Design of a Three-Phase Micromotor Driver for use in Optical Coherence Tomography

Scott Bernard
scott.bernard@sfu.ca
301314636

Undergraduate Thesis
ENSC 499
Simon Fraser University
May 2021

Academic Supervisor: Dr. Pierre Lane, P. Eng.
Technical Supervisor: Mr. Geoffrey Hohert
Committee Member: Dr. Anita Tino, P. Eng.

APPROVAL

Name: Scott Bernard

Degree: Bachelor of Applied Science (Honours)

Title of Thesis: Design of a Three-Phase Micromotor Driver for use in Optical Coherence Tomography

Glenn Chapman

Dr. Glenn Chapman, P.Eng
Director, School of Engineering Science

Examining Committee:

Dr. Pierre Lane, P.Eng (Academic Supervisor and Chair)
Associate Professor of Professional Practice, School of Engineering Science

Mr. Geoffrey Hohert (Technical Supervisor)
BC Cancer Research Centre

Dr. Anita Tino, P.Eng
Lecturer, School of Engineering Science

Date Approved: May 6, 2021

ABSTRACT

Optical coherence tomography (OCT) is a photonics-based medical imaging technique under active research within the BC Cancer Research Centre's (BCCRC) Optical Cancer Imaging Lab (OCIL). A current focus within the OCIL is the development and applications of micromotor-driven catheter probes for OCT imaging. The current micromotor driver used in the OCIL's research is independent from the imaging system and suffers from frequency mismatches between imaging and motor rotation speeds, resulting in instability of the output images. Image instability limits clinical applications of the micromotor probes that require analysis and comparison between consecutively acquired images.

To eliminate rotational instabilities from the micromotor catheter probes we design and construct a micromotor driver that is frequency-matched with the OCT imaging system. The new micromotor driver generates three-phase motor drive waveforms through updating a digital-to-analog converter (DAC) synchronously with the OCT imaging sample rate. DAC outputs are fed through analog filtering and current amplification circuitry to allow for smooth rotation of the micromotor. The new micromotor driver was designed for control through software developed within the OCIL.

To validate the new micromotor driver we use a printed paper phantom with regularly-spaced azimuthal fiducial markings to compare the imaging performance of the current micromotor driver to the performance of the new micromotor driver. We find that the new micromotor driver solves the frequency-mismatch issue in the current micromotor imaging system and allows for stable imaging with micromotor probes.

ACKNOWLEDGEMENTS

I would like to thank Dr. Anthony Lee and Dr. Pierre Lane for their support and guidance on this project, and Eric Brace for his help testing and setting up the optical equipment used for OCT imaging.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 OPTICAL COHERENCE TOMOGRAPHY

Optical coherence tomography (OCT) is a photonics-based medical imaging technique under active research within the BC Cancer Research Centre's (BCCRC) Optical Cancer Imaging Lab (OCIL). OCT allows for medical imaging through the measurement of surface and sub-surface reflection and scattering of light from an illuminated tissue sample. In essence, an OCT system operates on similar principles as an ultrasound, however OCT systems use laser light instead of sound for imaging [1]. The use of laser light allows OCT to achieve high-resolution images as image resolution is limited only by the optical bandwidth of the laser source [2]. For typical applications, OCT image resolution is in the order of 10 micrometers [3]. This work focuses on the OCIL's catheter OCT imaging systems, for which a thin (sub-1mm to 5mm diameter), rotating fiber-optic catheter probe directs a laser source to perform radial imaging sweeps within a sample. As the beam is swept over the tissue, OCT imaging is performed along the one-dimensional axial line (*A-line*) in the direction of the beam. A full "frame" (also referred to as a *B-scan*) in the output is constructed from a fixed number of A-line samples corresponding to one full radial sweep of the probe. The data processing software for image construction was developed by the OCIL. A block diagram showing the major physical components of the OCT system is shown in Figure 1. The OCIL's catheter OCT imaging systems are used in both *in-vivo* and *ex-vivo* applications.

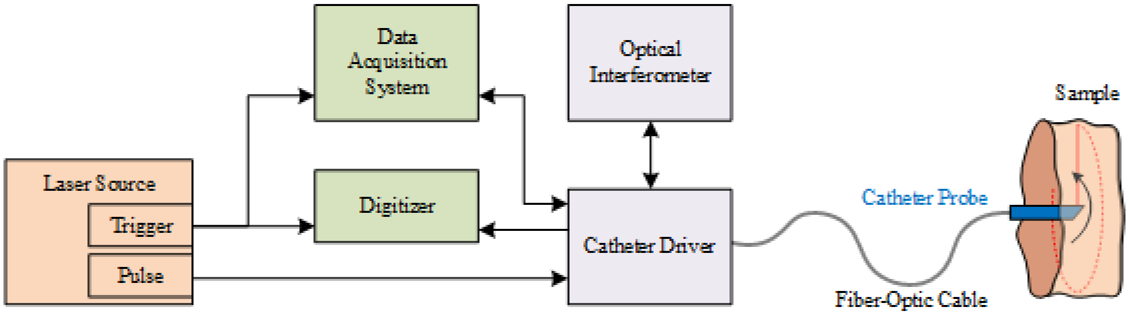


Figure 1: Major components of the OCIL's OCT imaging system.

1.2 MICROMOTOR CATHETERS

The rotational motion enabling azimuthal sweeping of the imaging laser is generated in one of two ways: the first sweep method involves attaching an angle-polished segment of optical fiber to the tip of a catheter probe to redirect the laser beam. In this method, a motor within the catheter housing rotates the entire fiber-optic cable and connected probe to perform azimuthal sweeping as shown in Figure 2(a). A second method involves the use of a micromotor within the catheter probe to redirect the beam as shown in Figure 2(b). The use of a micromotor allows for greater scanning speeds and eliminates the need to rotate the fiber-optic cable, thereby eliminating image distortions caused by rotational friction along the cable shaft. The use of micromotor catheters for OCT imaging is a current research area within the OCIL and is the main focus of this work.

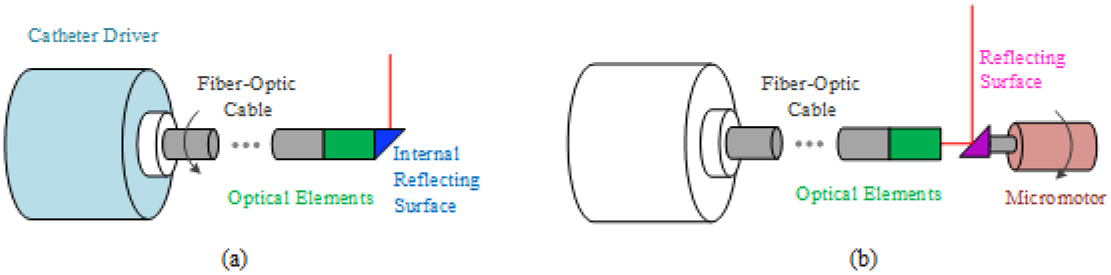


Figure 2: Methods of generating rotational motion of rotary-pullback devices: (a) Rotation is driven by a motor within the catheter housing with the entire fibre-optic core being rotated. (b) Rotation is generated by a micromotor within the catheter probe.

1.3 STABLE IMAGING

To obtain a stable image output from the OCT system when using micromotor probes there must be matching between the A-line clock frequency, image acquisition frequency, and the driving frequency of the micromotor. The current OCIL OCT system is designed such that there is matching between the A-line clock and image acquisition frequencies, however matching of the micromotor frequency is not guaranteed. Small differences between the motor and image acquisition frequencies result in azimuthal "rolling" of the output frames. A visualization of the cause of the rolling effect is shown in Figure 3 and explained below.

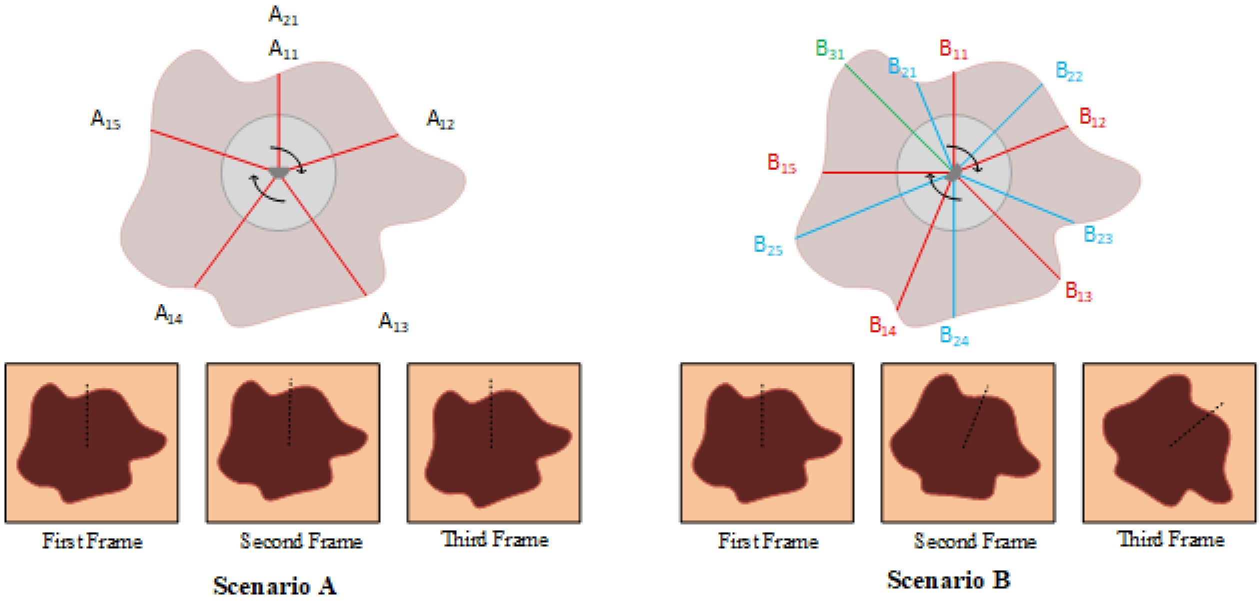


Figure 3: Visual depiction of the cause of azimuthal "rolling" in image output due to frequency mismatching between micromotor rotation frequency and imaging frequency.

Figure 3 shows the A-line imaging locations for two simplified scenarios: in scenario A there is matching between the micromotor rotation frequency and the imaging frequency, while scenario B involves frequency mismatching. In both scenarios the imaging software expects five A-lines per frame. In scenario A, A-lines for the first frame are sampled in locations A_{11} to A_{15}. Due to matching between the micromotor rotation frequency and A-line sample frequency, the location of the first A-line sample for the second frame, A_{21}, is the same as the location of the first A-line sample for the first frame, A_{11}. Thus, the output frames are stable. It can be seen that to achieve image stability, the micromotor rotation frequency must be an exact integer divisor of the A-line sample frequency.

In scenario B there is a frequency mismatch. Here, the first frame is constructed from samples in locations B_{11} to B_{15}. Due to the frequency mismatch, the location of the first A-line sample for the second frame, B_{21}, is *NOT* the same as the location of the first A-line sample for the first frame, B_{21}. The result is instability in the output image, with the output image experiencing azimuthal rolling at a rate of $\varphi B_{21} - \varphi B_{11}$ radians per frame.

The current micromotor catheter imaging setup within the OCIL uses a commercial motor driver that is independent of the imaging system. As a result, frequency matching cannot be guaranteed and the output of the current setup experiences azimuthal rolling. Image post-processing can be applied to correct for the rolling effect, however such techniques can introduce image distortions as well as introduce a computational overhead that limits the system's ability to be used in real-time imaging applications.

A specific application of interest within the OCIL is optical coherence tomography angiography (OCT-A). OCT-A applies motion-detection algorithms to standard structural OCT imaging in order to visualize blood flow within vascular tissue [4][5]. While there are many ways in which the motion detection is performed, the general methodology behind OCT-A algorithms is to detect blood flow through comparisons between consecutive OCT frames acquired at the same location [4][6]. OCT-A therefore requires excellent image stability between consecutive frames to obtain accurate motion estimations.

1.4 PROJECT SCOPE

In order to allow the OCIL micromotor OCT setup to be used for real-time imaging applications or for reliable OCT-A, the frequency matching issue will be solved at a hardware level. This thesis work describes the development and validation of a motor driver capable of smoothly running a micromotor at integer divisors of an A-line scanning frequency to allow for stable imaging.

At the time of this project's proposal the scope of this project included the development and fabrication of a printed circuit board (PCB) containing the micromotor driver hardware, with the micromotor driver being in the form of a box allowing for user controls through pushbuttons and dials. During development, the project scope changed to instead be for integration within the OCIL's micromotor imaging system (as opposed to an external controller). The design and fabrication of a micromotor driver PCB was then removed from this project's scope as PCB design and fabrication will be done after development of the remaining components of the OCIL's micromotor imaging system are completed.

2 MICROMOTOR DRIVER REQUIREMENTS

The existing micromotor OCT system within the lab used three-phase brushless DC (BLDC) micromotors (SBL04-072, Namiki Precision of California, Inc.) driven by an independent commercial motor driver (SOD12ST, Namiki Precision of California, Inc.). The micromotor driver developed in this work was designed to offer equivalent motor control functionalities to the SOD12ST, with extended functionalities included to meet the needs of the OCIL. The following sections outline the functional requirements of the micromotor driver.

2.1 PERFORMANCE

The micromotor driver must:

(i) Be capable of driving the micromotor synchronously with an external A-line clock signal.

(ii) Be capable of driving the micromotor independently of an external A-line clock signal.

(iii) Support driving the micromotor at rotation frequencies up to 500Hz.

