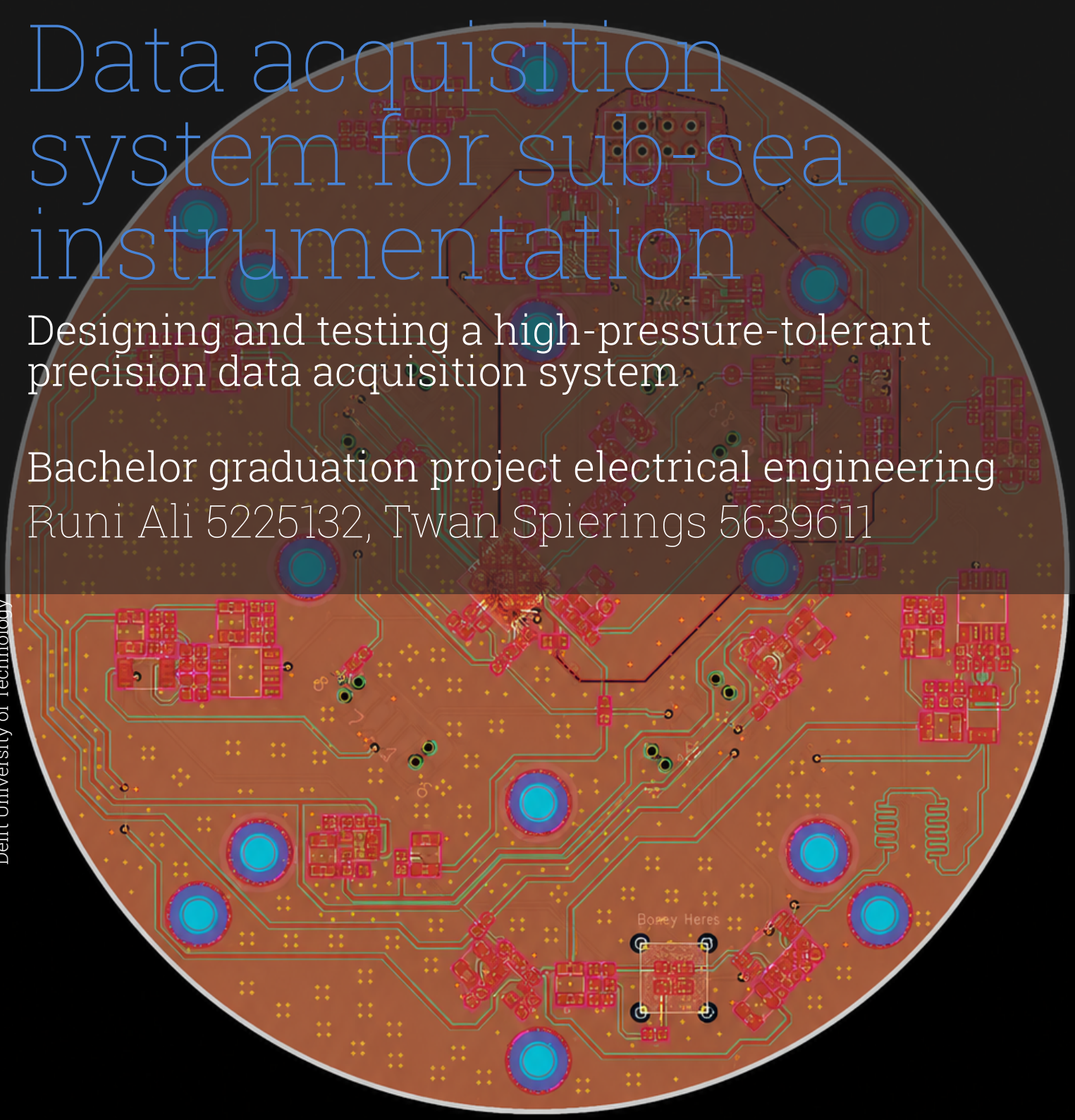
Data acquisition system for sub-sea instrumentation

Designing and testing a high-pressure-tolerant precision data acquisition system

Bachelor graduation project electrical engineering

Runi Ali 5225132, Twan Spierings 5639611



Delft University of Technology

TU Delft

Abstract

This report describes the design, building and testing of a high precision data acquisition system that is able to function in high pressure environments. The system uses an FPGA as its controller sending and receiving information to and from a data conversion device with the AD4630 as its ADC and the DAC8811 as its DAC. The system is tested in a pressure chamber which was pressurised up to 500 bars of pressure and the performance difference was found to be negligible. The system had to be proven to be scalable to 16 channels and these 16 channels had to fit inside a volume of $100cm^3$.

Preface

This report presents the design, implementation, and evaluation of a data conversion system developed for operation in a pressure chamber. The project focused on building a prototype that could be tested under demanding environmental conditions while still meeting the required electrical specifications.

Runi Ali 5225132, Twan Spierings 5639611
Delft, July 2026

Contents

1

Introduction

1.1. Background

Precision data acquisition systems are a critical component in a wide range of industrial and scientific applications. In many of these applications, performance alone is insufficient — the system must also operate reliably under demanding physical conditions. Harsh environments such as high-temperature industrial processes, aerospace deployments, and subsea installations impose constraints that go far beyond what standard laboratory instrumentation is designed to handle. Meeting the performance specification is only half the challenge; doing so under the operating environment is the other.

Subsea deployment presents a particularly demanding set of constraints. Hydrostatic pressure increases with depth, subjecting all electronics to mechanical stress that can shift component parameters and degrade signal integrity. Existing high-precision DAQ[1] systems capable of meeting stringent distortion and dynamic range requirements are predominantly designed for laboratory use and have not been characterized under hydrostatic pressure. Where pressure protection is applied, it typically requires bulky sealed enclosures that are incompatible with compact, weight-constrained platforms. There is no compact DAQ system that combines this level of signal quality with verified performance under subsea pressure conditions.

1.2. State-of-the-art analysis

Several commercial DAQ systems address parts of the problem. Spectrum Instrumentation offers the hybridNETBOX, a combined DAC and ADC solution with Ethernet connectivity, as well as PCIe variants for high-throughput applications[1]. Scopefun provides an open-source platform with a USB 3.0 interface, though it is limited to acquisition only with no signal generation capability[2]. SIRIUS offers a 24-bit ADC solution targeting precision measurement[3], and National Instruments provides a multi-channel solution combining a 20-bit ADC at 250 kSps with a 16-bit DAC[4]. Hybrid systems that combine both generation and acquisition in a single platform are relatively uncommon; the market is largely served by separate oscilloscope and arbitrary waveform generator products. Despite their differences, these solutions share a common architectural pattern. An FPGA acts as the central controller, managing the converter interfaces and handling communication to the host PC over Ethernet, PCIe, or USB. This architecture is well-established and forms the basis of the design presented in this report. None of the listed solutions, however, satisfies the full set of requirements. The signal quality and sample rate targets of this project are at the boundary of what currently available converter ICs can achieve according to email contact with Analog devices. More importantly, none of these systems has been characterised under hydrostatic pressure, and their physical form factors are too large for compact subsea platforms. A system that combines the required performance, compact form factor, and verified behaviour under pressure does not yet exist.

[1] Data Acquisition

1.3. Project Objective

The objective of this project is to design a compact, high-precision data acquisition system capable of both signal generation and acquisition, and to characterise its performance under hydrostatic pressure.

The system must meet the signal quality requirements identified as missing from existing solutions, achieving sample rates of at least 2 MSps, distortion levels below -100 dB, and a dynamic range exceeding 160 dB/Hz, all within a single unified system. The full set of system requirements is defined in Chapter 2.

To evaluate its behaviour under realistic operating conditions, the system was tested inside a pressure chamber, providing a controlled environment that replicates the hydrostatic conditions of subsea deployment. The result is a fully characterised DAQ system whose performance under pressure is documented and verified against its specifications.

1.4. System Overview

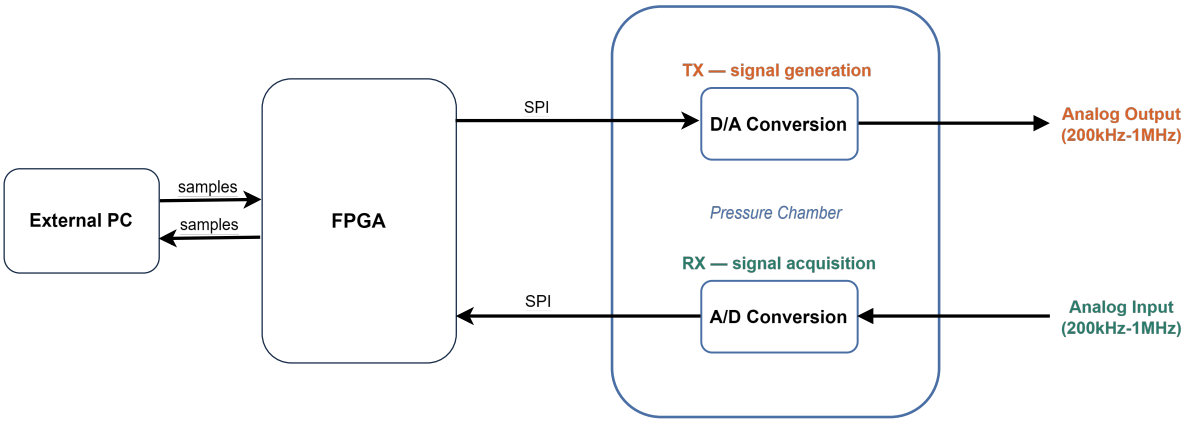


Figure 1.1: High-level DAQ system overview.

The system consists of two physically separated parts, as shown in figure 1.1. The first is a PCB containing the converter ICs and their surrounding circuitry, which is placed inside the pressure chamber and subjected to hydrostatic pressure during testing. The second is the communications and control side, comprising an FPGA and an external PC, which remain outside the chamber. The FPGA drives the converters and manages the data flow between the PC and the converter ICs.

The system operates in two directions. In the transmit path, digital samples are generated by the PC, sent to the FPGA, and converted to an analog signal by the DAC. In the receive path, an analog signal is sampled by the ADC, and the resulting digital data is sent back through the FPGA to the PC.

1.5. Project Structure

The task of this project is divided across three subgroups:

- **The control group** is responsible for the communication link between the external PC and the FPGA, including Ethernet protocol handling and data buffering
- **The circuit design group** handles the analog front-end, including the converter circuitry, voltage references, and PCB layout
- **The interface group** designs the digital interface between the FPGA and the converter ICs, covering the SPI masters, clock architecture, and synchronisation

The work of each subgroup is documented in a separate thesis, which together provide a complete account of the full system design.

2

Program of requirements

In this chapter the program of requirements will be discussed. First the functional requirements will be given. This is what the product must do if the product is finished. Then the non-functional requirements will be given. These are the bounds which the product should comply to. At last the constraints of the project will be discussed.

