Position Sensor of a Low-Frequency Loudspeaker Cone

José Rafael Mateus Henriques Santos
IST, University of Lisbon, Portugal
jose.r.m.h.santos@ist.utl.pt

Abstract—This paper presents a system designed to measure in real time the position of any speaker’s cone, so that that information can be used in a speaker frequency response correction device. This was accomplished by designing an optical system composed of a laser, a mirror on the cone and a linear light sensor. A dedicated circuit board was also built for that purpose, and by taking advantage of the versatibility of an Xilinx FPGA, and a Microchip PIC to read signals from the sensor and run an algorithm that will grant the measurement of the cone position with greater resolution than the one that is offered by the sensor. This algorithm runs by extracting the phase of one single discrete fourier transform coefficient from the sensor pixel array samples, which can be implemented in hardware in simple way. However, since this system will have to send this data to a digital signal processor, the board features a microchip PIC, that is responsible for communicating with outside devices. The results from this system are very promising, as they reflex exactly what was expected from it, and because there is room for improvement with the hardware implementation, this project promises to be a great innovation to speaker response measurement techniques.

Index Terms—Speaker frequency response, Optical system, DFT, Hardware design, Mixed signal, Circuit design

I. INTRODUCTION

For many years, people have been surrounded by electronic devices with which they can send, receive, record or play acoustic signals like speech or music. Those devices range from the early telephones, radios, tape and CD recorder, high fidelity reproduction systems and so many others. What these devices have in common is that all of them make an interface between the the physical world where the sound waves propagate to the electric world, where those waves are represented as electrical signals, either in voltage or in current.

Technology allowed the enhancement and creation of new kinds of microphones that operate very differently from each other, but with loudspeakers, for many years, the operation principle and mechanical structure remained the same. Modern loudspeakers are still composed of a moving cone that creates sound by oscillating, which in turn compresses air and generates sound waves. This cone is attached to a moving coil positioned between strong magnets, and displaces itself, and the cone, when a current flows through it, this current being the audio signal to be played.

In fact, to achieve a well balanced and accurate sound reproduction system, different kinds of loudspeakers are used, so that each one operates at the frequencies where it excels. This range of frequencies is defined by their variation of the size, materials and enclosures. Because speakers are designed to operate under a specific range of frequencies, when they are forced to operate under other frequencies, the quality with which they output greatly decreased.

These quality decrease can yet be attenuated, or even completely nullified, using control techniques that change the speaker’s input signal so that it is adapted to the speaker itself, making it look like the speaker has very good frequency characteristics. To apply this techniques, though, it is absolutely necessary to acquire data from the speaker as it is operating, so that the input signal can be altered according to the speaker output. The most common way to acquire this data is by using a microphone directly in front of the speaker. This method, however, has its limitations, as the microphone can pick up sounds resulting from the room acoustics, and very depedant on whether the microphone is positioned closer or further away from the speaker. In turn that causes the microphone to have an inaccurate reading of the speaker output [1].

The system developed for this project was designed to acquire that same information from the loudspeaker, but instead of measuring the sounds waves it generates like with a microphone, this system measures the speaker’s cone position in real time, using an optical apparatus, along with a dedicated circuit board where that information will be mathematically calculated and transmitted.

It was mentioned before that the developed system contrasts greatly with the microphone acquisition method. As an acquisition system that is room reflection independent, it can be considered as a more suitable solution for loudspeaker frequency response measurement.

Once the overall system design was outlined, the goal of this project is to design a cone displacement measurement tool that can be used with any speaker response correction system. We hope to achieve this by designing a versatile optical module to apply to the speaker, and by building a compact and low price circuit that can both be connected easily to a wide range of devices, and still run the necessary calculations to output an accurate result of the cone positioning.

This document will provide an extensive description of the system designed and built to read the speaker cone position in real time. Firstly, section II presents how the system was designed and the algorithm it will use. Secondly, chapter III demonstrates how this system was implemented. The results of the built system will be showed in chapter IV as well as the testing framework. The last section will finally conclude this document.
II. Cone Position Measurement

A. Optical module

This project features one way to obtain the positioning signal, by measuring it directly from the speaker, using an optic system specifically created for this functionality. This optic system makes use of a linear charge-coupled device (CCD), similar to those used for reading code bars, where the reflection of the light signal from a laser diode is aimed at. That refraction comes from a small mirror placed at the center of the speaker cone, that moves along with the speaker. This configuration allows the system to be placed at a fair distance from the speaker itself, reducing the effects caused by acoustic vibrations, that could compromise the accuracy with which the cone position is obtained. However, there are characteristics of this system that must be taken into account, in order to maximize its capabilities.