(iv) Smoothly accelerate the micromotor up to the final drive frequency.

(v) Allow for a minimum output drive waveform amplitude of 3V (6Vpp).

(vi) Allow for clockwise and counter-clockwise rotation of the micromotor.

2.2 COMMANDS

The micromotor driver must:

(i) Support the following user controls::

(a) Start / Stop.

(b) Selection of synchronous or asynchronous operation.

(c) Setting of synchronous rotation frequency (A-lines per B-scan).

(d) Setting of asynchronous rotation frequency.

(e) Setting of acceleration rate.

(f) Setting of output voltage amplitude.

(g) Setting of rotation direction.

(ii) Allow for OCIL software to query the micromotor driver state (idle, accelerating or running).

2.3 INTERFACING

The micromotor driver must:

(i) Allow for user control through a Universal Asynchronous Receiver/Transmitter (UART) serial communication interface.

(ii) Allow for user control of main driver functionalities (start/stop, frequency selection) through hardware pushbuttons (to allow for use of the driver while the final OCIL micromotor system is under development).

3. SYSTEM INFORMATION

3.1 OCT IMAGING SYSTEM

The main electrical components in the OCT imaging system are a laser source (AXP50125, Axsun Technologies, Inc.) a signal digitizer (ATS 935X, Alazar Technologies, Inc.), a data acquisition (DAQ) system (USB-634X, National Instruments Corp.), and a rotary pullback drive (RPD) used to control OCT imaging probes (developed by the OCIL). A block diagram showing connections between the components of the OCT imaging system is shown in Figure 4.

The AXP50125 is a swept-source laser designed for use in OCT systems. The AXP50125 generates a trigger pulse at the start of an A-line sample, and transmits the trigger pulse to the signal digitizer and DAQ. The micromotor driver developed in this work used the trigger pulse generated by the AXP50125 to synchronize the rotation of the micromotor with the A-line scanning frequency. Figure 4 shows the location where the AXP50125 trigger signal is pulled from the OCT system to be used by the micromotor driver.

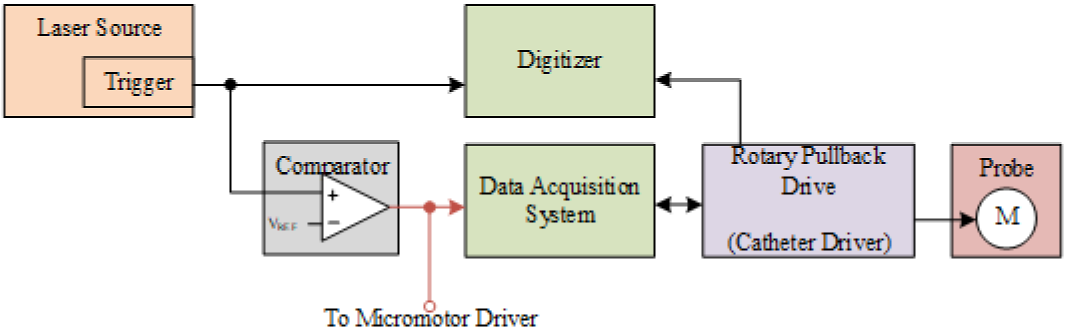
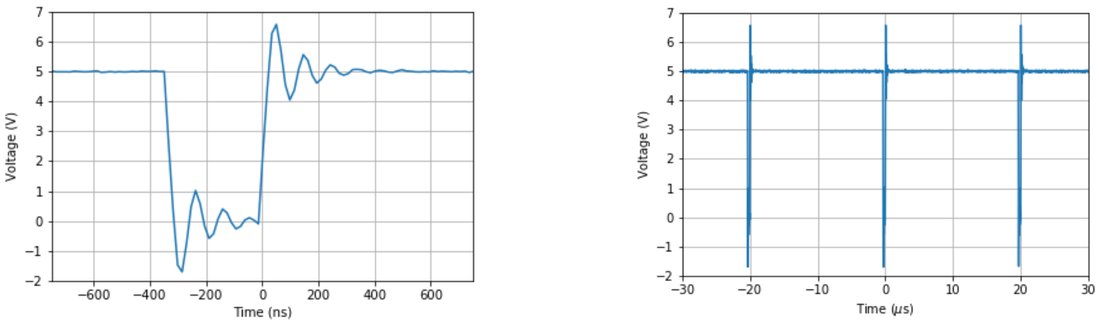


Figure 4. Electrical components of OCT imaging system.

3.1.1 A-LINE TRIGGER SIGNAL

The A-line trigger signal is in the form of a 5V, ~335ns active-low pulse. The OCIL uses lasers producing A-line pulses at rates of either 50kHz or 100kHz. An oscilloscope screenshot showing the electrical characteristics of the A-line trigger signal provided to the micromotor driver is shown in Figure 5.



(a) (b)

Figure 5. A-line trigger profile. (a) Single trigger pulse. (b) Multiple trigger pulses from 50kHz laser.

3.2 BRUSHLESS DC MOTORS

The micromotor probes in the OCIL are built around SBL04-072 BLDC micromotors (Namiki Precision of California, Inc.). This section briefly describes the theory of operation of BLDC motors.

A BLDC motor is composed of two main parts: a rotor and a stator. The rotor of a BLDC motor is a permanent magnet connected to the motor shaft, and the stator consists of stationary current-carrying coils fixed to the motor body [7]. The current-carrying coils in the stator are connected in a *wye* configuration. Figure 6 shows a simplified mechanical and electrical schematics of a BLDC motor.

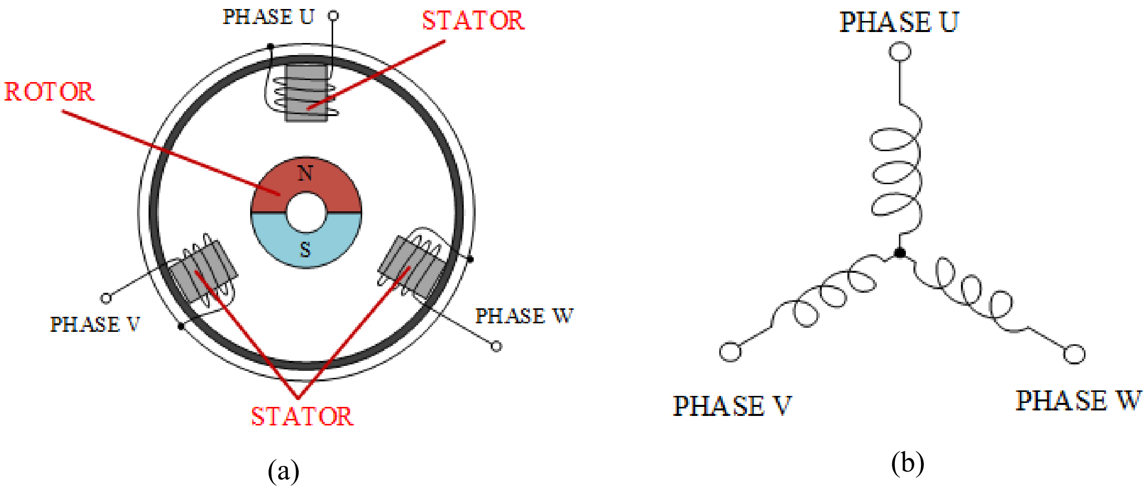


Figure 6. Simplified BLDC motor diagrams. (a) Simplified mechanical diagram of three-phase BLDC motor containing three stator coils. Diagram based on [8]. (b) Simplified electrical schematic of three-phase BLDC motor.

Passing a current through coils in the stator will generate a magnetic field and will cause the rotor to move in order to align with the resultant magnetic field. The rotation of a BLDC motor is achieved by varying the magnitude and direction of the magnetic fields generated by the stator coils through varying the magnitude and direction of the applied current.

To achieve smooth rotation of the BLDC motor shaft, the current profiles of the three motor phases must be sinusoidal, with each phase sinusoid being 120° apart [9]. Sinusoidal current profiles can be achieved through varying the voltage applied to each phase through pulse-width-modulation (PWM) or analog control methods [9]. As smooth, uniform rotation is desired for OCT imaging with micromotor probes, analog control methods were selected for the development of this work.

4 CORE DESIGN METHODOLOGY

4.1 FREQUENCY SYNTHESIS

The major design choice involved in the development of the micromotor driver was the selection of how the three-phase drive waveforms were generated. As the core aim of the micromotor driver is to generate sinusoidal signals at integer divisions of an external (A-line) clock, the design problem can be interpreted as a problem in signal processing: given an input frequency, synthesize an output frequency that is an integer division of the input.

Three common methods of frequency synthesis include the use of phase-locked loops (PLL), direct digital synthesis (DDS) devices and real-time programming of digital-to-analog converters (DAC) [10]. Due to the requirement to easily program the frequency via a microcontroller, DDS or real-time DAC methods were preferable for this project. The method of programming a DAC was selected for the motor driver design as it ensures alignment with the laser pulse frequency while significantly reducing development costs compared to commercially available DDS devices suitable for this application. Generating motor drive sinusoids by updating a DAC through a sine look-up table in synchronization with the laser A-line clock provides a robust solution to the problem of frequency mismatches. In a mechanical analogy, our laser and motor drive signals can be thought of as two rotating shafts for which we wish the "motor drive" shaft to spin at an integer division of the "laser" shaft. In this analogy, the real-time DAC method solves the problem by using gears to connect the two shafts.

4.2 HARDWARE OVERVIEW

The micromotor driver was designed around three main stages: a "frequency synthesis" stage consisting of a microcontroller and three DACs (one for each phase of the motor drive); a "filtering" stage used to produce smooth drive waveforms; and a "current amplification" stage used to source the current required by the micromotor. As shown in Figure 7, the circuitry controlling the three phases will be identical. The only difference between phases is the corresponding sine look-up tables being calculated with 0°, 120° or 240° offsets.

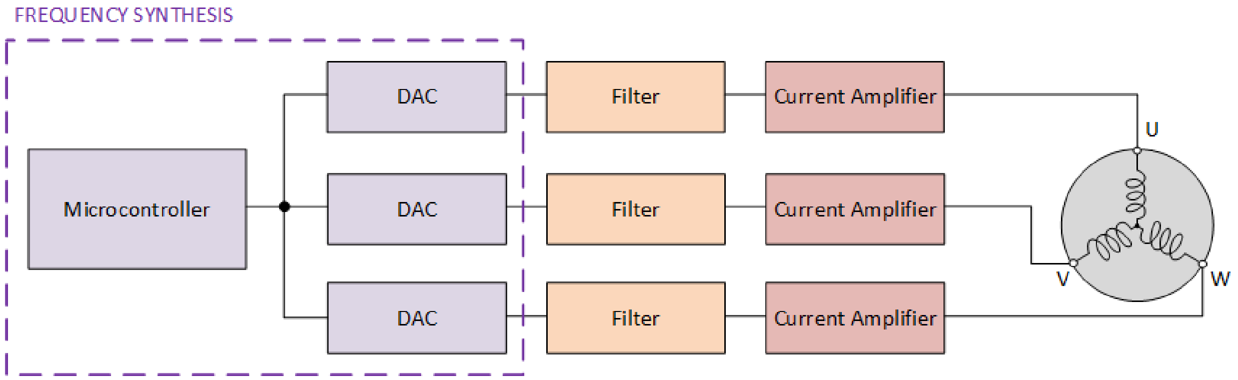


Figure 7: Overview of main micromotor driver components.

4.2.1 FREQUENCY SYNTHESIS STAGE

4.2.1.1 MICROCONTROLLER

The microcontroller is responsible for programming the waveform-generation DACs, interfacing with user commands, and running the driver firmware processes. Due to the use of PIC-series microcontrollers in other OCIL projects, the PIC24FJ1024GB610 (PIC24) microcontroller (Microchip Technology Inc.) was used for development of the driver. Firmware development was performed using Microchip's MPLAB X integrated development environment (IDE).

The PIC24 is a 16-bit, 100-pin microcontroller suitable for embedded applications. Table 1 lists relevant PIC24 features and parameters.

Name	Value
Max MCU Speed	32 MHz
Program Memory Size	1024 KB
Operating Voltage Range	2V to 3.6V
UART Channels	6
SPI Channels	3
Timers	5 (16-bit)
Direct Memory Access Channels	8

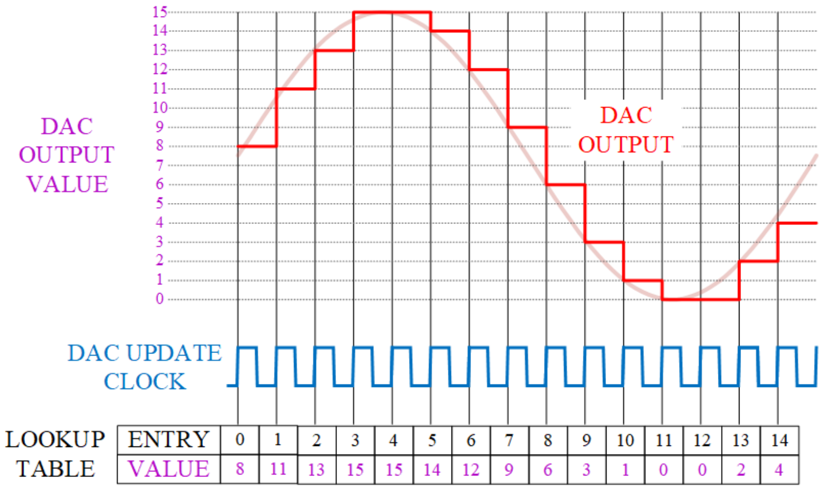
Table 1. Relevant features and parameters on the PIC24FJ1024GB610.