2.1. Functional requirements

These are the functional requirements of the system:

- The system is able to convert a digital data stream to an analog signal;
- The system is able to convert an analog signal to a digital data stream;
- The system is able to do digital-to-analog conversion and analog-to-digital conversion at the same time;
- The system is able to stream the data or have a one-second buffer;
- The system should have at least 2 different channels working and have an architecture that is scalable to up to 16 channels;
- The system is able to be tested under high pressure.

2.2. Non-functional requirements

Table 2.1: Non-Functional Requirements

ID	Requirement	Target / Description
R1	Sample rate	≥ 2 MS/s
R2	Dynamic range	≥ 160 dB/Hz
R3	Bandwidth	≥ 1 MHz
R4	Total harmonic distortion	$\text{THD}_{BW} < -100$ dB
R5	Data handling	Stream data or provide a 1-second buffer
R6	Number of channels	At least 2 channels
R7	Channel scalability	Scalable up to 16 channels
R8	User interface	Python or C(++) script

2.3. Project constraints

Table 2.2: Constraints

ID	Constraint	Limit / Condition
C1	Physical volume	$\leq 100\ \text{cm}^3$
C2	Development time	10 weeks
C3	Pressure rating	500 bar
C4	Operating temperature range	−20°C to +85°C

3

System Architecture

3.1. Subgroup Scope

The goal of the interface subgroup is to design a reliable datalink between the FPGA and the converter PCB. The datalink must transmit and receive data at the rate imposed by the system specifications, and ensure that the signals arrive at the converter ICs with sufficient integrity to meet the performance requirements. The datalink must also function within the constraints of the pressure chamber test setup. This thesis covers the key decisions made to achieve this goal. The work is divided per signal path between the two subgroup members. This chapter introduces the relevant constraints and the top-level decisions made to address them; each decision is elaborated in the chapters that follow.

3.2. Interface Subsystem Overview

During pressure testing, the converter PCB is placed inside the pressure chamber while the FPGA remains outside. The two are connected through a bulkhead connector, which is limited to 16 wires. These 16 wires must carry all signals crossing the boundary: the converter control and data lines, power supplies, and analog I/O connections. This constraint is also reinforced by the system's size limitation, which limits the number of traces connector on the PCB. These factors directly limit what interfaces are feasible and shapes every design decision in this thesis.

3.3. Serial Interface Selection

For high-resolution converters with a word length of 16 bits or more, a parallel interface would require at minimum one wire per data bit plus control lines, which will exceed the 16-wire budget before power and analog signals are even considered. A serial interface, transfers data sequentially over a single data line, keeping the wire count small and fixed regardless of the converter word length. The choice was therefore made to select converters that support a serial interface. The specific interface standard and the converters selected are discussed in Chapters 4 and 5.

3.4. Cable Allocation

The cable allocation was established collectively by the project group to satisfy the 16-wire bulkhead constraint. Table 3.1 lists the full allocation. The allocation covers all signals crossing the pressure boundary: converter control and data lines, power supplies, analog I/O, and the oscillator phase reference. The DAC serial clock is not included, as it is generated directly from the SiT8208 oscillator on the converter PCB and does not cross the bulkhead, as described in Chapter 6.

#	Signal	Direction	Description
1	$\overline{\text{CS}}$	FPGA → PCB	DAC chip select
2	SDI	FPGA → PCB	DAC serial data, channel 1
3	SDI1	FPGA → PCB	DAC serial data, channel 2
4	$\overline{\text{CS}}$1	FPGA → PCB	ADC chip select
5	SCK	FPGA → PCB	ADC SPI clock
6	CNV	FPGA → PCB	ADC conversion trigger
7	SDO0	PCB → FPGA	ADC serial data output
8	BSY	PCB → FPGA	ADC busy / ready signal
9	CLK1	PCB → FPGA	Oscillator phase reference
10	DGND	—	Digital ground reference
11	$+V_{\text{in}}$	PSU → PCB	Positive analog supply
12	$-V_{\text{in}}$	PSU → PCB	Negative analog supply
13	AGND	—	Analog ground reference
14	V_{ref}	PSU → PCB	Voltage reference
15	Ext	Meas. → PCB	Analog input signal
16	DAC_ext	PCB → Meas.	DAC analog output

Table 3.1: 16-wire bulkhead cable allocation

4

DAC Interface Design

This chapter covers the selection of the DAC, the SPI protocol analysis, and the design of the SPI master finite state machine.

4.1. DAC Selection

The DAC selected for this design is the DAC8811 from Texas Instruments [5]. It meets the required minimum resolution of 16 bits and offers high linearity, with a THD of −105 dB and an INL of ±1 LSB. More relevant for this design, the DAC8811 uses an SPI interface, which is well suited to the wire count constraint imposed by the bulkhead connector.

Higher-end converters with comparable or better analog performance were considered but ruled out due to their use of the JESD204B interface. While JESD204B [6] is a capable high-speed serial standard, it introduces significant implementation complexity that is disproportionate for a 2 MSps application and not feasible within the project time constraints. The full candidate comparison is provided in Appendix B.

SPI is a synchronous serial communication protocol commonly used for chip-to-chip communication [7]. In its full form it consists of four wires: a clock, a chip select, a data line from master to slave, and a data line from slave to master. Its simplicity and low wire count make it a natural fit for this application.

4.2. DAC8811 SPI Protocol

The DAC8811 uses a 3-wire serial interface consisting of a clock line (**SCLK**), a serial data input (**SDI**), and a chip select ($\overline{\textbf{CS}}$). Data is transferred MSB first into an internal 16-bit shift register. On each rising edge of **SCLK**, while $\overline{\textbf{CS}}$ is held low, the shift register advances by one bit. Once all 16 bits have been clocked in, $\overline{\textbf{CS}}$ is de-asserted. The rising edge of $\overline{\textbf{CS}}$ latches the contents of the shift register into the DAC output register, immediately updating the analog output.

The maximum frequency for **SCLK** is 50 MHz. For a 16-bit transfer, this results in a minimum transfer time of $16 \times 20\,\text{ns} = 320\,\text{ns}$, giving a theoretical maximum sample rate of 3.125 MSps. However, the DAC8811 requires 500 ns to settle to within ±0.015% of the final value [5]. Since the system requires the output to be fully settled before the next sample is loaded, the effective sample period is set to 500 ns, corresponding to a sample rate of 2 MSps exactly meeting the system requirement.

The key timing parameters of the DAC8811 SPI interface are summarised in Table 4.1. The chip select must be held low for the entire 16-bit transfer and de-asserted only after the last rising edge of **SCLK**. A minimum hold time of 10 ns must be observed between the last clock edge and the rising edge of $\overline{\text{CS}}$. Between consecutive transfers, $\overline{\text{CS}}$ must remain high for at least 20 ns before the next transfer can begin.

Parameter	Symbol	Constraint
SCLK frequency	f_{CLK}	$\leq 50\,\text{MHz}$
Data setup time	t_{DS}	$\geq 5\,\text{ns}$
Data hold time	t_{DH}	$\geq 10\,\text{ns}$
CS setup time	t_{CSS}	$\geq 0\,\text{ns}$
CS hold time	t_{CSH}	$\geq 10\,\text{ns}$
CS high time	t_{CSmin}	$\geq 20\,\text{ns}$

Table 4.1: DAC8811 SPI timing parameters [5].

Figure 4.1 shows the SPI timing waveforms for one complete 16-bit transfer, as specified in the DAC8811 datasheet [5].

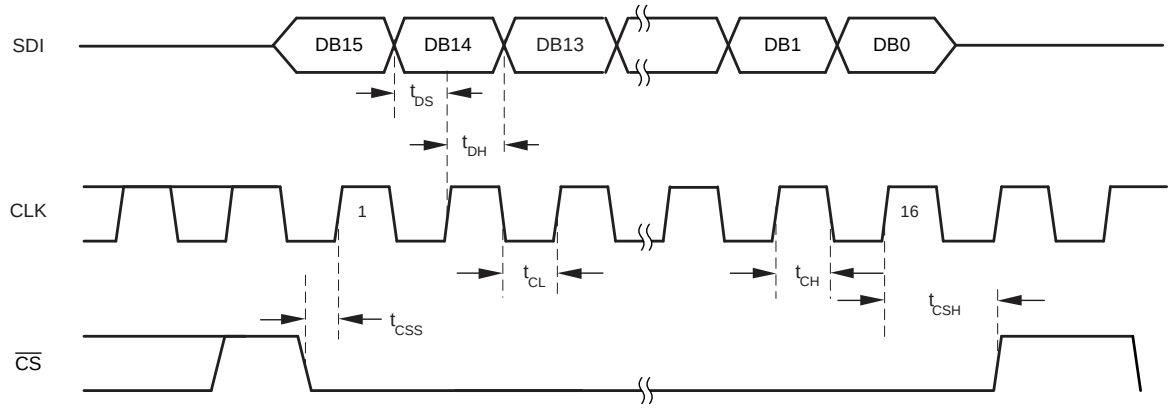


Figure 4.1: DAC8811 SPI timing diagram [5].

4.3. SPI Master FSM Design

Figure 4.2 shows the entity interface of the SPI master. The `Num_Channels` generic allows the design to scale to multiple DAC channels without modification to the core logic.

Figure 4.2: SPI master entity interface diagram.

The SPI master is implemented as a finite state machine with five states [8]. The implementation uses two processes: a combinational process for next-state logic and a sequential process clocked on the rising edge of the clock for state registration. A third process clocked on the falling edge drives the SDI output, ensuring data is stable before the DAC samples it on the rising edge [9]. Figure 4.3 shows the state diagram.

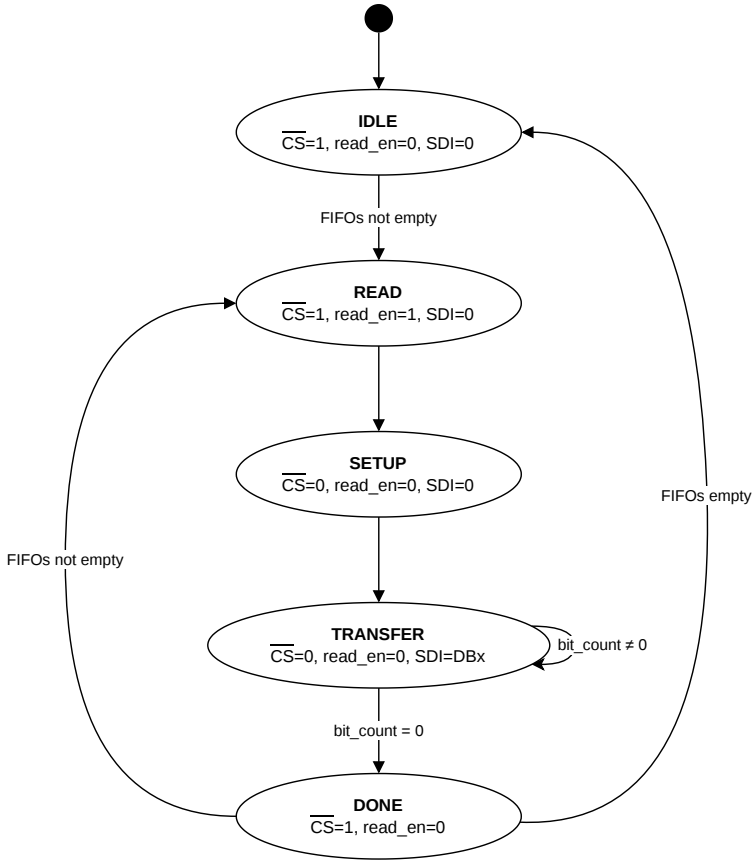


Figure 4.3: FSM diagram of SPI master for DAC.

