

AUTHORIZATION FOR DIGITALIZATION, STORAGE AND DISSEMINATION IN THE NETWORK OF END-OF-DEGREE PROJECTS, MASTER PROJECTS, DISSERTATIONS OR BACHILLERATO REPORTS

1. Declaration of authorship and accreditation thereof.

The author Mr. /Ms. Ignacio Escudero Sarabia

HEREBY DECLARES that he/she owns the intellectual property rights regarding the piece of work:

Implementation of different types of controllers to reduce rotor blade vibrations - An experimental approach

that this is an original piece of work, and that he/she holds the status of author, in the sense granted by the Intellectual Property Law.

2. Subject matter and purpose of this assignment.

With the aim of disseminating the aforementioned piece of work as widely as possible using the University's Institutional Repository the author hereby **GRANTS** Comillas Pontifical University, on a royalty-free and non-exclusive basis, for the maximum legal term and with universal scope, the digitization, archiving, reproduction, distribution and public communication rights, including the right to make it electronically available, as described in the Intellectual Property Law. Transformation rights are assigned solely for the purposes described in a) of the following section.

3. Transfer and access terms

Without prejudice to the ownership of the work, which remains with its author, the transfer of rights covered by this license enables:

- a) Transform it in order to adapt it to any technology suitable for sharing it online, as well as including metadata to register the piece of work and include "watermarks" or any other security or protection system.
- b) Reproduce it in any digital medium in order to be included on an electronic database, including the right to reproduce and store the work on servers for the purposes of guaranteeing its security, maintaining it and preserving its format.
- c) Communicate it, by default, by means of an institutional open archive, which has open and cost-free online access.
- d) Any other way of access (restricted, embargoed, closed) shall be explicitly requested and requires that good cause be demonstrated.
- e) Assign these pieces of work a Creative Commons license by default.
- f) Assign these pieces of work a **HANDLE** (*persistent URL*). by default.

4. Copyright.

The author, as the owner of a piece of work, has the right to:

- a) Have his/her name clearly identified by the University as the author
- b) Communicate and publish the work in the version assigned and in other subsequent versions using any medium.
- c) Request that the work be withdrawn from the repository for just cause.
- d) Receive reliable communication of any claims third parties may make in relation to the work and, in particular, any claims relating to its intellectual property rights.

5. Duties of the author.

The author agrees to:

- a) Guarantee that the commitment undertaken by means of this official document does not infringe any third party rights, regardless of whether they relate to industrial or intellectual property or any other type.

- b) Guarantee that the content of the work does not infringe any third party honor, privacy or image rights.
- c) Take responsibility for all claims and liability, including compensation for any damages, which may be brought against the University by third parties who believe that their rights and interests have been infringed by the assignment.
- d) Take responsibility in the event that the institutions are found guilty of a rights infringement regarding the work subject to assignment.

6. Institutional Repository purposes and functioning.

The work shall be made available to the users so that they may use it in a fair and respectful way with regards to the copyright, according to the allowances given in the relevant legislation, and for study or research purposes, or any other legal use. With this aim in mind, the University undertakes the following duties and reserves the following powers:

- a) The University shall inform the archive users of the permitted uses; however, it shall not guarantee or take any responsibility for any other subsequent ways the work may be used by users, which are non-compliant with the legislation in force. Any subsequent use, beyond private copying, shall require the source to be cited and authorship to be recognized, as well as the guarantee not to use it to gain commercial profit or carry out any derivative works.
- b) The University shall not review the content of the works, which shall at all times fall under the exclusive responsibility of the author and it shall not be obligated to take part in lawsuits on behalf of the author in the event of any infringement of intellectual property rights deriving from storing and archiving the works. The author hereby waives any claim against the University due to any way the users may use the works that is not in keeping with the legislation in force.
- c) The University shall adopt the necessary measures to safeguard the work in the future.
- d) The University reserves the right to withdraw the work, after notifying the author, in sufficiently justified cases, or in the event of third party claims.

Madrid, on 25..... of February, 2019.....,

HEREBY ACCEPTS

Signed, Ignacio Escudero Sarabia.....

Reasons for requesting the restricted, closed or embargoed access to the work in the Institution's Repository

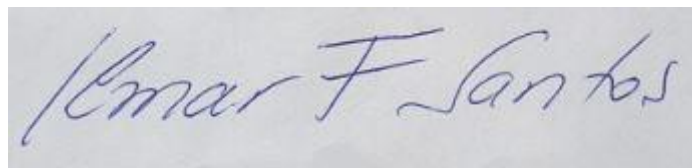
Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título
“Implementation of different types of controllers to reduce rotor blade
vibrations – An experimental approach” en la ETS de Ingeniería - ICAI de la
Universidad Pontificia Comillas en el
curso académico 2018/2019 es de mi autoría, original e inédito y
no ha sido presentado con anterioridad a otros efectos. El Proyecto no es
plagio de otro, ni total ni parcialmente y la información que ha sido tomada
de otros documentos está debidamente referenciada.

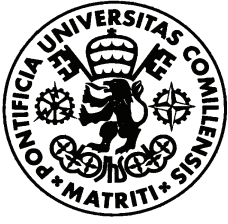
Fdo.: Fecha: 25/02/ 2019



Autorizada la entrega del proyecto
EL DIRECTOR DEL PROYECTO

Fdo.: Fecha: 25/ 02/ 2019





UNIVERSIDAD PONTIFICIA COMILLAS

**ESCUELA TÉCNICA SUPERIOR DE
INGENIERÍA (ICAI)**

MÁSTER EN INGENIERÍA INDUSTRIAL (MII)

PROYECTO FIN DE MÁSTER

**IMPLEMENTATION OF DIFFERENT TYPES OF
CONTROLLERS TO REDUCE ROTOR-BLADE
VIBRATIONS**

AN EXPERIMENTAL APPROACH

AUTHOR: IGNACIO ESCUDERO SARABIA

DIRECTOR: Ilmar Ferreira Santos

MADRID, 25/02/2019

IMPLEMENTACIÓN DE DIFERENTES TIPOS DE CONTROLES PARA REDUCIR LAS VIBRACIONES DE UN SISTEMA ROTATORIO

Autor: **Escudero Sarabia, Ignacio.**

Director: **Santos, Ilmar**

Entidad Colaboradora: **Technical University of Denmark - DTU**

RESUMEN DEL PROYECTO

Introducción

El objetivo principal de este proyecto es investigar experimentalmente la posibilidad de una compensación activa de las vibraciones paramétricas en sistemas rotatorios poco amortiguados mediante técnicas de control activo.

La pregunta principal a ser respondida en este proyecto se formula como:

¿Es posible reducir simultáneamente las vibraciones de las palas del rotor solo mediante el uso de actuadores instalados en el marco estático, o las vibraciones paramétricas de las palas deben compensarse con los actuadores montados directamente en el marco de referencia giratorio?

Este proyecto busca analizar la viabilidad de aumentar la seguridad operacional y reducir los costos de fallos de un sistema rotatorio reduciendo las vibraciones de las palas cuando el sistema está en operación. El control de las vibraciones de las palas puede ayudar a reducir la tensión adicional a la que están sujetos cuando el sistema está funcionando, mejorar su vida útil efectiva y reducir los fallos en las palas.

Tomando como ejemplo las turbinas de vapor, las vibraciones mecánicas de los álabes generalmente producen grandes esfuerzos en el material de las palas cuando vibran en condiciones de resonancia y, por lo tanto, generalmente es una causa frecuente de fallo de la turbina. El fallo de esta turbina puede incurrir en riesgos de seguridad, costes de mantenimiento adicionales y costes no operacionales. Las fuentes más comunes de estas excitaciones mecánicas son los campos de flujo no uniformes generados por el fluido operacional y las diferencias de presión entre las palas [1]. Las vibraciones causadas deben controlarse para garantizar un funcionamiento adecuado del sistema rotatorio.

Dado que es más complicado implementar actuadores montados en el marco giratorio, un tema de estudio muy importante es probar si existe la posibilidad de evitar estos actuadores y solo aplicar controles en el marco inercial, que se estudiará en este proyecto. La aplicabilidad de los resultados obtenidos en máquinas industriales reales dependerá en gran medida de la respuesta a esta pregunta.

Metodología

En este proyecto, el objetivo es el estudio experimental de la planta y el control de sus vibraciones sobre el banco de ensayos que modela el sistema de rotor-pala que ya está construido y es utilizado para este proyecto. Este banco de pruebas se ha estudiado anteriormente y se diseñaron algunos controladores activos [2] [3]. El estudio que se realizará se basará principalmente en una tesis anterior en la que se obtuvo el modelo matemático del banco de pruebas [4]. Este análisis también se basa en un trabajo contemporáneo sobre la construcción del banco de pruebas [5] y un análisis teórico desde el punto de vista del control [3].

Para la construcción del banco de pruebas, la estructura mecánica ya estaba construida como se describe a continuación. Fue construido en un trabajo anterior sobre el mismo proyecto, analizando las posibilidades de control y el estudio teórico de los sistemas estándar de rotor-pala. La única parte que se necesitaba para diseñar y construir era la electrónica para medir y actuar. El hardware instalado anteriormente no era suficiente para medir y controlar las vibraciones del rotor y las 4 cuchillas disponibles.

El diseño y construcción se realizó junto con otros trabajos de proyecto y también se incluye información sobre la instalación y el diseño de este banco de pruebas en [5]. En el ámbito de esta tesis específica, solo se diseñó e implementó el hardware para las mediciones y la actuación, todo el software se desarrolló en [5].

Un diagrama de flujo de señal del banco de pruebas en el que se basa este proyecto se muestra a continuación en la Figura 1, y muestra todas las partes pertinentes del equipo de pruebas:

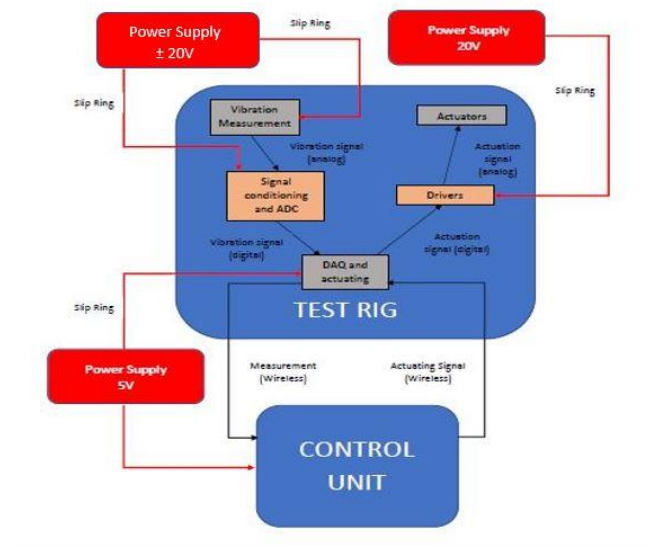


Figura 1: Diagrama de flujo de señal del banco de pruebas

Los actuadores son electroimanes instalados en cada lado de cada pala y en los lados de la torre del rotor en ambas direcciones de movimiento (x e y). La vibración de todo el sistema de palas del rotor se mide con sensores de medición de distancia también colocados en el lado de cada pala y la torre. Un sistema inalámbrico está disponible para controlar la vibración de las palas y la torre del rotor en función de las medidas del

sensor y la activación con los imanes en consecuencia. Los actuadores necesitan un controlador externo para ser controlados, ya que los electroimanes son impulsados por la corriente, y es más fácil medir el voltaje. Los sensores también necesitan un circuito de acondicionamiento externo para que la unidad DAQ lea sus señales.

La unidad de control se comunica a través de Wi-Fi con las unidades de adquisición de datos. Recibe las mediciones del DAQ, calcula la actuación deseada (bucle abierto, señal de control) y envía la señal de activación al sistema DAQ, que luego actúa en consecuencia. La unidad de control consistirá en el ordenador disponible que ejecuta el código desarrollado en [5].

A continuación se muestra un modelo mecánico teórico del banco de pruebas en la Figura 2:

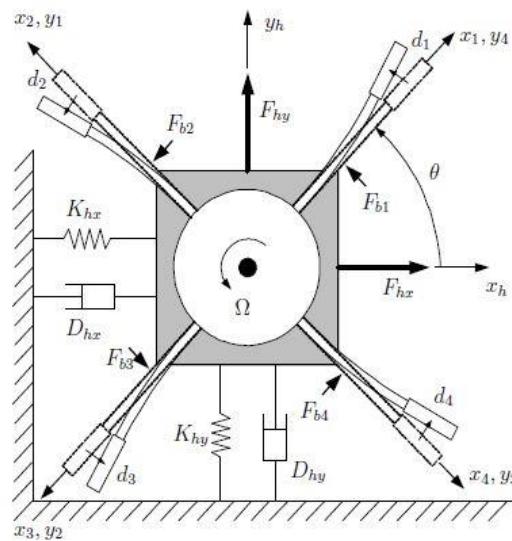


Figura 2: Modelo mecánico del banco de ensayos

Primero, una identificación experimental de la planta a controlar se realizará en condiciones estáticas y en condiciones de rotación y se comparará con el modelo teórico supuesto, seguido del diseño de diferentes tipos de controladores y estrategias de control que luego se implementarán y probarán. Los controladores elegidos son control PID, control de retroalimentación de estado completo y control adaptativo.

Resultados

El resultado de la identificación es pertinente en el caso de la operación de rotación, ya que se verificará la variabilidad con el tiempo de los parámetros de la planta. Una vez realizada la identificación, se comprueba su similitud con el modelo teórico, ya que los parámetros varían con el tiempo de forma lineal. La variabilidad de los polos

identificados del sistema se muestra en la Figura 3 a continuación, donde solo se calcula un período, mostrando la dependencia lineal con el tiempo.

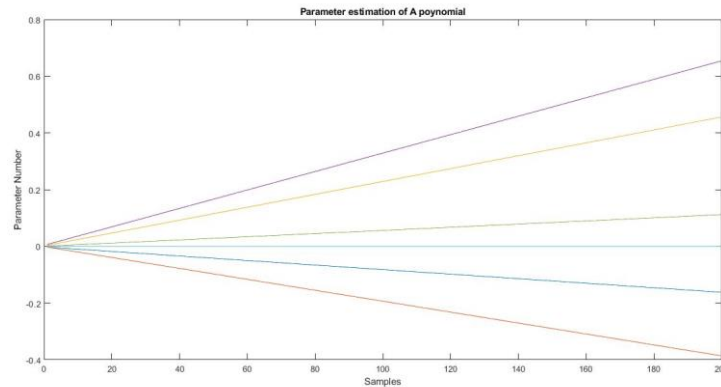


Figura 3: Parámetros estimados de los polos del sistema variantes con el tiempo

En cuanto a los resultados de los controladores PID, se observa en los gráficos que los controladores PID son ineficientes incluso para el caso estático, porque el sistema tiene un acoplamiento fuerte entre los canales de entrada y salida cruzados y las diferentes plantas no pueden considerarse desacopladas.

Para el controlador de realimentación de estado se diseña e implementa un controlador LQR. Los resultados muestran que es capaz de controlar el sistema en funcionamiento estático cuando se ejerce una perturbación externa, pero es ineficiente para reducir las vibraciones del sistema en rotación del sistema.

Para el controlador adaptativo se vio que cuando la planta está discretizada sus polos están realmente cerca de la inestabilidad, debido a los parámetros elegidos de la planta utilizada. Sin embargo, cuando esos parámetros cambian en el tiempo la planta se vuelve inestable en algunos puntos y el controlador adaptativo diseñado no pudo calcular una señal de control estable en esos puntos. Sin embargo, se demuestra que si se modifican los parámetros del equipo de prueba para que la planta se encuentre en la región estable, el controlador adaptativo es la mejor opción entre los controladores probados para amortiguar las vibraciones de la planta en condiciones de rotación. El cambio en las dinámicas del sistema determinará si el sistema puede controlarse en la situación de desacoplamiento, solo controlando las vibraciones del rotor, ya que la potencia de control que pueden emitir los imanes es limitada.

Conclusiones

Las conclusiones para la identificación del sistema fueron la evidencia experimental del análisis teórico. El sistema, cuando está en funcionamiento estático, se comporta como un sistema estándar de segundo orden amortiguado. Sin embargo, cuando el sistema está girando, sus parámetros se convierten en variantes con el tiempo en una tendencia periódica.

Después de la identificación de la planta, varios controladores fueron diseñados para suprimir las vibraciones causadas por la rotación del banco de pruebas. Los controladores elegidos para operación estática fueron controladores PID para cada sistema SISO no acoplado y un control por realimentación de estado con diseño LQR. Los resultados para los controladores PID fueron que el acoplamiento entre los canales de entrada / salida cruzados es demasiado grande como para suponer que es insignificante. En cuanto a los controladores de realimentación de estado LQR, podrían amortiguar las vibraciones en funcionamiento estático, pero cuando el sistema está en rotación no se pudo controlar las vibraciones, para el sistema acoplado o no acoplado.

Por último, para el controlador adaptativo, al realizar la identificación del sistema en tiempo real, se observó que el sistema está realmente cerca de la inestabilidad, en marcos continuos y discretos, debido a la dinámica agregada de los electroimanes y los parámetros inestables de esta planta específica. Esto hizo que el sistema se volviera inestable cuando los parámetros varían con el tiempo en la operación de rotación y, por lo tanto, el sistema no pudo ser controlado. Se hizo un cambio en los parámetros de la planta para forzarla en la región estable, y el controlador adaptativo era capaz de controlar las vibraciones del sistema en la operación de rotación.

La conclusión principal es que el sistema puede controlarse controlando solo las vibraciones del rotor, pero se logra una mejor actuación del control si se controlan tanto el rotor como las palas. La técnica de control necesaria para amortiguar las vibraciones debe ser una técnica compleja, la realimentación de estado estándar y los controladores PID no pueden aplicarse cuando el sistema está girando debido a la variabilidad temporal de la planta. Una técnica sugerida y probada es la técnica de control adaptativo, capaz de controlar las vibraciones de las palas.

Referencias

- [1] A. Sarkar R.S. Mohan and A.S. Sekhar. "Vibration analysis of a steam turbine blade". International Congress on Noise Control Engineering, 2014.
- [2] Juan Camino and Ilmar Santos. "A periodic h2 state feedback controller for a rotor-blade system". In "A periodic H2 state feedback controller for a rotor-blade system", 09/2018.
- [3] Maria Beneyto Gomez-Polo. "Classical control design theory applied to mitigate rotor-blade vibrations – a numerical investigation". In "Classical Control Design Theory Applied to Mitigate Rotor-Blade Vibrations – A Numerical Investigation", 02/2019.
- [4] René H. Christensen and Ilmar Santos. "A study of active rotor-blade vibration control using electro-magnetic actuation: Part 1 — theory". In "A Study of Active Rotor-Blade Vibration Control Using Electro-Magnetic Actuation: Part 1 — Theory", volume 6, 01/2004.
- [5] Alvaro Brandez Gorriz. "Design, implementation, and testing of hardware for sensing and controlling the dynamics of rotor-bladed systems". In "Design, Implementation and Testing of Hardware for Sensing and Controlling the Dynamics of Rotor-Blades Systems", 02/2019.

IMPLEMENTATION OF DIFFERENT TYPES OF CONTROLLERS TO REDUCE ROTOR-BLADE VIBRATIONS

Author: **Escudero Sarabia, Ignacio.**

Director: **Santos, Ilmar**

Collaborating Entity: **Technical University of Denmark - DTU**

SUMMARY OF THE PROJECT

Introduction

The main aim of this project is to investigate experimentally the possibility of active compensation of parametric vibrations in low dampened rotor-blade systems by means of active control techniques.

The major question to be answered in this project is formulated as:

Is it possible to simultaneously reduce rotor-blade vibrations alone by using actuators attached to the inertial frame, or do the parametric blade vibrations have to be compensated using actuators directly mounted in the rotating reference frame?

This project seeks to analyze the viability of increasing the operational safety and reduce the failure costs of a rotor-blade system by reducing the vibrations of the blades when the system is in operation. Controlling the vibrations of the blades can help reduce the additional stress they are subject to when the system is operating, improve its effective lifetime and reduce blade failure.

Taking as an example the steam turbines, the mechanical vibrations of the blades commonly results in big stresses to the blade material when vibrating in resonance conditions, and therefore it is usually a frequent cause of turbine failure. This turbine failures may incur in safety risks, additional maintenance costs and non-operational costs. The most common sources for these mechanical excitations are the non-uniform flow fields generated by the operational fluid and pressure differences between blades [1] and the vibrations caused need to be controlled to ensure a proper performance of the rotor system.

Since it is more complicated to implement actuators mounted in the rotating frame, a very important study topic is to test if there is a possibility to avoid these actuators and only apply controls in the inertial frame, which will be studied in this project. The applicability of the obtained results in real industrial machines will rely heavily on the answer to this question.

Methodology

In this project the experimental study of the plant and control of its vibrations is targeted, being the test rig to model the rotor-blade system already designed and used for this project. This test rig has been studied before and some active controllers were designed [2] [3]. The study to be performed will be mainly based on a previous thesis where the mathematical model of the test rig has been obtained [4]. This analysis is also based on some contemporary work on building the test rig [5] and a theoretical analysis from the control point of view [3].

For the building of the test rig, the mechanical structure was already built as described below. It was constructed in some previous work about the same project, analyzing possibilities of control and theoretical study of standard rotor-blade systems. The only part that was needed to be designed and built were the electronics for measuring and actuating. The previous hardware installed was not sufficient to measure and control the vibrations of the rotor and the 4 blades available.

This design and construction was done along with other project work and some documentation about the installation and design of this test rig is also included in [5]. In the scope on this specific thesis only the hardware for the measurements and actuation was designed and implemented, all the software was developed in [5].

A signal flow diagram of the test rig on which this project is based is shown below in Figure 1, and it shown all the pertinent parts of the test rig:

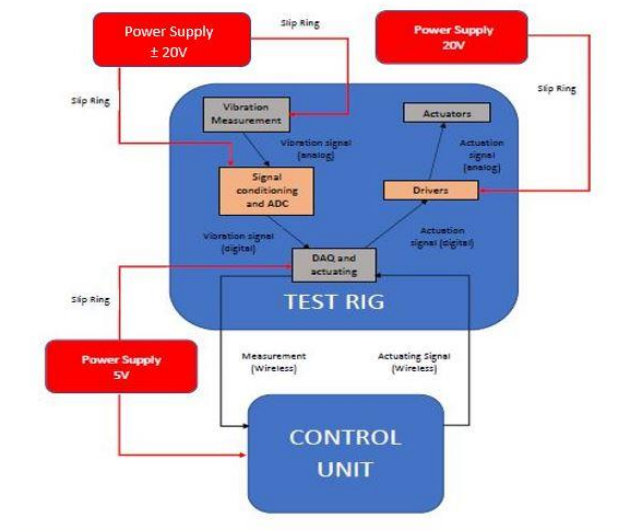


Figure 1: Signal flow diagram of the test rig

The actuators are electromagnets installed on each side of each blade and on the sides of the rotor tower in both movement directions (x and y). The vibration of the whole rotor-blade system is measured with distance measurement sensors also positioned on the side of each blade and the tower. A wireless system is available to control the vibration of the blades and the rotor tower based on the sensor's measurements and actuating with

the magnets accordingly. The actuators need an external driver to be controlled, since the electromagnets are driven by current, and it is easier to measure voltage. The sensors also need conditioning for the DAQ unit to read their signals.

The control unit communicates via Wi-Fi with the data acquisition units. It receives the measurements from the DAQ, calculates the desired actuation (open-loop, control signal,) and sends the actuation signal back to the DAQ system, which then acts accordingly. The control unit will consist on the available computer running the code developed in [5].

A theoretical mechanical model of the test rig is shown below in Figure 2:

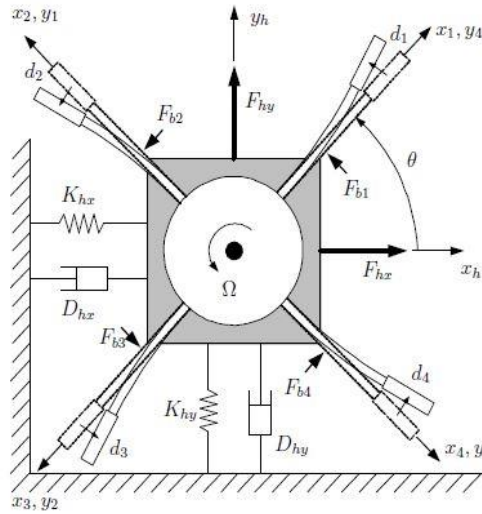


Figure 2: Mechanical modelling of the test rig

First, an experimental identification of the plant to control will be performed in static conditions and in rotating conditions and compared with the assumed theoretical model, followed by the design of different types of controllers and control strategies which will then be implemented and tested. The controllers chosen are PID control, full state feedback control and adaptive control.

Results

The result of the identification is pertinent in the rotating operation case, since the parameter time variability will be checked. After the identification is performed, the theoretical model is matched, since the parameters vary with time in a linear way. The variability of the identified poles of the system is shown in Figure 3 below, where only one period is calculated, showing the linear dependence with time.

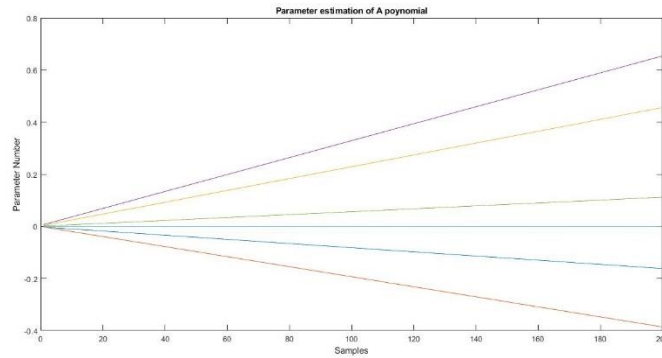


Figure 3: Parameter estimation of the poles of the system in rotating operation

As for the results in the PID controllers, it is seen in the signal plots that the PID controllers are inefficient even for the static case, because the system has hard coupling between crossed input and output channels and the different plants cannot be considered separated.

For the full state feedback controller, an LQR controller is designed and implemented. The results show that it is able to control the system in static operation when it is subject to an external disturbance, but it is inefficient in reducing the vibrations of the system in rotating operation.

For the adaptive controller, it was seen that when the plant is discretized, its poles are really close to instability, due to the chosen parameters of the plant used. However, when those parameters change in time, the plant becomes unstable at some points, and the designed adaptive controller could not calculate a stable controls signal. However, it is proved that if the test rig's parameters are modified so the plant is in the stable region the adaptive controller is the best way among the tried controllers to dampen the vibrations of the plant in rotating conditions. The change in the system's dynamics will determine if the system can be controlled in the uncoupled situation, only controlling the hub's vibrations, since the control power the magnets can output is limited.

Conclusions

The conclusions for the system identification was the experimental evidence of the theoretical analysis. The system, when in static operation, behaves as a standard dampened second order system. However, when the system is rotating, its parameters become time variant in a periodic trend.

After the identification of the plant, several controllers were designed to suppress the vibrations caused by the rotation of the test rig. The controllers chosen for static operation were PID controllers for every uncoupled SISO system and a full state feedback with LQR design controller. The results for the PID controllers were that the coupling between crossed input/output channels is too big to be assumed negligible. As for the LQR full state feedback controllers, they could dampen the vibrations in static

operation, but when the system was rotating it was unable to control the measurements, for coupled or uncoupled system.

Last, for the adaptive controller, when performing the online system identification it was seen that the system is really close to instability, in continuous and discrete frameworks, due to the added dynamics of the electromagnets and the close to unstable parameters of this specific plant. This caused the system to become unstable when the parameters vary with time in rotating operation, and therefore the system could not be controlled. A change in the plant's parameters was made to force it into the stable region, and the adaptive controller was capable of controlling the vibrations of the system in rotating operation.

The main conclusion is that the system can be controlled by controlling only the vibrations of the rotor, but a better control performance is achieved if both rotor and blades are controlled. The control technique needed to dampen the vibrations must be a complex technique, standard full state feedback and PID controllers cannot succeed when the system is rotating due to the time variability of the plant. One technique suggested and tested is the adaptive control technique, capable of controlling the vibrations of the blades.

References

- [1] A. Sarkar R.S. Mohan and A.S. Sekhar. "Vibration analysis of a steam turbine blade". International Congress on Noise Control Engineering, 2014.
- [2] Juan Camino and Ilmar Santos. "A periodic h2 state feedback controller for a rotor-blade system". In "A periodic H2 state feedback controller for a rotor-blade system", 09/2018.
- [3] Maria Beneyto Gomez-Polo. "Classical control design theory applied to mitigate rotor-blade vibrations – a numerical investigation". In "Classical Control Design Theory Applied to Mitigate Rotor-Blade Vibrations – A Numerical Investigation", 02/2019.
- [4] René H. Christensen and Ilmar Santos. "A study of active rotor-blade vibration control using electro-magnetic actuation: Part 1 — theory". In "A Study of Active Rotor-Blade Vibration Control Using Electro-Magnetic Actuation: Part 1 — Theory", volume 6, 01/2004.
- [5] Alvaro Brandez Gorriz. "Design, implementation, and testing of hardware for sensing and controlling the dynamics of rotor-bladed systems". In "Design, Implementation and Testing of Hardware for Sensing and Controlling the Dynamics of Rotor-Blades Systems", 02/2019.

ABSTRACT

One of the main causes of failure of rotor-blade systems is blade deterioration. When the system rotates, the blades start to vibrate causing them to fail eventually. Controlling those vibrations can help reduce the additional stress they are subject to, improve its effective lifetime and reduce the failure costs of maintenance and repair. This thesis seeks to analyze the effect of active vibration control on standard rotor-blade systems. However, the installation of sensors and actuators directly on the blades can be difficult and expensive, depending on the specific application. The main study focus on this project is to test if active vibration control can be applied only to the rotor's vibrations or it is necessary to install sensors and actuators on the blades as well.

For this analysis a test rig of a rotor-blade system has been studied. This test rig in particular measures the vibrations with eddy current sensors and actuates in the rotor and in the blades with electromagnets. First thing is analyzing the system theoretically and the built test rig with its components. After that, an experimental open loop identification is performed, recording input/output data from the test rig and identifying the parameters of the system to compare with the theoretical analysis and check for non modeled dynamics. After the experimental system identification, an analysis of the plant revealed that when the system is in rotating operation the plant's parameters become periodic time-variant, matching the theoretical description. With the identified system, some standard controllers are designed, but their performance fails under time-varying conditions of the plant. That is why an adaptive controller is designed and implemented, resulting in a control that can dampen the vibrations of the system in rotating conditions. The adaptive controller technique works for both coupled (actuating on blades and rotor) or uncoupled (actuating only in the rotor) situations, with more controller effort on the uncoupled situation.

PREFACE

This thesis is carried out as partial fulfillment of the Master Degree on Industrial Engineering at the Universidad Pontificia de Comillas (ICAI), and it is focused on the system analysis, stochastic identification and control study lines. The thesis is written alongside with Alvaro Brandez Gorriz and Maria Beneyto Gomez-Polo's thesis, and all three projects are part of one big study on the same test rig. Some previous work was performed in the same project by René Hardam Christensen, and this project is based on the work of that thesis, as well as in the work of my colleagues. The supervisor of the project is professor Ilmar Ferreira Santos, who also helped on the building and analysis of the test rig.

I would like to thank my supervisor for his guidance through this project, and for the opportunity to work alongside him. I would also like to thank Maria Beneyto and Alvaro Brandez for sharing the workload during this project.

Universidad Pontificia de Comillas ICAI

Madrid, February 2019

Ignacio Escudero Sarabia

TABLE OF CONTENTS

Abstract	x i
Preface	x ii
Table of Contents	x iii
List of Figures	x v
1 Introduction	1
2 State of the Art	2
3 Project Objectives	3
4 Chapter 1: Physical Description of the Test Rig	5
4.1 Description of the system	5
4.2 Theoretical model of the system	11
4.3 Simulation model	14
5 Chapter 2: Experimental Identification of the System	17
5.1 Experimental data design	19
5.2 Model structure for identification	28
5.3 Model identification of the plant	29
5.4 Model validation	37
5.5 Conclusion of the identification	41
6 Chapter 3: Control Designs for the System	43
6.1 PID controller design	44
6.2 Full state feedback controller design	49
6.3 Adaptive controller design	52
7 Chapter 4: Control Implementation and Results	59
7.1 PID controller implementation	59
7.2 Full state feedback controller implementation	69
7.3 Adaptive controller implementation	80
8 Conclusions and Future Implementations	85
8.1 Future Improvements	85
References	86
A Appendix A: Experimental Identification MatLab Code	88
A.1 Chopping and pre-filtering data	88
A.2 Time variable estimation	119

B	Appendix B: MatLab Code for Control Design	121
B.1	PID controllers design	121
B.2	LQR controller design	124
B.3	Online identification design	126
C	Appendix C: MatLab Code for the Implementation of Controllers	128
C.1	PID controllers implementation	128
C.2	LQR controllers implementation	129
C.3	Adaptive controller implementation	129
D	Appendix D: Hardware Components Data Sheets	133

LIST OF FIGURES

4.1	Picture taken of the used test rig	6
4.2	Flow diagram of the test rig designed	7
4.3	Signal conditioning circuit for each sensor	8
4.4	PCB layout for the conditioning circuit	9
4.5	Control loop of the built system	11
4.6	Mechanical model of the test rig	11
4.7	Geometric and physical parameters of the test rig	12
4.8	Block diagram of the mechanical plant	14
4.9	Block diagram of the magnets dynamics	15
4.10	Block diagram of the open loop plant	15
5.1	Flow scheme of the identification procedure	19
5.2	dSpace interface used for the measurement of all displacements	20
5.3	Output measurements with unexcited inputs and static operation ($\Omega = 0Hz$) .	23
5.4	Output measurements with unexcited inputs and rotating operation ($\Omega = 5Hz$)	23
5.5	Output measurements with impulse input on the stator in Y direction and static operation ($\Omega = 0Hz$)	24
5.6	Output measurements with impulse input on the stator in Y direction and rotating operation ($\Omega = 5Hz$)	24
5.7	Output measurements with step input on the stator in Y direction and static operation ($\Omega = 0Hz$)	25
5.8	Output measurements with step input on the stator in Y direction and rotating operation ($\Omega = 5Hz$)	25
5.9	Output measurements with chirp input on blade 1 and static operation ($\Omega = 0Hz$)	26
5.10	Output measurements with random inputs with different seeds on every channel and static operation ($\Omega = 0Hz$)	26
5.11	Output measurements with random inputs with different seeds on every channel and rotating operation ($\Omega = 5Hz$)	27
5.12	Output measurements with shifted square inputs in static operation ($\Omega = 0Hz$)	27
5.13	Output measurements with shifted square inputs in rotating operation ($\Omega = 5Hz$)	27
5.14	Layout of MatLab's System Identification Toolbox as it is used in this project .	30
5.15	SVD for static data for model order selection in control focus	31
5.16	SVD for rotating data for model order selection in control focus	31
5.17	SVD for static data for model order selection in prediction focus	32
5.18	SVD for rotating data for model order selection in prediction focus	33
5.19	Step response for the identified model in static operation	34
5.20	Output response for the identified model in rotating operation	36
5.21	Output step response on blade 1 and 99 % confidence interval for the response on static identified model	37
5.22	Output step response on blade 2 and 99 % confidence interval for the response on static identified model	38

5.23	Output step response on blade 3 and 99 % confidence interval for the response on static identified model	38
5.24	Output step response on blade 4 and 99 % confidence interval for the response on static identified model	39
5.25	Output step response on stator in direction x and 99 % confidence interval for the response on static identified model	39
5.26	Output step response on stator in direction y and 99 % confidence interval for the response on static identified model	40
5.27	Standard errors for each MISO system on every sample on rotating identified model	40
5.28	Comparison of step response of simulated model and estimated model	41
5.29	Parameter variation of the poles of the identified time variable system	41
6.1	Output response for the simulated model in rotating operation with null inputs applied	43
6.2	Disturbance applied to the control signal $u(t)$ for the design on the PID controllers	44
6.3	Disturbance applied to the output measurements $y(t)$ for the design on the PID controllers	45
6.4	Step responses of reduced and original systems for every SISO system	45
6.5	Layout of the PID Tuner Toolbox in MatLab	46
6.6	Comparison between closed and open loop response with designed PID for SISO system on blade 1	46
6.7	Comparison between closed and open loop response with designed PID for SISO system on blade 2	47
6.8	Comparison between closed and open loop response with designed PID for SISO system on blade 3	47
6.9	Comparison between closed and open loop response with designed PID for SISO system on blade 4	48
6.10	Comparison between closed and open loop response with designed PID for SISO system on stator in x direction	48
6.11	Comparison between closed and open loop response with designed PID for SISO system on stator in y direction	49
6.12	Block diagram of the implemented Luenberger observer	50
6.13	Full state feedback controller for coupled system	51
6.14	Full state feedback controller for uncoupled system	51
6.15	Block diagram of the basic structure of an adaptive control design [1]	52
6.16	Block diagram of the basic structure of an adaptive control design with the adaptation scheme displayed[1]	53
6.17	Block diagram of an indirect adaptive controller design[1]	54
6.18	Block diagram of the basic online estimator functioning [1]	54
6.19	Block diagram of the built online estimator with the simulated plant [1]	56
6.20	Prediction error of the online estimation of every MISO subsystem of the plant	57
6.21	Time variability of the second parameter in the A polynomial for the first MISO system estimated	57
6.22	Block diagram of the adjustable controller designed	58
7.1	Block diagram of the implementation of the PID controllers	59

7.2	Vibration measurement and control signal of blade 1 under PID controller with coupled system	60
7.3	Vibration measurement and control signal of blade 2 under PID controller with coupled system	61
7.4	Vibration measurement and control signal of blade 3 under PID controller with coupled system	61
7.5	Vibration measurement and control signal of blade 4 under PID controller with coupled system	62
7.6	Vibration measurement and control signal of stator in x direction under PID controller with coupled system	62
7.7	Vibration measurement and control signal of stator in y direction under PID controller with coupled system	63
7.8	Vibration measurement and control signal of blade 1 under PID controller with uncoupled system	63
7.9	Vibration measurement and control signal of blade 2 under PID controller with uncoupled system	64
7.10	Vibration measurement and control signal of blade 3 under PID controller with uncoupled system	64
7.11	Vibration measurement and control signal of blade 4 under PID controller with uncoupled system	65
7.12	Vibration measurement and control signal of stator in x direction under PID controller with uncoupled system	65
7.13	Vibration measurement and control signal of stator in y direction under PID controller with uncoupled system	66
7.14	Vibration measurement and control signal of blade 1 under PID controller with rotating system	66
7.15	Vibration measurement and control signal of blade 2 under PID controller with rotating system	67
7.16	Vibration measurement and control signal of blade 3 under PID controller with rotating system	67
7.17	Vibration measurement and control signal of blade 4 under PID controller with rotating system	68
7.18	Vibration measurement and control signal of stator in x direction under PID controller with rotating system	68
7.19	Vibration measurement and control signal of stator in y direction under PID controller with rotating system	69
7.20	Block diagram of the implementation of the full state feedback controllers . . .	70
7.21	Block diagram of the subsystem with controller and observer	70
7.22	Vibration measurement and control signal of blade 1 under LQR controller on coupled system	71
7.23	Vibration measurement and control signal of blade 2 under LQR controller on coupled system	71
7.24	Vibration measurement and control signal of blade 3 under LQR controller on coupled system	72
7.25	Vibration measurement and control signal of blade 4 under LQR controller on coupled system	72

7.26	Vibration measurement and control signal of stator in x direction under LQR controller on coupled system	73
7.27	Vibration measurement and control signal of stator in y direction under LQR controller on coupled system	73
7.28	Vibration measurement and control signal of blade 1 under LQR controller on uncoupled system	74
7.29	Vibration measurement and control signal of blade 2 under LQR controller on uncoupled system	74
7.30	Vibration measurement and control signal of blade 3 under LQR controller on uncoupled system	75
7.31	Vibration measurement and control signal of blade 4 under LQR controller on uncoupled system	75
7.32	Vibration measurement and control signal of stator in x direction under LQR controller on uncoupled system	76
7.33	Vibration measurement and control signal of stator in y direction under LQR controller on uncoupled system	76
7.34	Vibration measurement and control signal of blade 1 under LQR controller on rotating system	77
7.35	Vibration measurement and control signal of blade 2 under LQR controller on rotating system	77
7.36	Vibration measurement and control signal of blade 3 under LQR controller on rotating system	78
7.37	Vibration measurement and control signal of blade 4 under LQR controller on rotating system	78
7.38	Vibration measurement and control signal of stator in x direction under LQR controller on rotating system	79
7.39	Vibration measurement and control signal of stator in y direction under LQR controller on rotating system	79
7.40	Block diagram of the whole implementation of the adaptive controller	80
7.41	Block diagram of the online identification system implementation	81
7.42	Vibration measurement and control signal for stator in X direction with adaptive controller	81
7.43	Vibration measurement and control signal for stator in X direction with adaptive controller	82
7.44	Vibration measurement and control signal for stator in Y direction with adaptive controller	82
7.45	Vibration measurement and control signal for blade 1 with adaptive controller	83
7.46	Vibration measurement and control signal for blade 2 with adaptive controller	83
7.47	Vibration measurement and control signal for blade 3 with adaptive controller	84
7.48	Vibration measurement and control signal for blade 4 with adaptive controller	84

1 INTRODUCTION

The main aim of this project is to investigate experimentally the possibility of active compensation of parametric vibrations in low dampened rotor-blade systems by means of active control techniques.

The major question to be answered in this project is formulated as:

Is it possible to simultaneously reduce rotor-blade vibrations alone by using actuators attached to the inertial frame, or do the parametric blade vibrations have to be compensated using actuators directly mounted in the rotating reference frame?

This project seeks to analyze the viability of increasing the operational safety and reduce the failure costs of a rotor-blade system by reducing the vibrations of the blades when the system is in operation. Controlling the vibrations of the blades can help reduce the additional stress they are subject to when the system is operating, improve its effective lifetime and reduce blade failure.

Taking as an example the steam turbines, the mechanical vibrations of the blades commonly results in big stresses to the blade material when vibrating in resonance conditions, and therefore it is usually a frequent cause of turbine failure. This turbine failures may incur in safety risks, additional maintenance costs and non-operational costs. The most common sources for these mechanical excitations are the non-uniform flow fields generated by the operational fluid and pressure differences between blades [2], and the vibrations caused need to be controlled to ensure a proper performance of the rotor system.

Since it is more complicated to implement actuators mounted in the rotating frame, a very important study topic is to test if there is a possibility to avoid these actuators and only apply controls in the inertial frame, which will be studied in this project. The applicability of the obtained results in real industrial machines will rely heavily on the answer to this question.

For this approach the focus will be on any standard rotor-blade system, and the analysis can be applied on any kind of system with these conditions. However, the application and positioning of sensors and actuators to control the blade's vibrations will depend on the layout of the specific system, and can be limited to the proper boundaries of that explicit application.

2 STATE OF THE ART

Some previous attempts to reduce rotor-blade systems parametric vibrations have mechanical solutions, such as limiting the vibration amplitude of the blades or using a snubbing mechanism. This is achieved by limiting the blade tip vibration amplitude by leaving only small gaps between the shrouds of neighbouring blades. The main problem with this approach is that the heat produced by the friction between the blades needs to be dissipated, either with under platform dampers or friction rings. This results in a difficult installation of the extra components and limitation of the blade tip vibrations with snubbing mechanisms is only successful in reducing the blade's vibrations around the resonance frequencies of the blade [3].

Another approach to reduce the resonance vibration of a rotor-blades system is to use a passive control device such as a mass tuned damper. This device reduces the mechanical vibrations of the blades, but also needs to be taken into consideration when modeling the whole operating plant to analyze its vibrations along with the blades, making it difficult for some applications to be controlled properly [4].

A different method studied is to control the vibration of the blades using piezo electric blades and imposing a torque to counteract the vibration by generating an electric field. Some studies used the quantitative feedback theory to design a controller for the piezo electric blades, since the analysis of the vibrations showed non-linear behavior on turbine blades when operating. The study shows a successful minimization of the blades vibrations, but it requires a change in the material of the blade, and sometimes also a change in its shape, which cannot be possible for many rotor-blade systems due to their already optimized operation conditions or limited availability of feasible materials [5].

Several modal shape analysis have been done to obtain an accurate mathematical model of the rotor blade system and be able to study the control of the blade's vibrations in more depth. Application to helicopter blades derives immediately, since they have already built sensors measuring the blades deflections, and actuators to reduce their vibrations can be easily mounted. Some approaches use active twist control to achieve this dampening of the deflections [6] where the elastic coupling between blades is studied and controlled with piezoceramic materials.

3 PROJECT OBJECTIVES

In this project the experimental study of the plant and control of its vibrations is targeted, being the test rig to model the rotor-blade system already designed and used for this project. This test rig has been studied before and some active controllers were designed [7] [8]. The study to be performed will be mainly based on a previous thesis where the mathematical model of the test rig has been obtained [9]. This analysis is also based on some contemporary work on building the test rig [10] and a theoretical analysis from the control point of view [8]. For an actual rotor-blade system application the installation of the needed components as they are displayed in the test rig will have to be studied depending on the conditions of the system to be applied to.

The approach to reduce the vibration of the rotor-blades plant will use electromagnetic forces to counteract the amplitude of such vibrations. For this project the dependence of the blades vibrations with respect to the rotor vibrations will be analyzed, so the vibrations of the rotor will also be taken into consideration when it comes to modelling the vibrations of the blades, and controllers for both the blades and the rotor will be designed and tested to check such interdependence.

The electromagnets are installed on each side of each blade and on the sides of the rotor tower in both movement directions (x and y). The vibration of the whole rotor-blade system is measured with distance measurement sensors also positioned on the side of each blade and the tower. A wireless system is available to control the vibration of the blades and the rotor tower based on the sensor's measurements and actuating with the magnets accordingly. Further explanations on the project layout are shown in Chapter 1 below.

First, an experimental identification of the plant to control will be performed in static conditions and in rotating conditions and compared with the assumed theoretical model, followed by the design of different types of controllers and control strategies which will then be implemented and tested.

For this project the main objectives have been calculated according to the dimensions and physical limitations of the test rig built. For objective 1, the vibration of the blades and the tower is limited to 3 mm peak to peak vibration amplitude by the actuators on each side. For objective 2, the material of both the blades and the rotor tower has already been identified, its elasticity properties calculated, and the strength needed to reduce the possible maximum vibrations has been studied [9]. The rest of the objectives are explained as follows:

1. Measure the vibrations of blades and rotor tower with precision in the 3 mm vibration peak to peak amplitude of both blades and tower to distinguish its frequencies in static and rotating conditions.
2. Actuate on the blades and the rotor tower with enough strength to reduce the

3. Project Objectives

- measured vibrations in the 3 mm peak to peak vibrations for each actuator.
3. The data acquisition unit has to obtain the measurements and send them to the control center wirelessly due to having the sensors and actuators on a rotating frame.
 4. The actuation unit has to receive the actuation from the control center wirelessly and be able to actuate accordingly.
 5. Wireless connections have to be effective in only a 2-meter range, since the control unit does not need to be away from the test rig.
 6. Stochastic system identification will need to be accurate and validated to be considered similar enough to the real system.
 7. Any design will need to reduce the blade's vibrations at least one magnitude order to be considered a good control.

The objectives for the designed controllers are just to analyze the reduction of blade vibrations in both blades and rotor. If the controllers are successful, the control accuracy of actuating in both blades and rotor (coupled system) or only in the rotor (uncoupled system) will be compared.

CHAPTER 1: PHYSICAL DESCRIPTION OF

4 THE TEST RIG

4.1 Description of the system

For the building of the test rig, the mechanical structure was already built as described below. It was constructed in some previous work about the same project, analyzing possibilities of control and theoretical study of standard rotor-blade systems. The only part that was needed to be designed and built were the electronics for measuring and actuating. The previous hardware installed was not sufficient to measure and control the vibrations of the rotor and the 4 blades available.

This design and construction was done along with other project work and some documentation about the installation and design of this test rig is also included in [10]. In the scope on this specific thesis only the hardware for the measurements and actuation was designed and implemented, all the software was developed in [10].

The test rig consists on a static squared aluminum frame resting on the reference ground, which will be the static reference of the system, and it bears the rotor-blade system inside the frame, which from now on will be referred as the inertial frame, the rotor or the hub. The rotor is coupled to the static frame with a structure of thin beams that bear the hub with an already characterized total stiffness and damping. The rotor can move in horizontal and vertical directions inside the frame, with the boundaries of the elastic couplings that attach it to the static frame. Any other movement of the hub with respect to the static frame such as angular rotation is reduced by the structure of the hub and considered negligible. The sensors and actuators to measure the hub's horizontal and vertical vibration with respect to the static frame are mounted on the beam structure in a static reference.

The blades are radially attached to the shaft of the inertial frame on its tip. The sensors and actuators for the blade's vibrations with respect to the inertial frame are mounted on a solid rigid aluminum disc perpendicular to the shaft and rotating along with the rotor. Each blade has a mass attached to their tip end, assuming the whole mass of the blade is concentrated on the tip. The blades are mounted so that they can only vibrate perpendicular to the shaft. The sensor's measurements of the blade's vibrations are taken from the rotating reference.

The electronics for the blades components are mounted behind the perpendicular disc and rotate along with the inertial frame. The same electronics for the measurements and actuation performed in the hub are mounted on a separate box on the static frame. The layout of the test rig is shown below on Figure 4.1.

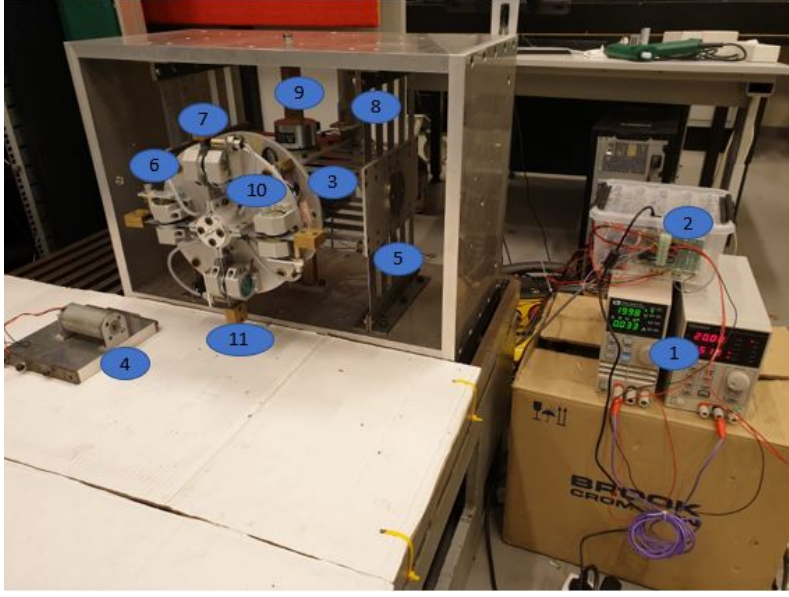


Figure 4.1. Picture taken of the used test rig

The elements included in the picture of the test rig are denoted with numbers as follows:

1. Power supply sources powering all the electronics.
2. Box containing the electronics for the hub.
3. Electronics of the inertial frame (behind the aluminum disc).
4. DC motor that will be coupled to the shaft of the test rig.
5. Structure of beams that support the rotor and both compose the hub.
6. Actuation magnets of one blade.
7. Measurement sensors of one blade.
8. Measurement sensor of the hub in one direction.
9. Actuation magnet of the hub in one direction.
10. Aluminum rigid disc perpendicular to the blades vibration.
11. Blade with tip mass and attached to the rotor.

The whole system ,which has been built and is operational, takes the measurements and actuates with the commands given in the C code. The code for sending and receiving data from the DAQ unit as well as controlling the test rig is developed and included in [10]. The functioning of the whole test rig is explained in the following flow diagram in Figure 4.2:

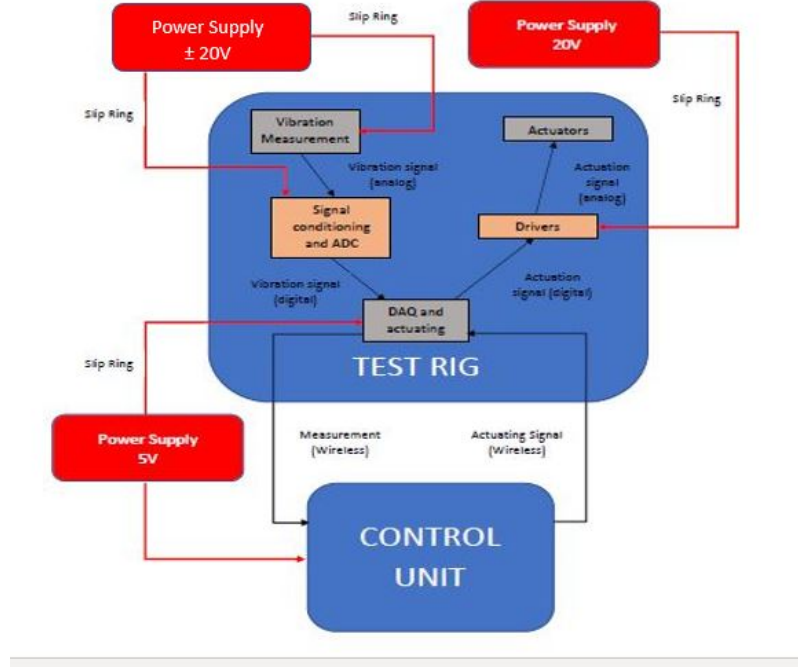


Figure 4.2. Flow diagram of the test rig designed

Inside the inertial frame the blade's actuators, drivers, measurement sensors, analog to digital converter and data acquisition unit are mounted and are capable of sending measurements, receiving actuation signals and acting accordingly. On the static frame, the rotor's sensors and actuation electronics are also mounted and are also capable of sending measurements and receiving actuation signals. For both the static and the inertial frame the data acquisition and actuation procedures are the same. These procedures are explained in depth in the following sections.

The control unit communicates via WiFi with the data acquisition units. It receives the measurements from the DAQ, calculates the desired actuation (open-loop, control signal,...) and sends the actuation signal back to the DAQ system, which then acts accordingly. The control unit will consist of the available computer running the code developed in [10]. The control unit is explained in more detail in the following sections.

As for the power supply, the powering needed is for the actuators, the sensors, the electronics and the DAQ units. For the static frame the electronics are powered directly from the available power supplies, giving $+20\text{ V}$, -20 V for the sensors and actuators and $+5\text{ V}$ for the DAQ unit. For the inertial frame the same power levels are needed, and they are supplied via the slip ring already built with the test rig. This slip ring allows sending any type of signal from a static source to the rotating inertial frame without direct cable connection. A slip ring is an electromechanical device that allows electrical signals to be transmitted from a static frame to a rotating frame with brush connections [11].

In the following sections the measurement sensors, the actuation system, the DAQ units and the control unit will be explained in more depth, followed by the theoretical model of the described system and the MatLab simulation model. The measurement and actuation

procedure is the same for the static and inertial frame, with the same electronics mounted on both and taking into account the different number of signals, sensors and actuators.

4.1.1 Measurement system

For the measurement sensors, the need is to measure the distance of the blades or the hub to a reference frame. Eddy current sensors (Pulsotronic kj4-m12mn50-anu) are used to measure the distance of the blades and the hub to the sensor itself, since the materials of both the blades and the hub are ferromagnetic. These sensors have a sensibility of $\pm 4V$ in the 4 mm linear range of operation. They have to be powered in the range of 11 V to 35 V DC, and they output an analog signal of 4.07 V at the distance they are installed (same height as the actuator magnets, to measure 0 V when the magnet is reached). The maximum and minimum signal levels of each sensor for the allowed ± 1.5 mm vibration distance of the hub and blades are of 0.3 V and 8.7 V. They are powered at 20V, to unify power sources with the operational amplifiers needed for the signal conditioning which are powered at $\pm 20V$ DC. Seven sensors are installed in total, one for each of the 4 blades, plus 3 for the tower vibrations (horizontal movement, vertical movement and rotation speed), and they are all attached to a different frame of that which they are measuring the distance of to get measurements with respect to a "static" frame.

The signal conditioning circuits for each sensor are just a standard two-phase inverter amplifier with a final gain of 0.5 V/V. The circuit is shown below in Figure 4.3. The final signal level of each sensor is of 2.07 V, with its range from 0.15 V to 4.5 V. The operational amplifiers (LM747) are powered to ± 20 V to be able to use unified power supplies. The resistors and the operational amplifiers have been chosen accordingly to obtain the desired gain.

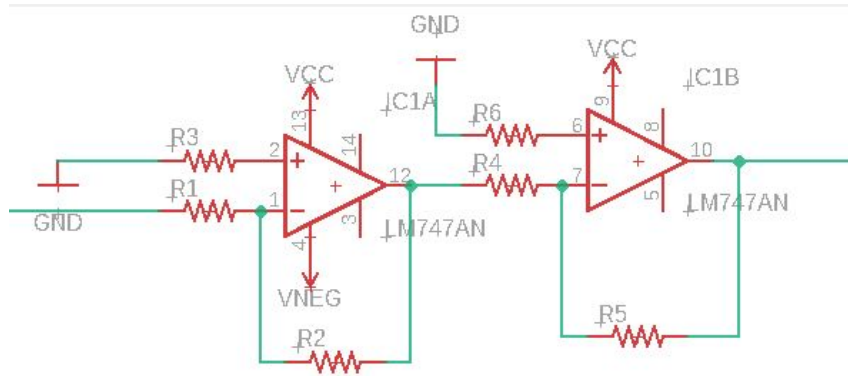


Figure 4.3. Signal conditioning circuit for each sensor

The conditioning circuit was embedded in a printed circuit board (PCB), which layout is shown in Figure 4.4 below. The connectors chosen for the layout are standard commercial connectors.

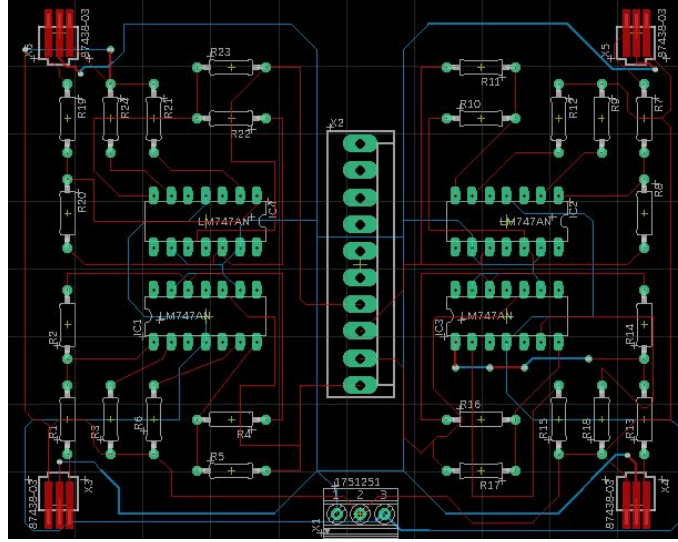


Figure 4.4. PCB layout for the conditioning circuit

The signal obtained after the conditioning circuit is still an analog signal, so an analog to digital converter (ADC) is needed to measure the vibrations and process them with a digital computer. The module being used is the ADS1105, with 12 bits resolution and 4 analog channels. The sample rate is maximum 3300 samples per second given by the hardware of the chip, which is more than enough for the system, since it vibrates at a maximum frequency of 15 Hz. However, the final sample rate is not that fast, since a communication system is needed from the control unit and that slows down the data acquisition procedure. The final sample rate is explained in the control unit section. The ADC module needs to be powered to +5 V, and it is powered from the DAQ units. It communicates with the DAQ via I2C protocol. Two ADC modules are installed, since there are 7 signals to convert and each ADC module has only 4 channels. One module is installed in the inertial frame and the other in the hub.

4.1.2 Actuation system

For the actuators some electromagnets are used, because the materials of the desired test rig are all ferromagnetic. The magnets have been tested and the magnetic field applied does not interfere with the measurements of the eddy current sensors. The installed magnets are standard 24 V DC electromagnets and have a voltage range of 0 to 20 V in the desired actuating range of ± 1.5 mm distance for blades and hub. Its force-to-voltage relationship has been obtained experimentally [9] and they have enough force to actuate in the needed distance range. Since the electromagnets can only exert attraction forces a total of 12 magnets are installed, 2 for each of the 4 blades plus 2 for each direction of the hub.

To actuate the magnets between their voltage ranges some drivers are needed. Each driver is an H-Bridge circuit with a power supply regulator that can supply the desired voltage to each magnet according to an enable signal given to the driver. The circuit installed is a L298N driver module, which allows to operate 2 magnets each module. Each driver module has 3 digital signals per magnet controlled, as well as powering inputs for the

magnets and the power regulator. The power regulator is powered from the DAQ unit to +5 V and the magnets are powered to a maximum of +20 V to unify power sources. The digital signals are two for controlling the direction of the H-Bridge (which are always set to one current direction) and one enable signal that indicates the power regulator the voltage to apply to the actuators. If this enable signal is a PWM signal fast enough, the driver will see it as a changing DC voltage signal. The duty cycle of the PWM applied is the percentage of the changing voltage over the total voltage supplied that the enable signal reads, therefore applying to the actuators the read voltage. This PWM signal is applied from the DAQ unit, the complete duty cycle (100%) indicates maximum input magnet voltage (20 V), and it switches voltages proportionally. Since each driver module can operate 2 electromagnets at a time a total of 6 modules are installed.

4.1.3 DAQ units

The DAQ units are responsible of obtaining the measurement signal for each sensor, sending it to the control unit via WiFi communication, receiving the actuation signal from the control unit and actuating accordingly. Therefore, each DAQ unit interacts directly with the ADC module, the actuators drivers modules and the control unit via WiFi communication. The DAQ unit has been chosen to be a Raspberry Pi 3B+ module, with integrated WiFi communication module and Raspbian operating system installed to run and compile the necessary code. It has I2C communication to obtain the measurements from the ADC modules and can apply PWM signals to operate the actuators drivers. The ADC library to read from the module, the communication protocol to send and receive from the control unit via WiFi and the PWM signal application are all included and explained with more detail in the C code for the test rig in [10]. A total of two DAQ units are installed, one for the inertial frame and one for the hub, and they are synchronized by the control unit.

4.1.4 Control unit

The control unit receives the measurements from the DAQ units, performs the desired control calculations and sends the resulting actuation signal back to the DAQ units for their implementation. It is also in charge of synchronizing the communications from the two DAQ units, the rotor and the hub. The available system for this task is the computer accessible in the lab, running Linux operating system for a faster and more accurate performance. The computer runs the C code explained in [10], which asks the DAQ units for the required measurements, synchronizes them, calculates the control signal in duty cycle percentage of the PWM and sends them back at the same time to the DAQ units. The WiFi communication is performed only with the two Raspberry Pi, the computer has a local WiFi hot-spot to which only the DAQ units will connect, to assure that there are few interference with the communications. The final sample rate of the whole system, including communications and synchronization is of 170 Hz, one magnitude order higher than the systems maximum vibrating frequency (15 Hz). The following block diagram in Figure 4.5 shows the control loop performed.

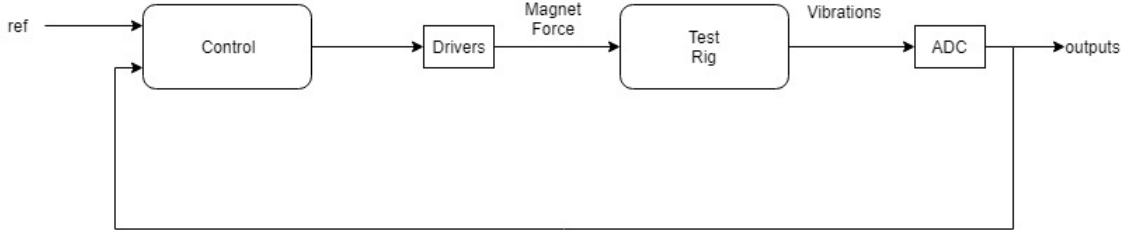


Figure 4.5. Control loop of the built system

The data sheets of every element installed in the test rig are all included in Appendix D.

4.2 Theoretical model of the system

The theoretical model of the test rig has already been described in a previous thesis [9] and also studied in [8]. For this specific application and this test rig, angular movements and gyroscopic effects of the hub have been neglected, since they are reduced by the design of the test rig. Also, the blades are attached non-twisted onto the rotor, which reduces the flexible blade bending motion to the x and y planes, and only those directions are considered. Coupling effects between blades flexible motion and hub motion are neglected in order to reduce complexity of the analysis. The following scheme in Figure 4.6 shows the mechanical model of the built test rig.

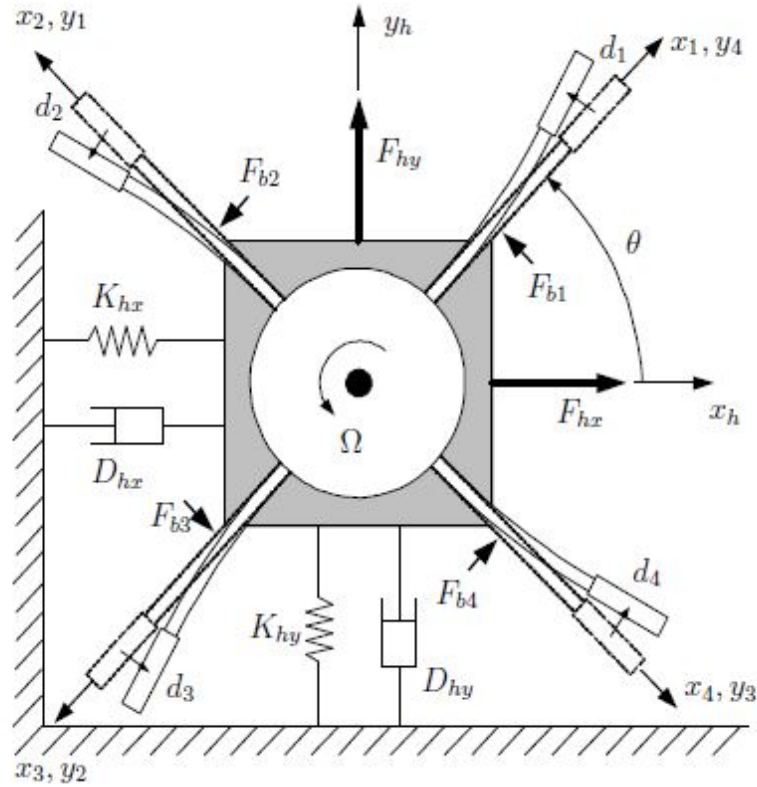


Figure 4.6. Mechanical model of the test rig

difficult task. Therefore, the analysis of this model has been done mixing the finite element analysis technique with the modal shape analysis. This leads to augmented coordinates of the system, because of the decomposition on modal coordinates. For deeper insight on this method refer to [9].

On this mathematical model analysis, the general equation of motion for mechanical systems is considered, as is shown below:

$$M(\theta)\ddot{z}(t) + H(\theta, \Omega)\dot{z}(t) + K(\theta, \Omega, \dot{\Omega})z(t) = p(\theta, \Omega, \dot{\Omega}) + Qu(t)$$

Where $z(t) = [x_h, y_h, d_1, d_2, d_3, d_4]^T$ is the displacement vector of the variables of interest. The matrix $M(\theta)$ is the inertia distribution matrix of the system, and only depends on the angular position of the rotor (θ). The matrix $H(\theta, \Omega)$ is the damping matrix of all elements of the system, and it depends on the rotor's angular position and velocity (θ and Ω). Matrix $K(\theta, \Omega, \dot{\Omega})$ is the stiffness matrix of the system, depending on the rotor's angular velocity, position and acceleration. The external forces matrices are divided in two elements, disturbance forces ($p(\theta, \Omega, \dot{\Omega})$) and controlled forces ($u(t)$) applied by a weighting matrix Q , and they are both time dependant. Since the angular position is changing in time periodically, the system matrices (M , H and K) are also periodic time varying matrices, as well as the external forces matrix p .

All these system matrices have been calculated numerically along with its dependence with time in [9]. They are of dimension 10 because of the modal decomposition done in the analysis, which decomposes each blade into the coordinates of its two principal vibration modal shapes. The calculated system can then be converted to state space form, which is as follows:

$$\begin{aligned}\dot{x}(t) &= A(t)x(t) + B(t)u(t) + B_d(t)p(t) \\ y(t) &= C(t)x(t) + D(t)u(t)\end{aligned}$$

Then the state space matrices are built following the general equation of motion for mechanical systems and with the calculated numeric system matrices. The state vector is built with the modal decomposed deflections of the blades, and is called $z(t)$. Therefore, the size of vector $z(t)$ is of 10 coordinates (each of the 4 blades decomposed in 2 coordinates plus the hub's coordinates), and the system's order is 20.

$$\begin{aligned}x(t) &= [z(t), \dot{z}(t)] \\ A(t) &= \begin{bmatrix} 0 & I \\ -M(t)^{-1}K(t) & -M(t)^{-1}H(t) \end{bmatrix} \\ B(t) &= \begin{bmatrix} 0 \\ -M(t)^{-1}Q \end{bmatrix} \\ B_d(t) &= \begin{bmatrix} 0 \\ -M(t)^{-1} \end{bmatrix} \\ C(t) &= [I \quad 0] \quad ; \quad D(t) = 0\end{aligned}$$

This is built with the knowledge of the modal decomposition of the measures, which is available, so that we can directly measure the 10 modal measures, even if we only have 6 sensors. For the actuation the principle is the same, each actuation coordinate will give the force needed to actuate in each considered mode shape of each blade, and they just have to be composed again with known relations to get the real forces applied.

4.3 Simulation model

The simulation model to test the controllers before their implementation has been designed in Simulink and MatLab in [8]. The model is based on the theoretical mathematical model explained above, and is implemented using the general equation of motion for mechanical systems to create the block diagram with the numerical calculated matrices in [9]. The block diagram of the mechanical plant is shown on Figure 4.8.

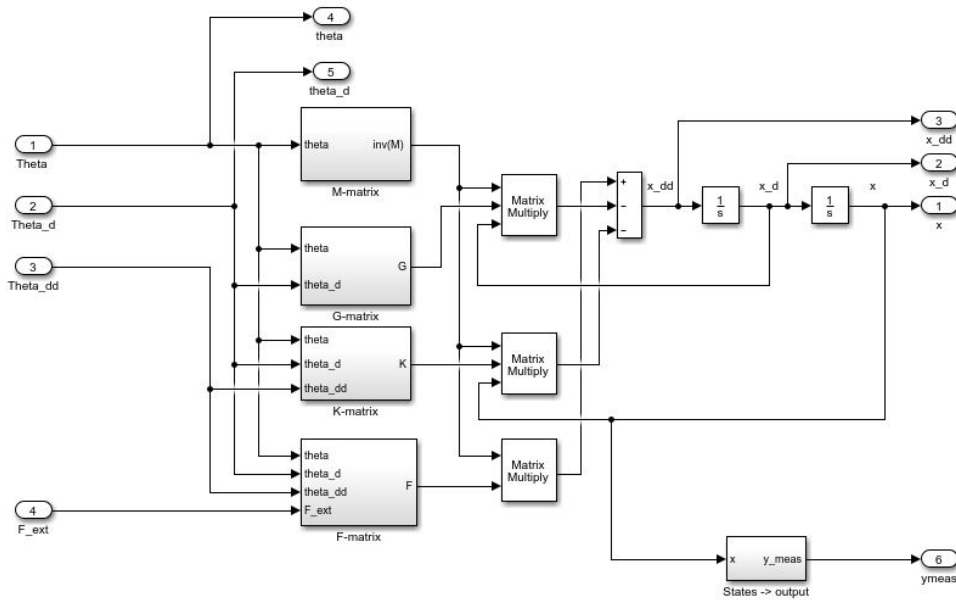


Figure 4.8. Block diagram of the mechanical plant

The subsystems included in the blocks M-matrix, G-matrix, K-matrix and F-matrix are the functions that calculate the necessary matrices $M(\theta)$, $H(\theta, \Omega)$, $K(\theta, \Omega, \dot{\Omega})$ and $F = u(t) + p(\theta, \Omega, \dot{\Omega})$. Those matrices are time dependant (because they depend on the angular position of the rotor, θ which is time variable) and in those blocks they are calculated on every update of θ, Ω and $\dot{\Omega}$, which are given as external inputs. Once those matrices are calculated in the current angular position and velocity the operations are performed to obtain the whole vector $\ddot{z}(t)$, which is integrated twice to obtain the velocities ($\dot{z}(t)$) and the angular positions ($z(t)$) of the variables of interest. The operation performed in the block diagram is the following:

$$\ddot{z}(t) = M(\theta)^{-1}(p(\theta, \Omega, \dot{\Omega}) + Qu(t)) + M(\theta)^{-1}H(\theta, \Omega)\dot{z}(t) + M(\theta)^{-1}K(\theta, \Omega, \dot{\Omega})z(t)$$

The block called States -> Output is the modal transform from 10 coordinates to 6 coordinates (which are the real measurements). The specific transformation is shown on the attached MatLab code included in [8], and has already been calculated in the previous modal decomposition [9].