4.2.1.2 DIGITAL-TO-ANALOG CONVERTERS

DACs are programmable devices that allow for the generation of analog voltage outputs, with the outputs of DACs being limited to discrete fractions of a given reference voltage. To program a DAC, a microcontroller (or other device) supplies the DAC with an integer data value, D_{IN}. The output, V_{OUT}, of an n-bit DAC supplied with a reference voltage V_{REF} is then:

$$V_{OUT} = \frac{D_{IN}}{2^n} V_{REF} \tag{1}$$

By using a look-up table to iteratively update the DAC output value to follow a sinusoidal pattern, the DAC can be used to synthesize an approximate sine wave output. The frequency of the DAC's output sine wave is dependent on the frequency at which the DAC is updated and the number of entries in the sine look-up table. Figure 8 shows an example of a full DAC update cycle for a 4-bit DAC and a 15-entry lookup table. In this example, the DAC is updated on the rising edge of the DAC update clock.



ENTRY	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
VALUE	8	11	13	15	15	14	12	9	6	3	1	0	0	2	4

Figure 8. Sine-wave generation through iteratively updating the output of a DAC through a lookup table. Updates are performed synchronously with a clock signal.

The i-th entry of an M-entry sine look-up table with phase offset φ, as implemented for an n-bit DAC is found as:

$$\text{entry}[i] = \text{int}\left(\frac{2^n - 1}{2}[\sin(2\pi(\frac{i}{M} + \phi)) + 1]\right), \qquad 0 \leq i < M \tag{2}$$

The frequency f_{MOTOR} of the output waveform generated by iterating a DAC through an *M*-entry look-up table, with DAC update frequency $f_{ALINE}$is found as:

$$f_{MOTOR} = \frac{f_{ALINE}}{M} \tag{3}$$

Further control over the output frequency can be achieved by first pre-scaling the DAC update frequency by a factor *d,* resulting in:

$$f_{MOTOR} = \frac{f_{ALINE}}{dM} \tag{4}$$

In the context of the micromotor driver, f_{ALINE} represents the laser A-line frequency. Through unit analysis it can be seen that d represents the number of A-line clocks per DAC update, and *M* represents the number of DAC updates per revolution of the micromotor. The product *dM* therefore represents the number of A-line clocks per revolution, or the number of A-lines per B-scan.

$$\frac{ALINES}{BSCAN} = dM \tag{5}$$

4.2.3 FILTERING STAGE

As the output of the waveform-generation DACs are not pure sinusoids, a filtering stage is required to produce a smooth motor drive signal. The DAC output can be represented as the summation of a pure sine wave and an error signal, err(ω, t), as shown in Figure 9. The error signal represents the quantization noise of the DAC conversion and is a function of the DAC resolution n, sine table size M, and driving frequency ω.

$$\mathrm{DAC}(n, M, \omega, t) = \mathrm{A}[\sin(2\pi\omega t + \varphi) + \mathrm{err}(n, M, \omega, t)] \qquad (6)$$

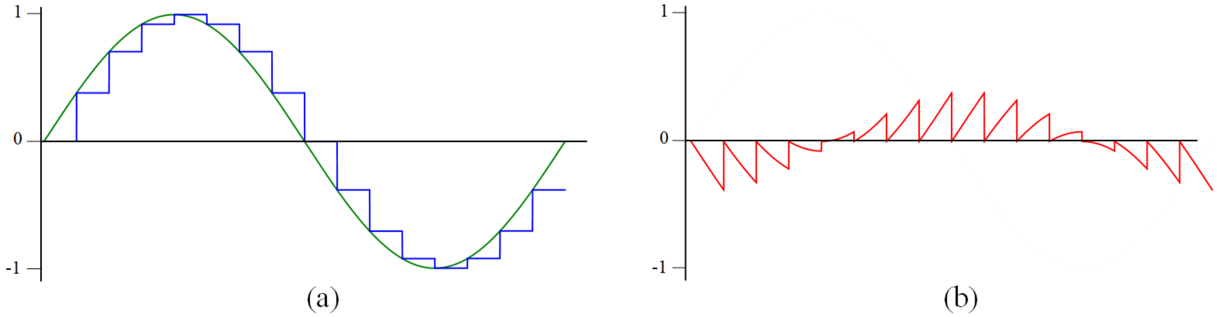


Figure 9. (a) Normalized DAC-representation (blue) and underlying pure sinusoid (green). (b) Normalized error signal between DAC output and pure sinusoid. It should be noted that the form of the error signal is a function of the DAC update frequency and lookup-table size.

In the frequency domain, the error signal mainly consists of high-frequency components that can be eliminated through the use of a low-pass filter. As typical DACs have output resolutions ranging from 8 to 16 bits (with DACs considered for this project having 12-bit or greater resolutions), it can be seen that the main contributing factor to the magnitude of the error signal is the number of entries in the DAC sine look-up tables. Decreasing the size of the sine look-up tables increases the amount of filtering required to eliminate the high-frequency error signals and produce a smooth sinusoidal output. However, *increasing* the size of the sine look-up tables also increases the memory requirements of the microcontroller. There is a tradeoff between quantization noise and memory requirements: larger look-up tables will result in less noise with greater memory requirements, while smaller look-up tables will result in more noise with lesser memory requirements.

Additionally, it is important to note that a phase shift is introduced between the input and output of the filter (equal to 45° per pole at the filter cutoff frequency) and component tolerances can result in misalignment of phases when using higher-order filters, if the filter cutoff frequency is too close to the motor output frequency [11].

4.2.4 CURRENT AMPLIFICATION STAGE

A current amplification stage is included at the output of the micromotor driver to prevent the filtering circuitry from being required to source the current drawn by the micromotor.

5 PROTOTYPE DRIVER DESIGN

A prototype micromotor driver was designed to verify the feasibility of real-time programming of a DAC to achieve synchronous rotation of the micromotor. The prototype micromotor driver was capable of accelerating and running the micromotor at a drive frequency set by the user. A schematic diagram showing the prototype driver is presented in Figure 10. The prototype driver was controlled through the buttons and the slider on the Explorer 16/32 board, with buttons for starting and stopping the motor, and the slider used for setting the motor frequency.

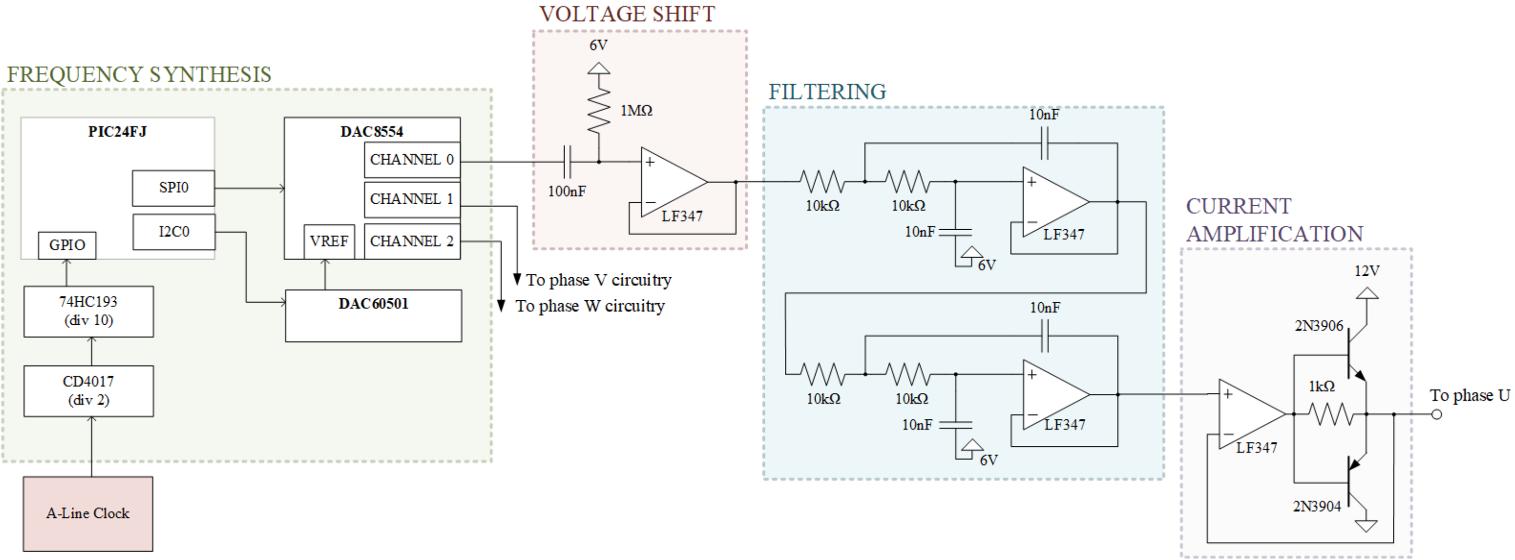


Figure 10. Overview of prototype motor driver. The voltage shifting, filtering, and current amplification stages for phases V and W are identical to phase U.

5.1 FREQUENCY SYNTHESIS

The prototype driver performed frequency synthesis through varying sine lookup table sizes while keeping the A-line pre-scale division (as discussed in Equation 4) constant. The prototype driver design was based around the DAC 8 Click board (MikroElektronika d.o.o.). The DAC 8 Click board was chosen for initial prototyping as it is easily compatible with the mikroBus ports on Explorer 16/32 developer board. The DAC 8 Click features a DAC8554 (Texas Instruments, Inc.): a 16-bit, four-channel DAC programmed through the Serial Peripheral Interface (SPI) communication protocol. The DAC 8 Click also includes a DAC60501 (Texas Instruments, Inc.) chip that was used to set the reference voltage for the DAC8554 in order to specify the output amplitude of the motor drive waveforms.

The prototype driver used the PIC's MCU to program the DAC8554 over SPI. Using the DAC8554, the prototype driver was able to update all three waveform-generation DAC channels at a maximum rate of approximately 5.5kHz. The prototype driver used external hardware counters (a CD4017 and 74HC193, Texas Instruments, Inc.) to prescale the 100kHz A-line clock signal by a factor of 20 to accommodate the maximum update rate of the DACs, resulting in a final DAC update rate of 5kHz. To achieve different drive frequencies, the prototype driver varied the length of its sine look-up tables, with a new look-up table created each time a new frequency was set by the user.

5.2 FILTERING

With the DAC update rate of the prototype driver fixed at 5kHz, the prototype driver had sine look-up tables with as few as ten entries to allow for generation of 500Hz drive waveforms. Small look-up table sizes resulted in blocky outputs of the waveform-generation DACs. A fourth-order active low pass filter was designed using a Sallen-Key topology to produce smooth motor drive signals from the blocky outputs of the waveform-generation DACs.

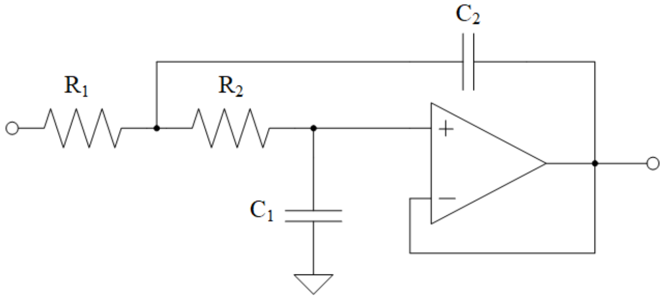


Figure 11. Second-order Sallen-Key low-pass filter.

The prototype driver's filter stage consisted of two cascaded second-order Sallen-Key low pass filters. Using component labels as presented in Figure 11, the cutoff frequency, f_C, of a single second-order Sallen-Key low pass filter is found as:

$$f_C = \frac{1}{2\pi\sqrt{R_1 R_2 C_1 C_2}} \tag{7}$$

Using $R_1 = R_2 = R$ and $C_1 = C_2 = C$ we have:

$$f_C = \frac{1}{2\pi RC} \tag{8}$$

The filters in the prototype driver were designed with R = 10kΩ and C = 10nF to achieve a cutoff frequency of $f_C \approx$ 1.5kHz. Initially, the prototype driver cutoff frequency was set to half the 10 DAC update frequency (2.5kHz), however it was found that this did not provide sufficient filtering to produce a smooth sine wave when the prototype driver was generating an output frequency of 500Hz. f_C = 1.5kHz was selected to achieve sufficient smoothing of output waveforms while allowing for minimal signal attenuation and phase shifting.

5.3 CURRENT AMPLIFICATION

The prototype driver used an active class-B push-pull current amplifier in its output stage, following a design presented by Analog Devices [12]. The purpose of the transistors in the current amplifier design is to source current to the load, while the op-amp serves to reduce crossover distortion between the transistors by maintaining one of the transistors in conduction through negative feedback. The resistor in this design is not required, however it allows for further reduction of crossover distortion [12]

5.4 REMARKS

The prototype driver design successfully demonstrated the feasibility of using DACs to produce micromotor drive waveforms at an integer division of the A-line frequency to achieve stable imaging. However, there were several flaws in the prototype design.

First, the use of the DAC8554 necessitated programming updates to the three drive phases sequentially, limiting the rate at which the DACs could be updated and reducing the number of entries allowable in the DAC sine look-up tables. Reducing the look-up table size increases the amount of filtering required to produce smooth drive waveforms and increases both the component count and analog complexity of the driver circuit. Additionally, as the DACs were programmed by the PIC MCU, constant MCU attention was required when the driver was running, limiting the MCU's ability to handle user input commands. Second, the use of external clock scaling hardware increases the component count and does not allow for easily changing the pre-scale value. Finally, the use of an active Class-B output amplifier introduces a feedback path connecting the motor coils to the input of an op-amp. Feedback from the motor can (and did) cause oscillations in the output of the amplification stage, resulting in non-uniform rotation of the micromotor. Non-uniform rotations of the micromotor caused a "semi-stable" output image: the image did not experience azimuthal drift but instead "wobbled" about its stable position. Additionally, oscillations in the output stage resulted in extra noise in the motor drive waveforms which can cause unwanted heating of the motor. A second, final driver design was created to address the issues present in the prototype design.