IDLE The master waits in this state until data is available in the FIFO. $\overline{\text{CS}}$ is held high and no data is driven on SDI.

READ The read enable signal is asserted for one cycle to pull the next sample from the FIFO. The data is latched into the internal shift register.

SETUP $\overline{\text{CS}}$ is asserted one cycle before the transfer begins. This gives the DAC one clock cycle of setup time before the first bit is presented.

TRANSFER The shift register is shifted out MSB first, one bit per clock cycle. The DAC samples SDI on each rising edge of SCLK. After 16 bits, the state transitions to DONE.

DONE $\overline{\text{CS}}$ is de-asserted, latching the received word into the DAC output register. The state machine then waits for `DONE_WAIT_CYCLS` additional cycles before returning to READ or IDLE. This wait period, combined with the transfer time, enforces a total cycle time of 500 ns, corresponding to exactly 2 MSps.

The timing of $\overline{\text{CS}}$ was subsequently adjusted to compensate for the reclocking chain propagation delay, as described in Chapter 6.

The SPI master is designed to support a configurable number of channels through the `Num_Channels` generic. The `data_in` port scales accordingly to `Num_Channels` $\times$ 16 bits, and `read_en` is a vector of size `Num_Channels`, allowing each channel's FIFO to be read independently. Data is transmitted over multiple SDI lines, one per channel, while all DAC chips share the same $\overline{\text{CS}}$ and SCLK signals, ensuring

all channels are updated simultaneously. In the current test setup, a single channel is routed through the pressure chamber bulkhead. Additional channels can be added outside the chamber without any modification to the interface design.

5

ADC Interface Design

This chapter covers the selection of the ADC, the AD4630-24 interface operation, and the FPGA implementation of the configuration and acquisition controllers.

5.1. ADC Chip Selection

Several ADCs were evaluated against the requirements defined in Chapter 2. The complete comparison is provided in Appendix B. Key selection criteria included the minimum sample rate of 2 MSPS (R1), dynamic range (R2), distortion performance (R4), scalability to 16 channels (R7), and the system volume constraint (C1).

Of the evaluated devices, the AD4630 offered the best overall performance, providing the highest dynamic range and the lowest specified THD+N. It also meets the required 2 $MSPS$ sample rate while maintaining relatively low power consumption. These characteristics make it the most suitable candidate for meeting the signal-quality requirements of the system.

An additional advantage of the AD4630 is its dual-channel architecture. Since the project requires a scalable solution supporting up to 16 channels, integrating two channels per package reduces the number of ADCs required and simplifies PCB area utilisation compared to single-channel alternatives.

The AD4630 also supports a flexible SPI interface, which aligns with the limited signal count available between the FPGA and converter PCB. Based on these considerations, the AD4630 was selected as the ADC for this project.

5.2. AD4630 Interface Operation

The AD4630 communicates with the FPGA through a 6-wire SPI interface consisting of **CNV**, $\overline{\textbf{CS}}$, **MOSI(SDI)**, **MISO(SDO)**, **SCK**, and **BUSY**. The **CNV** signal initiates a conversion, $\overline{\textbf{CS}}$ enables SPI communication, **SCK** provides the serial clock, **MOSI** is used to transmit configuration data to the ADC, **MISO** returns conversion data to the FPGA, and **BUSY** indicates when a conversion is in progress. The operation and timing requirements described in this section are derived from the AD4630 datasheet [10].

The device operates in two distinct modes: *Configuration Mode* and *Conversion Mode*. In Configuration Mode, the ADC registers can be accessed through the SPI interface to configure operating parameters such as the output format and operating mode. Once configuration is complete, the ADC is switched into Conversion Mode, where register access is disabled and the device continuously performs analog-to-digital conversions.

For this project, the ADC was configured to operate in **Zone 2 (SPI-Compatible Mode, 1-lane SDR)**. This mode was selected because it minimises the number of interface signals required between the FPGA and the converter PCB while still supporting the required 2 MSPS sample rate. In addition,

SPI-Compatible Mode allows the FPGA to directly generate the serial clock and control all interface timing.

Configuration Mode timing is shown in Figure 5.1. During this phase, the ADC registers are accessed through the SPI interface to configure the device before entering Conversion Mode.

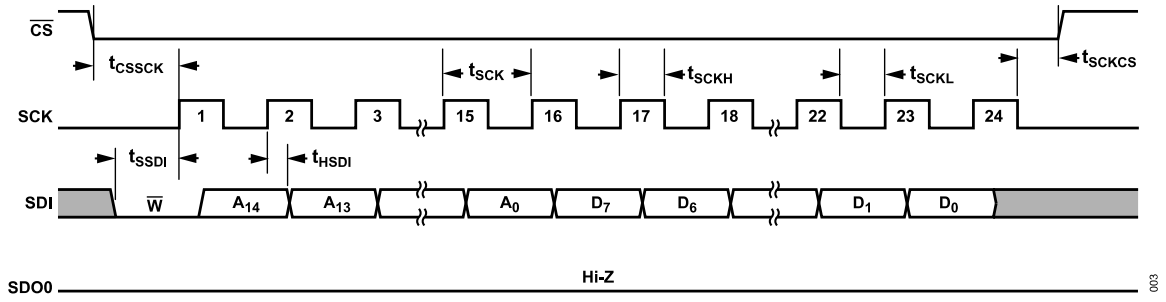


Figure 5.1: AD4630 register write timing diagram.

The timing requirements relevant to register configuration are summarised in Table 5.1. These constraints define the minimum setup and hold times that must be satisfied during SPI register transactions.

Table 5.1: Configuration mode timing constraints.

Parameter	Symbol	Requirement
CS to first SCK edge	t_{CSSCK}	$\geq 9.8\,ns$
Last SCK to CS edge	t_{SCKCS}	$\geq 4.2\,ns$
SDI setup time	t_{SSDI}	$\geq 1.5\,ns$
SDI hold time	t_{HSDI}	$\geq 1.5\,ns$

In Conversion Mode, a rising edge on the **CNV** signal initiates a new conversion. The AD4630 supports two data-transfer zones. In Zone 1, the conversion result is read immediately following the completion of the conversion. In Zone 2, the readout of Sample (N) is delayed until after the start of Sample (N+1). This increases the available data-transfer window and relaxes the SPI timing requirements. The difference between Zone 1 and Zone 2 operation is illustrated in Figure 5.2.

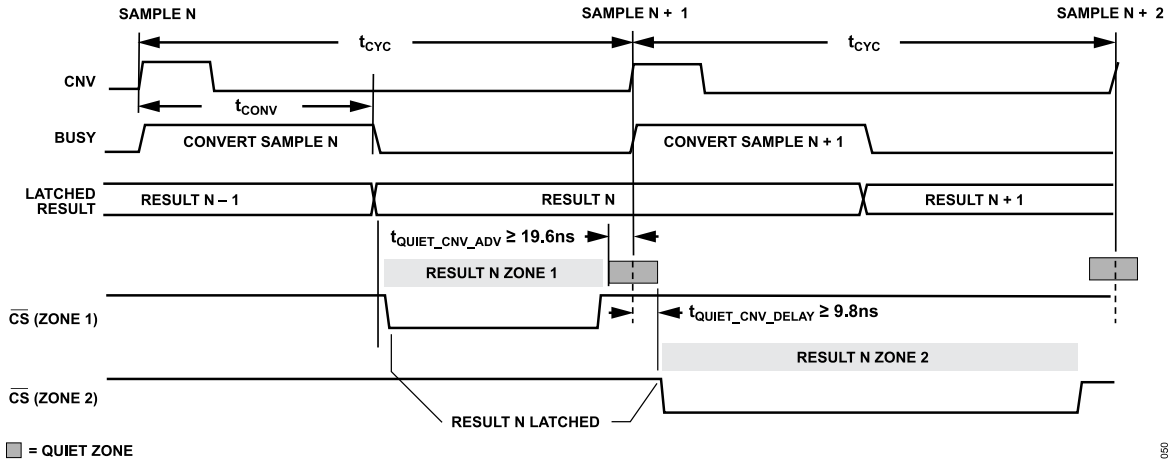


Figure 5.2: AD4630 data-transfer timing zones.

At the maximum sample rate of 2 $MSPS$, the conversion cycle period is $t_{CYC} = 500\,ns$ and the typical conversion time is $t_{CONV} = 282\,ns$.

To minimise coupling of digital switching noise into the analog circuitry, the datasheet specifies quiet zones around the rising edge of the **CNV** signal. The quiet time immediately before the conversion start is $t_{QUIET\_CNV\_ADV} = 19.6\,ns$, while the quiet time immediately after the conversion start is $t_{QUIET\_CNV\_DELAY} = 9.8\,ns$.

For Zone 2 operation, the available data transfer window is

$$t_{READ} = t_{CYC} - t_{QUIET\_CNV\_ADV} - t_{QUIET\_CNV\_DELAY} \tag{5.1}$$

Substituting the values specified in the datasheet gives

$$t_{READ} = 500\,ns - 19.6\,ns - 9.8\,ns = 470.6\,ns \tag{5.2}$$

resulting in an available read window of approximately $470.6\,ns$ at a sample rate of $2\,MSPS$.

The timing constraints governing the conversion process are summarised in Table 5.2.

Table 5.2: Conversion timing constraints.

Parameter	Symbol	Requirement
Conversion cycle period	t_{CYC}	$500\,ns$
Conversion time	t_{CONV}	$282\,ns$ (typ.)
Pre-CNV quiet time	$t_{QUIET\_CNV\_ADV}$	$\geq 19.6\,ns$
Post-CNV quiet time	$t_{QUIET\_CNV\_DELAY}$	$\geq 9.8\,ns$

Once the conversion result becomes available, the data is retrieved through the SPI interface. The timing relationships relevant to SPI data retrieval are shown in Figure 5.3.