Firstly, the sampling rate at which the cone position can be acquired is completely dependent on the maximum CCD data rate. Assuming that the signal processing and transmission does not pose the operating frequency bottleneck of this system, the cone position sampling rate is given by equation 1, where \( F_r \) is the CCD pixel refresh rate, \( N \) is its number of pixels, \( f_{\text{max}} \) is maximum cone position sampling rate (equal to this system’s Nyquist frequency), and \( f_s \) is the overall CCD sampling frequency.

\[
f_{\text{max}} = \frac{f_s}{2} = \frac{1}{2} \frac{F_r}{N}
\]  

Secondly, the signal obtained by the optical sensor does not directly correspond to the cone’s position. In truth, this signal is a merely a section of light signal, whose intensity is very similar to a 2-dimensional gaussian curve in space. This way, the curve presented by the sensor moves along with the cone displacement, and its peak corresponds directly to its position. In fact, any point from this curve can be used to acquire the cone’s absolute position, provided that the same point is used as a reference. Nevertheless, the curve peak will be used, for the simplicity of the algorithm applied in its calculation, that will readily be referred in subsection II-B.

Moreover, note that it is necessary to extract as much information as possible with this module, in order to attain data that is the closest to reality, since this project will be implemented digitally. With that in mind, it is essential to optimize both this optical system and the propagation of the digital signals, when they are processed.

This way, because the speaker cone maximum displacement is different from linear sensor window length, it is clear that the optical system should be adapted so that both the total laser incidence excursion and the CCD length match each other, thus maximizing the number of positions the light beam can reach in the sensor. This adaptation can be done by applying trigonometric and reflection properties, using different angles and distances between the components, namely between the laser, the mirror on the speaker cone and the CCD chip. This solution bestows several degrees of liberty to the system, and are used to optimize the spatial arrangement of the components, against the restrictions caused by the excursion matching. The overall arrangement of those components is presented in figure 1, where \( \alpha \) is the angle of the laser beam with respect to the \( y \) axis (perpendicular to the cone plane), \( \theta \) is the angle of the mirror axis with respect to the \( y \) axis, \( \phi \) is the reflection angle, \( \beta \) is the CCD angle with respect to the plane which is parallel to the speaker cone, \( d \) is the distance between the mirror and the remaining components’ plane, and \( l \) is the distance between the CCD center and the speaker cone axis.

The equations that rule this system are presented from 3 to 7, where \( \Delta x \) represents the speaker cone displacement (along with the mirror), \( \Delta y \) is the laser excursion when projected onto the laser and CCD plane and \( \Delta z \) indicates the laser excursion when projected on the CCD window surface. Note that 3 indicates the location of the laser beam while the cone is in its natural position, and 7 represents the variation of the total excursion of the laser beam on the sensor in function of the displacement of the speaker cone.

\[
\phi = 2\theta + \alpha
\]

\[
l = d \tan \phi
\]

\[
\Delta y = \Delta x \cos \phi (\cos \alpha + \sin \alpha \tan(\theta + \alpha))
\]

\[
\Delta z = \Delta y (\cos \beta + \sin \beta \tan(\phi + \beta))
\]

\[
\Delta z = \Delta x \cos \phi (\cos \beta + \sin \beta \tan(\phi + \beta)) \times (\cos \alpha + \sin \alpha \tan(\theta + \alpha))
\]

In order to be able to change the different angles and position of the different components of this optical system, a structure like the one exemplified in 2 was manufactured to hold them. This structure was carved in metal, which is a hard material that will not vibrate with the sound pressure emitted by the speaker. It also features two slots where the laser and the CCD circuit board will be able to slide and rotate, to define their distance and the angles at which they either send or receive the laser beam. It can be placed at any distance from the speaker, as long as the rules for the laser excursion still apply.
B. Peak position calculation.

As mentioned previously (see subsection II-A), the signal acquired by the CCD sensor is similar in nature to a gaussian curve, and as such, it is characterized by its amplitude, mean and variance. We are interested in obtaining its mean, which represents the speaker’s cone position, and is where its peak is located. However, the CCD sensor has a finite number of pixels, and without any processing, the only way to acquire the curve’s peak position would be by simply searching for the pixel sample with the highest value. The number of mean values that could be extracted would then be limited by the number of pixels existent in the sensor chip, in other words with a low resolution. On the other hand, if one were to use all the samples and interpolate the exact peak position, the cone position would then only be limited by the resolution of its digital representation, thus being far more suitable in this application. To do so, a special algorithm is applied to the CCD output pixel samples, that will be presented promptly.

The fact that the signal acquired by the CCD is similar to a gaussian curve is what led to the development of this algorithm. In truth, any regular curve that has similar characteristics to a gaussian curve such as the Poisson distribution function, some particular cases of the Beta distribution function and Gamma distribution function. That is due to the fact that this algorithm is based on locating the mean value of those distributions.