6 FINAL DRIVER DESIGN

The final micromotor driver was designed to maximize the allowable DAC update frequency to increase sine look-up table sizes and reduce the amount of filtering required to produce a smooth output waveform. Additionally, the final micromotor driver was designed to utilize features on the PIC24 to allow for greater programmatic control over driver properties. An overview of the final driver hardware components is shown in Figure 12, and a picture of the physical driver is shown in Figure 13. A detailed schematic showing the hardware pin connections is provided in Appendix A. Filtering capacitors (47uF electrolytic and 0.1uF ceramic) were added in parallel with the 12V DC input to reduce power supply noise.

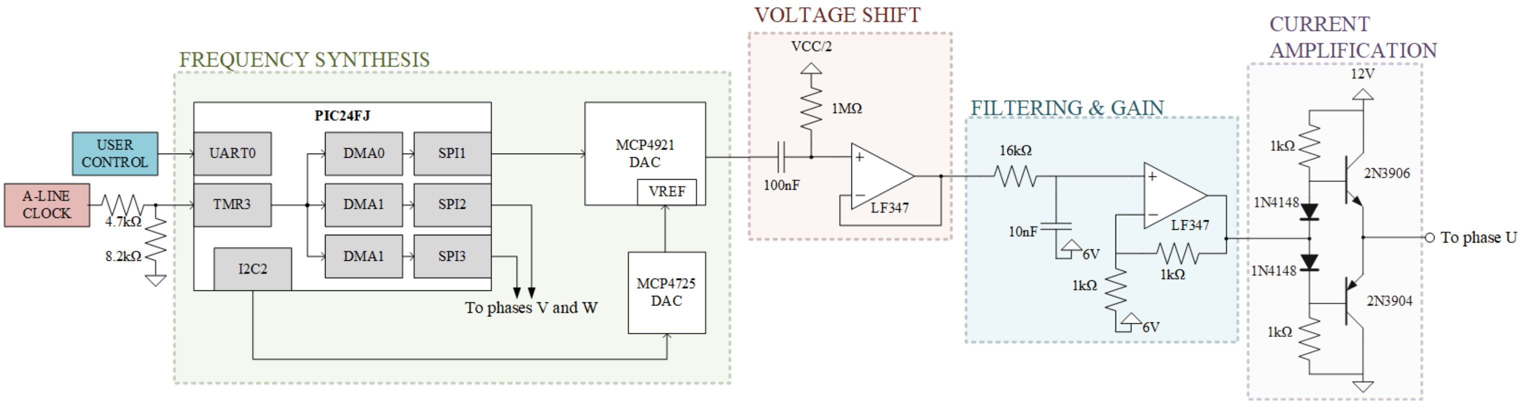


Figure 12. Overview of final motor driver hardware components. The voltage shifting, filtering & gain, and current amplification stages for phases V and W are identical to phase U.

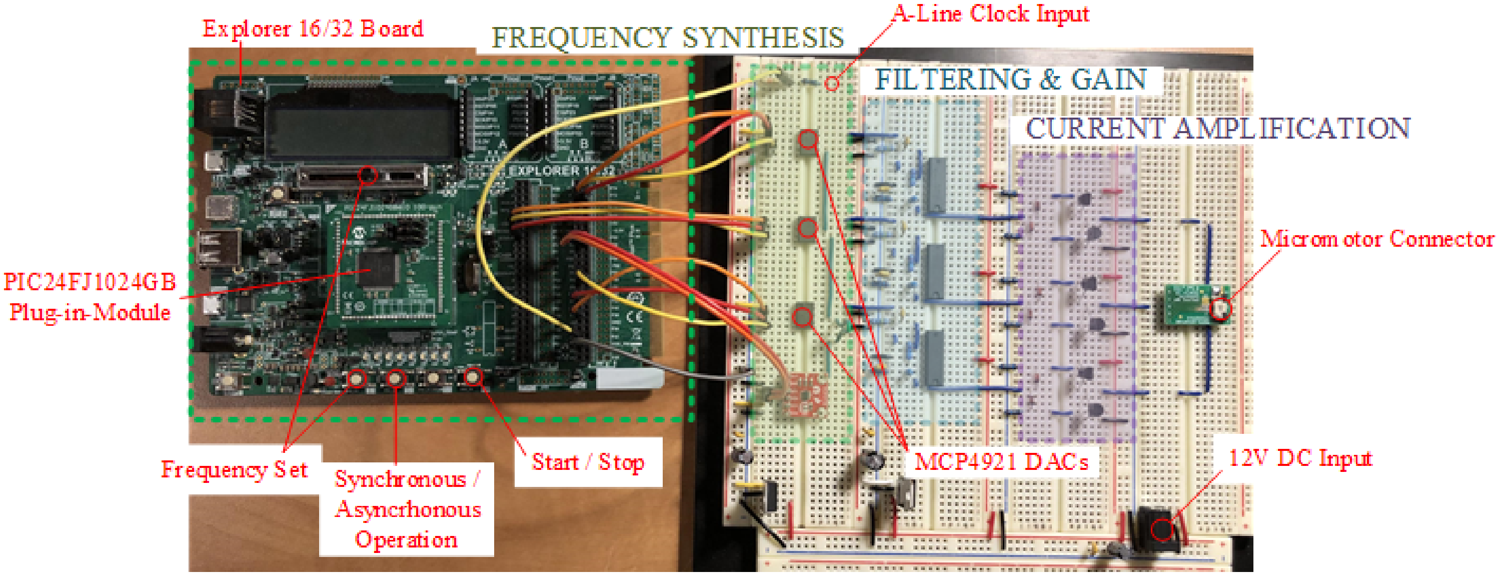


Figure 13. Overview of physical implementation of micromotor driver.

6.1 FREQUENCY SYNTHESIS

The micromotor driver performs frequency synthesis through programming a MCP4921 DAC (Microchip Technology Inc.), with one DAC per motor phase and each DAC programmed through its own SPI port on the PIC24. The MCP4921 is a single-channel DAC with 12-bit resolution. The MCP4921 DACs are supplied with a reference voltage generated by a MCP4725 DAC (Microchip Technology Inc.), programmed through the I2C2 port on the PIC24. The micromotor driver uses fixed 100-entry sine look-up tables and allows for changing the micromotor drive frequency through changing the pre-scaling of the A-line clock signal.

6.1.1 TIMER MODULES

To allow for programmatic scaling of the A-line clock signal, the micromotor driver uses a timer (TMR) module on the PIC24. TMR modules function as counters, with a hardware (or software) interrupt initiated once the count reaches a set value. The micromotor driver uses the TMR3 module for A-line clock scaling, with pin 33 on the PIC24 set as the TMR3 source input. A voltage divider is used to bring the 5V A-line signal down to 3.3V to avoid damaging the PIC24.

6.1.2 DIRECT MEMORY ACCESS MODULES

The micromotor driver uses direct memory access (DMA) modules on the PIC24 to send sine look-up data through the SPI ports. DMA modules allow for data within a specified memory location (or memory block) on the PIC24 to be transferred to another memory location without the need for MCU intervention. The DMA modules are first configured with a transfer source, a transfer destination, a transfer mode and a transfer trigger. The micromotor driver uses DMA modules in “repeated one-shot” mode, where the same memory block (the sine look-up table) is transferred, one element at a time, per trigger signal. Using individual DMA modules to handle each of the three phase outputs both frees the MCU to handle other tasks (leading to faster, more efficient firmware) and allows all three SPI channels to be updated in parallel (increasing the maximum DAC update frequency).

The DMA modules were configured using the MPLAB Code Configurator (MCC) within the MPLAB X IDE to use the memory location of the pre-computed sine look-up tables as the DMA transfer source, and the corresponding SPI transmit buffers as the DMA transfer destination. All DMA modules used the TMR3 hardware interrupt as the DMA transfer trigger.

The usage of DMA to handle SPI transfers, along with running the PIC24 at its maximum clock frequency of 32MHz, allowed for a maximum waveform-generation DAC update frequency of approximately 250kHz.

6.1.3 SPI MODULES

The SPI modules used for sine look-up data transfers were configured using “framed” SPI mode. A typical SPI master-mode transfer involves three wires: a clock line (labelled SCK), a data line (labelled SDO) and a slave-select line (labelled SS). In “normal” mode, multiple slave devices can be controlled through a single bus. A “normal” mode transfer is completed through the following process: the SS line corresponding to the recipient slave device is brought low, data is loaded into the SPI transmit buffer, and data is clocked to the slave device through the SCK and SDO lines. In “normal” mode, the SPI module is only “active” during the data transfer process and is initiated through an interrupt service routine (ISR).

In "framed" mode, the SPI module is always active, and the SPI module controls the SS line. As the SPI module controls the SS line, only one slave can be attached to the SPI bus, however greater transfer rates can be achieved. In "framed mode", the SCK line is always running and the SS line is brought low synchronously with the SS line once data is placed in the SPI transmit buffer. "Framed mode" allows for DMA modules to transfer data through SPI without the need for an ISR, resulting in greater data transfer rates [14].

6.1.4 OVERVIEW

Figure 14 shows an overview of the frequency synthesis hardware. Synchronous and asynchronous operations are achieved through changing the TMR3 clock source. Synchronous operation uses the OCT A-line clock signal, while asynchronous operation uses an internal 2MHz oscillator on the PIC24 to generate DMA trigger signals through TMR3. As the A-line clock signal and PIC24 internal oscillator are different frequencies, the TMR3 prescaler (PR3) range differs for synchronous and asynchronous operations, as can be seen from (4). With a 100-entry sine look-up table and a maximum output frequency of 500Hz, the value of PR3 is restricted to be an integer greater than or 1 for synchronous operation and an integer greater than 39 for asynchronous operation. The amplitude of the waveform-generation DAC output is controlled by interfacing with a MCP4725 through the I2C clock (SCL2) and data (SDA2) lines on the PIC24 I2C2 port. The driver can be switched between synchronous and asynchronous operation through a UART command from the user.

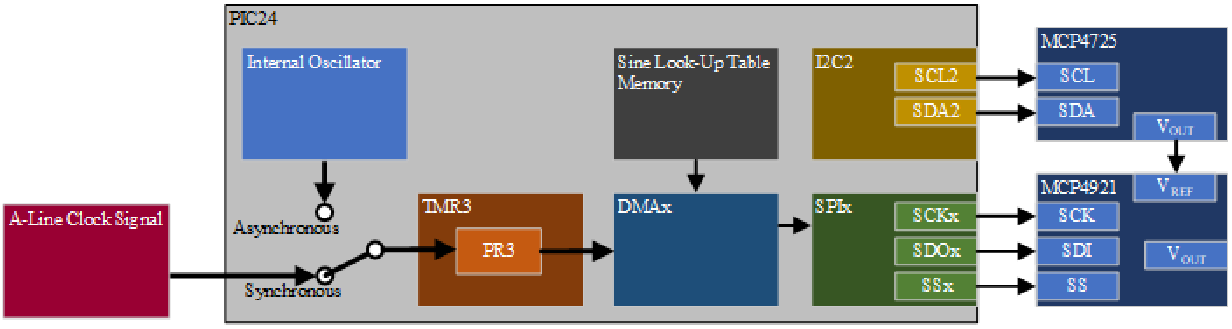


Figure 14. Overview of frequency synthesis hardware. One DMAx and SPIx channel is used for each waveform-generation DAC (x = 1, 2, 3).

6.2 FILTERING & GAIN

As the micromotor driver uses large (100-entry) sine look-up tables, a simple RC filter can be used to achieve sufficient smoothing for the three-phase motor drive sinusoids. The cutoff frequency of an RC low-pass filter is found as:

$$f_C = \frac{1}{2\pi RC} \tag{9}$$

The micromotor driver low-pass filter was designed with R =16kΩ and C =10nF to achieve a cutoff frequency f_C ≈1kHz.

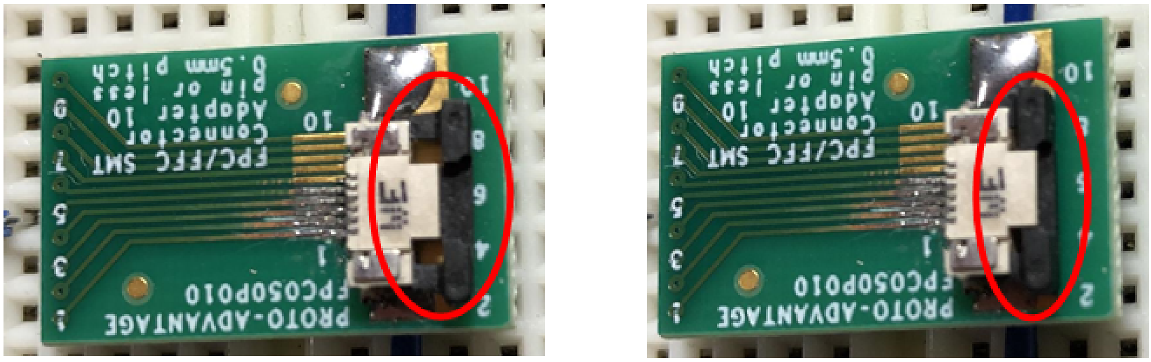
The selection of 1kHz for the low-pass filter cutoff frequency was based on the goal of maximizing attenuation of quantization noise, while minimizing unwanted attenuation of the output drive signal. As the micromotor driver output frequencies can range from 1Hz to 500Hz, any frequency component greater than 500Hz is undesirable and should be removed to produce a smooth drive waveform. 1kHz was selected as the cutoff frequency to provide the greatest attenuation of undesired frequencies (to create the largest stop-band) while minimizing unwanted attenuation or phase shift of pass-band frequencies. With the cutoff frequency set at 1kHz, the micromotor driver experiences approximately -1dB of attenuation at the maximum operation frequency of 500Hz, and approximately -0.05dB of attenuation (negligible) at a standard operation frequency of 100Hz. It should be noted that

A non-inverting amplifier stage was included to increase the allowable voltage output of the micromotor driver in addition to providing a buffer between the filtering and current amplification stages.