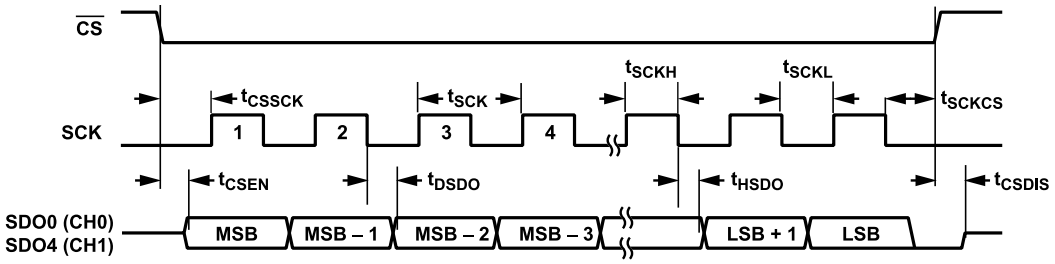


Figure 5.3: AD4630 Zone 2 SPI readout timing.

The timing constraints relevant to SPI data retrieval are summarised in Table 5.3.

Table 5.3: Zone 2 SPI readout timing constraints.

Parameter	Symbol	Requirement
SPI clock period	t_{SCK}	$\geq 9.8\,ns$
CS to first SCK edge	t_{CSSCK}	$\geq 9.8\,ns$
Data valid delay	t_{DSDO}	$\leq 5.6\,ns$

The timing requirements presented in this section impose several constraints on the FPGA interface. During start-up, the controller must perform the required register configuration while satisfying the SPI setup and hold requirements. During normal operation, conversions must be initiated at a fixed $2\,MSPS$ rate, the quiet zones around the CNV signal must be respected, and conversion data must be retrieved within the available Zone 2 read window. These requirements form the basis of the FPGA architecture described in the following section.

5.3. FPGA Interface Implementation

The FPGA interface was implemented as a hierarchical state-machine architecture consisting of a top-level controller, a configuration controller, and an acquisition controller. The timing requirements presented in Section 5.2 form the basis of the implementation. The top-level controller is responsible for sequencing the ADC start-up procedure, the configuration controller performs the required register transactions, and the acquisition controller generates the conversion timing and retrieves conversion results.

5.3.1. Top-Level ADC Controller

The top-level controller coordinates the overall operation of the ADC interface. Its state diagram is shown in Figure 5.4.

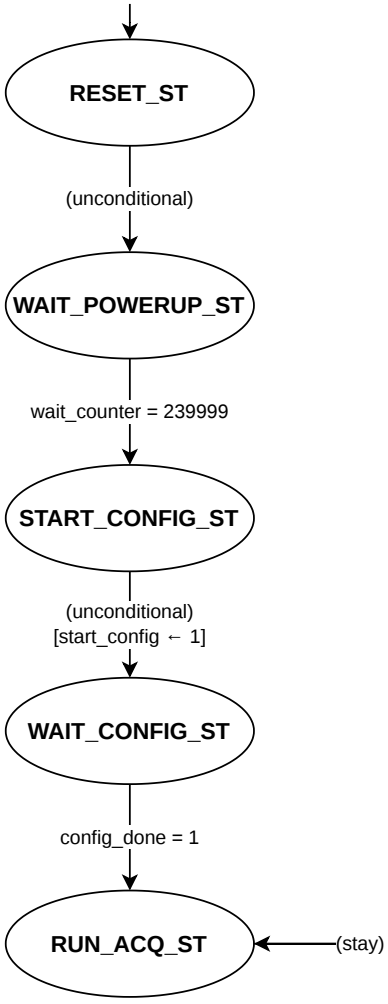


Figure 5.4: Top-level ADC controller state machine.

Following reset, the controller enters the `WAIT_POWERUP_ST` state, where a fixed delay is applied to satisfy the ADC power-up timing requirements specified in the AD4630 datasheet [10]. Once this delay has elapsed, the controller asserts the `start_config` signal and transitions to `WAIT_CONFIG_ST`.

The controller remains in this state until the configuration sequence has completed. Upon reception of the `config_done` signal from the configuration controller, the top-level controller enters `RUN_ACQ_ST`, where normal data acquisition begins. The controller remains in this state for the remainder of operation.

5.3.2. Configuration Controller

The configuration controller performs the register transactions required to initialise the AD4630 before entering Conversion Mode. The controller implements a dedicated SPI master designed to satisfy the timing requirements summarised in Table 5.1.

Figure 5.5 shows the configuration-controller state machine.

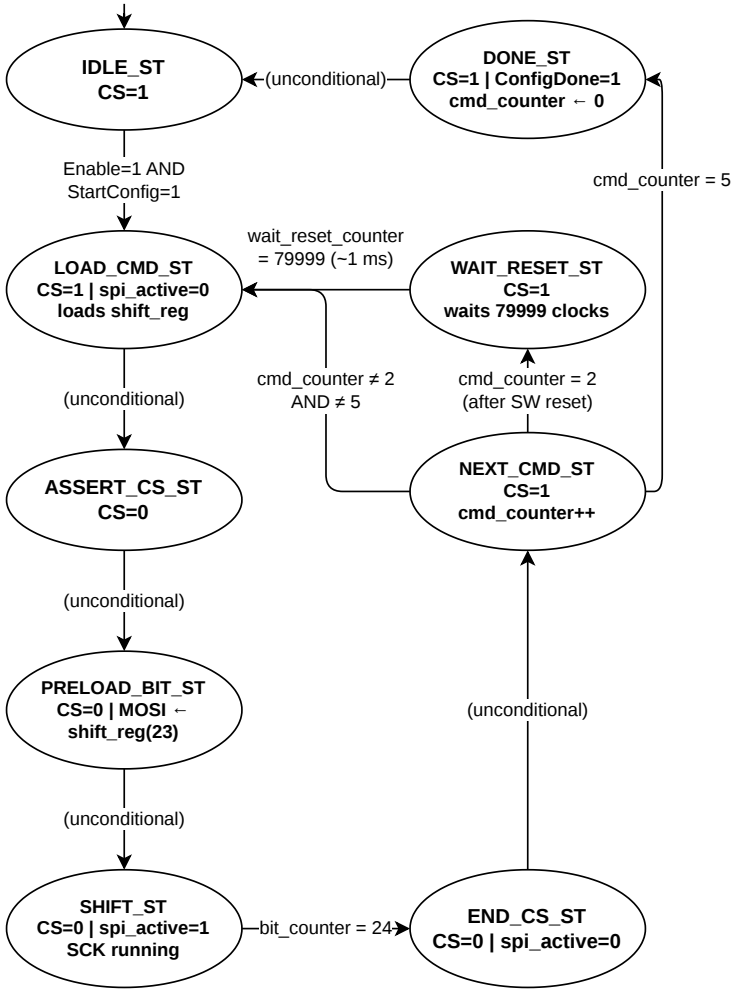


Figure 5.5: Configuration controller state machine.

The configuration sequence consists of five SPI transactions derived from the AD4630 register map [10]. These commands configure the ADC for SPI-Compatible Mode, Zone 2 operation, and single-lane SDR data transfer. A software reset command is included as part of the sequence, after which the controller inserts the reset delay specified in the datasheet before continuing with the remaining configuration

commands.

When configuration is started, the controller loads a predefined command into a shift register and asserts $\overline{\mathbf{CS}}$. A dedicated state is used to satisfy the minimum delay between the assertion of $\overline{\mathbf{CS}}$ and the first SPI clock edge (t_{CSSCK}). The most significant bit is then preloaded onto the SDI line before clock generation begins.

The command word is transmitted in the `SHIFT_ST` state, where a 24-bit SPI transfer is performed. After the final bit has been transmitted, the controller waits the required delay between the final clock edge and the deassertion of $\overline{\mathbf{CS}}$ (t_{SCKCS}). Once all configuration commands have been sent, the controller asserts the `ConfigDone` signal and returns to the idle state.

5.3.3. Acquisition Controller

The acquisition controller is responsible for generating the conversion timing and retrieving conversion results from the ADC. The controller was designed directly from the timing requirements presented in Section 5.2, in particular the quiet-zone constraints listed in Table 5.2 and the SPI timing requirements listed in Table 5.3.

The controller state machine is shown in Figure 5.6.

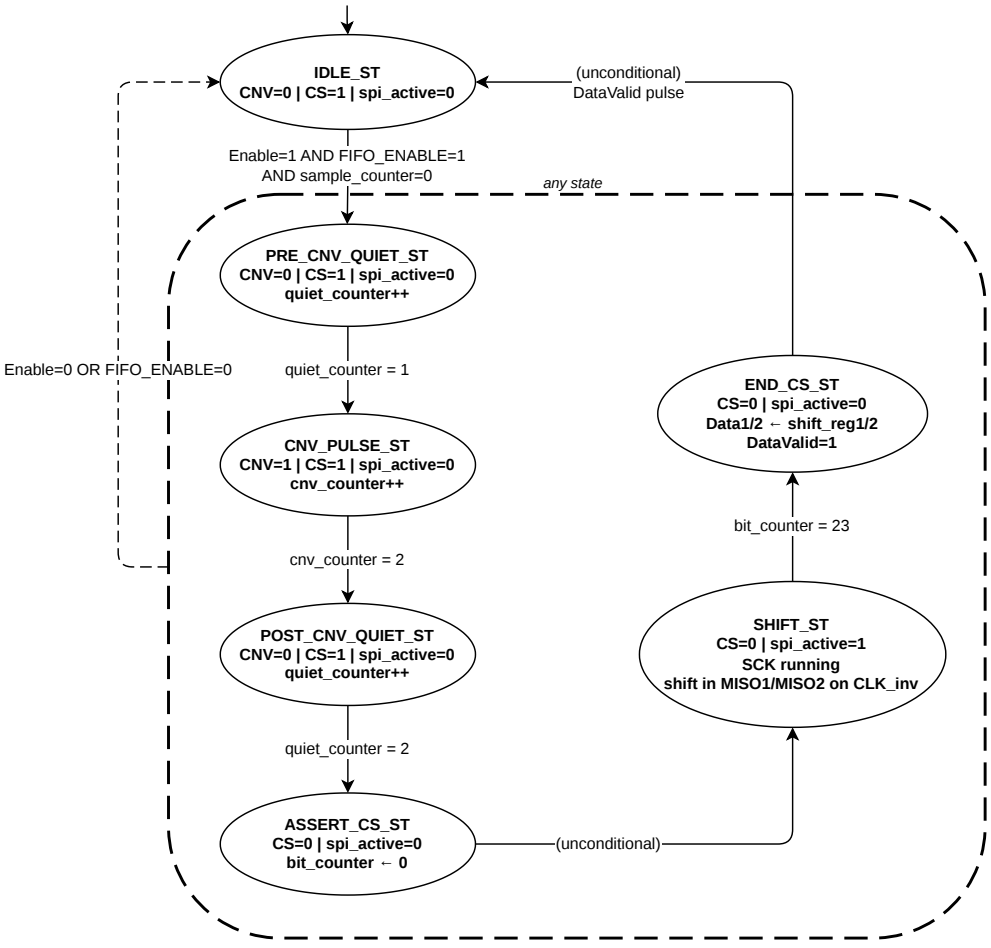


Figure 5.6: Acquisition controller state machine.

