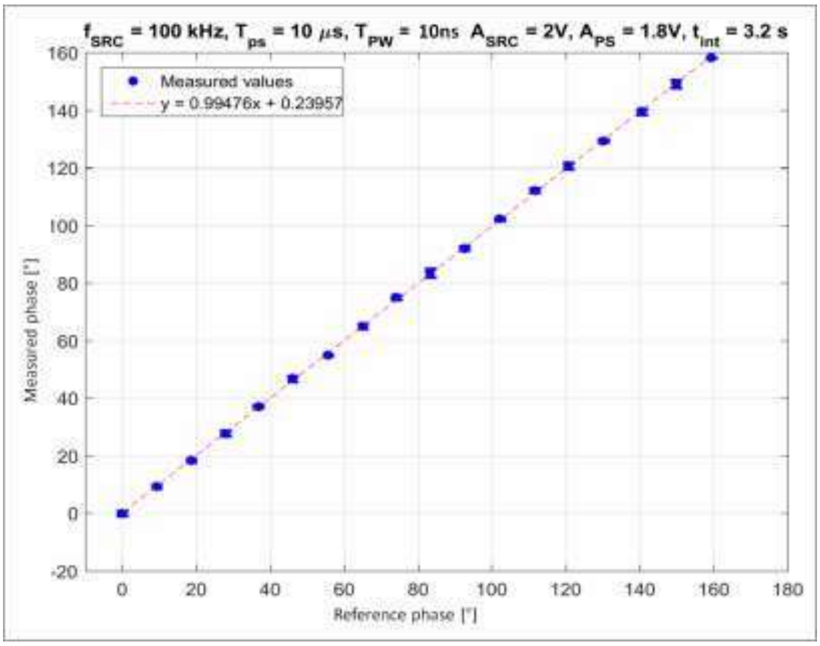

Fig.14: testing at 100KHz

TEST AT 1MHz, with a pulse width of 100 ns and a pulse train frequency of 1 MHz. The next figure, fig15 represents the measured phases respect to the reference phases measured.

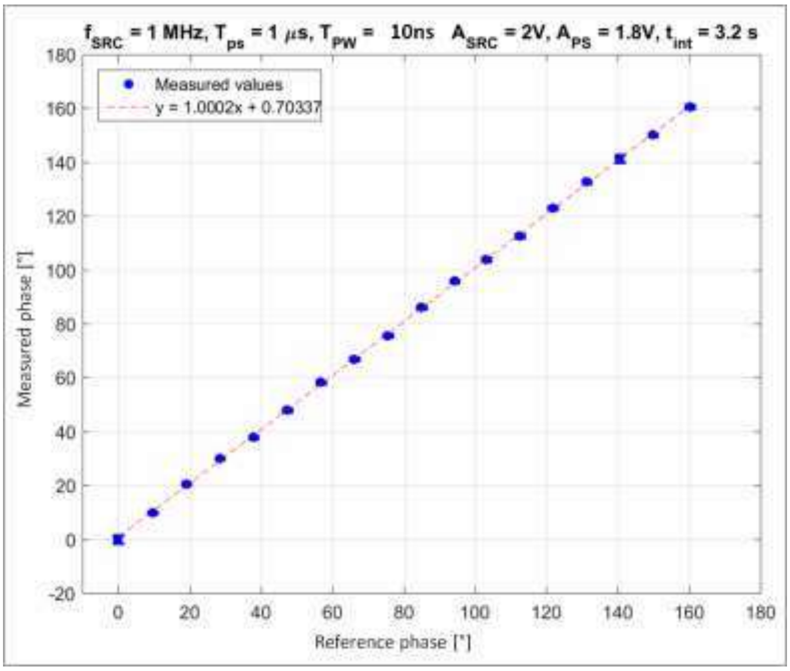

Fig. 15: testing at 1 MHz

TEST AT 10 MHz: : here, a pulse width is considering of 10 ns and a pulse train frequency of 1 MHz. Figure16 represents the measured phases respect to the reference phases measured with universal

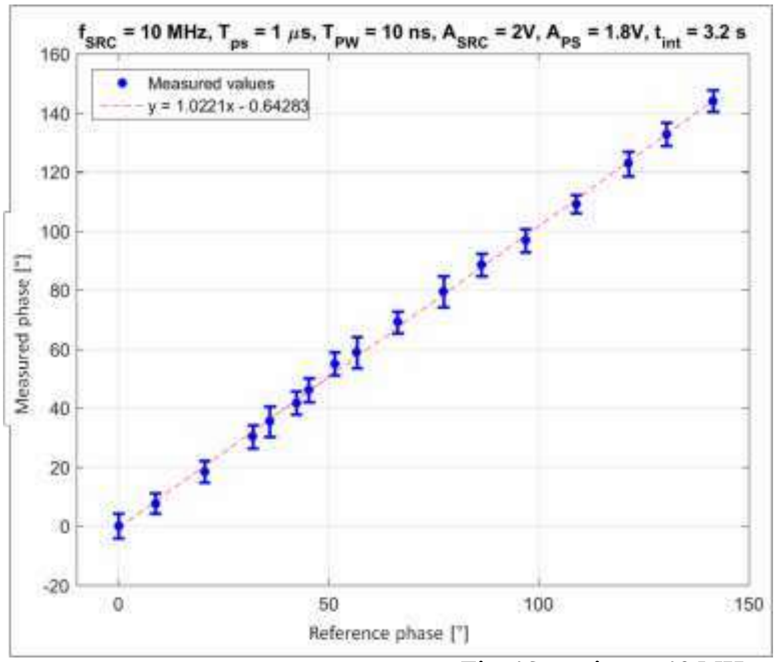

Fig. 16: testing at 10 MHz

Before seen the last five figures its possible conclude that for testing at a higher frequency or at lower the deviation is bigger, while for the rest of values (10 Khz, 100 KHz and 1MHz) the measured values has no variation here are better (lower) than the results that were expected while in the 100 Hz and 10MHz frequency are affected by system more than the ones obtained in the section2.

4 CONCLUSIONS

With this prototype the test at low (100Hz) and high (10MHz) effects are bigger than spectated. The other frequency have better adaption to the results spectated and the real one.
For the future works, changing the FPGA, using a new counter for will be possible working in lowers frequency. It will change the generation of the pulse using a new programmable device where try to get a better resolution for a bigger range of frequencies.

5 REFERENCES

[1] T. Atmel, H. Performance, L. Power, A. Avr, and M. Family, “ATmega328 / P,” 2016. – DATASHEET ATmega328/P

[2] P. Bosco-lucarelli and P. Roma, *Draft version 3 Project : 2014-NIST-MSE-01 Second year*. 2017.

[3] W. Drive, “Test Equipment Solutions Datasheet Agilent E8663B Analog Signal Generator Data Sheet,” vol. 44, no. 0, pp. 0–20. – DATASHEET E8663B

[4] X. Gao *et al.*, “Real-Time Phase Correction Based on FPGA in the Beam Position and Phase Measurement System,” *IEEE Trans. Nucl. Sci.*, vol. 63, no. 6, pp. 2903–2908, 2016.

[5] U. Sannio-benevento, *Draft version 2 Project : 2014-NIST-MSE-01 Second year*. 2016.

[6] A. W. G. Series, “Arbitrary Waveform Generators,” vol. 102, pp. 3–6. – DATASHEET AWG420