6.3 CURRENT AMPLIFICATION & OUTPUT

The micromotor driver uses a class AB amplifier in its current amplification stage. The class AB amplifier uses the voltage drop across its two diodes to ensure at least one of the two output transistors is always conducting [14].

To connect to the SBL04-072 micromotor used in OCIL probes, the micromotor driver uses a 6-position, 0.5mm pitch Flat Flexible Cable (FFC) connector as shown in Figure 15. The FFC connector was soldered to a generic 10-position surface-mount to dual in-line packaging (DIP) breakout board to allow for use with a breadboard. Table 2 outlines the connections between the FFC connector and micromotor driver phases.



(a)

(b)

Figure 15. (a) FFC connector in open position. (b) FFC connector in locked position. The connector must be locked once the micromotor cable is inserted in order to ensure proper connection.

Position	1	2	3	4	5	6
Usage	Unused	Phase W	GND	Phase V	Phase U	Unused

Table 2. Connections between FFC connector and motor phases.

6.4 INTERFACING

The micromotor driver was designed for host control through the Recommended Standard 232 (RS-232) serial protocol as RS-232 communications are used for control of similar peripherals in the OCIL's RPD. To allow for use of the micromotor driver while the full OCIL micromotor imaging system is under development, the micromotor driver was designed to allow for control through the Explorer 16/32 board. A RS-232 Click board (MikroElektronika d.o.o.) was used to interface with the RS-232 bus through the UART1 port on the PIC24. Table 3 presents the pin connections required for serial communication with the micromotor driver's PIC24.

Usage	Pin Label	MCC Label	Pin Number
RS-232 Control	UART1 TX	U1TX	50
	UART1 RX	U1RX	49
	UART1 RTS	U1RTS	23
	UART1 CTS	U1CTS	18

Table 3. Pin connections required for serial control of the micromotor driver through PIC24 UART1.

Interfacing with the micromotor driver is accomplished through sending either one or two control bytes. Table 4 shows the list of one-byte commands and Table 5 shows the list of two-byte commands. Two-byte commands are used for setting driver parameters, with the first byte being used to specify the parameter to change, and the second byte holding the data to update the parameter. On receipt of a valid command, the micromotor transmits an "acknowledgement" byte (equal to 0x79 or the character 'O') to the host over the RS-232 bus.

Command Name	Command Byte	Description
START	0x20	Starts the micromotor
STOP	0x21	Stops the micromotor
R_STATE	0x10	Reads the state of the micromotor: 0x01: Idle 0x02: Accelerating 0x03: Running 0x04: Deccelerating 0x05: Standby
R_ALINES_PER_BSCAN	0x11	Reads the set value of the number of A-lines per BScan.
R_ASYNC_FREQ	0x12	Reads the set value of the asynchronous frequency.
R_SYNC	0x13	Reads whether the driver is in synchronous or asynchronous operation.
R_ACCEL_RATE	0x14	Reads the acceleration rate of the driver.
R_VOLTAGE	0x15	Reads the value of the MCP4725 reference-voltage DAC.
R_DIRECTION	0x16	Reads the rotation direction of the driver.

Table 4. Single-byte serial commands used for control of the micromotor driver.

Command Name	Command Byte	Data Byte	Description
W_ALINES_PER_BSCAN	0x01	Integer multiple of 100. Must be greater than 200.	Writes the value of the number of A-lines per B-Scan. The data byte corresponds to the number of A-lines per B-Scan.
W_ASYNC_FREQ	0x02	Integer from 1 to 500	Writes the value of the asynchronous frequency. The data byte corresponds to the output frequency in Hz.
W_SYNC	0x03	0x00 or 0x01	Writes whether the driver is in synchronous(0x01) or asynchronous (0x01) operation.
W_ACCEL_RATE	0x04	Integer from 0 to 250	Writes the acceleration rate of the driver. The data byte corresponds to the acceleration rate in Hz/s.
W_VOLTAGE	0x05	Integer from 0 to 4095	Writes the value of the MCP4725 reference-voltage DAC.
W_DIRECTION	0x06	0x00 or 0x01	Writes the rotation direction of the driver. As viewed from the motor shaft end: 0x00: clockwise rotation, 0x01: counter-clockwise rotation.

Table 4. Two-byte serial commands used for control of the micromotor driver.

The function of buttons used for control through the Explorer 16/32 board are shown in Figure 16.

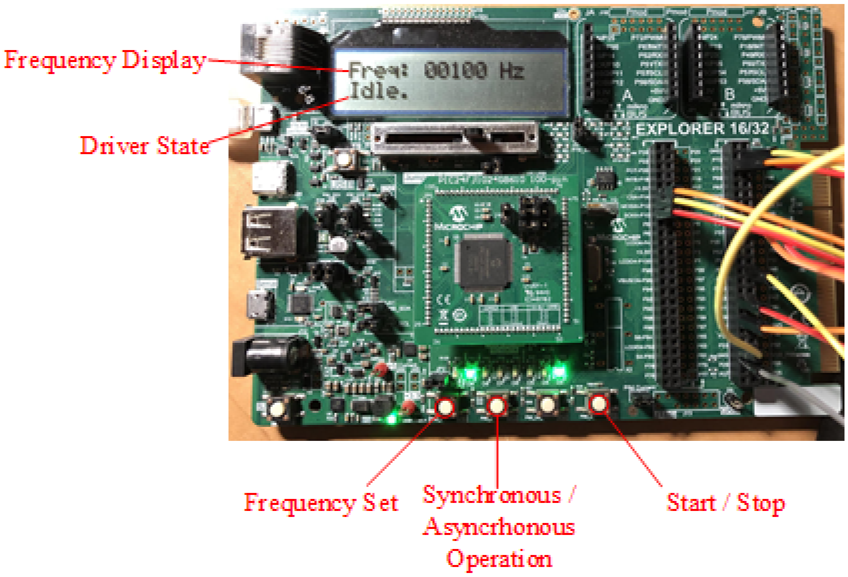


Figure 16. User interfacing with the micromotor driver through the Explorer 16/32 board.

6.5 FIRMWARE

6.5.1 FIRMWARE OVERVIEW

The micromotor driver was implemented in the form of a hierarchical state machine (HSM) using the Quantum Leaps "Quantum Platform in C" (QP/C) framework [15]. QP/C provides a lightweight, modular state-machine development framework for use in embedded systems. The state machine diagram of the micromotor driver is shown in Figure 17. The micromotor driver firmware is based around the usage of QP/C events: all updates of driver properties or state transitions are initiated by posting event commands and data to the event queue instantiated by the QP/C framework.

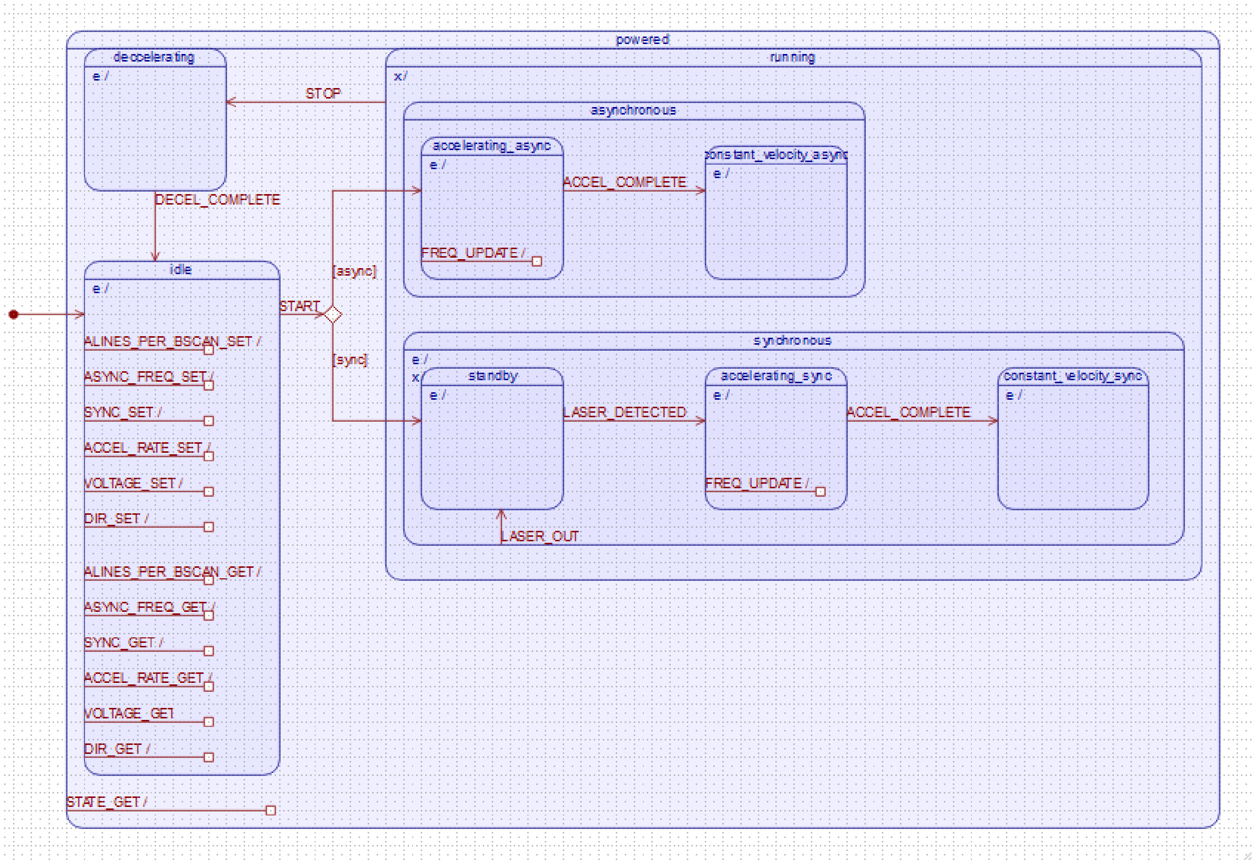


Figure 17. State machine diagram for the micromotor driver as viewed in the QPC framework.

The micromotor driver consists of three high-level states: *idle*, *running* and *decelerating*. The idle state allows for the setting and reading of driver parameters. The running state contains two sub-states: *synchronous* and *asynchronous*, corresponding to synchronous or asynchronous operation of the driver. Within the *synchronous* and *asynchronous* states are states corresponding to accelerating or running the micromotor at a constant velocity. The *synchronous* state also contains a *standby* state that is used to stop the micromotor from running if an A-line trigger signal is not detected.

6.5.2 FIRMWARE PROCESSES

6.5.2.1 SYNCHRONOUS & ASYNCHRONOUS OPERATION

Synchronous and asynchronous operations are achieved by changing the clock source of the TMR3 counter module used to initiate waveform-generation DAC updates through the DMA modules. Synchronous operation configures the TMR3 module to use the A-line clock signal present on PIC24 pin 33 as the TMR3 clock source, while asynchronous operation configures the TMR3 module to use an internal oscillator within the PIC24 as the TMR3 clock source.

6.5.2.2 ACCELERATION

Acceleration of the micromotor is achieved by changing the pre-scale value, *PR3*, of the TMR3 clock. With an input clock frequency of f_{ALINE} and sine look-up tables at a constant size of *SINE_TABLE_SIZE*, the output frequency, f_{MOTOR}, of the micromotor drive waveforms is found as:

$$f_{MOTOR} = \frac{f_{ALINE}}{(PR3+1)\cdot SINE\_TABLE\_SIZE} \tag{10}$$

When in synchronous mode, the number of A-lines per B-scan is given by the product $(PR3 + 1)$ ·*SINE_TABLE_SIZE.*

To accelerate the micromotor, the micromotor driver updates the value of *PR3* along a set of fixed values calculated to result in a linear acceleration profile. Once the value of *PR3* becomes less than or equal to the final pre-scale value required to achieve the output frequency set by the user, the micromotor driver assumes the final *PR3 value* and switches from an *accelerating* state to a *constant_velocity* state.

PR3 values are updated through an ISR for the PIC24 TMR2 module which is clocked using a 2MHz oscillator on the PIC24. The 2MHz oscillator is pre-scaled by a factor, PR2, and used to generate an ISR each time the TMR2 module counts to the value in *PR2.* The value of *PR2* is changed to result in different ISR callback rates (typically on the order of 10ms to 20ms) in order to achieve different acceleration rates of the micromotor. The value of *PR2* can be calculated to achieve an acceleration rate *A* with the following relation:

$$PR2 = \frac{2000000}{A} - 1 \tag{11}$$

Table 5 shows an example of the *PR3* and *PR2* values required to accelerate the micromotor from 1Hz to 100Hz at 50Hz/s in both synchronous (with a 100kHz A-line frequency) and asynchronous operation modes. In practice, acceleration is performed in steps of 1Hz.

	f_{MOTOR}	1Hz	10Hz	20Hz	30Hz	40Hz	50Hz	60Hz	70Hz	80Hz	90Hz	100Hz
Sync	*PR2*	39999	39999	39999	39999	39999	39999	39999	39999	39999	39999	39999
	PR3	999	99	49	32*	24	19	16*	13*	11*	10*	9
Async	*PR2*	39999	39999	39999	39999	39999	39999	39999	39999	39999	39999	39999
	PR3	19999	1999	999	666*	499	399	322*	285*	249	221*	199

Table 5. Example PR3 and PR2 values required to accelerate the micromotor from 1Hz to 100Hz at 50Hz/s in synchronous and asynchronous operation modes. Entries marked with an asterisk denote output frequencies that are not perfect divisors of the input frequency.