A conversion cycle begins in the `PRE_CNV_QUIET_ST` state, which implements the pre-conversion quiet period $t_{QUIET\_CNV\_ADV}$. The controller then enters `CNV_PULSE_ST`, where the CNV signal is asserted to

initiate a conversion. Following the pulse, the controller enters `POST_CNV_QUIET_ST`, implementing the required post-conversion quiet period $t_{QUIET\_CNV\_DELAY}$.

After the quiet period has elapsed, the controller asserts $\overline{\mathbf{CS}}$ and begins the SPI readout sequence. Data retrieval is performed in the `SHIFT_ST` state, where the serial clock is generated and conversion data is shifted into the FPGA.

As described in Section 5.2, the ADC is operated in Zone 2 SPI-Compatible Mode. This mode provides an available data-transfer window of 470.6 ns. The SPI clock is generated at 80 MHz, corresponding to a clock period of

$$T_{SCK} = 12.5\,ns.$$

This exceeds the minimum clock-period requirement of $t_{SCK} \geq 9.8\,ns$ specified in Table 5.3. A complete 24-bit SPI transfer therefore requires

$$t_{SHIFT} = 24 \cdot T_{SCK} = 24 \cdot 12.5\,ns = 300\,ns.$$

The complete transfer therefore fits comfortably within the available Zone 2 read window, leaving approximately 170 ns of timing margin before the next conversion cycle begins. The decision to leave such a big margin was made to have a greater timing budget to solve possible timing issues and trivial implementation of a 2 MHz CNV signal.

The ADC output data is sampled on the inverted FPGA clock. Sampling on the opposite clock edge increases the available setup margin and satisfies the t_{DSDO} timing requirement specified in the AD4630 datasheet [10].

A sample counter is used to maintain the conversion timing. Unlike the state-specific counters used for the quiet periods and CNV pulse generation, the sample counter runs continuously throughout the entire acquisition cycle. The counter is therefore independent of the current FSM state and defines the overall 500 nsconversion period corresponding to the required 2 $MSPS$ sample rate. Individual states occupy fixed portions of this cycle while the sample counter provides the global timing reference.

To allow data streaming to be stopped without requiring a reset of the acquisition engine, a `FIFO_ENABLE` signal is monitored continuously. If `FIFO_ENABLE` is deasserted, the controller immediately returns to the idle state regardless of the current state of the acquisition cycle. This mechanism allows data transfer to the FIFO to be halted safely without waiting for the completion of an ongoing conversion sequence.

After all bits have been received, the conversion results are transferred to the output registers and a single-cycle data-valid pulse is generated before the next acquisition cycle begins.

5.3.4. BUSY Signal Usage

The AD4630 provides a BUSY output that indicates when a conversion is in progress. Although this signal is available on the PCB, it was not used as the primary timing reference in the final implementation.

During PCB assembly, rework modifications introduced additional propagation delay on signals travelling from the FPGA to the ADC. Since BUSY is generated directly by the ADC, synchronising interface timing to BUSY would not account for the delayed arrival of FPGA-generated control signals at the ADC inputs. As a result, a BUSY-based implementation would provide no advantage while introducing additional timing uncertainty.

Instead, a deterministic timing schedule derived directly from the datasheet parameters was adopted. All conversion and readout timing is generated from the FPGA system clock and the timing constraints discussed in Section 5.2, ensuring repeatable operation independent of BUSY timing variations.

6

Clock Architecture and Reclocking

This chapter analyses the jitter problem introduced by the FPGA PLL, presents the external reclocking solution, and derives the jitter budget and its impact on the SPI master timing.

6.1. The Jitter Problem

In an ideal clock signal, transitions between logic levels occur at perfectly regular intervals. In practice, noise in the oscillator circuit and the driving logic causes these transitions to occur slightly earlier or later than expected. This timing uncertainty is known as jitter [11], [12]. As a result, any signal derived from or synchronised to a jittery clock inherits that uncertainty in its own edges.

In this design, the FPGA generates the control signals $\overline{\text{CS}}$ and CNV from its internal PLL clock. The Lattice ECP5 PLL specifies a period jitter of 100 ps peak-to-peak [13]. This jitter is inherited by both control signals.

For the DAC, $\overline{\text{CS}}$ jitter introduces timing uncertainty on the latch event. The DAC8811 requires a minimum CS hold time $t_{\text{CSH}} \geq 10\,\text{ns}$ and a minimum CS high time $t_{\text{CSmin}} \geq 20\,\text{ns}$ [5]. A jitter of 100 ps peak-to-peak represents up to 1% of these margins. This is a non-negligible fraction at 2 MSps where timing margins are already tight.

For the ADC, the effect is more severe. The rising edge of CNV triggers the sample-and-hold circuit of the AD4630-24 [10], freezing the analog input at that instant. Any timing uncertainty on this edge directly translates to uncertainty in the sampled voltage. For a sinusoidal input at frequency f_{in}, the resulting SNR degradation is given by [14], [15]:

$$\text{SNR} = -20\log_{10}(2\pi \cdot f_{\text{in}} \cdot t_j) \tag{6.1}$$

where t_j is the RMS aperture jitter. Assuming a Gaussian distribution [11], the 100 ps peak-to-peak PLL jitter corresponds to approximately 16.7 ps RMS [13]. Substituting into equation 6.1 at the maximum input frequency of 1 MHz:

$$\text{SNR} = -20\log_{10}(2\pi \times 10^{6} \times 16.7 \times 10^{-12}) \approx 89\,\text{dB} \tag{6.2}$$

This falls 11 dB short of the 100 dB system requirement. From the system SNR requirement, the maximum allowable aperture jitter on CNV is derived by rearranging equation 6.1:

$$t_{j,\text{max}} = \frac{1}{2\pi \cdot f_{\text{in}} \cdot 10^{\text{SNR}/20}} = \frac{1}{2\pi \times 10^{6} \times 10^{100/20}} \approx 1.6\,\text{ps RMS} \tag{6.3}$$

The FPGA PLL jitter of 16.7 ps RMS exceeds this budget by a factor of ten. A reclocking solution is therefore required for both $\overline{\text{CS}}$ and CNV before they reach the converter ICs.

6.2. Reclocking

The solution to the jitter problem is reclocking [15], [16]. The principle is straightforward: instead of driving the converter control signals directly from the FPGA, the signals are passed through an external D flip-flop clocked by a low-jitter oscillator. The flip-flop samples the FPGA signal on each rising edge of the oscillator clock, and the output transitions are aligned to the oscillator edges rather than the original FPGA edges. The jitter of the output signal is therefore determined by the oscillator, not the FPGA PLL.

For this to be effective, the oscillator and flip-flops must be placed as close as possible to the converter ICs on the PCB, minimising the propagation delay between the reclocked signal and the converter control inputs. This motivated the decision to place the oscillator inside the pressure chamber on the converter PCB, rather than outside. This placement has two additional benefits: it eliminates the need for a dedicated clock wire through the bulkhead, and ensures that all converter timing is referenced to the same physical oscillator, with no asynchronous clock domain crossing across the pressure boundary.

"An important property of this approach is that the oscillator frequency does not need to match the rate of the signal being reclocked [15], [16]. In this design, $\overline{\text{CS}}$ and CNV toggle at 2 MHz, while the reclocking oscillator runs at 50 MHz. At this frequency, 25 oscillator ticks occur within each $\overline{\text{CS}}$ or CNV cycle. The flip-flop acts as a transparent relay. It simply re-edges the signal at the next oscillator tick, introducing no synchronisation complexity because the FPGA and oscillator share a known phase relationship through the system architecture.

6.3. Reclocking Circuit

Figure 6.1 shows the reclocking circuit. The chain consists of four components: a MEMS oscillator, a fanout buffer, two ECL flip-flops, and two level translators. Each component serves a specific role in delivering a clean, low-jitter control signal to the converter ICs.

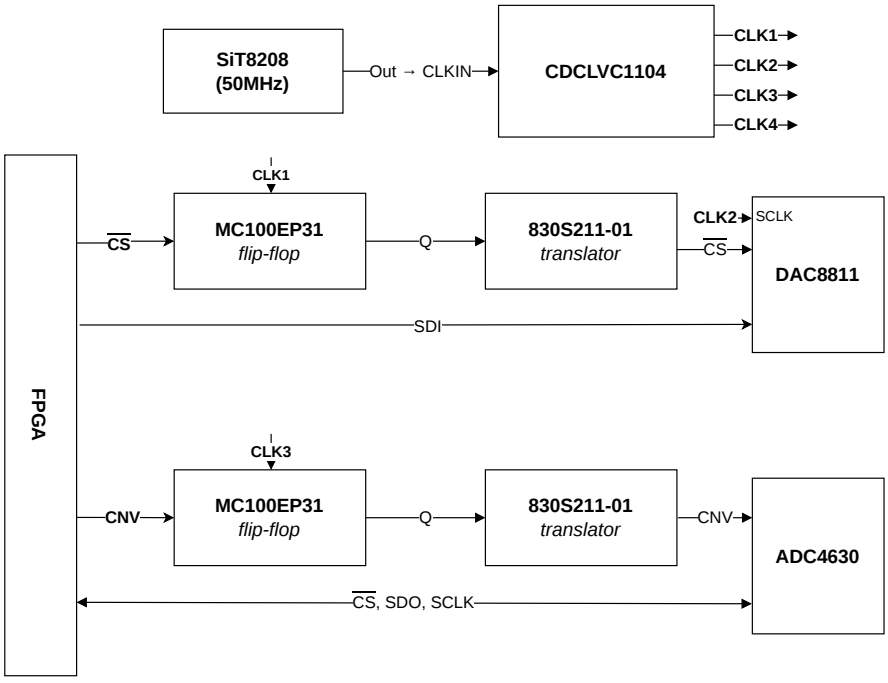


Figure 6.1: Reclocking circuit block diagram.

SiT8208 MEMS Oscillator The SiT8208 is a MEMS oscillator running at 50 MHz with a phase jitter of 500 fs RMS integrated from 12 kHz to 20 MHz [17]. It serves as the low-jitter reference clock for the entire reclocking chain.

DAC SCLK A dedicated output of the SiT8208 is routed directly to the DAC8811 as SCLK [5]. This provides a clean serial clock for the SPI transfer directly from the oscillator, without passing through the reclocking chain. It also eliminates the need for the FPGA to generate and send the DAC clock through the bulkhead, saving one wire.

CDCLVC1104 Fanout Buffer The SiT8208 output is distributed through the CDCLVC1104 fanout buffer [18]. The buffer serves two purposes. First, it fans out the oscillator clock to both MC100EP31 flip-flop clock inputs, preventing the oscillator output from being loaded by multiple clock inputs directly. Second, one buffer output drives the CLKin signal through the bulkhead to the FPGA, providing a phase reference for synchronisation. A dedicated buffer output is used for this path to ensure the bulkhead crossing does not degrade the clock signal seen by the flip-flops. The CDCLVC1104 provides four matched output copies with minimal additive jitter.