Since the real system includes the electromagnets on the actuation, their dynamics also interfere with the physical system dynamics. The study and parameter measurement of the installed electromagnets is performed in [8]. The force exerted by the magnets depends on the power applied to them, the measured distance of the control target (rotor and blades) and the velocity of the control target. The block diagram built in [8] that represents the magnets dynamics is shown below in Figure 4.9.

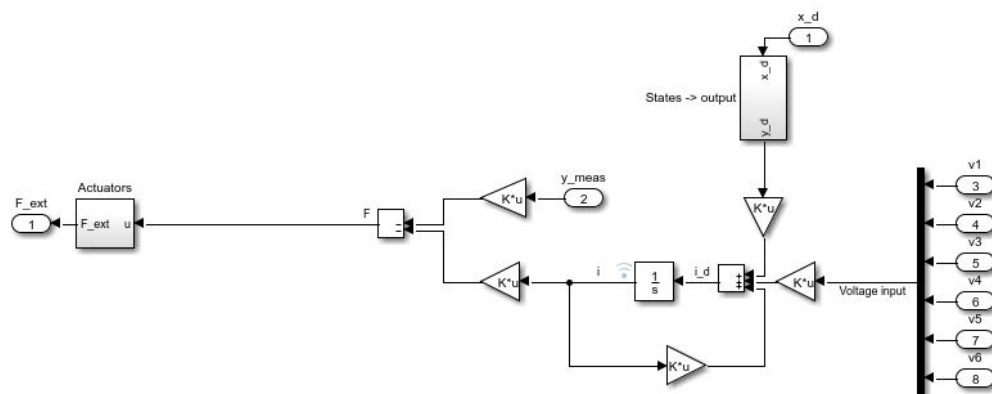


Figure 4.9. Block diagram of the magnets dynamics

The overall Simulink system of the open loop plant is shown below in Figure 4.10.

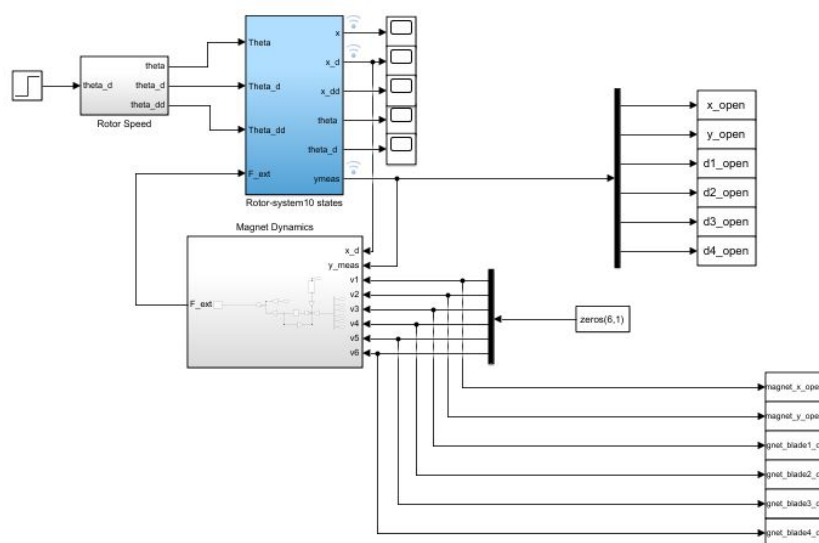


Figure 4.10. Block diagram of the open loop plant

In the overall system the external inputs are the angular position, velocity and acceleration of the rotor (θ, Ω and $\dot{\Omega}$) as well as the 6 actuation voltages of the electromagnets. Those actuation voltages on the magnets produce 6 forces that are decomposed on their modal shapes. That decomposition has also been studied before [9] and is available on the MatLab code in [8]. The measured outputs of the system are the 6 measurements made ($x_h, y_h, d_1, d_2, d_3, d_4$) and the angular velocity of the rotor, also measured (Ω).

The Simulink model of the plant and the MatLab codes for calculating the numerical matrices and initializing the blocks are available in [8]. The initialization of the parameters of the plant has been used for simulating the systems in this project.

CHAPTER 2: EXPERIMENTAL

5 IDENTIFICATION OF THE SYSTEM

The first objective of this project is to identify experimentally the plant that is object of the control. This way it can be compared with the theoretical plant to check if there are non considered dynamics or model errors, as well as process or measurement noise. It is also useful to identify the time variability of the system when it is rotating. If this variability is small enough a robust controller or even a static PID or full state space feedback controller would be the easiest controllers to design, since a change in the controller parameters would not be needed. However, if the variability is large, new techniques such as adaptive control could be a good approach to successfully reduce the vibrations of the blades when rotating.

The first part of the identification needs to analyze the system with the available knowledge about it. The theoretical model of the system explained above has 6 force inputs (actuation voltages applied to the electromagnets) and 6 displacement outputs (deflection of the blades and displacement of the hub in x and y directions), making it a MIMO (Multiple Input and Multiple Output) system. However, some dynamics of the electromagnets are usually non linear at several operational points, and those dynamics are not captured in the linearized model of the magnets. This dynamics could change the design of the controller and the stability of the plant when the system is in rotation. This system identification will also be helpful in the identification of those dynamics, since the whole plant will be identified. Those results will be useful to the design of the controllers.

As for the system identification procedure, the whole model to identify is stated as a dynamic mechanical mathematical model, which will be destined to control design. This is a necessary analysis at the beginning of the identification procedure, since the characteristics of the identification will vary depending on the type of model to obtain and the objective for this model. It is also known from the analysis of the system that the model is a continuous linear time-variant model, where some small non-linearities may be present (electromagnet characterization). However, the time varying parameters are periodic, as it has been shown in previous studies [9]. This periodicity is caused by the rotating movement and it makes it easier to identify the system because it is only necessary to be identified in one of the periods.

The system does also have external disturbances, which are the forces of gravity, the angular acceleration forces and the centrifugal forces due to the rotation of the inertial frame (all gathered in the vector $p(\theta, \Omega, \dot{\Omega})$ in the general equation of motion). Those disturbances are time varying as well, they are external non-controlled forces and they will be treated as disturbances. Another external input is the rotor's angular velocity, Ω , which will be given by the external motor and controlled by such. In the identified model

the control of this angular speed will not be included and the variation over time of the angular position (θ) and the angular velocity (Ω) will be given and calculated externally. The angular acceleration ($\dot{\Omega}$) for the operation of this test rig at all moment will be null, since it is assumed to only operate at constant speeds for this whole project.

This model identification is meant to obtain also the variability of the system when it is rotating, therefore the angular velocity of the rotor will be modified for some experiments. That means that the identification procedure will be divided in two components: with static rotor ($\Omega = 0$) and with rotating system ($\Omega > 0$). As it has already been analyzed [9] for low rotating speeds the variability of the system's parameters is linear, those speeds being around 300 rpm or 5 Hz. The system will be identified at both static ($\Omega = 0$) and maximum linear operation speed ($\Omega = 300rpm$), to identify the maximum range of the system's parameters. Since the system has been studied to be time variable when rotating [9], the identification procedure may need online identification when the system is rotating, and that possibility will be studied in the process of identification.

The system's order can be deduced from its theoretical mathematical model, and would be around order 20 (10 displacement modal coordinates and 10 velocity modal coordinates). However, the non-linearities of the electromagnets, the drivers modules and the PWM signals combined could add some dynamics, increasing the order of the system. That increment will be studied during the identification procedure.

The sensors are known to be linear, as have been explained in previous chapters, and proportional to the measurement they take. That proportional gain has been calculated in the simulation model, included in the MatLab scripts for the simulation of the plant developed in [8]. The measurement noise has been studied and its results are explained in the following chapters.

Once the system has been analyzed with the previous available knowledge, the identification procedure follows these principal choices and steps to be performed [12]:

1. **Experimental data design:** where the input to the system to be identified is designed to obtain the maximum information about it, and the outputs are recorded, pre-filtered and analyzed.
2. **Model structure:** choice of the model structure to be identified, including the order model, the layout of the system's parameters and the system's noise.
3. **System identification:** where the actual identification procedure is performed, including the recorded data and the choice of the model structure.
4. **Model validation:** some validation criteria are chosen and the performance of the identification is checked.

This identification procedure is reflected in the following scheme [12] in Figure 5.1.

Since the identification procedure is determined to analyze and compare the identified model with the theoretical system, and only an identification of the plant is needed, it will be performed in open-loop. Closed loop identification introduces measurement noise in the feedback loop, and can corrupt the identification. The closed loop system identification is only performed when the system cannot be operated in open loop.

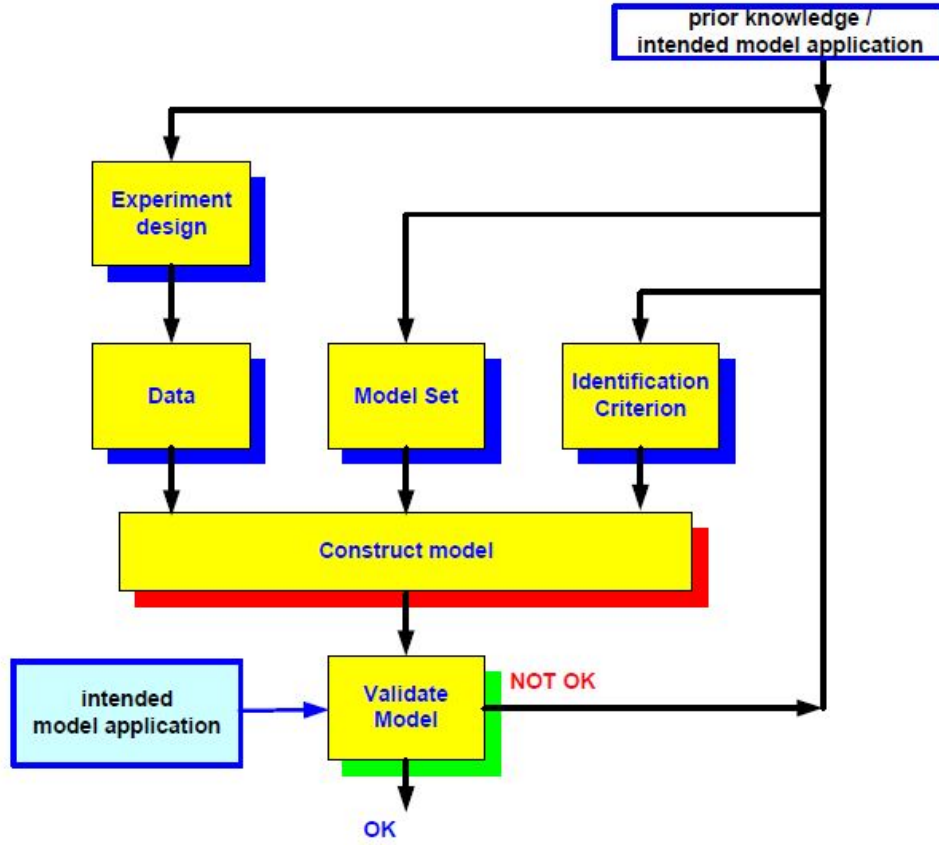


Figure 5.1. Flow scheme of the identification procedure

On the following chapters the design and choices made for the identification of the system are explained, attending the steps explained above.

5.1 Experimental data design

The described test rig has been modified due to some complications in its functioning. The final implementation of the WiFi communication protocol could not assure constant time steps between messages, making the sample time not constant. Therefore, the slip ring structure needed to be handled and its channels augmented, allowing the 4 signals from the sensors in the rotor to be connected to an external static analog to digital converter. This external ADC has been chosen to be dSpace, a real-time processor with I/O ports and data acquisition integrated system which includes MatLab and Simulink interfaces. This platform has not been chosen to be used for closing the loop in the full system since the slip ring does not have enough channels to control the 8 magnets, 4 sensors of the rotor, power and ground signals, therefore the needed signals could not be connected to this static processor. The processor module is already available in the laboratory and all sensors were connected properly to the ADC ports of the module. The experimental data is taken with the MatLab interface included in dSpace, and it records the data in a *.mat* file to be pre-processed in a MatLab script. The layout used of the user interface with the processor is shown in Figure 5.2 below, where all the measurement channels are shown.

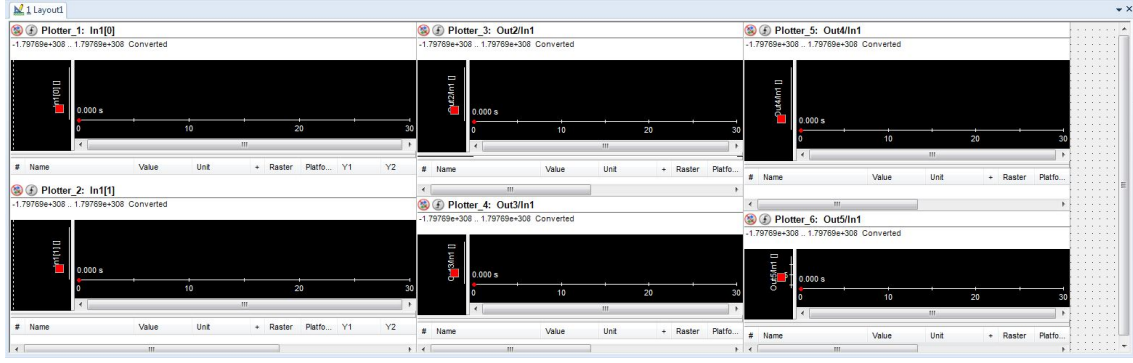


Figure 5.2. dSpace interface used for the measurement of all displacements

The experimental data to be taken is determined by the available sensors. The data will consist on 6 measurements over time, one for the displacement of each blade and one for each direction of the hub, horizontal (x_h) and vertical (y_h). The input signals that we can control are the 6 forces applied by the actuators, the electromagnets for each blade plus the electromagnets for each direction of the hub. This forces are applied in the same direction as the vibration movement, and in the middle point of each element. The other external inputs of the system are the disturbance forces (vector p) and the rotor's angular velocity (Ω). The latter will be constant and given externally by the motor. Those external disturbance forces are considered non-controlled, therefore they cannot be modified to obtain the experimental data and they are taken into account as external disturbances. As for the rotor's angular velocity, Ω , it will be given externally and is controlled by the motor imposing the torque, outside of the test rig. The angular velocity of the rotor will be changed for the experimental data recording as has been explained before.

Talking about the inputs and outputs, its amplitude and type of signal are studied. The outputs are the digital signals measured by the ADC, as have been explained above, and they will be recorded in volts (V) by the ADC interface. Since the sensors are linear in the 4 mm range and they are calibrated to 4 V at the installed distance, their conversion rate is proportional and linear. The sensitivity and amplitude range of the vibrations has been explained in Chapter 1 above. The amplitude of the vibrations is physically limited to $\pm 1.5mm$ vibration by the design of the structure of the test rig. The corresponding gain is calculated as follows:

$$K_{sensor} = \pm 1.5mm / \pm 4V = \pm 0.375mm/V$$

As for the input signals to the system, the electromagnets are controlled by digital drivers, explained before in Chapter 1. Those drivers can control the electromagnets with a digital PWM signal where only the duty cycle must be specified. This duty cycle can go from 0% to 100%. Since this identification is meant to include the electromagnets dynamics, the real inputs of the the system are the duty cycle (percentage of voltage applied over a maximum of 20 V) of each magnet. Those inputs need to be recorded as well for the identification process, they are measured in volts applied to the magnet (V) and their range is from 0 V to 20 V.

Once the inputs and outputs of the system are determined, the sampling time is to be

analyzed. The sampling time is determined by the ADC, and in the MatLab interface used for the dSpace module the chosen sample time is $T_s = 0.001s$, meaning that the sampling frequency is $f_s = 1000Hz$. This is two magnitude orders higher than the natural vibration frequency of the system (15 Hz) and therefore it is enough to avoid aliasing and to have accurate sampling. However, the input signals are generated by C code in the control unit and sent to the Raspberry Pis for their execution. Since the execution in C code has faster and non constant sampling times for the inputs, the inputs have been designed to have a sampling time faster than 0.001 s, and when the data is recorded and analyzed they will be re-sampled. Because of the properties of the inputs (binary and periodic) the re-sampling process will be accurate and not time-consuming for the control unit. Once the sampling time is discussed and chosen, the experiment time must be analyzed. Since this is a vibrating mechanical system it is fast enough to be in steady state in short time, depending on the excitation force applied. Some previous simulations on the system have determined that with an excitation force of 1000 N (around the weight of the whole test rig) the settling time was around 6 seconds. Therefore, the experiment time for this test rig has been chosen to be 60 seconds per experiment (10 times more than the settling time) to analyze the dynamics of the system and its steady state. With the chosen sample time and the experiment time the number of samples that will be obtained from the experiments is around 60000 samples. Whether they are enough to identify all system's parameters is discussed in the next section.

With all the considerations studied above, the design of the inputs to obtain the experimental data is performed. The experiments have to be designed to be informative, meaning that they should contain enough information about the system so that it can be distinguished from other similar systems [13]. It is also necessary from the input signal of the experiment to have persistence of excitation, meaning that it is constantly exciting the system in order to obtain information about it during the time of the experiment [13]. Finally, the input must have limited amplitude (limited by their physical application) and be applied in all its amplitude to maximize the information of the experiment (also maximizing the spectrum of the input signal, where the information matrix is obtained from) [13]. Some experiments should also be periodic, so that the excitation frequencies of the system can be shown and identified [13]. With all this considerations, the experiments are chosen to be the next ones, with the explanation of the input signal, the number of experiments of each type of input and the explanation of the choice of input indicated:

- **Noise:** simultaneously apply null input in all channels to measure the output unexcited, in order to measure and analyze the measurement noise and process noise present in the system (1 experiment).
- **Impulse:** separately apply impulse signals to every channel to check the coupling in every output measurement (6 experiments, one for every channel).
- **Step:** separately apply unit steps (20 V in magnet) to all input channels to check the steady state and the second order dynamics of the system (6 experiments), as well as the coupling for the first order dynamics between channels.
- **Random:** separately apply pseudo binary random signals (PRBS) to every input channel to check non linearities in each input/output channel and coupling of those non linearities (6 experiments).

- **Chirp:** separately apply frequency changing signals to input channels changing from 1 to 50 Hz, to identify all frequency responses of the system in the operation range (6 experiments).
- **Square wave:** simultaneously apply square wave signals shifted in every channel to identify couplings in the frequency response. The signals are shifted equally in the period, and the square wave has a frequency of 15 Hz, theoretical natural frequency of the system (1 experiment).
- **Random different seed:** simultaneously apply PRBS signals to every input channel with different random seed to check non linear response in the system (1 experiment).

The PRBS signals applied are random binary signals that come from a known algorithm (therefore they are pseudo random) and only take the maximum and minimum values of the signal (therefore they are binary). The input signals have been studied when it comes to their application. Theoretically, only one force is applied in every input channel, and it can take both directions of application. In practice, electromagnets can only attract, and that is why there is one installed at each side of each actuation point. Therefore, when designing the input signals for the experiments they needed to be divided into positive (attraction from one magnet) and negative (attraction from the opposite magnet). The impulse and step experiments have been applied only in one magnet in the positive direction (contrary to the side of the sensor). The chirp, random, square and PRBS signals have been applied using the full amplitude of our actuators, dividing the signal in positive and negative and applying the corresponding voltage to the positive or negative side magnet. In the input recording, however, all inputs are taken into account as unique voltages (positive and negative) applied to the actuators, which is theoretically the same approach.

There are designed a total of 27 experiments. However, they have to be taken in static ($\Omega = 0$) and rotating ($\Omega = 300rpm$) operation, making it a total of 54 experiments. With all design variables chosen, the 54 experiments are designed and recorded. The experiments were performed and the available data recorded and saved in the MatLab script included in Appendix A.

Once the experimental data was recorded the experimental information was analyzed to pre-process it. The preparation of the data is mainly divided into 3 operations [14]:

- **Handling of missing data and outliers:** where the possible missing data from the acquisition procedure is estimated with the MatLab function *misdata*. This function estimates the Nan (not a number) values in the recorded data set from the previous time values and tendencies of the same data. The outliers are estimated and eliminated after the model identification step with the residual analysis.
- **De-trending of data:** where the voltage levels of the sensors are normalized to 0, in order to identify the changes in the output from the applied input, not the levels of the signals. This is done using the MatLab function *detrend*.
- **Filtering of the data:** where the data is filtered to eliminate the high frequency noise and possible drifts in the signal levels. The filtering also focus the procedure in a specific frequency range, which has been chosen to be the 50 Hz bandwidth, since that is the maximum vibration frequency the magnets can actuate, and the bandwidth wanted to control. This operation is performed with the MatLab function *idfilt*

Before the data handling, de-trending and filtering some output and input handling is needed from the experiments taken. The built system applies the input via the Raspberry Pi modules, therefore the input recorded is taken from there, with changing sampling time between experiments due to clock limitations in the Raspberry Pi. The outputs are measured with the dSpace module, with the same sampling time ($T_s = 0.001s$). Therefore, the input data has been re-sampled to fulfill the wanted sampling time for the experiments. The loading of data and re-sampling is included in the MatLab scripts `data-handle-static.m` and `data-handle-rotating.m` included in Appendix A. Because the input and output are taken from different sources, the number of samples is not exactly the same, so the recorded data has been divided for every experiment to match the number of samples of input and output. The criteria for chopping data depended on how the experiments were recorded and is comprised in the corresponding MatLab code in Appendix A.

When the recorded data was taken and pre-filtered as explained above it was plotted to analyze its content. The first difference that is noticed is the vibration of every element of the test rig when operating in rotating conditions. This vibration is generated by the rotation, and is analyzed from the noise experiments. It can be seen from Figure 5.3 that the measurement noise (static operation) is limited to the $\pm 0.05V$ bandwidth. But, as can be seen from Figure 5.4, the measurement with unexcited inputs when in rotating operation is vibrating in a $\pm 1.75V$ bandwidth, due to the rotation.

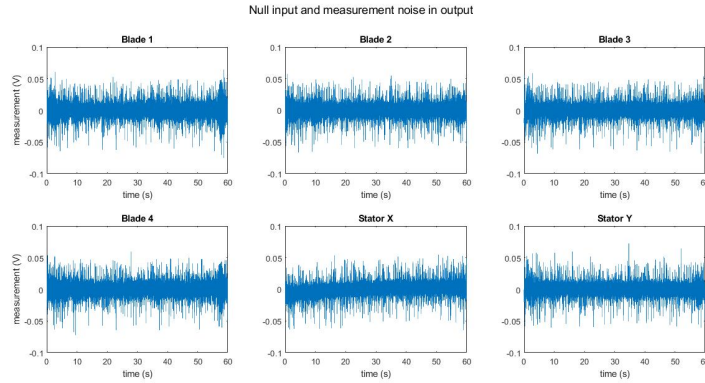


Figure 5.3. Output measurements with unexcited inputs and static operation ($\Omega = 0Hz$)

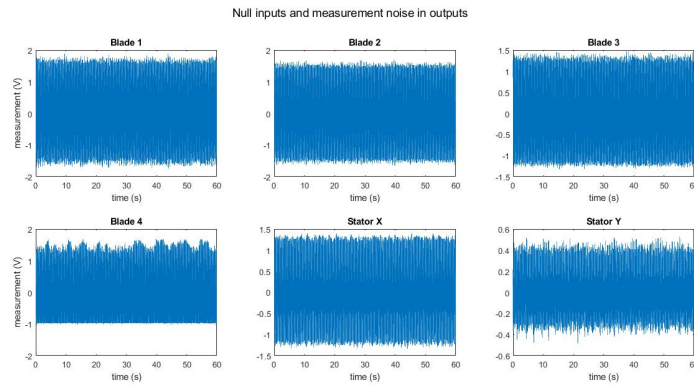


Figure 5.4. Output measurements with unexcited inputs and rotating operation ($\Omega = 5Hz$)

This increase on the vibration measurements indicates that when the system is unexcited and in open loop, the maximum vibration of the blades and the hub is of amplitude $A_{vibration} = K_{sensor} * (1.75V - 0.05V) = 0.6375mm$.

This is the vibration amplitude whose control and reduction is the purpose of this thesis. The saturation of the sensor of blade 4 in the noise experiment for rotating conditions is due to a misalignment in the blade when it was built in previous projects. This misalignment causes blade 4 to be asymmetric in the vibration space, and therefore saturating in one vibration side. This effect is also taken into account in the identification procedure as part of the system.

It could also be seen from the plots (examples in Figure 5.5 and Figure 5.6) that the impulse response on every channel in both static and rotating experiments is not informative at all. The measured outputs are essentially noise on every channel, without any dynamic response from the excited input. This is because the limitations on the system input do not allow to apply a short impulse big enough so that the response is larger than the measurement noise. Because of this, the impulse response data is ineffective and needs to be separated from the estimation data. The effect is taken into account when identifying the system, and the impulse response of the system is analyzed with the random input signals instead.

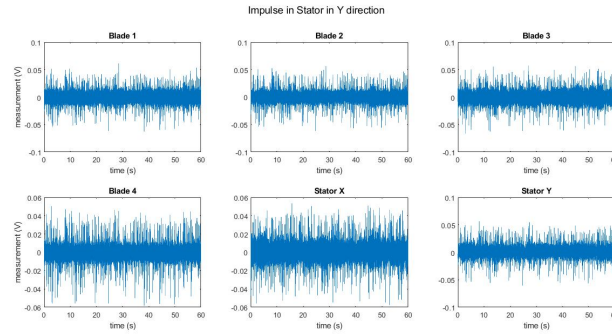


Figure 5.5. Output measurements with impulse input on the stator in Y direction and static operation ($\Omega = 0Hz$)

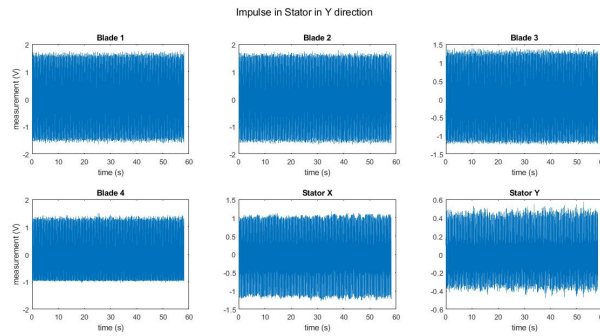


Figure 5.6. Output measurements with impulse input on the stator in Y direction and rotating operation ($\Omega = 5Hz$)

Analyzing the step responses of the system, when in static operation a change in the signal levels can be appreciated (Figure 5.7). However, when the system operates in rotating conditions, the vibrations caused by the rotation are much bigger than the effect of the constant force in the magnets when applying a step, therefore the step excitation of the input becomes invisible to the system when rotating. This is also caused because of the limitation in the input signal, and therefore the step response data is taken away from the estimation data when in rotating operation. The effect also needs to be taken into account in the identification procedure.

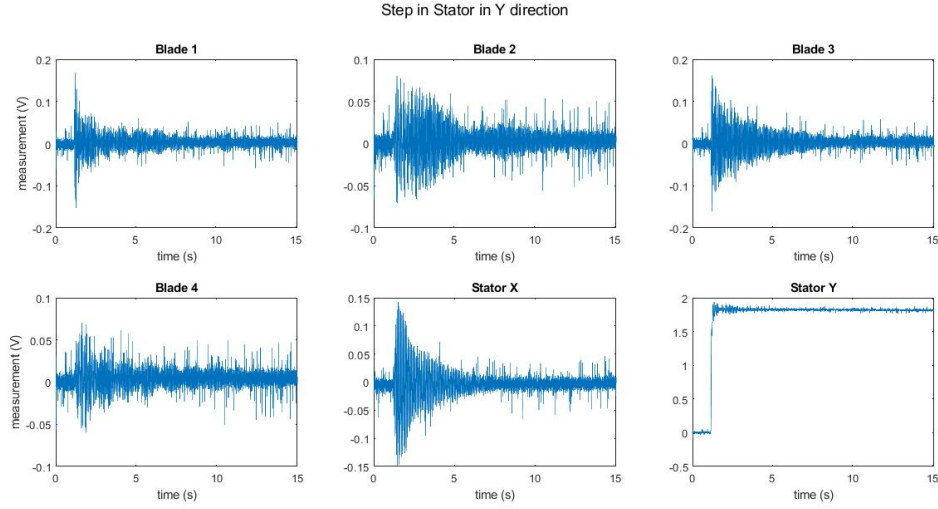


Figure 5.7. Output measurements with step input on the stator in Y direction and static operation ($\Omega = 0Hz$)

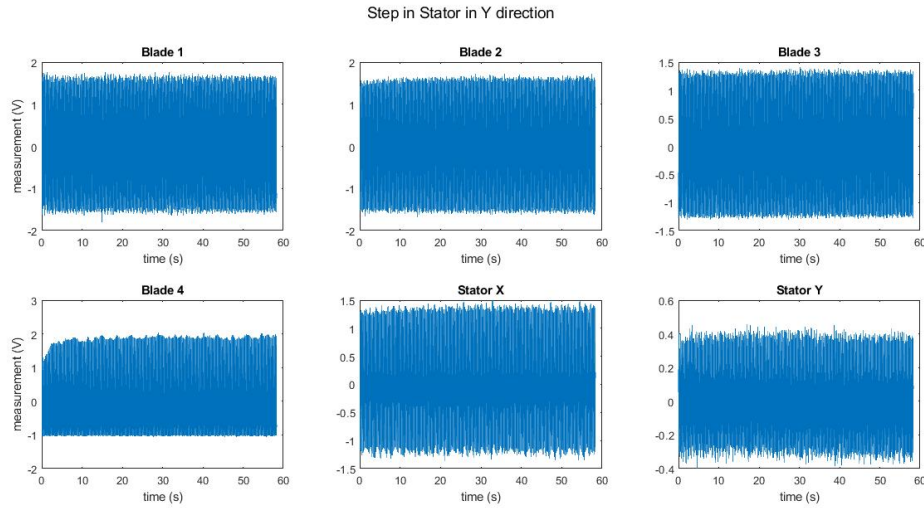


Figure 5.8. Output measurements with step input on the stator in Y direction and rotating operation ($\Omega = 5Hz$)

As for the PRBS and chirp experiments, their response can be appreciated in both static and rotating operations without any hidden dynamics. The system is measuring the biggest

changes in the signals when the input frequency in the chirp experiments reaches the natural frequencies of the system for the different elements. This phenomenon can be seen in Figure 5.9, where the measured output increases its amplitude on several points, meaning that the system is excited at those times at different natural frequencies.

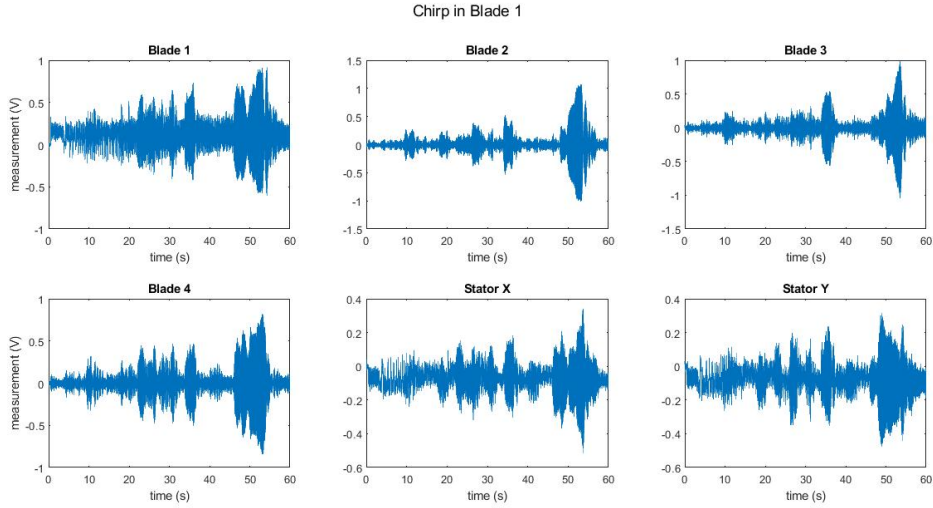


Figure 5.9. Output measurements with chirp input on blade 1 and static operation ($\Omega = 0Hz$)

Last, the random and square experiments are also seen on the output, with the clear difference of the rotating experiments, that dampen the dynamics any input can force on the system. The experiments are plotted and shown in Figure 5.10, Figure 5.11, Figure 5.12 and Figure 5.13 below.

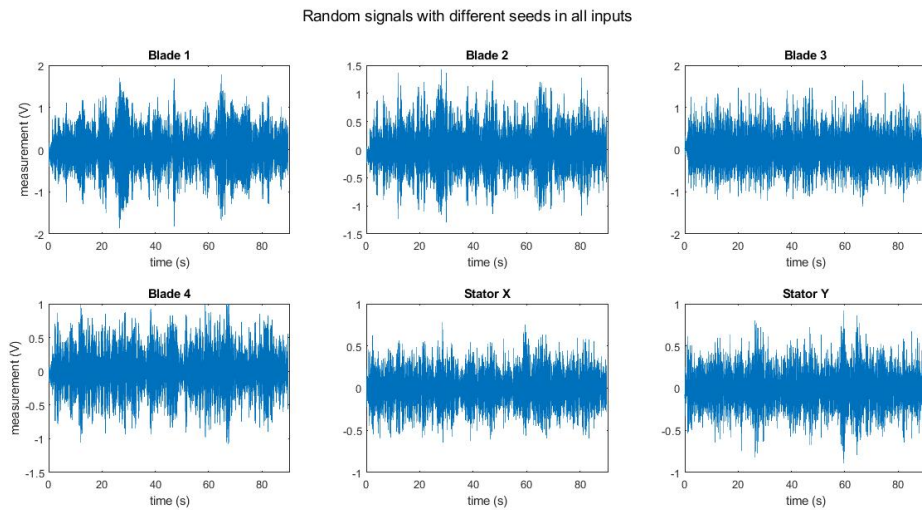


Figure 5.10. Output measurements with random inputs with different seeds on every channel and static operation ($\Omega = 0Hz$)

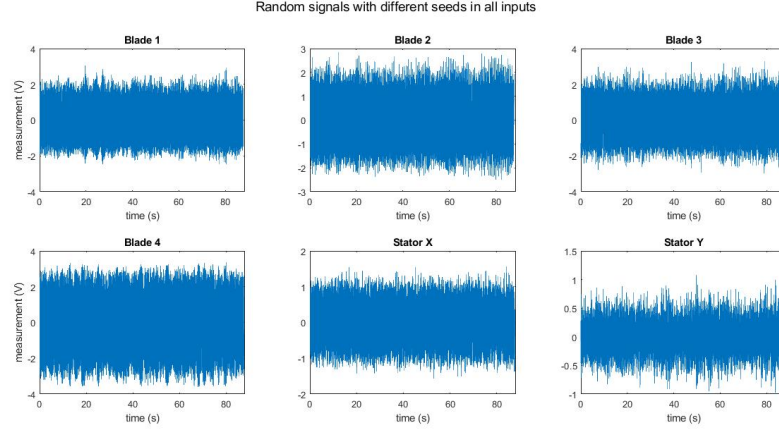


Figure 5.11. Output measurements with random inputs with different seeds on every channel and rotating operation ($\Omega = 5Hz$)

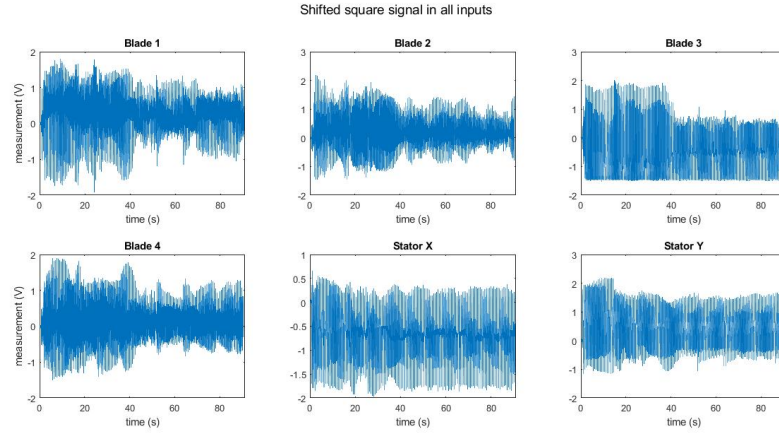


Figure 5.12. Output measurements with shifted square inputs in static operation ($\Omega = 0Hz$)

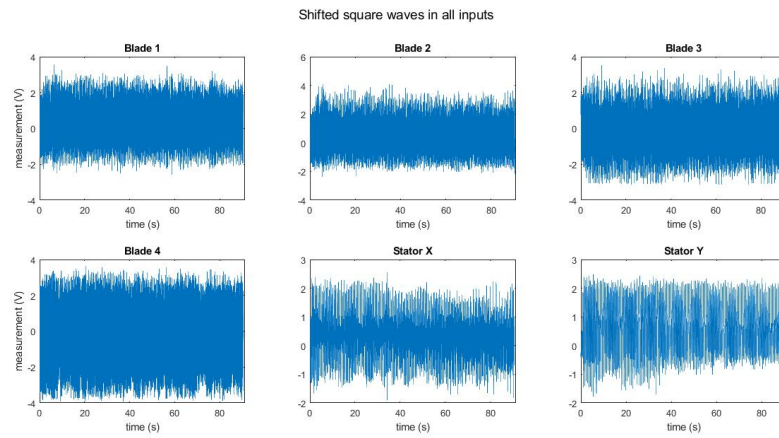


Figure 5.13. Output measurements with shifted square inputs in rotating operation ($\Omega = 5Hz$)

After the data preparation, filtering and analysis the main conclusion is that the transient response of the system will be difficult to analyze due to hidden transients in step and impulse experiments. Also, the rotating experiments show that the vibrations due to the rotation are sometimes higher than the input forces the electromagnets can apply, limiting the control actuation.

Each useful data set is merged into different MatLab iddata objects, that contain input data, output data, sampling time and the different experiments taken. Those iddata objects are used by MatLab to perform the identification. After the data is analyzed and pre-filtered a total of 4 iddata objects are generated by the code in Appendix A, one for estimation in static operation, one for validation in static operation, one for estimation in rotating operation and one for validation in rotating operation.

Once the experimental data is taken and analyzed, and before the identification procedure, the model structure from known data is studied to obtain the model structure and magnitude order about the order of the total model to identify.

5.2 Model structure for identification

After the experimental data has been taken and before the identification procedure takes part, an analysis of the structure of the plant is needed to narrow down the possible models to be identified. This step clarifies the identification and helps ensure the accuracy of the identification. The analysis is done with the available information about the physical system, some of that information has already been studied at the beginning of this chapter.

The system is a MIMO system, because it has 6 inputs and 6 outputs. It also is continuous, since all signals are analog mechanical signals in continuous time, even though the measurements and the actuation have to be sampled to be computed. The model is a time variant plant, as it has been explained before and studied in previous approaches [9]. It is also a linear plant, because the mechanical model can be explained with linear equations, although it may include some non-linearities which will be linearized in the operation point of the system.

The plant can be expressed as a standard state space model with external disturbances, process noise and measurement noise, as is shown below:

$$\begin{aligned}\dot{x}(t) &= A(t)x(t) + B(t)u(t) + B_d(t)d(t) + Ke(t) \\ y(t) &= C(t)x(t) + D(t)u(t) + D_d(t)d(t) + e(t)\end{aligned}$$

Being $x(t)$ the state of the system, $u(t)$ the inputs to the system (6 magnet actuations), $y(t)$ the measured outputs (6 vibration displacements of blades and hub in both directions), $d(t)$ the uncontrolled disturbances present and $e(t)$ a white Gaussian noise representing the measurement and process noise present in the system. The matrices $A(t)$, $B(t)$, $B_d(t)$, $C(t)$, $D(t)$ and $D_d(t)$ are the time variant parameter matrices needed to specify the whole system. The matrix K is the covariance matrix representing the relationship between the process noise and the measurement noise, since both noises need to be assumed as white and Gaussian. The dynamics of the disturbances and the controlled

inputs can be merged in one matrix $B(t)$, so that only one matrix is identified. All the state defining matrices are expected to be free form, not modal or canonical, due to the theoretical analysis made in Chapter 1.

The physical model can be expected to have 20 states (10 for the modal displacements and 10 for the modal speeds), as has been explained in Chapter 1. However, the electromagnets introduce some extra dynamics that will affect the whole plant. A basic linearized model of an electromagnet has 2 poles, but every electromagnet has non linearities due to magnetic losses and eddy currents [15]. Those non linearities will be included in the identification, but the basic linearized model has to be taken into account when identifying. That said, the whole expected system to identify would have 32 states, 20 for the physical system and 2 for each of the 6 electromagnets. The electromagnets are considered to be paired in the real layout, being the same magnet on one side and the opposite side, counted as one actuator each pair of magnets.

The inputs of the system are considered to not have any physical delay between input excitation and output response. This means that applying a voltage to the drivers of the electromagnets produces a force instantaneously that excites the system and produces a vibration in blades and hub. This is a reasonable assumption, since no delay is expected in an electronic system. The model to identify would also have null feed-through term (matrix D in the state space description is all zeros), since all the dynamics can be captured directly in the state matrix A , as has been analyzed in the theoretical study of the system.

When it comes to the number of parameters to be estimated in the model, the total number would follow this equation, being n the number of states (32), m the number of outputs (6) and p the number of inputs(6):

$$N_{parameters} = n * n + n * p + n * m = 1408 \text{ parameters}$$

In every identification procedure, the amount of data samples per experiment needed to identify those parameters is at least 10 times the number of parameters to estimate in the model [13]. So, with the nearly 60000 samples available in each experiment, the limit of 14080 minimum samples is surpassed, and therefore the data is useful for the structure proposed.

Before the identification procedure takes place, the experimental data recorded is divided in two sets. The first set is the estimation data, which will be used for estimating the model. The second set is the validation data, which is a data set that has the same experiments as the estimation data but taken at a different time, to be able to validate the identified model against some system response.

Once the system to be identified is analyzed, the identification procedure begins, using the experimental data and the model information.

5.3 Model identification of the plant

The plant of the system at hand will be identified with the experimental data taken and using the System Identification Toolbox available in MatLab. This toolbox allows to select

estimation and validation data sets and also the selection of the model structure and method of identification. A layout of the toolbox used is shown in Figure 5.14 below, where the estimation and validation data are already loaded.

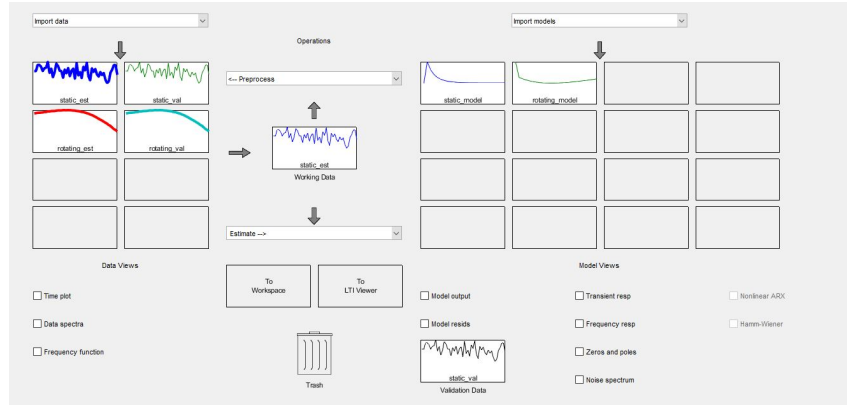


Figure 5.14. Layout of MatLab's System Identification Toolbox as it is used in this project

The first step on the system identification is loading the recorded data sets, as it is shown in the toolbox layout. Since the data is already filtered, analyzed and prepared for identification, the second step is the model structure selection and model order selection. The system to identify is a state space model, and the model structure has been selected as analyzed in the previous section. The model order is not known, but from previous knowledge it is expected to be around 32 states. The model order selection procedure is explained hereafter.

5.3.1 Model order selection

Once the model structure is selected, the toolbox will iterate to find the model order appropriate for this plant that best fits the recorded data. It will first evaluate the relative measurement of how much every state of the range selected contributes to the experimental response behaviour of the system. This way, an appropriate order for the system can be found, with a number of states that can define the system but such as it is not overestimated, which would add computational time and complexity to the model. Over-fitting the model would not give an accurate description of the plant, since it will include some dynamics that are not present or wanted to control. Since the objective of this identification is simulation of the plant to control its basic dynamics, the model needs to be simple enough to be controlled. That is why with this model order selection the minimal realization of the identified system is sought, enough to include only the dynamics wanted to control and not to predict the output data, which would require much more complex models.

The model order calculation is done using subspace identification methods, specifically N4SID (Numerical algorithm For Subspace Identification). This is a non iterative method for state space models identification that uses the Kalman prediction approach and input/output data for identifying state space models. It uses mathematical properties of the oblique and orthogonal algebraic matrix projection to estimate the column subspace

of the observability matrix (O) of the system and the row subspace of the state sequence matrix (X) from the input/output data available. With those subspaces calculated, the system order can be obtained by evaluating the singular value decomposition (SVD) of the product matrix of the observability matrix and the state sequence vector ($\Gamma = X * O$). The order of the system is chosen from the number of significant singular values obtained, which indicate the contribution of all states to the information of the system [16]. In the MatLab toolbox, after all singular values decomposition are evaluated, an interactive window is displayed, showing the results and the N4Horizons (prediction horizons for the algorithm) for each state, as well as the recommended model order for the stated purpose of the identification model. The interactive window for both static and rotating windows is shown below on Figure 5.15 and Figure 5.16.

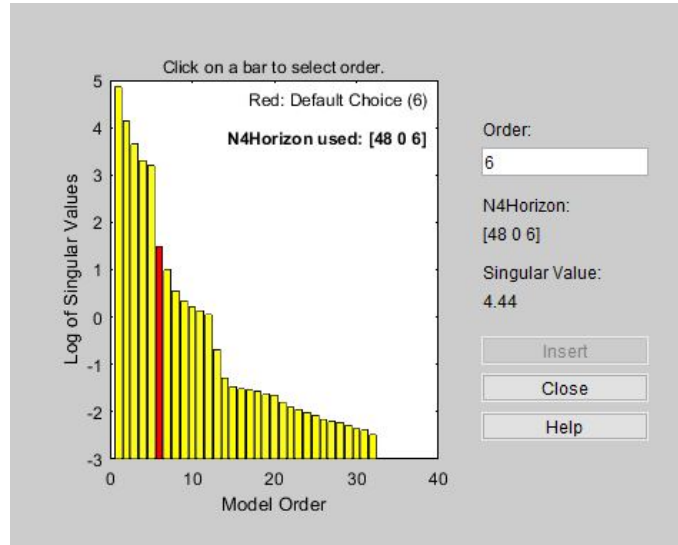


Figure 5.15. SVD for static data for model order selection in control focus

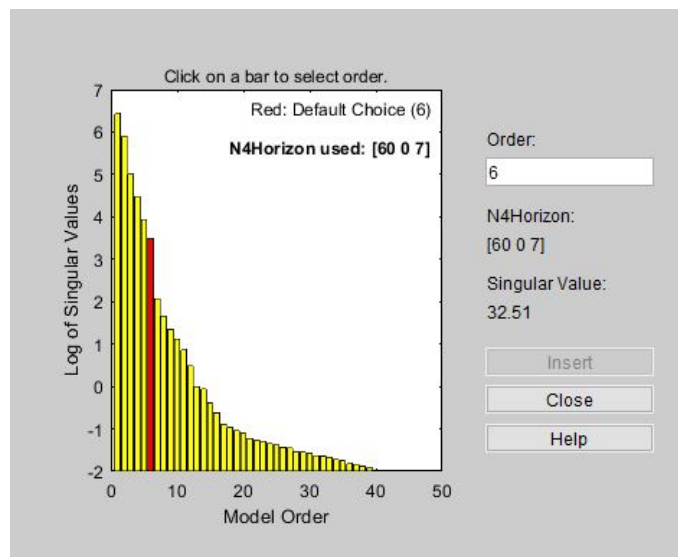


Figure 5.16. SVD for rotating data for model order selection in control focus

As it can be seen in the figures, the SVD gives several singular values for every state in the wanted range, and there is not a clear differentiation between significant singular values and disposable ones. This is because the system is complex, and it can be defined with the 32 states that were indicated. However, since the system is destined to control application, a minimal realization is needed. That is the reason why the recommended option is order 6 in both static and rotating models, because that is the minimal number of states necessary to explain the basic dynamics of the system for control purposes. More states would add complexity to the model and with only 6 states the system's information is explained in the necessary depth. The explanation about the choice of the 6 states is the presence of 6 measured outputs, which are usually the chosen states in every state space description, even though the identification method gives free form of the state matrices, therefore non direct states. So, even if the states of the identified model are not directly the taken measurements, they can be assumed to be a linear combination of them.

The same model order selection procedure is performed in the System Identification Toolbox, but instead of simulation focus it is changed to prediction focus. This outputs the same interactive window, but the recommended model order is increased to 32, the theoretically expected model order. This is because the real system is likely to have 32 real identifiable states, and the Toolbox is expecting to predict the output of the identified model with the data given. However, the complexity of this identified model is not necessary for the control purposes, just to check the theoretical results, and for this project the 6 states identification is used. The interactive windows of the model order selection for both static and rotating operation are shown in Figure 5.17 and Figure 5.18 below.

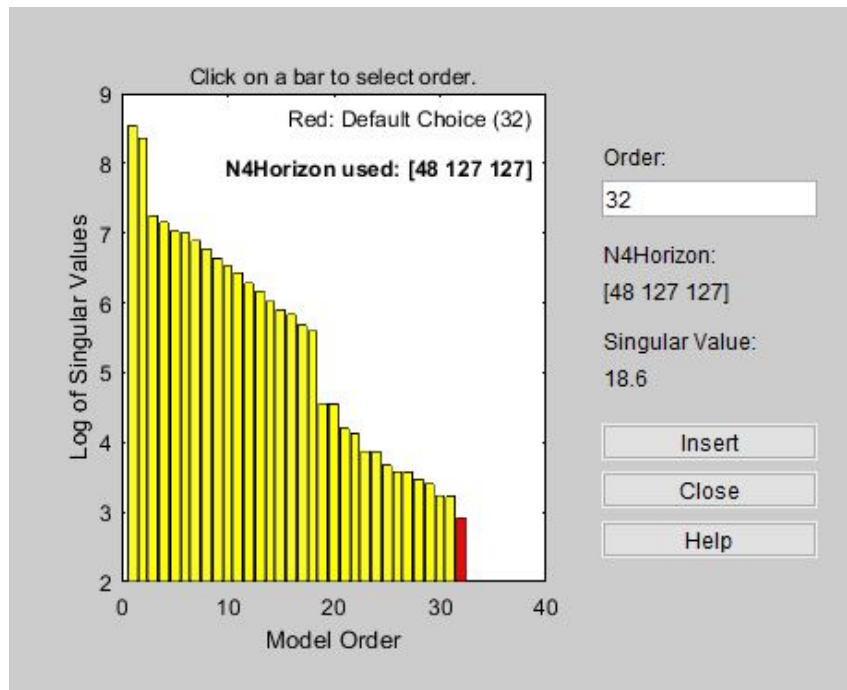


Figure 5.17. SVD for static data for model order selection in prediction focus



Figure 5.18. SVD for rotating data for model order selection in prediction focus

Once the model order is estimated and chosen as 6 states for simulation purposes, the actual model is identified.

5.3.2 Identification procedure

The plant that is wanted to be identified is clearly different depending on its operation conditions. When in static conditions the system behaves different than when it is rotating, where a frequency term appears in the vibration measurements of every element. This is why the identification procedure has to be separated in two components: static operation and rotating operation. In the last section a comparison between the two identified models and the real model simulated will be performed.

For the static model identification the same method used for determining the model order (N4SID) is the chosen approach. The system to identify is a MIMO system, and the purpose of the identification is to control the plant, therefore a model reduction is very useful. The numerical robustness of this subspace methods makes the identification accurate and a good comparison and validation can be made with the theoretical plant. Those are some of the advantages of the N4SID subspace identification method [16], therefore that method is the chosen one. The method needs the assumption that the process and measurement noise are not correlated with the input $u(t)$, which is a reasonable assumption in our process, since we are performing an open-loop identification. It also needs large amounts of experimental data, which we have proven to possess, and persistently exciting data [16]. That property was evaluated in the experimental data analysis and design before, and it was achieved for the recorded data .

The N4SID algorithm is performed by MatLab and its steps are explained in depth in

[16]. The weighting matrices for the basic 4SID algorithm are identity matrices, used for the singular value decomposition (SVD) of the weighted oblique projection. After the model order is selected as 6 states, the System Identification Toolbox operates several calculations for the identification of the system. The identified system is a continuous state space model with 6 states and quite small noise variance, in magnitude order of 10^{-6} . This reduced noise variance means that the measurement noise is small in the system, and the magnitude of the process noise is dependant on this measurement noise by the matrix K . This means that the noise present in the system in static operation is almost negligible, as it has been identified in this model. The measurement noise covariance (e_{cov}) and the relationship between process and measurement noise (matrix K) are shown below.

$$e_{cov} = \begin{bmatrix} 0.099 & -0.0177 & -0.0441 & 0.0079 & 0.0065 & 0.0023 \\ -0.0177 & 0.0562 & 0.0035 & 0.005 & -0.0033 & 0.0063 \\ -0.0441 & 0.0035 & 0.1093 & -0.0099 & -0.0058 & 0.0053 \\ 0.0079 & 0.005 & -0.0099 & 0.0574 & -0.0019 & 0.0077 \\ 0.0065 & -0.0033 & -0.0058 & -0.0019 & 0.0169 & 0.0027 \\ 0.0023 & 0.0063 & 0.0053 & 0.0077 & 0.0027 & 0.0184 \end{bmatrix} * 10^{-5}$$

$$K = \begin{bmatrix} 9.8122 & -0.4969 & 2.963 & -2.4903 & -1.636 & 0.8676 \\ 2.5855 & -7.0601 & 2.1136 & -2.2419 & -0.8031 & 0.2651 \\ 2.5498 & 0.2626 & 4.6164 & 8.2007 & -0.6606 & -0.4686 \\ -0.6847 & -0.2075 & -5.0590 & 1.2414 & 0.3384 & -1.3317 \\ 1.5884 & 0.8579 & 0.3504 & -2.8097 & 3.1139 & -1.5354 \\ 2.9477 & -3.5383 & -5.0750 & 3.3717 & 4.2578 & -0.4581 \end{bmatrix}$$

The step response of the identified static system is shown on Figure 5.19 below, where just the step from input i to output $j \forall i = j$ are shown.

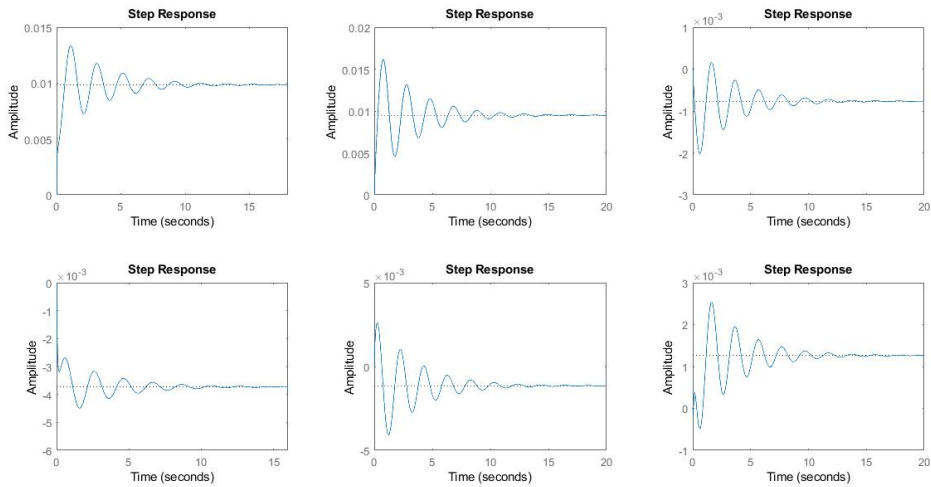


Figure 5.19. Step response for the identified model in static operation

The second part of the identification procedure is with the rotating operation data set. It is known from the theoretical analysis of the system that when the test rig is rotating the plant becomes time-varying, and its parameters change with time. However, the system identification procedure that MatLab has available for time varying plants is on-line identification, and that is not possible for the taken experimental data, which is off-line. That is why another approach when identifying the time-varying plant is needed. It was also concluded from the theoretical analysis that the time variation was periodic, therefore the parameters vary in a specific periodic range [9].

Because of the impossibility of identification of time variable parameters (TVP) a new MatLab toolbox is used. The CAPTAIN MatLab Toolbox [17] is a Computer Aided Program for Time series Analysis and Identification of Noisy systems that includes several functions for estimation of time variable parameters with offline data. It is an open source MatLab software provided by the University of Lancaster [17] that has a built in module specifically for TVP models.

This software performs the estimation of time variable plants using Kalman filter operation along with optimal recursive smoothing. The code implements the algorithm known as Fixed Interval Smoothing (FIS), where the smoothed estimate of the data vector of our system is obtained from all the data recorded in a fixed interval of samples, which is usually the full sample length of the experimental data [18]. This FIS algorithm assumes that the time varying parameters vary as a general random walk variation, which is a logical assumption for this system, since the time variation of the parameters is linear and periodic [19].

For this system the best identification method is the Dynamic Transfer Function (DTF) identification algorithm. In order to implement this algorithm, the system is considered a multiple-input, single-output (MISO) system, and will be analyzed for every output available to build the whole system. The time variability of the parameters of this plant follows an integrated random walk scheme because of the periodicity of the parameter's variation, as has been proved before, and for this identification the hyper-parameters of this random walk variations will be optimized beforehand by maximum likelihood [19]. Those called hyper-parameters are the characterization of how the system's parameters change with time, specifically the offset number and the time variation slope [19].

Since the plant's parameters are periodic and are similar within each period, for the model identification only one period of the data set is taken, to reduce computational time and effort. Because the period of the variability of the parameters depends on the rotating speed (5 Hz or 300 rpm), the period is of 5 revolutions per second, therefore one revolution in 0.2 seconds. This means that 200 samples have to be taken from the data set, since the sampling time is of 0.001 seconds. That is why the PRBS with different seed in every channel is the data set chosen to identify on, because a separate data set can be taken without losing the persistent excitation of the input.

Each MISO model takes the following form:

$$y_k = \frac{B_i(z^{-1}, k)}{A(z^{-1}, k)} * u_{i,k} + e(k)$$

Where y_k is the output measured at the k^{th} sampling time, $u_{i,k}$ is the input number i measured at the k^{th} sampling time, $e_i(k)$ is the noise term of each output at the k^{th} sampling time, z^{-1} is the backward discrete shift operator and $B_i(z^{-1}, k)$ and $A(z^{-1}, k)$ are the systems poles and zeroes for each input i with time varying parameters measured at the k^{th} sampling time. Each polynomial of the dynamic transfer function have the following forms:

$$A(z^{-1}, k) = 1 + a_{1,k} * z^{-1} + a_{2,k} * z^{-2} + \dots + a_{n,k} * z^{-n}$$

$$B_i(z^{-1}, k) = b_{0i,k} + b_{1i,k} * z^{-1} + b_{2i,k} * z^{-2} + \dots + b_{mi,k} * z^{-m}$$

With polynomial parameters measured at each k^{th} sampling time, and time variant. For identifying this plant, the system has to be divided into 6 MISO transfer functions to estimate and built after the estimation. For the characteristics described of the system, the algorithm is implemented by the MatLab toolbox and its parameters are estimated [19]. The experiment chosen to run the data on is the simultaneous PRBS signal with different seed on every input channel, since it outputs the most information about the non linearities of the system. Once the estimation is completed, the simulated output of the estimated transfer functions when the input chosen (simultaneous PRBS signals) is run through the estimated system are shown in Figure 5.20, with the system's noise filtered.

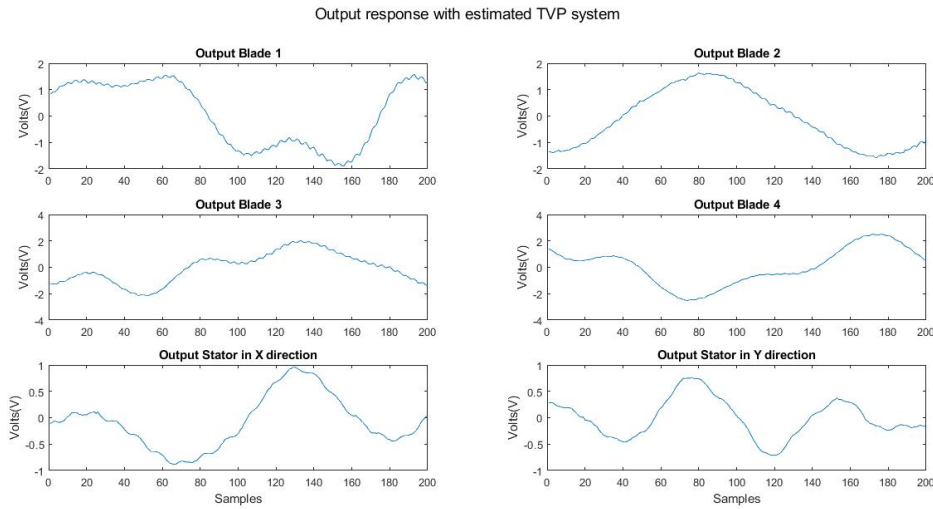


Figure 5.20. Output response for the identified model in rotating operation

As it can be seen in the plots, the system's parameters vary in time periodically when it is in rotating operation, and the input forces from the actuators have less impact in the output response than the parameter's time variability. This property was already observed in the data recording process, and it is a characteristic of the real system.

After each operating condition's model is estimated, they proceed to be validated to check the accuracy of the estimation.

5.4 Model validation

For the model validation the separation between rotating and static operating conditions will also be applied, and the different models will be validated separately. For static operation, since the plant has static parameters only the transient response of the system is of interest. Because it has been identified with simulation purposes and a minimal realization is made, the estimated model could not have good fit with validation data, also because of the noise model. When estimating this model only the noise and the transient dynamics of the model were wanted to be estimated, since those are directly related to the measured physical parameters explained in [9]. Those parameters are the static offset for the time variable parameters when the system is rotating. For that purpose, the validation of this estimation is done by analyzing the step response of the estimated model. The responses are shown in Figure 5.21, Figure 5.22, Figure 5.23, Figure 5.24, Figure 5.25 and Figure 5.26 below. Each figure shows the simulated step response of the static identified model and its 99% confidence intervals depending on the same confidence interval of the estimated parameters. In every plot only the response in each channel is shown where an input step is simulated in the same input channel (i.e. Figure 5.21 shows the simulated step response of blade 1 measurement when a step input is applied in the magnet actuating in blade 1).

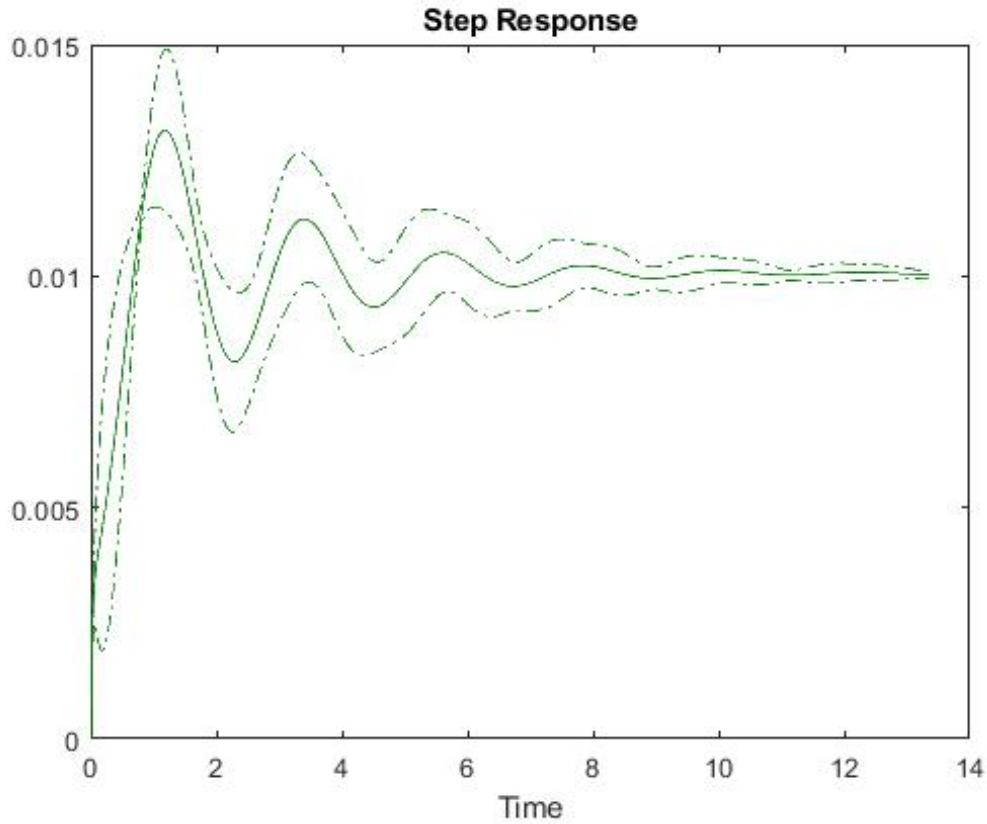


Figure 5.21. Output step response on blade 1 and 99 % confidence interval for the response on static identified model

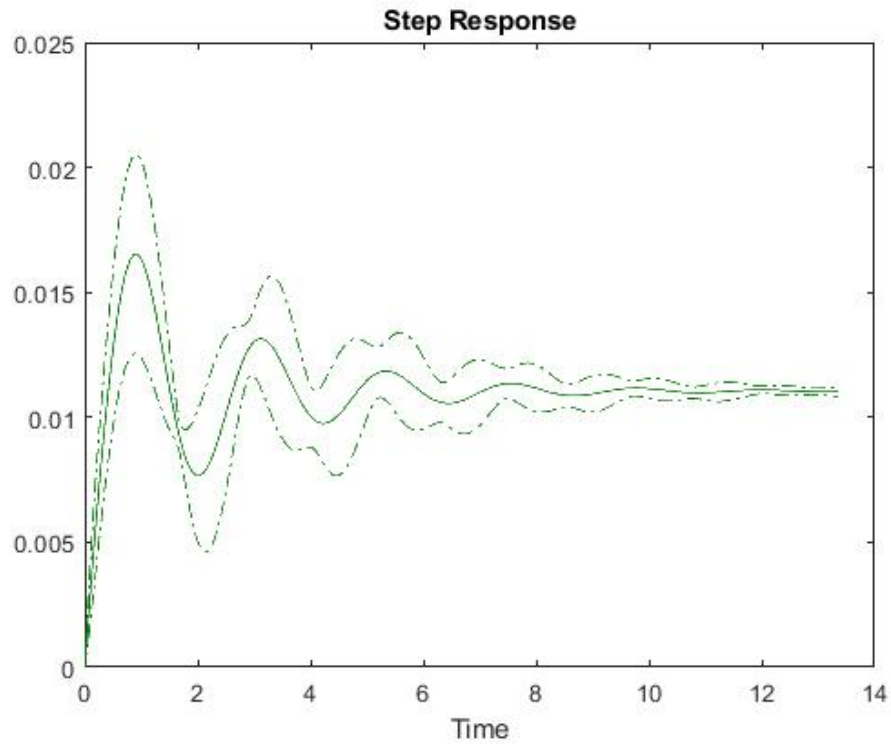


Figure 5.22. Output step response on blade 2 and 99 % confidence interval for the response on static identified model

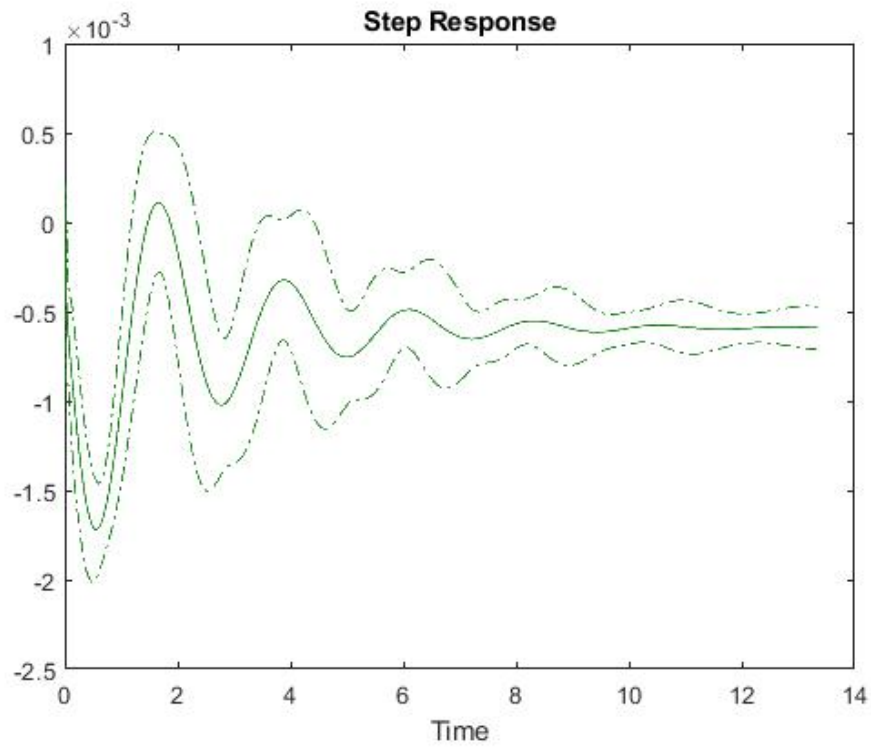


Figure 5.23. Output step response on blade 3 and 99 % confidence interval for the response on static identified model

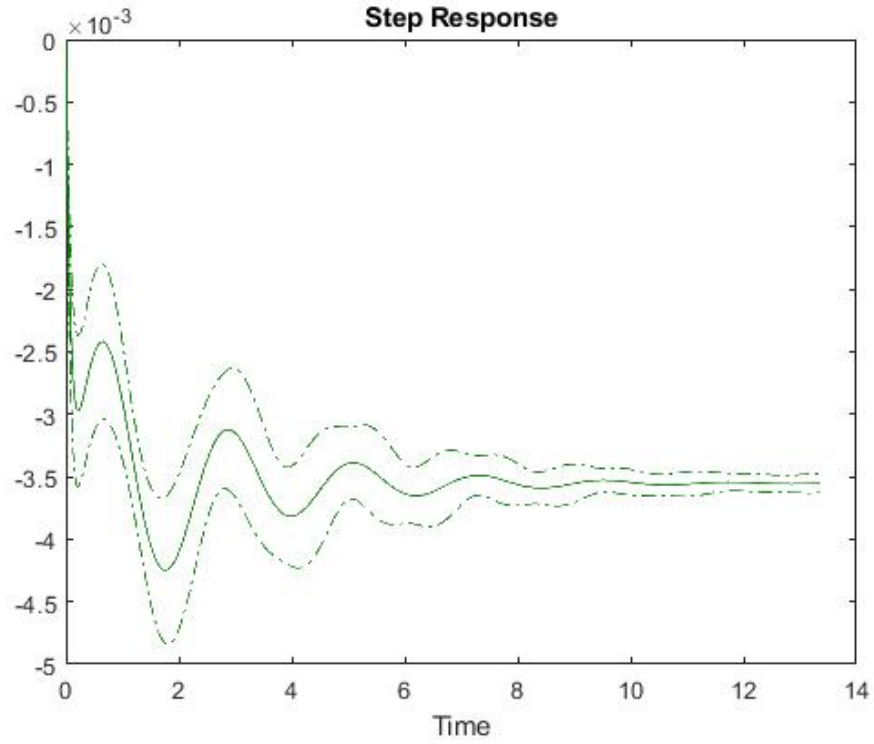


Figure 5.24. Output step response on blade 4 and 99 % confidence interval for the response on static identified model

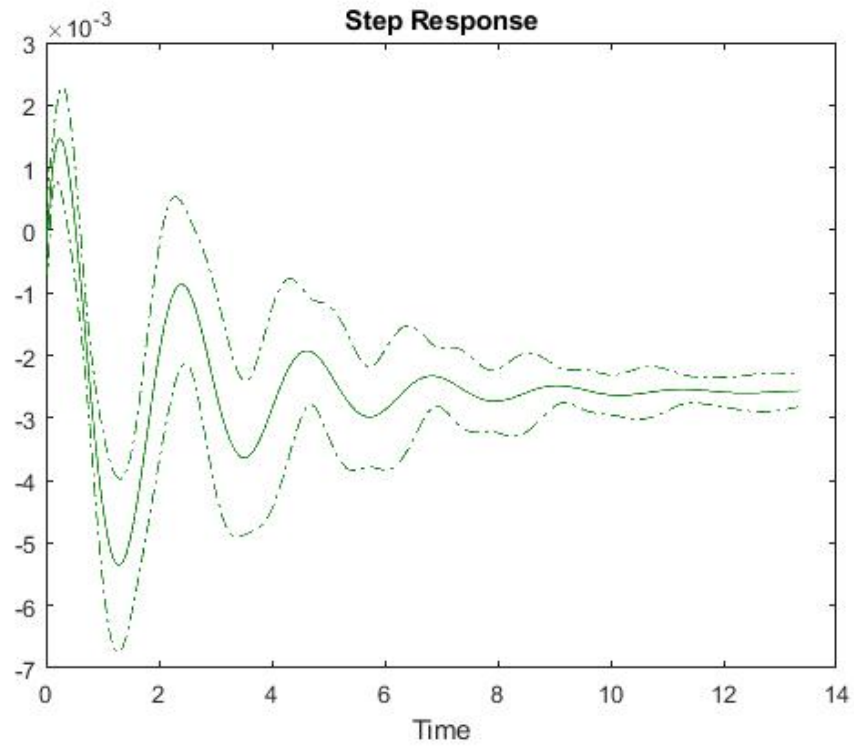


Figure 5.25. Output step response on stator in direction x and 99 % confidence interval for the response on static identified model

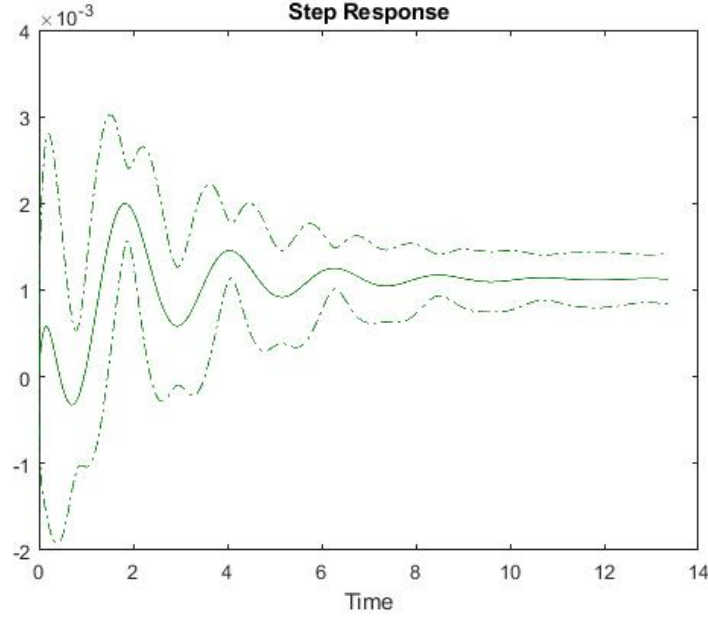


Figure 5.26. Output step response on stator in direction y and 99 % confidence interval for the response on static identified model

As it can be seen in the plots, the simulated step responses with filtered noise are very similar to that of a standard second order damped system, which is the simplified version of the system at hand. The plots show that the response of blade 3 and blade 4 are negative at first, showing that they are facing negative axis, in the sides of the rotor. They also display that the simulated step response when applied in the stator magnets in both directions is really damped (order 10^{-3}), which coincides with the high dampening values of the physical system displayed and measured in [9].