Firstly, by analyzing the equation of a gaussian function 8, it is defined by an amplitude A, a mean value \( \mu \), a standard deviation \( \sigma \). Note that when we apply the Fourier transform to this function we get 9. From that expression we know that the mean of the gaussian function is proportional to the transform phase, because \( e^{i\mu t} \) is the only imaginary section of that result, and this way the mean value of the function can be extracted and converter into a position in space if we calculate its transform. We can conclude as well that the mean value of the gaussian function is then similar to a delay in space from its origin. By applying a discrete Fourier transform to the array of pixels output by the CCD sensor, we can extract the second DFT coefficient’s phase and calculate the exact position of the laser on the CDD. This can of course be calculated with more precision, as we calculate more and more DFT coefficients, because the phase of the transformed signal is always proportional to the curve’s mean [9]. Finally, to convert the phase value obtained with this method into the true pixel position we need to use equation 10, which is merely a conversion calculation, and where \( y \) is the true position of the curve’s position, \( N \) is the number of effective pixels in the CCD, and \( t = 1 \) represents the second coefficient of the discrete Fourier transform(\( t = 0 \) being the first.).

\[
f(x) = A e^{-(x-\mu)^2/2\sigma^2} \tag{8}
\]

\[
F(t) = \int_{-\infty}^{\infty} f(x)e^{i\mu t}dx = Ae^{i\mu t}e^{-\frac{i}{2}(\sigma t)^2} \tag{9}
\]

\[
y = -\frac{N}{2\pi} \times \text{arg} F(t = 1) \tag{10}
\]

III. Acquisition Circuit Boards and Firmware Implementation

A. Designing the Printed Circuit Board

The communication to and from this system is done using an USB interface, implemented via the FTDI FT2232HL, chosen for its flexibility and speed capability. This IC is absolutely compatible with USB 2.0 High Speed (480 Mbits/s) and features two-way interfaces of several protocols, both serial and parallel, to and from USB [2]. Such features make this IC very useful while debugging and transmitting large amounts of data, in addition to the simplicity it carries to perform the communication in the serial/parallel end.

In fact, this IC is closely related to the Microchip microcontroller used in this system, so that it acts as the mediator between the microcontroller and the outside world. As such, it must share at least one communication protocol with the microcontroller. In the FT2232HL end, this protocol is described by the manufacturer as a CPU-style FIFO. This is a parallel interface designed to for direct communication with a CPU, where it essentially sees the USB end as a FIFO to where it can write and from where it can read freely, as long as there is available memory or available data in the FIFO, respectively.

This interface operates with 4 input control signals and 8 bidirectional data pins. The 4 control signals are: active low chip select #CS, address bit A0, active low FIFO read #RD and active low FIFO write #WR. This mode uses a combination of #CS and A0 to determine the operation to be carried out. Possible operations are pointed out in table I, with the associated control signals [2]. All these signals are controlled in the microcontroller end, and shall be further discussed promptly.

<table>
<thead>
<tr>
<th>#CS</th>
<th>A0</th>
<th>#RD</th>
<th>#WR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>Read Data Pipe</td>
<td>Write Data Pipe</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Read Status</td>
<td>Send Immediate</td>
</tr>
</tbody>
</table>

This USB interface circuit is designed to run strictly with a 12MHz (±30 ppm) crystal or oscillator. FTDI’s documentation also suggests the use of capacitors connected to ground and
to the crystal terminals, whose capacitance is defined by the crystal manufacturer. The crystal applied to this system was the Abracon AB308-12.000MHz for its dimensions, stability and load capacitance [3]. Upon designing the circuit, the oscillating signal generated by the crystal-capacitor-FT2232 set was distributed to the other main integrated circuits: Microchip PIC microcontroller and Xilinx Spartan-3A FPGA, as their main clock source.

The complexity of this system requires that there must be distinct hardware solutions for the different kinds of operations to be carried out. Also, the fact that most integrated circuits in this system communicate with each other suggests the use of a centralized communication solution. There must be an overseer of the entire system, which holds responsibility for receiving and carrying out commands from outside sources, and communicate with other integrated circuits across the board. This communication central must then be absolutely programmable, be able to communicate via different serial and parallel protocols, run on a high enough clock frequency, and hold enough memory to store either processed data from the Xilinx FPGA or raw data from the CCD sensor. With that in mind, PIC microcontrollers, manufactured by Microchip, are known for their versatility when it comes to embedded communication modules, and for their simplicity to use, mount and program. These factors altogether motivated the use of the PIC24FJ128GB106 microcontroller for this system. It is a 16-bit modified Harvard architecture processor in a 64-pin package, that runs at a maximum frequency of 32 MHz, with 128 KBytes of program memory, 16384 Bytes of data memory, serial communication via UART, SPI and 12C, parallel communication port, analog to digital converters, and several other features, irrelevant to the scope of this project [4].