6.5.2.3 LASER-OUT DETECTION

To prevent the driver phases from being held at imbalanced DC voltages if the A-line clock signal stops while the driver is in synchronous operation, the micromotor driver implements "laser-out" detection. Laser-out detection was implemented through the use of the TMR4 and TMR5 modules on the PIC24. The TMR4 module is implemented as a simple up-counter using the A-line clock (PIC24 pin 33) as the counter source. The TMR5 module is configured to call an ISR every 10ms to check the TMR4 count value. If the TMR4 count value has not changed between TMR5 ISR calls, a *LASER_OUT* signal is generated and the driver enters a *standby* mode. The driver restarts accelerating and running the micromotor once an A-line clock signal is detected.

A hardware implementation of laser-out detection (as shown in Figure 18) was considered but not used in the driver design as the software-based implementation does not require additional circuitry and allows for easier design changes.

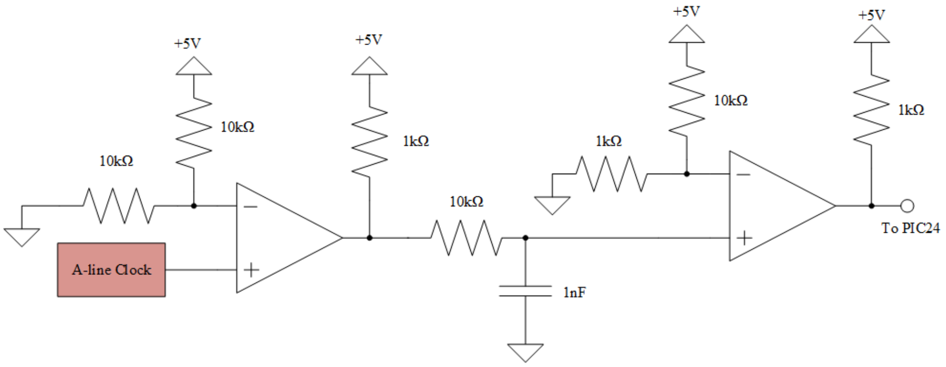


Figure 18. Hardware implementation of laser-out detection consisting of two comparators and a low-pass filter. When a clock signal is present, the PIC24 output will be +5V. When no clock is present the output will be 0V.

6.5.3 FIRMWARE FILE STRUCTURE

Figure 19 shows an overview of the file structure within the micromotor driver MPLAB X project and Table 6 provides a description of what is implemented in each file.

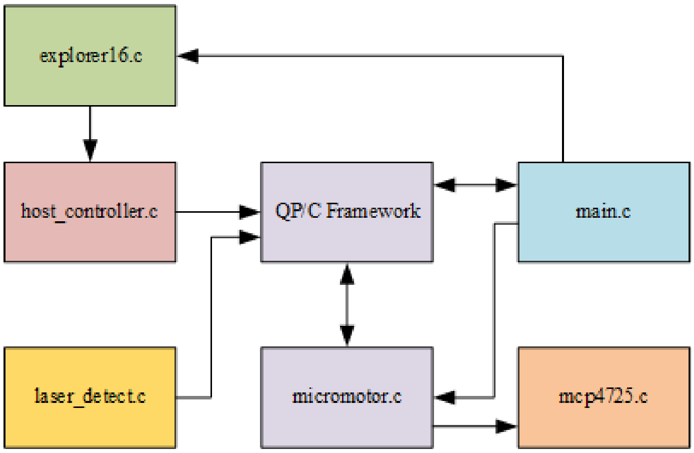


Figure 19. Implementation files used in micromotor driver MPLAB X project.

File	Description
main.c	Initializes the PIC24FJ modules, QP/C framework, micromotor class object within the QP/C framework, and the interfaces used on the Explorer 16/32 board (buttons, LEDs, LCD screen, slider).
micromotor.c	Initializes driver sine look-up tables, holds all driver status and configuration variables, and contains all QP/C state machine code used by the micromotor driver. The code implementing the state machine is generated through the Quantum Leaps QP Modeler (QM) [15].
host_controller.c	Implements handling of commands outlined in Table 4 and Table 5.
laser_detect.c	Implements laser-out detection as described in Section 6.5.2.3
mcp4725.c	Implements a method to interface with the MCP4725 DAC used to set the micromotor driver output voltage amplitude.
explorer16.c	Implements user interfacing through the buttons and the slider on the Explorer 16/32 board.

Table 6. Overview of files in micromotor MPLAB X project.

7 DRIVER VALIDATION

The micromotor driver was validated through visual inspection during real-time imaging, as well as through analysis of the output drive waveforms and collected imaging data. The goal of the validation was to test the micromotor driver's ability to allow for stability between frames (*inter*-frame stability) and the uniformity of rotation within a given frame (*intra*-frame stability).

7.1 OUTPUT WAVEFORMS

Figure 20 and Figure 21 show oscilloscope measurements of one phase of the micromotor driver output during synchronous operation set to 1000 A-lines / B-scan with a 50kHz A-line clock and a 1.4V amplitude drive signal. A perfect sinusoid (black, dotted line) has been best-fit to the oscilloscope data in Figure 20 for comparison.

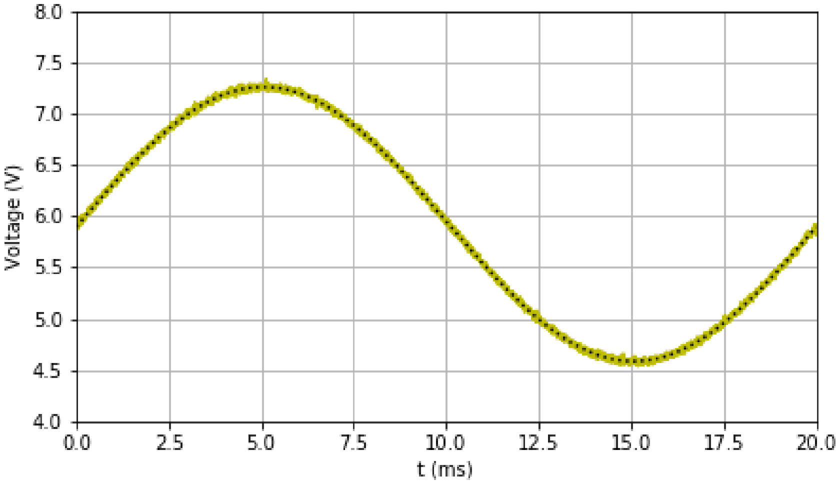


Figure 20. Oscilloscope data of one period of the micromotor waveforms (yellow) versus real sinusoid (black, dotted).

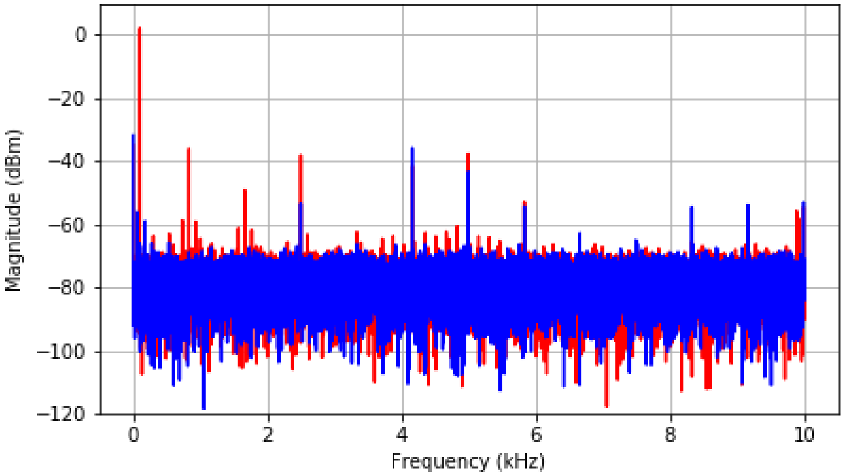


Figure 21. Oscilloscope data of the frequency spectrum of micromotor driver. Blue data shows the measured spectrum while the motor is <u>not</u> running. Red data shows the frequency spectrum of the micromotor drive waveform when running at 1000 A-Lines / B-scan.

7.2 ACCELERATION PROFILE

Figure 22 shows oscilloscope data of the micromotor driver output during initial acceleration at a set acceleration rate of 50 Hz/s. Figure 23 plots the frequency of the waveform as calculated through sequential peaks in the oscilloscope data for the full acceleration duration up to a final running frequency set to 100Hz. Acceleration profile data was collected during asynchronous operation of the micromotor.

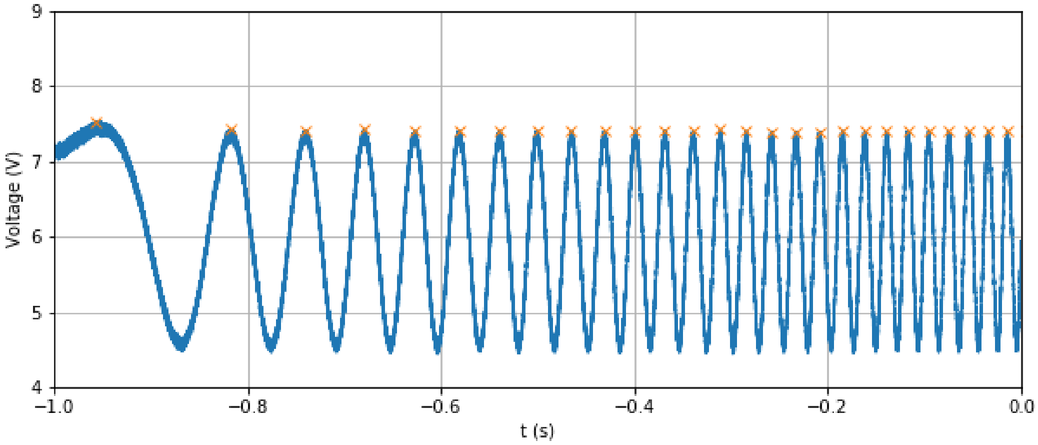


Figure 22. Oscilloscope data showing micromotor driver output during acceleration. Orange "X" marks denote the calculated peaks of the sine wave. Peaks were calculated using the find_peaks method from the scipy.signal python library.

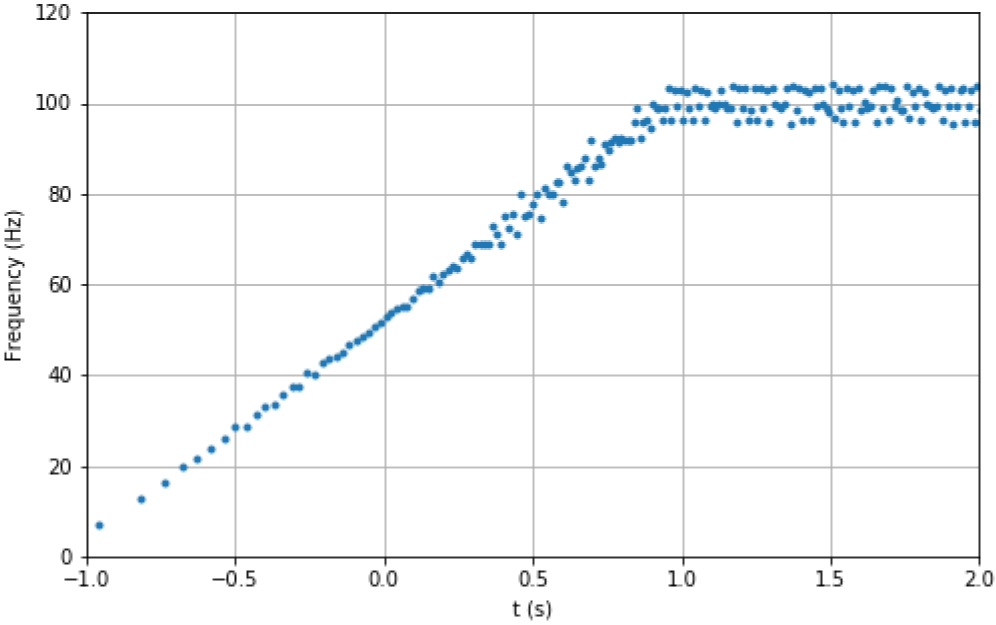
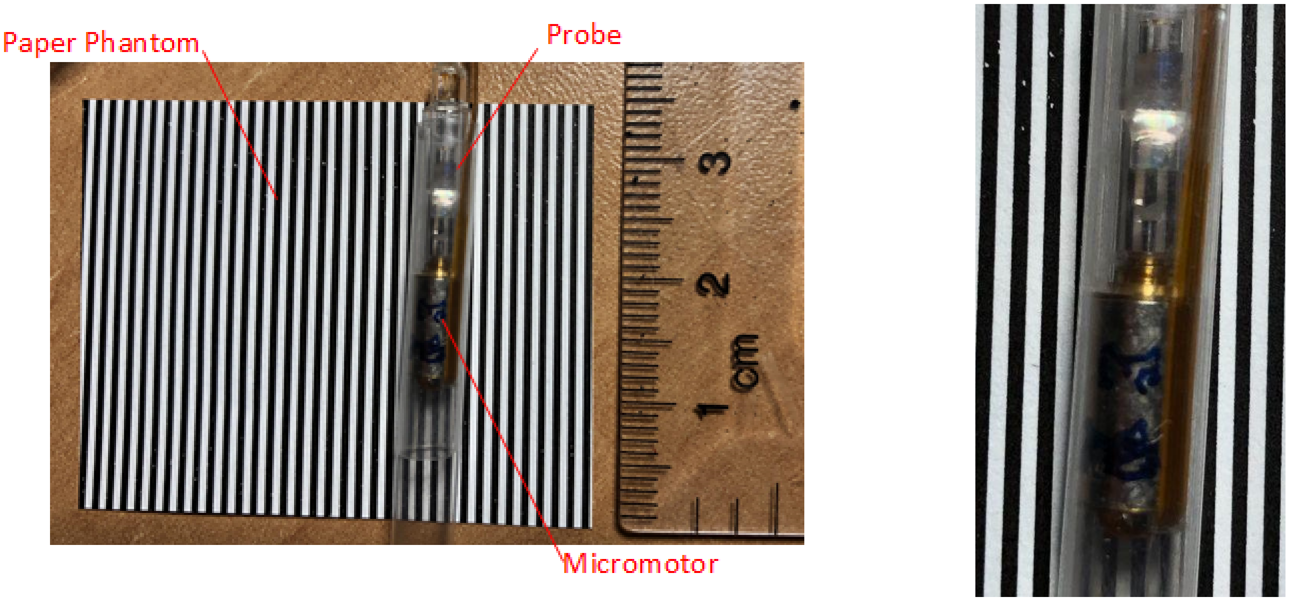


Figure 23. Frequency profile of micromotor during acceleration as calculated from consecutive peaks in oscilloscope data.