As for the rotating conditions model estimated, the standard error of the fit of the estimated parameters is an output of the *dtfm* function used for the TVP estimation, provided by the Captain Toolbox in MatLab [17]. Therefore, to validate the estimated model, the standard error of the estimation of each output at each sample are shown in Figure 5.27 below.

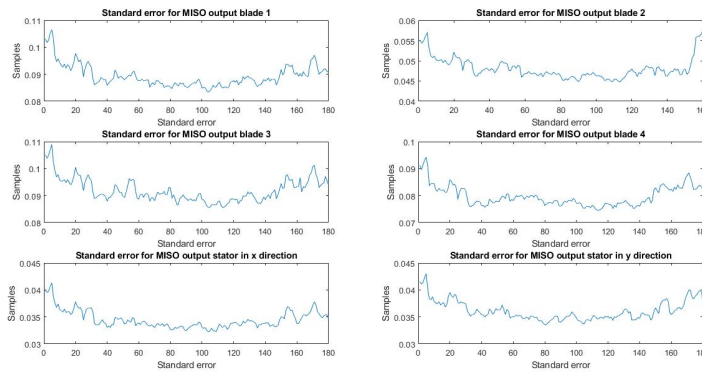


Figure 5.27. Standard errors for each MISO system on every sample on rotating identified model

The plots show that the standard error stays within the 10% on every estimated MISO system, therefore the estimation of the time variable systems can be considered an accurate estimation.

5.5 Conclusion of the identification

Once the model have been estimated and validated, a comparison with the theoretical model is made. For the static model, an example step response of the estimated model and a step response on the simulation model are both applied in blade 1 and compared in Figure 5.28 below.

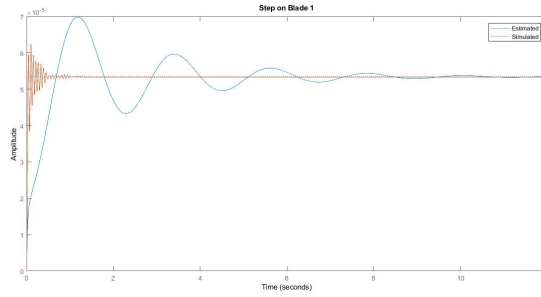


Figure 5.28. Comparison of step response of simulated model and estimated model

It can be seen in the response output that the estimated dynamics are much slower than the simulated ones, due to overestimation of the delays in the system. This overestimation, however, could fit the control purposes, since a more conservative control would be designed, and due to the non linearities of the real system, that could be useful.

As for the rotational model, the theoretical analysis of the system showed that the time variability of the parameters is completely linear. As it is shown in Figure 5.29 below, the estimated parameters variate linearly in time, reaching their peak value at the 200 samples, since only an estimation of one period was made. Figure 5.29 shows the time variability of the estimated model dynamics (denominator on every MISO system, or A polynomial).

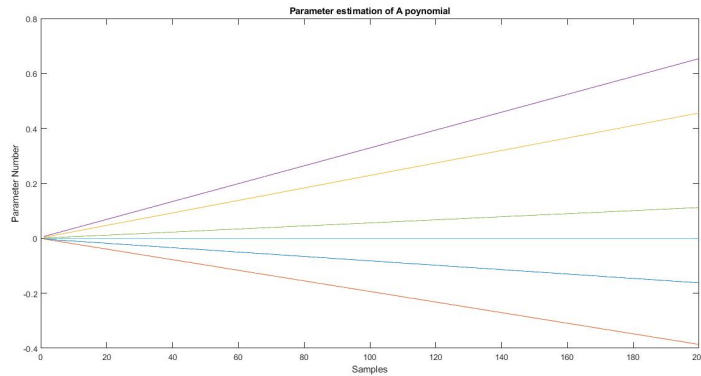


Figure 5.29. Parameter variation of the poles of the identified time variable system

The plot shows the parameters vary from 0 because it only shows the linear changes of each parameter, not the offset value, which are the already estimated values in static operation conditions.

In conclusion, the theoretical plant coincides with the estimated one at least in their main assumptions pertinent to the control of this model. The second order dynamics in static operation and the linear time variability of the parameters when in rotating operation are as was analyzed theoretically in [9].

CHAPTER 3: CONTROL DESIGNS FOR THE

6 SYSTEM

The second part and main objective of this project is to control the vibrations of the blades by either controlling just the hub (from now referred to as uncoupled system) or both blades and hub (for now referred to as coupled system). This analysis is directly applicable to a real system, since installing sensors and actuators in the blades is a challenging task for many real applications. Therefore, the real analysis to be made is if it is possible to control the weathering of the blades by avoiding the installment of a control system in them. If it is not possible to control their vibrations, a control for the whole system (blades and hub) will still be designed for its possible application.

The first step before designing any controller is performing an analysis of the plant. This analysis has been developed in the previous chapters. The plant has been analysed theoretically in previous work on this test rig [9] [8], and an experimental identification of the system has been performed in the previous chapter. As was shown in that study, when the plant is in static operation it behaves as a standard second order dampened mechanical system. However, when it operates in rotating conditions the parameters of the system become time variant in a linear, periodic variation. This variation is considerable compared with the actuation forces the system's actuators can exert. When the system is rotating, the measured outputs show a vibration of frequency similar to the rotation frequency and around 0.70 mm amplitude, half of the physically limited total amplitude of the system, as it can be seen in Figure 6.1. This vibration caused just by the rotation is enough to improve the effort of the controller when it tries to dampen those vibrations.

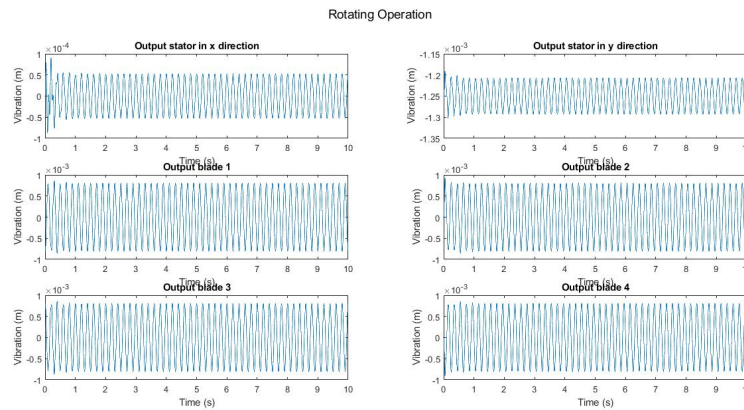


Figure 6.1. Output response for the simulated model in rotating operation with null inputs applied

The simulation model of the plant explained in Chapter 1 will be used to test the controllers after their design phase. The controllers chosen to be designed and applied are PID controllers for every uncoupled plant, full state space feedback controller and stochastic adaptive controller. The PID and full state feedback controllers will be designed on the identified plant in static operation, since they are static controllers. The adaptive controller will need to be designed on the simulated plant. All of them will be essentially regulators, since the wanted performance of the closed loop is to keep the vibration measurements at their static point, at null vibrations. In this chapter only the design of the controllers will be performed, and their implementation will be included in the next chapter.

6.1 PID controller design

As it was stated before, the PID controller design is performed on the identified plant in static operation because it is a static controller. The plant has been divided into 6 uncoupled SISO (Single-Input Single-Output) systems from each input to their corresponding output. This operation can be done only if the systems can be considered uncoupled, meaning that the crossed inputs do not interfere much in each other output. This assumption is made for the design of the PID controllers and any coupling between crossed inputs will be considered an external disturbance on each SISO system.

Since the designed controllers are regulators (the reference to track is to keep vibrations at 0), the controllers are sought to have good disturbance rejection properties. For that purpose, while designing them the different disturbances applicable to the system are studied, specifically the disturbances applied in the control signal $u(t)$, which is the input to the plant, and in the output measurements $y(t)$. The chosen disturbances are a sine wave of 0.7 mm amplitude and 5 Hz frequency (similar to the disturbance caused by the rotation of the system in rotating operation) applied to the output $y(t)$ and a step of 20 V for a short period of time (1 second), simulating the coupling between input signals applied to the control signal $u(t)$. The disturbances present in the system are plotted and shown below in Figure 6.2 and Figure 6.3.

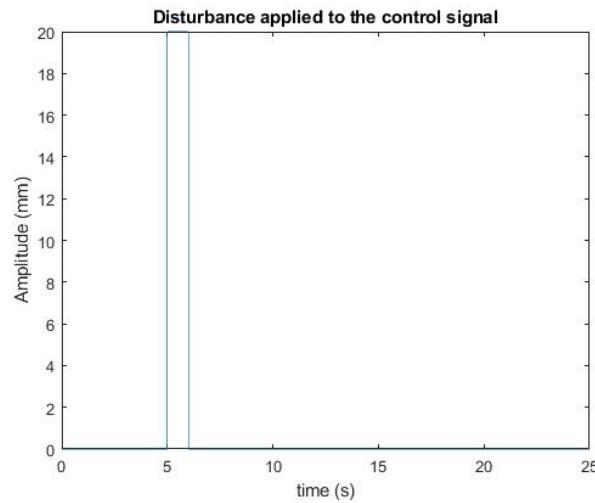


Figure 6.2. Disturbance applied to the control signal $u(t)$ for the design on the PID controllers

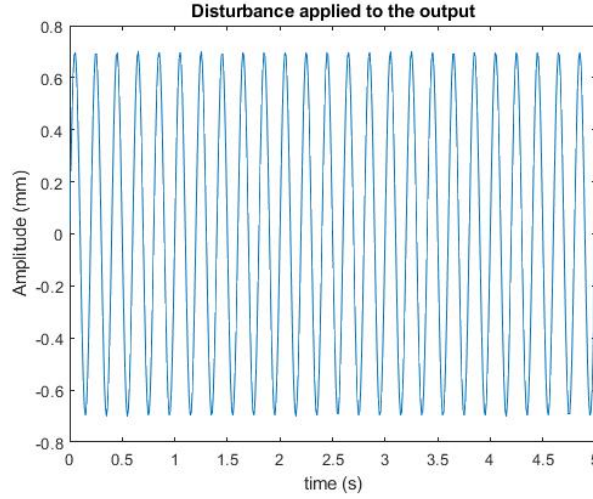


Figure 6.3. Disturbance applied to the output measurements $y(t)$ for the design on the PID controllers

Since the sine wave disturbance simulates the system in rotating operation, the static controllers will only be designed for the control signal step disturbance. That is the only one that can interfere with the system in static operation, and simulates a disturbance force in the system (external or coupling between crossed inputs).

Before the design of the controllers, a deeper analysis of the different SISO plants is performed. It was seen that every identified SISO transfer function has some poles and zeros that do not interfere in the transient dynamics of each plant. Those dynamics are much slower than the system itself, and therefore they are never actuating in the normal operation for the static operation system. Those dynamics have been removed in order to simplify the plants and the design of the controllers. To prove that those dynamics did not interfere in the system transient a step response of the original system and a step response of the reduced system is shown in Figure 6.4 below. It can be seen in the plots that the systems responses have very slight variations from the original systems.

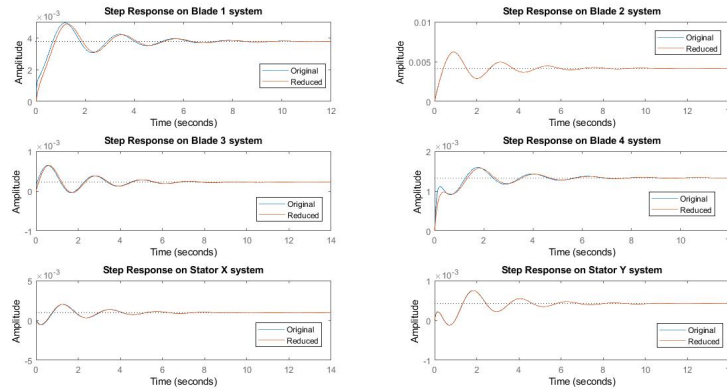


Figure 6.4. Step responses of reduced and original systems for every SISO system

Once the plants have been simplified, the PID controllers for those plants have been designed. The controllers have been designed with the PID Tuner Toolbox available from MatLab. This toolbox allows to work with any plant available in the workspace, and by selecting the plant, the wanted controller and settings (time or frequency domain, plot to show) it is possible to tune the PID controller. By means of how fast or slow is the closed loop response time, and how robust or aggressive is the control signal the PIDs for every plant are tuned, and their parameters are exported to MatLab workspace. This design method is purely visual and iterative, since several different gains are tested until the closed loop transient response is good enough for the control objectives. A layout of the toolbox is shown in Figure 6.5 below.

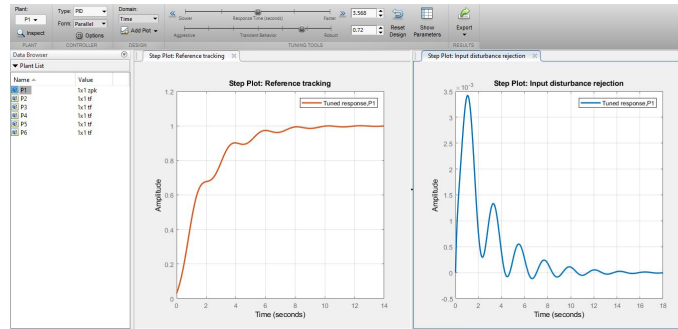


Figure 6.5. Layout of the PID Tuner Toolbox in MatLab

The designed controllers have been proven to control the vibrations in closed loop for the identified plant. Once their parameters are calculated, their closed loop responses are tested against the open loop response of the identified plant with the same disturbances on the control signal, to check if they are accurate enough to control the estimated system. The responses are shown below in Figure 6.6, Figure 6.7, Figure 6.8, Figure 6.9, Figure 6.10 and Figure 6.11.

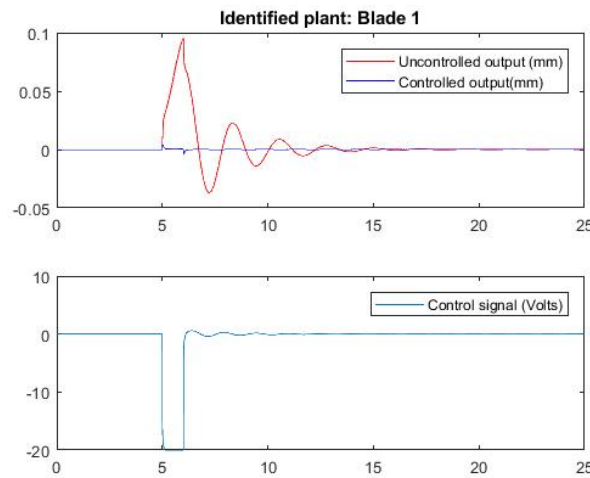


Figure 6.6. Comparison between closed and open loop response with designed PID for SISO system on blade 1

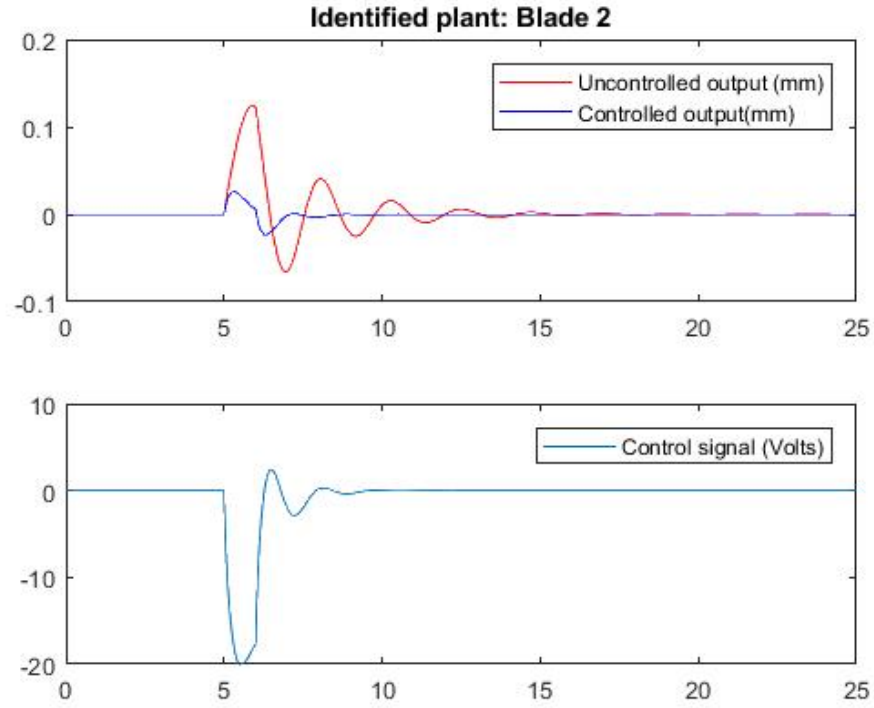


Figure 6.7. Comparison between closed and open loop response with designed PID for SISO system on blade 2

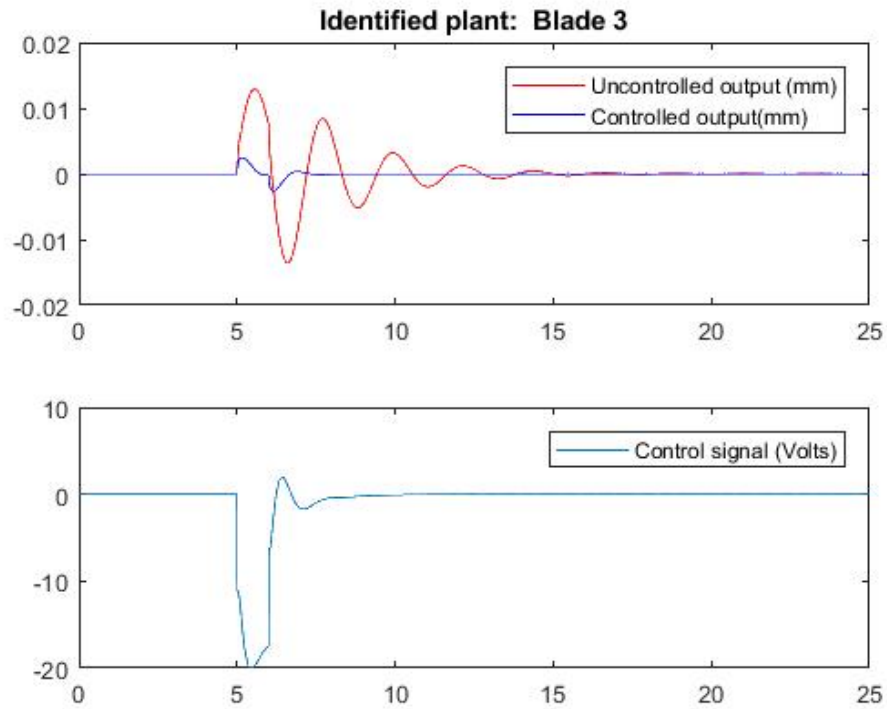


Figure 6.8. Comparison between closed and open loop response with designed PID for SISO system on blade 3

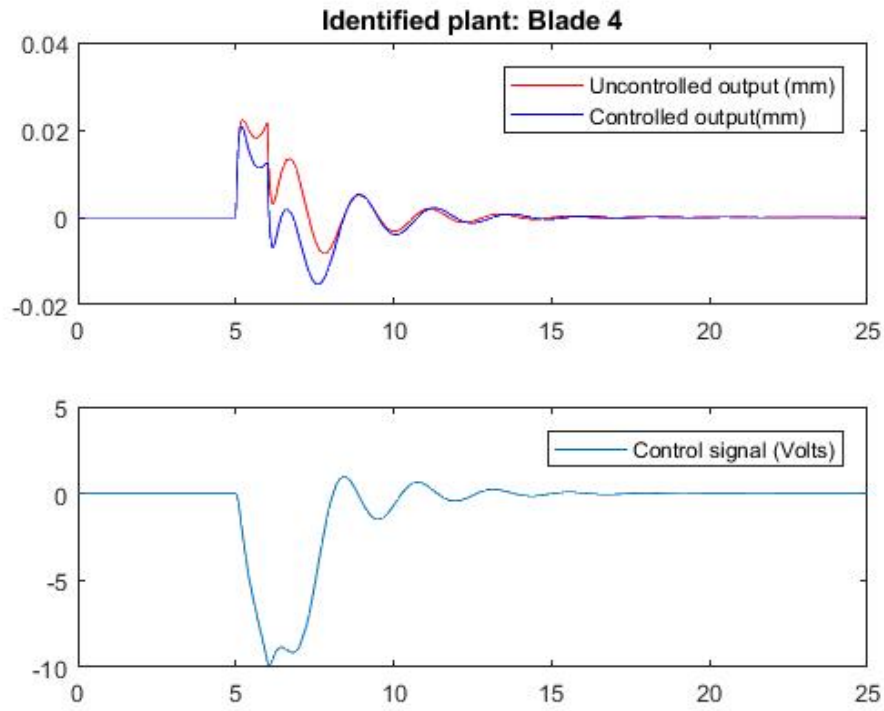


Figure 6.9. Comparison between closed and open loop response with designed PID for SISO system on blade 4

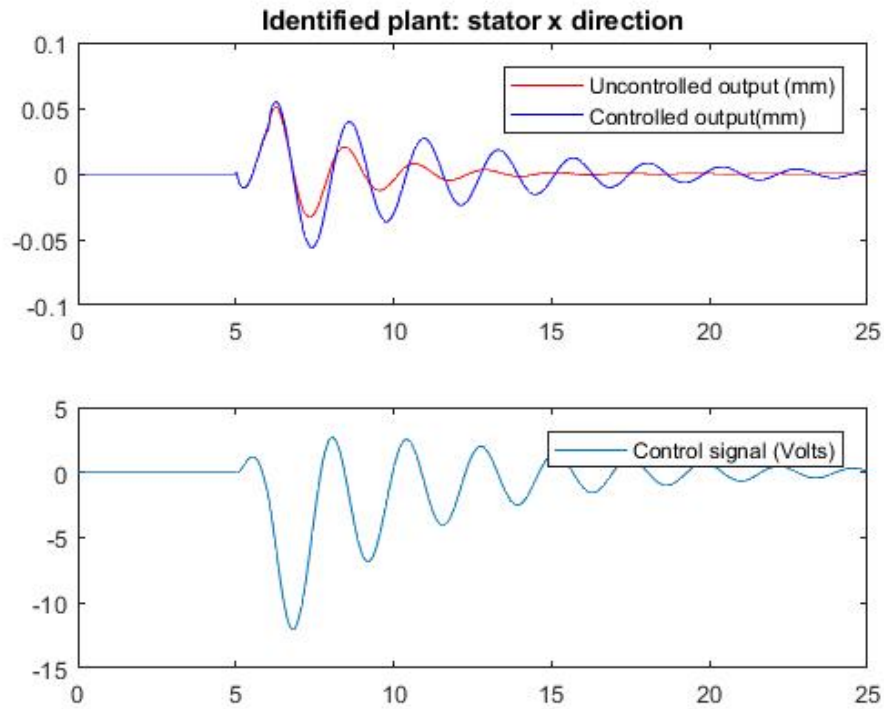


Figure 6.10. Comparison between closed and open loop response with designed PID for SISO system on stator in x direction

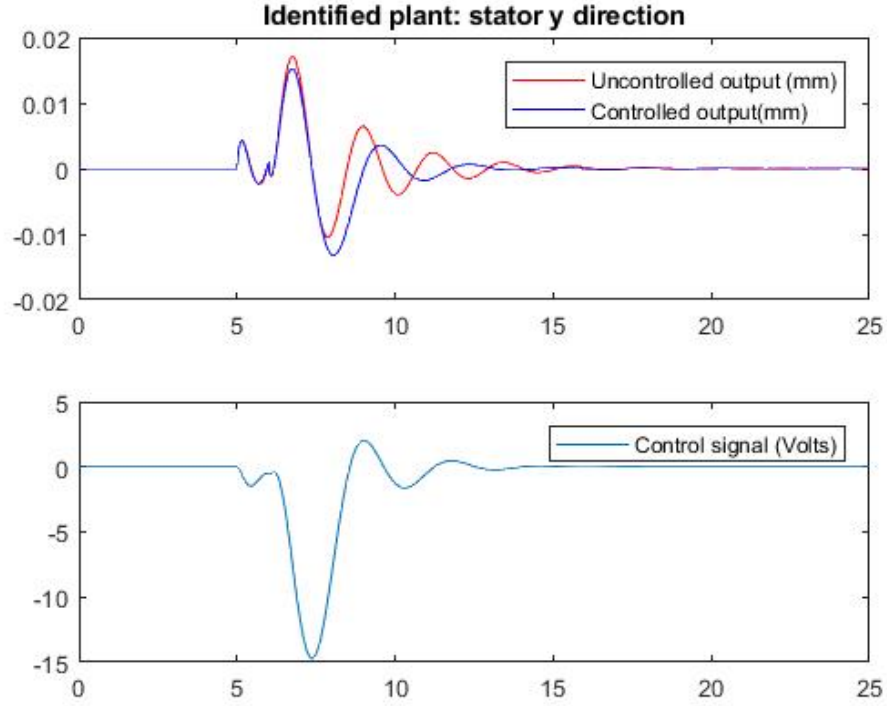


Figure 6.11. Comparison between closed and open loop response with designed PID for SISO system on stator in y direction

It can be seen in the plots that the closed loop response is faster and better in regulating, except for the SISO system of the stator in X direction, which has oscillating behaviour because of the plant dynamics, which are heavily oscillatory.

The MatLab scripts used to design the PID controllers are included in Appendix B. Now that the PID controllers are designed, the design of the rest of controllers will be performed, before their implementation and results, shown in the next chapter.

6.2 Full state feedback controller design

For the full state space feedback controller the system cannot directly be separated into stator and blades controllers. The plant has been estimated as a whole state space system, with 6 input magnet voltages and 6 output vibration measurements. However, to simulate the situation where the measurements and actuations of the blades are not available an observer is implemented. The observer estimates all the states of the system even if less measurements are available only if the system is observable. This observability analysis was also previously done, and resulted positive [8]. This is why a full state feedback controller and a full observer will be designed for the estimated system in static operation.

The controller to be designed is also a regulator, and has the same control objectives as the PID controllers, to have a good disturbance rejection. The disturbances added to test the controllers are the same as in the PID controllers, the ones shown in Figure 6.2. Those disturbances are added to the control signals as well.

The linear continuous observer estimates the states of the system with the inputs and outputs of the real system. The dynamics of the estimation error need to be faster than the real plant dynamics, and that is why the output error is given in feedback to the observer, building a full Luenberger observer and speeding up the convergence of the estimation error [20].

The block diagram of the basic Luenberger observer implemented is shown in Figure 6.12 below. The design of the observer is done using the *place* function in MatLab. This function allows to find the feedback gain that forces the closed loop system to have the specified eigenvalues in closed loop. Those eigenvalues have been chosen to be 10 times the eigenvalues of the estimated plant in open loop.

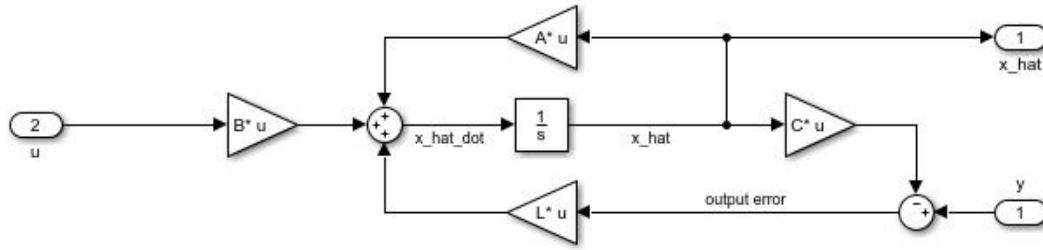


Figure 6.12. Block diagram of the implemented Luenberger observer

Once the observer is designed, the full state feedback controllers are designed. The optimal controllers are designed with the linear quadratic regulator approach, which finds the optimal control feedback gain that minimizes the following cost function:

$$J = \int_0^{\infty} (x^T Q x + u^T R u + 2x^T N u) \delta x$$

The solution to that optimization is the solution to the standard algebraic Ricatti equation. The weight matrices Q , R and N are user's choice, and they can be modified to give more or less optimization weight to one or several control signals. The Q matrix is chosen to be a diagonal order 10^6 matrix, to speed the convergence of the states since all states are wanted to be controlled, and the N matrix will always be set to the null matrix, since no cross correlation between states and input is wanted. However, for the choice of the weight of the control signals the design will be divided.

If both blades and hub can be controlled (coupled system), the R matrix will be set to the identity matrix, since no special focus is given to any control signal. But, in the case of the decoupled system, when only the hub's vibrations can be controlled, this matrix will be set so that the control signals for the hub's magnets have all control power for the entire plant and no control effort is wasted on unavailable actuators (blade's magnets).

The state feedback gain matrices are calculated with the MatLab function *lqr*. The disturbance is applied only to the magnet in blade 1 for an example of the performance of the controller. For both coupled and uncoupled systems the performance of the controllers

is compared to the response of the plant in open loop to the same disturbance. The results of the design are shown in Figure 6.13 and Figure 6.14 below, where the outputs of the system as well as the control signals are plotted.

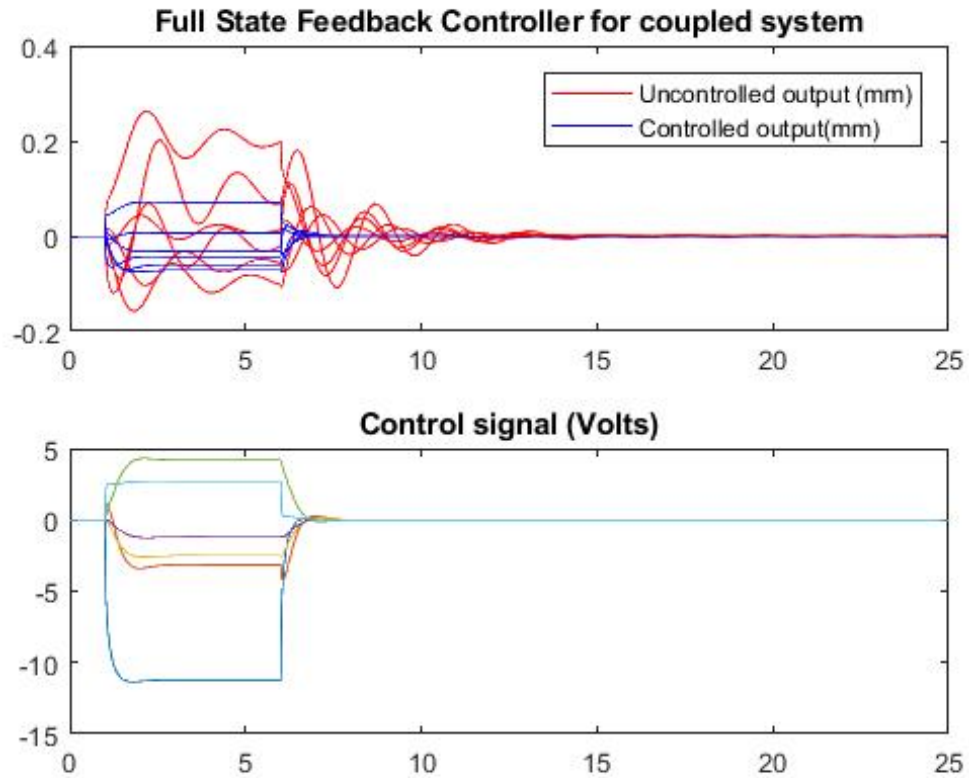


Figure 6.13. Full state feedback controller for coupled system

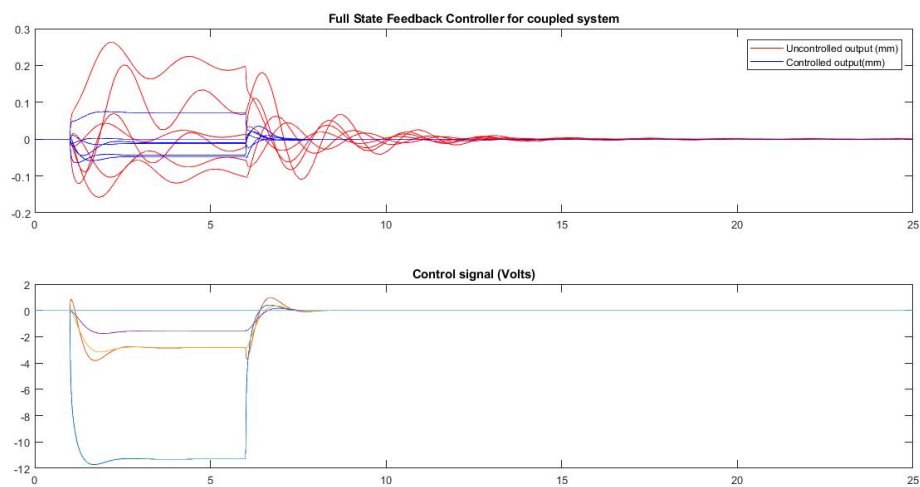


Figure 6.14. Full state feedback controller for uncoupled system

It can be seen in the designs that both coupled and uncoupled controllers can attenuate the disturbance vibrations of the outputs, but the uncoupled controller takes much more controller effort to do so. The MatLab scripts for the design of the LQR controller and observer are included in Appendix B. With both static controllers designed, the adaptive controller design comes next, to finalize with their implementation on the real plant, in the last chapter.

6.3 Adaptive controller design

Adaptive control is a control design technique that covers several approaches to automatically adjust controllers in real time by identifying the parameters of the plant to control each sampling period. This is specifically useful when the parameters of the plant to control are unknown or time-varying. For those cases, an adaptive controller provides an adequate level of control performance that no static controller can achieve, since either it can't be designed a priori (unknown plant parameters) or they only control the plant in one moment in time (time-varying plant parameters) [1]. The basic scheme for an adaptive control design is shown below in Figure 6.15.

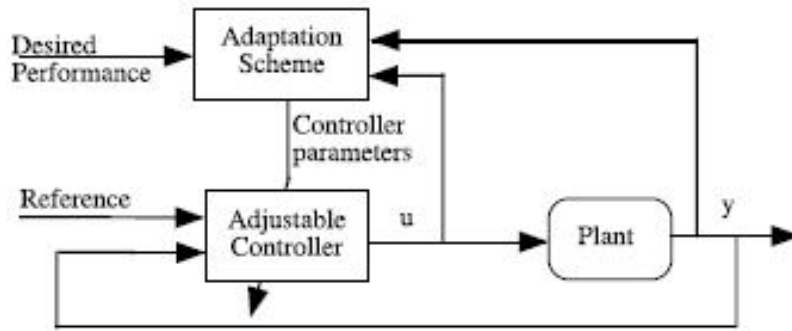


Figure 6.15. Block diagram of the basic structure of an adaptive control design [1]

In the structure shown above the adaptive controller updates the controller parameters depending on the measured inputs and outputs of the plant at each sample period. As in every control design problem, the reference for the system to follow and the desired closed loop performance are needed. The main difference between this scheme and the conventional feedback control is that in the feedback control the adaptation of the controller is only done once, designed a priori, and the controller parameters remain static during the whole operation of the plant. The dynamic model of the plant can be obtained in open or closed loop, depending on the accuracy of the identification wanted and the operation conditions of the plant [1].

The adaptive control problem can also be reformulated as a stochastic non linear control with incomplete information, since the parameters of the controller depend on measured inputs and output included in the adaptation loop. If the unknown parameters are considered as auxiliary states the control design becomes non linear, so the adaptive control

design problem can also be considered an approximation of a non linear stochastic control design [1].

For conventional feedback design controllers the design focus is on reference tracking and external disturbance rejection. This disturbance is an external unexpected change in the controlled variables. However, if an big unexpected change happens on the parameters of the plant (parameter disturbance), the conventional control problem is unable in most cases to reject this disturbance, since its parameters are static. This is a case where the adaptive controller scheme should be adopted. The other design problem approach is to measure a performance index which indicates the accuracy of the performance of the closed loop according to the desired performance. An adaptive controller measures that performance index at each sampling period, compares it with the desired one and feed that comparison to the adaptation mechanism to adjust the controller accordingly. The illustration of this mechanism is shown below in Figure 6.16.

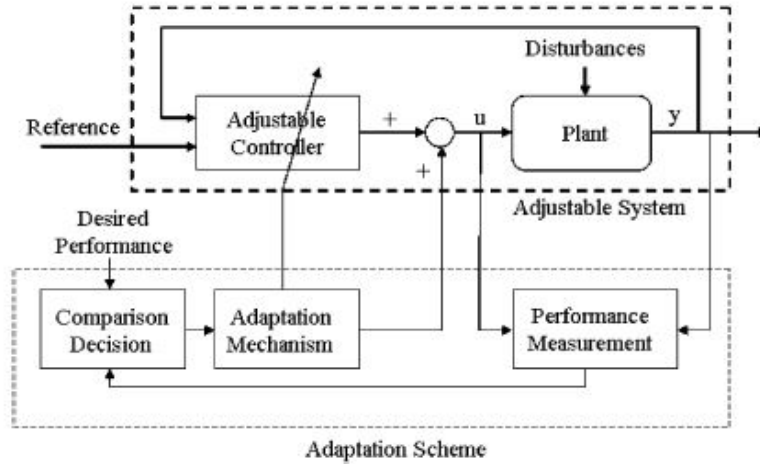


Figure 6.16. Block diagram of the basic structure of an adaptive control design with the adaptation scheme displayed[1]

For this specific application of this project, the controllers designed before are part of the conventional feedback controllers. In this system it has been analyzed [9] that the plant's parameters become time variant when the system is rotating, and a parameter disturbance is present in the system [8]. The design done previously has been performed on the identified plant in open loop. Therefore, the already designed controllers have been designed a priori on the measured inputs and outputs of the system in open loop and with an off-line identification procedure. Now, with the adaptive controller design, a different approach is taken, since the identification will be performed on-line and the controller parameters will be adjusted according to that identification.

Depending on the adaptation scheme adopted, different types of adaptive controller can be designed. The application of one type or another depends mainly on the availability of a reference model, the uncertainty of the plant's parameters or the accessibility of measurements for the outputs and control signals. For the system in this project, a reference model is available (static operation). However, its parameters vary in a wide

range, making the reference model inaccurate. The control signals and vibration output can be measured in this system. Those are the reasons why the indirect adaptive control model is chosen to be applied to this plant, since it has more flexibility (several algorithms for on-line identification and adjustable controller) and the plant needs to be identified, since it varies linearly with time in a wide range [1]. The block diagram of the indirect adaptive control model is shown in Figure 6.17 below, illustrating the different phases of the controller.

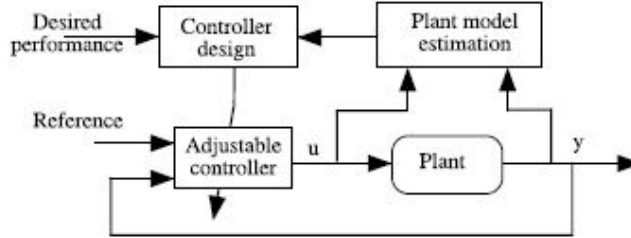


Figure 6.17. Block diagram of an indirect adaptive controller design[1]

With the type of adaptive controller chosen, the design is divided into its two phases: on-line plant model estimation and adjustable controller design. The specific adaptive controller to design is essentially a regulator, since the vibrations are wanted to be eliminated, so the input reference is null.

6.3.1 Online identification procedure

The online model estimation system uses the measured control signal and output at each sampling time and estimates the parameters of the plant with that information. The estimation seeks to make the estimation error $y(t) - \hat{y}(t)$ null and the convergence of its dynamics fast. A basic block diagram of the structure of the online estimator is shown in Figure 6.18 below.

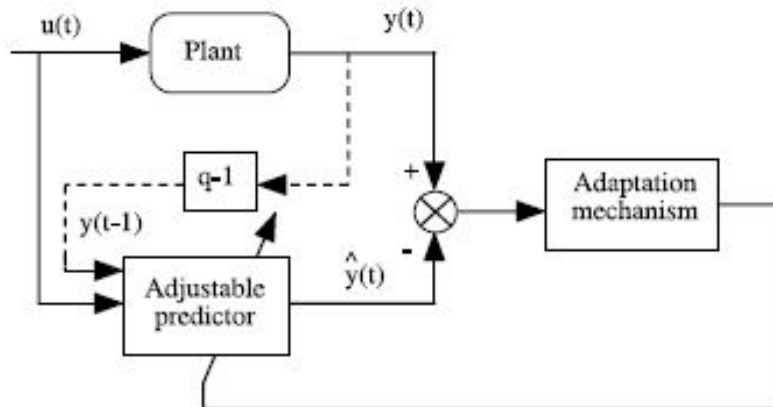


Figure 6.18. Block diagram of the basic online estimator functioning [1]

The estimated plant model parameters have to be assumed equal to the real parameters to calculate the controller. This recursive model identification follows the same steps as in the general system identification procedure, explained in Chapter 2. Those steps were illustrated in Figure 5.1. The main difference between off-line and online parameter estimation is the experimental data available for the estimation. For offline estimation, the experimental data had to be recorded a priori, and in large numbers, to be input in the identification algorithm. For online estimation, the only data available is the measured data in each sampling time and the saved past estimations, that is why a recursive method is used to estimate the plant's parameters in online estimation.

For any recursive estimation algorithm, the general equations of estimation are:

$$\hat{\theta}(t) = \hat{\theta}(t-1) + K(t)(y(t) - \hat{y}(t))$$

Where $\hat{\theta}(t)$ is the estimate of the parameters at time t , $y(t)$ is the measured output at time t and $\hat{y}(t)$ is the prediction of the output bases on the observations up to time $t-1$. The gain $K(t)$ determines how much the prediction error affects the update of the parameter estimate [13]. The different recursive estimation methods differ on how to update this matrix $K(t)$. The built estimator seeks that the dynamics of the prediction error converge fast to zero, to obtain an accurate estimation of the parameters. This algorithms only requires memory for storing the previous estimate of the model parameters, and therefore is more efficient in terms of memory and computational effort.

Other choice when performing the online model identification is whether to update the plant's parameters every sampling time or only every N sampling times. If the variability of the parameters is not significant within a time window of N samples, updating the estimation algorithm only every N samples reduces considerably the computational load and time. For a fast system in closed loop, this time reduction can be helpful to increase the complexity of the controller, or even to be able to close the loop in steady time [1]. However, for this model the time variability is linearly dependent with time, and with a high slope, and the plant's parameters need to be estimated each sampling time.

The choice for the online estimation algorithm is determined by the available functions in MatLab, since this is the platform used for the implementation of the controller. MatLab's System Identification Toolbox used previously on the open-loop identification in Chapter 2 also provides functions and Simulink blocks for performing online parameter estimation. For estimating state space models, the Recursive Polynomial Model Estimator and the Model Type converter block can be used together. The recursive polynomial model estimator algorithm is the one chosen for this application, since the MIMO system at hand can be identified as different ARX (Auto-Regressive Model with Exogenous Inputs) polynomial models. The basic description of an ARX structure is as follows:

$$A(t)y(t) = B(t)u(t) + e(t)$$

$$y(t) + a_1y(t-1) + \dots + a_my(t-m) = b_0u(t) + b_1u(t-1) + \dots + b_nu(t-n) + e(t)$$

Where $y(t)$ is the vector of outputs of the system, $u(t)$ is the vector of inputs of the system and $e(t)$ is the vector of noise measurements applied in each channel. This description is based on the polynomial parameters of the signals ($A_m(t)$, $B_n(t)$ and $C_p(t)$) which are estimated at each time [13]. The system at hand can be separated in different ARX models for every output, as has been done in Chapter 2 for the offline time variable parameter estimation. This is the reason why this model structure is the chosen one.

The second choice to be made is the estimation algorithm. The options available for the online estimation Simulink block are forgetting factor algorithm, Kalman filter and normalized and unnormalized gradients. The latter method is less computationally intensive, but has worst convergence properties than the first two methods. The Kalman filter method is chosen, since an initial estimation of the parameters is available (parameters of the plant for static operation), and with a good estimation of initial conditions this algorithm converges much faster [21]. The equations for the $K(t)$ update matrix are explained in detail in [21], and correspond to an updating Kalman filter.

The final requirements for this estimation method is that the model is discrete and linear. This is why a discretization of the model is performed, with a chosen sample time of $T_s = 0.001s$, the same sample time as in the data recording procedure for the offline identification. The Simulink block also requires that the structure of the model to estimate does not change in time, which doesn't for this application.

To test the implementation of this online identification algorithm on the simulated plant the Simulink block diagram shown in Figure 6.19 below is built.

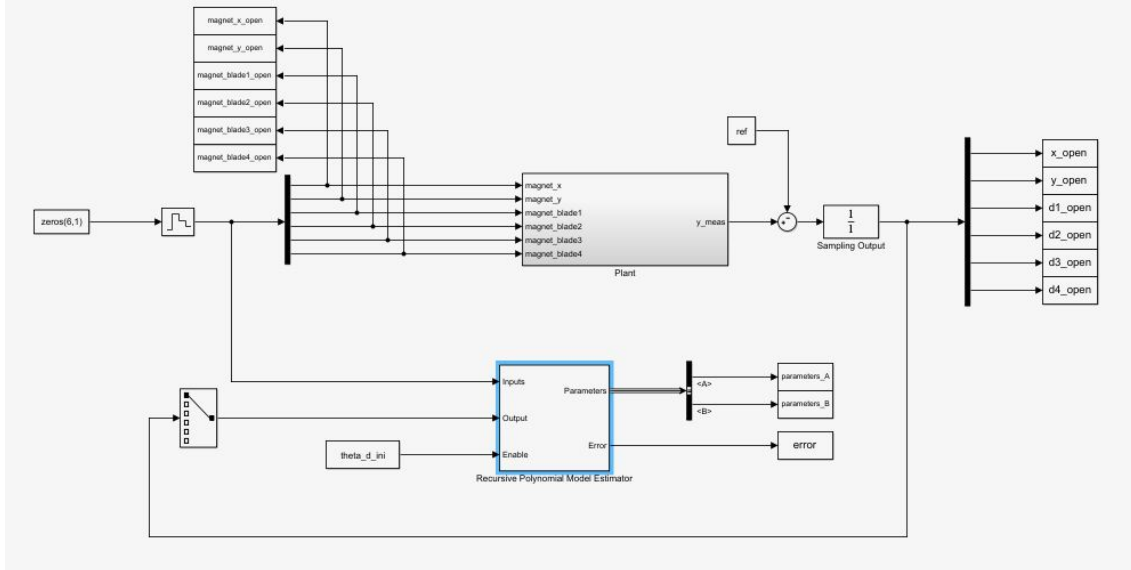


Figure 6.19. Block diagram of the built online estimator with the simulated plant [1]

The online identification block also needs as user choice the order of the model polynomials, the initial estimation of the polynomials (static operation linearized), and a previous discretization of the static system is performed in order to obtain this model order. The order of each polynomial to estimate, as well as the input delays are calculated and given

as inputs in the online estimation block. The initial estimation of the parameter covariance matrix is set to a high value since the initial estimation is highly unlikely to be similar to the real values. The plant has a white noise signal as an input in order to have persistently exciting input for this open loop test. Since the real noise in the system has been proved to be small, no importance is given to the noise model and it is not included in the test of the online identification subsystem.

The prediction error for every MISO subsystem defined is shown in Figure 6.20 below.

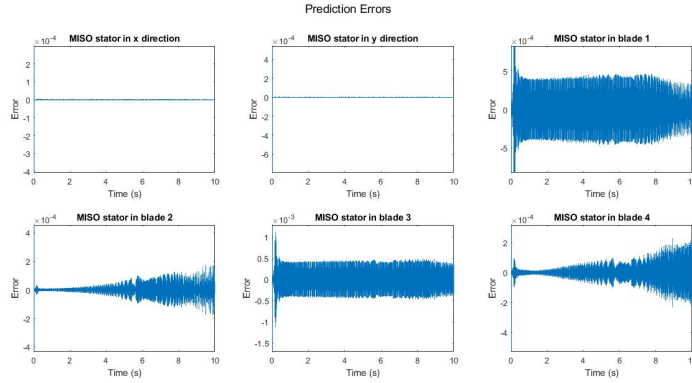


Figure 6.20. Prediction error of the online estimation of every MISO subsystem of the plant

Also, an example of the time variability of the plant's parameters is shown in Figure 6.21. In the figure, the second parameter of the A polynomial (system dynamics) of the estimated MISO system for the stator output in x direction is illustrated.

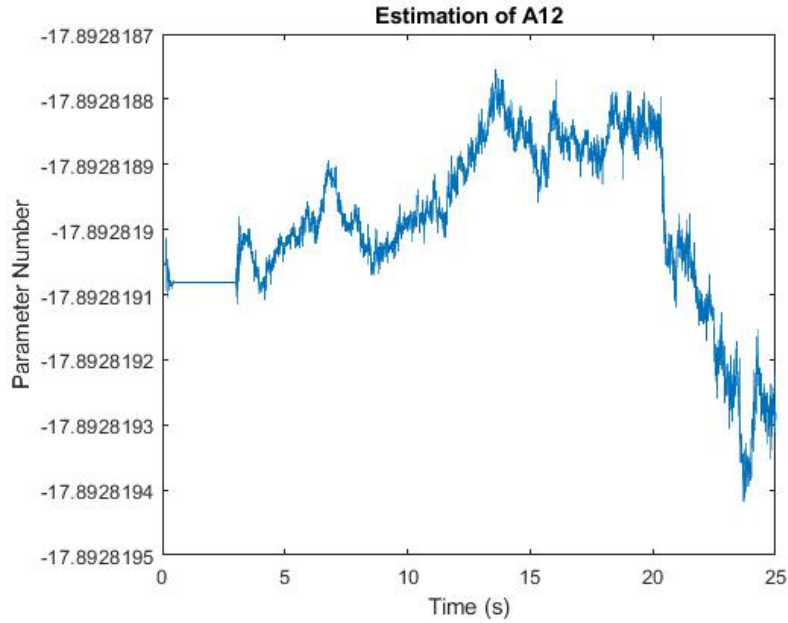


Figure 6.21. Time variability of the second parameter in the A polynomial for the first MISO system estimated

It can be seen in the plots that the online system identification is successful, since all prediction errors remain close to 0 (order 10^{-4}) in steady state. The method can also estimate the periodic time variability of the system's parameters. This means that this algorithm can be used for the adaptive controller to estimate online the plant parameters. All the MatLab code used is included in Appendix B. Now, the adjustable controller is designed.

6.3.2 Adjustable controller design

Once the plant can be identified each sampling time online, the parameters of the estimated plant have to be used to design a controller. To convert the estimated ARX parameters to a state space model that can be controlled the Simulink block Model Type Converter is used. After having built the model plant at each sampling time, the choice and design of the controller is performed.

For the choice of the controller, the PID and state space controllers are evaluated. For the static operation showed above, the LQR full state feedback controller had better performance, since the coupling between crossed input/output channels is considered and the controller effort is optimized. This controller design needs an estimate of the states of the system, which can be done with a Luenberger observer, as done previously in the static operation case. The same equations as in the design for the static model are implemented, but each sampling time and with the each identified plant. The same controller weights are chosen as in the static case, dividing also into coupled or decoupled system. The MatLab functions built that calculate the control takes the state space matrices as input and outputs the observer gain L and the controller gain K . This information is then input in a discrete time varying Luenberger observer block available in Simulink, which estimates the states of the system each sampling time. Those states are then given as feedback into the real plant.

A block diagram of the implementation of the adjustable controller is shown in Figure 6.22 below.

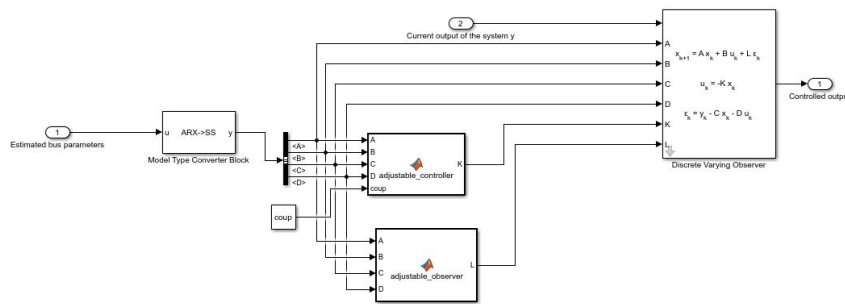


Figure 6.22. Block diagram of the adjustable controller designed

The test of its performance is done in the following chapter, under the implementation of the adaptive controller section and along with the online identification.

CHAPTER 4: CONTROL IMPLEMENTATION

7 AND RESULTS

After the different controllers have been designed in the previous chapter, an implementation and performance analysis is done. The PID and LQR controllers have been designed in the identified static plant, and they are implemented in the simulated plant explained before in Chapter 1. The adaptive controller is directly applied to the simulated plant. This simulated plant represents the real system, and models its dynamics in an accurate way, being a good description of the test rig.

7.1 PID controller implementation

For the PID controller implementation the block diagram shown in Figure 7.1 has been built in Simulink with the parameters of the previously designed controllers. The flag parameter allows to select whether the control is coupled (controlling blades and hub) or uncoupled (controlling only the hub). The subsystem nominated as Plant is the system's model explained in Chapter 1.

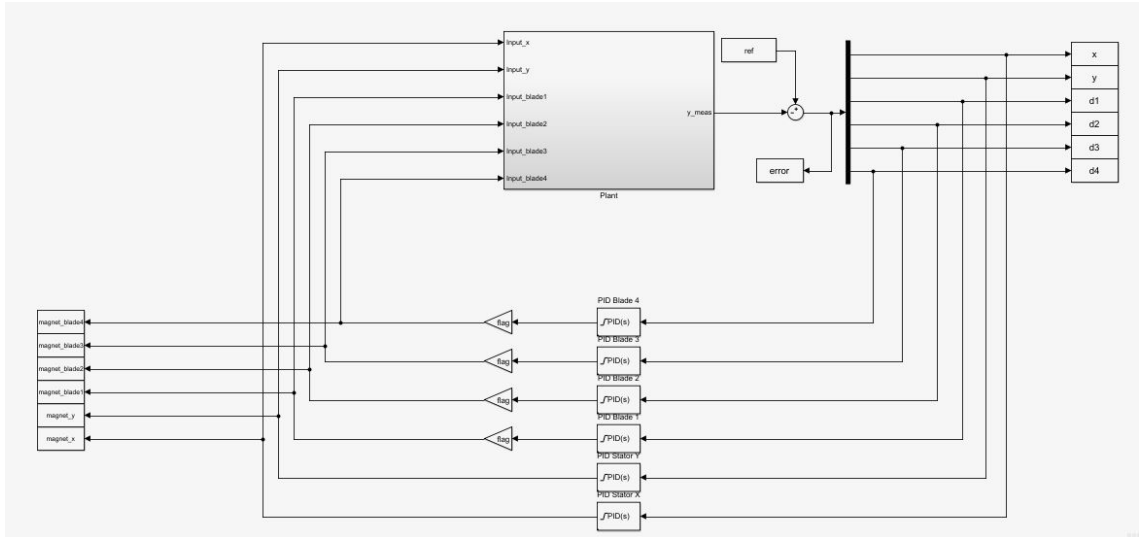


Figure 7.1. Block diagram of the implementation of the PID controllers

To test the controller in static operation a disturbance has been applied to different input channels. Since in the simulation model the plant's equations are accessible, a force disturbance can be applied, and it models a real disturbance better than applying it in the control signal. That force disturbance has been chosen to be of magnitude 20 N and same shape as the step disturbances in the design phase, enough to alter the system and cause

an output response. For rotating operation no additional disturbance has been applied to test the controller, since the output of the system is already affected by the rotation. The flag named *flag - d* allows to select whether to apply the disturbance in the input and in which input channel. The MatLab script used to test and plot the result of the controllers is included in Appendix C.

The implementation is now divided in static and rotating operations.

7.1.1 Static operation

First, the coupling of the PID controllers designed in static operation will be tested. The controllers will be tested against the force disturbance explained before, applied in a different input channel each time.

For the coupled system the output and control signals with disturbance applied in the corresponding input channel are shown in Figure 7.2, Figure 7.3, Figure 7.4, Figure 7.5, Figure 7.6 and Figure 7.7 below. The output for the open loop system with the same disturbance applied is also shown in the plots.

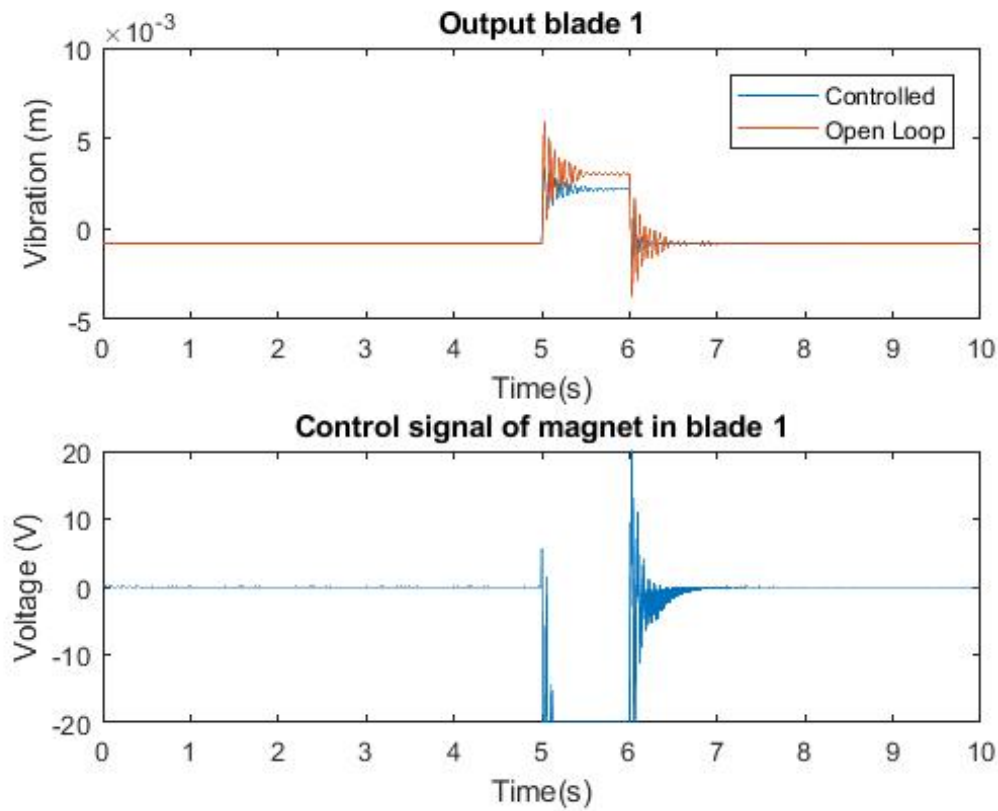


Figure 7.2. Vibration measurement and control signal of blade 1 under PID controller with coupled system

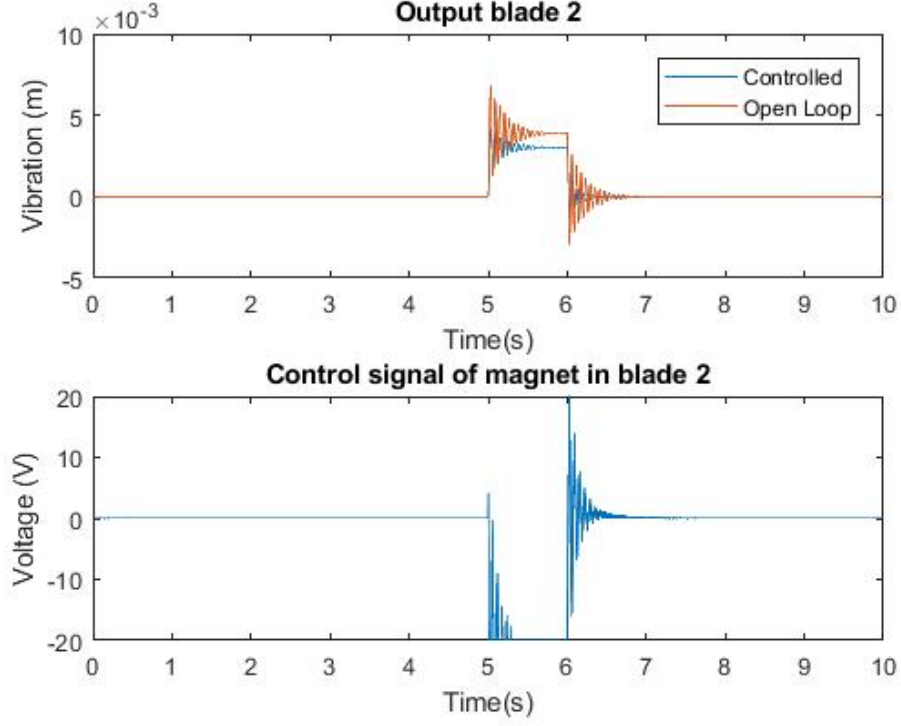


Figure 7.3. Vibration measurement and control signal of blade 2 under PID controller with coupled system

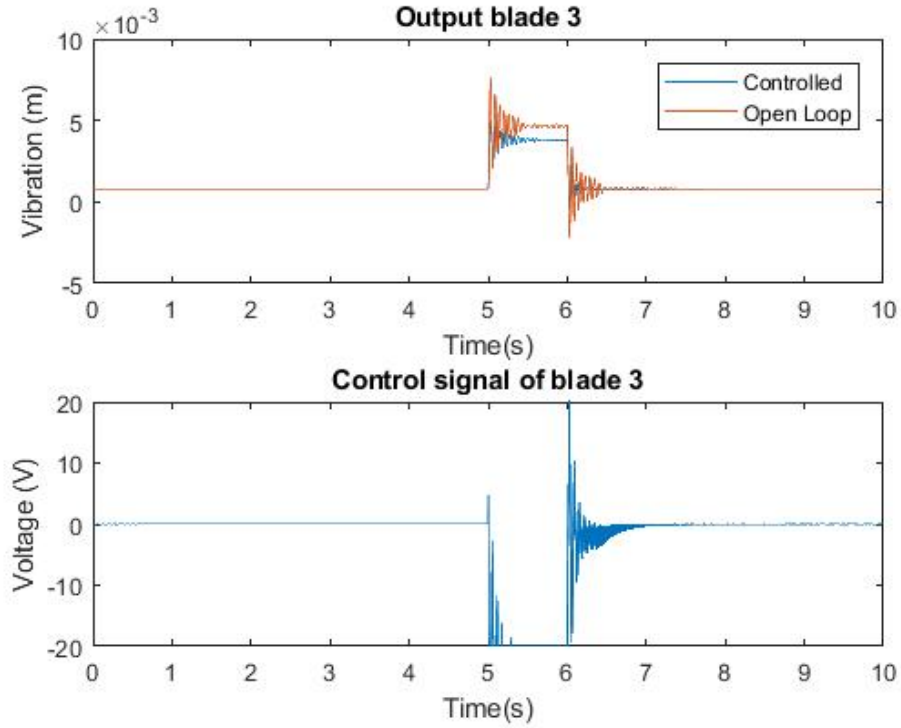


Figure 7.4. Vibration measurement and control signal of blade 3 under PID controller with coupled system

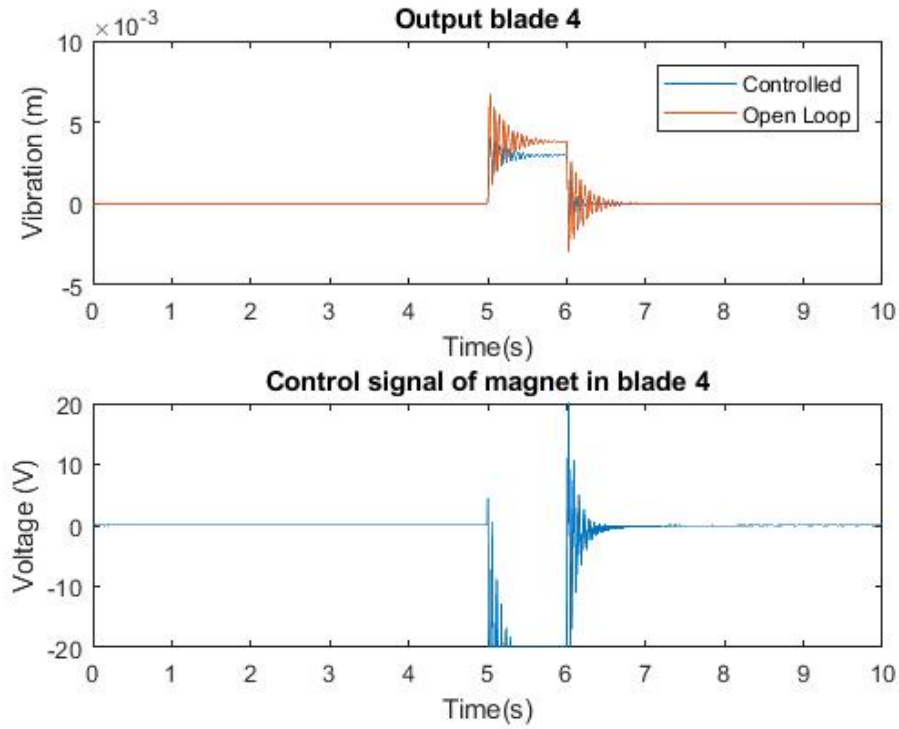


Figure 7.5. Vibration measurement and control signal of blade 4 under PID controller with coupled system

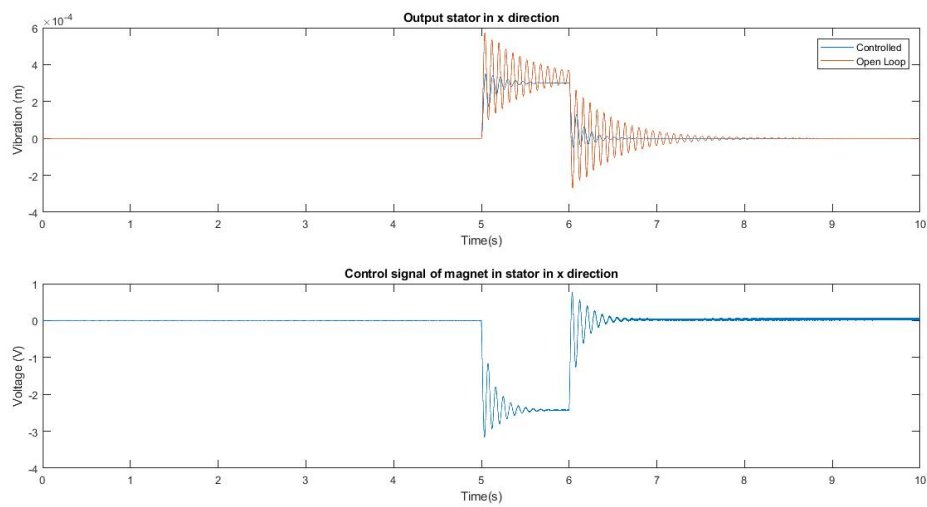


Figure 7.6. Vibration measurement and control signal of stator in x direction under PID controller with coupled system

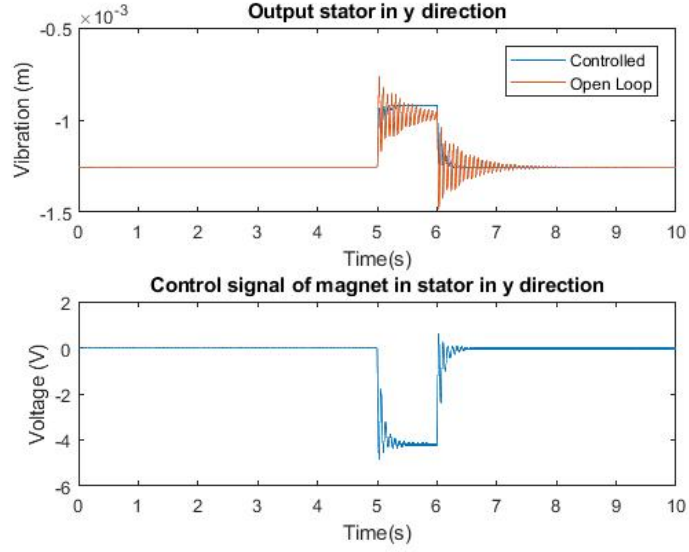


Figure 7.7. Vibration measurement and control signal of stator in y direction under PID controller with coupled system

As it can be seen in the plots, the PID controllers can dampen the force disturbance, but cannot eliminate it completely. This is due to the complexity of the plant, the coupling between crossed inputs and outputs appears to be much bigger than estimated at first. This makes the design of individual PIDs for every input/output channel ineffective.

For the uncoupled system the disturbance is only applied in the blades input channels and only controlling the hub in both directions. The measured outputs are shown below in Figure 7.8, Figure 7.9, Figure 7.10, Figure 7.11, Figure 7.12 and Figure 7.13. The output for the open loop system with the same disturbance applied is also shown in the plots.