Firstly, and as will be approached later, most calculations and data are represented in 16-bit, and so is the data transmission width. According to that information, and as was described for the light sensor, each reading is composed of 3648 samples. This means that each line of the sensor can be stored across the integrated circuit efficiently and with a small delay. Both signals are assigned to global clock pins, which are clock signal offered by the programmable oscillator peripheral. Amidst two chip select lines, 16-bit address capability, read and write strobes, multiplexing, and further programmable options, the PMP must be defined to communicate with the FT2232HL USB interface circuit, and one arbitrary frequency clock signal offered by the programmable oscillator peripheral. Both signals are assigned to global clock pins, which are specifically designed to route clock or high frequency signals across the integrated circuit efficiently and with a small delay.

In the second place, this IC operates as the SPI master to acquire the data converted by the analog to digital converter (ADC). The digital section of this ADC uses only three SPI signals, due to the fact that no MOSI signal is needed. There was an effort to assign these signals to global clock pins for the same reason as the clock signals. Thirdly and in opposition to what was just mentioned, the FPGA communicates with the PIC using SPI as well, except that in slave mode, and in full duplex mode, which is a SPI protocol mode where all four signal lines are used. In particular for this circuit, the MOSI and serial clock are forwarded through the FPGA to the peripherals (mentioned while describing the PIC setup), while the MISO signal is solely existent between the FPGA and the PIC, due to the fact that no data is sent back from the peripherals. After that, it is the FPGA’s responsibility to control how the light sensor operates. In fact, the CCD requires only three control signals, which are all handled by the former: one integration clear gate pulse (ICG), one shift gate pulse (SH) and one master clock signal. Last of all, the FPGA can
be configured using a dedicated JTAG programmer. In a way very similar to the one used to configure the Microchip PIC, the FPGA can be configured while already mounted in the printed circuit, using Joint Test Action Group (JTAG) instead of ICSP. This is a protocol widely used for testing printed circuit boards using boundary scan and communicating with chips.

The analog section of this system is divided in two separate boards. The first section is located in a very small board containing the CCD and the power drive circuit for the laser, where a general 10-pin header offers power and several lines to transmit signals and data, to and from this daughter board, respectively. The second section is located on the processing board along with the rest of the digital components.

As mentioned, 3 control signals are sent to this daughter board to drive the light sensor, and 2 signals are returned: the pixel analog value and the CCD indicator. However, since there are two different ICs that can be used as sensors for this project, the Sony ILX554B and the Toshiba TCD1304AP, the control and analog output signals must be correctly routed to the same pins in the connection socket, as they are tied to different pins in the sensors integrated circuits. This is achieved using a set of jumpers that act as manual multiplexers to route such signals to the correct corresponding pins in the sensor. It should be noted that the analog output signal is buffered using one operation amplifier, in order to avoid load effects caused by additional components in the signal chain.

Before the analog value output from the CCD is converted to its digital representation, it must be analogically processed, so that the conversion can be performed with the highest effective resolution available. As a result, knowing how the CCD output signal is characterized, the second analog section located in the processing board is responsible for conditioning the sensor’s output signal, before it is converted.

\[
V_{OS} = V_{DAC} - V_{CCD} \approx v_{DAC} - v_{CCD}
\]

As it can be observed in figure 3(a), the sensor’s output signal is an array of pixels whose analog voltage value is essentially composed of a continuous voltage \( V_{OS} \) that lowers when light is pointed at them [7]. As the light intensity increases, the output voltage decreases. This voltage must be conditioned to fit the ADC range and must also be inverted. In the processing board, this was achieved using one single operational amplifier, mounted as an inverting amplifier with unity gain (figure 4), where the positive terminal of the OpAmp is tied to the analog output of a voltage digital to analog converter - DAC121S101.

However, this intermediate signal is not absolutely fit to the maximum voltage 4.096V, assigned by the voltage reference, as mentioned earlier. To amplify it in order to fit the maximum voltage excursion possible, an OpAmp is used along with a different DAC. This IC is an Analog Devices AD5444 which is, in essence, a programmable resistor. Together with the operational amplifier, it is possible to create a simple non-inverting amplifier, like the one represented in figure 5. The gain equation for this amplifier is therefore \( v_o \approx (1 + \frac{W}{R})v_i \), where the gain ranges from 2 to 4097. All in all, the first DAC extracts the DC value \( V_{OS} \), and the second DAC acts as a variable gain amplifier to adjust \( V_{MDK} \) to the reference voltage of the analog to digital converter.