7.3 IMAGING

Validation of imaging performance was completed using a printed paper phantom consisting of alternating, regularly-spaced black and white lines. The purpose of the paper phantom was to provide a reference to gauge intra-frame and inter-frame stability. Figure 24 shows the micromotor probe and printed paper phantom used for image validation. Figure 25 presents a view of how the printed phantom appears in the OCT image viewed from the probe. All imaging was performed using the OCIL's DBMC5 micromotor probe.



(a) (b)

Figure 24. (a) Micromotor probe on printed paper phantom. (b) Enlarged view of micromotor probe.

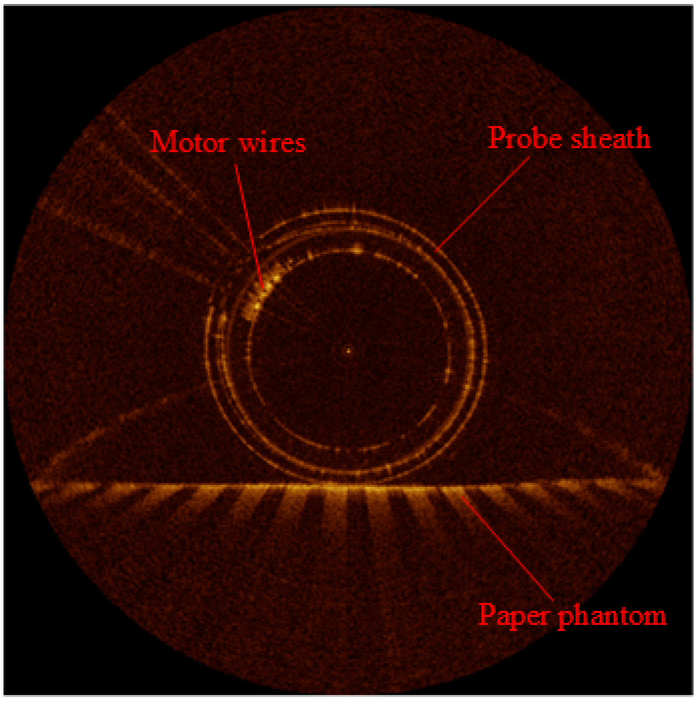
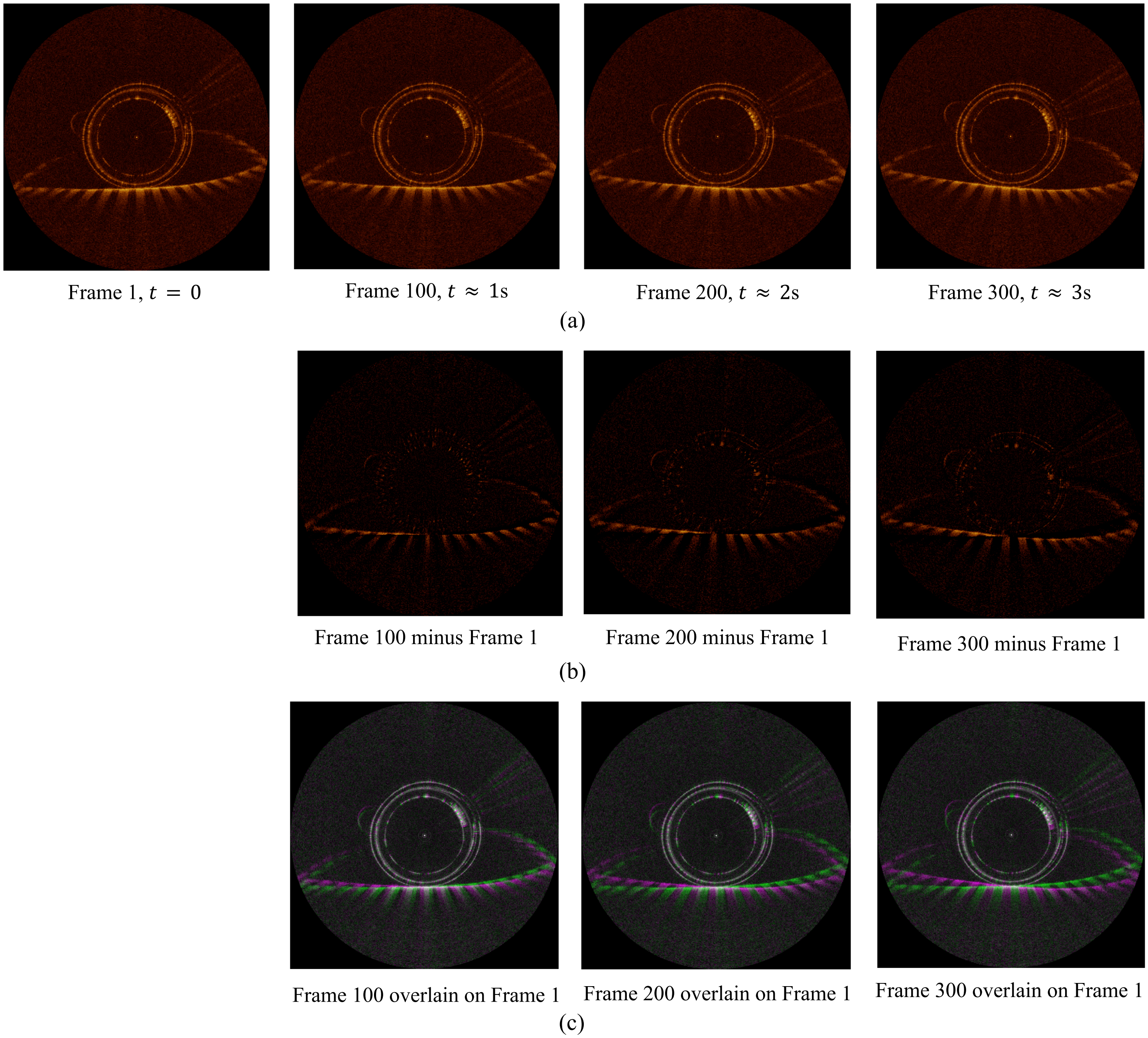


Figure 25. OCT image from micromotor probe showing printed paper phantom.

To validate the micromotor driver's inter-frame stability, OCT imaging data was collected using both the SOD12ST and new micromotor driver to run the micromotor probe. Data was collected for ten seconds at a scan rate of 1000 A-lines / B-scan using a 100kHz A-line clock. The SOD12ST was set to drive the micromotor at 6000RPM with a 3Vpp drive waveform. The new micromotor driver was set to 1000 A-lines / B-scan with a 3Vpp drive waveform.

7.3.1 IMAGING WITH SOD12ST DRIVER

Figure 26 shows images acquired 100 frames apart using SOD12ST driver and compares the differences between the first frame and consecutive frames.



Frame 1, $t = 0$ Frame 100, $t \approx 1$s Frame 200, $t \approx 2$s Frame 300, $t \approx 3$s

(a)

Frame 100 minus Frame 1 Frame 200 minus Frame 1 Frame 300 minus Frame 1

(b)

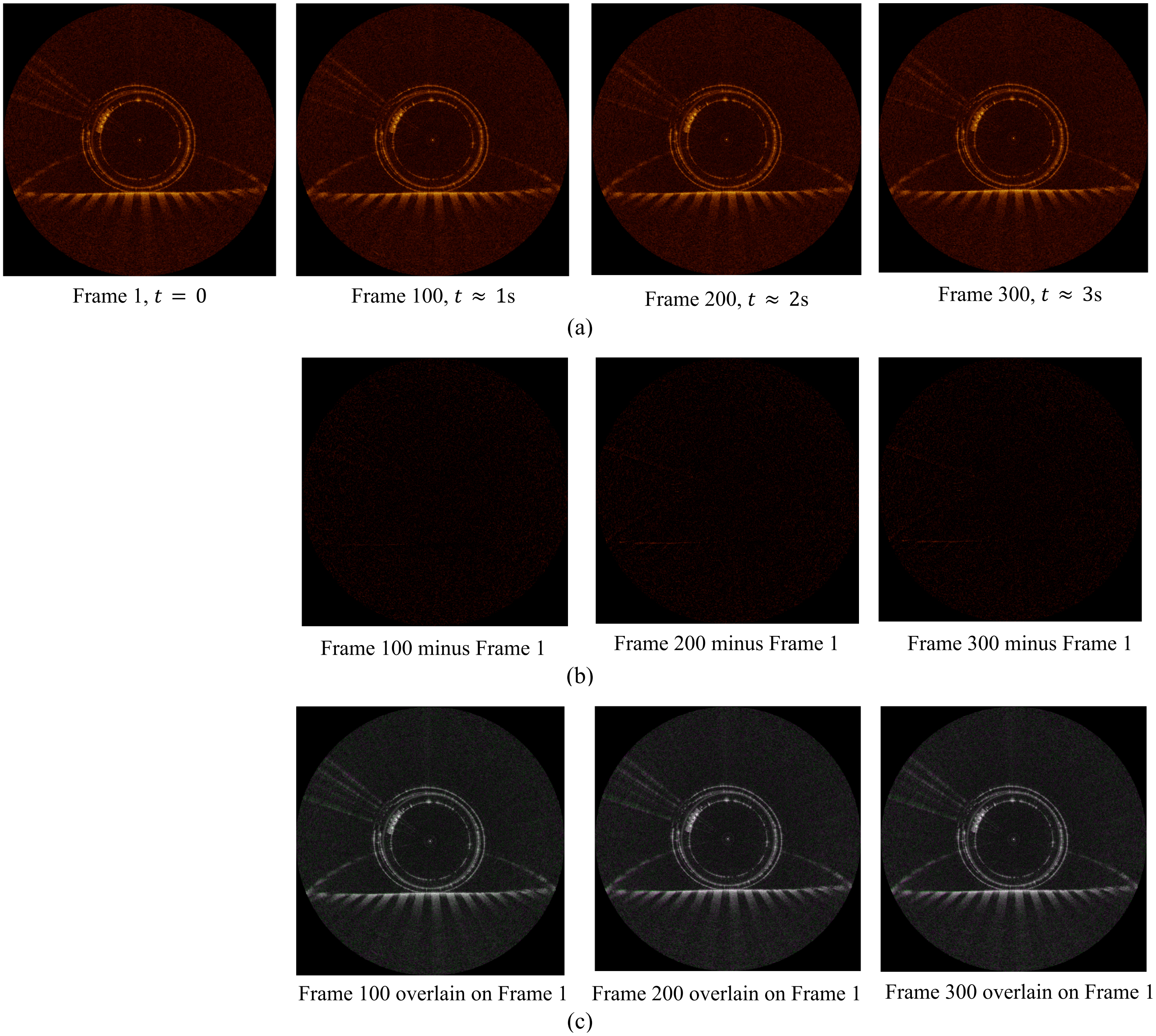
Frame 100 overlain on Frame 1 Frame 200 overlain on Frame 1 Frame 300 overlain on Frame 1

(c)

Figure 26. Comparison between imaging frames acquired with SOD12ST. (a) Frame outputs. (b) Absolute difference between frame n and frame 1. (c) Overlain image comparing frame n and frame 1. Magenta and green regions show where pixel intensities are different, grey regions show where pixel intensities are the same.

7.3.2 IMAGING WITH NEW MICROMOTOR DRIVER

Figure 27 shows images acquired 100 frames apart using the new micromotor driver and compares the differences between the first frame and consecutive frames.



Frame 1, $t = 0$

Frame 100, $t \approx 1$s

Frame 200, $t \approx 2$s

Frame 300, $t \approx 3$s

(a)

Frame 100 minus Frame 1

Frame 200 minus Frame 1

Frame 300 minus Frame 1

(b)

Frame 100 overlain on Frame 1

Frame 200 overlain on Frame 1

Frame 300 overlain on Frame 1

(c)

Figure 27. Comparison between imaging frames acquired with new micromotor driver. (a) Frame outputs. (b) Absolute difference between frame 1 and frame n. (c) False-colour image comparing frame 1 and frame n. Magenta and green regions show where pixel intensities are different, grey regions show where pixel intensities are the same.