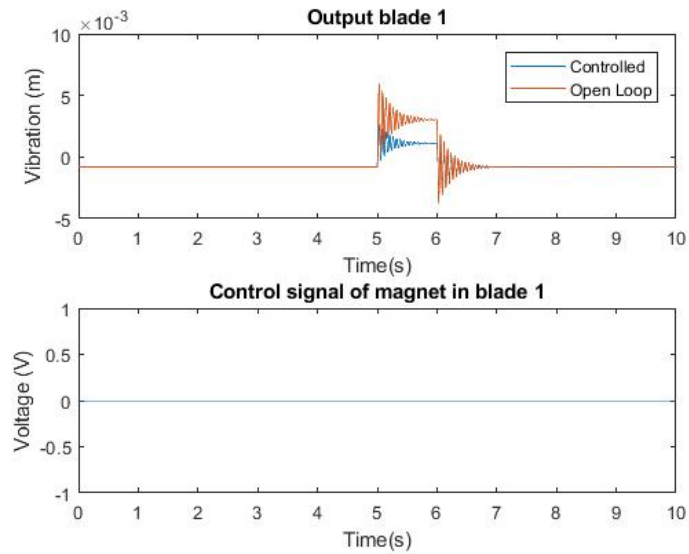


Figure 7.8. Vibration measurement and control signal of blade 1 under PID controller with uncoupled system

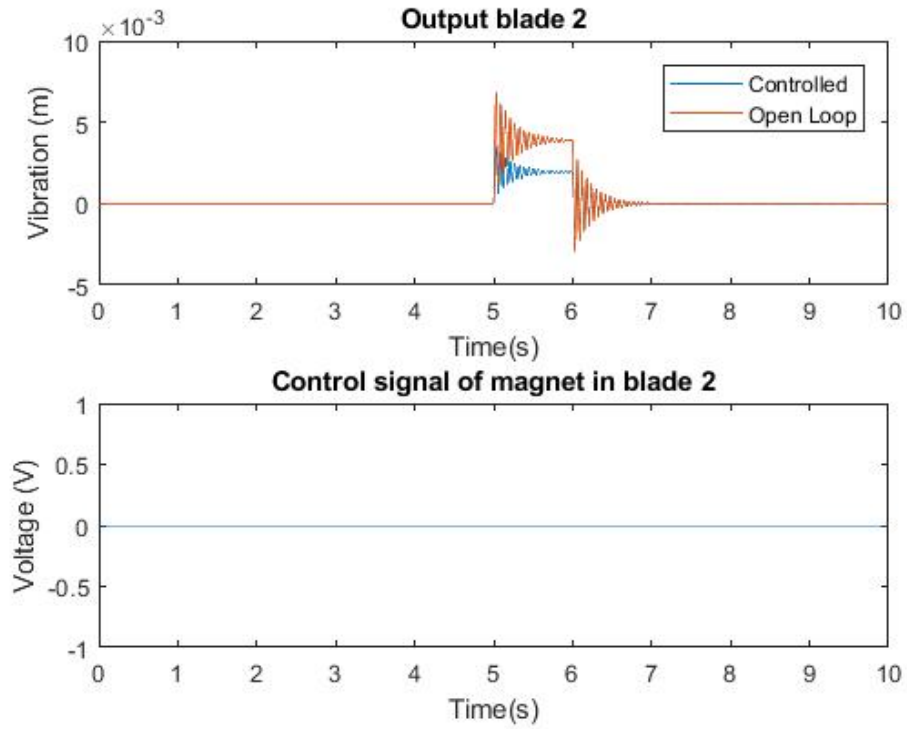


Figure 7.9. Vibration measurement and control signal of blade 2 under PID controller with uncoupled system

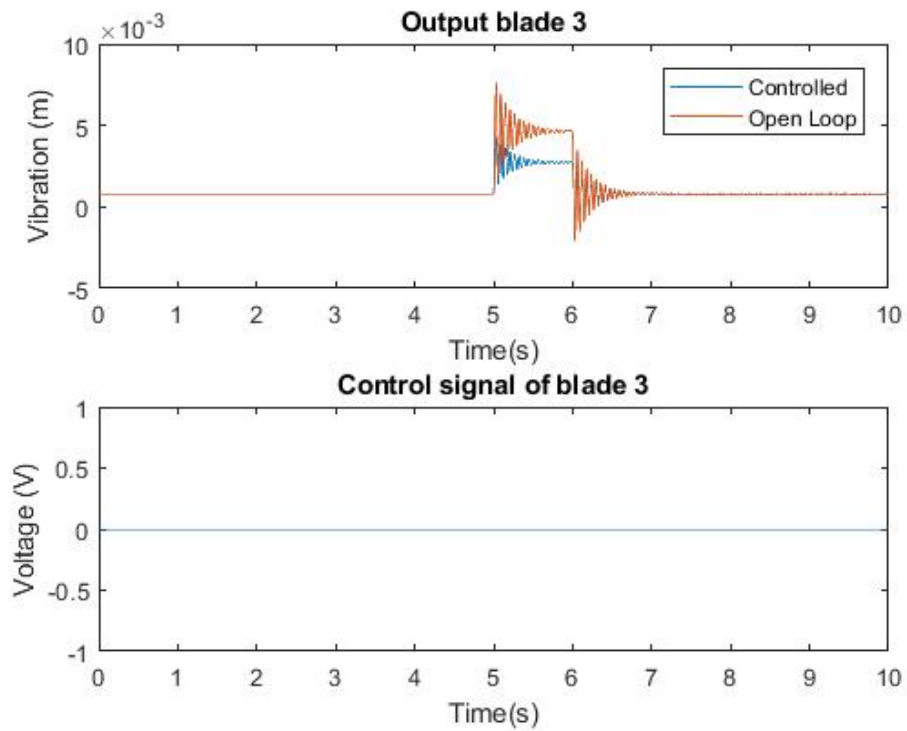


Figure 7.10. Vibration measurement and control signal of blade 3 under PID controller with uncoupled system

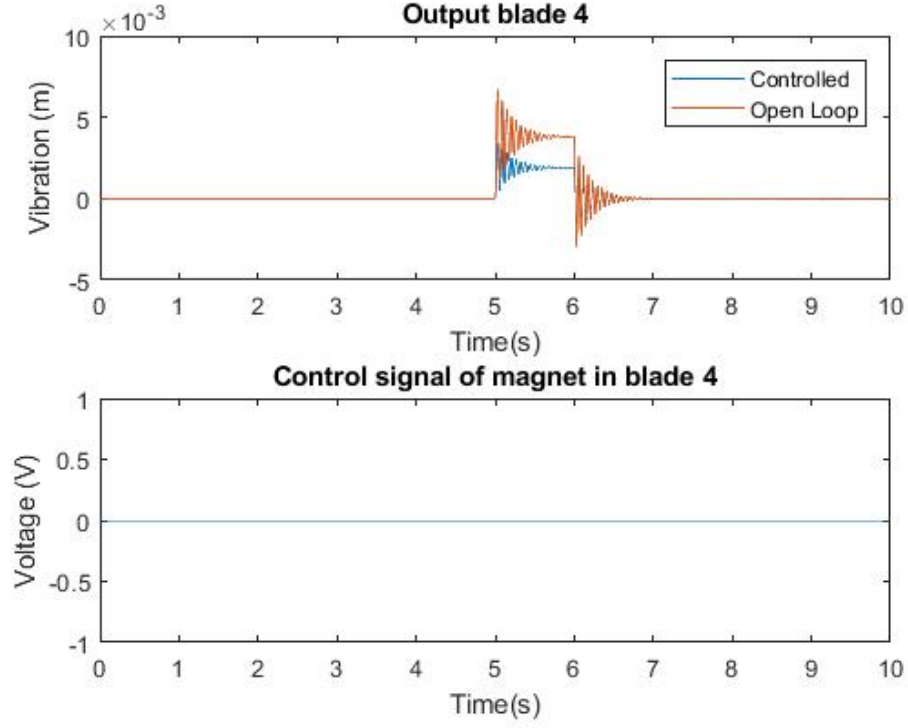


Figure 7.11. Vibration measurement and control signal of blade 4 under PID controller with uncoupled system

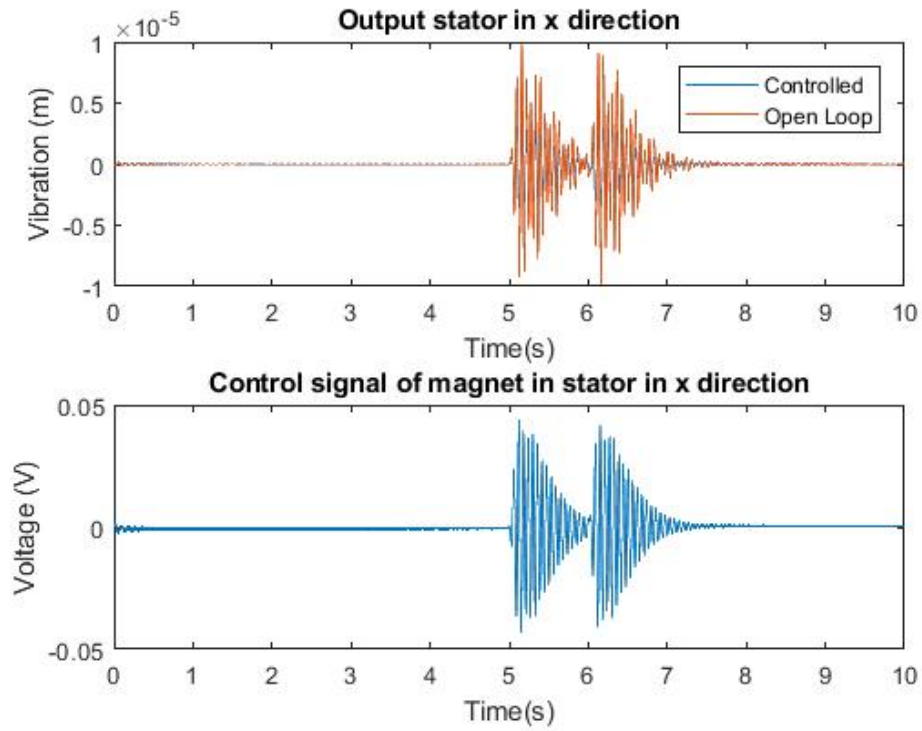


Figure 7.12. Vibration measurement and control signal of stator in x direction under PID controller with uncoupled system

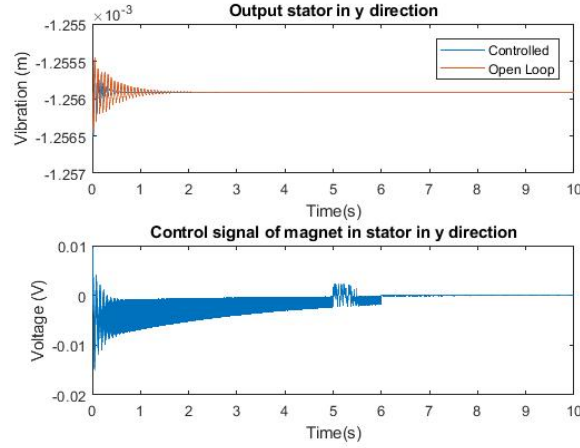


Figure 7.13. Vibration measurement and control signal of stator in y direction under PID controller with uncoupled system

As it can be seen in the plots, the disturbance is dampened similar to the coupled system but not entirely, also because of the correlation between crossed inputs and outputs. As was argued before, the individual controller design is ineffective for this plant, even in static operation, in the presence of disturbances.

After the analysis of the behaviour of the system in static operation, the controllers are tested with the system in rotation.

7.1.2 Rotating operation

For the application of the controllers in rotating operation a rotating speed of 5 Hz has been applied to the system. No other external disturbances have been applied, since only the rejection of the rotation's influence is sought. The outputs for the controlled system, the open loop system and the control signals are shown below in Figure 7.14, Figure 7.15, Figure 7.16, Figure 7.17, Figure 7.18 and Figure 7.19.

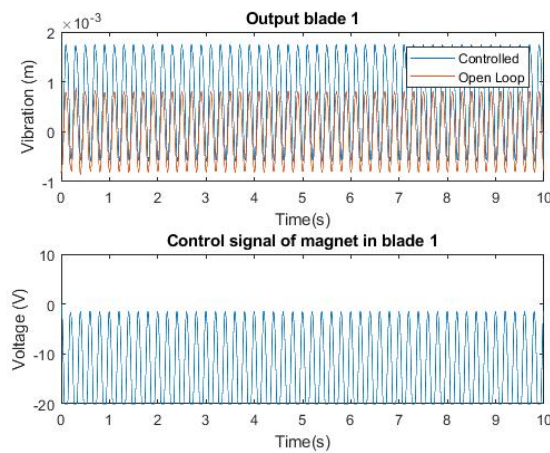


Figure 7.14. Vibration measurement and control signal of blade 1 under PID controller with rotating system

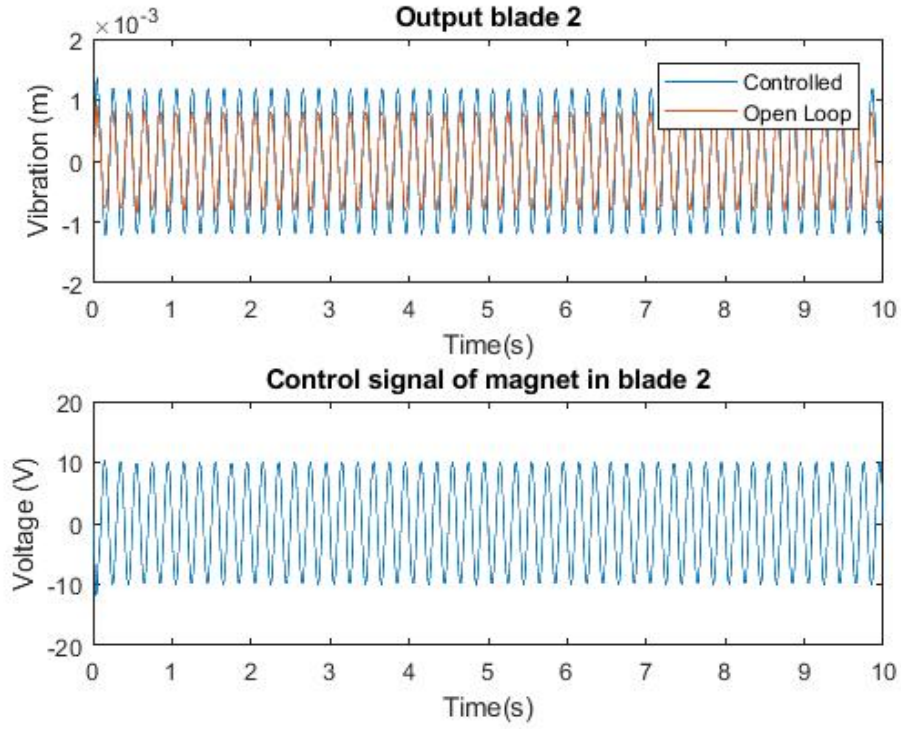


Figure 7.15. Vibration measurement and control signal of blade 2 under PID controller with rotating system

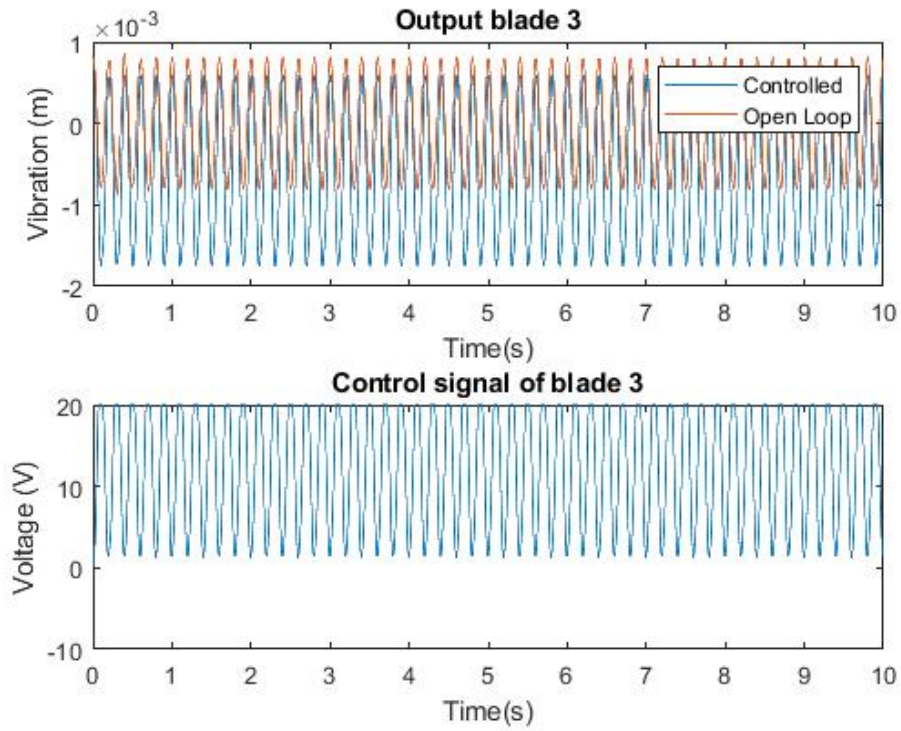


Figure 7.16. Vibration measurement and control signal of blade 3 under PID controller with rotating system

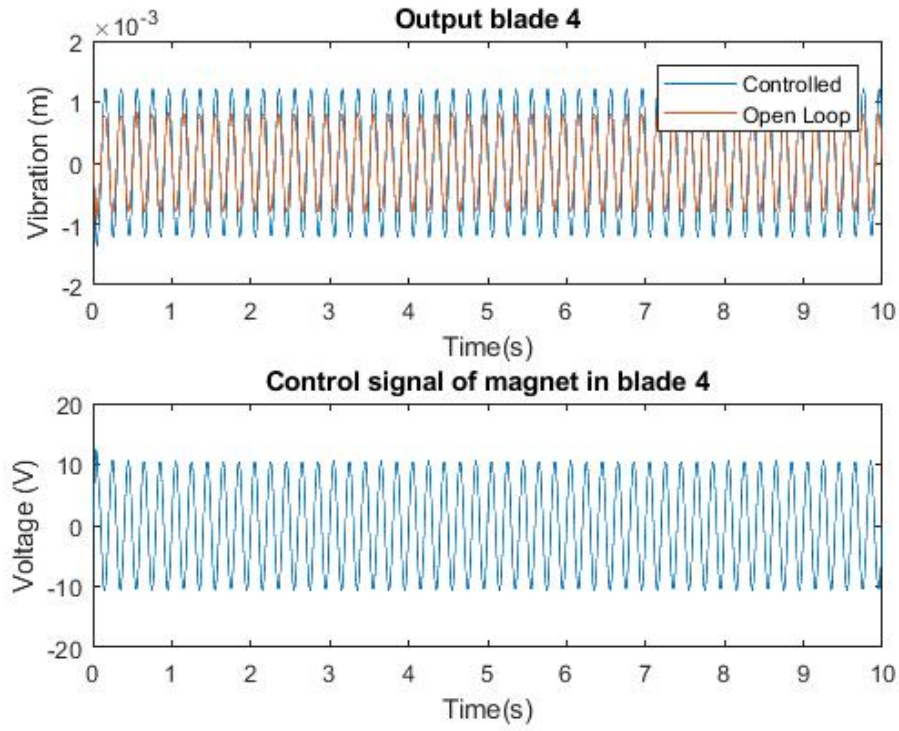


Figure 7.17. Vibration measurement and control signal of blade 4 under PID controller with rotating system

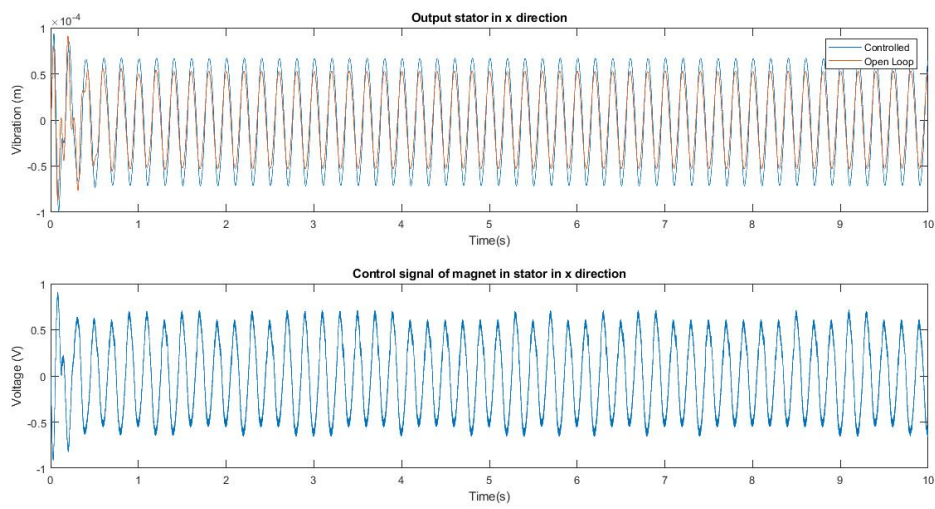


Figure 7.18. Vibration measurement and control signal of stator in x direction under PID controller with rotating system

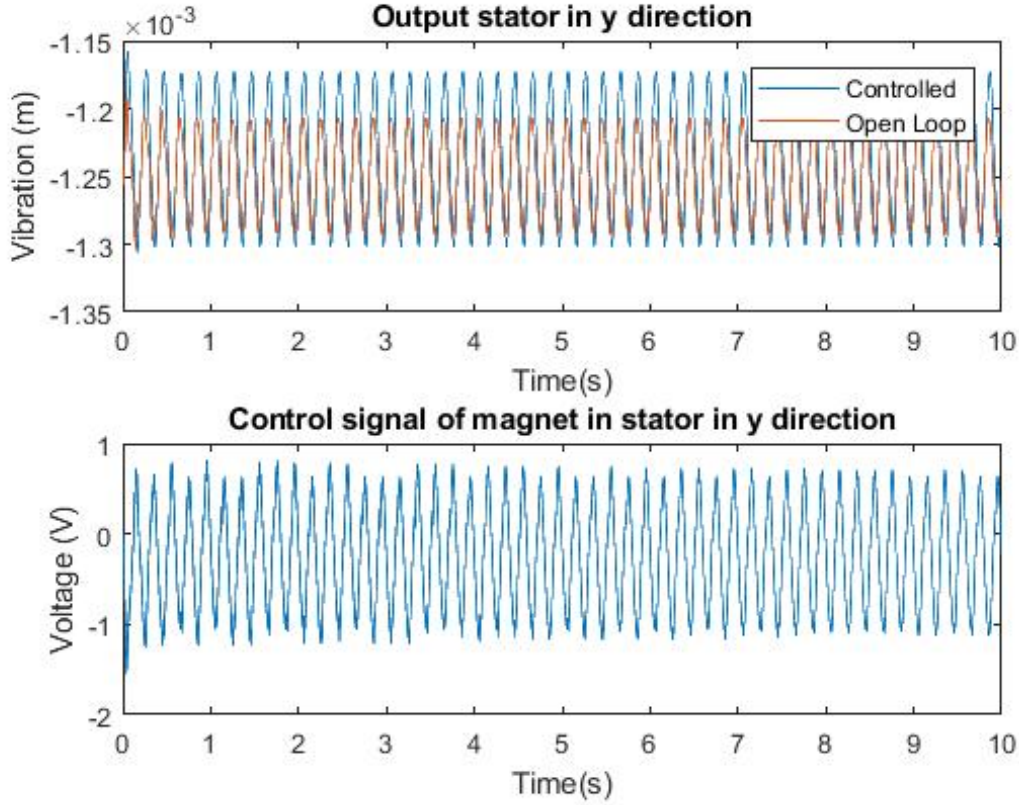


Figure 7.19. Vibration measurement and control signal of stator in y direction under PID controller with rotating system

The frequency vibration added by the rotation on the measured outputs cannot be dampened by the PID controllers, as it is seen in the figures. Even so, it is sometimes increased, since the delay of the controller inputs a force that increases the vibration in the next period.

It can be concluded from the figures that the PID cannot reject the vibration caused by the rotation of the system effectively, therefore a new control strategy should be used to dampen the vibrations of the blades.

7.2 Full state feedback controller implementation

The full state feedback controller has been implemented as shown in the block diagram in Figure 7.20 below. The same Luenberger observer designed for estimating the states when designing the LQR is used. This observer estimates the states of the identified plant, which may not be the same as the simulation plant's states. Therefore, the designed observer is needed to estimate the state to feedback with the calculated gains. Since the simulation plant and the identified plant have a different order of inputs and outputs they need to be arranged, which is done in the block diagram. The subsystem consisting of the LQR controller and observer is shown in Figure 7.21. The MatLab script used for the implementation of the state feedback controller is included in Appendix C.

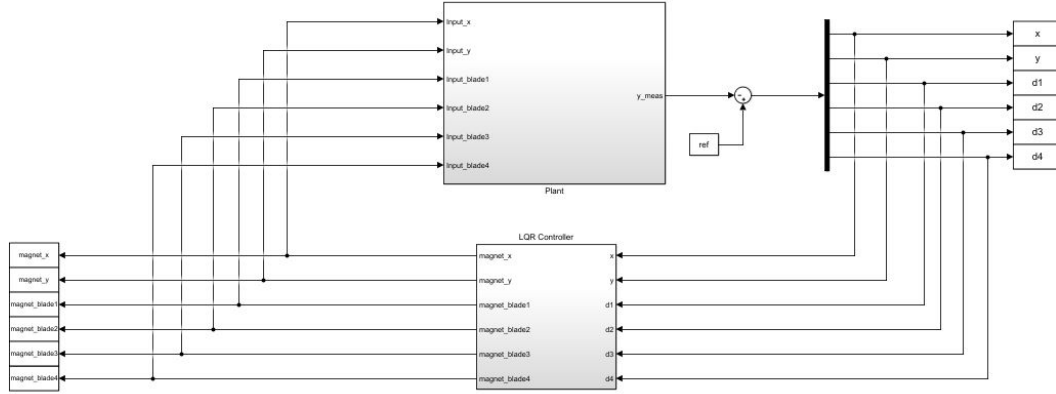


Figure 7.20. Block diagram of the implementation of the full state feedback controllers

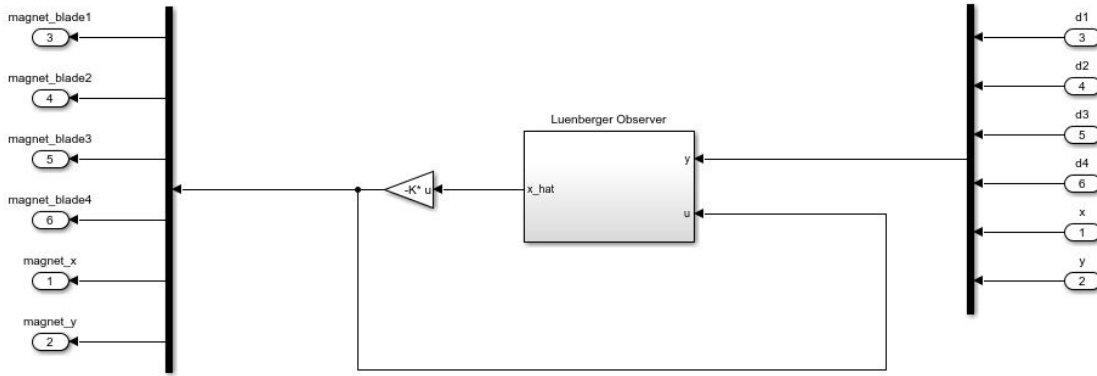


Figure 7.21. Block diagram of the subsystem with controller and observer

The implementation is also divided into static and rotating operation.

7.2.1 Static operation

The same disturbances as in the PID controllers implementation are used to test the controller in static operation, applied to different inputs. The measured outputs and the control signals of each input/output channel for the coupled system (controlling both blades and hub) are shown below in Figure 7.22, Figure 7.23, Figure 7.24, Figure 7.25, Figure 7.26 and Figure 7.27. The output for the open loop system with the same disturbance applied is also shown in the plots.

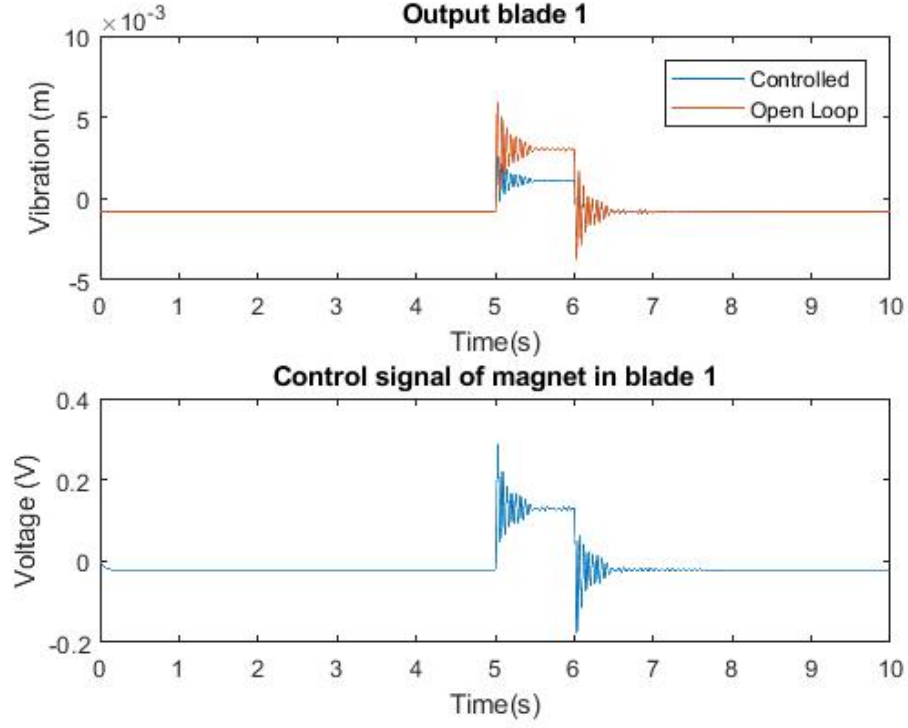


Figure 7.22. Vibration measurement and control signal of blade 1 under LQR controller on coupled system

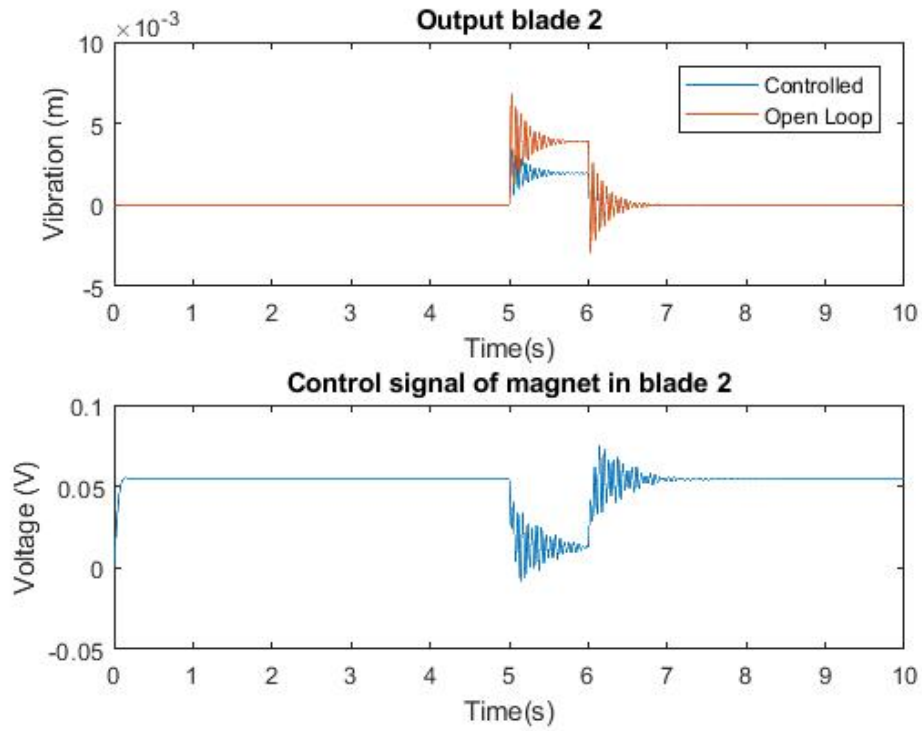


Figure 7.23. Vibration measurement and control signal of blade 2 under LQR controller on coupled system

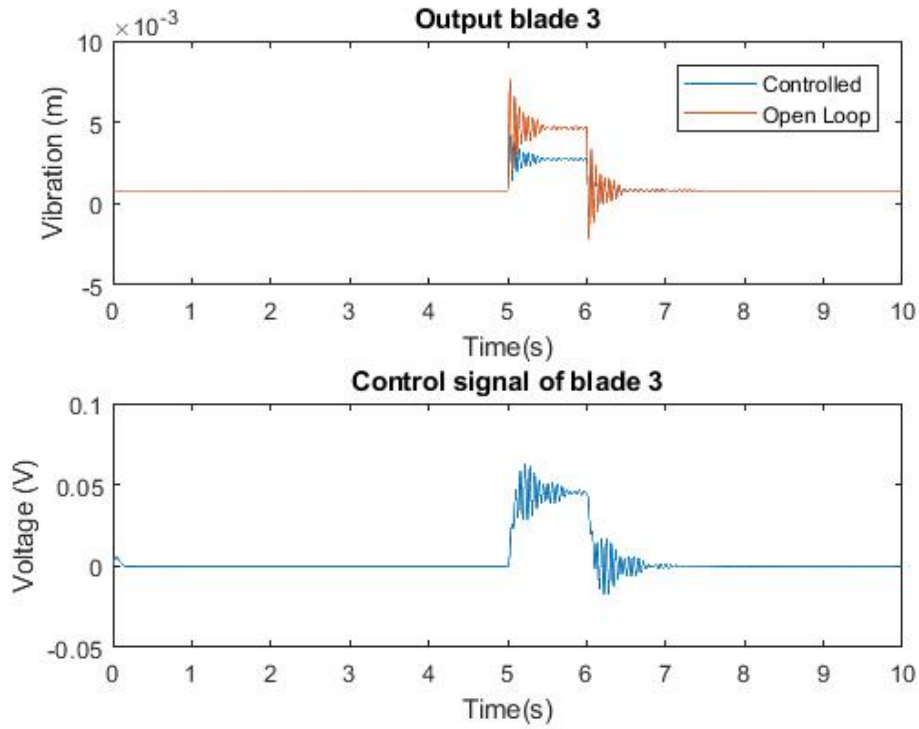


Figure 7.24. Vibration measurement and control signal of blade 3 under LQR controller on coupled system

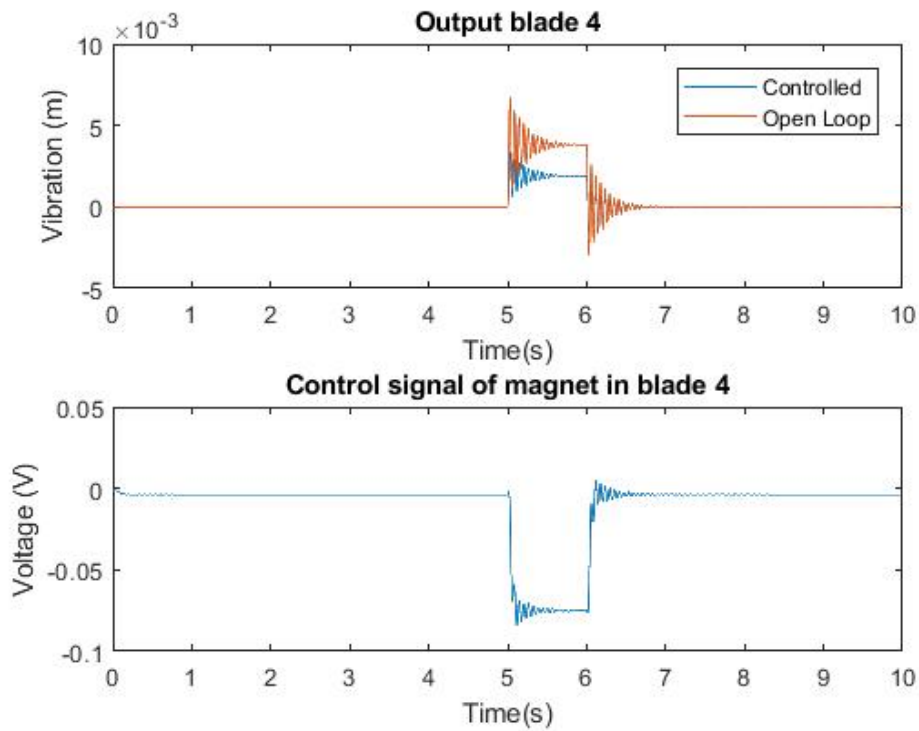


Figure 7.25. Vibration measurement and control signal of blade 4 under LQR controller on coupled system

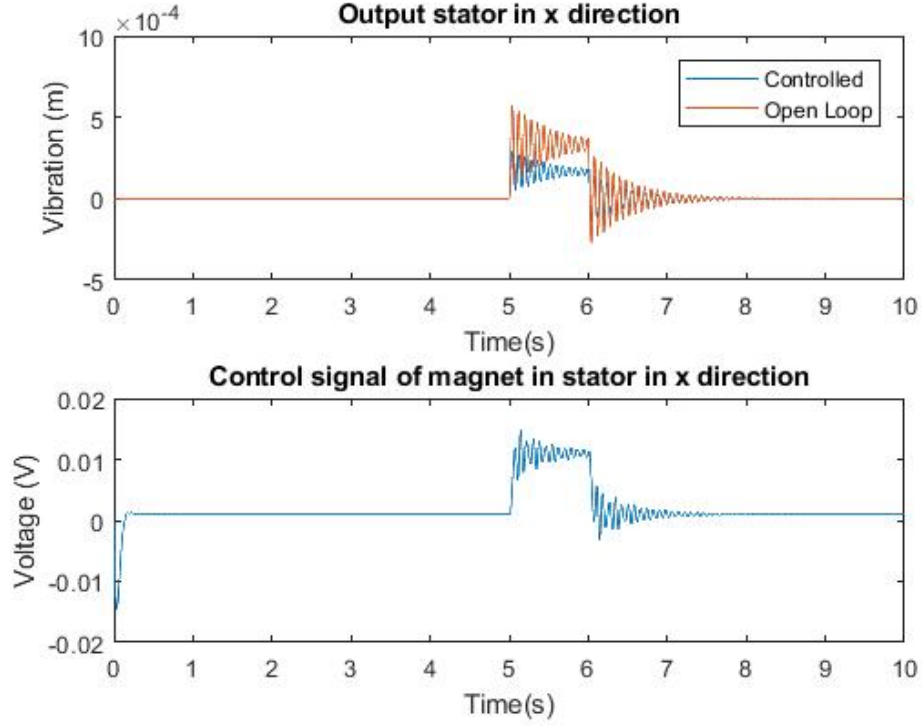


Figure 7.26. Vibration measurement and control signal of stator in x direction under LQR controller on coupled system

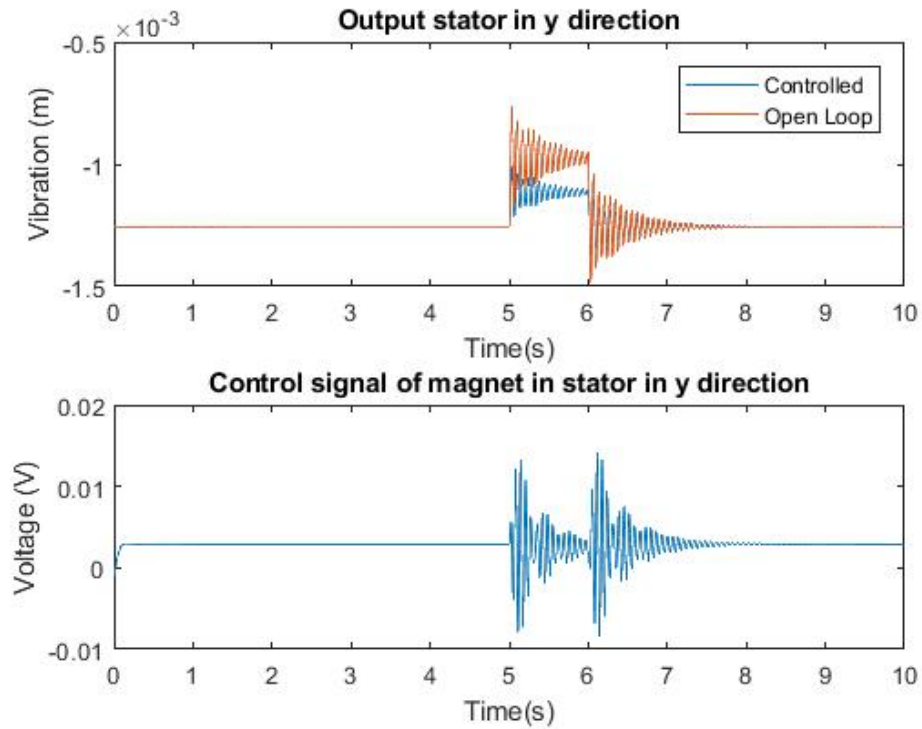


Figure 7.27. Vibration measurement and control signal of stator in y direction under LQR controller on coupled system

As it can be seen in the plots the output measurements are dampening the disturbances applied much better than the PID controllers, with some small offset and with less control effort, since the control effort has been optimised.

The measured outputs and the control signals of each input/output channel for the uncoupled system (controlling only the hub) are shown below in Figure 7.28, Figure 7.29, Figure 7.30, Figure 7.31, Figure 7.32 and Figure 7.33. The disturbance is only applied on the blades inputs, simultaneously. The output for the open loop system with the same disturbance applied is also shown in the plots.

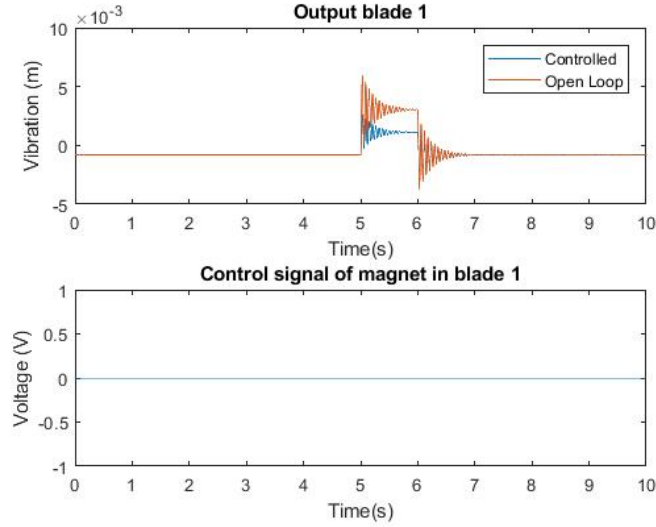


Figure 7.28. Vibration measurement and control signal of blade 1 under LQR controller on uncoupled system

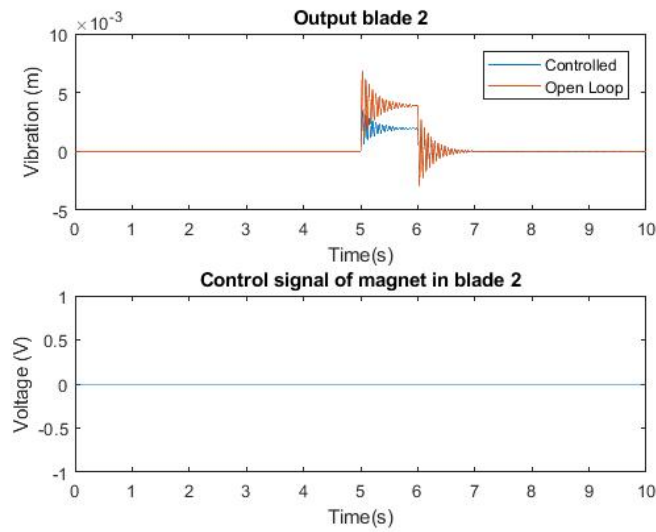


Figure 7.29. Vibration measurement and control signal of blade 2 under LQR controller on uncoupled system

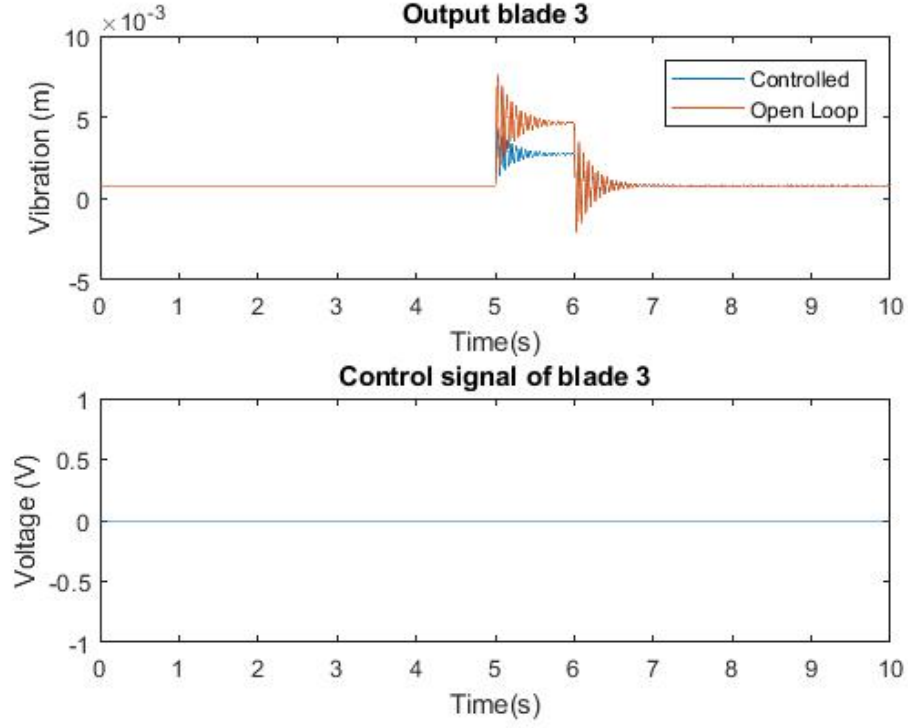


Figure 7.30. Vibration measurement and control signal of blade 3 under LQR controller on uncoupled system

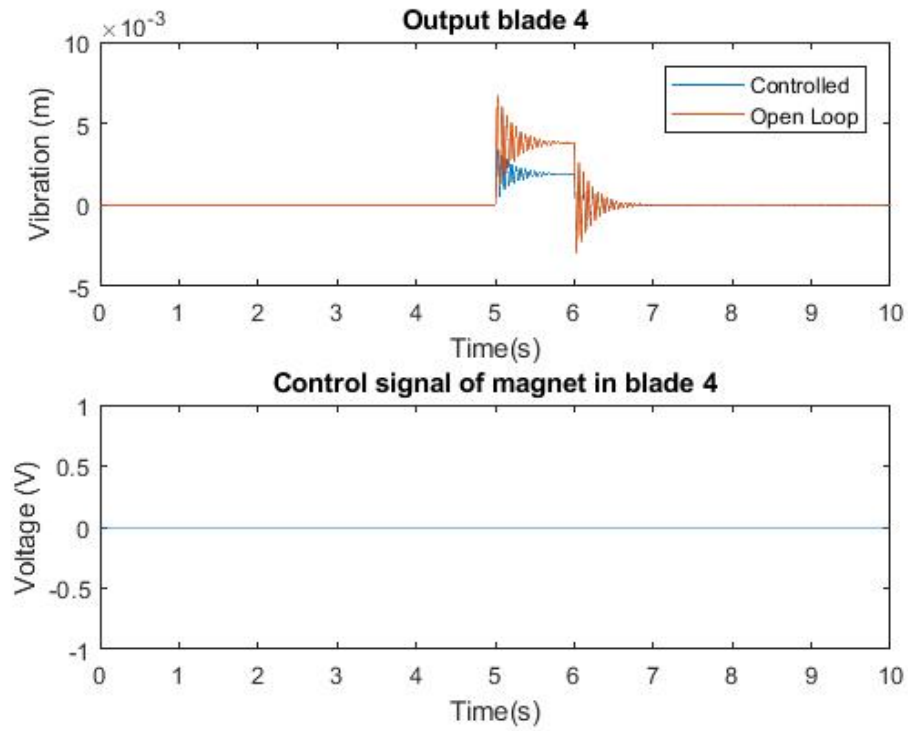


Figure 7.31. Vibration measurement and control signal of blade 4 under LQR controller on uncoupled system

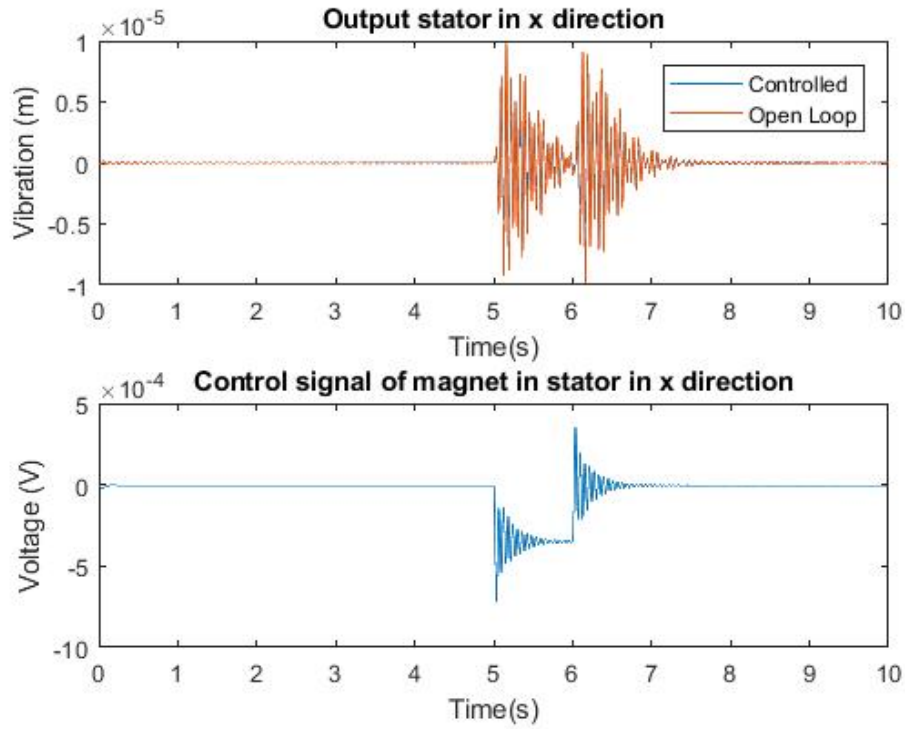


Figure 7.32. Vibration measurement and control signal of stator in x direction under LQR controller on uncoupled system

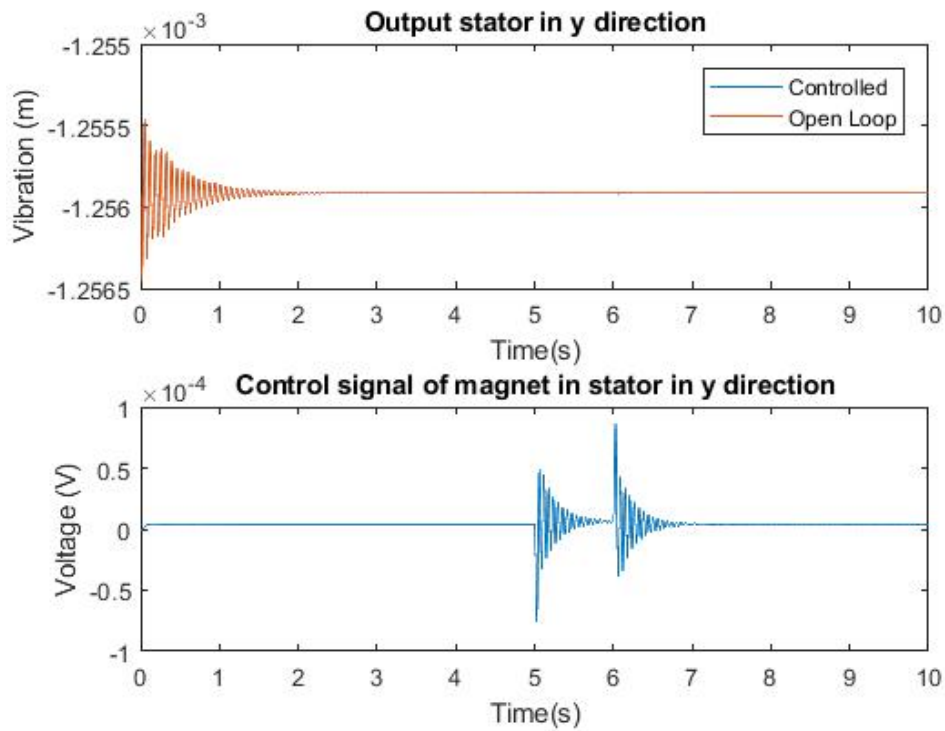


Figure 7.33. Vibration measurement and control signal of stator in y direction under LQR controller on uncoupled system

As it can be seen in the plots, the disturbance is rejected quite accurately with little control effort and almost as good as in the coupled situation. This is a prove of the optimization performance when designing the controller.

7.2.2 Rotating operation

For the application of the full state feedback controller in rotating operation also a rotation speed of 5Hz has been chosen. No other external disturbances are applied to test the controller. The output of the controlled system, output of the open loop system and the control signals for each input/output channel are shown below in Figure 7.34, Figure 7.35, Figure 7.36, Figure 7.37, Figure 7.38 and Figure 7.39.

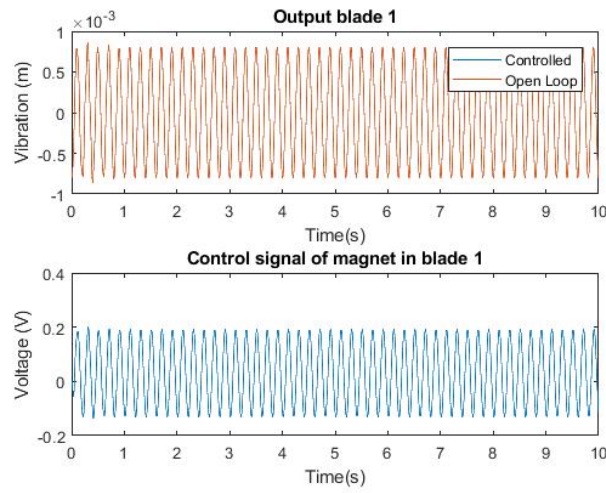


Figure 7.34. Vibration measurement and control signal of blade 1 under LQR controller on rotating system

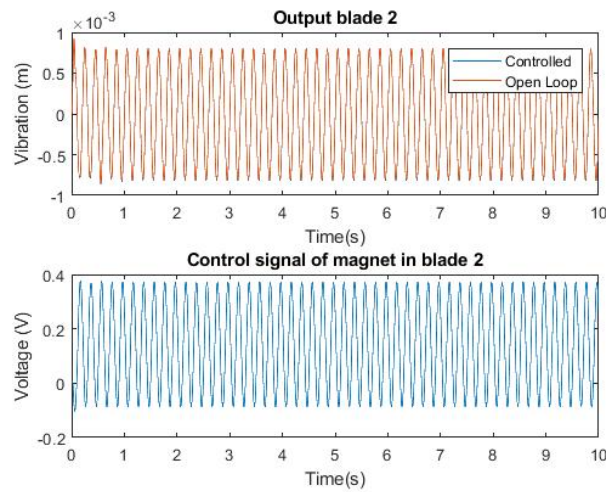


Figure 7.35. Vibration measurement and control signal of blade 2 under LQR controller on rotating system

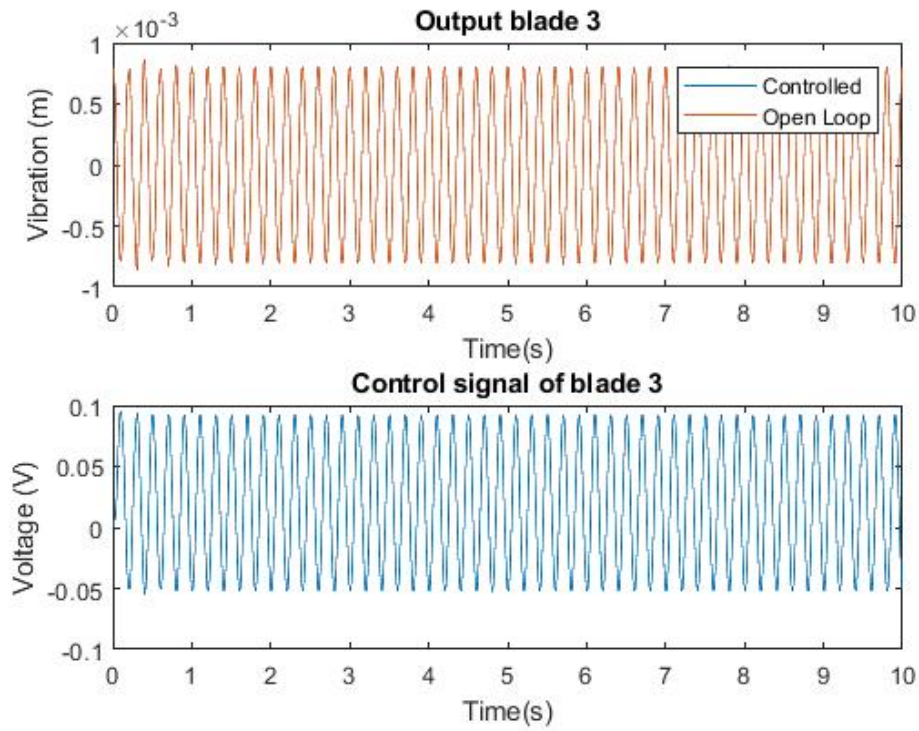


Figure 7.36. Vibration measurement and control signal of blade 3 under LQR controller on rotating system

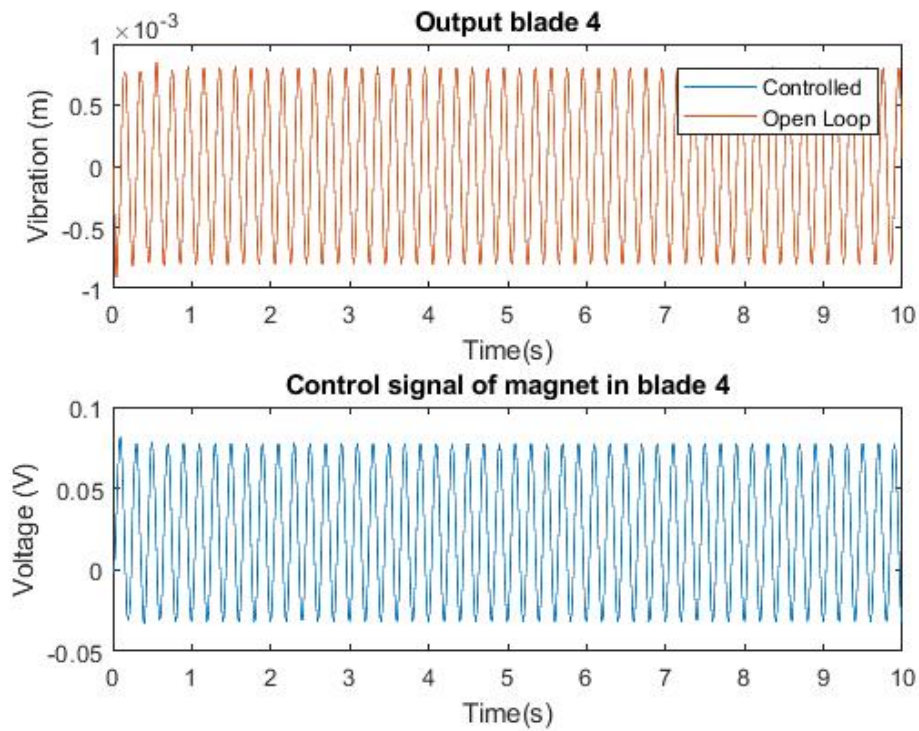


Figure 7.37. Vibration measurement and control signal of blade 4 under LQR controller on rotating system

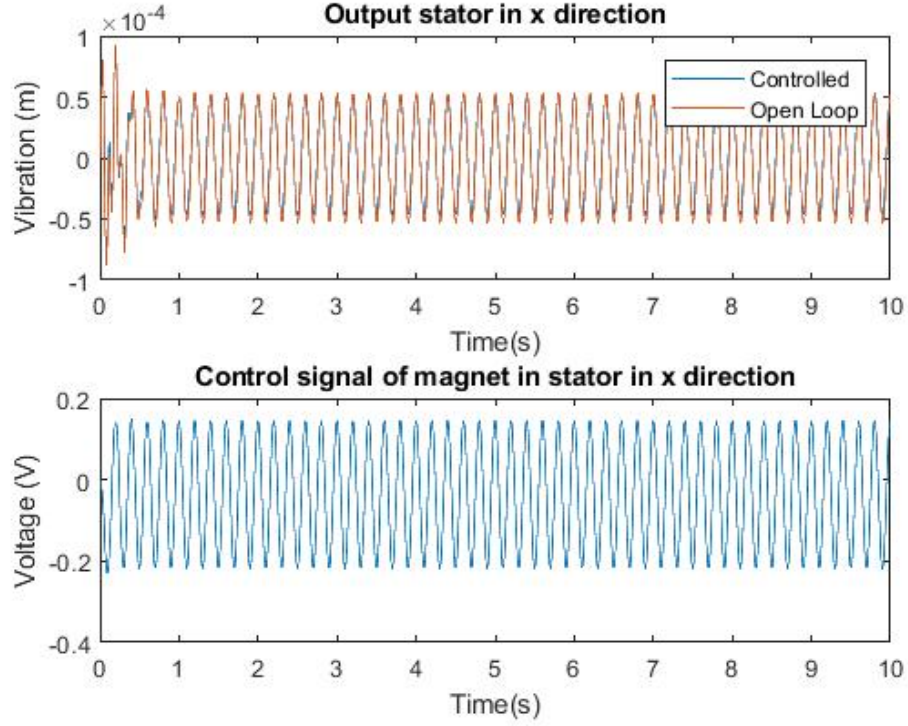


Figure 7.38. Vibration measurement and control signal of stator in x direction under LQR controller on rotating system

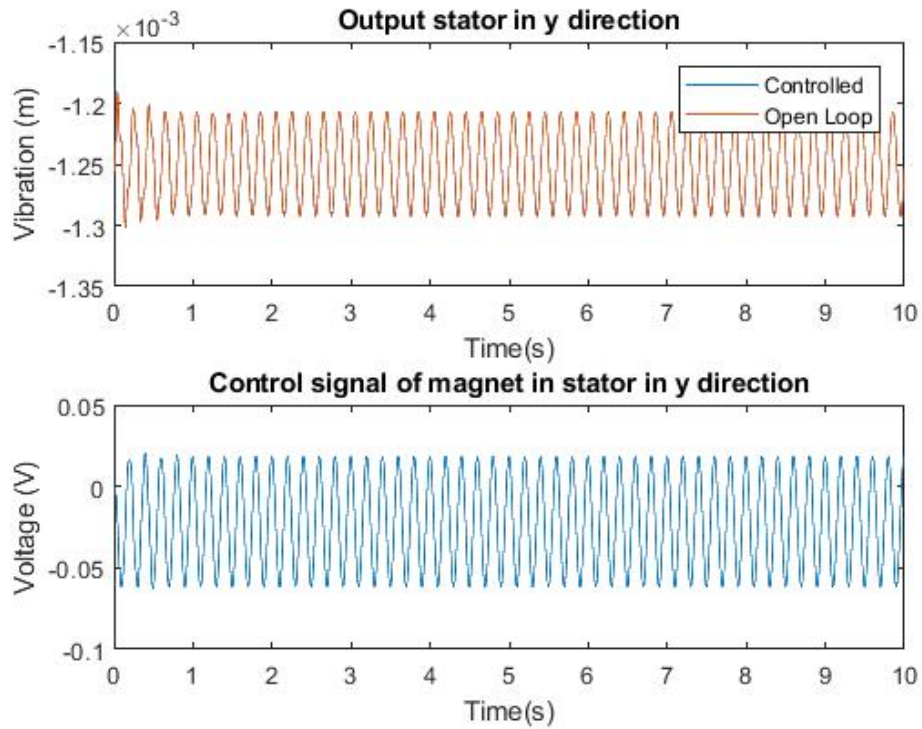


Figure 7.39. Vibration measurement and control signal of stator in y direction under LQR controller on rotating system

It was seen that in static operation the disturbance rejection was fast and accurate enough for the system, however, when in rotating conditions, the controller cannot perform as wanted and fails to dampen the vibrations of the system.

It can be concluded from the LQR controller that its operation in static conditions is good enough, but it is not valid for rotating conditions, and therefore a more complex controller needs to be implemented.

7.3 Adaptive controller implementation

For the implementation of the adaptive controller the two blocks of online identification and adjustable controller are set together. Those blocks have been illustrated previously on the design phase, in Figure 6.22 and Figure 6.19. The whole implementation of the adaptive controller is shown in Figure 7.40 below, where the online identification and adjustable controller are included in the subsystems named as such.

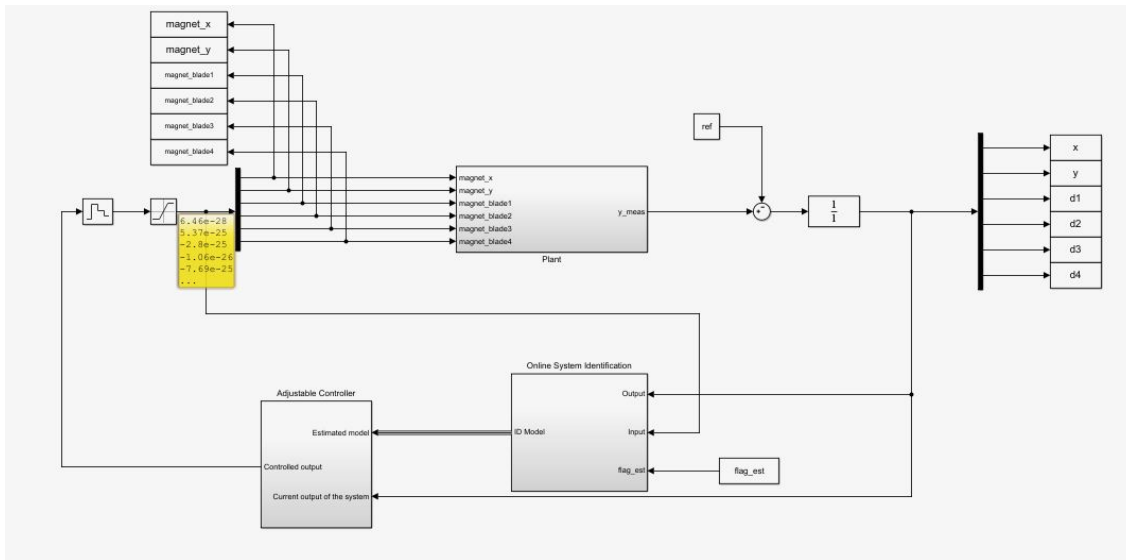


Figure 7.40. Block diagram of the whole implementation of the adaptive controller

The online identification block includes the identification of the 6 MISO systems, as it is shown in Figure 7.41 below. The figure shows the whole content of the online identification subsystem.

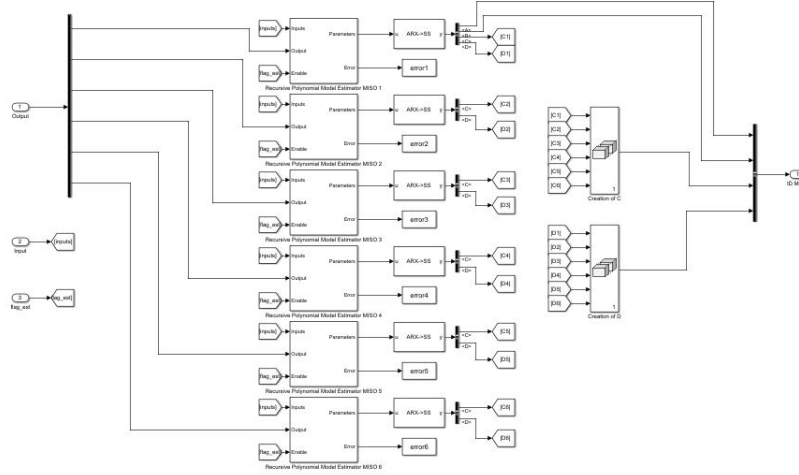


Figure 7.41. Block diagram of the online identification system implementation

For better estimation and adaptation purposes, the rotation speed of the system is input as a step from static operation (null speed) to 5 Hz rotation, step time in 3 seconds. The initial estimation in static operation of the system is input into the online identification blocks for a better convergence of the method. The only input to the adjustable controller block is the functions for calculating the observer and the LQR at each step time. The MatLab code used for the implementation of the adaptive controller is included in Appendix C.

In the simulations it can be seen that the real system is very close to instability. It has some oscillatory poles and the others are really close to being unstable. When the system is discretized for the online identification and the adjustable controller, the poles are really close to the unit circle when in static operation. When the system is rotating, however, those poles values vary with time in an oscillatory trend, making the identified varying system unstable most of the time. This instability makes the observer for the adjustable controller difficult to implement. The observer's feedback gain (L) has been chosen null, since the poles of the time variant system cannot be modified to stability and the observer cannot be designed. Having these considerations into account, the adaptive controller for the coupled system (controlling both hub and blades) is tested, and an example of an output result is shown below in Figure 7.42.

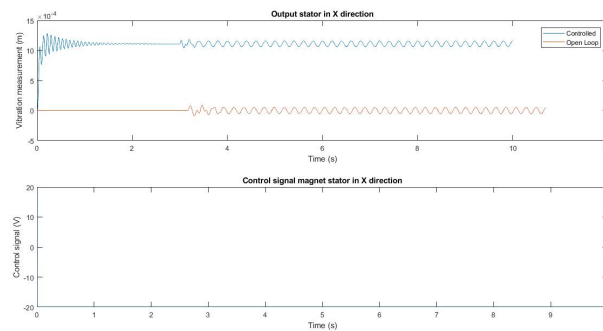


Figure 7.42. Vibration measurement and control signal for stator in X direction with adaptive controller

It can be seen in the plot that the system's vibrations cannot be controlled, since the identified plant is unstable and the controller saturates at the maximum or minimum value allowed by the electromagnets. This instability of the plant is caused by the added dynamics of the magnets, which add two oscillatory poles. However, if the system parameters in the real test rig were to be dampened (decreased) in order to enter the stability zone, the system would become controllable with this adaptive approach. The output for the changed system dynamics (in the stability zone) is shown below in the figures.