The advantage of fitting the input signal of the analog to digital converter to the whole VCC range is closely related to the effective resolution the converter can output. For example, if this input maximum voltage level would be 2.048V - half of the current VCC maximum - the digital words that represent voltage levels between 2.048 and 4.095V would never be reached, reducing the effective bit resolution of the ADC from 12 to 11 bits, and consequently the precision with which the cone position can be calculated.

Despite all conditioning, the analog to digital converter plays a vital role in the entire circuit, because it represents the bridge between the analog and digital domains. The Analog Devices AD7883 operates 3-MSPS, sampling three times faster than the CCD pixel value can be updated, leaving enough room (approximately 667 ns = 1 us (CCD refresh period) - 333 ns (ADC sampling rate)) for the calculations performed to run the algorithm presented earlier in real time.

B. Firmware

1) FPGA system: Previously, it was clarified how the FPGA is connected to other components in the circuit, but not how
it would use these connections and operate to carry out the calculations it is intended to. In fact, configuring this FPGA for that effect required several steps, starting from simple hardware implementations that would, for example, send the control signals to the CCD sensor, or send data to the PIC through SPI. The final working hardware design solution is an incorporation of several small modules like these, each with a specific functionality, that can together control the CCD, acquire the converted samples, process or store them and finally send them to the PIC microcontroller.

In this hardware solution, there are four state machines [8]: three responsible for controlling small modules, namely the SPI module linked to the ADC, the sensor controller and the SPI module linked to the PIC microcontroller, and one responsible for controlling the datapath together with the other machines. The top level of the hardware structure runs under the following set of operations, repeated for as long as the master machine is not reset:

1) The CCD controller starts its working cycle to read a line from the sensor, by setting and resetting the CCD1, CCD2 and CCD3 pins (SH, Clock and ICG respectively) according to the timing specifications in the sensor documentation.

2) The CCD controller sends out a sample flag to the master machine (which is on hold) indicating that one pixel has been output, and is ready to be converted to its digital representation. Furthermore, the CCD controller state machine operates in parallel with the rest of the system, so this point is repeated until there are no more pixels available to read from the sensor, upon which this controller return to step number 1.

3) As the sample flag is toggled, the master machine issues the ADC SPI module to start a read, immediately entering a state holding for the completion of the conversion. Consequently, the conversion happens in parallel, as the master machine holds.

4) When the conversion is completed, the ADC SPI module toggles a flag - done flag - to which the master machine responds enabling the datapath during one single clock cycle, because the sample is ready to be read and processed. The master machine then enters a holding state, until it is time to acquire another pixel, as in step number 3.

5) After all effective pixels have been read and the real and imaginary values have been calculated, the master machine toggles yet another flag - send flag - to the sending state machine. As the data is being sent, the master machine reinitializes the registers in the datapath, and then waits for the beginning of a new CCD line read, as in step number 2.

6) Upon being signaled to send the calculated data to the PIC microcontroller, the sending machine resets the PIC trigger pin, which will initiate an interrupt routine in the microcontroller, and selects the calculated real value to be sent first through the PIC SPI module.

7) The PIC SPI module indicates that the transfer is complete and signals the sending machine, which in turn selects the imaginary calculated value and prompts it to be transmitted. The sending machine then enters a hold state that is again unlocked when there is new processed data to send.

A graphical demonstration of how the master state machine operates is represented in figure 6, in order to clarify the set of steps just explained.

Because the state machines in this system run in parallel, a set of flags indicate when certain operations in this chain must be executed. This set is composed of 9 flags: beginning, fourth, first, last, final, master done, processed, send and slave done.

- The beginning flag indicates that the reading of a new pixel array is about to start.
- All four flags fourth, first, last and final refer to pixels from the CCD. fourth marks that a new pixel has been output, because it happens every four clock cycles of the CCD clock signal; first points out that the output pixel is the first effective one; last indicates the opposite, that the last effective pixel has been output and final designates that the absolute last pixel has been output, and the pixel array read is complete.
- master done is a flag designed for showing that the SPI master module has successfully acquired the pixel value digital representation, and is ready to be sent to the datapath.
- The processed flag is toggled by the datapath when all the calculations are concluded.
- send is a flag issued by the master machine to order the sending machine to start its operations.
- Exactly like master done, the slave done flag is designed to mark that the SPI slave module has successfully transmitted the data it was issued with, and is ready for another one.

As introduced before, the CCD controller is responsible for the generation of the control signals for that IC, and provide information about which pixel has been output and whether the pixel array reading has just started. Together with two
counters, a state machine times and controls these features. One of them is essentially a timer used in the beginning of the pixel array read, so that the two CCD control signals are timed correctly, according to the values specified by the manufacturer, on table II and figure 7. The purpose of the second counter is to count how many pixels have been output so far, and signal them and indicate some of the pixels position in particular. Because the timer and the pixel counter are timed under different time bases, the timer runs at the main clock frequency in the system - 50 MHz - and the sample counter runs at the maximum master clock frequency allowed by the CCD - 4 MHz. Why these are the chosen frequencies of operation will be approached later. Knowing how this block is composed, these three components work together according to the following steps:

1) One CCD reading cycle starts by resetting the timer and the sample counter, and setting the beginning flag as LOW.