Figure 28 shows an image taken with the printed paper phantom wrapped around the motor sheath to observe intra-frame stability with the new micromotor driver.

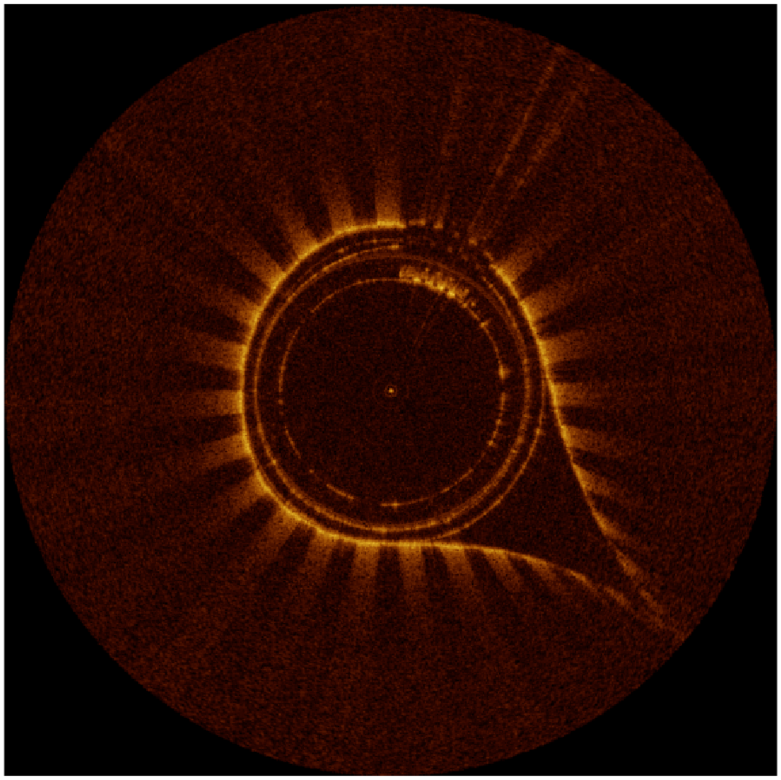


Figure 28. Image taken with paper phantom wrapped around motor sheath while the micromotor is controlled with the new micromotor driver.

8. DISCUSSION

8.1 OUTPUT WAVEFORMS

The micromotor driver's output waveforms showed agreement with a pure sinusoid as seen in Figure 20. The recorded output waveforms exhibited a small amount of high-frequency noise most easily seen in the sine "peak" and "trough" in Figure 20 and as shown in the spectral analysis in Figure 21. The source of the high-frequency noise could be due to noise from the DAC outputs, noise from the 12V power supply line, noise due to the breadboard implementation of the driver, or simply noise present in the oscilloscope probes used to collect the measurements. Noise on the micromotor phases has the potential to cause jitter in the rotation of the micromotor and can result in "shaky" imaging.

8.2 IMAGING

The new micromotor driver did not result in azimuthal drifting between frames while imaging, and good inter-frame stability was observed as demonstrated in Figure 27. No azimuthal drifting was observed over extended operation of the micromotor (>10 minutes). Small "vibrations" in the image were occasionally observed while using the new micromotor driver. The vibrations could be caused by the noise on the micromotor driver's output waveforms as mentioned in Section 8.1, or due to mechanical vibrations within the micromotor itself (which could be caused by the center-of-mass of the mirror mounted on the micromotor shaft being slightly off-axis).

The new micromotor driver also exhibited good intra-frame stability, with no evidence of non-uniform rotational distortions (NURD) being observed during imaging.

8.3 FUTURE WORK

The main work left to be completed is the design and fabrication of a PCB for the micromotor driver hardware and the investigation into noise reduction on the output waveforms. The PCB layout should take the physical structure of the final OCT imaging system into account when designing the location of the micromotor and UART connectors. Additionally, a majority of the available memory (90% of data memory and 96% of program memory) on the PIC24FJ1024GB610 microcontroller was unused by the micromotor driver program, and an alternative device from the PIC24FJ-X-GB610 family could be implemented in the final PCB design (such as the PIC24FJ128GB610 or PIC24FJ256GB610). When switching to a new PIC24FJ model it will be important to verify that the supported functions of the pins match the PIC24FJ1024GB610.

The method in which the micromotor driver program performs frequency synthesis could also be changed to improve the number of supported A-lines per B-scan. Instead of a fixed sine look-up table, a variable-size sine look-up table could be implemented (similar to the prototype micromotor driver design). A variable-size sine look-up table design would leave the final A-line clock prescaling value (TMR3 PR3) constant and instead would generate a new sine look-up table each time a new micromotor frequency was specified by the user. Such a design would allow for the A-line per B-scan value to be any integer multiple of 2 (as opposed to the current support for any multiple of 100).

9. CONCLUSION

This document describes the theory, development and validation of a micromotor driver designed to achieve stable OCT imaging with micromotor catheter probes. The problem of mismatching between OCT-imaging and micromotor-rotation frequencies present within the OCIL's current micromotor imaging system are discussed and a solution involving real-time updating of DACs to generate micromotor drive waveforms is presented.

The rationale behind the micromotor driver hardware is discussed, and the final hardware and firmware designs are presented along with information about how to control the micromotor driver.

The micromotor driver is validated through the use of a printed paper phantom to test intra-frame and inter-frame stability. We find that the new micromotor driver solves the frequency-mismatch issues present with the OCIL's current micromotor imaging system and allows for stable imaging with micromotor OCT probes.

REFERENCES

[1] "*Ultrasound,*" Jul. 2016. Accessed on: Apr. 4, 2021. [Online]. Available: https://www.nibib.nih.gov/science-education/science-topics/ultrasound.

[2] D. Huang et al., "Optical coherence tomography," Science, vol. 254, issue 5035, pp. 1178-1181, Nov. 1991.

[3] S. Aumann et al., "Optical Coherence Tomography (OCT): Principle and Technical Realization", in High Resolution Imaging in Microscopy and Ophthalmology, Heidelberg, Germany: Springer Publishing, 2019. Accessed on: Apr. 4, 2021. [Online]. Available: https://doi.org/10.1007/978-3-030-16638-0_3

[4] A. H. Kashani et al., "Optical coherence tomography angiography: A comprehensive review of current methods and clinical applications," Progress in Retinal and Eye Research, vol. 60, pp. 66-100, Sept. 2017.

[5] T. Tsai et al., "Ultrahigh speed endoscopic optical coherence tomography for gastroenterology," Biomedical optics express, vol. 5, pp. 4387-4404, Dec. 1, 2014.

[6] R. Spaide, J. Fujimoto and N. Waheed, "IMAGE ARTIFACTS IN OPTICAL COHERENCE TOMOGRAPHY ANGIOGRAPHY," Retina, vol. 35, pp. 2163-2180, 2105.

[7] J. Zhao and Y. Yu, "Brushless DC Motor Fundamentals," Monolithic Power Systems, WA, USA, Jul. 2011. [Online]. Available: https://www.monolithicpower.com/pub/media/document/Brushless_DC_Motor_Fundamentals.pdf

[8] *"What are Brushless DC Motors,"* 2021. Accessed on: Apr. 14, 2021. [Online]. Available: https://www.renesas.com/us/en/support/engineer-school/brushless-dc-motor-01-overview

[9] *"Controlling BLDC Motors,"* 2021. Accessed on: Apr. 14, 2021. [Online]. Available: https://www.renesas.com/us/en/support/engineer-school/brushless-dc-motor-01-overview

[10] E. Murphy and C. Slattery, "All About Direct Digital Synthesis," Analog Dialogue, vol. 38, Aug. 2004. Accessed on: Apr. 7, 2021. [Online]. Available: https://www.analog.com/en/analog-dialogue/articles/ all-about-direct-digital-synthesis.html

[11] H. Zumbahlen, "Phase Response in Active Filters Part 2, the Low-Pass and High-Pass Response," Analog Dialogue, vol. 43, Sept. 2009. Accessed on: Apr. 5, 2021. [Online]. Available: https://www.analog.com/media/en/analog-dialogue/volume-43/number-3/articles/phase-response-in-active-filters-2.pdf

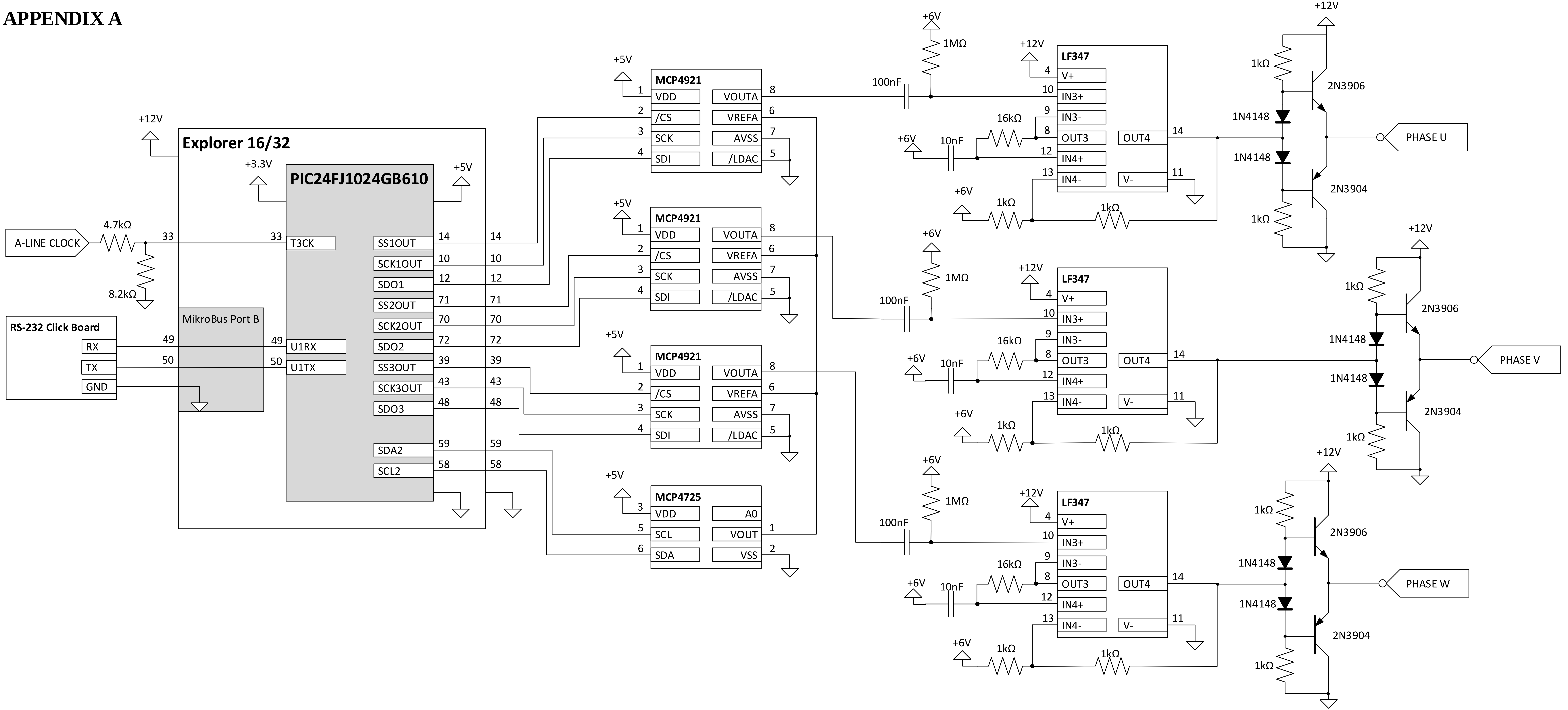
[12] Analog Devices, *Increase Amplifier Output Drive Using a Push-Pull Amplifier Stage,* Analog Devices, n.d. Accessed on: Apr. 4, 2021. [Online]. Available: https://www.analog.com/en/technical-articles/increase-amplifier-output-drive-using-a-push-pull-amplifier-stage.html

[13] "*Class AB Amplifier,"* n.d. [Online]. Accessed on: Apr. 4, 2021. Available: https://www.electronics-tutorials.ws/amplifier/class-ab-amplifier.html

[14] B. Land, Class Lecture, Topic: "Serial Peripheral Interface (SPI)." ECE4760, School of Electrical and Computer Engineering, Cornell University, New York, Jul., 16, 2020.

[15] Quantum Leaps, *About CP/C,* Quantum Leaps, 2021. Accessed on: Apr. 5, 2021. Available: https://www.state-machine.com/qpc/

APPENDIX A


Explorer 16/32
PIC24FJ1024GB610
+12V
+3.3V
+5V
A-LINE CLOCK
4.7kΩ
8.2kΩ
33
T3CK
RS-232 Click Board
RX
TX
GND
MikroBus Port B
49
50
U1RX
U1TX
SS1OUT
SCK1OUT
SDO1
SS2OUT
SCK2OUT
SDO2
SS3OUT
SCK3OUT
SDO3
SDA2
SCL2
14
10
12
71
70
72
39
43
48
59
58
MCP4921
VDD
/CS
SCK
SDI
VOUTA
VREFA
AVSS
/LDAC
1
2
3
4
8
6
7
5
MCP4725
VDD
SCL
SDA
A0
VOUT
VSS
+6V
1MΩ
100nF
10nF
16kΩ
1kΩ
LF347
V+
IN3+
IN3-
OUT3
IN4+
IN4-
OUT4
V-
9
13
11
2N3906
2N3904
1N4148
PHASE U
PHASE V
PHASE W