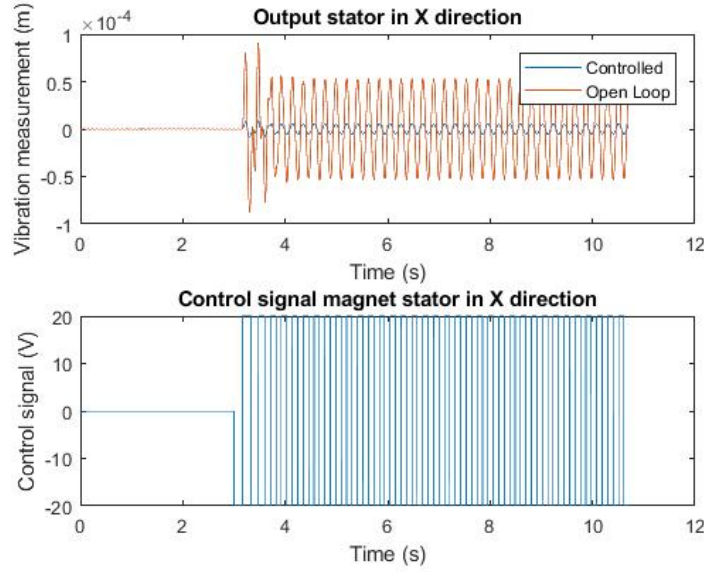


Figure 7.43. Vibration measurement and control signal for stator in X direction with adaptive controller

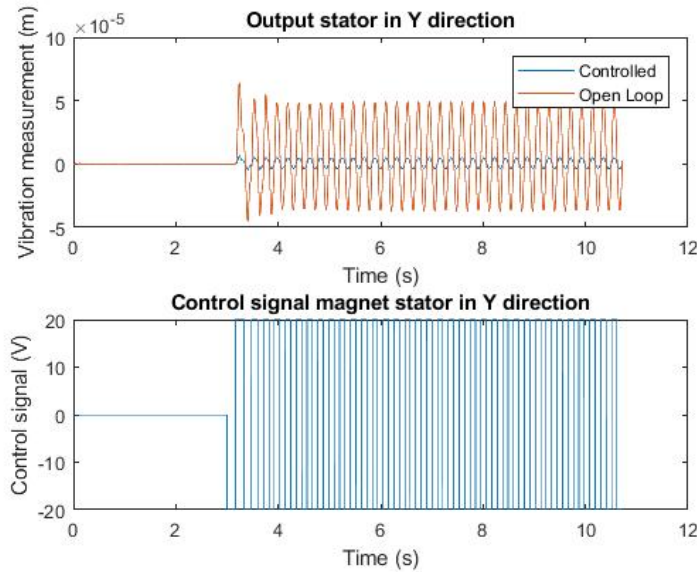


Figure 7.44. Vibration measurement and control signal for stator in Y direction with adaptive controller

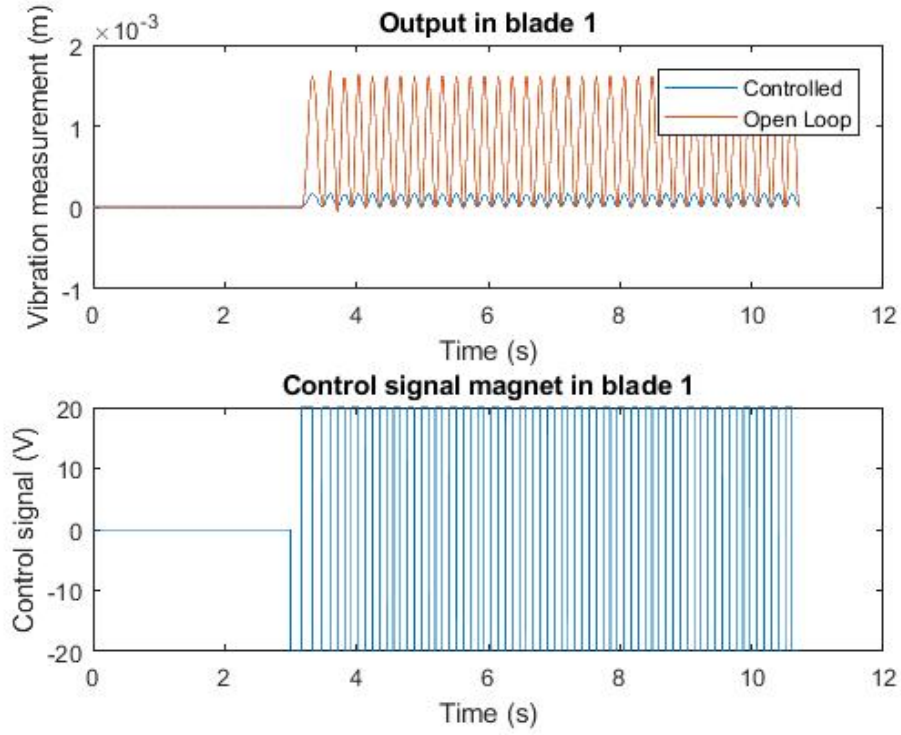


Figure 7.45. Vibration measurement and control signal for blade 1 with adaptive controller

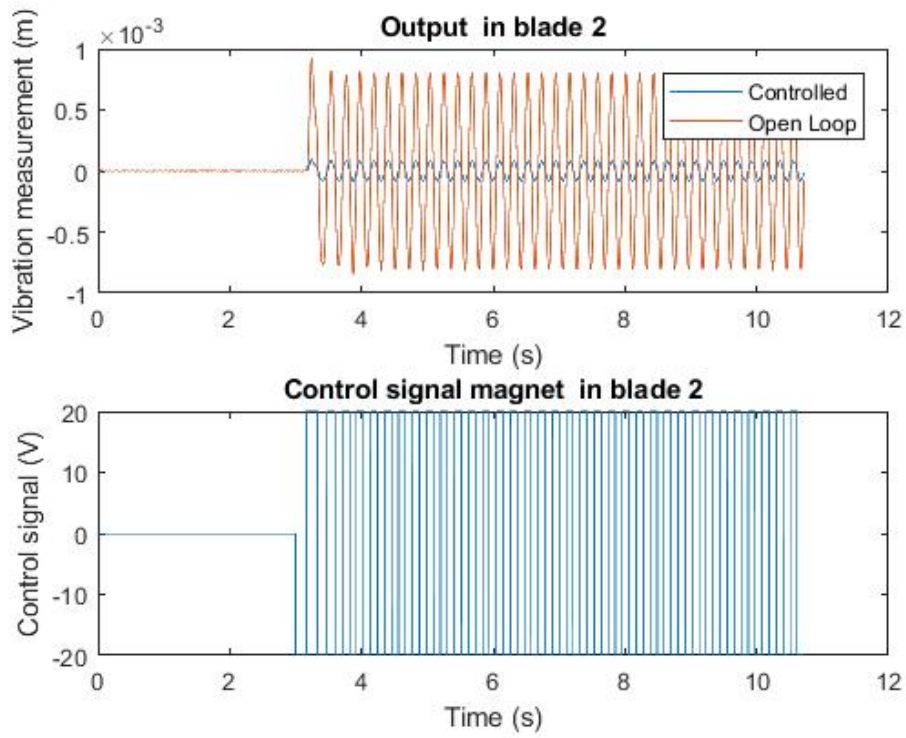


Figure 7.46. Vibration measurement and control signal for blade 2 with adaptive controller

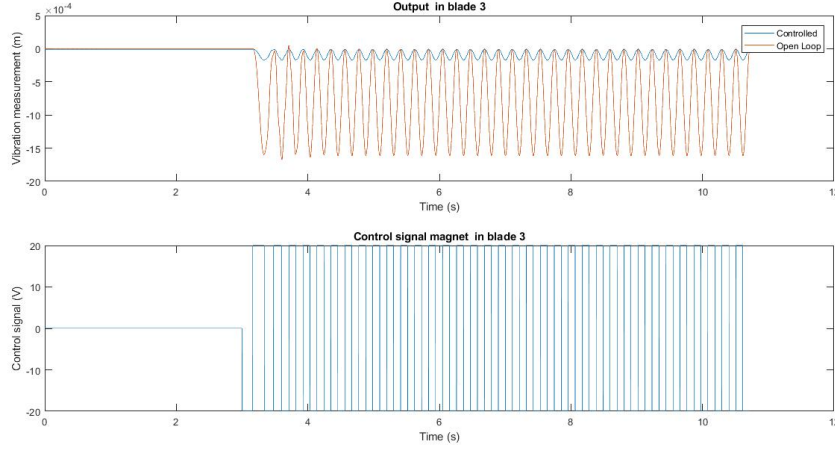


Figure 7.47. Vibration measurement and control signal for blade 3 with adaptive controller

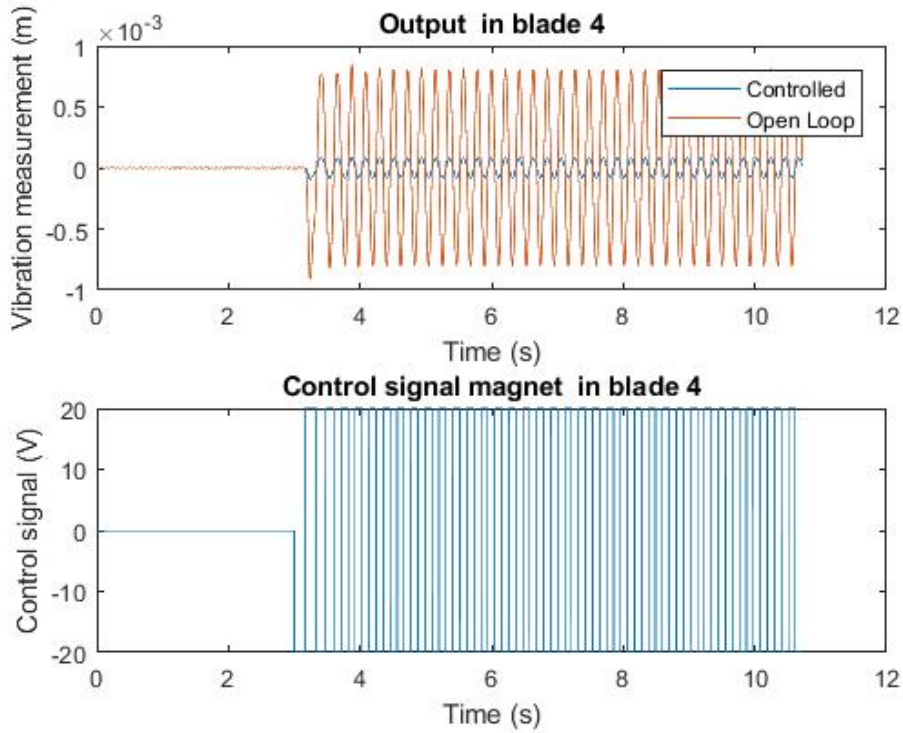


Figure 7.48. Vibration measurement and control signal for blade 4 with adaptive controller

As it is seen in the plots, the adaptive controller can dampen the vibrations of the rotating test rig at least one magnitude order, enough to make them almost negligible. For this adaptive controller approach it can be concluded that if the test rig's parameters are modified so the plant is in the stable region the adaptive controller is the best way among the tried controllers to dampen the vibrations of the plant in rotating conditions. The change in the system's dynamics will determine if the system can be controlled in the uncoupled situation, only controlling the hub's vibrations, since the control power the magnets can output is limited.

CONCLUSIONS AND FUTURE

8 IMPLEMENTATIONS

In this project the first task was to identify the system experimentally to compare it with the theoretical system and find possible non modeled dynamics. The conclusions for the system identification was the experimental evidence of the theoretical analysis. The system, when in static operation, behaves as a standard dampened second order system. However, when the system is rotating, its parameters become time variant in a periodic trend.

After identification, several controllers were designed to suppress those vibrations. The controllers chosen for static operation were PID controllers for every uncoupled SISO system and a full state feedback with LQR design controller. The results for the PID controllers were that the coupling between crossed input/output channels is too big to be assumed negligible. As for the LQR full state feedback controllers, they could dampen the vibrations in static operation, but when the system was rotating it was unable to control the measurements, for coupled or uncoupled system.

Last, for the adaptive controller, when performing the online system identification it was seen that the system is really close to unstability, in continuous and discrete frameworks, due to the added dynamics of the electromagnets and the close to unstable parameters of this specific plant. This caused the system to become unstable when the parameters vary with time in rotating operation, and therefore the system could not be controlled. A change in the plant's parameters was made to force it into the stable region, and the adaptive controller was capable of controlling the vibrations of the system in rotating operation.

The main conclusion is that the system can be controlled by controlling only the vibrations of the rotor, but a better control performance is achieved if both rotor and blades are controlled. The control technique needed to dampen the vibrations must be a complex technique, standard full state feedback and PID controllers cannot succeed when the system is rotating due to the time variability of the plant. One technique suggested and tested is the adaptive control technique, capable of controlling the vibrations of the blades.

8.1 Future Improvements

For future improvements on the physical test rig different parameters are suggested for more stability margin of the plant. Another improvement suggested is the installment of more powerful actuators and an a closed loop implementation via WiFi communication.

For the controller part, for future improvements and studies is the C code design and implementation of the designed controllers, to implement them on the real test rig.

REFERENCES

- [1] Ioan Landau, Rogelio Lozano, Mohammed M'SAAD, and Alireza Karimi. *Adaptive Control: Algorithms, Analysis and Applications*. Springer, 03 2011. ISBN 978-0-85729-663-4.
- [2] A. Sarkar R.S. Mohan and A.S. Sekhar. Vibration analysis of a steam turbine blade. *International Congress on Noise Control Engineering*, 2014.
- [3] Paolo Pennacchi, Steven Chatterton, N Bachschmid, Emanuel Pesatori, and Giorgio Turozzi. A model to study the reduction of turbine blade vibration using the snubbing mechanism. *Mechanical Systems and Signal Processing*, 25:1260–1275, 05 2011. doi: 10.1016/j.ymssp.2010.10.006.
- [4] P. J. Murtagh, A. Ghosh, B. Basu, and B. M. Broderick. Passive control of wind turbine vibrations including blade/tower interaction and rotationally sampled turbulence. *Wind Energy*, 11(4):305–317, 2008. doi: 10.1002/we.249. URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/we.249>.
- [5] Keivan Torabi, A Azadi, and Ehsan Zafari. Vibration control of the turbine blade using quantitative feedback theory. *ASME Press Ebook*, 2:159–166, 08 2011.
- [6] Prashant Pawar, Sung Jung, and Y.H. Yu. Helicopter vibration reduction using active twist control of composite rotor blades. *Annual Forum Proceedings - AHS International*, 3:2137–2146, 01 2008.
- [7] Juan Camino and Ilmar Santos. A periodic h2 state feedback controller for a rotor-blade system. In *A periodic H2 state feedback controller for a rotor-blade system*, 09 2018.
- [8] Maria Beneyto Gomez-Polo. Classical control design theory applied to mitigate rotor-blade vibrations – a numerical investigation. In *Classical Control Design Theory Applied to Mitigate Rotor-Blade Vibrations – A Numerical Investigation*, 02 2019.
- [9] René H. Christensen and Ilmar Santos. A study of active rotor-blade vibration control using electro-magnetic actuation: Part 1 — theory. In *A Study of Active Rotor-Blade Vibration Control Using Electro-Magnetic Actuation: Part 1 — Theory*, volume 6, 01 2004. doi: 10.1115/GT2004-53509.
- [10] Alvaro Brandez Gorriz. Design, implementation, and testing of hardware for sensing and controlling the dynamics of rotor-bladed systems. In *Design, Implementation, and Testing of Hardware for Sensing and Controlling the Dynamics of Rotor-Bladed Systems*, 02 2019.
- [11] What is a slip ring?
<http://www.trolexengineering.co.uk/what-is-a-slip-ring.html>, 2019.
Accessed: 2019-01-01.

-
- [12] Paul M.J. Van den Hof. *System Identification*. Eindhoven University of Technology, 2012.
 - [13] Lennart Ljung. *System Identification: Theory for the user*. Prentice Hall, 1998.
 - [14] Lennart Ljung. Ways to prepare data for system identification, 1999. URL <https://se.mathworks.com/help/ident/ug/ways-to-prepare-data-for-system-identification.html>.
 - [15] S. Fizek, M. Reisinger, S. Silbers, and W. Amrhein. An electromagnet model comprehending eddy current and end effects. In *2015 IEEE 11th International Conference on Power Electronics and Drive Systems*, pages 668–672, June 2015. doi: 10.1109/PEDS.2015.7203454.
 - [16] Pavel Trnka. Subspace identification methods. In *Subspace Identification Methods*, 09 2005.
 - [17] Pedregal D.J. Young P.C. Taylor, C.J. and W. Tych. Environmental time series analysis and forecasting with the captain toolbox,. In *Environmental Modelling and Software*, pages 797–814, 2007.
 - [18] Arthur E. Bryson, Y.-C Ho, and George M. Siouris. Applied optimal control: Optimization, estimation, and control. *Systems, Man and Cybernetics, IEEE Transactions on*, 9:366 – 367, 07 1979. doi: 10.1109/TSMC.1979.4310229.
 - [19] Peter C. Young. Time variable parameter estimation. *IFAC Proceedings Volumes*, 42 (10):432 – 437, 2009. ISSN 1474-6670. doi: <https://doi.org/10.3182/20090706-3-FR-2004.00071>. URL <http://www.sciencedirect.com/science/article/pii/S1474667016386852>. 15th IFAC Symposium on System Identification.
 - [20] D. Luenberger. An introduction to observers. *IEEE Transactions on Automatic Control*, 16(6):596–602, December 1971. ISSN 0018-9286. doi: 10.1109/TAC.1971.1099826.
 - [21] *System Identification Toolbox: User’s Guide*, 2015.

APPENDIX A: EXPERIMENTAL

A IDENTIFICATION MATLAB CODE

Once the input and output data for both static and rotating experiments have been loaded in MatLab structures, the data is chopped and pre-filtered.

A.1 Chopping and pre-filtering data

```
1 data_handle_static.m
2
3 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Data Chopping and Correlating Static %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
4 clc
5
6 %%Loading the input and output from static experiments
7 input_load_static
8 output_load_static
9
10 %%Making input and output with same samples, chopping unnecessary data
11 Ts = 0.001;
12 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Impulse%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
13 Ny = length(y_impulse.blade1.time);
14 Nu = length(u_impulse.blade1.time);
15 if Ny > Nu
16     y =
17         [y_impulse.blade1.blade1_data(Ny-Nu+1:end)';y_impulse.blade1.blade2_data(Ny-Nu+1:end)';y_impulse.blade1.blade3_data(Ny-Nu+1:end)';
18         y_impulse.blade1.blade4_data(Ny-Nu+1:end)';y_impulse.blade1.statorx_data(Ny-Nu+1:end)';y_impulse.blade1.statory_data(Ny-Nu+1:end)'];
19     u = [u_impulse.blade1.data';zeros(5,Nu)];
20 else
21     y =
22         [y_impulse.blade1.blade1_data';y_impulse.blade1.blade2_data';y_impulse.blade1.blade3_data';
23         y_impulse.blade1.blade4_data';y_impulse.blade1.statorx_data';y_impulse.blade1.statory_data'];
24     u = [u_impulse.blade1.data(1:Ny)';zeros(5,Ny)];
25 end
26 data_impulse_blade1 = iddata(y',u',Ts);
27
28 clear y u Nu Ny
29
30 Ny = length(y_impulse.blade2.time);
31 Nu = length(u_impulse.blade2.time);
32 if Ny > Nu
33     y =
34         [y_impulse.blade2.blade1_data(Ny-Nu+1:end)';y_impulse.blade2.blade2_data(Ny-Nu+1:end)';y_impulse.blade2.blade3_data(Ny-Nu+1:end)';
35         y_impulse.blade2.blade4_data(Ny-Nu+1:end)';y_impulse.blade2.statorx_data(Ny-Nu+1:end)';y_impulse.blade2.statory_data(Ny-Nu+1:end)'];
36     u = [zeros(1,Nu);u_impulse.blade2.data';zeros(4,Nu)];
```

```

34 else
35     y =
        [y_impulse.blade2.blade1_data';y_impulse.blade2.blade2_data';y_impulse.blade2.blade3_data';
36         y_impulse.blade2.blade4_data';y_impulse.blade2.statorx_data';y_impulse.blade2.statory_data'];
37     u = [zeros(1,Ny);u_impulse.blade2.data(1:Ny)';zeros(4,Ny)];
38 end
39 data_impulse_blade2 = iddata(y',u',Ts);
40
41 clear y u Nu Ny
42
43 Ny = length(y_impulse.blade3.time);
44 Nu = length(u_impulse.blade3.time);
45 if Ny > Nu
46     y =
        [y_impulse.blade3.blade1_data(Ny-Nu+1:end)';y_impulse.blade3.blade2_data(Ny-Nu+1:end)';y_impulse.blade3.blade3_data(Ny-Nu+1:end)';
47         y_impulse.blade3.blade4_data(Ny-Nu+1:end)';y_impulse.blade3.statorx_data(Ny-Nu+1:end)';y_impulse.blade3.statory_data(Ny-Nu+1:end)'];
48     u = [zeros(2,Nu);u_impulse.blade3.data';zeros(3,Nu)];
49 else
50     y =
        [y_impulse.blade3.blade1_data';y_impulse.blade3.blade2_data';y_impulse.blade3.blade3_data';y_impulse.blade3.blade4_data';
51         y_impulse.blade3.statorx_data';y_impulse.blade3.statory_data'];
52     u = [zeros(2,Ny);u_impulse.blade3.data(1:Ny)';zeros(3,Ny)];
53 end
54 data_impulse_blade3 = iddata(y',u',Ts);
55
56 clear y u Nu Ny
57
58 Ny = length(y_impulse.blade4.time);
59 Nu = length(u_impulse.blade4.time);
60 if Ny > Nu
61     y =
        [y_impulse.blade4.blade1_data(Ny-Nu+1:end)';y_impulse.blade4.blade2_data(Ny-Nu+1:end)';y_impulse.blade4.blade3_data(Ny-Nu+1:end)';
62         y_impulse.blade4.blade4_data(Ny-Nu+1:end)';y_impulse.blade4.statorx_data(Ny-Nu+1:end)';y_impulse.blade4.statory_data(Ny-Nu+1:end)'];
63     u = [zeros(3,Nu);u_impulse.blade4.data';zeros(2,Nu)];
64 else
65     y =
        [y_impulse.blade4.blade1_data';y_impulse.blade4.blade2_data';y_impulse.blade4.blade3_data';y_impulse.blade4.blade4_data';
66         y_impulse.blade4.statorx_data';y_impulse.blade4.statory_data'];
67     u = [zeros(3,Ny);u_impulse.blade4.data(1:Ny)';zeros(2,Ny)];
68 end
69 data_impulse_blade4 = iddata(y',u',Ts);
70
71 clear y u Nu Ny
72
73 Ny = length(y_impulse.statorx.time);
74 Nu = length(u_impulse.statorx.time);
75 if Ny > Nu
76     y =
        [y_impulse.statorx.blade1_data(Ny-Nu+1:end)';y_impulse.statorx.blade2_data(Ny-Nu+1:end)';y_impulse.statorx.blade3_data(Ny-Nu+1:end)';
77         y_impulse.statorx.blade4_data(Ny-Nu+1:end)';y_impulse.statorx.statorx_data(Ny-Nu+1:end)'];
78     u = [zeros(4,Nu);u_impulse.statorx.data';zeros(1,Nu)];
79 else

```

```
80     y =  
        [y_impulse.statorx.blade1_data';y_impulse.statorx.blade2_data';y_impulse.statorx.blade3_data'  
81         y_impulse.statorx.blade4_data';y_impulse.statorx.statorx_data';y_impulse.statorx.statory_data'];  
82     u = [zeros(4,Ny);u_impulse.statorx.data(1:Ny)';zeros(1,Ny)];  
83 end  
84 data_impulse_statorx = iddata(y',u',Ts);  
85  
86 clear y u Nu Ny  
87  
88 Ny = length(y_impulse.statory.time);  
89 Nu = length(u_impulse.statory.time);  
90 if Ny > Nu  
91     y =  
        [y_impulse.statory.blade1_data(Ny-Nu+1:end)';y_impulse.statory.blade2_data(Ny-Nu+1:end)';y_im  
92         y_impulse.statory.blade4_data(Ny-Nu+1:end)';y_impulse.statory.statorx_data(Ny-Nu+1:end)';y_i  
93     u = [zeros(5,Nu);u_impulse.statory.data'];  
94 else  
95     y =  
        [y_impulse.statory.blade1_data';y_impulse.statory.blade2_data';y_impulse.statory.blade3_data'  
96         y_impulse.statory.blade4_data';y_impulse.statory.statorx_data';y_impulse.statory.statory_data'];  
97     u = [zeros(5,Ny);u_impulse.statory.data(1:Ny)'];  
98 end  
99 data_impulse_statory = iddata(y',u',Ts);  
100  
101 clear y u Nu Ny  
102 clear u_impulse y_impulse  
103  
104 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Step%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
105 Ny = length(y_step.blade1.time);  
106 Nu = length(u_step.blade1.time);  
107 if Ny > Nu  
108     y =  
        [y_step.blade1.blade1_data(Ny-Nu+1:end)';y_step.blade1.blade2_data(Ny-Nu+1:end)';y_step.blade  
109         y_step.blade1.blade4_data(Ny-Nu+1:end)';y_step.blade1.statorx_data(Ny-Nu+1:end)';y_step.blad  
110     u = [u_step.blade1.data';zeros(5,Nu)];  
111 else  
112     y =  
        [y_step.blade1.blade1_data';y_step.blade1.blade2_data';y_step.blade1.blade3_data';  
113         y_step.blade1.blade4_data';y_step.blade1.statorx_data';y_step.blade1.statory_data'];  
114     u = [u_step.blade1.data(1:Ny)';zeros(5,Ny)];  
115 end  
116 data_step_blade1 = iddata(y',u',Ts);  
117  
118 clear y u Nu Ny  
119  
120 Ny = length(y_step.blade2.time);  
121 Nu = length(u_step.blade2.time);  
122 if Ny > Nu  
123     y =  
        [y_step.blade2.blade1_data(Ny-Nu+1:end)';y_step.blade2.blade2_data(Ny-Nu+1:end)';y_step.blade  
124         y_step.blade2.blade4_data(Ny-Nu+1:end)';y_step.blade2.statorx_data(Ny-Nu+1:end)';y_step.blad  
125     u = [zeros(1,Nu);u_step.blade2.data';zeros(4,Nu)];
```

```

126 else
127     y =
        [y_step.blade2.blade1_data';y_step.blade2.blade2_data';y_step.blade2.blade3_data';
128         y_step.blade2.blade4_data';y_step.blade2.statorx_data';y_step.blade2.statory_data'];
129     u = [zeros(1,Ny);u_step.blade2.data(1:Ny)';zeros(4,Ny)];
130 end
131 data_step_blade2 = iddata(y',u',Ts);
132
133 clear y u Nu Ny
134
135 Ny = length(y_step.blade3.time);
136 Nu = length(u_step.blade3.time);
137 if Ny > Nu
138     y =
        [y_step.blade3.blade1_data(Ny-Nu+1:end)';y_step.blade3.blade2_data(Ny-Nu+1:end)';y_step.
139         y_step.blade3.blade4_data(Ny-Nu+1:end)';y_step.blade3.statorx_data(Ny-Nu+1:end)';y_step.
140         u = [zeros(2,Nu);u_step.blade3.data';zeros(3,Nu)];
141     else
142         y =
        [y_step.blade3.blade1_data';y_step.blade3.blade2_data';y_step.blade3.blade3_data';
143         y_step.blade3.blade4_data';y_step.blade3.statorx_data';y_step.blade3.statory_data'];
144         u = [zeros(2,Ny);u_step.blade3.data(1:Ny)';zeros(3,Ny)];
145     end
146 data_step_blade3 = iddata(y',u',Ts);
147
148 clear y u Nu Ny
149
150 Ny = length(y_step.blade4.time);
151 Nu = length(u_step.blade4.time);
152 if Ny > Nu
153     y =
        [y_step.blade4.blade1_data(1:Nu)';y_step.blade4.blade2_data(1:Nu)';y_step.blade4.blade3.
154         y_step.blade4.blade4_data(1:Nu)';y_step.blade4.statorx_data(1:Nu)';y_step.blade4.stator
155         u = [zeros(3,Nu);u_step.blade4.data';zeros(2,Nu)];
156     else
157         y =
        [y_step.blade4.blade1_data';y_step.blade4.blade2_data';y_step.blade4.blade3_data';
158         y_step.blade4.blade4_data';y_step.blade4.statorx_data';y_step.blade4.statory_data'];
159         u = [zeros(3,Ny);u_step.blade4.data(1:Ny)';zeros(2,Ny)];
160     end
161 data_step_blade4 = iddata(y',u',Ts);
162
163 clear y u Nu Ny
164
165 Ny = length(y_step.statorx.time);
166 Nu = length(u_step.statorx.time);
167 if Ny > Nu
168     y =
        [y_step.statorx.blade1_data(Ny-Nu+1:end)';y_step.statorx.blade2_data(Ny-Nu+1:end)';y_ste
169         y_step.statorx.blade4_data(Ny-Nu+1:end)';y_step.statorx.statorx_data(Ny-Nu+1:end)';y_st
170         u = [zeros(4,Nu);u_step.statorx.data';zeros(1,Nu)];
171     else

```

```
172     y =  
        [y_step.statorx.blade1_data';y_step.statorx.blade2_data';y_step.statorx.blade3_data';  
173         y_step.statorx.blade4_data';y_step.statorx.statorx_data';y_step.statorx.statory_data'];  
174     u = [zeros(4,Ny);u_step.statorx.data(1:Ny)';zeros(1,Ny)];  
175 end  
176 data_step_statorx = iddata(y',u',Ts);  
177  
178 clear y u Nu Ny  
179  
180 Ny = length(y_step.statory.time);  
181 Nu = length(u_step.statory.time);  
182 if Ny > Nu  
183     y =  
        [y_step.statory.blade1_data(Ny-Nu+1:end)';y_step.statory.blade2_data(Ny-Nu+1:end)';y_step.sta  
184         y_step.statory.blade4_data(Ny-Nu+1:end)';y_step.statory.statorx_data(Ny-Nu+1:end)';y_step.st  
185     u = [zeros(5,Nu);u_step.statory.data'];  
186 else  
187     y =  
        [y_step.statory.blade1_data';y_step.statory.blade2_data';y_step.statory.blade3_data';  
188         y_step.statory.blade4_data';y_step.statory.statorx_data';y_step.statory.statory_data'];  
189     u = [zeros(5,Ny);u_step.statory.data(1:Ny)'];  
190 end  
191 data_step_statory = iddata(y',u',Ts);  
192  
193 clear y u Nu Ny  
194 clear u_step y_step  
195  
196 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Chirp%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
197 u_scaled = interpolate_dataset(u_chirp.blade1,Ts);  
198 Ny = length(y_chirp.blade1.time);  
199 Nu = length(u_scaled);  
200 if Ny > Nu  
201     y =  
        [y_chirp.blade1.blade1_data(Ny-Nu+1:end)';y_chirp.blade1.blade2_data(Ny-Nu+1:end)';y_chirp.bl  
202         y_chirp.blade1.blade4_data(Ny-Nu+1:end)';y_chirp.blade1.statorx_data(Ny-Nu+1:end)';y_chirp.b  
203     u = [u_scaled;zeros(5,Nu)];  
204 else  
205     y =  
        [y_chirp.blade1.blade1_data';y_chirp.blade1.blade2_data';y_chirp.blade1.blade3_data';  
206         y_chirp.blade1.blade4_data';y_chirp.blade1.statorx_data';y_chirp.blade1.statory_data'];  
207     u = [u_scaled(1:Ny);zeros(5,Ny)];  
208 end  
209 data_chirp_blade1 = iddata(y',u',Ts);  
210  
211 clear y u Ny Nu u_scaled  
212  
213 u_scaled = interpolate_dataset(u_chirp.blade2,Ts);  
214 Ny = length(y_chirp.blade2.time);  
215 Nu = length(u_scaled);  
216 if Ny > Nu  
217     y =  
        [y_chirp.blade2.blade1_data(Ny-Nu+1:end)';y_chirp.blade2.blade2_data(Ny-Nu+1:end)';y_chirp.bl
```

```

218         y_chirp.blade2.blade4_data(Ny-Nu+1:end)';y_chirp.blade2.statorx_data(Ny-Nu+1:end)';y_ch
219     u = [zeros(1,Nu);u_scaled;zeros(4,Nu)];
220 else
221     y =
222         [y_chirp.blade2.blade1_data';y_chirp.blade2.blade2_data';y_chirp.blade2.blade3_data';
223         y_chirp.blade2.blade4_data';y_chirp.blade2.statorx_data';y_chirp.blade2.statory_data'];
224     u = [zeros(1,Ny);u_scaled(1:Ny);zeros(4,Ny)];
225 end
226 data_chirp_blade2 = iddata(y',u',Ts);
227 clear y u Ny Nu u_scaled
228
229 u_scaled = interpolate_dataset(u_chirp.blade3,Ts);
230 Ny = length(y_chirp.blade3.time);
231 Nu = length(u_scaled);
232 if Ny > Nu
233     y =
234         [y_chirp.blade3.blade1_data(Ny-Nu+1:end)';y_chirp.blade3.blade2_data(Ny-Nu+1:end)';y_ch
235         y_chirp.blade3.blade4_data(Ny-Nu+1:end)';y_chirp.blade3.statorx_data(Ny-Nu+1:end)';y_ch
236     u = [zeros(2,Nu);u_scaled;zeros(3,Nu)];
237 else
238     y =
239         [y_chirp.blade3.blade1_data';y_chirp.blade3.blade2_data';y_chirp.blade3.blade3_data';
240         y_chirp.blade3.blade4_data';y_chirp.blade3.statorx_data';y_chirp.blade3.statory_data'];
241     u = [zeros(2,Ny);u_scaled(1:Ny);zeros(3,Ny)];
242 end
243 data_chirp_blade3 = iddata(y',u',Ts);
244 clear y u Ny Nu u_scaled
245
246 u_scaled = interpolate_dataset(u_chirp.blade4,Ts);
247 Ny = length(y_chirp.blade4.time);
248 Nu = length(u_scaled);
249 if Ny > Nu
250     y =
251         [y_chirp.blade4.blade1_data(Ny-Nu+1:end)';y_chirp.blade4.blade2_data(Ny-Nu+1:end)';y_ch
252         y_chirp.blade4.blade4_data(Ny-Nu+1:end)';y_chirp.blade4.statorx_data(Ny-Nu+1:end)';y_ch
253     u = [zeros(3,Nu);u_scaled;zeros(2,Nu)];
254 else
255     y =
256         [y_chirp.blade4.blade1_data';y_chirp.blade4.blade2_data';y_chirp.blade4.blade3_data';
257         y_chirp.blade4.blade4_data';y_chirp.blade4.statorx_data';y_chirp.blade4.statory_data'];
258     u = [zeros(3,Ny);u_scaled(1:Ny);zeros(2,Ny)];
259 end
260 data_chirp_blade4 = iddata(y',u',Ts);
261 clear y u Ny Nu u_scaled
262
263 u_scaled = interpolate_dataset(u_chirp.statorx,Ts);
264 Ny = length(y_chirp.statorx.time);
265 Nu = length(u_scaled);
266 if Ny > Nu

```



```
265     y =
        [y_chirp.statorx.blade1_data(Ny-Nu+1:end)',y_chirp.statorx.blade2_data(Ny-Nu+1:end)',y_chirp.
266         y_chirp.statorx.blade4_data(Ny-Nu+1:end)',y_chirp.statorx.statorx_data(Ny-Nu+1:end)',y_chirp
267     u = [zeros(4,Nu);u_scaled;zeros(1,Nu)];
268 else
269     y =
        [y_chirp.statorx.blade1_data',y_chirp.statorx.blade2_data',y_chirp.statorx.blade3_data',
270         y_chirp.statorx.blade4_data',y_chirp.statorx.statorx_data',y_chirp.statorx.statory_data'];
271     u = [zeros(4,Ny);u_scaled(1:Ny);zeros(1,Ny)];
272 end
273 data_chirp_statorx = iddata(y',u',Ts);
274
275 clear y u Ny Nu u_scaled
276
277 u_scaled = interpolate_dataset(u_chirp.statory,Ts);
278 Ny = length(y_chirp.statory.time);
279 Nu = length(u_scaled);
280 if Ny > Nu
281     y =
        [y_chirp.statory.blade1_data(Ny-Nu+1:end)',y_chirp.statory.blade2_data(Ny-Nu+1:end)',y_chirp.
282         y_chirp.statory.blade4_data(Ny-Nu+1:end)',y_chirp.statory.statorx_data(Ny-Nu+1:end)',y_chirp
283     u = [zeros(5,Nu);u_scaled];
284 else
285     y =
        [y_chirp.statory.blade1_data',y_chirp.statory.blade2_data',y_chirp.statory.blade3_data',
286         y_chirp.statory.blade4_data',y_chirp.statory.statorx_data',y_chirp.statory.statory_data'];
287     u = [zeros(5,Ny);u_scaled(1:Ny)];
288 end
289 data_chirp_statory = iddata(y',u',Ts);
290
291 clear y u Ny Nu u_scaled
292 clear u_chirp y_chirp
293
294 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%PRBS%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
295 u_scaled = interpolate_dataset(u_prbs.blade1,Ts);
296 Ny = length(y_prbs.blade1.time);
297 Nu = length(u_scaled);
298 if Ny > Nu
299     y =
        [y_prbs.blade1.blade1_data(Ny-Nu+1:end)',y_prbs.blade1.blade2_data(Ny-Nu+1:end)',y_prbs.blade
300         y_prbs.blade1.blade4_data(Ny-Nu+1:end)',y_prbs.blade1.statorx_data(Ny-Nu+1:end)',y_prbs.blad
301     u = [u_scaled;zeros(5,Nu)];
302 else
303     y =
        [y_prbs.blade1.blade1_data',y_prbs.blade1.blade2_data',y_prbs.blade1.blade3_data',
304         y_prbs.blade1.blade4_data',y_prbs.blade1.statorx_data',y_prbs.blade1.statory_data'];
305     u = [u_scaled(1:Ny);zeros(5,Ny)];
306 end
307 data_prbs_blade1 = iddata(y',u',Ts);
308
309 clear y u Ny Nu u_scaled
310
```

```

311 u_scaled = interpolate_dataset(u_prbs.blade2,Ts);
312 Ny = length(y_prbs.blade2.time);
313 Nu = length(u_scaled);
314 if Ny > Nu
315     y =
316         [y_prbs.blade2.blade1_data(Ny-Nu+1:end)';y_prbs.blade2.blade2_data(Ny-Nu+1:end)';y_prbs.
317         y_prbs.blade2.blade4_data(Ny-Nu+1:end)';y_prbs.blade2.statorx_data(Ny-Nu+1:end)';y_prbs
318     u = [zeros(1,Nu);u_scaled;zeros(4,Nu)];
319 else
320     y =
321         [y_prbs.blade2.blade1_data';y_prbs.blade2.blade2_data';y_prbs.blade2.blade3_data';
322         y_prbs.blade2.blade4_data';y_prbs.blade2.statorx_data';y_prbs.blade2.statory_data'];
323     u = [zeros(1,Ny);u_scaled(1:Ny);zeros(4,Ny)];
324 end
325 data_prbs_blade2 = iddata(y',u',Ts);
326
327 u_scaled = interpolate_dataset(u_prbs.blade3,Ts);
328 Ny = length(y_prbs.blade3.time);
329 Nu = length(u_scaled);
330 if Ny > Nu
331     y =
332         [y_prbs.blade3.blade1_data(Ny-Nu+1:end)';y_prbs.blade3.blade2_data(Ny-Nu+1:end)';y_prbs.
333         y_prbs.blade3.blade4_data(Ny-Nu+1:end)';y_prbs.blade3.statorx_data(Ny-Nu+1:end)';y_prbs
334     u = [zeros(2,Nu);u_scaled;zeros(3,Nu)];
335 else
336     y =
337         [y_prbs.blade3.blade1_data';y_prbs.blade3.blade2_data';y_prbs.blade3.blade3_data';
338         y_prbs.blade3.blade4_data';y_prbs.blade3.statorx_data';y_prbs.blade3.statory_data'];
339     u = [zeros(2,Ny);u_scaled(1:Ny);zeros(3,Ny)];
340 end
341 data_prbs_blade3 = iddata(y',u',Ts);
342
343 u_scaled = interpolate_dataset(u_prbs.blade4,Ts);
344 Ny = length(y_prbs.blade4.time);
345 Nu = length(u_scaled);
346 if Ny > Nu
347     y =
348         [y_prbs.blade4.blade1_data(Ny-Nu+1:end)';y_prbs.blade4.blade2_data(Ny-Nu+1:end)';y_prbs.
349         y_prbs.blade4.blade4_data(Ny-Nu+1:end)';y_prbs.blade4.statorx_data(Ny-Nu+1:end)';y_prbs
350     u = [zeros(3,Nu);u_scaled;zeros(2,Nu)];
351 else
352     y =
353         [y_prbs.blade4.blade1_data';y_prbs.blade4.blade2_data';y_prbs.blade4.blade3_data';
354         y_prbs.blade4.blade4_data';y_prbs.blade4.statorx_data';y_prbs.blade4.statory_data'];
355     u = [zeros(3,Ny);u_scaled(1:Ny);zeros(2,Ny)];
356 end
357 data_prbs_blade4 = iddata(y',u',Ts);
358

```

```
357 clear y u Ny Nu u_scaled
358
359 u_scaled = interpolate_dataset(u_prbs.statorx,Ts);
360 Ny = length(y_prbs.statorx.time);
361 Nu = length(u_scaled);
362 if Ny > Nu
363     y =
364         [y_prbs.statorx.blade1_data(Ny-Nu+1:end)';y_prbs.statorx.blade2_data(Ny-Nu+1:end)';y_prbs.sta
365         y_prbs.statorx.blade4_data(Ny-Nu+1:end)';y_prbs.statorx.statorx_data(Ny-Nu+1:end)';y_prbs.st
366     u = [zeros(4,Nu);u_scaled;zeros(1,Nu)];
367 else
368     y =
369         [y_prbs.statorx.blade1_data';y_prbs.statorx.blade2_data';y_prbs.statorx.blade3_data';
370         y_prbs.statorx.blade4_data';y_prbs.statorx.statorx_data';y_prbs.statorx.statory_data'];
371     u = [zeros(4,Ny);u_scaled(1:Ny);zeros(1,Ny)];
372 end
373 data_prbs_statorx = iddata(y',u',Ts);
374
375 clear y u Ny Nu u_scaled
376
377 u_scaled = interpolate_dataset(u_prbs.statory,Ts);
378 Ny = length(y_prbs.statory.time);
379 Nu = length(u_scaled);
380 if Ny > Nu
381     y =
382         [y_prbs.statory.blade1_data(Ny-Nu+1:end)';y_prbs.statory.blade2_data(Ny-Nu+1:end)';y_prbs.sta
383         y_prbs.statory.blade4_data(Ny-Nu+1:end)';y_prbs.statory.statorx_data(Ny-Nu+1:end)';y_prbs.st
384     u = [zeros(5,Nu);u_scaled];
385 else
386     y =
387         [y_prbs.statory.blade1_data';y_prbs.statory.blade2_data';y_prbs.statory.blade3_data';
388         y_prbs.statory.blade4_data';y_prbs.statory.statorx_data';y_prbs.statory.statory_data'];
389     u = [zeros(5,Ny);u_scaled(1:Ny)];
390 end
391 data_prbs_statory = iddata(y',u',Ts);
392
393 clear y u Ny Nu u_scaled
394 clear u_prbs y_prbs
395
396 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Square Wave%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
397 Ny = length(y_square.time);
398 Nu = length(u_square.stator_time);
399 if Ny > Nu
400     y =
401         [y_square.blade1_data(Ny-Nu+1:end)';y_square.blade2_data(Ny-Nu+1:end)';y_square.blade3_data(N
402         y_square.blade4_data(Ny-Nu+1:end)';y_square.statorx_data(Ny-Nu+1:end)';y_square.statory_data
403     u =
404         [u_square.data_blade1(1:Nu)';u_square.data_blade2(1:Nu)';u_square.data_blade3(1:Nu)';
405         u_square.data_blade4(1:Nu)';u_square.data_statorx(1:Nu)';u_square.data_statory(1:Nu)'];
406 else
407     y = [y_square.blade1_data';y_square.blade2_data';y_square.blade3_data';
408         y_square.blade4_data';y_square.statorx_data';y_square.statory_data'];
```

```

403     u =
        [u_square.data_blade1(1:Ny)';u_square.data_blade2(1:Ny)';u_square.data_blade3(1:Ny)';
404         u_square.data_blade4(1:Ny)';u_square.data_statorx(1:Ny)';u_square.data_statory(1:Ny)'];
405 end
406 data_square_est = iddata(y(:,1:60000)',u(:,1:60000)',Ts);
407 data_square_val = iddata(y(:,60001:end)',u(:,60001:end)',Ts);
408
409 clear y u Nu Ny
410 clear u_square y_square
411
412 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Simultaneous Random%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
413 Ny = length(y_random.time);
414 Nu = length(u_random.rotor_time);
415 if Ny > Nu
416     y =
        [y_random.blade1_data(Ny-Nu+1:end)';y_random.blade2_data(Ny-Nu+1:end)';y_random.blade3_data
417         y_random.blade4_data(Ny-Nu+1:end)';y_random.statorx_data(Ny-Nu+1:end)';y_random.statory_data
418     u =
        [u_random.data_blade1(1:Nu)';u_random.data_blade2(1:Nu)';u_random.data_blade3(1:Nu)';
419         u_random.data_blade4(1:Nu)';u_random.data_statorx(1:Nu)';u_random.data_statory(1:Nu)'];
420 else
421     y = [y_random.blade1_data';y_random.blade2_data';y_random.blade3_data';
422         y_random.blade4_data';y_random.statorx_data';y_random.statory_data'];
423     u =
        [u_random.data_blade1(1:Ny)';u_random.data_blade2(1:Ny)';u_random.data_blade3(1:Ny)';
424         u_random.data_blade4(1:Ny)';u_random.data_statorx(1:Ny)';u_random.data_statory(1:Ny)'];
425 end
426 data_random_est = iddata(y(:,1:60000)',u(:,1:60000)',Ts);
427 data_random_val = iddata(y(:,60001:end)',u(:,60001:end)',Ts);
428
429 clear y u Nu Ny
430 clear u_random y_random
431
432 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Noise%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
433 Ny = length(y_noise.time);
434 Nu = length(u_noise.time);
435 if Ny > Nu
436     y =
        [y_noise.blade1_data(Ny-Nu+1:end)';y_noise.blade2_data(Ny-Nu+1:end)';y_noise.blade3_data
437         y_noise.blade4_data(Ny-Nu+1:end)';y_noise.statorx_data(Ny-Nu+1:end)';y_noise.statory_data
438     u =
        [u_noise.data_blade1(1:Nu)';u_noise.data_blade2(1:Nu)';u_noise.data_blade3(1:Nu)';
439         u_noise.data_blade4(1:Nu)';u_noise.data_statorx(1:Nu)';u_noise.data_statory(1:Nu)'];
440 else
441     y = [y_noise.blade1_data';y_noise.blade2_data';y_noise.blade3_data';
442         y_noise.blade4_data';y_noise.statorx_data';y_noise.statory_data'];
443     u =
        [u_noise.data_blade1(1:Ny)';u_noise.data_blade2(1:Ny)';u_noise.data_blade3(1:Ny)';
444         u_noise.data_blade4(1:Ny)';u_noise.data_statorx(1:Ny)';u_noise.data_statory(1:Ny)'];
445 end
446 data_noise = iddata(y',u',Ts);
447

```

```

448 clear y u Ts Nu Ny
449 clear u_noise y_noise
450
451 data_handle_rotating.m
452
453 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% Data Chopping and Correlating Rotating %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
454 clc
455
456 %%Loading the input and output from rotating experiments
457 input_load_rotating
458 output_load_rotating
459
460 %%Making input and output with same samples, chopping unnecessary data
461 Ts = 0.001;
462
463 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Impulse%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
464 Ny = length(y_impulse.blade1.time);
465 Nu = length(u_impulse.blade1.time);
466 if Ny > Nu
467     y =
468         [y_impulse.blade1.blade1_data(Ny-Nu+1:end)';y_impulse.blade1.blade2_data(Ny-Nu+1:end)';y_impulse.blade1.blade3_data(Ny-Nu+1:end)';
469         y_impulse.blade1.blade4_data(Ny-Nu+1:end)';y_impulse.blade1.statorx_data(Ny-Nu+1:end)';y_impulse.blade1.statory_data(Ny-Nu+1:end)'];
470     u = [u_impulse.blade1.data';zeros(5,Nu)];
471 else
472     y =
473         [y_impulse.blade1.blade1_data';y_impulse.blade1.blade2_data';y_impulse.blade1.blade3_data';
474         y_impulse.blade1.blade4_data';y_impulse.blade1.statorx_data';y_impulse.blade1.statory_data'];
475     u = [u_impulse.blade1.data(1:Ny)';zeros(5,Ny)];
476 end
477 data_impulse_blade1 = iddata(y',u',Ts);
478
479 clear y u Nu Ny
480
481 Ny = length(y_impulse.blade2.time);
482 Nu = length(u_impulse.blade2.time);
483 if Ny > Nu
484     y =
485         [y_impulse.blade2.blade1_data(Ny-Nu+1:end)';y_impulse.blade2.blade2_data(Ny-Nu+1:end)';y_impulse.blade2.blade3_data(Ny-Nu+1:end)';
486         y_impulse.blade2.blade4_data(Ny-Nu+1:end)';y_impulse.blade2.statorx_data(Ny-Nu+1:end)';y_impulse.blade2.statory_data(Ny-Nu+1:end)'];
487     u = [zeros(1,Nu);u_impulse.blade2.data';zeros(4,Nu)];
488 else
489     y =
490         [y_impulse.blade2.blade1_data';y_impulse.blade2.blade2_data';y_impulse.blade2.blade3_data';
491         y_impulse.blade2.blade4_data';y_impulse.blade2.statorx_data';y_impulse.blade2.statory_data'];
492     u = [zeros(1,Ny);u_impulse.blade2.data(1:Ny)';zeros(4,Ny)];
493 end
494 data_impulse_blade2 = iddata(y',u',Ts);
495
496 clear y u Nu Ny
497
498 Ny = length(y_impulse.blade3.time);
499 Nu = length(u_impulse.blade3.time);

```

```

496 if Ny > Nu
497     y =
        [y_impulse.blade3.blade1_data(Ny-Nu+1:end)';y_impulse.blade3.blade2_data(Ny-Nu+1:end)';y
498         y_impulse.blade3.blade4_data(Ny-Nu+1:end)';y_impulse.blade3.statorx_data(Ny-Nu+1:end)'];
499     u = [zeros(2,Nu);u_impulse.blade3.data';zeros(3,Nu)];
500 else
501     y =
        [y_impulse.blade3.blade1_data';y_impulse.blade3.blade2_data';y_impulse.blade3.blade3_data';
502         y_impulse.blade3.blade4_data';y_impulse.blade3.statorx_data';y_impulse.blade3.statory_data'];
503     u = [zeros(2,Ny);u_impulse.blade3.data(1:Ny)';zeros(3,Ny)];
504 end
505 data_impulse_blade3 = iddata(y',u',Ts);
506
507 clear y u Nu Ny
508
509 Ny = length(y_impulse.blade4.time);
510 Nu = length(u_impulse.blade4.time);
511 if Ny > Nu
512     y =
        [y_impulse.blade4.blade1_data(Ny-Nu+1:end)';y_impulse.blade4.blade2_data(Ny-Nu+1:end)';y
513         y_impulse.blade4.blade4_data(Ny-Nu+1:end)';y_impulse.blade4.statorx_data(Ny-Nu+1:end)'];
514     u = [zeros(3,Nu);u_impulse.blade4.data';zeros(2,Nu)];
515 else
516     y =
        [y_impulse.blade4.blade1_data';y_impulse.blade4.blade2_data';y_impulse.blade4.blade3_data';
517         y_impulse.blade4.blade4_data';y_impulse.blade4.statorx_data';y_impulse.blade4.statory_data'];
518     u = [zeros(3,Ny);u_impulse.blade4.data(1:Ny)';zeros(2,Ny)];
519 end
520 data_impulse_blade4 = iddata(y',u',Ts);
521
522 clear y u Nu Ny
523
524 Ny = length(y_impulse.statorx.time);
525 Nu = length(u_impulse.statorx.time);
526 if Ny > Nu
527     y =
        [y_impulse.statorx.blade1_data(Ny-Nu+1:end)';y_impulse.statorx.blade2_data(Ny-Nu+1:end)';
528         y_impulse.statorx.blade4_data(Ny-Nu+1:end)';y_impulse.statorx.statorx_data(Ny-Nu+1:end)'];
529     u = [zeros(4,Nu);u_impulse.statorx.data';zeros(1,Nu)];
530 else
531     y =
        [y_impulse.statorx.blade1_data';y_impulse.statorx.blade2_data';y_impulse.statorx.blade3_data';
532         y_impulse.statorx.blade4_data';y_impulse.statorx.statorx_data';y_impulse.statorx.statory_data'];
533     u = [zeros(4,Ny);u_impulse.statorx.data(1:Ny)';zeros(1,Ny)];
534 end
535 data_impulse_statorx = iddata(y',u',Ts);
536
537 clear y u Nu Ny
538
539 Ny = length(y_impulse.statory.time);
540 Nu = length(u_impulse.statory.time);
541 if Ny > Nu

```

```

542     y =
        [y_impulse.statory.blade1_data(Ny-Nu+1:end)';y_impulse.statory.blade2_data(Ny-Nu+1:end)';y_im
543         y_impulse.statory.blade4_data(Ny-Nu+1:end)';y_impulse.statory.statorx_data(Ny-Nu+1:end)';y_i
544     u = [zeros(5,Nu);u_impulse.statory.data'];
545 else
546     y =
        [y_impulse.statory.blade1_data';y_impulse.statory.blade2_data';y_impulse.statory.blade3_data'
547         y_impulse.statory.blade4_data';y_impulse.statory.statorx_data';y_impulse.statory.statory_dat
548     u = [zeros(5,Ny);u_impulse.statory.data(1:Ny)'];
549 end
550 data_impulse_statory = iddata(y',u',Ts);
551
552 clear y u Nu Ny
553 clear u_impulse y_impulse
554
555 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Step%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
556 Ny = length(y_step.blade1.time);
557 Nu = length(u_step.blade1.time);
558 if Ny > Nu
559     y =
        [y_step.blade1.blade1_data(Ny-Nu+1:end)';y_step.blade1.blade2_data(Ny-Nu+1:end)';y_step.blade
560         y_step.blade1.blade4_data(Ny-Nu+1:end)';y_step.blade1.statorx_data(Ny-Nu+1:end)';y_step.blad
561     u = [u_step.blade1.data';zeros(5,Nu)];
562 else
563     y =
        [y_step.blade1.blade1_data';y_step.blade1.blade2_data';y_step.blade1.blade3_data';
564         y_step.blade1.blade4_data';y_step.blade1.statorx_data';y_step.blade1.statory_data'];
565     u = [u_step.blade1.data(1:Ny)';zeros(5,Ny)];
566 end
567 data_step_blade1 = iddata(y',u',Ts);
568
569 clear y u Nu Ny
570
571 Ny = length(y_step.blade2.time);
572 Nu = length(u_step.blade2.time);
573 if Ny > Nu
574     y =
        [y_step.blade2.blade1_data(Ny-Nu+1:end)';y_step.blade2.blade2_data(Ny-Nu+1:end)';y_step.blade
575         y_step.blade2.blade4_data(Ny-Nu+1:end)';y_step.blade2.statorx_data(Ny-Nu+1:end)';y_step.blad
576     u = [zeros(1,Nu);u_step.blade2.data';zeros(4,Nu)];
577 else
578     y =
        [y_step.blade2.blade1_data';y_step.blade2.blade2_data';y_step.blade2.blade3_data';
579         y_step.blade2.blade4_data';y_step.blade2.statorx_data';y_step.blade2.statory_data'];
580     u = [zeros(1,Ny);u_step.blade2.data(1:Ny)';zeros(4,Ny)];
581 end
582 data_step_blade2 = iddata(y',u',Ts);
583
584 clear y u Nu Ny
585
586 Ny = length(y_step.blade3.time);
587 Nu = length(u_step.blade3.time);

```

```

588 if Ny > Nu
589     y =
        [y_step.blade3.blade1_data(Ny-Nu+1:end)', y_step.blade3.blade2_data(Ny-Nu+1:end)', y_step.
590         y_step.blade3.blade4_data(Ny-Nu+1:end)', y_step.blade3.statorx_data(Ny-Nu+1:end)', y_step.
591         u = [zeros(2, Nu); u_step.blade3.data'; zeros(3, Nu)];
592 else
593     y =
        [y_step.blade3.blade1_data'; y_step.blade3.blade2_data'; y_step.blade3.blade3_data';
594         y_step.blade3.blade4_data'; y_step.blade3.statorx_data'; y_step.blade3.statory_data'];
595     u = [zeros(2, Ny); u_step.blade3.data(1:Ny)'; zeros(3, Ny)];
596 end
597 data_step_blade3 = iddata(y', u', Ts);
598
599 clear y u Nu Ny
600
601 Ny = length(y_step.blade4.time);
602 Nu = length(u_step.blade4.time);
603 if Ny > Nu
604     y =
        [y_step.blade4.blade1_data(1:Nu)', y_step.blade4.blade2_data(1:Nu)', y_step.blade4.blade3.
605         y_step.blade4.blade4_data(1:Nu)', y_step.blade4.statorx_data(1:Nu)', y_step.blade4.stator
606         u = [zeros(3, Nu); u_step.blade4.data'; zeros(2, Nu)];
607 else
608     y =
        [y_step.blade4.blade1_data'; y_step.blade4.blade2_data'; y_step.blade4.blade3_data';
609         y_step.blade4.blade4_data'; y_step.blade4.statorx_data'; y_step.blade4.statory_data'];
610     u = [zeros(3, Ny); u_step.blade4.data(1:Ny)'; zeros(2, Ny)];
611 end
612 data_step_blade4 = iddata(y', u', Ts);
613
614 clear y u Nu Ny
615
616 Ny = length(y_step.statorx.time);
617 Nu = length(u_step.statorx.time);
618 if Ny > Nu
619     y =
        [y_step.statorx.blade1_data(Ny-Nu+1:end)', y_step.statorx.blade2_data(Ny-Nu+1:end)', y_step.
620         y_step.statorx.blade4_data(Ny-Nu+1:end)', y_step.statorx.statorx_data(Ny-Nu+1:end)', y_step.
621         u = [zeros(4, Nu); u_step.statorx.data'; zeros(1, Nu)];
622 else
623     y =
        [y_step.statorx.blade1_data'; y_step.statorx.blade2_data'; y_step.statorx.blade3_data';
624         y_step.statorx.blade4_data'; y_step.statorx.statorx_data'; y_step.statorx.statory_data'];
625     u = [zeros(4, Ny); u_step.statorx.data(1:Ny)'; zeros(1, Ny)];
626 end
627 data_step_statorx = iddata(y', u', Ts);
628
629 clear y u Nu Ny
630
631 Ny = length(y_step.statory.time);
632 Nu = length(u_step.statory.time);
633 if Ny > Nu

```



```
634     y =  
        [y_step.statory.blade1_data(Ny-Nu+1:end)';y_step.statory.blade2_data(Ny-Nu+1:end)';y_step.sta  
635        y_step.statory.blade4_data(Ny-Nu+1:end)';y_step.statory.statorx_data(Ny-Nu+1:end)';y_step.st  
636     u = [zeros(5,Nu);u_step.statory.data'];  
637 else  
638     y =  
        [y_step.statory.blade1_data';y_step.statory.blade2_data';y_step.statory.blade3_data';  
639        y_step.statory.blade4_data';y_step.statory.statorx_data';y_step.statory.statory_data'];  
640     u = [zeros(5,Ny);u_step.statory.data(1:Ny)'];  
641 end  
642 data_step_statory = iddata(y',u',Ts);  
643  
644 clear y u Nu Ny  
645 clear u_step y_step  
646  
647 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Chirp%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
648 u_scaled = interpolate_dataset(u_chirp.blade1,Ts);  
649 Ny = length(y_chirp.blade1.time);  
650 Nu = length(u_scaled);  
651 if Ny > Nu  
652     y =  
        [y_chirp.blade1.blade1_data(Ny-Nu+1:end)';y_chirp.blade1.blade2_data(Ny-Nu+1:end)';y_chirp.bl  
653        y_chirp.blade1.blade4_data(Ny-Nu+1:end)';y_chirp.blade1.statorx_data(Ny-Nu+1:end)';y_chirp.b  
654     u = [u_scaled;zeros(5,Nu)];  
655 else  
656     y =  
        [y_chirp.blade1.blade1_data';y_chirp.blade1.blade2_data';y_chirp.blade1.blade3_data';  
657        y_chirp.blade1.blade4_data';y_chirp.blade1.statorx_data';y_chirp.blade1.statory_data'];  
658     u = [u_scaled(1:Ny);zeros(5,Ny)];  
659 end  
660 data_chirp_blade1 = iddata(y',u',Ts);  
661  
662 clear y u Ny Nu u_scaled  
663  
664 u_scaled = interpolate_dataset(u_chirp.blade2,Ts);  
665 Ny = length(y_chirp.blade2.time);  
666 Nu = length(u_scaled);  
667 if Ny > Nu  
668     y =  
        [y_chirp.blade2.blade1_data(Ny-Nu+1:end)';y_chirp.blade2.blade2_data(Ny-Nu+1:end)';y_chirp.bl  
669        y_chirp.blade2.blade4_data(Ny-Nu+1:end)';y_chirp.blade2.statorx_data(Ny-Nu+1:end)';y_chirp.b  
670     u = [zeros(1,Nu);u_scaled;zeros(4,Nu)];  
671 else  
672     y =  
        [y_chirp.blade2.blade1_data';y_chirp.blade2.blade2_data';y_chirp.blade2.blade3_data';  
673        y_chirp.blade2.blade4_data';y_chirp.blade2.statorx_data';y_chirp.blade2.statory_data'];  
674     u = [zeros(1,Ny);u_scaled(1:Ny);zeros(4,Ny)];  
675 end  
676 data_chirp_blade2 = iddata(y',u',Ts);  
677  
678 clear y u Ny Nu u_scaled  
679
```

```

680 u_scaled = interpolate_dataset(u_chirp.blade3,Ts);
681 Ny = length(y_chirp.blade3.time);
682 Nu = length(u_scaled);
683 if Ny > Nu
684     y =
        [y_chirp.blade3.blade1_data(Ny-Nu+1:end)';y_chirp.blade3.blade2_data(Ny-Nu+1:end)';y_chirp.blade3.blade3_data(Ny-Nu+1:end)';
685         y_chirp.blade3.blade4_data(Ny-Nu+1:end)';y_chirp.blade3.statorx_data(Ny-Nu+1:end)';y_chirp.blade3.statory_data(Ny-Nu+1:end)'];
686     u = [zeros(2,Nu);u_scaled;zeros(3,Nu)];
687 else
688     y =
        [y_chirp.blade3.blade1_data';y_chirp.blade3.blade2_data';y_chirp.blade3.blade3_data';
689         y_chirp.blade3.blade4_data';y_chirp.blade3.statorx_data';y_chirp.blade3.statory_data'];
690     u = [zeros(2,Ny);u_scaled(1:Ny);zeros(3,Ny)];
691 end
692 data_chirp_blade3 = iddata(y',u',Ts);
693
694 clear y u Ny Nu u_scaled
695
696 u_scaled = interpolate_dataset(u_chirp.blade4,Ts);
697 Ny = length(y_chirp.blade4.time);
698 Nu = length(u_scaled);
699 if Ny > Nu
700     y =
        [y_chirp.blade4.blade1_data(Ny-Nu+1:end)';y_chirp.blade4.blade2_data(Ny-Nu+1:end)';y_chirp.blade4.blade3_data(Ny-Nu+1:end)';
701         y_chirp.blade4.blade4_data(Ny-Nu+1:end)';y_chirp.blade4.statorx_data(Ny-Nu+1:end)';y_chirp.blade4.statory_data(Ny-Nu+1:end)'];
702     u = [zeros(3,Nu);u_scaled;zeros(2,Nu)];
703 else
704     y =
        [y_chirp.blade4.blade1_data';y_chirp.blade4.blade2_data';y_chirp.blade4.blade3_data';
705         y_chirp.blade4.blade4_data';y_chirp.blade4.statorx_data';y_chirp.blade4.statory_data'];
706     u = [zeros(3,Ny);u_scaled(1:Ny);zeros(2,Ny)];
707 end
708 data_chirp_blade4 = iddata(y',u',Ts);
709
710 clear y u Ny Nu u_scaled
711
712 u_scaled = interpolate_dataset(u_chirp.statorx,Ts);
713 Ny = length(y_chirp.statorx.time);
714 Nu = length(u_scaled);
715 if Ny > Nu
716     y =
        [y_chirp.statorx.blade1_data(Ny-Nu+1:end)';y_chirp.statorx.blade2_data(Ny-Nu+1:end)';y_chirp.statorx.blade3_data(Ny-Nu+1:end)';
717         y_chirp.statorx.blade4_data(Ny-Nu+1:end)';y_chirp.statorx.statorx_data(Ny-Nu+1:end)';y_chirp.statorx.statory_data(Ny-Nu+1:end)'];
718     u = [zeros(4,Nu);u_scaled;zeros(1,Nu)];
719 else
720     y =
        [y_chirp.statorx.blade1_data';y_chirp.statorx.blade2_data';y_chirp.statorx.blade3_data';
721         y_chirp.statorx.blade4_data';y_chirp.statorx.statorx_data';y_chirp.statorx.statory_data'];
722     u = [zeros(4,Ny);u_scaled(1:Ny);zeros(1,Ny)];
723 end
724 data_chirp_statorx = iddata(y',u',Ts);
725

```

```
726 clear y u Ny Nu u_scaled
727
728 u_scaled = interpolate_dataset(u_chirp.statory,Ts);
729 Ny = length(y_chirp.statory.time);
730 Nu = length(u_scaled);
731 if Ny > Nu
732     y =
733         [y_chirp.statory.blade1_data(Ny-Nu+1:end)';y_chirp.statory.blade2_data(Ny-Nu+1:end)';y_chirp.
734         y_chirp.statory.blade4_data(Ny-Nu+1:end)';y_chirp.statory.statorx_data(Ny-Nu+1:end)';y_chirp
735     else
736         y =
737         [y_chirp.statory.blade1_data';y_chirp.statory.blade2_data';y_chirp.statory.blade3_data';
738         y_chirp.statory.blade4_data';y_chirp.statory.statorx_data';y_chirp.statory.statory_data'];
739     u = [zeros(5,Ny);u_scaled(1:Ny)];
740 end
741 data_chirp_statory = iddata(y',u',Ts);
742
743 clear y u Ny Nu u_scaled
744 clear u_chirp y_chirp
745 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%PRBS%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
746 u_scaled = interpolate_dataset(u_prbs.blade1,Ts);
747 Ny = length(y_prbs.blade1.time);
748 Nu = length(u_scaled);
749 if Ny > Nu
750     y =
751         [y_prbs.blade1.blade1_data(Ny-Nu+1:end)';y_prbs.blade1.blade2_data(Ny-Nu+1:end)';y_prbs.blade
752         y_prbs.blade1.blade4_data(Ny-Nu+1:end)';y_prbs.blade1.statorx_data(Ny-Nu+1:end)';y_prbs.blad
753     else
754         y =
755         [y_prbs.blade1.blade1_data';y_prbs.blade1.blade2_data';y_prbs.blade1.blade3_data';
756         y_prbs.blade1.blade4_data';y_prbs.blade1.statorx_data';y_prbs.blade1.statory_data'];
757     u = [u_scaled(1:Ny);zeros(5,Ny)];
758 end
759 data_prbs_blade1 = iddata(y',u',Ts);
760
761 clear y u Ny Nu u_scaled
762 u_scaled = interpolate_dataset(u_prbs.blade2,Ts);
763 Ny = length(y_prbs.blade2.time);
764 Nu = length(u_scaled);
765 if Ny > Nu
766     y =
767         [y_prbs.blade2.blade1_data(Ny-Nu+1:end)';y_prbs.blade2.blade2_data(Ny-Nu+1:end)';y_prbs.blade
768         y_prbs.blade2.blade4_data(Ny-Nu+1:end)';y_prbs.blade2.statorx_data(Ny-Nu+1:end)';y_prbs.blad
769     else
770         y =
771         [y_prbs.blade2.blade1_data';y_prbs.blade2.blade2_data';y_prbs.blade2.blade3_data';
772         y_prbs.blade2.blade4_data';y_prbs.blade2.statorx_data';y_prbs.blade2.statory_data'];
```

```

772     u = [zeros(1,Ny);u_scaled(1:Ny);zeros(4,Ny)];
773 end
774 data_prbs_blade2 = iddata(y',u',Ts);
775
776 clear y u Ny Nu u_scaled
777
778 u_scaled = interpolate_dataset(u_prbs.blade3,Ts);
779 Ny = length(y_prbs.blade3.time);
780 Nu = length(u_scaled);
781 if Ny > Nu
782     y =
783         [y_prbs.blade3.blade1_data(Ny-Nu+1:end)';y_prbs.blade3.blade2_data(Ny-Nu+1:end)';y_prbs.
784         y_prbs.blade3.blade4_data(Ny-Nu+1:end)';y_prbs.blade3.statorx_data(Ny-Nu+1:end)';y_prbs.
785     u = [zeros(2,Nu);u_scaled;zeros(3,Nu)];
786 else
787     y =
788         [y_prbs.blade3.blade1_data';y_prbs.blade3.blade2_data';y_prbs.blade3.blade3_data';
789         y_prbs.blade3.blade4_data';y_prbs.blade3.statorx_data';y_prbs.blade3.statory_data'];
790     u = [zeros(2,Ny);u_scaled(1:Ny);zeros(3,Ny)];
791 end
792 data_prbs_blade3 = iddata(y',u',Ts);
793
794 clear y u Ny Nu u_scaled
795
796 u_scaled = interpolate_dataset(u_prbs.blade4,Ts);
797 Ny = length(y_prbs.blade4.time);
798 Nu = length(u_scaled);
799 if Ny > Nu
800     y =
801         [y_prbs.blade4.blade1_data(Ny-Nu+1:end)';y_prbs.blade4.blade2_data(Ny-Nu+1:end)';y_prbs.
802         y_prbs.blade4.blade4_data(Ny-Nu+1:end)';y_prbs.blade4.statorx_data(Ny-Nu+1:end)';y_prbs.
803     u = [zeros(3,Nu);u_scaled;zeros(2,Nu)];
804 else
805     y =
806         [y_prbs.blade4.blade1_data';y_prbs.blade4.blade2_data';y_prbs.blade4.blade3_data';
807         y_prbs.blade4.blade4_data';y_prbs.blade4.statorx_data';y_prbs.blade4.statory_data'];
808     u = [zeros(3,Ny);u_scaled(1:Ny);zeros(2,Ny)];
809 end
810 data_prbs_blade4 = iddata(y',u',Ts);
811
812 clear y u Ny Nu u_scaled
813
814 u_scaled = interpolate_dataset(u_prbs.statorx,Ts);
815 Ny = length(y_prbs.statorx.time);
816 Nu = length(u_scaled);
817 if Ny > Nu
818     y =
819         [y_prbs.statorx.blade1_data(Ny-Nu+1:end)';y_prbs.statorx.blade2_data(Ny-Nu+1:end)';y_prbs.
820         y_prbs.statorx.blade4_data(Ny-Nu+1:end)';y_prbs.statorx.statorx_data(Ny-Nu+1:end)';y_prbs.
821     u = [zeros(4,Nu);u_scaled;zeros(1,Nu)];
822 else