2) The first control signal is then toggled and the state machine enters a holding state, until the timer points out that the first timing requirement has been met. This step is then repeated until all control signals have been toggled correctly for the right periods of time.

3) After this initialization of the sensor IC, the beginning flag is set HIGH, and the state machine enters a long hold state, until all pixels have been output, signaled by the final flag.

4) As the state machine is on hold (step 3), the sample counter is running, updating its flags.

5) When the pixel array has been fully read, the CCD controller state machine resets itself, returning to step 1.

As it has been said, this is solely a finite state machine that handles the slave SPI module. Firstly, while the reset input forces the machine into the idle state, the send flag is a trigger for it to leave that state. slave done is the flag marking the completion of the data transmission by the SPI slave module. Then, this machine directly drives the interrupt signal sent to the PIC microcontroller, and the data selector. This selector is a mandatory feature in this system, as there are two values coming from the datapath that need to be transmitted. It is done using a 2x1 multiplexer, with the named selector. This machine performs the following steps:

1) The machine holds until send flag is toggled.

2) Upon being ready to send data, the machine sets the interrupt trigger to LOW and sets the selector to prompt the calculations real result to the SPI slave module, and waits for the microcontroller to read it.

3) After read, the slave done flag is set, the machine switches to the imaginary value, and waits for the microcontroller to read it once again.

4) It finally acknowledges that it was sent correctly, and enters the idle state once again.

The module containing the datapath is the reason behind why an FPGA was chosen for the circuit, and it is the heart of the calculations performed with this board. The algorithm exposed in section II-B is fully implemented here, completely in hardware, as an accelerated way to run it. As explained before, the algorithm mathematical operation that will be solved is given by equation 10, and it is composed of a series of multiplications and sums of complex numbers. While the multiplication of complex numbers represented in their polar form is a time and resource consuming operation, hard to run in hardware, the multiplication the same numbers represented in their Cartesian form is far easier, as the real and imaginary coefficients are real values, suitable to use in common multiplication architectures, which are much simpler and efficient to implement in hardware. To make use of this property, it is necessary to decompose the algorithm into smaller operations, that are easy to implement and run in this FPGA.

\[
\text{max} = \frac{N}{2\pi} \arg X(k = 1) \text{where} X(k) = \sum_{n=0}^{N-1} x(n) e^{-i \frac{2\pi kn}{N}}
\]

As \( x(n) \) is real and \( e^{-i \frac{2\pi kn}{N}} \) is complex and the product of two imaginary numbers is given by

\[
p = a \times b = (a_r + ia_i) \times (b_r + ib_i)
\]
A simple protocol was therefore created to discriminate the different operations the PIC must perform. In this protocol, every command is issued with either one or three bytes, where the first one always indicates the type of operation to be executed, and the remaining two are optional and carry the data to be sent to one of the peripherals. There are five different operations that can be carried out by the microcontroller: acknowledge that the USB connection has been established, read data processed in the FPGA, send a word to the DAC121S101, send a word to the AD5444 or indicate which of the two CCDs is connected to the daughter board. The corresponding command bytes can be found at Table III. 

The decomposition of the algorithm then results in the sum of a large amount of multiplications of the samples with the corresponding cosine and sine waves. However, since all the pixel samples are acquired sequentially, only two multipliers and two adders are required to do all the calculations. The result of the sum can be stored in a register and added to the result of the next sample multiplication.

Although this is a 16-bit datapath, the adders and accumulators function with 32-bit buses, as a result of the signal extension of the 16-bit signals. These have wider bus entries to be able to store the accumulated values with full precision, even though only 16-bit words will be passed along to the SPI slave module. The 16-bit word resolution with which the results are represented is therefore arbitrary, and is done by applying higher or lower number of shifts to the 32-bit buses. As with the multipliers, because there is only one shift operation that maximizes the resolution in the 16-bit word (why and how do i know *i* the C routine,.) this shift is hardwired into the multiplexer prior to the SPI slave module.

2) **PIC microcontroller:** As mentioned before, the PIC microcontroller in this circuit is held responsible of handling the communication between the several components across the board. It is why it is connected the FTDI IC, the FPGA, the programmable oscillator LTC6903 and the two DACs DAC121S101 and AD5444. The PIC exchanges data with the FT2232HL via its parallel master port module (PMP) while one single SPI module is used to transmit and receive data to the other four components. To select which of these four components the PIC communicates with, there is a chip select line (CS) dedicated to each of them, as required by the SPI protocol.