MC100EP31 ECL D Flip-Flop Two MC100EP31 ECL D flip-flops [19] perform the actual reclocking. The $\overline{\text{CS}}$ signal from the FPGA connects to the D input of the first flip-flop, and CNV connects to the D input of the second. On each rising edge of the 50 MHz oscillator clock, the flip-flop samples its D input and presents the result at Q, aligning the output edge to the clean oscillator reference. ECL logic is chosen for its sub-picosecond additive jitter [19], which ensures the flip-flop itself does not significantly degrade the oscillator's jitter performance.

830S21AMI Level Translator The MC100EP31 produces LVPECL output levels, which are not compatible with the LVCMOS inputs of the DAC8811 and AD4630-24. The 830S21AMI [20] translates the LVPECL output to 3.3 V LVCMOS, making the reclocked signals compatible with the converter control inputs. One translator is used per signal path, giving two translators in total, one for $\overline{\text{CS}}$ and one for CNV.

6.4. Jitter Budget

The total jitter of the reclocking chain is modelled as the root-sum-of-squares (RSS) of the individual component contributions, assuming uncorrelated noise sources [11]:

$$t_{j,\text{total}} = \sqrt{t_{j,\text{osc}}^2 + t_{j,\text{buf}}^2 + t_{j,\text{ff}}^2 + t_{j,\text{tr}}^2} \tag{6.4}$$

Table 6.1 lists the jitter contribution of each component in the chain.

Component	Part	Jitter (ps RMS)
MEMS Oscillator	SiT8208	0.50 [17]
Fanout Buffer	CDCLVC1104	0.10 [18]
ECL Flip-Flop	MC100EP31	1.00 [19]
Level Translator	830S21AMI	0.11 [20]
Total (RSS)		**1.14**
Budget		1.60
Margin		0.46

Table 6.1: Reclocking chain jitter budget.

The total chain jitter of 1.14 ps RMS is within the 1.6 ps RMS budget derived in Section 6.1, leaving a margin of 0.46 ps. The dominant contributor is the MC100EP31 flip-flop, contributing 1.00 ps RMS, which dominates the total.

6.5. Impact on SPI Master Timing

The reclocking chain introduces a propagation delay between the FPGA output and the converter control input. In the reclocked design, the DAC SCLK is generated directly from the SiT8208 oscillator on the

PCB and is not driven by the FPGA. The $\overline{\text{CS}}$ signal, however, originates from the FPGA and passes through the reclocking chain before reaching the DAC. This creates a relative delay between SCLK and $\overline{\text{CS}}$ at the DAC input.

The relative delay is modelled as:

$$\delta_{\text{CS}} = T_{\text{FF}} + t_{\text{CKQ}} + t_{\text{PD}} + \Delta_{\text{trace}} \tag{6.5}$$

where T_{FF} is the time until the flip-flop registers $\overline{\text{CS}}$ on the next oscillator edge (always exactly one full clock period), t_{CKQ} is the MC100EP31 clock-to-Q propagation delay, t_{PD} is the 830S21AMI propagation delay, and Δ_{trace} accounts for the routing difference between the $\overline{\text{CS}}$ and SCLK paths on the PCB.

Term	Description	Typical	Max
T_{FF}	Flip-flop registration delay (1 clock period)	20 ns	20 ns
t_{CKQ}	MC100EP31 CLK-to-Q [19]	0.34 ns	0.41 ns
t_{PD}	830S21AMI propagation [20]	0.95 ns	1.95 ns
Δ_{trace}	PCB routing difference	0 ns	1 ns
δ_{CS}	Total relative delay	21.3 ns	22.4 ns

Table 6.2: $\overline{\text{CS}}$ relative delay budget with respect to SCLK.

The dominant term is $T_{\text{FF}} = 20\,\text{ns}$, corresponding to exactly one oscillator clock period. This term is fully deterministic. The remaining 2.4 ns arises from analog propagation delays in the reclocking chain components. The total delay means that $\overline{\text{CS}}$ arrives at the DAC8811 approximately one SCLK period later than it would without reclocking. To compensate, the DAC SPI master FSM was modified to assert $\overline{\text{CS}}$ one clock cycle earlier, in the READ state rather than the SETUP state. This ensures that $\overline{\text{CS}}$ arrives at the DAC8811 within the required setup time t_{CSS}, as listed in Table 4.1.

The reclocking circuit described above was implemented on the converter PCB by the circuit design subgroup, including the KiCad schematic and PCB layout.

7

Verification and Results

7.1. DAC

7.1.1. Verification

The SPI signals were captured using an oscilloscope. As the oscilloscope could not reliably capture the 50 MHz SCLK, the clock frequency was temporarily reduced to allow reliable triggering. The captured waveforms were post-processed to remove interference and present clean digital transitions. Figure 7.1 shows $\overline{\text{CS}}$, SCLK, and SDI for one transfer of `0x5555`, confirming correct Mode 0 operation on hardware.

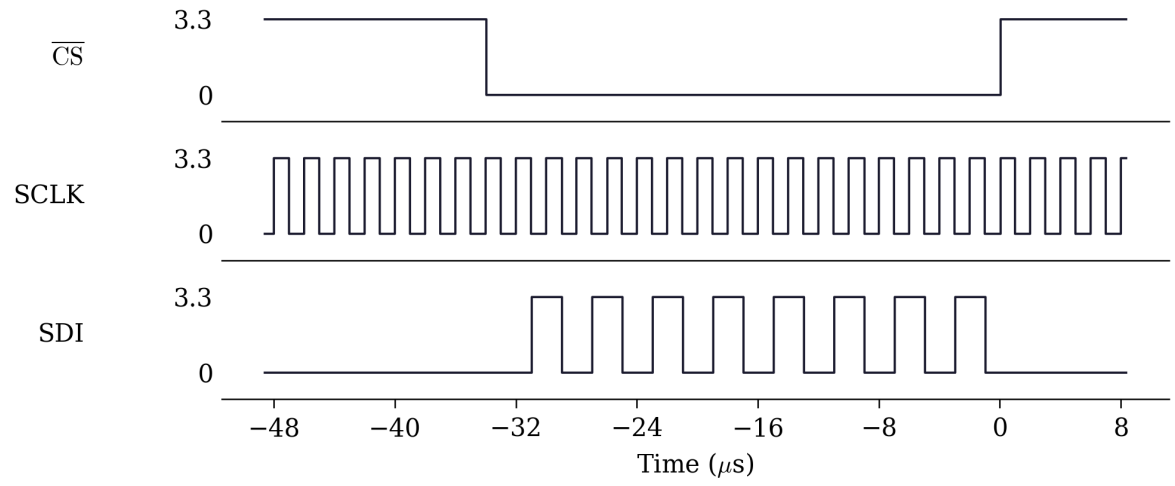


Figure 7.1: Oscilloscope capture of $\overline{\text{CS}}$, SCLK, and SDI for one 16-bit transfer of `0x5555`. Clock frequency reduced for capture; waveforms post-processed to remove interference.

The DAC8811 EVM was then connected to verify that the FSM produces a correct analog output in response to the SPI transfers. Writing constant codes across the full range confirmed that the DAC output scales correctly with the input word, validating the complete digital-to-analog signal path from the FPGA through the SPI interface to the analog output.

7.1.2. Results

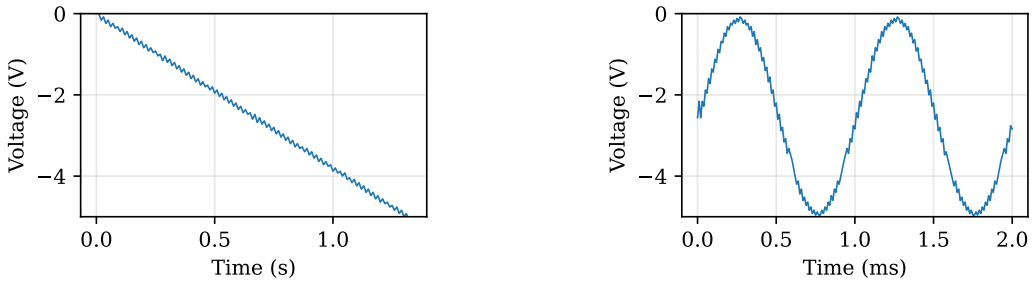
The DAC8811 is a multiplying DAC with an inverting output characteristic. The output voltage is given by [5]:

$$V_{\text{out}} = -V_{\text{ref}} \cdot \frac{D}{2^{16}} \tag{7.1}$$

where D is the 16-bit input code and V_{ref} is the reference voltage, set to 5 V in this design. As a result, a ramp from `0x0000` to `0xFFFF` produces a decreasing output voltage from 0 V to −5 V, as visible in Figure 7.2a.

The DAC interface was tested on the final converter PCB with the reclocking circuit integrated, outside the pressure chamber. Two test signals were generated to verify the analog output quality.

Figure 7.2 shows the DAC output for a full-scale ramp and a 1 kHz sinusoidal input. The ramp output decreases linearly across the full 16-bit code range, confirming correct end-to-end data transfer and monotonic behaviour across all codes. The sine wave is correctly reproduced, demonstrating that the interface sustains 2 MSps operation and that the DAC8811 can faithfully reproduce signals within the target acoustic band.



(a) Full-scale ramp output.

(b) 1 kHz sinusoidal output.

Figure 7.2: DAC8811 analog output measured on the final converter PCB with reclocking, outside the pressure chamber.

These results confirm that the SPI master, reclocking circuit, and DAC8811 operate correctly together as a complete interface. The reclocked PCB was subsequently used for the pressure characterisation conducted by the full project group.

7.2. ADC

7.2.1. Verification

The functionality of the ADC controller was verified using simulation in GHDL and waveform inspection in GTKWave. Figure 7.3 shows the complete ADC initialisation sequence and the subsequent conversion cycle.

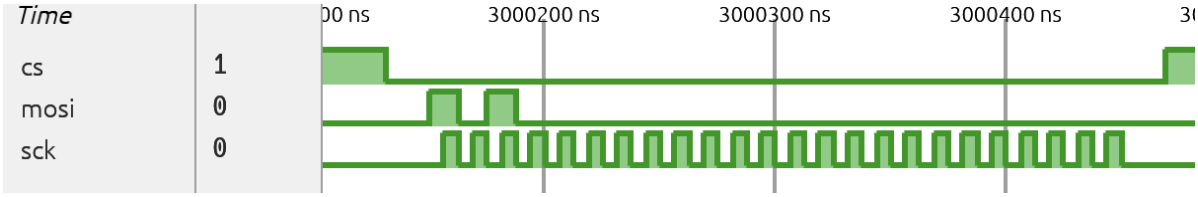
During configuration, six 24-bit SPI commands are transmitted to the ADC. Chip-select remains asserted for the duration of each transfer while the serial clock generates 24 clock pulses to shift the command word into the device. Once configuration is complete, the controller enters continuous acquisition mode.