```

```
818     y =  
        [y_prbs.statorx.blade1_data';y_prbs.statorx.blade2_data';y_prbs.statorx.blade3_data';  
819         y_prbs.statorx.blade4_data';y_prbs.statorx.statorx_data';y_prbs.statorx.statory_data'];  
820     u = [zeros(4,Ny);u_scaled(1:Ny);zeros(1,Ny)];  
821 end  
822 data_prbs_statorx = iddata(y',u',Ts);  
823  
824 clear y u Ny Nu u_scaled  
825  
826 u_scaled = interpolate_dataset(u_prbs.statory,Ts);  
827 Ny = length(y_prbs.statory.time);  
828 Nu = length(u_scaled);  
829 if Ny > Nu  
830     y =  
        [y_prbs.statory.blade1_data(Ny-Nu+1:end)';y_prbs.statory.blade2_data(Ny-Nu+1:end)';y_prbs.sta  
831         y_prbs.statory.blade4_data(Ny-Nu+1:end)';y_prbs.statory.statorx_data(Ny-Nu+1:end)';y_prbs.st  
832         u = [zeros(5,Nu);u_scaled];  
833 else  
834     y =  
        [y_prbs.statory.blade1_data';y_prbs.statory.blade2_data';y_prbs.statory.blade3_data';  
835         y_prbs.statory.blade4_data';y_prbs.statory.statorx_data';y_prbs.statory.statory_data'];  
836     u = [zeros(5,Ny);u_scaled(1:Ny)];  
837 end  
838 data_prbs_statory = iddata(y',u',Ts);  
839  
840 clear y u Ny Nu u_scaled  
841 clear u_prbs y_prbs  
842  
843 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Square Wave%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
844 Ny = length(y_square.time);  
845 Nu = length(u_square.stator_time);  
846 if Ny > Nu  
847     y =  
        [y_square.blade1_data(Ny-Nu+1:end)';y_square.blade2_data(Ny-Nu+1:end)';y_square.blade3_data(N  
848         y_square.blade4_data(Ny-Nu+1:end)';y_square.statorx_data(Ny-Nu+1:end)';y_square.statory_data  
849         u =  
        [u_square.data_blade1(1:Nu)';u_square.data_blade2(1:Nu)';u_square.data_blade3(1:Nu)';  
850         u_square.data_blade4(1:Nu)';u_square.data_statorx(1:Nu)';u_square.data_statory(1:Nu)'];  
851 else  
852     y = [y_square.blade1_data';y_square.blade2_data';y_square.blade3_data';  
853         y_square.blade4_data';y_square.statorx_data';y_square.statory_data'];  
854     u =  
        [u_square.data_blade1(1:Ny)';u_square.data_blade2(1:Ny)';u_square.data_blade3(1:Ny)';  
855         u_square.data_blade4(1:Ny)';u_square.data_statorx(1:Ny)';u_square.data_statory(1:Ny)'];  
856 end  
857 data_square_est = iddata(y(:,1:60000)',u(:,1:60000)',Ts);  
858 data_square_val = iddata(y(:,60001:end)',u(:,60001:end)',Ts);  
859  
860 clear y u Nu Ny  
861 clear u_square y_square  
862  
863 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Simultaneous Random%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```

864 Ny = length(y_random.time);
865 Nu = length(u_random.stator_time);
866 if Ny > Nu
867     y =
            [y_random.blade1_data(Ny-Nu+1:end)';y_random.blade2_data(Ny-Nu+1:end)';y_random.blade3_data(Ny-Nu+1:end)';
            y_random.blade4_data(Ny-Nu+1:end)';y_random.statorx_data(Ny-Nu+1:end)';y_random.statory_data(Ny-Nu+1:end)'];
869     u =
            [u_random.data_blade1(1:Nu)';u_random.data_blade2(1:Nu)';u_random.data_blade3(1:Nu)';
            u_random.data_blade4(1:Nu)';u_random.data_statorx(1:Nu)';u_random.data_statory(1:Nu)'];
871 else
872     y = [y_random.blade1_data';y_random.blade2_data';y_random.blade3_data';
            y_random.blade4_data';y_random.statorx_data';y_random.statory_data'];
874     u =
            [u_random.data_blade1(1:Ny)';u_random.data_blade2(1:Ny)';u_random.data_blade3(1:Ny)';
            u_random.data_blade4(1:Ny)';u_random.data_statorx(1:Ny)';u_random.data_statory(1:Ny)'];
876 end
877 data_random_est = iddata(y(:,1:60000)',u(:,1:60000)',Ts);
878 data_random_val = iddata(y(:,60001:end)',u(:,60001:end)',Ts);
879
880 clear y u Nu Ny
881 clear u_random y_random
882
883 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Noise%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
884 Ny = length(y_noise.time);
885 Nu = length(u_noise.time);
886 if Ny > Nu
887     y =
            [y_noise.blade1_data(Ny-Nu+1:end)';y_noise.blade2_data(Ny-Nu+1:end)';y_noise.blade3_data(Ny-Nu+1:end)';
            y_noise.blade4_data(Ny-Nu+1:end)';y_noise.statorx_data(Ny-Nu+1:end)';y_noise.statory_data(Ny-Nu+1:end)'];
889     u =
            [u_noise.data_blade1(1:Nu)';u_noise.data_blade2(1:Nu)';u_noise.data_blade3(1:Nu)';
            u_noise.data_blade4(1:Nu)';u_noise.data_statorx(1:Nu)';u_noise.data_statory(1:Nu)'];
891 else
892     y = [y_noise.blade1_data';y_noise.blade2_data';y_noise.blade3_data';
            y_noise.blade4_data';y_noise.statorx_data';y_noise.statory_data'];
894     u =
            [u_noise.data_blade1(1:Ny)';u_noise.data_blade2(1:Ny)';u_noise.data_blade3(1:Ny)';
            u_noise.data_blade4(1:Ny)';u_noise.data_statorx(1:Ny)';u_noise.data_statory(1:Ny)'];
896 end
897 data_noise = iddata(y',u',Ts);
898
899 clear y u Nu Ny Ts
900 clear u_noise y_noise
901
902 data_prefiltering_static.m
903
904 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Data prefiltering static experiments%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
905 close all
906 clc
907
908 %%Loading the chopped data of the experiments into iddata objects
909 run('data_handle_static.m')

```

```
910
911 %%Pre-filter of experiments as advised by the advice command run
912 %There are three possible pre-filtering operations to be performed: missing
913 %data handling, detrending and filtering
914 %First, if any missing data, an interpolation will be done
915 %Second, we detrend data and take away offsets of the signals
916 %Last, the filtering will be performed in the 50 Hz bandwidth, since it is
917 %the operation bandwidth of the system
918
919 filter = [0,2*pi*50];
920
921 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Impulse%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
922 %After plotting the impulse responses, it has been seen that they offer
923 %little information about the system, since the output is essentially noise
924 %However, they include the offset values for the step experiments
925 offset = mean(data_impulse_blade1.OutputData);
926
927 data_impulse_blade1 = misdata(data_impulse_blade1);
928 t = getTrend(data_impulse_blade1);
929 t.OutputOffset = offset;
930 data_impulse_blade1 = detrend(data_impulse_blade1,t);
931 data_impulse_blade1.OutputData = idfilt(data_impulse_blade1.OutputData,filter);
932 clear t
933
934 data_impulse_blade2 = misdata(data_impulse_blade2);
935 t = getTrend(data_impulse_blade2);
936 t.OutputOffset = offset;
937 data_impulse_blade2 = detrend(data_impulse_blade2,t);
938 data_impulse_blade2.OutputData = idfilt(data_impulse_blade2.OutputData,filter);
939 clear t
940
941 data_impulse_blade3 = misdata(data_impulse_blade3);
942 t = getTrend(data_impulse_blade3);
943 t.OutputOffset = offset;
944 data_impulse_blade3 = detrend(data_impulse_blade3,t);
945 data_impulse_blade3.OutputData = idfilt(data_impulse_blade3.OutputData,filter);
946 clear t
947
948 data_impulse_blade4 = misdata(data_impulse_blade4);
949 t = getTrend(data_impulse_blade4);
950 t.OutputOffset = offset;
951 data_impulse_blade4 = detrend(data_impulse_blade4,t);
952 data_impulse_blade4.OutputData = idfilt(data_impulse_blade4.OutputData,filter);
953 clear t
954
955 data_impulse_statorx = misdata(data_impulse_statorx);
956 t = getTrend(data_impulse_statorx);
957 t.OutputOffset = offset;
958 data_impulse_statorx = detrend(data_impulse_statorx,t);
959 data_impulse_statorx.OutputData = idfilt(data_impulse_statorx.OutputData,filter);
960 clear t
961
```

```

962 data_impulse_statory = misdata(data_impulse_statory);
963 t = getTrend(data_impulse_statory);
964 t.OutputOffset = offset;
965 data_impulse_statory = detrend(data_impulse_statory,t);
966 data_impulse_statory.OutputData = idfilt(data_impulse_statory.OutputData,filter);
967 clear t
968
969 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Step%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
970 data_step_blade1 = misdata(data_step_blade1);
971 t = getTrend(data_step_blade1);
972 t.OutputOffset = offset;
973 data_step_blade1 = detrend(data_step_blade1,t);
974 data_step_blade1.OutputData = idfilt(data_step_blade1.OutputData,filter);
975 clear t
976
977 data_step_blade2 = misdata(data_step_blade2);
978 t = getTrend(data_step_blade2);
979 t.OutputOffset = offset;
980 data_step_blade2 = detrend(data_step_blade2,t);
981 data_step_blade2.OutputData = idfilt(data_step_blade2.OutputData,filter);
982 clear t
983
984 data_step_blade3 = misdata(data_step_blade3);
985 t = getTrend(data_step_blade3);
986 t.OutputOffset = offset;
987 data_step_blade3 = detrend(data_step_blade3,t);
988 data_step_blade3.OutputData = idfilt(data_step_blade3.OutputData,filter);
989 clear t
990
991 data_step_blade4 = misdata(data_step_blade4);
992 t = getTrend(data_step_blade4);
993 t.OutputOffset = offset;
994 data_step_blade4 = detrend(data_step_blade4,t);
995 data_step_blade4.OutputData = idfilt(data_step_blade4.OutputData,filter);
996 clear t
997
998 data_step_statorx = misdata(data_step_statorx);
999 t = getTrend(data_step_statorx);
1000 t.OutputOffset = offset;
1001 data_step_statorx = detrend(data_step_statorx,t);
1002 data_step_statorx.OutputData = idfilt(data_step_statorx.OutputData,filter);
1003 clear t
1004
1005 data_step_statory = misdata(data_step_statory);
1006 t = getTrend(data_step_statory);
1007 t.OutputOffset = offset;
1008 data_step_statory = detrend(data_step_statory,t);
1009 data_step_statory.OutputData = idfilt(data_step_statory.OutputData,filter);
1010 clear t offset
1011
1012 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Chirp%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1013 data_chirp_blade1 = misdata(data_chirp_blade1);

```



```
1014 offset = mean(data_chirp_blade1.OutputData(1:300,:));
1015 t = getTrend(data_chirp_blade1);
1016 t.OutputOffset = offset;
1017 data_chirp_blade1 = detrend(data_chirp_blade1,t);
1018 data_chirp_blade1.OutputData = idfilt(data_chirp_blade1.OutputData,filter);
1019 clear t offset
1020
1021 data_chirp_blade2 = misdata(data_chirp_blade2);
1022 offset = mean(data_chirp_blade2.OutputData(1:300,:));
1023 t = getTrend(data_chirp_blade2);
1024 t.OutputOffset = offset;
1025 data_chirp_blade2 = detrend(data_chirp_blade2,t);
1026 data_chirp_blade2.OutputData = idfilt(data_chirp_blade2.OutputData,filter);
1027 clear t offset
1028
1029 data_chirp_blade3 = misdata(data_chirp_blade3);
1030 offset = mean(data_chirp_blade3.OutputData(1:300,:));
1031 t = getTrend(data_chirp_blade3);
1032 t.OutputOffset = offset;
1033 data_chirp_blade3 = detrend(data_chirp_blade3,t);
1034 data_chirp_blade3.OutputData = idfilt(data_chirp_blade3.OutputData,filter);
1035 clear t offset
1036
1037 data_chirp_blade4 = misdata(data_chirp_blade4);
1038 offset = mean(data_chirp_blade4.OutputData(1:300,:));
1039 t = getTrend(data_chirp_blade4);
1040 t.OutputOffset = offset;
1041 data_chirp_blade4 = detrend(data_chirp_blade4,t);
1042 data_chirp_blade4.OutputData = idfilt(data_chirp_blade4.OutputData,filter);
1043 clear t offset
1044
1045 data_chirp_statorx = misdata(data_chirp_statorx);
1046 offset = mean(data_chirp_statorx.OutputData(1:300,:));
1047 t = getTrend(data_chirp_statorx);
1048 t.OutputOffset = offset;
1049 data_chirp_statorx = detrend(data_chirp_statorx,t);
1050 data_chirp_statorx.OutputData = idfilt(data_chirp_statorx.OutputData,filter);
1051 clear t offset
1052
1053 data_chirp_statory = misdata(data_chirp_statory);
1054 offset = mean(data_chirp_statory.OutputData(1:300,:));
1055 t = getTrend(data_chirp_statory);
1056 t.OutputOffset = offset;
1057 data_chirp_statory = detrend(data_chirp_statory,t);
1058 data_chirp_statory.OutputData = idfilt(data_chirp_statory.OutputData,filter);
1059 clear t offset
1060
1061 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%PRBS%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1062 data_prbs_blade1 = misdata(data_prbs_blade1);
1063 offset = mean(data_prbs_blade1.OutputData(1:300,:));
1064 t = getTrend(data_prbs_blade1);
1065 t.OutputOffset = offset;
```

```

1066 data_prbs_blade1 = detrend(data_prbs_blade1,t);
1067 data_prbs_blade1.OutputData = idfilt(data_prbs_blade1.OutputData,filter);
1068 clear t offset
1069
1070 data_prbs_blade2 = misdata(data_prbs_blade2);
1071 offset = mean(data_prbs_blade2.OutputData(1:300,:));
1072 t = getTrend(data_prbs_blade2);
1073 t.OutputOffset = offset;
1074 data_prbs_blade2 = detrend(data_prbs_blade2,t);
1075 data_prbs_blade2.OutputData = idfilt(data_prbs_blade2.OutputData,filter);
1076 clear t offset
1077
1078 data_prbs_blade3 = misdata(data_prbs_blade3);
1079 offset = mean(data_prbs_blade3.OutputData(1:300,:));
1080 t = getTrend(data_prbs_blade3);
1081 t.OutputOffset = offset;
1082 data_prbs_blade3 = detrend(data_prbs_blade3,t);
1083 data_prbs_blade3.OutputData = idfilt(data_prbs_blade3.OutputData,filter);
1084 clear t offset
1085
1086 data_prbs_blade4 = misdata(data_prbs_blade4);
1087 offset = mean(data_prbs_blade4.OutputData(1:300,:));
1088 t = getTrend(data_prbs_blade4);
1089 t.OutputOffset = offset;
1090 data_prbs_blade4 = detrend(data_prbs_blade4,t);
1091 data_prbs_blade4.OutputData = idfilt(data_prbs_blade4.OutputData,filter);
1092 clear t offset
1093
1094 data_prbs_statorx = misdata(data_prbs_statorx);
1095 offset = mean(data_prbs_statorx.OutputData(1:300,:));
1096 t = getTrend(data_prbs_statorx);
1097 t.OutputOffset = offset;
1098 data_prbs_statorx = detrend(data_prbs_statorx,t);
1099 data_prbs_statorx.OutputData = idfilt(data_prbs_statorx.OutputData,filter);
1100 clear t offset
1101
1102 data_prbs_statory = misdata(data_prbs_statory);
1103 offset = mean(data_prbs_statory.OutputData(1:300,:));
1104 t = getTrend(data_prbs_statory);
1105 t.OutputOffset = offset;
1106 data_prbs_statory = detrend(data_prbs_statory,t);
1107 data_prbs_statory.OutputData = idfilt(data_prbs_statory.OutputData,filter);
1108 clear t offset
1109
1110 %%%%%%%%%%%%%Random%%%%%%%%%%%%
1111 data_random_est = misdata(data_random_est);
1112 offset = mean(data_random_est.OutputData);
1113 t = getTrend(data_random_est);
1114 t.OutputOffset = offset;
1115 data_random_est = detrend(data_random_est,t);
1116 data_random_est.OutputData = idfilt(data_random_est.OutputData,filter);
1117 clear t offset

```

1118

1119 data_random_val = misdata(data_random_val);

1120 offset = mean(data_random_val.OutputData);

1121 t = getTrend(data_random_val);

1122 t.OutputOffset = offset;

1123 data_random_val = detrend(data_random_val,t);

1124 data_random_val.OutputData = idfilt(data_random_val.OutputData,filter);

1125 clear t offset

1126

1127 %%%Square%%

1128 data_square_est = misdata(data_square_est);

1129 offset = mean(data_square_est.OutputData(1:300,:));

1130 t = getTrend(data_square_est);

1131 t.OutputOffset = offset;

1132 data_square_est = detrend(data_square_est,t);

1133 data_square_est.OutputData = idfilt(data_square_est.OutputData,filter);

1134 clear t offset

1135

1136 data_square_val = misdata(data_square_val);

1137 offset = mean(data_square_val.OutputData(1:300,:));

1138 t = getTrend(data_square_val);

1139 t.OutputOffset = offset;

1140 data_square_val = detrend(data_square_val,t);

1141 data_square_val.OutputData = idfilt(data_square_val.OutputData,filter);

1142 clear t offset

1143

1144 %%%Noise%%

1145 data_noise = misdata(data_noise);

1146 offset = mean(data_noise.OutputData);

1147 t = getTrend(data_noise);

1148 t.OutputOffset = offset;

1149 data_noise = detrend(data_noise,t);

1150 data_noise.OutputData = idfilt(data_noise.OutputData,filter);

1151 clear t offset

1152

1153 %%%Merging all experiments in one iddata object%%

1154 %%The experiments are merged because they have been performed in the same

1155 %%conditions, having similar signal to noise ratios

1156

1157 static_est =

```
    merge(data_step_blade1,data_step_blade2,data_step_blade3,data_step_blade4,data_step_statorx,data_
1158         data_chirp_blade1,data_chirp_blade2,data_chirp_blade3,data_chirp_blade4,data_chirp_stat
1159         data_prbs_blade1,data_prbs_blade2,data_prbs_blade3,data_prbs_blade4,data_prbs_statorx,d
1160         data_noise,data_random_est,data_square_est);
```

1161

1162 static_val =

```
    merge(data_step_blade1,data_step_blade2,data_step_blade3,data_step_blade4,data_step_statorx,data_
1163         data_chirp_blade1,data_chirp_blade2,data_chirp_blade3,data_chirp_blade4,data_chirp_stat
1164         data_prbs_blade1,data_prbs_blade2,data_prbs_blade3,data_prbs_blade4,data_prbs_statorx,d
1165         data_noise,data_random_val,data_square_val);
```

1166

```

1167 clear data_impulse_blade1 data_impulse_blade2 data_impulse_blade3
      data_impulse_blade4 data_impulse_statorx data_impulse_statory...
1168     data_step_blade1 data_step_blade2 data_step_blade3 data_step_blade4
      data_step_statorx data_step_statory ...
1169     data_chirp_blade1 data_chirp_blade2 data_chirp_blade3 data_chirp_blade4
      data_chirp_statorx data_chirp_statory ...
1170     data_prbs_blade1 data_prbs_blade2 data_prbs_blade3 data_prbs_blade4
      data_prbs_statorx data_prbs_statory ...
1171     data_noise data_random_est data_square_est data_random_val data_square_val
      filter
1172
1173 static_est.una =
      {'Mag_blade1','Mag_blade2','Mag_blade3','Mag_blade4','Mag_statorx','Mag_statory'};
1174 static_val.una =
      {'Mag_blade1','Mag_blade2','Mag_blade3','Mag_blade4','Mag_statorx','Mag_statory'};
1175 static_est.yna = {'Blade1','Blade2','Blade3','Blade4','Statorx','Statory'};
1176 static_val.yna = {'Blade1','Blade2','Blade3','Blade4','Statorx','Statory'};
1177 static_est.exp = {'Step1','Step2','Step3','Step4','StepX','StepY',...
      'Chirp1','Chirp2','Chirp3','Chirp4','ChirpX','ChirpY',...
      'PRBS1','PRBS2','PRBS3','PRBS4','PRBSX','PRBSY',...
      'Noise','Random','Square'};
1181 static_val.exp = {'Step1','Step2','Step3','Step4','StepX','StepY',...
      'Chirp1','Chirp2','Chirp3','Chirp4','ChirpX','ChirpY',...
      'PRBS1','PRBS2','PRBS3','PRBS4','PRBSX','PRBSY',...
      'Noise','Random','Square'};
1185
1186 static_est.uu = {'Volts','Volts','Volts','Volts','Volts','Volts'};
1187 static_val.uu = {'Volts','Volts','Volts','Volts','Volts','Volts'};
1188 static_est.yu = {'Volts','Volts','Volts','Volts','Volts','Volts'};
1189 static_val.yu = {'Volts','Volts','Volts','Volts','Volts','Volts'};
1190
1191
1192 data_prefiltering_rotating.m
1193
1194 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Data prefiltering rotating experiments%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1195 close all
1196 clc
1197
1198 %%Loading the chopped data of the experiments into iddata objects
1199 run('data_handle_rotating.m')
1200
1201 %%Pre-filter of experiments as advised by the advice command run
1202 %%There are three possible pre-filtering operations to be performed: missing
1203 %%data handling, detrending and filtering
1204 %%First, if any missing data, an interpolation will be done
1205 %%Second, we detrend data and take away offsets of the signals
1206 %%Last, the filtering will be performed in the 50 Hz bandwidth, since it is
1207 %%the operation bandwidth of the system
1208
1209 filter = [0,2*pi*50];
1210
1211 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Impulse%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

1212 %After plotting the impulse responses, it has been seen that they offer
1213 %little information about the system, since the output is essentially noise
1214 %However, they include the offset values for the step experiments
1215 data_impulse_blade1 = misdata(data_impulse_blade1);
1216 offset = mean(data_impulse_blade1.OutputData);
1217 t = getTrend(data_impulse_blade1);
1218 t.OutputOffset = offset;
1219 data_impulse_blade1 = detrend(data_impulse_blade1,t);
1220 data_impulse_blade1.OutputData = idfilt(data_impulse_blade1.OutputData,filter);
1221 clear t offset
1222
1223 data_impulse_blade2 = misdata(data_impulse_blade2);
1224 offset = mean(data_impulse_blade2.OutputData);
1225 t = getTrend(data_impulse_blade2);
1226 t.OutputOffset = offset;
1227 data_impulse_blade2 = detrend(data_impulse_blade2,t);
1228 data_impulse_blade2.OutputData = idfilt(data_impulse_blade2.OutputData,filter);
1229 clear t offset
1230
1231 data_impulse_blade3 = misdata(data_impulse_blade3);
1232 offset = mean(data_impulse_blade3.OutputData);
1233 t = getTrend(data_impulse_blade3);
1234 t.OutputOffset = offset;
1235 data_impulse_blade3 = detrend(data_impulse_blade3,t);
1236 data_impulse_blade3.OutputData = idfilt(data_impulse_blade3.OutputData,filter);
1237 clear t offset
1238
1239 data_impulse_blade4 = misdata(data_impulse_blade4);
1240 offset = mean(data_impulse_blade4.OutputData);
1241 t = getTrend(data_impulse_blade4);
1242 t.OutputOffset = offset;
1243 data_impulse_blade4 = detrend(data_impulse_blade4,t);
1244 data_impulse_blade4.OutputData = idfilt(data_impulse_blade4.OutputData,filter);
1245 clear t offset
1246
1247 data_impulse_statorx = misdata(data_impulse_statorx);
1248 offset = mean(data_impulse_statorx.OutputData);
1249 t = getTrend(data_impulse_statorx);
1250 t.OutputOffset = offset;
1251 data_impulse_statorx = detrend(data_impulse_statorx,t);
1252 data_impulse_statorx.OutputData = idfilt(data_impulse_statorx.OutputData,filter);
1253 clear t offset
1254
1255 data_impulse_statory = misdata(data_impulse_statory);
1256 offset = mean(data_impulse_statory.OutputData);
1257 t = getTrend(data_impulse_statory);
1258 t.OutputOffset = offset;
1259 data_impulse_statory = detrend(data_impulse_statory,t);
1260 data_impulse_statory.OutputData = idfilt(data_impulse_statory.OutputData,filter);
1261 clear t offset
1262
1263 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Step%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

1264 data_step_blade1 = misdata(data_step_blade1);
1265 offset = mean(data_step_blade1.OutputData);
1266 t = getTrend(data_step_blade1);
1267 t.OutputOffset = offset;
1268 data_step_blade1 = detrend(data_step_blade1,t);
1269 data_step_blade1.OutputData = idfilt(data_step_blade1.OutputData,filter);
1270 clear t offset
1271
1272 data_step_blade2 = misdata(data_step_blade2);
1273 offset = mean(data_step_blade2.OutputData);
1274 t = getTrend(data_step_blade2);
1275 t.OutputOffset = offset;
1276 data_step_blade2 = detrend(data_step_blade2,t);
1277 data_step_blade2.OutputData = idfilt(data_step_blade2.OutputData,filter);
1278 clear t offset
1279
1280 data_step_blade3 = misdata(data_step_blade3);
1281 offset = mean(data_step_blade3.OutputData);
1282 t = getTrend(data_step_blade3);
1283 t.OutputOffset = offset;
1284 data_step_blade3 = detrend(data_step_blade3,t);
1285 data_step_blade3.OutputData = idfilt(data_step_blade3.OutputData,filter);
1286 clear t offset
1287
1288 data_step_blade4 = misdata(data_step_blade4);
1289 offset = mean(data_step_blade4.OutputData);
1290 t = getTrend(data_step_blade4);
1291 t.OutputOffset = offset;
1292 data_step_blade4 = detrend(data_step_blade4,t);
1293 data_step_blade4.OutputData = idfilt(data_step_blade4.OutputData,filter);
1294 clear t offset
1295
1296 data_step_statorx = misdata(data_step_statorx);
1297 offset = mean(data_step_statorx.OutputData);
1298 t = getTrend(data_step_statorx);
1299 t.OutputOffset = offset;
1300 data_step_statorx = detrend(data_step_statorx,t);
1301 data_step_statorx.OutputData = idfilt(data_step_statorx.OutputData,filter);
1302 clear t offset
1303
1304 data_step_statory = misdata(data_step_statory);
1305 offset = mean(data_step_statory.OutputData);
1306 t = getTrend(data_step_statory);
1307 t.OutputOffset = offset;
1308 data_step_statory = detrend(data_step_statory,t);
1309 data_step_statory.OutputData = idfilt(data_step_statory.OutputData,filter);
1310 clear t offset
1311
1312 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Chirp%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1313 data_chirp_blade1 = misdata(data_chirp_blade1);
1314 offset = mean(data_chirp_blade1.OutputData(1:300,:));
1315 t = getTrend(data_chirp_blade1);

```

```
1316 t.OutputOffset = offset;
1317 data_chirp_blade1 = detrend(data_chirp_blade1,t);
1318 data_chirp_blade1.OutputData = idfilt(data_chirp_blade1.OutputData,filter);
1319 clear t offset
1320
1321 data_chirp_blade2 = misdata(data_chirp_blade2);
1322 offset = mean(data_chirp_blade2.OutputData(1:300,:));
1323 t = getTrend(data_chirp_blade2);
1324 t.OutputOffset = offset;
1325 data_chirp_blade2 = detrend(data_chirp_blade2,t);
1326 data_chirp_blade2.OutputData = idfilt(data_chirp_blade2.OutputData,filter);
1327 clear t offset
1328
1329 data_chirp_blade3 = misdata(data_chirp_blade3);
1330 offset = mean(data_chirp_blade3.OutputData(1:300,:));
1331 t = getTrend(data_chirp_blade3);
1332 t.OutputOffset = offset;
1333 data_chirp_blade3 = detrend(data_chirp_blade3,t);
1334 data_chirp_blade3.OutputData = idfilt(data_chirp_blade3.OutputData,filter);
1335 clear t offset
1336
1337 data_chirp_blade4 = misdata(data_chirp_blade4);
1338 offset = mean(data_chirp_blade4.OutputData(1:300,:));
1339 t = getTrend(data_chirp_blade4);
1340 t.OutputOffset = offset;
1341 data_chirp_blade4 = detrend(data_chirp_blade4,t);
1342 data_chirp_blade4.OutputData = idfilt(data_chirp_blade4.OutputData,filter);
1343 clear t offset
1344
1345 data_chirp_statorx = misdata(data_chirp_statorx);
1346 offset = mean(data_chirp_statorx.OutputData(1:300,:));
1347 t = getTrend(data_chirp_statorx);
1348 t.OutputOffset = offset;
1349 data_chirp_statorx = detrend(data_chirp_statorx,t);
1350 data_chirp_statorx.OutputData = idfilt(data_chirp_statorx.OutputData,filter);
1351 clear t offset
1352
1353 data_chirp_statory = misdata(data_chirp_statory);
1354 offset = mean(data_chirp_statory.OutputData(1:300,:));
1355 t = getTrend(data_chirp_statory);
1356 t.OutputOffset = offset;
1357 data_chirp_statory = detrend(data_chirp_statory,t);
1358 data_chirp_statory.OutputData = idfilt(data_chirp_statory.OutputData,filter);
1359 clear t offset
1360
1361 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%PRBS%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1362 data_prbs_blade1 = misdata(data_prbs_blade1);
1363 offset = mean(data_prbs_blade1.OutputData(1:300,:));
1364 t = getTrend(data_prbs_blade1);
1365 t.OutputOffset = offset;
1366 data_prbs_blade1 = detrend(data_prbs_blade1,t);
1367 data_prbs_blade1.OutputData = idfilt(data_prbs_blade1.OutputData,filter);
```

```

1368 clear t offset
1369
1370 data_prbs_blade2 = misdata(data_prbs_blade2);
1371 offset = mean(data_prbs_blade2.OutputData(1:300,:));
1372 t = getTrend(data_prbs_blade2);
1373 t.OutputOffset = offset;
1374 data_prbs_blade2 = detrend(data_prbs_blade2,t);
1375 data_prbs_blade2.OutputData = idfilt(data_prbs_blade2.OutputData,filter);
1376 clear t offset
1377
1378 data_prbs_blade3 = misdata(data_prbs_blade3);
1379 offset = mean(data_prbs_blade3.OutputData(1:300,:));
1380 t = getTrend(data_prbs_blade3);
1381 t.OutputOffset = offset;
1382 data_prbs_blade3 = detrend(data_prbs_blade3,t);
1383 data_prbs_blade3.OutputData = idfilt(data_prbs_blade3.OutputData,filter);
1384 clear t offset
1385
1386 data_prbs_blade4 = misdata(data_prbs_blade4);
1387 offset = mean(data_prbs_blade4.OutputData(1:300,:));
1388 t = getTrend(data_prbs_blade4);
1389 t.OutputOffset = offset;
1390 data_prbs_blade4 = detrend(data_prbs_blade4,t);
1391 data_prbs_blade4.OutputData = idfilt(data_prbs_blade4.OutputData,filter);
1392 clear t offset
1393
1394 data_prbs_statorx = misdata(data_prbs_statorx);
1395 offset = mean(data_prbs_statorx.OutputData(1:300,:));
1396 t = getTrend(data_prbs_statorx);
1397 t.OutputOffset = offset;
1398 data_prbs_statorx = detrend(data_prbs_statorx,t);
1399 data_prbs_statorx.OutputData = idfilt(data_prbs_statorx.OutputData,filter);
1400 clear t offset
1401
1402 data_prbs_statory = misdata(data_prbs_statory);
1403 offset = mean(data_prbs_statory.OutputData(1:300,:));
1404 t = getTrend(data_prbs_statory);
1405 t.OutputOffset = offset;
1406 data_prbs_statory = detrend(data_prbs_statory,t);
1407 data_prbs_statory.OutputData = idfilt(data_prbs_statory.OutputData,filter);
1408 clear t offset
1409
1410 %%%%%%%%%%%%%Random%%%%%%%%%%%%
1411 data_random_est = misdata(data_random_est);
1412 offset = mean(data_random_est.OutputData);
1413 t = getTrend(data_random_est);
1414 t.OutputOffset = offset;
1415 data_random_est = detrend(data_random_est,t);
1416 data_random_est.OutputData = idfilt(data_random_est.OutputData,filter);
1417 clear t offset
1418
1419 data_random_val = misdata(data_random_val);

```



```

1420 offset = mean(data_random_val.OutputData);
1421 t = getTrend(data_random_val);
1422 t.OutputOffset = offset;
1423 data_random_val = detrend(data_random_val,t);
1424 data_random_val.OutputData = idfilt(data_random_val.OutputData,filter);
1425 clear t offset
1426
1427 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Square%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1428 data_square_est = misdata(data_square_est);
1429 offset = mean(data_square_est.OutputData(1:300,:));
1430 t = getTrend(data_square_est);
1431 t.OutputOffset = offset;
1432 data_square_est = detrend(data_square_est,t);
1433 data_square_est.OutputData = idfilt(data_square_est.OutputData,filter);
1434 clear t offset
1435
1436 data_square_val = misdata(data_square_val);
1437 offset = mean(data_square_val.OutputData(1:300,:));
1438 t = getTrend(data_square_val);
1439 t.OutputOffset = offset;
1440 data_square_val = detrend(data_square_val,t);
1441 data_square_val.OutputData = idfilt(data_square_val.OutputData,filter);
1442 clear t offset
1443
1444 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Noise%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1445 data_noise = misdata(data_noise);
1446 offset = mean(data_noise.OutputData);
1447 t = getTrend(data_noise);
1448 t.OutputOffset = offset;
1449 data_noise = detrend(data_noise,t);
1450 data_noise.OutputData = idfilt(data_noise.OutputData,filter);
1451 clear t offset
1452
1453 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Merging all experiments in one iddata object%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
1454 %%The experiments are merged because they have been performed in the same
1455 %%conditions, having similar signal to noise ratios
1456
1457 rotating_est =
    merge(data_chirp_blade1,data_chirp_blade2,data_chirp_blade3,data_chirp_blade4,data_chirp_statorx,
1458         data_prbs_blade1,data_prbs_blade2,data_prbs_blade3,data_prbs_blade4,data_prbs_statorx,d
1459         data_noise,data_random_est,data_square_est);
1460
1461 rotating_val =
    merge(data_chirp_blade1,data_chirp_blade2,data_chirp_blade3,data_chirp_blade4,data_chirp_statorx,
1462         data_prbs_blade1,data_prbs_blade2,data_prbs_blade3,data_prbs_blade4,data_prbs_statorx,d
1463         data_noise,data_random_val,data_square_val);
1464
1465 clear data_impulse_blade1 data_impulse_blade2 data_impulse_blade3
    data_impulse_blade4 data_impulse_statorx data_impulse_statory...
1466 data_step_blade1 data_step_blade2 data_step_blade3 data_step_blade4
    data_step_statorx data_step_statory ...

```

```

1467     data_chirp_blade1 data_chirp_blade2 data_chirp_blade3 data_chirp_blade4
        data_chirp_statorx data_chirp_statory ...
1468     data_prbs_blade1 data_prbs_blade2 data_prbs_blade3 data_prbs_blade4
        data_prbs_statorx data_prbs_statory ...
1469     data_noise data_random_est data_square_est data_random_val data_square_val
        filter
1470
1471     rotating_est.una =
        {'Mag_blade1', 'Mag_blade2', 'Mag_blade3', 'Mag_blade4', 'Mag_statorx', 'Mag_statory'};
1472     rotating_val.una =
        {'Mag_blade1', 'Mag_blade2', 'Mag_blade3', 'Mag_blade4', 'Mag_statorx', 'Mag_statory'};
1473     rotating_est.yna = {'Blade1', 'Blade2', 'Blade3', 'Blade4', 'Statorx', 'Statory'};
1474     rotating_val.yna = {'Blade1', 'Blade2', 'Blade3', 'Blade4', 'Statorx', 'Statory'};
1475     rotating_est.exp = {'Chirp1', 'Chirp2', 'Chirp3', 'Chirp4', 'ChirpX', 'ChirpY', ...
1476                        'PRBS1', 'PRBS2', 'PRBS3', 'PRBS4', 'PRBSX', 'PRBSY', ...
1477                        'Noise', 'Random', 'Square'};
1478     rotating_val.exp = {'Chirp1', 'Chirp2', 'Chirp3', 'Chirp4', 'ChirpX', 'ChirpY', ...
1479                        'PRBS1', 'PRBS2', 'PRBS3', 'PRBS4', 'PRBSX', 'PRBSY', ...
1480                        'Noise', 'Random', 'Square'};
1481
1482     rotating_est.uu = {'Volts', 'Volts', 'Volts', 'Volts', 'Volts', 'Volts'};
1483     rotating_val.uu = {'Volts', 'Volts', 'Volts', 'Volts', 'Volts', 'Volts'};
1484     rotating_est.yu = {'Volts', 'Volts', 'Volts', 'Volts', 'Volts', 'Volts'};
1485     rotating_val.yu = {'Volts', 'Volts', 'Volts', 'Volts', 'Volts', 'Volts'};

```

A.2 Time variable estimation

```

1  dynamicTF_TVP.m
2
3  %%%%%%%%%%%%%%% TVP estimation using DTF %%%%%%%%%%%%%%%
4  clear
5  clc
6
7  %%Load the rotating data set
8  data_prefiltering_rotating
9
10 %%Separating the system in MISO data sets
11 data = getexp(rotating_est, 'Random');
12 N = 200; %number of samples in a period
13 y1 = data.OutputData(1:N,1);
14 y2 = data.OutputData(1:N,2);
15 y3 = data.OutputData(1:N,3);
16 y4 = data.OutputData(1:N,4);
17 y5 = data.OutputData(1:N,5);
18 y6 = data.OutputData(1:N,6);
19 u = data.OutputData(1:N,:);
20
21
22 %%Optimize Noise Variance Ratio (NVR) hyper-parameters
23 na = 6; %no of parameters in A polynomial
24 nb = 6; %no of parameters in each B polynomial

```

```
25 nn = [na nb*ones(1,6) zeros(1,6)]; %no of parameter in A, in each B polynomial
    and delay in each input
26 nvrc = -2*ones(1,na+nb*6); %optimise all nvr
27 TVP = ones(1,na+nb*6); %since all parameters are periodic, integrated random
    walk (IRW) selected for all
28
29 nvr1 = dtfmopt(y1,u,nn,TVP,[],nvrc);
30 nvr2 = dtfmopt(y2,u,nn,TVP,[],nvrc);
31 nvr3 = dtfmopt(y3,u,nn,TVP,[],nvrc);
32 nvr4 = dtfmopt(y4,u,nn,TVP,[],nvrc);
33 nvr5 = dtfmopt(y5,u,nn,TVP,[],nvrc);
34 nvr6 = dtfmopt(y6,u,nn,TVP,[],nvrc);
35
36 %%Once the NVR optimised, estimating the DTF parameters
37
38 [tfs1, fit1, fitse1, par1, parse1, e1]=dtfm(y1,u,nn,TVP,nvr1);
39 [tfs2, fit2, fitse2, par2, parse2, e2]=dtfm(y2,u,nn,TVP,nvr2);
40 [tfs3, fit3, fitse3, par3, parse3, e3]=dtfm(y3,u,nn,TVP,nvr3);
41 [tfs4, fit4, fitse4, par4, parse4, e4]=dtfm(y4,u,nn,TVP,nvr4);
42 [tfs5, fit5, fitse5, par5, parse5, e5]=dtfm(y5,u,nn,TVP,nvr5);
43 [tfs6, fit6, fitse6, par6, parse6, e6]=dtfm(y6,u,nn,TVP,nvr6);
44
45 %%When the parameters are estimated, the model is built and their time
46 %%variation is plotted along with their confidence intervals
47 subplot(3,2,1)
48 plot(tfs1)
49 title('Output Blade 1')
50 ylabel('Volts(V)')
51 subplot(3,2,2)
52 plot(tfs2)
53 title('Output Blade 2')
54 ylabel('Volts(V)')
55 subplot(3,2,3)
56 plot(tfs3)
57 title('Output Blade 3')
58 ylabel('Volts(V)')
59 subplot(3,2,4)
60 plot(tfs4)
61 title('Output Blade 4')
62 ylabel('Volts(V)')
63 subplot(3,2,5)
64 plot(tfs5)
65 title('Output Stator in X direction')
66 ylabel('Volts(V)')
67 xlabel('Samples')
68 subplot(3,2,6)
69 plot(tfs6)
70 title('Output Stator in Y direction')
71 ylabel('Volts(V)')
72 xlabel('Samples')
73 suptitle('Output response with estimated TVP system')
```

APPENDIX B: MATLAB CODE FOR CONTROL

B DESIGN

B.1 PID controllers design

```
1 clear
2 clc
3
4 %% Load the estimated uncoupled plants original and reduced and the
5 %%controllers
6 load('static_model.mat')
7 load('PID Controllers\C1.mat')
8 load('PID Controllers\C2.mat')
9 load('PID Controllers\C3.mat')
10 load('PID Controllers\C4.mat')
11 load('PID Controllers\C5.mat')
12 load('PID Controllers\C6.mat')
13
14 sys_tf=tf(ss(static_model.A,static_model.B,static_model.C,static_model.D));
15 K_sensor = (1.5*10-3)/4; %m/V
16 P1 = sys_tf(1,1)*K_sensor;
17 P2 = sys_tf(2,2)*K_sensor;
18 P3 = sys_tf(3,3)*(-K_sensor); %negative plant estimated, sensor in negative
    direction
19 P4 = sys_tf(4,4)*(-K_sensor); %negative plant estimated, sensor in negative
    direction
20 P5 = sys_tf(5,5)*(-K_sensor); %negative plant estimated, sensor in negative
    direction
21 P6 = sys_tf(6,6)*K_sensor;
22
23 clear static_model sys_tf
24
25 %% Define disturbances applied
26 step_u=20; %V
27 T_u=5; %starting time of step (s)
28 T_u2=T_u+1; %finishing time of step (s)
29 Tsim=25; %time of simulation (s)
30 N = 100; %filtering factor for the PID controllers
31
32 %% PID 1
33 Ps=P1;
34 [Psz,Psp]=tfdata(Ps);
35 Kp=C1.Kp;
36 Ki=C1.Ki;
```

```
37 Kd=C1.Kd;
38 sim('simulink_PID');
39 time1=Output.Time;
40 y1_uncontrolled=Output.data(:,1);
41 y1_PID=Output.data(:,2);
42 u1=Control_signal.data;
43
44 figure
45 subplot(2,1,1)
46 plot(time1,y1_uncontrolled,'r')
47 hold on
48 plot(time1,y1_PID,'b')
49 hold off
50 legend('Uncontrolled output (mm)','Controlled output(mm)')
51 title('Identified plant: Blade 1')
52 subplot(2,1,2)
53 plot(time1,u1)
54 legend('Control signal (Volts)')
55
56 %% Plant 2
57 Ps=P2;
58 [Psz,Psp]=tfdata(Ps);
59 Kp=C2.Kp;
60 Ki=C2.Ki;
61 Kd=C2.Kd;
62 sim('simulink_PID');
63 time2=Output.Time;
64 y2_uncontrolled=Output.data(:,1);
65 y2_PID=Output.data(:,2);
66 u2=Control_signal.data;
67
68 figure
69 subplot(2,1,1)
70 plot(time2,y2_uncontrolled,'r')
71 hold on
72 plot(time2,y2_PID,'b')
73 hold off
74 legend('Uncontrolled output (mm)','Controlled output(mm)')
75 title('Identified plant: Blade 2')
76 subplot(2,1,2)
77 plot(time2,u2)
78 legend('Control signal (Volts)')
79
80 %% Plant 3
81 Ps=P3;
82 [Psz,Psp]=tfdata(Ps);
83 Kp=C3.Kp;
84 Ki=C3.Ki;
85 Kd=C3.Kd;
86 sim('simulink_PID');
87 time3=Output.Time;
88 y3_uncontrolled=Output.data(:,1);
```

```

89 y3_PID=Output.data(:,2);
90 u3=Control_signal.data;
91
92 figure
93 subplot(2,1,1)
94 plot(time3,y3_uncontrolled,'r')
95 hold on
96 plot(time3,y3_PID,'b')
97 hold off
98 legend('Uncontrolled output (mm)','Controlled output(mm)')
99 title('Identified plant: Blade 3')
100 subplot(2,1,2)
101 plot(time3,u3)
102 legend('Control signal (Volts)')
103
104 %% Plant 4
105 Ps=P4;
106 [Psz,Psp]=tfdata(Ps);
107 Kp=C4.Kp;
108 Ki=C4.Ki;
109 Kd=C4.Kd;
110 sim('simulink_PID');
111 time4=Output.Time;
112 y4_uncontrolled=Output.data(:,1);
113 y4_PID=Output.data(:,2);
114 u4=Control_signal.data;
115
116 figure
117 subplot(2,1,1)
118 plot(time4,y4_uncontrolled,'r')
119 hold on
120 plot(time4,y4_PID,'b')
121 hold off
122 legend('Uncontrolled output (mm)','Controlled output(mm)')
123 title('Identified plant: Blade 4')
124 subplot(2,1,2)
125 plot(time4,u4)
126 legend('Control signal (Volts)')
127
128 %% Plant 5
129 Ps=P5;
130 [Psz,Psp]=tfdata(Ps);
131 Kp=C5.Kp;
132 Ki=C5.Ki;
133 Kd=C5.Kd;
134 sim('simulink_PID');
135 time5=Output.Time;
136 y5_uncontrolled=Output.data(:,1);
137 y5_PID=Output.data(:,2);
138 u5=Control_signal.data;
139
140 figure

```

```
141 subplot(2,1,1)
142 plot(time5,y5_uncontrolled,'r')
143 hold on
144 plot(time5,y5_PID,'b')
145 hold off
146 legend('Uncontrolled output (mm)','Controlled output(mm)')
147 title('Identified plant: stator x direction')
148 subplot(2,1,2)
149 plot(time5,u5)
150 legend('Control signal (Volts)')
151
152 %% Plant 6
153 Ps=P6;
154 [Psz,Psp]=tfdata(Ps);
155 Kp=C6.Kp;
156 Ki=C6.Ki;
157 Kd=C6.Kd;
158 sim('simulink_PID');
159 time6=Output.Time;
160 y6_uncontrolled=Output.data(:,1);
161 y6_PID=Output.data(:,2);
162 u6=Control_signal.data;
163
164 figure
165 subplot(2,1,1)
166 plot(time6,y6_uncontrolled,'r')
167 hold on
168 plot(time6,y6_PID,'b')
169 hold off
170 legend('Uncontrolled output (mm)','Controlled output(mm)')
171 title('Identified plant: stator y direction')
172 subplot(2,1,2)
173 plot(time6,u6)
174 legend('Control signal (Volts)')
```

B.2 LQR controller design

```
1 clear
2 clc
3
4 %%Load the identified model
5 load('static_model.mat');
6 A=static_model.A;
7 B=static_model.B;
8 C=static_model.C;
9 D=static_model.D;
10 sys=ss(A,B,C,D);
11
12 clear static_model
13
14 %%Data for the simulation and design
```

```

15 Tstep = 1;
16 Tstep2 = Tstep+5;
17 Step_LQR = 20;
18 Tsim = 25;
19
20 %% Observer design
21 eig_ori = eig(A);
22 eig_obsrv = 10*eig_ori;
23 L = place(A',C',eig_obsrv);
24
25 %% LQR design coupled
26 Q = eye(6)*2e8;
27 R = eye(6);
28 K = lqr(sys,Q,R);
29 sim('simulink_LQR');
30 time = tout;
31 y_uncon = y_uncontrolled.data;
32 y = y.data;
33 u = u.data;
34 x = x_hat.data;
35
36 %% LQR design uncoupled
37 Q = eye(6)*2e5;
38 R = diag([0.001 0.001 0.001 0.001 1 1]);
39 K = lqr(sys,Q,R);
40 sim('simulink_LQR');
41 time = tout;
42 y_uncon = y_uncontrolled.data;
43 y = y.data;
44 u = u.data;
45 x = x_hat.data;
46
47 %% Simulate the comparison of LQR controller with the same identified open loop
    plant
48 sim('simulink_LQR')
49
50 %% Plots of every output
51 figure
52 subplot(2,1,1)
53 p1 = plot(time,y_uncon(:,1),'r');
54 hold on
55 p2 = plot(time,y_uncon(:,2),'r');
56 p3 = plot(time,y_uncon(:,3),'r');
57 p4 = plot(time,y_uncon(:,4),'r');
58 p5 = plot(time,y_uncon(:,5),'r');
59 p6 = plot(time,y_uncon(:,6),'r');
60 p7 = plot(time,y,'b');
61 hold off
62 legend([p1;p7], 'Uncontrolled output (mm)', 'Controlled output(mm)')
63 title('Full State Feedback Controller for coupled system')
64 subplot(2,1,2)
65 plot(time,u)

```



```
66 title('Control signal (Volts)')
```

B.3 Online identification design

```
1 clear
2 clc
3
4 %% Run parameters for the simulation of the plant
5 main_plant
6
7 %% Calculating the operation point wanted to reach (reference)
8 Tsim = 10; %time of simulation (s)
9 T_step = 0; %no step on rotating speed
10 theta_d_ini = 0; %no rotating speed for the reference
11 flag_u = 0; %no input for the reference
12 pow = 0.001; %spectral density power for input noises
13 flag_noise = 0; %no measurement noise for the reference
14 flag_est = 0;
15 ref = zeros(1,6);
16 B_q = [1 1 1 1 1 1]';
17 A_q = [1];
18 index = 1;
19 sim('OnlineID')
20 ref = [x_open(end) y_open(end) d1_open(end) d2_open(end) d3_open(end)
        d4_open(end)];
21
22 %% Calculating model orders for the wanted MISO system
23 index = 1; %index of the MISO system wanted to identify (1-6)
24 [A_d,B_d,C_d,D_d] = dlinmod('PlantLin',Ts);
25 sys_ss = ss(A_d,B_d,C_d,D_d,Ts);
26 sys_arx = idpoly(sys_ss);
27 A_q = sys_arx.a{index,index};
28 B_q =
    [sys_arx.b{index,1};sys_arx.b{index,2};sys_arx.b{index,3};sys_arx.b{index,4};sys_arx.b{index,5};
29
30 clear sys_ss num A_d B_d C_d D_d
31
32 %% Specify the wanted parameters for the system
33 Tsim = 10; %time of simulation (s)
34 T_step = 3; %step on rotating speed
35 theta_d_ini = 5*2*pi; %rotating speed
36 flag_u = 1; %include input
37 flag_est = 1; %estimate online
38
39 %% Simulating the parameter estimation
40 sim('OnlineID')
41
42 %% Managing the output data of the online estimation
43 par_A = squeeze(parameters_A);
44 par_B1 = squeeze(parameters_B(1,:,:));
45 par_B2 = squeeze(parameters_B(2,:,:));
```

```
46 par_B3 = squeeze(parameters_B(3, :, :));
47 par_B4 = squeeze(parameters_B(4, :, :));
48 par_B5 = squeeze(parameters_B(5, :, :));
49 par_B6 = squeeze(parameters_B(6, :, :));
50 pred_error(:, index) = error;
51 time = 0:Ts:(length(error)-1)*Ts;
52
53 %% Plotting the estimated data
54 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Plotting online estimation data%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
55 time = 0:Ts:(length(error)-1)*Ts;
56
57 %% Plot the prediction errors
58 figure
59 subplot(2,3,1)
60 plot(time, pred_error(:, 1))
61 xlabel('Time (s)')
62 ylabel('Error')
63 title('MISO stator in x direction')
64 subplot(2,3,2)
65 plot(time, pred_error(:, 2))
66 xlabel('Time (s)')
67 ylabel('Error')
68 title('MISO stator in y direction')
69 subplot(2,3,3)
70 plot(time, pred_error(:, 3))
71 xlabel('Time (s)')
72 ylabel('Error')
73 title('MISO stator in blade 1')
74 subplot(2,3,4)
75 plot(time, pred_error(:, 4))
76 xlabel('Time (s)')
77 ylabel('Error')
78 title('MISO stator in blade 2')
79 subplot(2,3,5)
80 plot(time, pred_error(:, 5))
81 xlabel('Time (s)')
82 ylabel('Error')
83 title('MISO stator in blade 3')
84 subplot(2,3,6)
85 plot(time, pred_error(:, 6))
86 xlabel('Time (s)')
87 ylabel('Error')
88 title('MISO stator in blade 4')
89 suptitle('Prediction Errors')
```

APPENDIX C: MATLAB CODE FOR THE C IMPLEMENTATION OF CONTROLLERS

C.1 PID controllers implementation

```
1 clear
2 clc
3
4 %% Run the plant parameters for the simulation and load controllers
5 run('main_plant.m');
6 load('PID Controllers\C1.mat')
7 load('PID Controllers\C2.mat')
8 load('PID Controllers\C3.mat')
9 load('PID Controllers\C4.mat')
10 load('PID Controllers\Cx.mat')
11 load('PID Controllers\Cy.mat')
12 theta_d_ini = 5*2*pi; %rotation speed, in rad/s
13 N = 10000; %filtering parameter for the PIDs
14 flag = 1; %flag for coupled (1) or uncoupled (0) simulation
15
16 %% Define disturbances applied
17 step_u=20; %N
18 step_open=20; %N
19 T_u=5; %starting time of step (s)
20 T_u2=T_u+1; %finishing time of step (s)
21 Tsim=10; %time of simulation (s)
22
23 %% Calculating the operation point wanted to reach (reference)
24 flag_d = [0 0 0 0 0 0]';
25 sim('PlantSim')
26 ref = [x_open(end) y_open(end) d1_open(end) d2_open(end) d3_open(end)
        d4_open(end)];
27
28 %% Flag for including the disturbance in selected input channel
29 flag_d = [0 0 0 0 0 0]'; %flag for including the disturbance on input channel
30
31 %% Simulation of the PID controllers with the real plant
32 sim('PID_test')
33
34 %% Simulation of the plant in open loop for comparison
35 sim('PlantSim')
36
37 %% Plotting of the outputs and control signals
38 plot_sim
```

C.2 LQR controllers implementation

```
1 clear
2 clc
3
4 %% Run the plant parameters for the simulation and designed controllers
5 run('main_plant.m');
6 load('K_coupled.mat')
7 load('K_uncoupled.mat')
8 load('observer.mat')
9
10 theta_d_ini = 5*2*pi; %rotation speed, in rad/s
11 flag = 1; %flag for coupled (1) or uncoupled (0) simulation
12
13 if flag==1
14     K = K_coupled;
15 else
16     K = K_uncoupled;
17 end
18
19 %% Define disturbances applied
20 step_u=10; %N
21 step_open=20; %N
22 T_u=5; %starting time of step (s)
23 T_u2=T_u+1; %finishing time of step (s)
24 Tsim=10; %time of simulation (s)
25
26 %% Calculating the operation point wanted to reach (reference)
27 flag_d = [0 0 0 0 0 0]';
28 sim('PlantSim')
29 ref = [x_open(end) y_open(end) d1_open(end) d2_open(end) d3_open(end)
        d4_open(end)];
30
31 %% Flag for including the disturbance in selected input channel
32 flag_d = [0 0 0 0 0 0]'; %flag for including the disturbance on input channel
33
34 %% Simulation of the PID controllers with the real plant
35 sim('LQR_test')
36
37 %% Simulation of the plant in open loop for comparison
38 sim('PlantSim')
39
40 %% Plotting of the outputs and control signals
41 plot_sim
```

C.3 Adaptive controller implementation

```
1 clear
2 clc
3
4 %% Run parameters for the simulation of the plant
```

```
5 main_plant
6
7 %% Calculating the operation point wanted to reach (reference)
8 Tsim = 10; %time of simulation (s)
9 theta_d_ini = 0; %no rotating speed for the reference
10 sim('PlantSim')
11 ref = [x_open(end) y_open(end) d1_open(end) d2_open(end) d3_open(end)
        d4_open(end)];
12
13 %% Online identification setup
14 [A_d,B_d,C_d,D_d] = linmod('PlantLin');
15 sys_ss_c = ss(A_d,B_d,C_d,D_d);
16 sys_ss_d = c2d(sys_ss_c,Ts,'zoh');
17 sys_arx = idpoly(sys_ss_d);
18 A1_q = sys_arx.a{1,1};
19 B1_q =
    [sys_arx.b{1,1};sys_arx.b{1,2};sys_arx.b{1,3};sys_arx.b{1,4};sys_arx.b{1,5};sys_arx.b{1,6}];
20 A2_q = sys_arx.a{2,2};
21 B2_q =
    [sys_arx.b{2,1};sys_arx.b{2,2};sys_arx.b{2,3};sys_arx.b{2,4};sys_arx.b{2,5};sys_arx.b{2,6}];
22 A3_q = sys_arx.a{3,3};
23 B3_q =
    [sys_arx.b{3,1};sys_arx.b{3,2};sys_arx.b{3,3};sys_arx.b{3,4};sys_arx.b{3,5};sys_arx.b{3,6}];
24 A4_q = sys_arx.a{4,4};
25 B4_q =
    [sys_arx.b{4,1};sys_arx.b{4,2};sys_arx.b{4,3};sys_arx.b{4,4};sys_arx.b{4,5};sys_arx.b{4,6}];
26 A5_q = sys_arx.a{5,5};
27 B5_q =
    [sys_arx.b{5,1};sys_arx.b{5,2};sys_arx.b{5,3};sys_arx.b{5,4};sys_arx.b{5,5};sys_arx.b{5,6}];
28 A6_q = sys_arx.a{6,6};
29 B6_q =
    [sys_arx.b{6,1};sys_arx.b{6,2};sys_arx.b{6,3};sys_arx.b{6,4};sys_arx.b{6,5};sys_arx.b{6,6}];
30
31 clear sys_arx sys_ss_c sys_ss_d A_d B_d C_d D_d
32
33 %% Adjustable controller setup
34 nx = 28; %number of states
35 nu = 6; %number of inputs
36 ny = 6; %number of outputs
37
38 %% Parameters for the simulation
39 Tsim = 10;
40 Ts = 0.001;
41 T_step = 3;
42 theta_d_ini = 5*2*pi;
43 flag_est = 1;
44 flag_c = 1;
45 initial_states = [xx_ini;xxd_ini;theta_ini;theta_d_ini;xxi_ini];
46
47 %% Simulation of the controller
48 set_param('adaptive_control','AlgebraicLoopSolver','LineSearch')
49 sim('adaptive_control')
```




```
50 sim('PlantSim')
51
52 %% Plotting of results
53 time = 0:Ts:Ts*(length(error1)-1); %time axis size example
54 time_open = 0:Ts:Ts*(length(x_open)-1); %time axis for open loop simulation
55 figure
56 subplot(2,1,1)
57 plot(time,x)
58 hold on
59 plot(time_open,x_open)
60 hold off
61 legend('Controlled','Open Loop')
62 xlabel('Time (s)')
63 ylabel('Vibration measurement (m)')
64 title('Output stator in X direction')
65 subplot(2,1,2)
66 plot(time,magnet_x)
67 xlabel('Time (s)')
68 ylabel('Control signal (V)')
69 title('Control signal magnet stator in X direction')
70
71 figure
72 subplot(2,1,1)
73 plot(time,y)
74 hold on
75 plot(time_open,y_open)
76 hold off
77 legend('Controlled','Open Loop')
78 xlabel('Time (s)')
79 ylabel('Vibration measurement (m)')
80 title('Output stator in Y direction')
81 subplot(2,1,2)
82 plot(time,magnet_y)
83 xlabel('Time (s)')
84 ylabel('Control signal (V)')
85 title('Control signal magnet stator in Y direction')
86
87 figure
88 subplot(2,1,1)
89 plot(time,d1)
90 hold on
91 plot(time_open,d1_open)
92 hold off
93 legend('Controlled','Open Loop')
94 xlabel('Time (s)')
95 ylabel('Vibration measurement (m)')
96 title('Output blade 1')
97 subplot(2,1,2)
98 plot(time,magnet_blade1)
99 xlabel('Time (s)')
100 ylabel('Control signal (V)')
101 title('Control signal magnet blade 1')
```

```
102
103 figure
104 subplot(2,1,1)
105 plot(time,d2)
106 hold on
107 plot(time_open,d2_open)
108 hold off
109 legend('Controlled','Open Loop')
110 xlabel('Time (s)')
111 ylabel('Vibration measurement (m)')
112 title('Output blade 2')
113 subplot(2,1,2)
114 plot(time,magnet_blade2)
115 xlabel('Time (s)')
116 ylabel('Control signal (V)')
117 title('Control signal magnet blade 2')
118
119 figure
120 subplot(2,1,1)
121 plot(time,d3)
122 hold on
123 plot(time_open,d3_open)
124 hold off
125 legend('Controlled','Open Loop')
126 xlabel('Time (s)')
127 ylabel('Vibration measurement (m)')
128 title('Output blade 3')
129 subplot(2,1,2)
130 plot(time,magnet_blade3)
131 xlabel('Time (s)')
132 ylabel('Control signal (V)')
133 title('Control signal magnet blade 3')
134
135 figure
136 subplot(2,1,1)
137 plot(time,d4)
138 hold on
139 plot(time_open,d4_open)
140 hold off
141 legend('Controlled','Open Loop')
142 xlabel('Time (s)')
143 ylabel('Vibration measurement (m)')
144 title('Output blade 4')
145 subplot(2,1,2)
146 plot(time,magnet_blade4)
147 xlabel('Time (s)')
148 ylabel('Control signal (V)')
149 title('Control signal magnet blade 4')
```

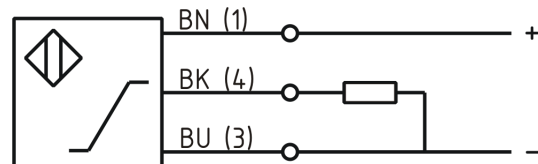
APPENDIX D: HARDWARE COMPONENTS

D DATA SHEETS

[Article Page](#)

Product Description	inductive sensor analogue
Name	KJ4-M12MN50-ANU
Order number	08317144800
Switching distance	4 mm
Mounting	 non shielded
Switch type	 analog
Signal Type	Analog Voltage
Connection	 Cable

Connection diagramm



Dimension (in mm)	fine-pitch thread M12 x 50
-------------------	----------------------------

Technical Specification	
Operating voltage	11 - 35 VDC
Max. ripple	$\leq 10 \%$
No load current	$\leq 5 \text{ mA}$
Switching frequency	400
Operating temperature	-25 ° C to 70 ° C
Temperature drift	+/-5 %
Repeat accuracy	$\leq 1 \%$
Linearity	$\leq 5 \%$
Analog output	Pulso_has1
Digital output	
Protection category	IP67
EMC-level	DIN EN 60947-5-7:2004-06
Material active face	PCB
Termination	2m PCV 3x0,34mm ²

Errors and omissions exceptet

LM747

LM747 Dual Operational Amplifier



Literature Number: SNOS661

LM747 Dual Operational Amplifier

General Description

The LM747 is a general purpose dual operational amplifier. The two amplifiers share a common bias network and power supply leads. Otherwise, their operation is completely independent.

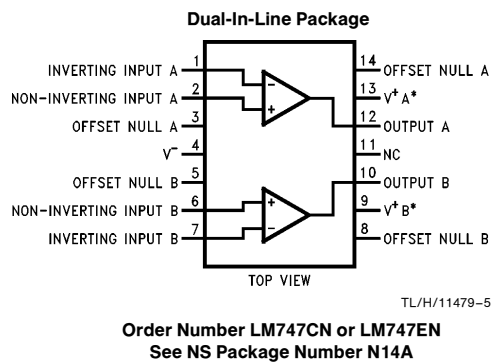
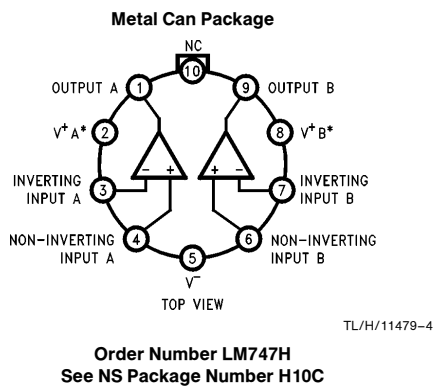
Additional features of the LM747 are: no latch-up when input common mode range is exceeded, freedom from oscillations, and package flexibility.

The LM747C/LM747E is identical to the LM747/LM747A except that the LM747C/LM747E has its specifications guaranteed over the temperature range from 0°C to +70°C instead of -55°C to +125°C.

Features

- No frequency compensation required
- Short-circuit protection
- Wide common-mode and differential voltage ranges
- Low power consumption
- No latch-up
- Balanced offset null

Connection Diagrams



*V+ A and V+ B are internally connected.

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage	
LM747/LM747A	±22V
LM747C/LM747E	±18V
Power Dissipation (Note 1)	800 mW
Differential Input Voltage	±30V

Input Voltage (Note 2)	±15V
Output Short-Circuit Duration	Indefinite
Operating Temperature Range	
LM747/LM747A	–55°C to +125°C
LM747C/LM747E	0°C to +70°C
Storage Temperature Range	–65°C to +150°C
Lead Temperature (Soldering, 10 sec.)	300°C

Electrical Characteristics (Note 3)

Parameter	Conditions	LM747A/LM747E			LM747			LM747C			Units
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Input Offset Voltage	$T_A = 25^\circ\text{C}$ $R_S \leq 10\text{ k}\Omega$ $R_S \leq 50\Omega$		0.8	3.0		1.0	5.0		2.0	6.0	mV
	$R_S \leq 50\Omega$ $R_S \leq 10\text{ k}\Omega$			4.0			6.0			7.5	mV
				15							$\mu\text{V}/^\circ\text{C}$
Average Input Offset Voltage Drift				15							$\mu\text{V}/^\circ\text{C}$
Input Offset Voltage Adjustment Range	$T_A = 25^\circ\text{C}$, $V_S = \pm 20\text{V}$	±10			±15			±15			mV
Input Offset Current	$T_A = 25^\circ\text{C}$	3.0	30		20	200		20	200		nA
			70		85	500			300		nA
Average Input Offset Current Drift				0.5							nA/ $^\circ\text{C}$
Input Bias Current	$T_A = 25^\circ\text{C}$	30	80		80	500		80	500		nA
	$T_{\text{AMIN}} \leq T_A \leq T_{\text{AMAX}}$		0.210			1.5			0.8		μA
Input Resistance	$T_A = 25^\circ\text{C}$, $V_S = \pm 20\text{V}$	1.0	6.0		0.3	2.0		0.3	2.0		M Ω
	$V_S = \pm 20\text{V}$	0.5									M Ω
Input Voltage Range	$T_A = 25^\circ\text{C}$							±12	±13		V
		±12	±13		±12	±13					V
Large Signal Voltage Gain	$T_A = 25^\circ\text{C}$, $R_L \geq 2\text{ k}\Omega$ $V_S = \pm 20\text{V}$, $V_O = \pm 15\text{V}$	50									V/mV
	$V_S = \pm 15\text{V}$, $V_O = \pm 10\text{V}$ $R_L \geq 2\text{ k}\Omega$				50	200		20	200		V/mV
	$V_S = \pm 20\text{V}$, $V_O = \pm 15\text{V}$	32									V/mV
	$V_S = \pm 15\text{V}$, $V_O = \pm 10\text{V}$				25			15			V/mV
	$V_S = \pm 5\text{V}$, $V_O = \pm 2\text{V}$	10									V/mV
Output Voltage Swing	$V_S = \pm 20\text{V}$ $R_L \geq 10\text{ k}\Omega$ $R_L \geq 2\text{ k}\Omega$	±16 ±15									V
	$V_S = \pm 15\text{V}$ $R_L \geq 10\text{ k}\Omega$ $R_L \geq 2\text{ k}\Omega$				±12 ±10	±14 ±13		±12 ±10	±14 ±13		V
Output Short Circuit Current	$T_A = 25^\circ\text{C}$	10 10	25	35 40		25			25		mA
Common-Mode Rejection Ratio	$R_S \leq 10\text{ k}\Omega$, $V_{\text{CM}} = \pm 12\text{V}$				70	90		70	90		dB
	$R_S \leq 50\text{ k}\Omega$, $V_{\text{CM}} = \pm 12\text{V}$	80	95								

Electrical Characteristics (Note 3) (Continued)

Parameter	Conditions	LM747A/LM747E			LM747			LM747C			Units
		Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Supply Voltage Rejection Ratio	$V_S = \pm 20V$ to $V_S = \pm 5V$ $R_S \leq 50\Omega$ $R_S \leq 10k\Omega$	86	96		77	96		77	96		dB
Transient Response Rise Time Overshoot	$T_A = 25^\circ C$, Unity Gain		0.25 6.0	0.8 20		0.3 5			0.3 5		μs %
Bandwidth (Note 4)	$T_A = 25^\circ C$	0.437	1.5								MHz
Slew Rate	$T_A = 25^\circ C$, Unity Gain	0.3	0.7		0.5			0.5			$V/\mu s$
Supply Current/Amp	$T_A = 25^\circ C$			2.5	1.7	2.8		1.7	2.8		mA
Power Consumption/Amp	$T_A = 25^\circ C$ $V_S = \pm 20V$ $V_S = \pm 15V$		80	150		50	85		50	85	mW
LM747A	$V_S = \pm 20V$ $T_A = T_{AMIN}$ $T_A = T_{AMAX}$			165 135							mW
LM747E	$V_S = \pm 20V$ $T_A = T_{AMIN}$ $T_A = T_{AMAX}$			150 150 150							mW
LM747	$V_S = \pm 15V$ $T_A = T_{AMIN}$ $T_A = T_{AMAX}$				60 45	100 75					mW

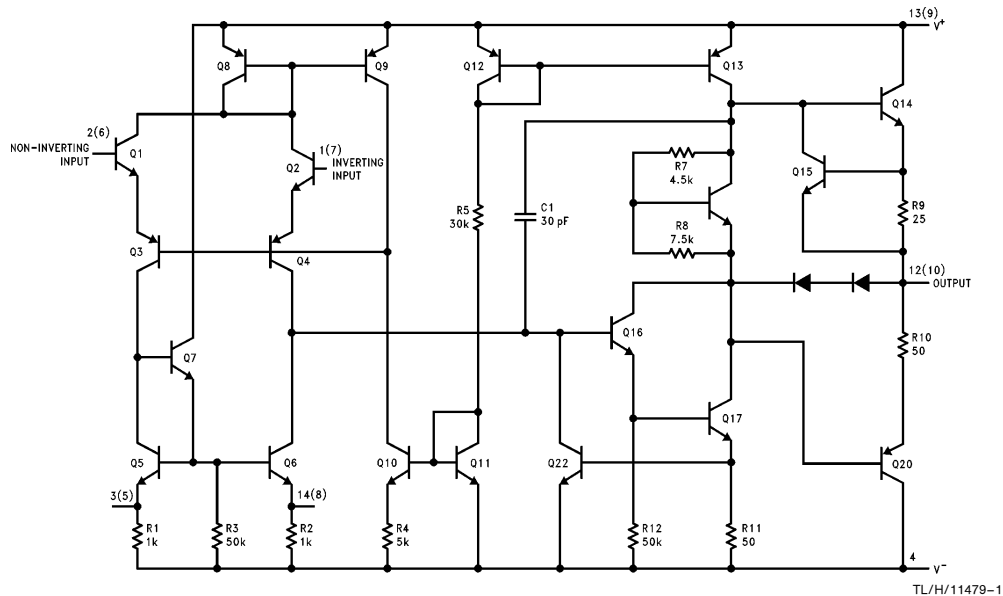
Note 1: The maximum junction temperature of the LM747C/LM747E is $100^\circ C$. For operating at elevated temperatures, devices in the TO-5 package must be derated based on a thermal resistance of $150^\circ C/W$, junction to ambient, or $45^\circ C/W$, junction to case. The thermal resistance of the dual-in-line package is $100^\circ C/W$, junction to ambient.

Note 2: For supply voltages less than $\pm 15V$, the absolute maximum input voltage is equal to the supply voltage.

Note 3: These specifications apply for $\pm 5V \leq V_S \leq \pm 20V$ and $-55^\circ C \leq T_A \leq 125^\circ C$ for the LM747A and $0^\circ C \leq T_A \leq 70^\circ C$ for the LM747E unless otherwise specified. The LM747 and LM747C are specified for $V_S = \pm 15V$ and $-55^\circ C \leq T_A \leq 125^\circ C$ and $0^\circ C \leq T_A \leq 70^\circ C$, respectively, unless otherwise specified.

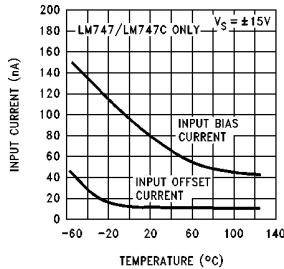
Note 4: Calculated value from: $0.35/\text{Rise Time } (\mu s)$.