From a software point of view, this microcontroller must receive commands transmitted via USB by reading them from the FT2232HL FIFO, decode them, and carry them out, should they be either to send data back or to send it somewhere across the circuit board. Those commands are sent from a personal computer, where a dedicated software application is running, and in which a user can freely issue them.

<table>
<thead>
<tr>
<th>Command</th>
<th>Code</th>
<th>Extra Bytes</th>
<th>Return</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledge</td>
<td>0x00</td>
<td>0</td>
<td>0xff</td>
</tr>
<tr>
<td>Scan</td>
<td>0x02</td>
<td>0</td>
<td>2×2-Byte words</td>
</tr>
<tr>
<td>Dark</td>
<td>0x06</td>
<td>2</td>
<td>None</td>
</tr>
<tr>
<td>Gain</td>
<td>0x08</td>
<td>2</td>
<td>None</td>
</tr>
<tr>
<td>CCD</td>
<td>0x0a</td>
<td>0</td>
<td>0x00 or 0x01</td>
</tr>
</tbody>
</table>

In consequence, the software run by the microcontroller is very straightforward to understand, but demands a close look while implementing. This IC executes the following steps, with respect to the sequential nature of the application:

1) The application starts by setting the pins that control the FPGAs master machine as outputs, and immediately resets it so that it will stay on hold, until the read command comes up.
2) The two communication modules are then initialized, along with the interrupt module.
3) With the SPI module initialized and ready to use, the programmable oscillator is configured next, setting it to output a 4MHz clock signal, used by the digital clock manager present in the FPGA.
4) At that time, the microcontroller starts looking for commands that have been sent, by reading data from the FT2232HL FIFO, making use of the PMP module, and reading its status from address number 1 (check table III) cyclically.
5) As soon as there is data in that FIFO, it is read, and the first byte is decoded into the command it issues:
   a) In response to the acknowledge command, one confirmation byte is sent back indicating the connection has been set up.
   b) If the command is to read data sent from the FPGA, interrupts are disabled, and the two global variables storing the most recent real and imaginary coefficients calculated are sent back. These coefficients are updated as soon as they are ready at the FPGA end, by means of an interrupt routine.
   c) When the dark calibration command is issued, the microcontroller reads two other bytes from the FIFO, merges them into one single 16-bit word, and transmits it to the DAC121S101.
   d) As with the dark command, two other bytes are read from the FIFO, merged, and sent to the AD5444.

\[
p = a_r b_r + i a_r b_i + i a_i b_r - a_i b_i
\]
\[
p_r = a_r b_r - a_i b_i
\]
\[
p_i = a_r b_i + a_i b_r
\]
the product of \(x(n)\) and \(e^{-i \frac{2\pi kn}{N}}\) can be decomposed in
\[
p = p_r + ip_i
\]
\[
p_r = x(n) \times \Re(e^{-i \frac{2\pi kn}{N}})
\]
\[
p_i = x(n) \times \Im(e^{-i \frac{2\pi kn}{N}})
\]
finally resulting in
\[
\Re(X(k = 1)) = \sum_{n=0}^{N-1} x(n) \cos \frac{2\pi n}{N}
\]
\[
\Im(X(k = 1)) = \sum_{n=0}^{N-1} x(n) \sin \frac{2\pi n}{N}
\]
e) Finally, the command that asks for which CCD is installed in the daughter board is answered by sending one byte back, that represents either one CCD or the other, according to the logic value present in the CCD check pin.

6) After completing any of the previous commands, the microcontroller returns to step 4, checking for any new incoming commands.

It should be noted that in spite of not being directly accessed by this application, there is an interrupt routine triggered by one FPGA signal that is toggled when calculations have been completed, as described previously. In this routine, two SPI reads from the FPGA are executed, in order to receive the most recently calculated real and imaginary coefficients, in that order.

Because of how the peripheral modules are implemented in the microcontroller’s integrated circuit, configuring them is a simple job. Every peripheral has a number of registers where its configuration bits are stored, and by writing on those registers, it is possible to define how that peripheral operates. The parallel master port configuration is executed once in this application, during the initializations. In general, the most important settings relating to this module are the enabling of the write, read and chip select strobes, their polarity, the number of address bits used, the master operation mode in which it operates and the communication width (either 8 or 16-bit wide). On the other hand, the SPI module is reconfigured several times during the course of the program, because each of the peripherals on the circuit board receives data under different serial clock polarity and phase conditions. However, most configurations stay the same, and are defined upon that module’s initialization. Those include the serial clock frequency, being set as the master device, the communication width (also either 8 or 16-bit), disabling framed mode and the sampling time. Serial clock phase and polarity are configured when they must be changed. Finally, the interrupt controller is also configured once during the initializations, and the features defining it are set by indicating its priority level, and on which external signal edge should the corresponding interrupt routine be triggered.