During acquisition, a conversion is initiated by the `CNV` pulse every 500 ns, corresponding to a sampling rate of 2 $MSPS$. Following the conversion interval, the controller asserts chip-select and reads the 24-bit conversion result using the SPI interface. The simulated timing confirms correct operation of both the configuration state machine and the acquisition engine.

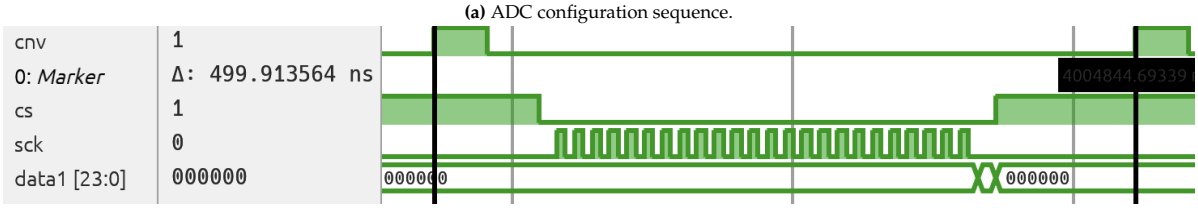
As the logic analyser could not reliably capture the 80 MHz SPI clock, the FPGA clock divider was temporarily adjusted to reduce the SPI clock frequency to approximately 413 kHz. Since only the clock divider was modified, all timing relationships remained unchanged and could be scaled back to the nominal operating frequency.

The scaling factor between the measured and operating frequencies is

$$\text{Scale Factor} = \frac{80\,\text{MHz}}{413.223\,\text{kHz}} \approx 193.6. \tag{7.2}$$



(a) ADC configuration sequence.



(b) Continuous conversion operation at 2 MSPS.

Figure 7.3: Simulation results obtained using GHDL and GTKWave.

Figure 7.4 shows the SPI configuration transaction captured by the logic analyser. The measured clock edge spacing was approximately 1.2 μs, which scales to

$$\frac{1.2\,\mu\text{s}}{193.6} \approx 6.2\,\text{ns}, \tag{7.3}$$

corresponding to a clock period of 12.5 ns and therefore an SPI clock frequency of 80 MHz.

Figure 7.4: Logic analyser capture of the ADC configuration SPI transaction

Figure 7.5 shows the ADC conversion timing sequence. The measured timing intervals of 4.84 μs, 29.04 μs and 96.8 μs scale to 25 ns, 150 ns and 500 ns respectively at 80 MHz. These correspond to 2, 12 and 40 FPGA clock cycles, meaning that in theory the generated timing signals should match the intended design.

Figure 7.5: Logic analyser capture of the ADC conversion timing sequence

7.2.2. Results

The ADC did not work correctly when the internal clock of the SPI interface was set to 80 MHz (and thus also the serial clock). There are unfortunately no recorded results for the interface at 80 MHz. A correct output waveform was successfully obtained when the ADC was operated with a lower clock frequency of 8 MHz. Figure 7.6 shows the captured waveform with no pressure applied to the sensor. The measured signal exhibits the expected sinusoidal behaviour, indicating that the ADC configuration, conversion timing and data acquisition process were functioning correctly at this operating frequency.

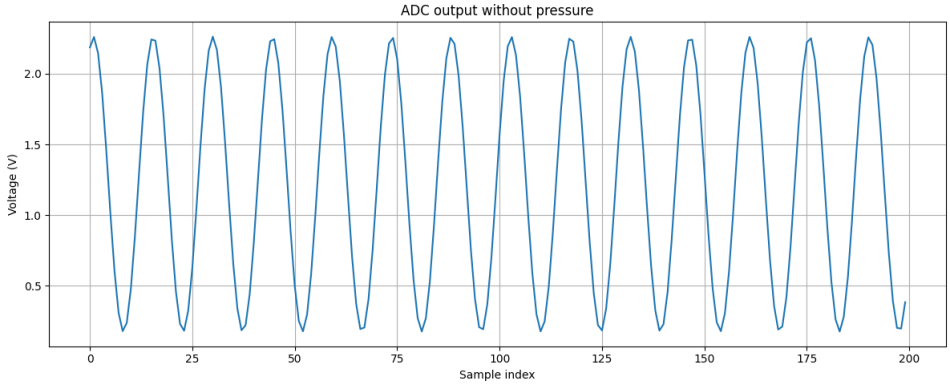


Figure 7.6: ADC output acquired at an SPI clock frequency of 8 MHz

To investigate the failure mechanism, the SPI clock frequency was also reduced to 50 MHz. The resulting waveform is shown in Figure 7.7. Although some structure can be observed in the captured data, the signal does not resemble the expected periodic waveform obtained at 8 MHz. In addition, noticeable variations and irregular transitions are present throughout the dataset, indicating that the acquired samples are not being reconstructed correctly.

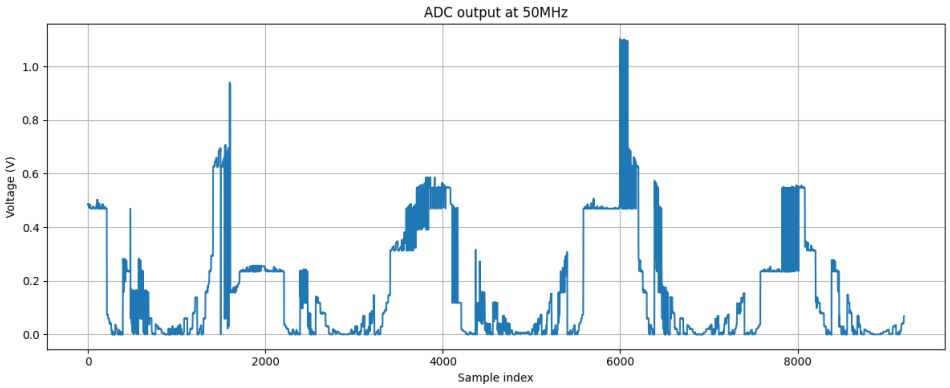


Figure 7.7: ADC output acquired at an SPI clock frequency of 50 MHz

The most likely cause of this behaviour is a timing mismatch within the ADC interface. During hardware bring-up, several bodge-wire modifications were required, introducing additional propagation delays on the control and data signals between the FPGA and ADC. The finite state machine (FSM) currently assumes ideal timing and does not compensate for these additional delays. Consequently, the ADC output bits may not have settled at the FPGA input when they are sampled by the FPGA, resulting in incorrect data capture. The distorted waveform observed at 50 MHz, and the complete failure at 80 MHz, are therefore believed to be caused by a mismatch between the arrival time of the ADC data and the sampling instant generated by the FSM.

7.3. Future Work

The current implementation samples the ADC output using a fixed timing relationship between the FPGA clock and the SPI data stream. Future work could compensate for propagation delays between the ADC and FPGA by adjusting the sampling instant relative to the incoming serial data. This could be achieved through the introduction of programmable delay elements or by experimentally tuning the sampling point. Such an approach would increase the available timing margin and improve robustness to variations in PCB routing and external interconnect delays.

Another possible improvement would be to configure the AD4630 for multi-lane SPI operation, as described in the datasheet [10]. By distributing the conversion result across multiple data lanes, the required SPI clock frequency could be reduced while maintaining the same throughput. Since the existing design already separates the conversion timing and SPI transfer logic, such a modification could potentially be implemented with relatively minor changes to the acquisition controller. However, this approach increases the number of FPGA I/O pins required per ADC channel. As a result, scaling the system to the project target of up to 16 channels would become more challenging and could impose additional constraints on FPGA selection and PCB routing.

A more substantial redesign could make use of the AD4630 echo-clock mode [10]. In this mode, the ADC outputs a clock signal alongside the conversion data, allowing the FPGA to sample the incoming bits using a clock generated by the ADC itself rather than a locally generated timing reference. This approach would significantly reduce sensitivity to timing variations in the data path and simplify the sampling process, as the data would be sampled using a clock that has traversed the same signal path as the ADC output. However, it would require a different FPGA interface architecture than the one presented in this work.

Finally, a synchronization handshake between the DAC and ADC paths could be implemented. The ADC requires a configuration sequence before it is ready to acquire, while the DAC can start immediately. An internal ready signal from the ADC controller, asserted once configuration is complete, could gate the start of DAC transmissions. This requires no hardware changes, only a small addition to the FPGA logic.

8

Conclusion

The project resulted in a DAQ prototype that was designed to satisfy the full set of non-functional requirements, but only a subset of them could be verified experimentally within the available project time. In particular, the system was tested and demonstrated to meet the requirements for data handling, number of channels, channel scalability, and pressure rating, corresponding to R5, R6, R7, and C3, in the chapter 2.

The data handling requirement, R5, was validated by the implemented system architecture, which is capable of streaming data or providing a one-second buffer. The design was structured in a way that remains scalable toward 16 channels. In addition, the complete system was tested successfully at the required pressure level of 500 bar, confirming compliance with C3.

Although the full specification set was not experimentally verified, the remaining requirements were met by design. This includes the sample rate, dynamic range, bandwidth, THD performance, and user interface requirements. These were addressed through component selection, circuit architecture, and the overall system implementation, but they could not all be fully validated under test conditions within the project scope.

The main reason for limiting the verification scope was the development-time constraint, C2, which had the highest priority in practice and was enforced by TU Delft. Given the available 10 weeks, the project had to focus on the most critical requirements and on getting a working prototype completed in time. As a result, the validation effort was concentrated on the requirements that were most essential to demonstrate system functionality and pressure survivability.

Overall, the project achieved its main goal: a working prototype was built, key requirements were demonstrated, and the design approach proved feasible for further development. With more time, the remaining design targets could be tested and refined, but the results already show that the proposed architecture is a solid basis for a more complete future implementation.

A

Ethical implications

This appendix provides an ethical analysis about the data acquisition system discussed in this report.

Societal impacts (product)
If fully implemented this data acquisition system could be used for research purposes and have a positive impact in that area because it allows for high precision gathering of data in high pressure environments where it would otherwise be hard to gather that data or at such high precision. the main stakeholders affected here would be researchers working with high pressure environments like sub-sea who would want to gather data about fish populations or the bottom of the sea, or environments in space, mainly on planets where gravity is higher than on earth. The data acquisition system could in these cases give a positive impact. Other stakeholders involved could be people developing spyware or developing weapons, where a high precision data acquisition system could have negative impacts. The system could be used in high pressure environments like sub-sea to build targeting systems for torpedos or other missiles in normal environments since they would need a high precision system like this one. It could be used in spyware to illegally gather data if the system falls into the wrong hands which would have a negative impact.