Schematic Diagram (Each Amplifier)

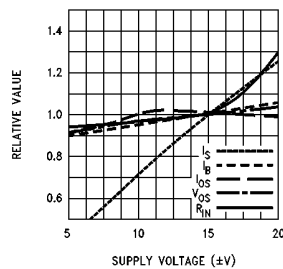


Typical Performance Characteristics

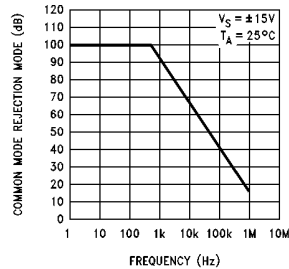
Input Bias and Offset Currents vs Ambient Temperature



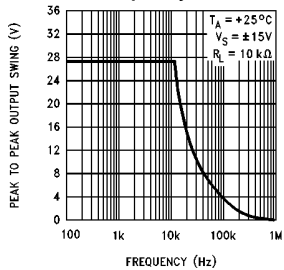
DC Parameters vs Supply Voltage



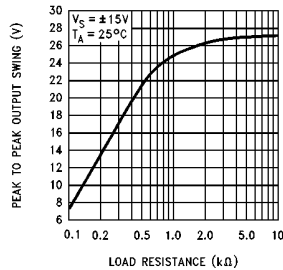
Common Mode Rejection Ratio vs Frequency



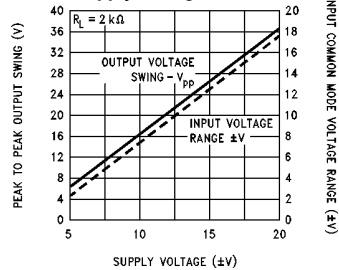
Output Voltage Swing vs Frequency



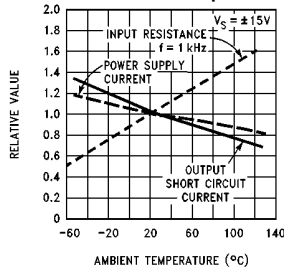
Output Voltage Swing vs Load Resistance



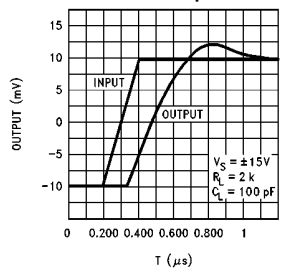
Output Swing and Input Range vs Supply Voltage



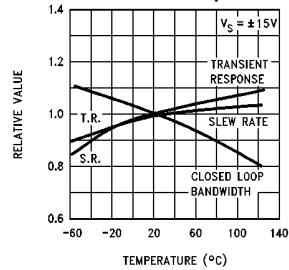
Normalized DC Parameters vs Ambient Temperature



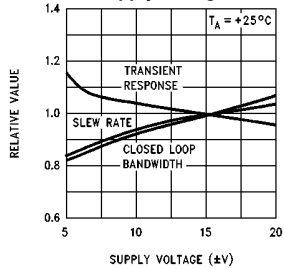
Transient Response



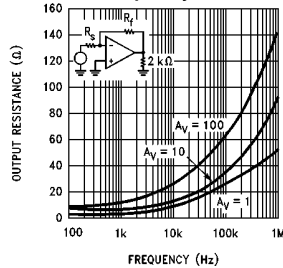
Frequency Characteristics vs Ambient Temperature



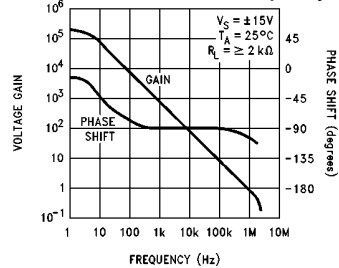
Frequency Characteristics vs Supply Voltage



Output Resistance vs Frequency

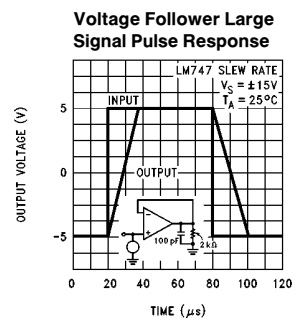
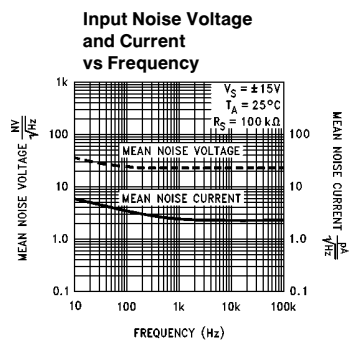
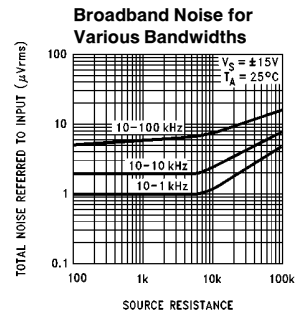
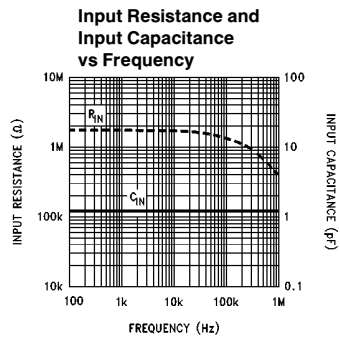


Open Loop Transfer Characteristics vs Frequency



TL/H/11479-2

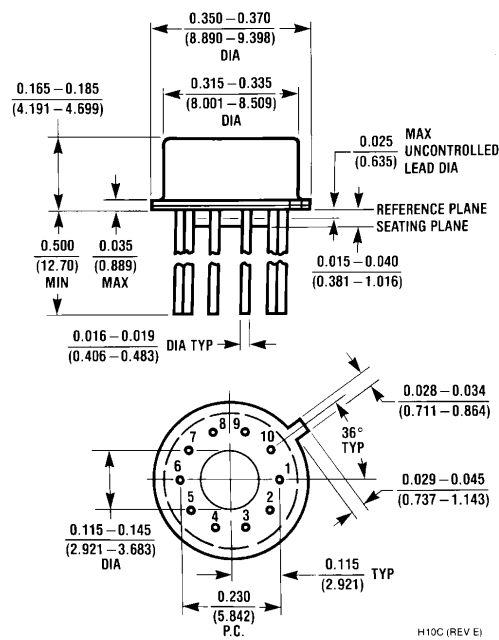
Typical Performance Characteristics (Continued)



TL/H/11479-3

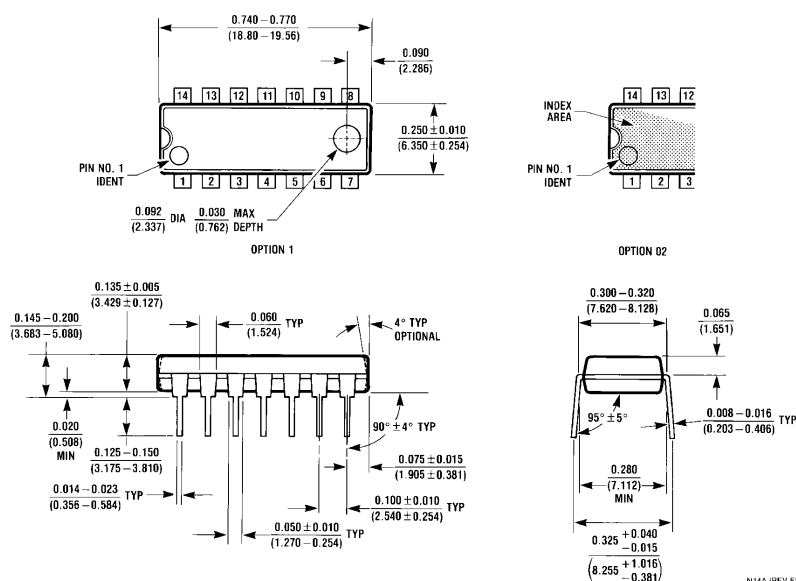


Physical Dimensions inches (millimeters)



H10C (REV E)

Metal Can Package (H)
Order Number LM747H
NS Package Number H10C

Physical Dimensions inches (millimeters) (Continued)

Dual-In-Line Package (N)
Order Number LM747CN or LM747EN
NS Package Number N14A

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.



National Semiconductor Corporation
 1111 West Bardin Road
 Arlington, TX 76017
 Tel: 1(800) 272-9959
 Fax: 1(800) 737-7018

National Semiconductor Europe
 Fax: (+49) 0-180-530 85 86
 Email: cnjwge@tevm2.nsc.com
 Deutsch Tel: (+49) 0-180-530 85 85
 English Tel: (+49) 0-180-532 78 32
 Français Tel: (+49) 0-180-532 93 58
 Italiano Tel: (+49) 0-180-534 16 80

National Semiconductor Hong Kong Ltd.
 13th Floor, Straight Block,
 Ocean Centre, 5 Canton Rd.
 Tsimshatsui, Kowloon
 Hong Kong
 Tel: (852) 2737-1600
 Fax: (852) 2736-9960

National Semiconductor Japan Ltd.
 Tel: 81-043-299-2309
 Fax: 81-043-299-2408

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products

Audio	www.ti.com/audio
Amplifiers	amplifier.ti.com
Data Converters	dataconverter.ti.com
DLP® Products	www.dlp.com
DSP	dsp.ti.com
Clocks and Timers	www.ti.com/clocks
Interface	interface.ti.com
Logic	logic.ti.com
Power Mgmt	power.ti.com
Microcontrollers	microcontroller.ti.com
RFID	www.ti-rfid.com
OMAP Mobile Processors	www.ti.com/omap
Wireless Connectivity	www.ti.com/wirelessconnectivity

Applications

Communications and Telecom	www.ti.com/communications
Computers and Peripherals	www.ti.com/computers
Consumer Electronics	www.ti.com/consumer-apps
Energy and Lighting	www.ti.com/energy
Industrial	www.ti.com/industrial
Medical	www.ti.com/medical
Security	www.ti.com/security
Space, Avionics and Defense	www.ti.com/space-avionics-defense
Transportation and Automotive	www.ti.com/automotive
Video and Imaging	www.ti.com/video

TI E2E Community Home Page

e2e.ti.com

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2011, Texas Instruments Incorporated

ADS111x Ultra-Small, Low-Power, I²C-Compatible, 860-SPS, 16-Bit ADCs With Internal Reference, Oscillator, and Programmable Comparator

1 Features

- Ultra-Small X2QFN Package:
2 mm × 1.5 mm × 0.4 mm
- Wide Supply Range: 2.0 V to 5.5 V
- Low Current Consumption: 150 μ A
(Continuous-Conversion Mode)
- Programmable Data Rate:
8 SPS to 860 SPS
- Single-Cycle Settling
- Internal Low-Drift Voltage Reference
- Internal Oscillator
- I²C Interface: Four Pin-Selectable Addresses
- Four Single-Ended or Two Differential Inputs
(ADS1115)
- Programmable Comparator (ADS1114 and
ADS1115)
- Operating Temperature Range:
–40°C to +125°C

2 Applications

- Portable Instrumentation
- Battery Voltage and Current Monitoring
- Temperature Measurement Systems
- Consumer Electronics
- Factory Automation and Process Control

3 Description

The ADS1113, ADS1114, and ADS1115 devices (ADS111x) are precision, low-power, 16-bit, I²C-compatible, analog-to-digital converters (ADCs) offered in an ultra-small, leadless, X2QFN-10 package, and a VSSOP-10 package. The ADS111x devices incorporate a low-drift voltage reference and an oscillator. The ADS1114 and ADS1115 also incorporate a programmable gain amplifier (PGA) and a digital comparator. These features, along with a wide operating supply range, make the ADS111x well suited for power- and space-constrained, sensor measurement applications.

The ADS111x perform conversions at data rates up to 860 samples per second (SPS). The PGA offers input ranges from ± 256 mV to ± 6.144 V, allowing precise large- and small-signal measurements. The ADS1115 features an input multiplexer (MUX) that allows two differential or four single-ended input measurements. Use the digital comparator in the ADS1114 and ADS1115 for under- and overvoltage detection.

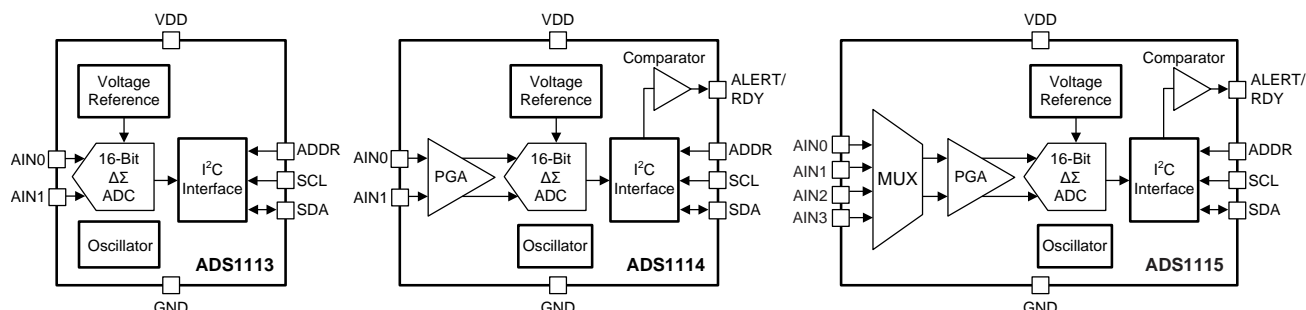
The ADS111x operate in either continuous-conversion mode or single-shot mode. The devices are automatically powered down after one conversion in single-shot mode; therefore, power consumption is significantly reduced during idle periods.

Device Information⁽¹⁾

PART NUMBER	PACKAGE	BODY SIZE (NOM)
ADS111x	X2QFN (10)	1.50 mm × 2.00 mm
	VSSOP (10)	3.00 mm × 3.00 mm

(1) For all available packages, see the package option addendum at the end of the data sheet.

Simplified Block Diagrams



Copyright © 2016, Texas Instruments Incorporated



Table of Contents

1 Features	1	9.5 Programming.....	22
2 Applications	1	9.6 Register Map.....	27
3 Description	1	10 Application and Implementation	31
4 Revision History	2	10.1 Application Information.....	31
5 Device Comparison Table	5	10.2 Typical Application	36
6 Pin Configuration and Functions	5	11 Power Supply Recommendations	40
7 Specifications	6	11.1 Power-Supply Sequencing.....	40
7.1 Absolute Maximum Ratings	6	11.2 Power-Supply Decoupling.....	40
7.2 ESD Ratings.....	6	12 Layout	41
7.3 Recommended Operating Conditions	6	12.1 Layout Guidelines	41
7.4 Thermal Information.....	6	12.2 Layout Example	42
7.5 Electrical Characteristics.....	7	13 Device and Documentation Support	43
7.6 Timing Requirements: I ² C.....	8	13.1 Documentation Support	43
7.7 Typical Characteristics	9	13.2 Related Links	43
8 Parameter Measurement Information	13	13.3 Receiving Notification of Documentation Updates.....	43
8.1 Noise Performance	13	13.4 Community Resources.....	43
9 Detailed Description	14	13.5 Trademarks	43
9.1 Overview	14	13.6 Electrostatic Discharge Caution.....	43
9.2 Functional Block Diagrams	14	13.7 Glossary	43
9.3 Feature Description.....	15	14 Mechanical, Packaging, and Orderable Information	44
9.4 Device Functional Modes.....	21		

4 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from Revision C (December 2016) to Revision D	Page
• Changed <i>Digital input voltage</i> max value from VDD + 0.3 V to 5.5 V in <i>Absolute Maximum Ratings</i> table	6
• Added "over temperature" to Offset drift parameter for clarity	7
• Added Long-term Offset drift parameter in <i>Electrical Characteristics</i> table	7
• Added "over temperature" to Gain drift parameter for clarity	7
• Added Long-term gain drift parameter in <i>Electrical Characteristics</i> table	7
• Changed V _{IH} parameter max value from VDD to 5.5 V in <i>Electrical Characteristics</i> table	7
• Added <i>Output Data Rate and Conversion Time</i> section for clarity.....	17
• Changed Figure 28, <i>ALERT Pin Timing Diagram</i> , for clarity.....	19
• Changed Figure 39, <i>Typical Connections of the ADS1115</i> , for clarity	31
• Changed the resistor values in Figure 43, <i>Basic Hardware Configuration</i> , from 10 Ω to 10 kΩ.....	35

Changes from Revision B (October 2009) to Revision C	Page
• Added <i>Device Information</i> , <i>ESD Ratings</i> , <i>Recommended Operating Conditions</i> , and <i>Thermal Information</i> tables, and <i>Parameter Measurement Information</i> , <i>Detailed Description</i> , <i>Application and Implementation</i> , <i>Power Supply Recommendations</i> , <i>Layout</i> , <i>Device and Documentation Support</i> , and <i>Mechanical, Packaging, and Orderable Information</i> sections.....	1
• Changed <i>Title</i> , and <i>Description</i> , <i>Features</i> , and <i>Applications</i> sections for clarity	1
• Deleted temperature range text from <i>Description</i> section and moved to <i>Features</i> section	1
• Changed <i>Product Family</i> table title to <i>Device Comparison Table</i> and deleted <i>Package Designator</i> column.....	5
• Changed <i>Pin Functions</i> table for clarity.....	5
• Changed <i>Power-supply voltage</i> max value from 5.5 V to 7 V in <i>Absolute Maximum Ratings</i> table	6
• Changed <i>Analog input voltage</i> min value from –0.3 V to GND – 0.3 V in <i>Absolute Maximum Ratings</i> table	6

• Changed <i>Digital input voltage</i> min value from –0.5 V to GND – 0.3 V in <i>Absolute Maximum Ratings</i> table.....	6
• Changed <i>Digital input voltage</i> max value from 5.5 V to VDD + 0.3 V in <i>Absolute Maximum Ratings</i> table	6
• Deleted <i>Analog input current</i> rows in <i>Absolute Maximum Ratings</i> table.....	6
• Added <i>Input current</i> row in <i>Absolute Maximum Ratings</i> table	6
• Added <i>Operating temperature</i> range of –40°C to +125°C back into <i>Absolute Maximum Ratings</i> table.....	6
• Added minimum specification of –40°C for T _J in <i>Absolute Maximum Ratings</i> table	6
• Changed <i>Electrical Characteristics</i> table conditions line for clarity	7
• Changed all instances of "FS" to "FSR"	7
• Deleted FSR from <i>Electrical Characteristics</i> and moved to <i>Recommended Operating Conditions</i> table	7
• Added values from Table 2 to <i>Differential input impedance</i> parameter in <i>Electrical Characteristics</i> table	7
• Changed <i>Output noise</i> parameter link from "see <i>Typical Characteristics</i> " to "see <i>Noise Performance</i> section" in <i>Electrical Characteristics</i> table	7
• Changed <i>Offset error</i> empty min value to –3, and max value from ±3 to 3 for clarity in <i>Electrical Characteristics</i> table	7
• Changed V _{IH} parameter max value from 5.5 V to VDD in <i>Electrical Characteristics</i> table	7
• Changed V _{IL} parameter min value from GND – 0.5 V to GND in <i>Electrical Characteristics</i> table	7
• Changed <i>Input leakage current</i> parameters from two rows to one row, changed test conditions from V _{IH} = 5.5V and V _{IL} = GND to GND < V _{DIG} < VDD, and changed min value from 10 µA to –10 µA in <i>Electrical Characteristics</i> table.....	7
• Changed text in note 1 of <i>Electrical Characteristics</i> table from "In no event should more than VDD + 0.3 V be applied to this device" to "No more than VDD + 0.3 V or 5.5 V (whichever is smaller) must be applied to this device. See Table 3 for more information."	7
• Deleted <i>Power-supply voltage</i> parameter from <i>Electrical Characteristics</i> and moved to <i>Recommended Operating Conditions</i> table	8
• Deleted <i>Specified temperature</i> parameter from <i>Electrical Characteristics</i> and moved to <i>Recommended Operating Conditions</i> table	8
• Deleted <i>Storage temperature</i> parameter from <i>Electrical Characteristics</i> and moved to <i>Absolute Maximum Ratings</i> table ..	8
• Added condition statement in <i>Timing Requirements: P_C</i> table	8
• Added note 1 to <i>Timing Requirements</i> table	8
• Changed Figure 22; deleted "Gain = 2/3, 1, 2, 4, 8, or 16"	14
• Added <i>Functional Block Diagrams</i> for ADS1114 and ADS1113	14
• Changed <i>Analog Inputs</i> section to provide LSB size information instead of PGA setting	16
• Changed <i>Full-Scale Input</i> section title to <i>Full-Scale Range (FSR) and LSB Size</i> , and updated section for clarity	17
• Added <i>Voltage Reference</i> and <i>Oscillator</i> sections	17
• Changed <i>Comparator</i> section title to <i>Digital Comparator</i> , and updated section for clarity.	17
• Changed <i>Conversion Ready Pin</i> section for clarity	19
• Changed <i>Register Map</i> section for clarity	27
• Changed <i>Application Information</i> section for clarity	31
• Added <i>Input Protection</i> section.....	32
• Added <i>Unused Inputs and Outputs</i> section.....	32
• Changed <i>Aliasing</i> section title to <i>Analog Input Filtering</i> and updated section for clarity	33
• Added <i>Typical Application</i> section	36

Changes from Revision A (August 2009) to Revision B
Page

• Deleted <i>Operating Temperature</i> bullet from <i>Features</i> section	1
• Deleted <i>Operating temperature range</i> from <i>Absolute Maximum Ratings</i> table.....	6
• Deleted <i>Operating temperature</i> parameter from <i>Temperature</i> section of <i>Electrical Characteristics</i> table.....	8
• Changed Figure 2, <i>Operating Current vs Temperature</i> , to reflect maximum operating temperature	9
• Changed Figure 3, <i>Power-Down Current vs Temperature</i> , to reflect maximum operating temperature.....	9

ADS1113, ADS1114, ADS1115

SBAS444D – MAY 2009 – REVISED JANUARY 2018

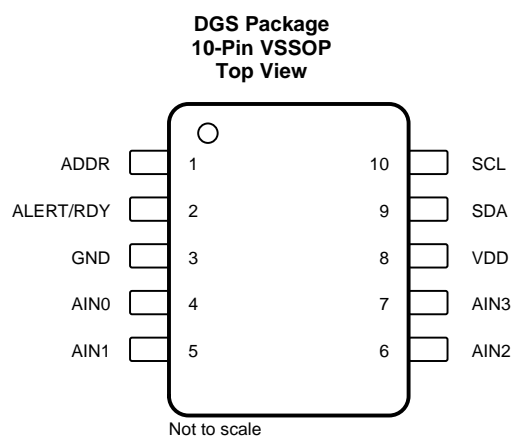
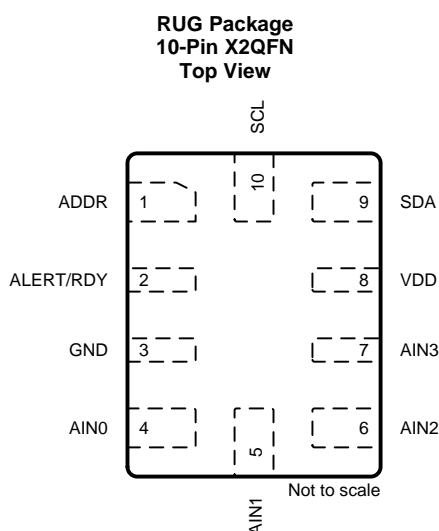
www.ti.com

• Changed Figure 4, Single-Ended Offset Error vs Temperature, to reflect maximum operating temperature	9
• Changed Figure 5, Differential Offset vs Temperature, to reflect maximum operating temperature	9
• Changed Figure 6, Gain Error vs Temperature, to reflect maximum operating temperature.....	9
• Changed 140°C to 125°C in Figure 9, INL vs Input Signal	9
• Changed +140°C to +125°C in Figure 10, INL vs Input Signal	9
• Changed +140°C to +125°C in Figure 11, INL vs Input Signal	9
• Changed +140°C to +125°C in Figure 12, INL vs Input Signal	9
• Changed Figure 13, INL vs Temperature, to reflect maximum operating temperature.....	9
• Changed Figure 16, Noise vs Temperature, to reflect maximum operating temperature	10
• Changed Figure 20, Data Rate vs Temperature, to reflect maximum operating temperature	11

5 Device Comparison Table

DEVICE	RESOLUTION (Bits)	MAXIMUM SAMPLE RATE (SPS)	INPUT CHANNELS Differential (Single-Ended)	PGA	INTERFACE	SPECIAL FEATURES
ADS1115	16	860	2 (4)	Yes	I ² C	Comparator
ADS1114	16	860	1 (1)	Yes	I ² C	Comparator
ADS1113	16	860	1(1)	No	I ² C	None
ADS1015	12	3300	2 (4)	Yes	I ² C	Comparator
ADS1014	12	3300	1 (1)	Yes	I ² C	Comparator
ADS1013	12	3300	1 (1)	No	I ² C	None
ADS1118	16	860	2 (4)	Yes	SPI	Temperature sensor
ADS1018	12	3300	2 (4)	Yes	SPI	Temperature sensor

6 Pin Configuration and Functions



Pin Functions

NAME	PIN ⁽¹⁾			TYPE	DESCRIPTION
	ADS1113	ADS1114	ADS1115		
ADDR	1	1	1	Digital input	I ² C slave address select
AIN0	4	4	4	Analog input	Analog input 0
AIN1	5	5	5	Analog input	Analog input 1
AIN2	—	—	6	Analog input	Analog input 2 (ADS1115 only)
AIN3	—	—	7	Analog input	Analog input 3 (ADS1115 only)
ALERT/RDY	—	2	2	Digital output	Comparator output or conversion ready (ADS1114 and ADS1115 only)
GND	3	3	3	Analog	Ground
NC	2, 6, 7	6, 7	—	—	Not connected
SCL	10	10	10	Digital input	Serial clock input. locks data on SDA
SDA	9	9	9	Digital I/O	Serial data. Transmits and receives data
VDD	8	8	8	Analog	Power supply. Connect a 0.1-μF, power-supply decoupling capacitor to GND.

(1) See the [Unused Inputs and Outputs](#) section for unused pin connections.

7 Specifications

7.1 Absolute Maximum Ratings

over operating free-air temperature range (unless otherwise noted)⁽¹⁾

		MIN	MAX	UNIT
Power-supply voltage	VDD to GND	−0.3	7	V
Analog input voltage	AIN0, AIN1, AIN2, AIN3	GND − 0.3	VDD + 0.3	V
Digital input voltage	SDA, SCL, ADDR, ALERT/RDY	GND − 0.3	5.5	V
Input current, continuous	Any pin except power supply pins	−10	10	mA
Temperature	Operating ambient, T _A	−40	125	°C
	Junction, T _J	−40	150	
	Storage, T _{stg}	−60	150	

- (1) Stresses beyond those listed under *Absolute Maximum Ratings* may cause permanent damage to the device. These are stress ratings only, which do not imply functional operation of the device at these or any other conditions beyond those indicated under *Recommended Operating Conditions*. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

7.2 ESD Ratings

		VALUE	UNIT
V _(ESD)	Electrostatic discharge	Human-body model (HBM), per ANSI/ESDA/JEDEC JS-001 ⁽¹⁾	±2000
		Charged-device model (CDM), per JEDEC specification JESD22-C101 ⁽²⁾	±500

- (1) JEDEC document JEP155 states that 500-V HBM allows safe manufacturing with a standard ESD control process.
(2) JEDEC document JEP157 states that 250-V CDM allows safe manufacturing with a standard ESD control process.

7.3 Recommended Operating Conditions

		MIN	NOM	MAX	UNIT
POWER SUPPLY					
	Power supply (VDD to GND)	2		5.5	V
ANALOG INPUTS⁽¹⁾					
FSR	Full-scale input voltage range ⁽²⁾ (V _{IN} = V _(AINP) − V _(AINN))	±0.256		±6.144	V
V _(AINx)	Absolute input voltage	GND		VDD	V
DIGITAL INPUTS					
V _{DIG}	Digital input voltage	GND		5.5	V
TEMPERATURE					
T _A	Operating ambient temperature	−40		125	°C

- (1) AINP and AINN denote the selected positive and negative inputs. AINx denotes one of the four available analog inputs.
(2) This parameter expresses the full-scale range of the ADC scaling. No more than VDD + 0.3 V must be applied to the analog inputs of the device. See [Table 3](#) more information.

7.4 Thermal Information

THERMAL METRIC ⁽¹⁾		ADS111x		UNIT
		DGS (VSSOP)	RUG (X2QFN)	
		10 PINS	10 PINS	
R _{θJA}	Junction-to-ambient thermal resistance	182.7	245.2	°C/W
R _{θJC(top)}	Junction-to-case (top) thermal resistance	67.2	69.3	°C/W
R _{θJB}	Junction-to-board thermal resistance	103.8	172.0	°C/W
ψ _{JT}	Junction-to-top characterization parameter	10.2	8.2	°C/W
ψ _{JB}	Junction-to-board characterization parameter	102.1	170.8	°C/W
R _{θJC(bot)}	Junction-to-case (bottom) thermal resistance	N/A	N/A	°C/W

- (1) For more information about traditional and new thermal metrics, see the [Semiconductor and IC Package Thermal Metrics](#) application report.

7.5 Electrical Characteristics

At VDD = 3.3 V, data rate = 8 SPS, and full-scale input voltage range (FSR) = ±2.048 V (unless otherwise noted). Maximum and minimum specifications apply from T_A = –40°C to +125°C. Typical specifications are at T_A = 25°C.

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
ANALOG INPUT						
Common-mode input impedance	FSR = ±6.144 V ⁽¹⁾		10		MΩ	
	FSR = ±4.096 V ⁽¹⁾ , FSR = ±2.048 V		6			
	FSR = ±1.024 V		3			
	FSR = ±0.512 V, FSR = ±0.256 V		100			
Differential input impedance	FSR = ±6.144 V ⁽¹⁾		22		MΩ	
	FSR = ±4.096 V ⁽¹⁾		15			
	FSR = ±2.048 V		4.9			
	FSR = ±1.024 V		2.4			
	FSR = ±0.512 V, ±0.256 V		710		kΩ	
SYSTEM PERFORMANCE						
	Resolution (no missing codes)		16			Bits
DR	Data rate		8, 16, 32, 64, 128, 250, 475, 860			SPS
	Data rate variation	All data rates	–10% 10%			
	Output noise		See Noise Performance section			
INL	Integral nonlinearity	DR = 8 SPS, FSR = ±2.048 V ⁽²⁾	1			LSB
Offset error	FSR = ±2.048 V, differential inputs		–3	±1	3	LSB
	FSR = ±2.048 V, single-ended inputs		±3			
Offset drift over temperature		FSR = ±2.048 V	0.005			LSB/°C
Long-term Offset drift		FSR = ±2.048 V, T _A = 125°C, 1000 hrs	±1			LSB
Offset power-supply rejection		FSR = ±2.048 V, DC supply variation	1			LSB/V
Offset channel match		Match between any two inputs	3			LSB
Gain error ⁽³⁾		FSR = ±2.048 V, T _A = 25°C	0.01% 0.15%			
Gain drift over temperature ⁽³⁾	FSR = ±0.256 V		7		40	ppm/°C
	FSR = ±2.048 V		5			
	FSR = ±6.144 V ⁽¹⁾		5			
Long-term gain drift ⁽³⁾		FSR = ±2.048 V, T _A = 125°C, 1000 hrs	±0.05			%
Gain power-supply rejection			80			ppm/V
Gain match ⁽³⁾		Match between any two gains	0.02% 0.1%			
Gain channel match		Match between any two inputs	0.05% 0.1%			
CMRR Common-mode rejection ratio	At DC, FSR = ±0.256 V		105		dB	
	At DC, FSR = ±2.048 V		100			
	At DC, FSR = ±6.144 V ⁽¹⁾		90			
	f _{CM} = 60 Hz, DR = 8 SPS		105			
	f _{CM} = 50 Hz, DR = 8 SPS		105			
DIGITAL INPUT/OUTPUT						
V _{IH}	High-level input voltage		0.7 VDD		5.5	V
V _{IL}	Low-level input voltage		GND		0.3 VDD	V
V _{OL}	Low-level output voltage	I _{OL} = 3 mA	GND	0.15	0.4	V
Input leakage current		GND < V _{DIG} < VDD	–10		10	μA

(1) This parameter expresses the full-scale range of the ADC scaling. No more than VDD + 0.3 V must be applied to the analog inputs of the device. See [Table 3](#) more information.

(2) Best-fit INL; covers 99% of full-scale.

(3) Includes all errors from onboard PGA and voltage reference.

ADS1113, ADS1114, ADS1115

SBAS444D – MAY 2009 – REVISED JANUARY 2018

www.ti.com
Electrical Characteristics (continued)

At VDD = 3.3 V, data rate = 8 SPS, and full-scale input voltage range (FSR) = ±2.048 V (unless otherwise noted). Maximum and minimum specifications apply from TA = –40°C to +125°C. Typical specifications are at TA = 25°C.

PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
POWER-SUPPLY					
IVDD Supply current	Power-down	TA = 25°C	0.5	2	μA
				5	
	Operating	TA = 25°C	150	200	
				300	
PD Power dissipation	VDD = 5.0 V		0.9		mW
	VDD = 3.3 V		0.5		
	VDD = 2.0 V		0.3		

7.6 Timing Requirements: I²C

over operating ambient temperature range and VDD = 2.0 V to 5.5 V (unless otherwise noted)

		FAST MODE		HIGH-SPEED MODE		UNIT
		MIN	MAX	MIN	MAX	
fSCL	SCL clock frequency	0.01	0.4	0.01	3.4	MHz
tBUF	Bus free time between START and STOP condition	600		160		ns
tHDSTA	Hold time after repeated START condition. After this period, the first clock is generated.	600		160		ns
tSUSTA	Setup time for a repeated START condition	600		160		ns
tSUSTO	Setup time for STOP condition	600		160		ns
tHDDAT	Data hold time	0		0		ns
tSUDAT	Data setup time	100		10		ns
tLOW	Low period of the SCL clock pin	1300		160		ns
tHIGH	High period of the SCL clock pin	600		60		ns
tF	Rise time for both SDA and SCL signals ⁽¹⁾		300		160	ns
tR	Fall time for both SDA and SCL signals ⁽¹⁾		300		160	ns

(1) For high-speed mode maximum values, the capacitive load on the bus line must not exceed 400 pF.

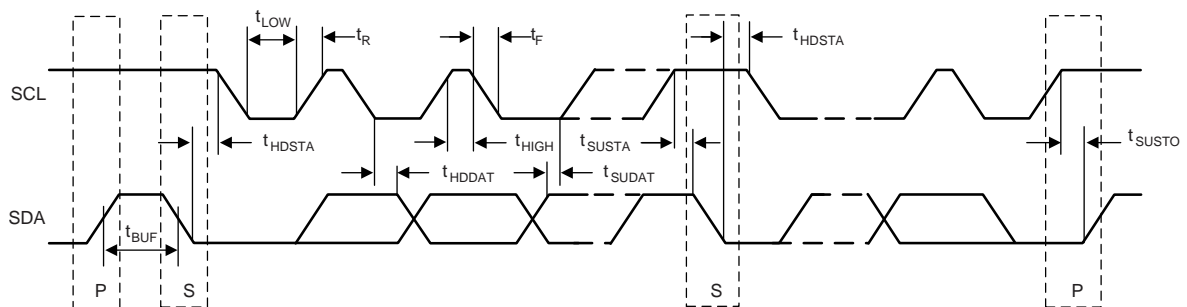


Figure 1. I²C Interface Timing

7.7 Typical Characteristics

at $T_A = 25^\circ\text{C}$, $V_{DD} = 3.3\text{ V}$, $\text{FSR} = \pm 2.048\text{ V}$, $\text{DR} = 8\text{ SPS}$ (unless otherwise noted)

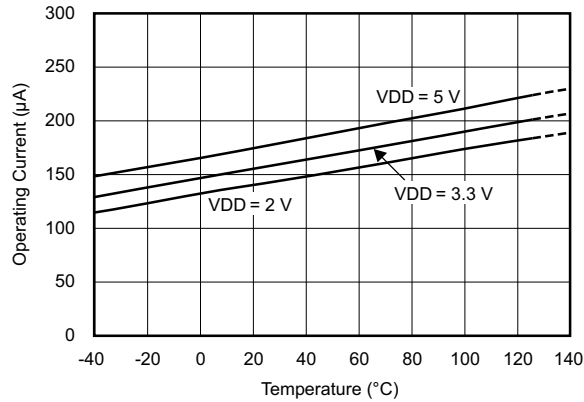


Figure 2. Operating Current vs Temperature

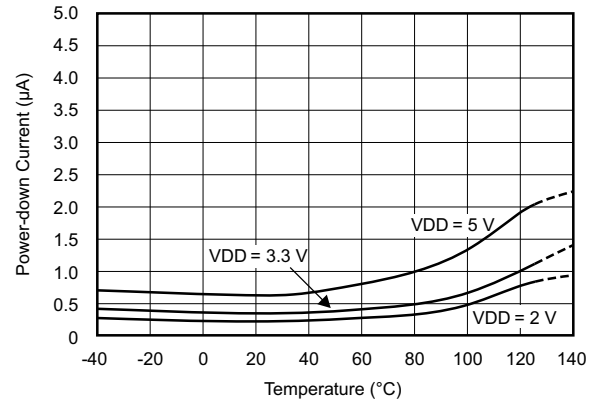


Figure 3. Power-Down Current vs Temperature

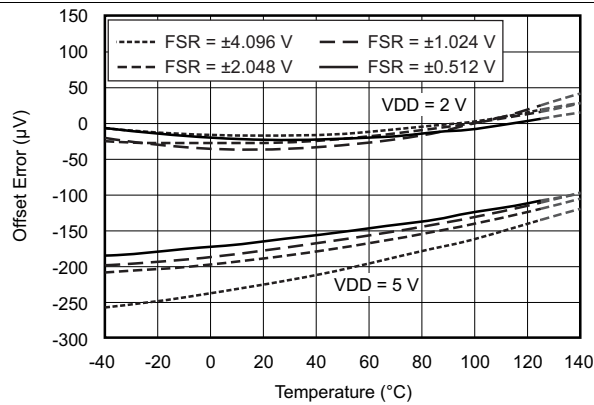


Figure 4. Single-Ended Offset Error vs Temperature

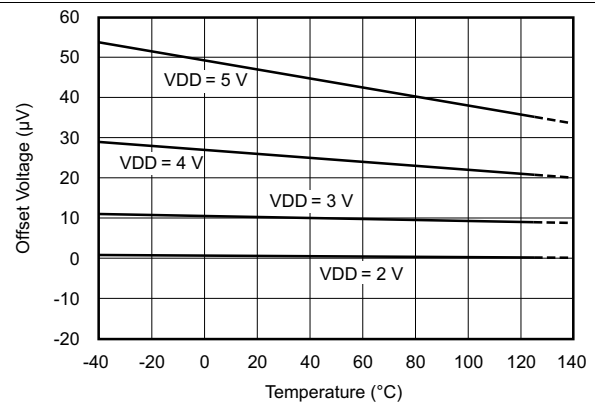


Figure 5. Differential Offset vs Temperature

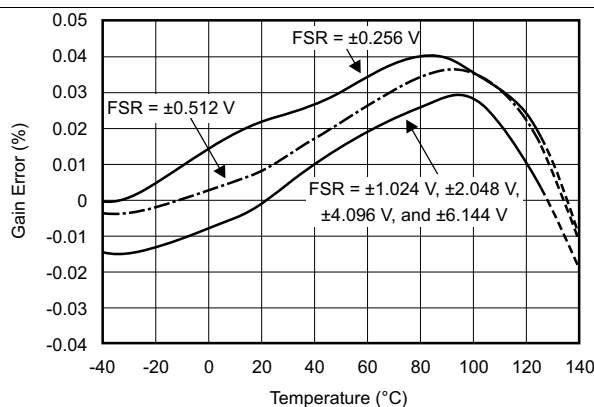


Figure 6. Gain Error vs Temperature

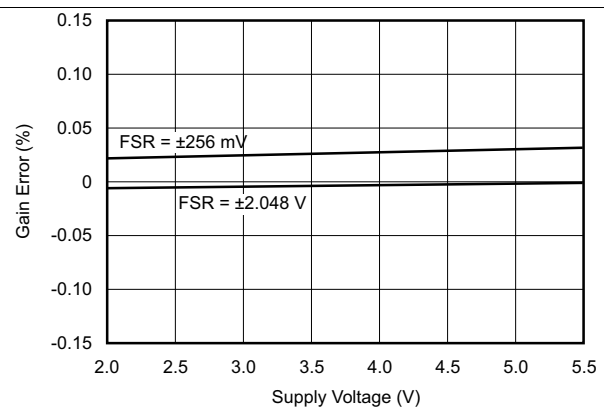


Figure 7. Gain Error vs Supply Voltage

ADS1113, ADS1114, ADS1115

SBAS444D – MAY 2009 – REVISED JANUARY 2018

www.ti.com
Typical Characteristics (continued)

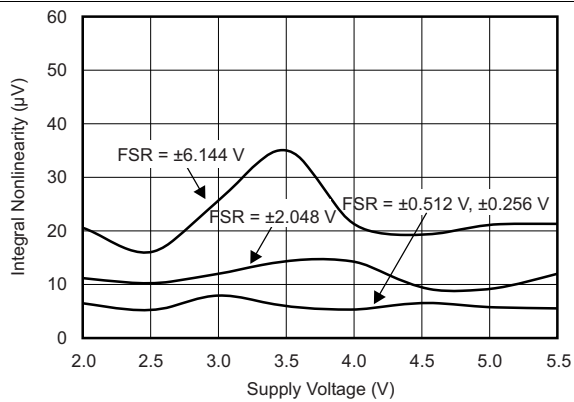
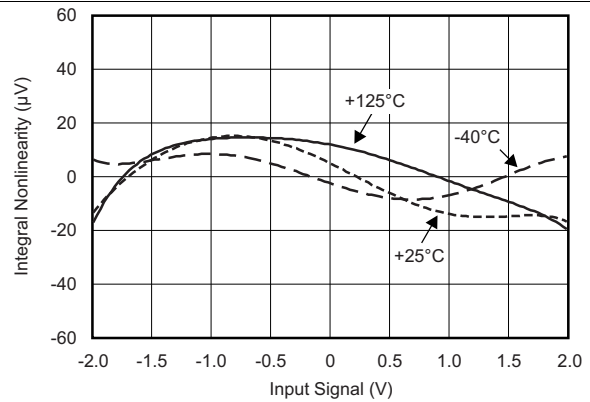
at $T_A = 25^\circ\text{C}$, $V_{DD} = 3.3\text{ V}$, $\text{FSR} = \pm 2.048\text{ V}$, $\text{DR} = 8\text{ SPS}$ (unless otherwise noted)

Figure 8. INL vs Supply Voltage

 $V_{DD} = 3.3\text{ V}$, $\text{FSR} = \pm 2.048\text{ V}$, $\text{DR} = 8\text{ SPS}$, best fit

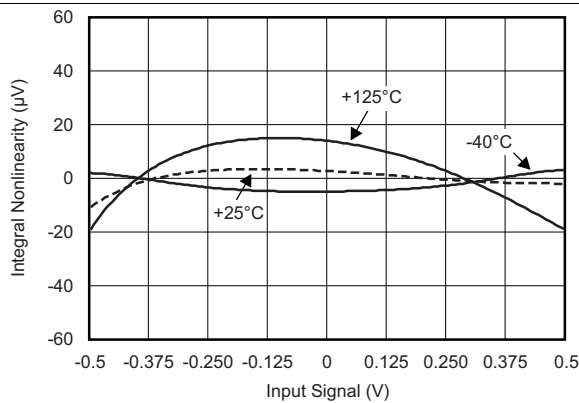
Figure 9. INL vs Input Signal

 $V_{DD} = 3.3\text{ V}$, $\text{FSR} = \pm 0.512\text{ V}$, $\text{DR} = 8\text{ SPS}$, best fit

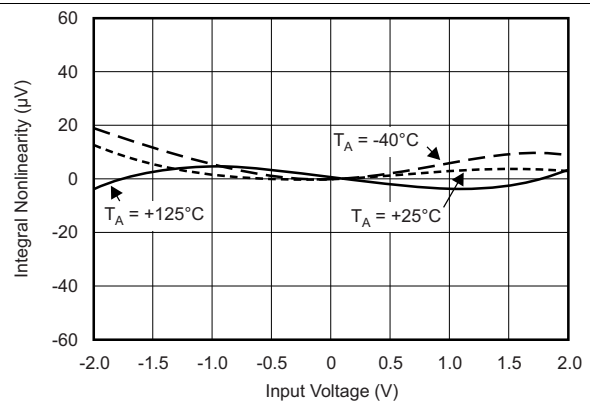
Figure 10. INL vs Input Signal

 $V_{DD} = 5\text{ V}$, $\text{FSR} = \pm 2.048\text{ V}$, $\text{DR} = 8\text{ SPS}$, best fit

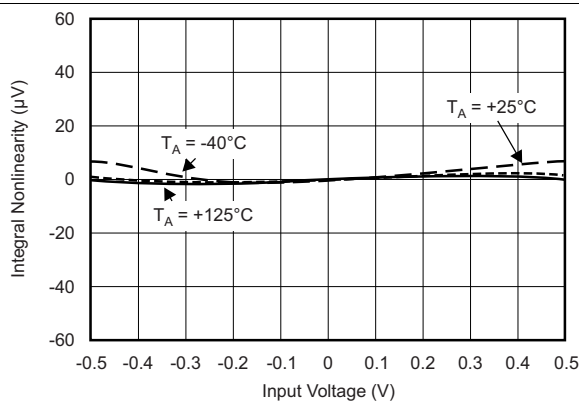
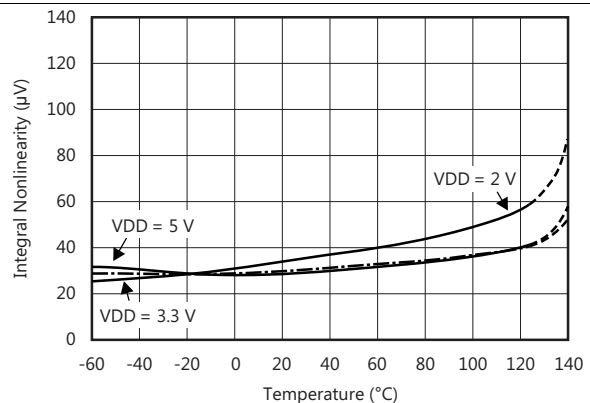
Figure 11. INL vs Input Signal

 $V_{DD} = 5\text{ V}$, $\text{FSR} = \pm 0.512\text{ V}$, $\text{DR} = 8\text{ SPS}$, best fit

Figure 12. INL vs Input Signal

Figure 13. INL vs Temperature

Typical Characteristics (continued)

at $T_A = 25^\circ\text{C}$, $V_{DD} = 3.3\text{ V}$, $\text{FSR} = \pm 2.048\text{ V}$, $\text{DR} = 8\text{ SPS}$ (unless otherwise noted)

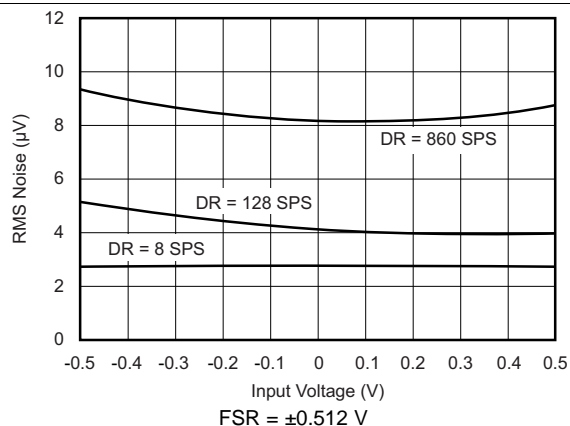


Figure 14. Noise vs Input Signal

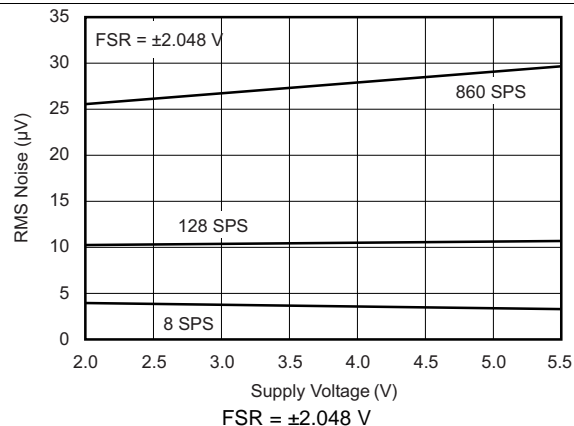


Figure 15. Noise vs Supply Voltage

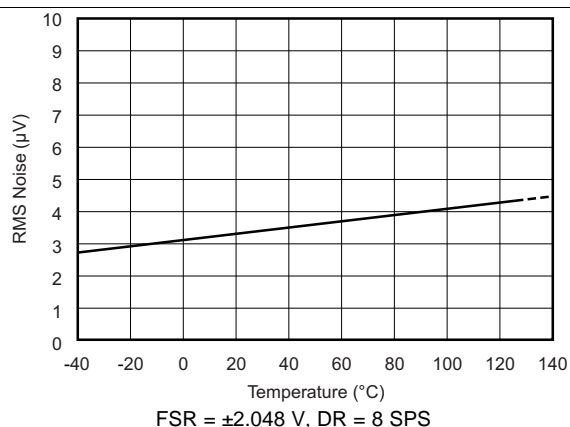
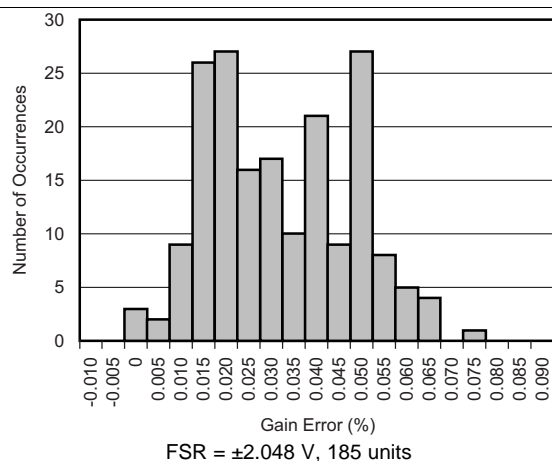
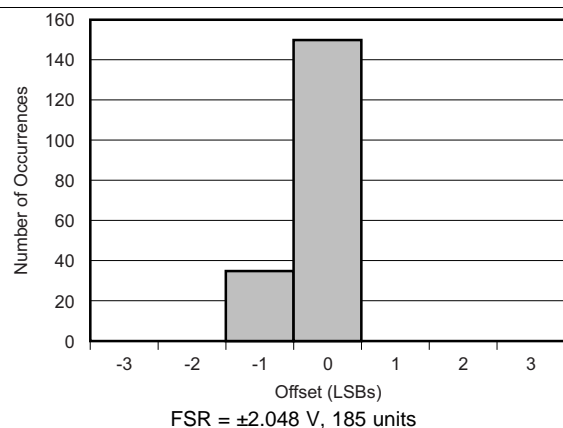


Figure 16. Noise vs Temperature



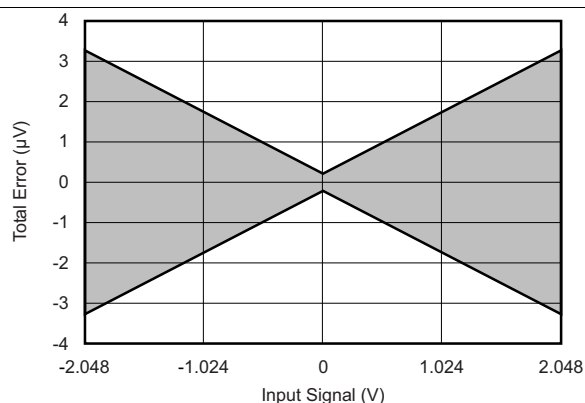
FSR = $\pm 2.048\text{ V}$, 185 units

Figure 17. Gain Error Histogram



FSR = $\pm 2.048\text{ V}$, 185 units

Figure 18. Offset Histogram



Differential inputs; includes noise, offset and gain error

Figure 19. Total Error vs Input Signal

ADS1113, ADS1114, ADS1115

SBAS444D – MAY 2009 – REVISED JANUARY 2018

www.ti.com

Typical Characteristics (continued)

at $T_A = 25^\circ\text{C}$, $V_{DD} = 3.3\text{ V}$, $\text{FSR} = \pm 2.048\text{ V}$, $\text{DR} = 8\text{ SPS}$ (unless otherwise noted)

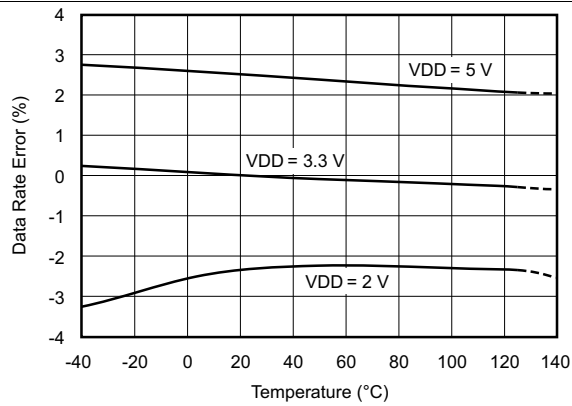


Figure 20. Data Rate vs Temperature

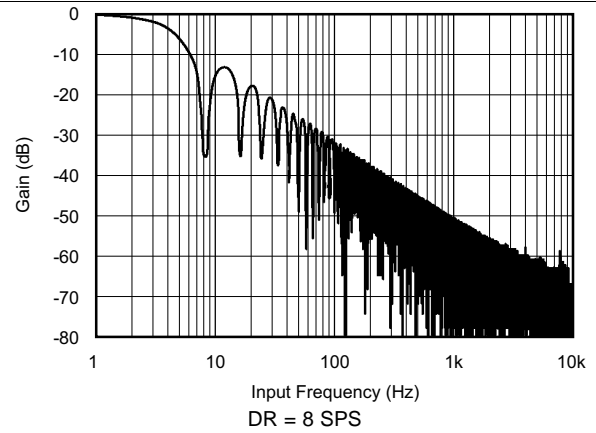


Figure 21. Digital Filter Frequency Response

8 Parameter Measurement Information

8.1 Noise Performance

Delta-sigma ($\Delta\Sigma$) analog-to-digital converters (ADCs) are based on the principle of oversampling. The input signal of a $\Delta\Sigma$ ADC is sampled at a high frequency (modulator frequency) and subsequently filtered and decimated in the digital domain to yield a conversion result at the respective output data rate. The ratio between modulator frequency and output data rate is called *oversampling ratio* (OSR). By increasing the OSR, and thus reducing the output data rate, the noise performance of the ADC can be optimized. In other words, the input-referred noise drops when reducing the output data rate because more samples of the internal modulator are averaged to yield one conversion result. Increasing the gain also reduces the input-referred noise, which is particularly useful when measuring low-level signals.

[Table 1](#) and [Table 2](#) summarize the ADS111x noise performance. Data are representative of typical noise performance at $T_A = 25^\circ\text{C}$ with the inputs shorted together externally. [Table 1](#) shows the input-referred noise in units of μV_{RMS} for the conditions shown. Note that μV_{PP} values are shown in parenthesis. [Table 2](#) shows the effective resolution calculated from μV_{RMS} values using [Equation 1](#). The noise-free resolution calculated from peak-to-peak noise values using [Equation 2](#) are shown in parenthesis.

$$\text{Effective Resolution} = \ln(\text{FSR} / V_{\text{RMS-Noise}}) / \ln(2) \quad (1)$$

$$\text{Noise-Free Resolution} = \ln(\text{FSR} / V_{\text{PP-Noise}}) / \ln(2) \quad (2)$$

Table 1. Noise in μV_{RMS} (μV_{PP}) at $\text{VDD} = 3.3 \text{ V}$

DATA RATE (SPS)	FSR (Full-Scale Range)					
	$\pm 6.144 \text{ V}$	$\pm 4.096 \text{ V}$	$\pm 2.048 \text{ V}$	$\pm 1.024 \text{ V}$	$\pm 0.512 \text{ V}$	$\pm 0.256 \text{ V}$
8	187.5 (187.5)	125 (125)	62.5 (62.5)	31.25 (31.25)	15.62 (15.62)	7.81 (7.81)
16	187.5 (187.5)	125 (125)	62.5 (62.5)	31.25 (31.25)	15.62 (15.62)	7.81 (7.81)
32	187.5 (187.5)	125 (125)	62.5 (62.5)	31.25 (31.25)	15.62 (15.62)	7.81 (7.81)
64	187.5 (187.5)	125 (125)	62.5 (62.5)	31.25 (31.25)	15.62 (15.62)	7.81 (7.81)
128	187.5 (187.5)	125 (125)	62.5 (62.5)	31.25 (31.25)	15.62 (15.62)	7.81 (12.35)
250	187.5 (252.09)	125 (148.28)	62.5 (84.03)	31.25 (39.54)	15.62 (16.06)	7.81 (18.53)
475	187.5 (266.92)	125 (227.38)	62.5 (79.08)	31.25 (56.84)	15.62 (32.13)	7.81 (25.95)
860	187.5 (430.06)	125 (266.93)	62.5 (118.63)	31.25 (64.26)	15.62 (40.78)	7.81 (35.83)

Table 2. Effective Resolution from RMS Noise (Noise-Free Resolution from Peak-to-Peak Noise) at $\text{VDD} = 3.3 \text{ V}$

DATA RATE (SPS)	FSR (Full-Scale Range)					
	$\pm 6.144 \text{ V}$	$\pm 4.096 \text{ V}$	$\pm 2.048 \text{ V}$	$\pm 1.024 \text{ V}$	$\pm 0.512 \text{ V}$	$\pm 0.256 \text{ V}$
8	16 (16)	16 (16)	16 (16)	16 (16)	16 (16)	16 (16)
16	16 (16)	16 (16)	16 (16)	16 (16)	16 (16)	16 (16)
32	16 (16)	16 (16)	16 (16)	16 (16)	16 (16)	16 (16)
64	16 (16)	16 (16)	16 (16)	16 (16)	16 (16)	16 (16)
128	16 (16)	16 (16)	16 (16)	16 (16)	16 (16)	16 (15.33)
250	16 (15.57)	16 (15.75)	16 (15.57)	16 (15.66)	16 (15.96)	16 (14.75)
475	16 (15.49)	16 (15.13)	16 (15.66)	16 (15.13)	16 (14.95)	16 (14.26)
860	16 (14.8)	16 (14.9)	16 (15.07)	16 (14.95)	16 (14.61)	16 (13.8)

9 Detailed Description

9.1 Overview

The ADS111x are very small, low-power, 16-bit, delta-sigma ($\Delta\Sigma$) analog-to-digital converters (ADCs). The ADS111x consist of a $\Delta\Sigma$ ADC core with an internal voltage reference, a clock oscillator and an I²C interface. The ADS1114 and ADS1115 also integrate a programmable gain amplifier (PGA) and a programmable digital comparator. Figure 22, Figure 23, and Figure 24 show the functional block diagrams of ADS1115, ADS1114, and ADS1113, respectively.

The ADS111x ADC core measures a differential signal, V_{IN} , that is the difference of $V_{(AINP)}$ and $V_{(AINN)}$. The converter core consists of a differential, switched-capacitor $\Delta\Sigma$ modulator followed by a digital filter. This architecture results in a very strong attenuation of any common-mode signals. Input signals are compared to the internal voltage reference. The digital filter receives a high-speed bitstream from the modulator and outputs a code proportional to the input voltage.

The ADS111x have two available conversion modes: single-shot and continuous-conversion. In single-shot mode, the ADC performs one conversion of the input signal upon request, stores the conversion value to an internal conversion register, and then enters a power-down state. This mode is intended to provide significant power savings in systems that only require periodic conversions or when there are long idle periods between conversions. In continuous-conversion mode, the ADC automatically begins a conversion of the input signal as soon as the previous conversion is completed. The rate of continuous conversion is equal to the programmed data rate. Data can be read at any time and always reflect the most recent completed conversion.

9.2 Functional Block Diagrams

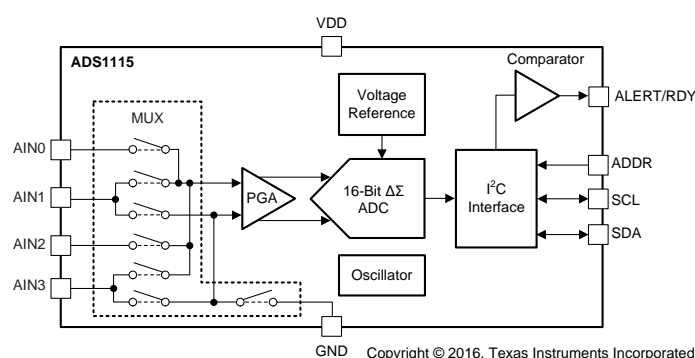


Figure 22. ADS1115 Block Diagram

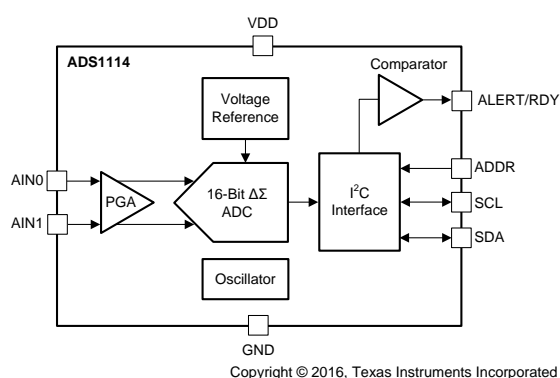


Figure 23. ADS1114 Block Diagram

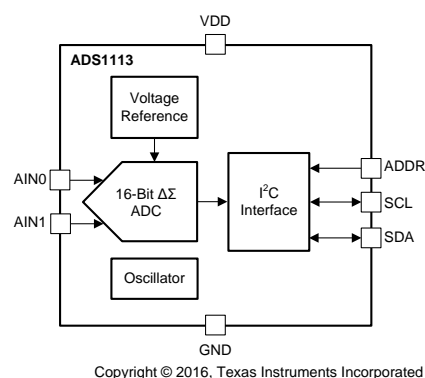
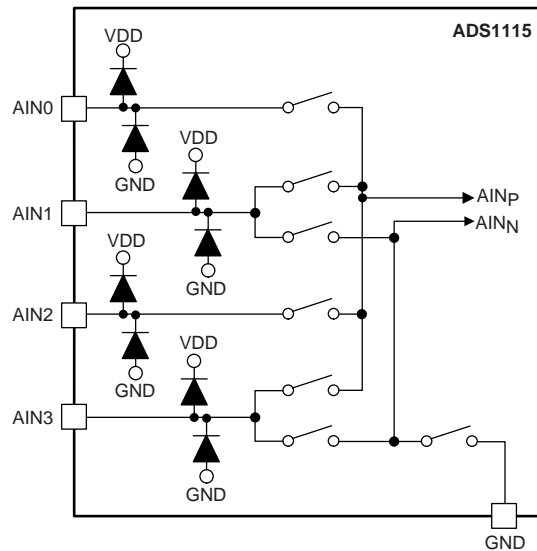


Figure 24. ADS1113 Block Diagram

9.3 Feature Description

9.3.1 Multiplexer

The ADS1115 contains an input multiplexer (MUX), as shown in Figure 25. Either four single-ended or two differential signals can be measured. Additionally, AIN0 and AIN1 may be measured differentially to AIN3. The multiplexer is configured by bits MUX[2:0] in the Config register. When single-ended signals are measured, the negative input of the ADC is internally connected to GND by a switch within the multiplexer.



Copyright © 2016, Texas Instruments Incorporated

Figure 25. Input Multiplexer

The ADS1113 and ADS1114 do not have an input multiplexer and can measure either one differential signal or one single-ended signal. For single-ended measurements, connect the AIN1 pin to GND externally. In subsequent sections of this data sheet, AIN_P refers to AIN0 and AIN_N refers to AIN1 for the ADS1113 and ADS1114.

Electrostatic discharge (ESD) diodes connected to VDD and GND protect the ADS111x analog inputs. Keep the absolute voltage of any input within the range shown in Equation 3 to prevent the ESD diodes from turning on.

$$\text{GND} - 0.3 \text{ V} < V_{(\text{AINX})} < \text{VDD} + 0.3 \text{ V} \quad (3)$$

If the voltages on the input pins can potentially violate these conditions, use external Schottky diodes and series resistors to limit the input current to safe values (see the [Absolute Maximum Ratings](#) table).

Feature Description (continued)

9.3.2 Analog Inputs

The ADS111x use a switched-capacitor input stage where capacitors are continuously charged and then discharged to measure the voltage between A_{INP} and A_{INN} . The frequency at which the input signal is sampled is called the sampling frequency or the modulator frequency (f_{MOD}). The ADS111x has a 1-MHz internal oscillator that is further divided by a factor of 4 to generate f_{MOD} at 250 kHz. The capacitors used in this input stage are small, and to external circuitry, the average loading appears resistive. Figure 26 shows this structure. The capacitor values set the resistance and switching rate. Figure 27 shows the timing for the switches in Figure 26. During the sampling phase, switches S_1 are closed. This event charges C_{A1} to $V_{(A_{INP})}$, C_{A2} to $V_{(A_{INN})}$, and C_B to $(V_{(A_{INP})} - V_{(A_{INN})})$. During the discharge phase, S_1 is first opened and then S_2 is closed. Both C_{A1} and C_{A2} then discharge to approximately 0.7 V and C_B discharges to 0 V. This charging draws a very small transient current from the source driving the ADS111x analog inputs. The average value of this current can be used to calculate the effective impedance (Z_{eff}), where $Z_{eff} = V_{IN} / I_{AVERAGE}$.

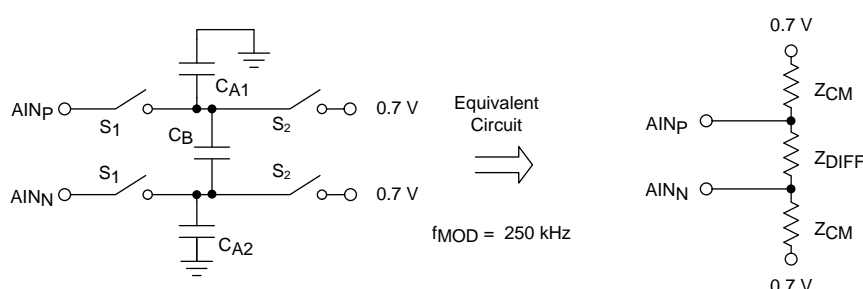


Figure 26. Simplified Analog Input Circuit

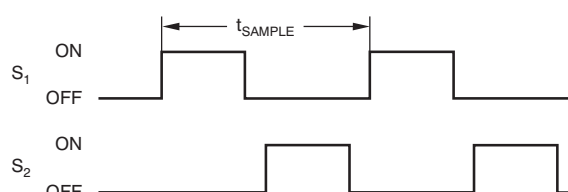


Figure 27. S_1 and S_2 Switch Timing

The common-mode input impedance is measured by applying a common-mode signal to the shorted A_{INP} and A_{INN} inputs and measuring the average current consumed by each pin. The common-mode input impedance changes depending on the full-scale range, but is approximately 6 M Ω for the default full-scale range. In Figure 26, the common-mode input impedance is Z_{CM} .

The differential input impedance is measured by applying a differential signal to A_{INP} and A_{INN} inputs where one input is held at 0.7 V. The current that flows through the pin connected to 0.7 V is the differential current and scales with the full-scale range. In Figure 26, the differential input impedance is Z_{DIFF} .

Make sure to consider the typical value of the input impedance. Unless the input source has a low impedance, the ADS111x input impedance may affect the measurement accuracy. For sources with high-output impedance, buffering may be necessary. Active buffers introduce noise, and also introduce offset and gain errors. Consider all of these factors in high-accuracy applications.

The clock oscillator frequency drifts slightly with temperature; therefore, the input impedances also drift. For most applications, this input impedance drift is negligible, and can be ignored.

Feature Description (continued)

9.3.3 Full-Scale Range (FSR) and LSB Size

A programmable gain amplifier (PGA) is implemented before the $\Delta\Sigma$ ADC of the ADS1114 and ADS1115. The full-scale range is configured by bits PGA[2:0] in the [Config register](#) and can be set to ± 6.144 V, ± 4.096 V, ± 2.048 V, ± 1.024 V, ± 0.512 V, ± 0.256 V. [Table 3](#) shows the FSR together with the corresponding LSB size. [Equation 4](#) shows how to calculate the LSB size from the selected full-scale range.

$$\text{LSB} = \text{FSR} / 2^{16} \quad (4)$$

Table 3. Full-Scale Range and Corresponding LSB Size

FSR	LSB SIZE
± 6.144 V ⁽¹⁾	187.5 μ V
± 4.096 V ⁽¹⁾	125 μ V
± 2.048 V	62.5 μ V
± 1.024 V	31.25 μ V
± 0.512 V	15.625 μ V
± 0.256 V	7.8125 μ V

(1) This parameter expresses the full-scale range of the ADC scaling. Do not apply more than VDD + 0.3 V to the analog inputs of the device.

The FSR of the ADS1113 is fixed at ± 2.048 V.

Analog input voltages must never exceed the analog input voltage limits given in the [Absolute Maximum Ratings](#). If a VDD supply voltage greater than 4 V is used, the ± 6.144 V full-scale range allows input voltages to extend up to the supply. Although in this case (or whenever the supply voltage is less than the full-scale range; for example, VDD = 3.3 V and full-scale range = ± 4.096 V), a full-scale ADC output code cannot be obtained. For example, with VDD = 3.3 V and FSR = ± 4.096 V, only signals up to $V_{IN} = \pm 3.3$ V can be measured. The code range that represents voltages $|V_{IN}| > 3.3$ V is not used in this case.

9.3.4 Voltage Reference

The ADS111x have an integrated voltage reference. An external reference cannot be used with these devices. Errors associated with the initial voltage reference accuracy and the reference drift with temperature are included in the gain error and gain drift specifications in the [Electrical Characteristics](#) table.

9.3.5 Oscillator

The ADS111x have an integrated oscillator running at 1 MHz. No external clock can be applied to operate these devices. The internal oscillator drifts over temperature and time. The output data rate scales proportionally with the oscillator frequency.

9.3.6 Output Data Rate and Conversion Time

The ADS111x offer programmable output data rates. Use the DR[2:0] bits in the [Config register](#) to select output data rates of 8 SPS, 16 SPS, 32 SPS, 64 SPS, 128 SPS, 250 SPS, 475 SPS, or 860 SPS.

Conversions in the ADS111x settle within a single cycle; thus, the conversion time is equal to 1 / DR.

9.3.7 Digital Comparator (ADS1114 and ADS1115 Only)

The ADS1115 and ADS1114 feature a programmable digital comparator that can issue an alert on the ALERT/RDY pin. The COMP_MODE bit in the [Config register](#) configures the comparator as either a traditional comparator or a window comparator. In traditional comparator mode, the ALERT/RDY pin asserts (active low by default) when conversion data exceeds the limit set in the high-threshold register (Hi_thresh). The comparator then deasserts only when the conversion data falls below the limit set in the low-threshold register (Lo_thresh). In window comparator mode, the ALERT/RDY pin asserts when the conversion data exceed the Hi_thresh register or fall below the Lo_thresh register value.

ADS1113, ADS1114, ADS1115

SBAS444D – MAY 2009 – REVISED JANUARY 2018

www.ti.com

In either window or traditional comparator mode, the comparator can be configured to latch after being asserted by the COMP_LAT bit in the Config register. This setting causes the assertion to remain even if the input signal is not beyond the bounds of the threshold registers. This latched assertion can only be cleared by issuing an SMBus alert response or by reading the [Conversion register](#). The ALERT/RDY pin can be configured as active high or active low by the COMP_POL bit in the Config register. Operational diagrams for both the comparator modes are shown in [Figure 28](#).

The comparator can also be configured to activate the ALERT/RDY pin only after a set number of successive readings exceed the threshold values set in the threshold registers (Hi_thresh and Lo_thresh). The COMP_QUE[1:0] bits in the Config register configures the comparator to wait for one, two, or four readings beyond the threshold before activating the ALERT/RDY pin. The COMP_QUE[1:0] bits can also disable the comparator function, and put the ALERT/RDY pin into a high state.

9.3.8 Conversion Ready Pin (ADS1114 and ADS1115 Only)

The ALERT/RDY pin can also be configured as a conversion ready pin. Set the most-significant bit of the Hi_thresh register to 1 and the most-significant bit of Lo_thresh register to 0 to enable the pin as a conversion ready pin. The COMP_POL bit continues to function as expected. Set the COMP_QUE[1:0] bits to any 2-bit value other than 11 to keep the ALERT/RDY pin enabled, and allow the conversion ready signal to appear at the ALERT/RDY pin output. The COMP_MODE and COMP_LAT bits no longer control any function. When configured as a conversion ready pin, ALERT/RDY continues to require a pullup resistor. The ADS111x provide an approximately 8- μ s conversion ready pulse on the ALERT/RDY pin at the end of each conversion in continuous-conversion mode, as shown in Figure 29. In single-shot mode, the ALERT/RDY pin asserts low at the end of a conversion if the COMP_POL bit is set to 0.