IV. THE RESULTS

Provided that a large number of tests all the tests were carried out, it is supposed that the results observed from those tests resemble the expected ones, or the outcomes of the FPGA hardware simulations. In this section we will approach solely the most important results observed during the testing phase, as many tests and measurements were performed during the construction of this board.

Having the CCD controller designed in VHDL for the FPGA, it was simulated, and this simulation was run under the same circumstances as if it was inserted with the rest of the hardware. The timings between signal pulses were measured and are in table IV. The timings are purposely set to be slightly higher than the minimum specified for the Toshiba CCD (see table II), so that there would be a margin that assured the good functioning of that sensor. Figure 9 features the oscilloscope view of those signals, and after implementing the hardware on the FPGA, the measured timings equal those in the simulation, which translates in the good functioning of the CCD.

![Fig. 9. CCD control signals after implementation. Note that channel 1 is the FPGA negated ICG signal, channel 3 in the SH signal, and channel 4 is the CCD master clock.](image)

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Expected</th>
<th>Simulation</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICG pulse delay</td>
<td>1100</td>
<td>1099.97</td>
<td>1108</td>
</tr>
<tr>
<td>Pulse timing of ICG and SH</td>
<td>180</td>
<td>179.98</td>
<td>174</td>
</tr>
<tr>
<td>SH pulse width</td>
<td>1080</td>
<td>1079.98</td>
<td>1079</td>
</tr>
</tbody>
</table>

The successful tests run to the CCD controller module opened up the possibility of testing the entire system. It was firstly performed using the debug build for both the PIC and the FPGA, where all pixel samples are acquired, but are stored, instead of processed. In the beginning they are stored in a FIFO memory structure in the FPGA, then in a short integer array in the PIC and late in an array at the end C# application. The results from the sensor read are presented in figure 10, and from the data that application provides, some aspects should be noted.

![Fig. 10. CCD output example with LASER input (PC view).](image)
Firstly, the refresh period for each data read in this debugging mode is far higher than the CCD line reading period (1 in figure 10). This is clearly due to the limits imposed by the transmission rate at which data is transferred from the FPGA to the microcontroller, and from there to the PC end application. Due to this latency, transmitting the raw samples from the CCD sensor, unprocessed, would require extremely high transmission rates and different, higher end integrated circuits to do so. It, in fact, proves that the important information must be extracted from this large amount of data, so that it can be transmitted within an acceptable time frame, which is exactly what is achieved with the FPGA.

Secondly, not all of the end features in this end application were used. Particularly, the "Set Frequency" command was not used, even though it was originally designed to set the frequency of the programmable oscillator. This feature in thus unnecessary, as the oscillator is configured once in the beginning of the microcontroller’s routine, and is set so that the ADC can convert the data at its maximum sampling rate. Changing this frequency would affect the sampling rate largely, which could compromise the functioning of the whole FPGA system.

Figure 10 represents an example of the output signal from the CCD that will be processed, in debug view, while figure 11 does not present the output samples themselves, it shows the received real and imaginary calculated coefficients(numbers 2 and 3 in picture 11, respectively), and the respective position of the light maximum on the CCD (number 4 in picture 11). That maximum calculation is performed by the application, as it is not a feature of the board in this project. First of all, It should be noted from picture 10 that the light curve acquired by the CCD is not a perfect gaussian curve but instead an approximation of it. Its imperfections result from various factors, namely the nature of the LASER diode itself, its lens or impurities and scratches on the CCD protection glass. This signal can though be interpreted as a gaussian signal with noise, that ultimately affects the calculations. Furthermore, the observed results can only be interpreted empirically, due to the fact that there is no way to specifically direct the light source to one CCD pixel in particular and have a deterministic input to know exactly where the true maximum would be located. This setup simply allows to observe the approximate position of the maximum light position, and compare it with the results from the FPGA and end application calculations.

V. Final remarks

The results reached with this implementation, and the results are very encouraging. They show that the system was able to operate in its entirety, at full speed, and present us with an accurate information about the first DFT coefficient real and imaginary parts, that were used in the calculation of the overall position where the laser signal at the CCD sensor. Therefore, it is safe to say that with the optical setting that was created and the system that was build, the objectives for this project were completely met.

In the end, this is clearly a very interdisciplinary project, as the knowledge that was required to develop it is extremely varied, from programming, to hardware development and printed circuit board design. It is hoped that the innovation presented by this system can be used in future speaker response measurements, either academically or industrially, to represent that response as accurately as possible.

REFERENCES