Ethical aspects of development (process)
To make the system, components and substances like solder and cleaning materials were used. These materials contain substances which are bad for human health and the environment. These products were therefore handled with care and ventilation systems with filters were used so that these substances could not harm the environment or the health of the ones building the PCBs of the data acquisition system.

Ethical evaluation
The most significant ethical concern raised by the product is its potential for dual use. While the system was designed for scientific research, the same precision and environmental robustness that make it valuable for subsea data collection could be exploited for weapons development or surveillance. Therefore the development of a product like this could have negative ethical impact. The fact that the system is a concept demonstrator rather than a finished product offers some mitigation, but does not eliminate the concern that technologies at early stages can still be used for unethical purposes. Publishing the design of the high precision data acquistition system is ethically valuable if used for research, supporting others with their scientific research, but it also enables misuse by parties with malicious intent. This is a trade-off between openness and security.

On the process side, the handling of hazardous materials such as solder and chemical cleaning agents represents a real, but manageable, ethical concern. The people most directly affected are the engineers building the system, whose health could be harmed by improper exposure. The use of ventilated workspaces and filtered extraction systems is ethically praiseworthy: the harm was anticipated, and concrete precautions were taken rather than ignored. The environmental impact of these substances, if improperly disposed of, would extend beyond the lab, affecting third parties who have no stake in the

project and no ability to consent to that risk. Proper disposal therefore carries a moral ethical weight beyond the legal obligations.

Reflection on design choices

Ethical considerations had a limited influence on the engineering process. The most direct example was the handling of environmentally harmfull materials during PCB assembly, the decision to use ventilated workstations with filtered extraction was a deliberate choice to protect both the people involved and the surrounding environment, rather than treating safety precautions as optional. Foreseeable harms to those involved in a project should be prevented when possible. However, the dual-use potential of the system was not explicitly addressed during the design phase. The focus remained on optimising technical performance such as precision, dynamic range, and high pressure environment suitability. The steps that could have been considered more to better address the ethical issues are things like deciding in advance which technical details to keep internal and which to publish openly.

Forward-looking responsibility

If developed into a real-world application, access to the system and its full design documentation should be restricted to research institutions, with clear contractual limits on permitted use. An export control review would be advisable given the subsea and high-pressure capabilities. On the process side, any scaling of production should comply with european RoHS regulations to minimise hazardous material use, and proper disposal procedures for chemical waste should be formalised. A brief ethics review at each major development milestone would help ensure that dual-use risks are reassessed as the system's capabilities grow.

B

DAC and ADC Selection

B.1. DAC Candidate Comparison

Table B.1 lists the specifications of the DAC candidates evaluated during the selection process described in chapter **??**. Eight candidates were considered across a range of manufacturers and interface types. The key selection criteria were serial interface compatibility, sample rate, resolution, dynamic range, and THD performance.

The majority of high-speed candidates (MAX5888, AD9726, AD9783, AD9144, AD9747), use parallel or high-speed differential interfaces (LVDS, JESD204B) that are incompatible with the 16-wire bulkhead constraint. The AD3552R offers a serial QSPI interface at 33 MSps, but its THD degrades to −84 dB at 100 kHz, falling short of the −100 dB system requirement. The DAC8811 is the only candidate that combines a serial SPI interface, 16-bit resolution, and a THD of −105 dB, while fitting within the wire count constraint. It was therefore selected as the DAC for this design.

Table B.1: DAC Specifications referenced from `DAC specs.xlsx`

Specification	MAX5888	MAX5195	AD9726	AD9783	AD9144	AD9747	DAC8811	AD3552R
Dynamic Range	155dB/Hz	160dB/Hz	160dB/Hz	157dB/Hz	162dB/Hz	160dB/Hz	169.4dB/Hz	158 dB/Hz
THD and Noise	-107dB	–	-95dB	–	-80/-85dB	-80dB	-105 @ 1kHz	-105 dB @1 kHz / -84 dB @100 kHz
Footprint	10mm x 10mm	–	–	–	–	–	very low	5×5 mm
Sampling Rate	500MS/s	260MS/s	400MS/s	500MS/s	2.8GS/s	250MS/s	2MS/s	33 MUPS (fast) / 22 MUPS (precision)
Power Consumption	250mW	–	575mW	315mW	2W	340mW	–	~200 mW est.
Power Rails	3.3V	–	2.5V, 3.3V	1.8V, 3.3V	3.3V, 1.8V	3.3V	–	5V, 1.8V
No. Bits	16	14	16	16 (dual)	16	16	16	16
No. Channels	1	1	1 (4 after demux)	2 (I & Q)	4	2	1	2
Package	68 QFN-EP	–	80 TQF-EP	72 LFCSP	72-pin LGA	48-pin LQFP	–	20-pin LFCSP
Thermal Range	-40 to +85	-40 to +85	-40 to +85	-40 to +85	-40 to +85	-40 to +85	-40 to +125	-40 to +105
Evaluation Board	yes	–	yes	yes	yes	yes	yes	Yes
Stocking Options	–	–	–	–	–	–	–	–
Interface	–	–	LVDS	LVDS	JESD204B	LVCMOS	SPI	QSPI (1/2/4-wire / SDR/DDR)

B.2. ADC Candidate Comparison

	ADS1675	AD7760	AD4630	AD4081
Dynamic Range	103dB	100dB	107dB	105 (164 dB/Hz)
THD and Noise	-107dB	-105dB	-127dB	−103.7dB at 1MHz
Footprint	1.2cm x 1.2cm	1.2cm x 1.2cm	7mm x 7mm	5mm x 5mm
Sampling Rate	4MSPS	2.5MSPS	2MSPS	20MSPS
Power Consumption	575mW	1W	30mW	68.6mW
Power Rails	5V, 3.3V	2.5V, 3.3V, 5V	5V, 1.8V	3.3V
No. Bits	24bit	24bit	24bit	20bit
No, Channels	1ch	1ch	2ch	1ch
Package	64 TQFP	64 TQFP	64 BGA 0.8mm	49 BGA 0.65mm
Thermal Range	-40 to 85	-40 to 85	-40 to 125	-40 to 85
Evaluation Board	Not really	Yes	Yes!	Yes
Stocking Options	775 (DigiKey)	706 (DigiKey)	1600 (DigiKey)	954 (DigiKey)
Communication Interfaces and Bandwith	LVDS or SPI (96Mbps)	16bit Parallel (5Mbps)	SPI (96Mbps, 1 - 4 SPI data lanes)	LVDS, QSPI (200Mbits), SPI (50Mbits)
Link	Link	Link	Link	Link
Notes		Built-in input filters		

References

[1] Spectrum Instrumentation GmbH, *DN2.80x/81x – hybridNETBOX up to 125 MS/s: Digitizer and AWG*, Datasheet, April 23, 2026, Grosshansdorf, Germany, Apr. 2026.

[2] ScopeFun, *ScopeFun User Manual*, Version 2.1, Velika Nedelja, Slovenia.

[3] Dewesoft d.o.o., *SIRIUSi-XHS Technical Reference Manual*, Document version V24-2, June 10, 2024, Trbovlje, Slovenia, Jun. 2024.

[4] National Instruments, *USB-6451 and USB-6451 (OEM) User Manual*, June 17, 2026, Jun. 2026.

[5] Texas Instruments, *DAC8811 16-bit, serial input multiplying digital-to-analog converter data sheet*, Rev. D, SLAS411D, Feb. 2016. Accessed: Jun. 7, 2026. [Online]. Available: https://www.ti.com/lit/ds/symlink/dac8811.pdf.

[6] Analog Devices, *JESD204B survival guide*, Analog Devices, 2012. [Online]. Available: https://www.analog.com/media/en/technical-documentation/technical-articles/JESD204B-Survival-Guide.pdf.

[7] Y. Pan, "Understanding the SPI bus," Texas Instruments, Tech. Rep. SBOA621, 2025. [Online]. Available: https://www.ti.com/lit/pdf/sboa621.

[8] P. P. Chu, *FPGA Prototyping by VHDL Examples: Xilinx Spartan-3 Version*. John Wiley & Sons, 2008, ISBN: 978-0-470-18531-5.

[9] P. J. Ashenden, *The Designer's Guide to VHDL*, 3rd ed. Morgan Kaufmann, 2008, ISBN: 978-0-12-088785-9.

[10] Analog Devices, Inc., *AD4630-24/AD4632-24 Data Sheet*, https://www.analog.com/media/en/technical-documentation/data-sheets/ad4630-24_ad4632-24.pdf, Accessed: 2026-04-21, 2023.

[11] N. Da Dalt and A. Sheikholeslami, *Understanding Jitter and Phase Noise*. Cambridge University Press, 2018, ISBN: 978-1-107-18857-0.

[12] J. M. Rabaey, A. P. Chandrakasan, and B. Nikolić, *Digital Integrated Circuits: A Design Perspective*, 2nd ed. Prentice Hall, 2003, ISBN: 978-0-13-090996-1.

[13] *ECP5 and ECP5-5G Family Data Sheet*, Document no. FPGA-DS-02012, Version 3.4. Accessed: Jun. 17, 2026, Lattice Semiconductor, Oct. 2025. [Online]. Available: https://www.latticesemi.com/en/Products/FPGAandCPLD/ECP5.

[14] B. Brannon, "Sampled systems and the effects of clock phase noise and jitter," Analog Devices, Tech. Rep. AN-756, 2004. [Online]. Available: https://www.scribd.com/document/382910751/AN-756.

[15] D. Redmayne, E. Trelewicz, and A. Smith, "Understanding the effect of clock jitter on high speed ADCs," Analog Devices (Linear Technology), Tech. Rep. DN-1013, 2006. [Online]. Available: https://www.analog.com/media/en/reference-design-documentation/design-notes/dn1013f.pdf.

[16] A. Raptakis, C. Oustoglou, and P. P. Sotiriadis, "Laboratory jitter removal circuit for single-bit all-digital frequency synthesis," in *2017 Panhellenic Conference on Electronics and Telecommunications (PACET)*, 2017, pp. 1–4. DOI: 10.1109/PACET.2017.8259964.

[17] SiTime, *SiT8208 high performance automotive MEMS oscillator*, SiTime Corporation, 2022. [Online]. Available: https://www.sitime.com/support/resource-library/datasheets/sit8208-datasheet.

[18] Texas Instruments, *CDCLVC1104 1-to-4 LVCMOS clock fanout buffer*, Texas Instruments, 2016. [Online]. Available: https://www.ti.com/lit/ds/symlink/cdclvc1104.pdf.

[19] onsemi, *MC100EP31 dual d flip-flop with set and reset*, onsemi, 2023. [Online]. Available: https://www.onsemi.com/pdf/datasheet/mc100ep31-d.pdf.

[20] Renesas Electronics, *830S21I-01 data sheet*, Renesas Electronics, 2021. [Online]. Available: https://www.renesas.com/en/document/dst/830s21i-01-final-data-sheet.