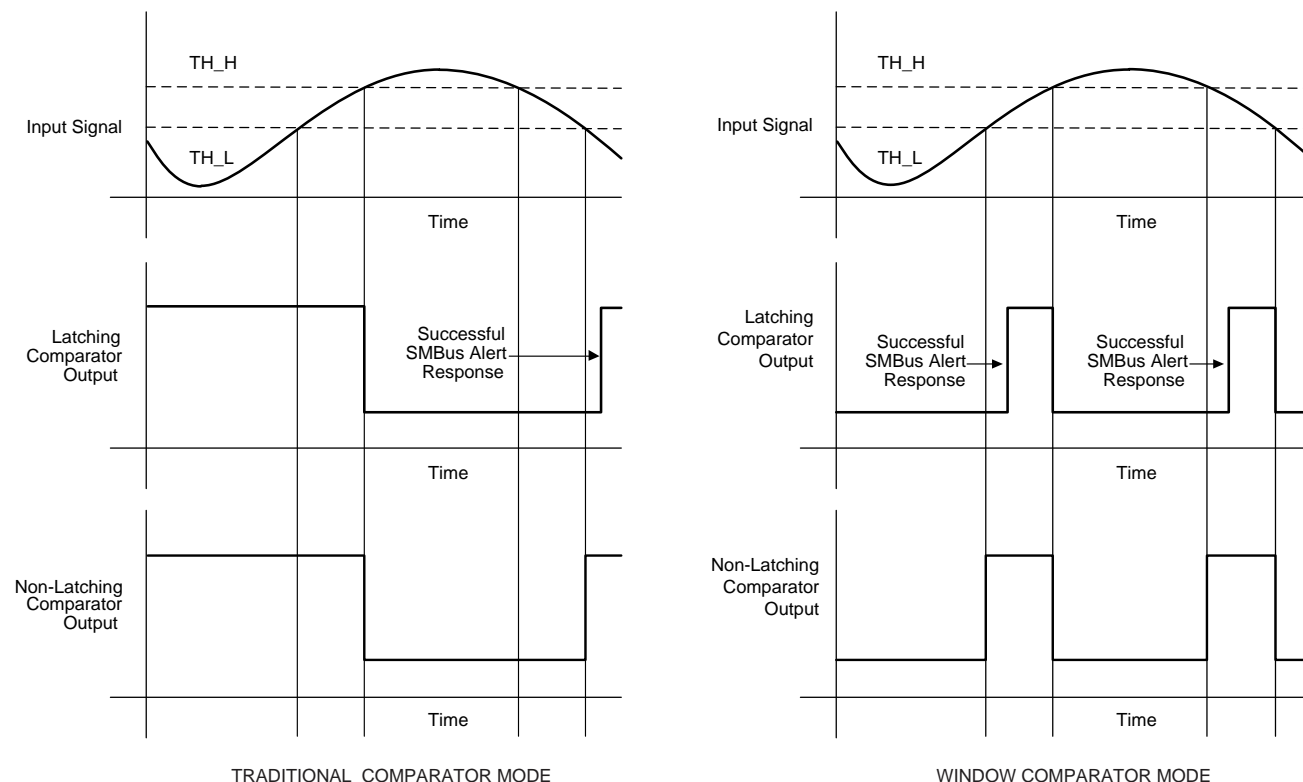


Figure 28. ALERT Pin Timing Diagram

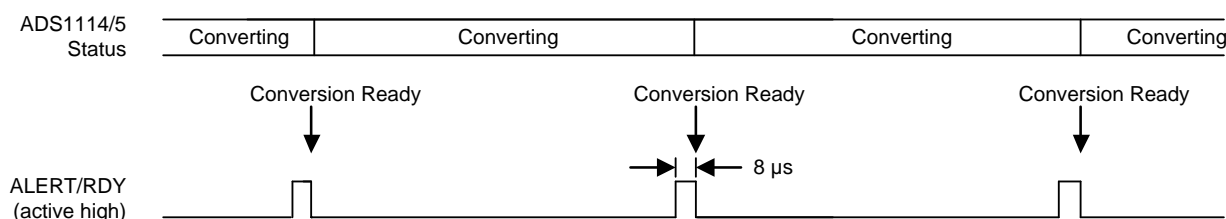


Figure 29. Conversion Ready Pulse in Continuous-Conversion Mode

ADS1113, ADS1114, ADS1115

SBAS444D – MAY 2009 – REVISED JANUARY 2018

www.ti.com**9.3.9 SMBus Alert Response**

In latching comparator mode ($\text{COMP_LAT} = 1$), the ALERT/RDY pin asserts when the comparator detects a conversion that exceeds the upper or lower threshold value. This assertion is latched and can be cleared only by reading conversion data, or by issuing a successful SMBus alert response and reading the asserting device I²C address. If conversion data exceed the upper or lower threshold values after being cleared, the pin reasserts. This assertion does not affect conversions that are already in progress. The ALERT/RDY pin is an open-drain output. This architecture allows several devices to share the same interface bus. When disabled, the pin holds a high state so that the pin does not interfere with other devices on the same bus line.

When the master senses that the ALERT/RDY pin has latched, the master issues an SMBus alert command (00011001) to the I²C bus. Any ADS1114 and ADS1115 data converters on the I²C bus with the ALERT/RDY pins asserted respond to the command with the slave address. If more than one ADS111x on the I²C bus assert the latched ALERT/RDY pin, arbitration during the address response portion of the SMBus alert determines which device clears assertion. The device with the lowest I²C address always wins arbitration. If a device loses arbitration, the device does not clear the comparator output pin assertion. The master then repeats the SMBus alert response until all devices have the respective assertions cleared. In window comparator mode, the SMBus alert status bit indicates a 1 if signals exceed the high threshold, and a 0 if signals exceed the low threshold.

9.4 Device Functional Modes

9.4.1 Reset and Power-Up

The ADS111x reset on power-up and set all the bits in the [Config register](#) to the respective default settings. The ADS111x enter a power-down state after completion of the reset process. The device interface and digital blocks are active, but no data conversions are performed. The initial power-down state of the ADS111x relieves systems with tight power-supply requirements from encountering a surge during power-up.

The ADS111x respond to the I²C general-call reset commands. When the ADS111x receive a general call reset command (06h), an internal reset is performed as if the device is powered-up.

9.4.2 Operating Modes

The ADS111x operate in one of two modes: continuous-conversion or single-shot. The MODE bit in the Config register selects the respective operating mode.

9.4.2.1 Single-Shot Mode

When the MODE bit in the Config register is set to 1, the ADS111x enter a power-down state, and operate in single-shot mode. This power-down state is the default state for the ADS111x when power is first applied. Although powered down, the devices still respond to commands. The ADS111x remain in this power-down state until a 1 is written to the operational status (OS) bit in the Config register. When the OS bit is asserted, the device powers up in approximately 25 μ s, resets the OS bit to 0, and starts a single conversion. When conversion data are ready for retrieval, the device powers down again. Writing a 1 to the OS bit while a conversion is ongoing has no effect. To switch to continuous-conversion mode, write a 0 to the MODE bit in the Config register.

9.4.2.2 Continuous-Conversion Mode

In continuous-conversion mode (MODE bit set to 0), the ADS111x perform conversions continuously. When a conversion is complete, the ADS111x place the result in the [Conversion register](#) and immediately begin another conversion. When writing new configuration settings, the currently ongoing conversion completes with the previous configuration settings. Thereafter, continuous conversions with the new configuration settings start. To switch to single-shot conversion mode, write a 1 to the MODE bit in the configuration register or reset the device.

9.4.3 Duty Cycling For Low Power

The noise performance of a $\Delta\Sigma$ ADC generally improves when lowering the output data rate because more samples of the internal modulator are averaged to yield one conversion result. In applications where power consumption is critical, the improved noise performance at low data rates may not be required. For these applications, the ADS111x support duty cycling that yield significant power savings by periodically requesting high data rate readings at an effectively lower data rate. For example, an ADS111x in power-down state with a data rate set to 860 SPS can be operated by a microcontroller that instructs a single-shot conversion every 125 ms (8 SPS). A conversion at 860 SPS only requires approximately 1.2 ms, so the ADS111x enter power-down state for the remaining 123.8 ms. In this configuration, the ADS111x consume approximately 1/100th the power that is otherwise consumed in continuous-conversion mode. The duty cycling rate is completely arbitrary and is defined by the master controller. The ADS111x offer lower data rates that do not implement duty cycling and also offer improved noise performance if required.

9.5 Programming

9.5.1 I²C Interface

The ADS111x communicate through an I²C interface. I²C is a two-wire open-drain interface that supports multiple devices and masters on a single bus. Devices on the I²C bus only drive the bus lines low by connecting them to ground; the devices never drive the bus lines high. Instead, the bus wires are pulled high by pullup resistors, so the bus wires are always high when no device is driving them low. As a result of this configuration, two devices cannot conflict. If two devices drive the bus simultaneously, there is no driver contention.

Communication on the I²C bus always takes place between two devices, one acting as the master and the other as the slave. Both the master and slave can read and write, but the slave can only do so under the direction of the master. Some I²C devices can act as a master or slave, but the ADS111x can only act as a slave device.

An I²C bus consists of two lines: SDA and SCL. SDA carries data; SCL provides the clock. All data are transmitted across the I²C bus in groups of eight bits. To send a bit on the I²C bus, drive the SDA line to the appropriate level while SCL is low (a low on SDA indicates the bit is zero; a high indicates the bit is one). After the SDA line settles, the SCL line is brought high, then low. This pulse on SCL clocks the SDA bit into the receiver shift register. If the I²C bus is held idle for more than 25 ms, the bus times out.

The I²C bus is bidirectional; that is, the SDA line is used for both transmitting and receiving data. When the master reads from a slave, the slave drives the data line; when the master sends to a slave, the master drives the data line. The master always drives the clock line. The ADS111x cannot act as a master, and therefore can never drive SCL.

Most of the time the bus is idle; no communication occurs, and both lines are high. When communication takes place, the bus is active. Only a master device can start a communication and initiate a START condition on the bus. Normally, the data line is only allowed to change state while the clock line is low. If the data line changes state while the clock line is high, it is either a START condition or a STOP condition. A START condition occurs when the clock line is high, and the data line goes from high to low. A STOP condition occurs when the clock line is high, and the data line goes from low to high.

After the master issues a START condition, the master sends a byte that indicates with which slave device to communicate. This byte is called the *address byte*. Each device on an I²C bus has a unique 7-bit address to which it responds. The master sends an address in the address byte, together with a bit that indicates whether the master wishes to read from or write to the slave device.

Every byte (address and data) transmitted on the I²C bus is acknowledged with an *acknowledge* bit. When the master finishes sending a byte (eight data bits) to a slave, the master stops driving SDA and waits for the slave to acknowledge the byte. The slave acknowledges the byte by pulling SDA low. The master then sends a clock pulse to clock the acknowledge bit. Similarly, when the master completes reading a byte, the master pulls SDA low to acknowledge this completion to the slave. The master then sends a clock pulse to clock the bit. The master always drives the clock line.

If a device is not present on the bus, and the master attempts to address it, it receives a *not-acknowledge* because no device is present at that address to pull the line low. A not-acknowledge is performed by simply leaving SDA high during an acknowledge cycle.

When the master has finished communicating with a slave, it may issue a STOP condition. When a STOP condition is issued, the bus becomes idle again. The master may also issue another START condition. When a START condition is issued while the bus is active, it is called a repeated start condition.

The [Timing Requirements](#) section shows a timing diagram for the ADS111x I²C communication.

Programming (continued)

9.5.1.1 I²C Address Selection

The ADS111x have one address pin, ADDR, that configures the I²C address of the device. This pin can be connected to GND, VDD, SDA, or SCL, allowing for four different addresses to be selected with one pin, as shown in [Table 4](#). The state of address pin ADDR is sampled continuously. Use the GND, VDD and SCL addresses first. If SDA is used as the device address, hold the SDA line low for at least 100 ns after the SCL line goes low to make sure the device decodes the address correctly during I²C communication.

Table 4. ADDR Pin Connection and Corresponding Slave Address

ADDR PIN CONNECTION	SLAVE ADDRESS
GND	1001000
VDD	1001001
SDA	1001010
SCL	1001011

9.5.1.2 I²C General Call

The ADS111x respond to the I²C general call address (0000000) if the eighth bit is 0. The devices acknowledge the general call address and respond to commands in the second byte. If the second byte is 00000110 (06h), the ADS111x reset the internal registers and enter a power-down state.

9.5.1.3 I²C Speed Modes

The I²C bus operates at one of three speeds. Standard mode allows a clock frequency of up to 100 kHz; fast mode permits a clock frequency of up to 400 kHz; and high-speed mode (also called Hs mode) allows a clock frequency of up to 3.4 MHz. The ADS111x are fully compatible with all three modes.

No special action is required to use the ADS111x in standard or fast mode, but high-speed mode must be activated. To activate high-speed mode, send a special address byte of 00001xxx following the START condition, where xxx are bits unique to the Hs-capable master. This byte is called the Hs master code, and is different from normal address bytes; the eighth bit does not indicate read/write status. The ADS111x do not acknowledge this byte; the I²C specification prohibits acknowledgment of the Hs master code. Upon receiving a master code, the ADS111x switch on Hs mode filters, and communicate at up to 3.4 MHz. The ADS111x switch out of Hs mode with the next STOP condition.

For more information on high-speed mode, consult the I²C specification.

9.5.2 Slave Mode Operations

The ADS111x act as slave receivers or slave transmitters. The ADS111x cannot drive the SCL line as slave devices.

9.5.2.1 Receive Mode

In slave receive mode, the first byte transmitted from the master to the slave consists of the 7-bit device address followed by a low R/W bit. The next byte transmitted by the master is the [Address Pointer register](#). The ADS111x then acknowledge receipt of the Address Pointer register byte. The next two bytes are written to the address given by the register address pointer bits, P[1:0]. The ADS111x acknowledge each byte sent. Register bytes are sent with the most significant byte first, followed by the least significant byte.

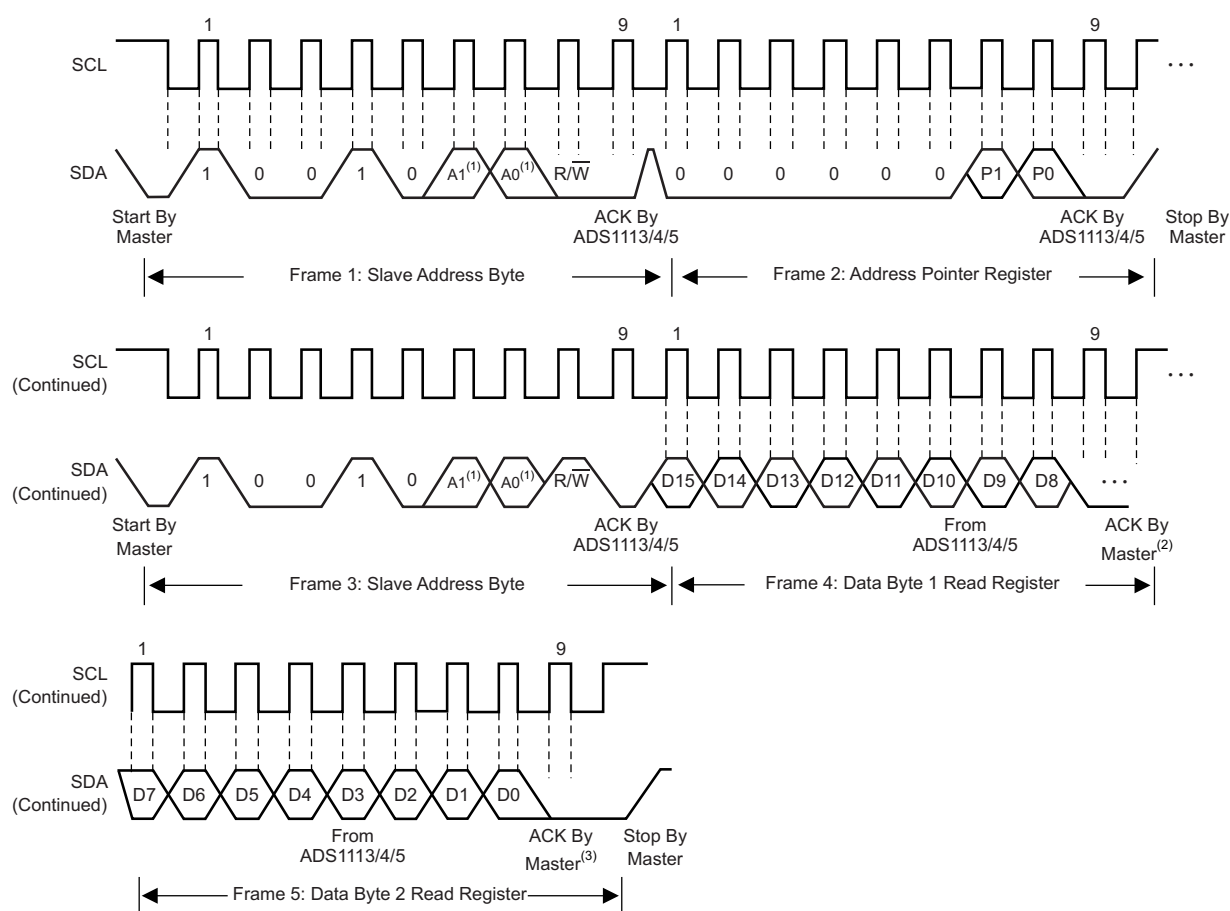
9.5.2.2 Transmit Mode

In slave transmit mode, the first byte transmitted by the master is the 7-bit slave address followed by the high R/W bit. This byte places the slave into transmit mode and indicates that the ADS111x are being read from. The next byte transmitted by the slave is the most significant byte of the register that is indicated by the register address pointer bits, P[1:0]. This byte is followed by an acknowledgment from the master. The remaining least significant byte is then sent by the slave and is followed by an acknowledgment from the master. The master may terminate transmission after any byte by not acknowledging or issuing a START or STOP condition.

9.5.3 Writing To and Reading From the Registers

To access a specific register from the ADS111x, the master must first write an appropriate value to register address pointer bits P[1:0] in the [Address Pointer register](#). The Address Pointer register is written to directly after the slave address byte, low R/W bit, and a successful slave acknowledgment. After the Address Pointer register is written, the slave acknowledges, and the master issues a STOP or a repeated START condition.

When reading from the ADS111x, the previous value written to bits P[1:0] determines the register that is read. To change which register is read, a new value must be written to P[1:0]. To write a new value to P[1:0], the master issues a slave address byte with the R/W bit low, followed by the Address Pointer register byte. No additional data has to be transmitted, and a STOP condition can be issued by the master. The master can now issue a START condition and send the slave address byte with the R/W bit high to begin the read. [Figure 37](#) details this sequence. If repeated reads from the same register are desired, there is no need to continually send the Address Pointer register, because the ADS111x store the value of P[1:0] until it is modified by a write operation. However, for every write operation, the Address Pointer register must be written with the appropriate values.



- (1) The values of A0 and A1 are determined by the ADDR pin.
- (2) Master can leave SDA high to terminate a single-byte read operation.
- (3) Master can leave SDA high to terminate a two-byte read operation.

Figure 30. Timing Diagram for Reading From ADS111x

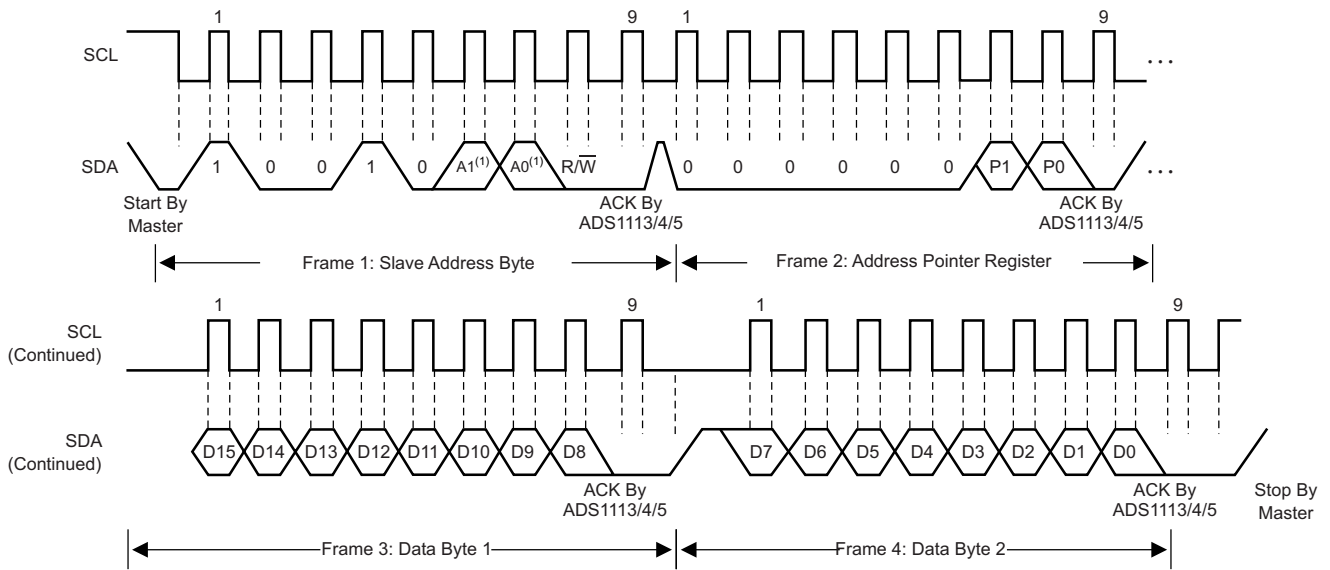


Figure 31. Timing Diagram for Writing to ADS111x

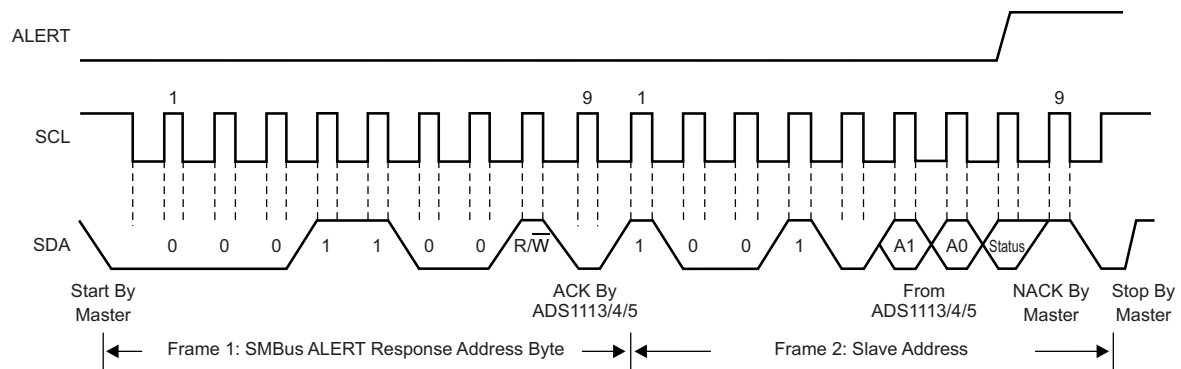


Figure 32. Timing Diagram for SMBus Alert Response

ADS1113, ADS1114, ADS1115

SBAS444D – MAY 2009 – REVISED JANUARY 2018

www.ti.com

9.5.4 Data Format

The ADS111x provide 16 bits of data in binary two's complement format. A positive full-scale (+FS) input produces an output code of 7FFFh and a negative full-scale (–FS) input produces an output code of 8000h. The output clips at these codes for signals that exceed full-scale. Table 5 summarizes the ideal output codes for different input signals. Figure 33 shows code transitions versus input voltage.

Table 5. Input Signal Versus Ideal Output Code

INPUT SIGNAL $V_{IN} = (V_{AINP} - V_{AINN})$	IDEAL OUTPUT CODE ⁽¹⁾⁽¹⁾
$\geq +FS (2^{15} - 1)/2^{15}$	7FFFh
$+FS/2^{15}$	0001h
0	0000h
$-FS/2^{15}$	FFFFh
$\leq -FS$	8000h

(1) Excludes the effects of noise, INL, offset, and gain errors.

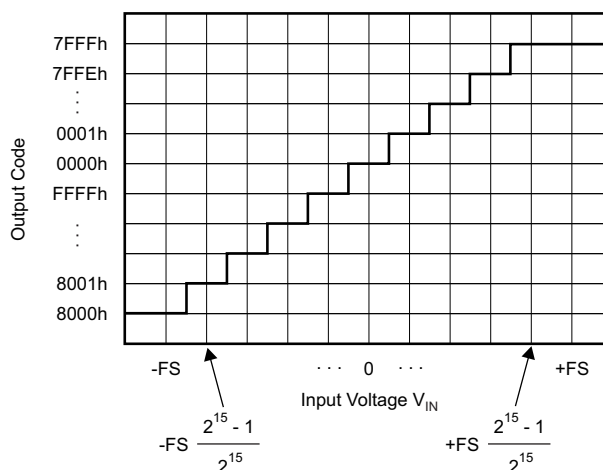


Figure 33. Code Transition Diagram

NOTE

Single-ended signal measurements, where $V_{AINN} = 0$ V and $V_{AINP} = 0$ V to +FS, only use the positive code range from 0000h to 7FFFh. However, because of device offset, the ADS111x can still output negative codes in case V_{AINP} is close to 0 V.

9.6 Register Map

The ADS111x have four registers that are accessible through the I²C interface using the [Address Pointer register](#). The [Conversion register](#) contains the result of the last conversion. The [Config register](#) is used to change the ADS111x operating modes and query the status of the device. The other two registers, Lo_thresh and Hi_thresh, set the threshold values used for the comparator function, and are not available in the ADS1113.

9.6.1 Address Pointer Register (address = N/A) [reset = N/A]

All four registers are accessed by writing to the Address Pointer register; see [Figure 30](#).

Figure 34. Address Pointer Register

7	6	5	4	3	2	1	0
0	0	0	0	0	0	P[1:0]	
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	

LEGEND: R/W = Read/Write; R = Read only; W = Write only; -n = value after reset

Table 6. Address Pointer Register Field Descriptions

Bit	Field	Type	Reset	Description
7:2	Reserved	W	0h	Always write 0h
1:0	P[1:0]	W	0h	Register address pointer 00 : Conversion register 01 : Config register 10 : Lo_thresh register 11 : Hi_thresh register

9.6.2 Conversion Register (P[1:0] = 0h) [reset = 0000h]

The 16-bit Conversion register contains the result of the last conversion in binary two's complement format. Following power-up, the Conversion register is cleared to 0, and remains 0 until the first conversion is completed.

Figure 35. Conversion Register

15	14	13	12	11	10	9	8
D15	D14	D13	D12	D11	D10	D9	D8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
D7	D6	D5	D4	D3	D2	D1	D0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

Table 7. Conversion Register Field Descriptions

Bit	Field	Type	Reset	Description
15:0	D[15:0]	R	0000h	16-bit conversion result

ADS1113, ADS1114, ADS1115

SBAS444D –MAY 2009–REVISED JANUARY 2018

www.ti.com

9.6.3 Config Register (P[1:0] = 1h) [reset = 8583h]

The 16-bit Config register is used to control the operating mode, input selection, data rate, full-scale range, and comparator modes.

Figure 36. Config Register

15	14	13	12	11	10	9	8
OS	MUX[2:0]				PGA[2:0]		MODE
R/W-1h	R/W-0h				R/W-2h		R/W-1h
7	6	5	4	3	2	1	0
DR[2:0]			COMP_MODE	COMP_POL	COMP_LAT	COMP_QUEUE[1:0]	
R/W-4h			R/W-0h	R/W-0h	R/W-0h	R/W-3h	

LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

Table 8. Config Register Field Descriptions

Bit	Field	Type	Reset	Description
15	OS	R/W	1h	Operational status or single-shot conversion start This bit determines the operational status of the device. OS can only be written when in power-down state and has no effect when a conversion is ongoing. When writing: 0 : No effect 1 : Start a single conversion (when in power-down state) When reading: 0 : Device is currently performing a conversion 1 : Device is not currently performing a conversion
14:12	MUX[2:0]	R/W	0h	Input multiplexer configuration (ADS1115 only) These bits configure the input multiplexer. These bits serve no function on the ADS1113 and ADS1114. 000 : AIN _P = AIN0 and AIN _N = AIN1 (default) 001 : AIN _P = AIN0 and AIN _N = AIN3 010 : AIN _P = AIN1 and AIN _N = AIN3 011 : AIN _P = AIN2 and AIN _N = AIN3 100 : AIN _P = AIN0 and AIN _N = GND 101 : AIN _P = AIN1 and AIN _N = GND 110 : AIN _P = AIN2 and AIN _N = GND 111 : AIN _P = AIN3 and AIN _N = GND
11:9	PGA[2:0]	R/W	2h	Programmable gain amplifier configuration These bits set the FSR of the programmable gain amplifier. These bits serve no function on the ADS1113. 000 : FSR = $\pm 6.144\text{ V}^{(1)}$ 001 : FSR = $\pm 4.096\text{ V}^{(1)}$ 010 : FSR = $\pm 2.048\text{ V}$ (default) 011 : FSR = $\pm 1.024\text{ V}$ 100 : FSR = $\pm 0.512\text{ V}$ 101 : FSR = $\pm 0.256\text{ V}$ 110 : FSR = $\pm 0.256\text{ V}$ 111 : FSR = $\pm 0.256\text{ V}$
8	MODE	R/W	1h	Device operating mode This bit controls the operating mode. 0 : Continuous-conversion mode 1 : Single-shot mode or power-down state (default)
7:5	DR[2:0]	R/W	4h	Data rate These bits control the data rate setting. 000 : 8 SPS 001 : 16 SPS 010 : 32 SPS 011 : 64 SPS 100 : 128 SPS (default) 101 : 250 SPS 110 : 475 SPS 111 : 860 SPS

(1) This parameter expresses the full-scale range of the ADC scaling. Do not apply more than VDD + 0.3 V to the analog inputs of the device.

Table 8. Config Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	COMP_MODE	R/W	0h	Comparator mode (ADS1114 and ADS1115 only) This bit configures the comparator operating mode. This bit serves no function on the ADS1113. 0 : Traditional comparator (default) 1 : Window comparator
3	COMP_POL	R/W	0h	Comparator polarity (ADS1114 and ADS1115 only) This bit controls the polarity of the ALERT/RDY pin. This bit serves no function on the ADS1113. 0 : Active low (default) 1 : Active high
2	COMP_LAT	R/W	0h	Latching comparator (ADS1114 and ADS1115 only) This bit controls whether the ALERT/RDY pin latches after being asserted or clears after conversions are within the margin of the upper and lower threshold values. This bit serves no function on the ADS1113. 0 : Nonlatching comparator . The ALERT/RDY pin does not latch when asserted (default). 1 : Latching comparator. The asserted ALERT/RDY pin remains latched until conversion data are read by the master or an appropriate SMBus alert response is sent by the master. The device responds with its address, and it is the lowest address currently asserting the ALERT/RDY bus line.
1:0	COMP_QUE[1:0]	R/W	3h	Comparator queue and disable (ADS1114 and ADS1115 only) These bits perform two functions. When set to 11, the comparator is disabled and the ALERT/RDY pin is set to a high-impedance state. When set to any other value, the ALERT/RDY pin and the comparator function are enabled, and the set value determines the number of successive conversions exceeding the upper or lower threshold required before asserting the ALERT/RDY pin. These bits serve no function on the ADS1113. 00 : Assert after one conversion 01 : Assert after two conversions 10 : Assert after four conversions 11 : Disable comparator and set ALERT/RDY pin to high-impedance (default)

ADS1113, ADS1114, ADS1115

SBAS444D – MAY 2009 – REVISED JANUARY 2018

www.ti.com

9.6.4 Lo_thresh (P[1:0] = 2h) [reset = 8000h] and Hi_thresh (P[1:0] = 3h) [reset = 7FFFh] Registers

The upper and lower threshold values used by the comparator are stored in two 16-bit registers in two's complement format. The comparator is implemented as a digital comparator; therefore, the values in these registers must be updated whenever the PGA settings are changed.

The conversion-ready function of the ALERT/RDY pin is enabled by setting the Hi_thresh register MSB to 1 and the Lo_thresh register MSB to 0. To use the comparator function of the ALERT/RDY pin, the Hi_thresh register value must always be greater than the Lo_thresh register value. The threshold register formats are shown in Figure 37. When set to RDY mode, the ALERT/RDY pin outputs the OS bit when in single-shot mode, and provides a continuous-conversion ready pulse when in continuous-conversion mode.

Figure 37. Lo_thresh Register

15	14	13	12	11	10	9	8
Lo_thresh15	Lo_thresh14	Lo_thresh13	Lo_thresh12	Lo_thresh11	Lo_thresh10	Lo_thresh9	Lo_thresh8
R/W-1h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
Lo_thresh7	Lo_thresh6	Lo_thresh5	Lo_thresh4	Lo_thresh3	Lo_thresh2	Lo_thresh1	Lo_thresh0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

Figure 38. Hi_thresh Register

15	14	13	12	11	10	9	8
Hi_thresh15	Hi_thresh14	Hi_thresh13	Hi_thresh12	Hi_thresh11	Hi_thresh10	Hi_thresh9	Hi_thresh8
R/W-0h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
Hi_thresh7	Hi_thresh6	Hi_thresh5	Hi_thresh4	Hi_thresh3	Hi_thresh2	Hi_thresh1	Hi_thresh0
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

LEGEND: R/W = Read/Write; R = Read only; -n = value after reset

Table 9. Lo_thresh and Hi_thresh Register Field Descriptions

Bit	Field	Type	Reset	Description
15:0	Lo_thresh[15:0]	R/W	8000h	Low threshold value
15:0	Hi_thresh[15:0]	R/W	7FFFh	High threshold value

10 Application and Implementation

NOTE

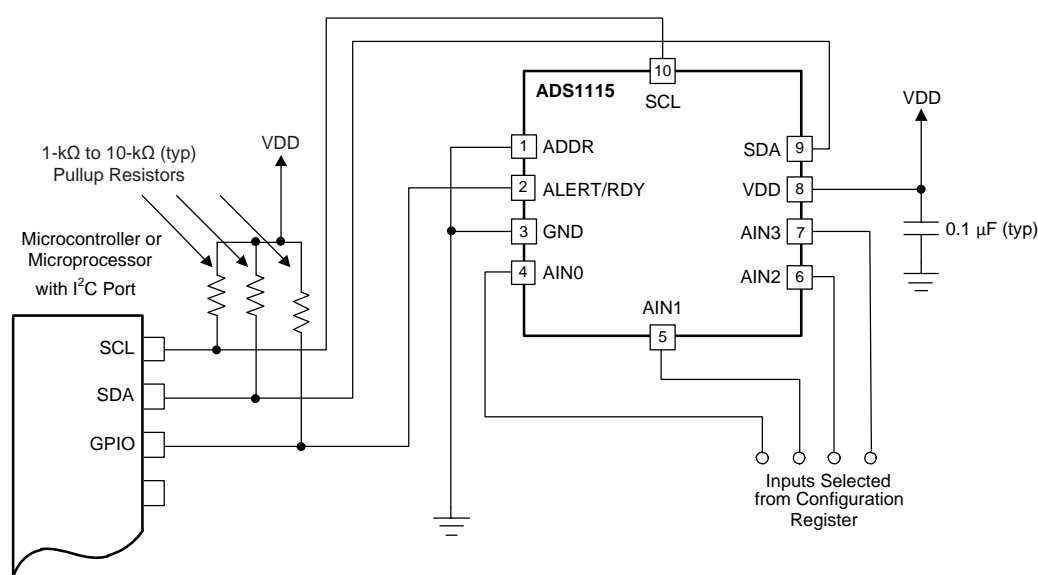
Information in the following applications sections is not part of the TI component specification, and TI does not warrant its accuracy or completeness. TI's customers are responsible for determining suitability of components for their purposes. Customers should validate and test their design implementation to confirm system functionality.

10.1 Application Information

The following sections give example circuits and suggestions for using the ADS111x in various situations.

10.1.1 Basic Connections

The principle I²C connections for the ADS1115 are shown in Figure 39.



Copyright © 2016, Texas Instruments Incorporated

Figure 39. Typical Connections of the ADS1115

The fully-differential voltage input of the ADS111x is ideal for connection to differential sources with moderately low source impedance, such as thermocouples and thermistors. Although the ADS111x can read bipolar differential signals, these devices cannot accept negative voltages on either input.

The ADS111x draw transient currents during conversion. A 0.1-µF power-supply bypass capacitor supplies the momentary bursts of extra current required from the supply.

The ADS111x interface directly to standard mode, fast mode, and high-speed mode I²C controllers. Any microcontroller I²C peripheral, including master-only and single-master I²C peripherals, operates with the ADS111x. The ADS111x does not perform clock-stretching (that is, the device never pulls the clock line low), so it is not necessary to provide for this function unless other clock-stretching devices are on the same I²C bus.

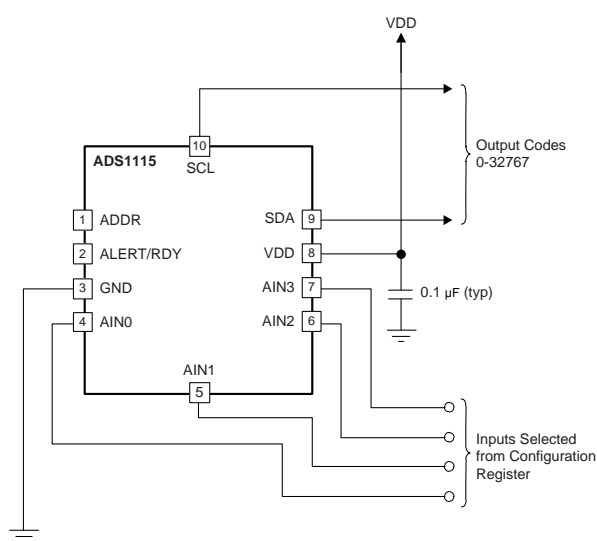
Pullup resistors are required on both the SDA and SCL lines because I²C bus drivers are open drain. The size of these resistors depends on the bus operating speed and capacitance of the bus lines. Higher-value resistors consume less power, but increase the transition times on the bus, thus limiting the bus speed. Lower-value resistors allow higher speed, but at the expense of higher power consumption. Long bus lines have higher capacitance and require smaller pullup resistors to compensate. Do not use resistors that are too small because the bus drivers may not be able to pull the bus lines low.

Application Information (continued)

10.1.2 Single-Ended Inputs

The ADS1113 and ADS1114 can measure one, and the ADS1115 up to four, single-ended signals. The ADS1113 and ADS1114 can measure single-ended signals by connecting AIN1 to GND externally. The ADS1115 measures single-ended signals by appropriate configuration of the MUX[2:0] bits in the [Config register](#). [Figure 40](#) shows a single-ended connection scheme for ADS1115. The single-ended signal ranges from 0 V up to positive supply or +FS, whichever is lower. Negative voltages cannot be applied to these devices because the ADS111x can only accept positive voltages with respect to ground. The ADS111x do not lose linearity within the input range.

The ADS111x offer a differential input voltage range of $\pm\text{FSR}$. Single-ended configurations use only one-half of the full-scale input voltage range. Differential configurations maximize the dynamic range of the ADC, and provide better common-mode noise rejection than single-ended configurations.



Copyright © 2016, Texas Instruments Incorporated

NOTE: Digital pin connections omitted for clarity.

Figure 40. Measuring Single-Ended Inputs

The ADS1115 also allows AIN3 to serve as a common point for measurements by appropriate setting of the MUX[2:0] bits. AIN0, AIN1, and AIN2 can all be measured with respect to AIN3. In this configuration, the ADS1115 operates with inputs, where AIN3 serves as the common point. This ability improves the usable range over the single-ended configuration because negative differential voltages are allowed when $\text{GND} < V_{(\text{AIN3})} < \text{VDD}$; however, common-mode noise attenuation is not offered.

10.1.3 Input Protection

The ADS111x are fabricated in a small-geometry, low-voltage process. The analog inputs feature protection diodes to the supply rails. However, the current-handling ability of these diodes is limited, and the ADS111x can be permanently damaged by analog input voltages that exceed approximately 300 mV beyond the rails for extended periods. One way to protect against overvoltage is to place current-limiting resistors on the input lines. The ADS111x analog inputs can withstand continuous currents as large as 10 mA.

10.1.4 Unused Inputs and Outputs

Either float unused analog inputs, or tie the unused analog inputs to midsupply or VDD. Connecting unused analog inputs to GND is possible, but may yield higher leakage currents than the previous options.

Either float NC (not-connected) pins, or tie the NC pins to GND. If the ALERT/RDY output pin is not used, leave the pin unconnected or tie the pin to VDD using a weak pullup resistor.

Application Information (continued)

10.1.5 Analog Input Filtering

Analog input filtering serves two purposes:

1. Limits the effect of aliasing during the sampling process
2. Reduces external noise from being a part of the measurement

Aliasing occurs when frequency components are present in the input signal that are higher than half the sampling frequency of the ADC (also known as the *Nyquist frequency*). These frequency components fold back and show up in the actual frequency band of interest below half the sampling frequency. The filter response of the digital filter repeats at multiples of the sampling frequency, also known as the modulator frequency (f_{MOD}), as shown in Figure 41. Signals or noise up to a frequency where the filter response repeats are attenuated to a certain amount by the digital filter depending on the filter architecture. Any frequency components present in the input signal around the modulator frequency, or multiples thereof, are not attenuated and alias back into the band of interest, unless attenuated by an external analog filter.

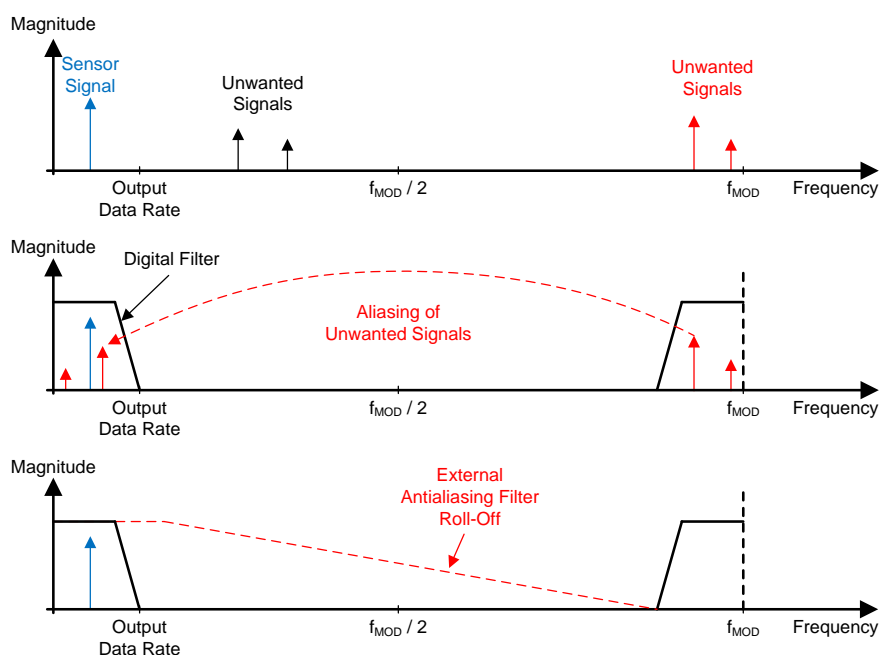


Figure 41. Effect of Aliasing

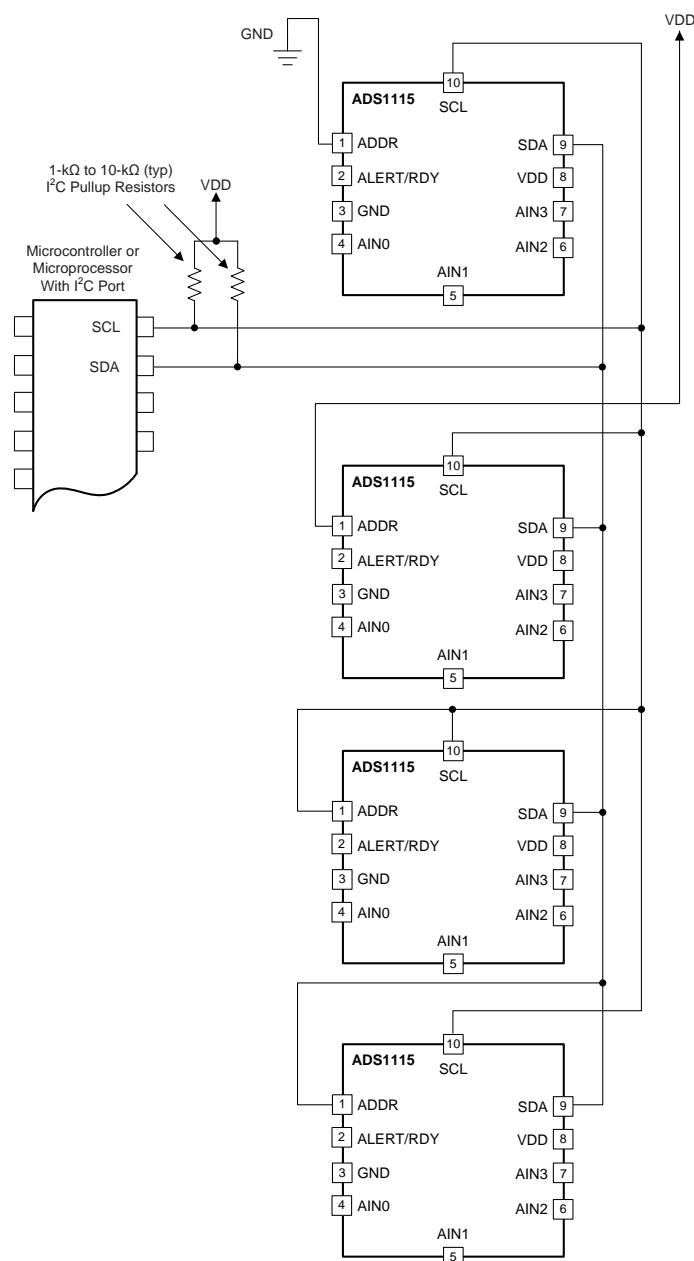
Many sensor signals are inherently band-limited; for example, the output of a thermocouple has a limited rate of change. In this case, the sensor signal does not alias back into the pass-band when using a $\Delta\Sigma$ ADC. However, any noise pick-up along the sensor wiring or the application circuitry can potentially alias into the pass-band. Power line-cycle frequency and harmonics are one common noise source. External noise can also be generated from electromagnetic interference (EMI) or radio frequency interference (RFI) sources, such as nearby motors and cellular phones. Another noise source typically exists on the printed-circuit-board (PCB) itself in the form of clocks and other digital signals. Analog input filtering helps remove unwanted signals from affecting the measurement result.

A first-order resistor-capacitor (RC) filter is (in most cases) sufficient to either totally eliminate aliasing, or to reduce the effect of aliasing to a level within the noise floor of the sensor. Ideally, any signal beyond $f_{MOD} / 2$ is attenuated to a level below the noise floor of the ADC. The digital filter of the ADS111x attenuate signals to a certain degree, as shown in Figure 21. In addition, noise components are usually smaller in magnitude than the actual sensor signal. Therefore, use a first-order RC filter with a cutoff frequency set at the output data rate or 10x higher as a generally good starting point for a system design.

Application Information (continued)

10.1.6 Connecting Multiple Devices

It is possible to connect up to four ADS111x devices to a single I²C bus using different address pin configurations for each device. Use the address pin to set the ADS111x to one of four different I²C addresses. Use the GND, VDD and SCL addresses first. If SDA is used as the device address, hold the SDA line low for at least 100 ns after the SCL line goes low to make sure the device decodes the address correctly during I²C communication. An example showing four ADS111x devices on the same I²C bus is shown in Figure 42. One set of pullup resistors is required per bus. The pullup resistor values may need to be lowered to compensate for the additional bus capacitance presented by multiple devices and increased line length.



Copyright © 2016, Texas Instruments Incorporated

NOTE: ADS111x power and input connections omitted for clarity. The ADDR pin selects the I²C address.

Figure 42. Connecting Multiple ADS111x Devices

Application Information (continued)

10.1.7 Quickstart Guide

This section provides a brief example of ADS111x communications. See subsequent sections of this data sheet for more detailed explanations. Hardware for this design includes: one ADS111x configured with an I²C address of 1001000; a microcontroller with an I²C interface; discrete components such as resistors, capacitors, and serial connectors; and a 2 V to 5 V power supply. Figure 43 shows the basic hardware configuration.

The ADS111x communicate with the master (microcontroller) through an I²C interface. The master provides a clock signal on the SCL pin and data are transferred using the SDA pin. The ADS111x never drive the SCL pin. For information on programming and debugging the microcontroller being used, see the device-specific product data sheet.

The first byte sent by the master is the ADS111x address, followed by the R/W bit that instructs the ADS111x to listen for a subsequent byte. The second byte is the Address Pointer register byte. The third and fourth bytes sent from the master are written to the register indicated in register address pointer bits P[1:0]. See Figure 30 and Figure 31 for read and write operation timing diagrams, respectively. All read and write transactions with the ADS111x must be preceded by a START condition, and followed by a STOP condition.

For example, to write to the configuration register to set the ADS111x to continuous-conversion mode and then read the conversion result, send the following bytes in this order:

1. Write to Config register:

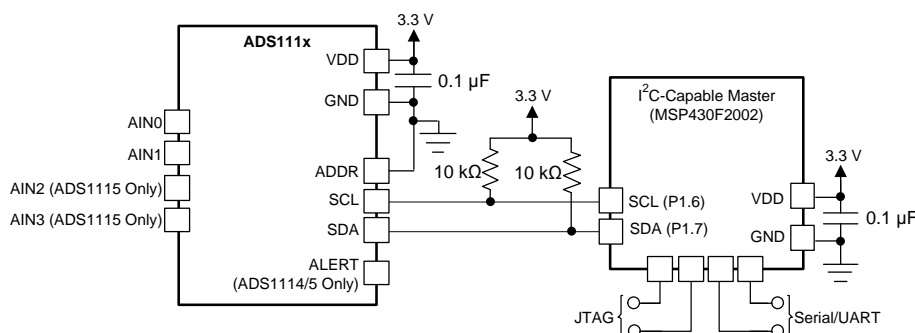
- First byte: 0b10010000 (first 7-bit I²C address followed by a low R/W bit)
- Second byte: 0b00000001 (points to Config register)
- Third byte: 0b10000100 (MSB of the Config register to be written)
- Fourth byte: 0b10000011 (LSB of the Config register to be written)

2. Write to Address Pointer register:

- First byte: 0b10010000 (first 7-bit I²C address followed by a low R/W bit)
- Second byte: 0b00000000 (points to Conversion register)

3. Read Conversion register:

- First byte: 0b10010001 (first 7-bit I²C address followed by a high R/W bit)
- Second byte: the ADS111x response with the MSB of the Conversion register
- Third byte: the ADS111x response with the LSB of the Conversion register



Copyright © 2016, Texas Instruments Incorporated

Figure 43. Basic Hardware Configuration

10.2 Typical Application

Shunt-based, current-measurement solutions are widely used to monitor load currents. Low-side, current-shunt measurements are independent of the bus voltage because the shunt common-mode voltage is near ground. [Figure 44](#) shows an example circuit for a bidirectional, low-side, current-shunt measurement system. The load current is determined by measuring the voltage across the shunt resistor that is amplified and level-shifted by a low-drift operational amplifier, [OPA333](#). The OPA333 output voltage is digitized with ADS1115 and sent to the microcontroller using the I²C interface. This circuit is capable of measuring bidirectional currents flowing through the shunt resistor with great accuracy and precision.

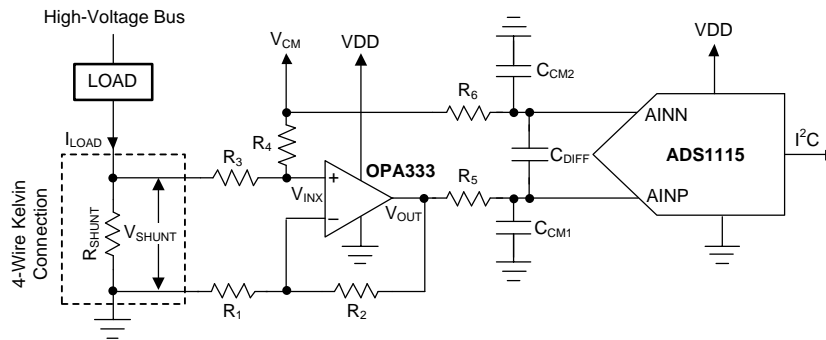


Figure 44. Low-Side Current Shunt Monitoring

10.2.1 Design Requirements

[Table 10](#) shows the design parameters for this application.

Table 10. Design Parameters

DESIGN PARAMETER	VALUE
Supply voltage (VDD)	5 V
Voltage across Shunt Resistor (V _{SHUNT})	±50 mV
Output Data Rate (DR)	≥200 readings per second
Typical measurement accuracy at T _A = 25°C ⁽¹⁾	±0.2%

(1) Does not account for inaccuracy of shunt resistor and the precision resistors used in the application.

10.2.2 Detailed Design Procedure

The first stage of the application circuit consists of an OPA333 in a noninverting summing amplifier configuration and serves two purposes:

1. To level-shift the ground-referenced signal to allow bidirectional current measurements while running off a unipolar supply. The voltage across the shunt resistor, V_{SHUNT}, is level-shifted by a common-mode voltage, V_{CM}, as shown in [Figure 44](#). The level-shifted voltage, V_{INX}, at the noninverting input is given by [Equation 5](#).
2. To amplify the level-shifted voltage (V_{INX}). The OPA333 is configured in a noninverting gain configuration with the output voltage, V_{OUT}, given by [Equation 6](#).

$$V_{INX} = (V_{CM} \cdot R_3 + V_{SHUNT} \cdot R_4) / (R_3 + R_4) \quad (5)$$

$$V_{OUT} = V_{INX} \cdot (1 + R_2 / R_1) \quad (6)$$

Using [Equation 5](#) and [Equation 6](#), V_{OUT} is given as a function of V_{SHUNT} and V_{CM} by [Equation 7](#).

$$V_{OUT} = (V_{CM} \cdot R_3 + V_{SHUNT} \cdot R_4) / (R_3 + R_4) \cdot (1 + R_2 / R_1) \quad (7)$$

Using [Equation 7](#) the ADC differential input voltage, before the first-order RC filter, is given by [Equation 8](#).

$$V_{OUT} - V_{CM} = V_{SHUNT} \cdot (1 + R_2 / R_1) / (1 + R_4 / R_3) + V_{CM} \cdot (R_2 / R_1 - R_3 / R_4) / (1 + R_3 / R_4) \quad (8)$$

If R₁ = R₃ and R₂ = R₄, [Equation 8](#) is simplified to [Equation 9](#).

$$V_{OUT} - V_{CM} = V_{SHUNT} \cdot (1 + R_2 / R_1) / (1 + R_4 / R_3) \quad (9)$$

10.2.2.1 Shunt Resistor Considerations

A shunt resistor (R_{SHUNT}) is an accurate resistance inserted in series with the load as shown in [Figure 44](#). If the absolute voltage drop across the shunt, $|V_{SHUNT}|$, is a larger percentage of the bus voltage, the voltage drop may reduce the overall efficiency and system performance. If $|V_{SHUNT}|$ is too low, measuring the small voltage drop requires careful design attention and proper selection of the ADC, operation amplifier, and precision resistors. Make sure that the absolute voltage at the shunt terminals does not result in violation of the input common-mode voltage range requirements of the operational amplifier. The power dissipation on the shunt resistor increases the temperature because of the current flowing through it. To minimize the measurement errors due to variation in temperature, select a low-drift shunt resistor. To minimize the measurement gain error, select a shunt resistor with low tolerance value. To remove the errors due to stray ground resistance, use a four-wire Kelvin-connected shunt resistor, as shown in [Figure 44](#).

10.2.2.2 Operational Amplifier Considerations

The operational amplifier used for this design example requires the following features:

- Unipolar supply operation (5 V)
- Low input offset voltage ($< 10 \mu V$) and input offset voltage drift ($< 0.5 \mu V/^{\circ}C$)
- Rail-to-rail input and output capability
- Low thermal and flicker noise
- High common-mode rejection ($> 100 \text{ dB}$)

OPA333 offers all these benefits and is selected for this application.

10.2.2.3 ADC Input Common-Mode Considerations

V_{CM} sets the V_{OUT} common-mode voltage by appropriate selection of precision resistors R_1 , R_2 , R_3 , and R_4 .

If $R_1 = R_3$, $R_2 = R_4$, and $V_{SHUNT} = 0 \text{ V}$, V_{OUT} is given by [Equation 10](#).

$$V_{OUT} = V_{CM} \quad (10)$$

If V_{OUT} is connected to the ADC positive input (AINP) and V_{CM} is connected to the ADC negative input (AINN), V_{CM} appears as a common-mode voltage to the ADC. This configuration allows pseudo-differential measurements and uses the maximum dynamic range of the ADC if V_{CM} is set at midsupply ($V_{DD} / 2$). A resistor divider from V_{DD} to GND followed by a buffer amplifier can be used to generate V_{CM} .

10.2.2.4 Resistor (R_1 , R_2 , R_3 , R_4) Considerations

Proper selection of resistors R_1 , R_2 , R_3 and R_4 is critical for meeting the overall accuracy requirements.

Using [Equation 8](#), the offset term, V_{OUT-OS} , and the gain term, A_{OUT} , of the differential ADC input are represented by [Equation 11](#) and [Equation 12](#) respectively. The error contributions from the first-order RC filters are ignored.

$$V_{OUT-OS} = V_{CM} \cdot (R_2 / R_1 - R_3 / R_4) / (1 + R_3 / R_4) \quad (11)$$

$$A_{OUT} = (1 + R_2 / R_1) / (1 + R_4 / R_3) \quad (12)$$

The tolerance, drift and linearity performance of these resistors is critical to meeting the overall accuracy requirements. In [Equation 11](#), if $R_1 = R_3$ and $R_2 = R_4$, $V_{OUT-OS} = 0 \text{ V}$ and therefore, the common-mode voltage, V_{CM} , only contributes to level-shift V_{SHUNT} and does not introduce any error at the differential ADC inputs. High-precision resistors provide better common-mode rejection from V_{CM} .

10.2.2.5 Noise and Input Impedance Considerations

If $v_{n_{res}}$ represents the input-referred rms noise from all the resistors, $v_{n_{op}}$ represents the input-referred rms noise of OPA333, and $v_{n_{ADC}}$ represents the input-referred rms noise of ADS1115, the total input-referred noise of the entire system, v_N , can be approximated by [Equation 13](#).

$$v_N^2 = v_{n_{res}}^2 + v_{n_{op}}^2 + v_{n_{ADC}}^2 / (1 + R_2 / R_1)^2 \quad (13)$$

It is important to note that the ADC noise contribution, $v_{n_{ADC}}$, is attenuated by the non-inverting gain stage.

ADS1113, ADS1114, ADS1115

SBAS444D – MAY 2009 – REVISED JANUARY 2018

www.ti.com

If the gain of the noninverting gain stage is high (≥ 5), a good approximation for $v_{n_res}^2$ is given by Equation 14. The noise contribution from resistors R_2 , R_4 , R_5 , and R_6 when referred to the input is smaller in comparison to R_1 and R_3 and can be neglected for approximation purposes.

$$v_{n_res}^2 = 4 \cdot k \cdot T \cdot (R_1 + R_3) \cdot \Delta f$$

where

- where k = Boltzmann constant
 - T = temperature (in kelvins)
 - Δf = noise bandwidth
- (14)

An approximation for the input impedance, R_{IN} , of the application circuit is given by Equation 15. R_{IN} can be modeled as a resistor in parallel with the shunt resistor, and can contribute to additional gain error.

$$R_{IN} = R_3 + R_4 \quad (15)$$

From Equation 14 and Equation 15, a trade-off exists between v_N and R_{IN} . If R_3 increases, v_{n_res} increases, and therefore, the total input-referred rms system noise, v_N , increases. If R_3 decreases, the input impedance, R_{IN} , drops, and causes additional gain error.

10.2.2.6 First-order RC Filter Considerations

Although the device digital filter attenuates high-frequency noise, use a first order low-pass RC filter at the ADC inputs to further reject out-of-bandwidth noise and avoid aliasing. A differential low-pass RC filter formed by R_5 , R_6 , and the differential capacitor C_{DIFF} sets the -3 -dB cutoff frequency, f_C , given by Equation 16. These filter resistors produce a voltage drop because of the input currents flowing into and out of the ADC. This voltage drop could contribute to an additional gain error. Limit the filter resistor values to below 1 k Ω .

$$f_C = 1 / [2\pi \cdot (R_5 + R_6) \cdot C_{DIFF}] \quad (16)$$

Two common-mode filter capacitors (C_{CM1} and C_{CM2}) are also added to offer attenuation of high-frequency, common-mode noise components. Select a differential capacitor, C_{DIFF} , that is at least an order of magnitude (10x) larger than these common-mode capacitors because mismatches in these common-mode capacitors can convert common-mode noise into differential noise.

10.2.2.7 Circuit Implementation

Table 11 shows the chosen values for this design.

Table 11. Parameters

PARAMETER	VALUE
V_{CM}	2.5 V
FSR of ADC	± 0.256 V
Output Data Rate	250 SPS
R_1 , R_3	1 k Ω ⁽¹⁾
R_2 , R_4	5 k Ω ⁽¹⁾
R_5 , R_6	100 Ω ⁽¹⁾
C_{DIFF}	0.22 μ F
C_{CM1} , C_{CM2}	0.022 μ F

(1) 1% precision resistors used

Using Equation 7, if V_{SHUNT} ranges from -50 mV to $+50$ mV, the application circuit produces a differential voltage ranging from -0.250 V to $+0.250$ V across the ADC inputs. The ADC is therefore configured at a FSR of ± 0.256 V to maximize the dynamic range of the ADC.

The -3 dB cutoff frequencies of the differential low-pass filter and the common-mode low-pass filters are set at 3.6 kHz and 0.36 kHz, respectively.

R_{SHUNT} typically ranges from 0.01 m Ω to 100 m Ω . Therefore, if $R_1 = R_3 = 1$ k Ω , a good trade-off exists between the circuit input impedance and input referred resistor noise as explained in the [Noise and Input Impedance Considerations](#) section.

A simple resistor divider followed by a buffer amplifier is used to generate V_{CM} of 2.5 V from a 5-V supply.

10.2.2.8 Results Summary

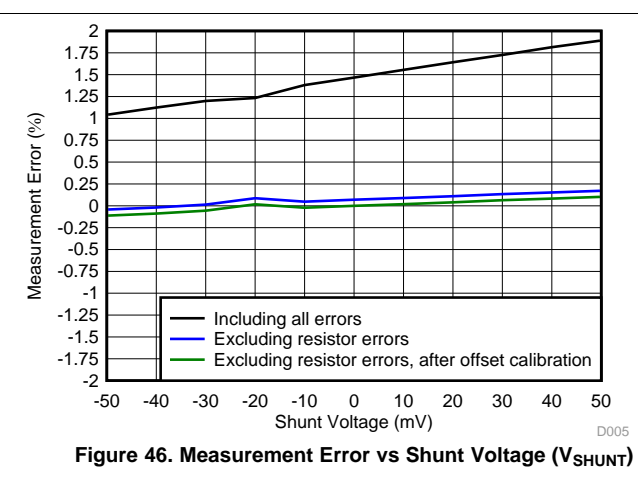
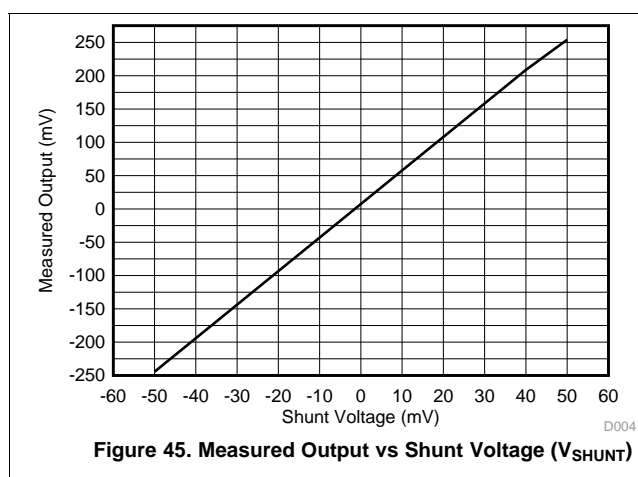
A precision voltage source is used to sweep V_{SHUNT} from -50 mV to $+50$ mV. The application circuit produces a differential voltage of -250 mV to $+250$ mV across the ADC inputs. Figure 45 and Figure 46 show the measurement results. The measurements are taken at $T_A = 25^\circ\text{C}$. Although 1% tolerance resistors are used, the exact value of these resistors are measured with a Fluke 4.5 digit multimeter to exclude the errors due to inaccuracy of these resistors. In Figure 45, the x-axis represents V_{SHUNT} and the black line represents the measured digital output voltage in mV. In Figure 46, the x-axis represents V_{SHUNT} , the black line represents the total measurement error in %, the blue line represents the total measurement error in % after excluding the errors from precision resistors and the green line represents the total measurement error in % after excluding the errors from precision resistors and performing a system offset calibration with $V_{SHUNT} = 0$ V. Table 12 shows a results summary.

Table 12. Results Summary⁽¹⁾

PARAMETER	VALUE
Total error, including errors from 1% precision resistors	1.89%
Total error, excluding errors from 1% precision resistors	0.17%
Total error, after offset calibration, excluding errors from 1% precision resistors	0.11%

(1) $T_A = 25^\circ\text{C}$, not accounting for inaccuracy of shunt resistor.

10.2.3 Application Curves



11 Power Supply Recommendations

The device requires a single unipolar supply, VDD, to power both the analog and digital circuitry of the device.

11.1 Power-Supply Sequencing

Wait approximately 50 μ s after VDD is stabilized before communicating with the device to allow the power-up reset process to complete.

11.2 Power-Supply Decoupling

Good power-supply decoupling is important to achieve optimum performance. VDD must be decoupled with at least a 0.1- μ F capacitor, as shown in Figure 47. The 0.1- μ F bypass capacitor supplies the momentary bursts of extra current required from the supply when the device is converting. Place the bypass capacitor as close to the power-supply pin of the device as possible using low-impedance connections. Use multilayer ceramic chip capacitors (MLCCs) that offer low equivalent series resistance (ESR) and inductance (ESL) characteristics for power-supply decoupling purposes. For very sensitive systems, or for systems in harsh noise environments, avoid the use of vias for connecting the capacitors to the device pins for better noise immunity. The use of multiple vias in parallel lowers the overall inductance, and is beneficial for connections to ground planes.

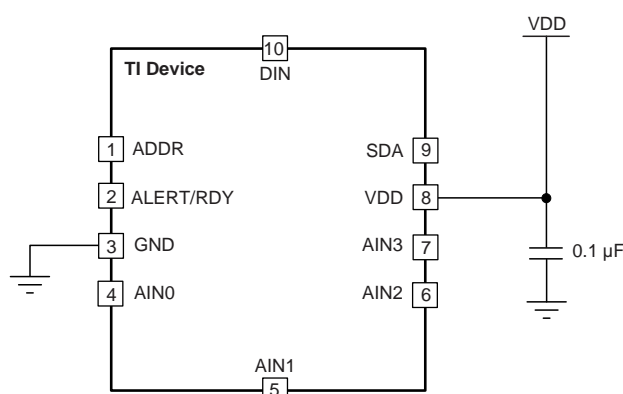


Figure 47. ADS1115 Power-Supply Decoupling

12 Layout

12.1 Layout Guidelines

Employ best design practices when laying out a printed-circuit board (PCB) for both analog and digital components. For optimal performance, separate the analog components [such as ADCs, amplifiers, references, digital-to-analog converters (DACs), and analog MUXs] from digital components [such as microcontrollers, complex programmable logic devices (CPLDs), field-programmable gate arrays (FPGAs), radio frequency (RF) transceivers, universal serial bus (USB) transceivers, and switching regulators]. An example of good component placement is shown in [Figure 48](#). Although [Figure 48](#) provides a good example of component placement, the best placement for each application is unique to the geometries, components, and PCB fabrication capabilities employed. That is, there is no single layout that is perfect for every design and careful consideration must always be used when designing with any analog component.

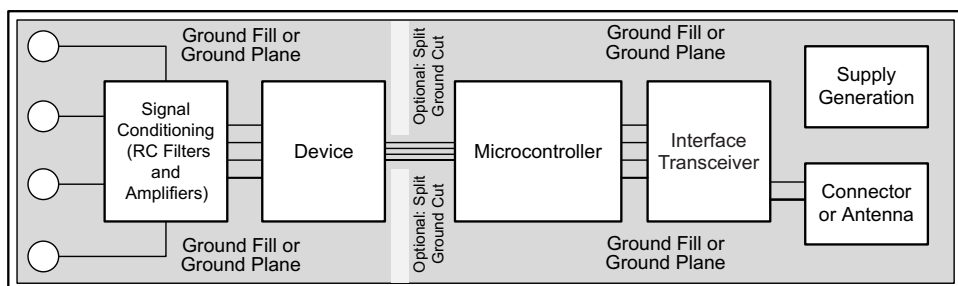


Figure 48. System Component Placement

The following outlines some basic recommendations for the layout of the ADS111x to get the best possible performance of the ADC. A good design can be ruined with a bad circuit layout.

- Separate analog and digital signals. To start, partition the board into analog and digital sections where the layout permits. Route digital lines away from analog lines. This prevents digital noise from coupling back into analog signals.
- Fill void areas on signal layers with ground fill.
- Provide good ground return paths. Signal return currents flow on the path of least impedance. If the ground plane is cut or has other traces that block the current from flowing right next to the signal trace, it has to find another path to return to the source and complete the circuit. If it is forced into a larger path, it increases the chance that the signal radiates. Sensitive signals are more susceptible to EMI interference.
- Use bypass capacitors on supplies to reduce high-frequency noise. Do not place vias between bypass capacitors and the active device. Placing the bypass capacitors on the same layer as close to the active device yields the best results.
- Consider the resistance and inductance of the routing. Often, traces for the inputs have resistances that react with the input bias current and cause an added error voltage. Reduce the loop area enclosed by the source signal and the return current in order to reduce the inductance in the path. Reduce the inductance to reduce the EMI pickup, and reduce the high frequency impedance seen by the device.
- Differential inputs must be matched for both the inputs going to the measurement source.
- Analog inputs with differential connections must have a capacitor placed differentially across the inputs. Best input combinations for differential measurements use adjacent analog input lines such as AIN0, AIN1 and AIN2, AIN3. The differential capacitors must be of high quality. The best ceramic chip capacitors are C0G (NPO), which have stable properties and low-noise characteristics.

ADS1113, ADS1114, ADS1115

SBAS444D – MAY 2009 – REVISED JANUARY 2018

www.ti.com

12.2 Layout Example

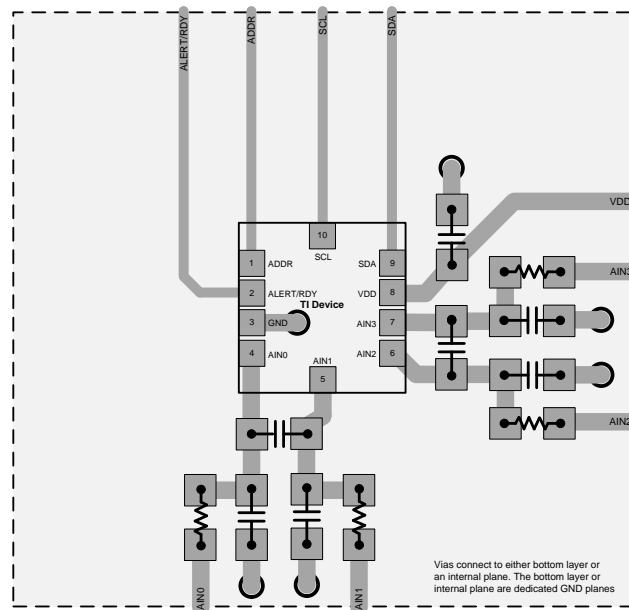


Figure 49. ADS1115 X2QFN Package

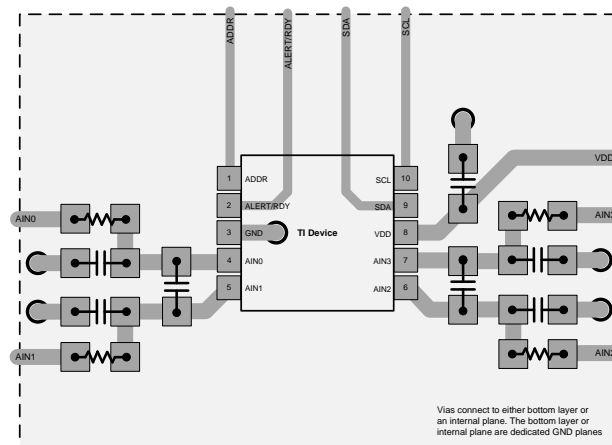


Figure 50. ADS1115 VSSOP Package

13 Device and Documentation Support

13.1 Documentation Support

13.1.1 Related Documentation

For related documentation see the following:

- [OPAx333 1.8-V, microPower, CMOS Operational Amplifiers, Zero-Drift Series](#) (SBOS351)
- [MSP430F20x1, MSP430F20x2, MSP430F20x3 Mixed Signal Microcontroller](#) (SLAS491)
- [TIDA-00824 Human Skin Temperature Sensing for Wearable Applications Reference Design](#) (TIDUAY7)

13.2 Related Links

The following table lists quick access links. Categories include technical documents, support and community resources, tools and software, and quick access to sample or buy.

Table 13. Related Links

PARTS	PRODUCT FOLDER	SAMPLE & BUY	TECHNICAL DOCUMENTS	TOOLS & SOFTWARE	SUPPORT & COMMUNITY
ADS1113	Click here	Click here	Click here	Click here	Click here
ADS1114	Click here	Click here	Click here	Click here	Click here
ADS1115	Click here	Click here	Click here	Click here	Click here

13.3 Receiving Notification of Documentation Updates

To receive notification of documentation updates, navigate to the device product folder on ti.com. In the upper right corner, click on *Alert me* to register and receive a weekly digest of any product information that has changed. For change details, review the revision history included in any revised document.

13.4 Community Resources

The following links connect to TI community resources. Linked contents are provided "AS IS" by the respective contributors. They do not constitute TI specifications and do not necessarily reflect TI's views; see TI's [Terms of Use](#).

TI E2E™ Online Community *TI's Engineer-to-Engineer (E2E) Community*. Created to foster collaboration among engineers. At e2e.ti.com, you can ask questions, share knowledge, explore ideas and help solve problems with fellow engineers.

Design Support *TI's Design Support* Quickly find helpful E2E forums along with design support tools and contact information for technical support.

13.5 Trademarks

E2E is a trademark of Texas Instruments.
All other trademarks are the property of their respective owners.

13.6 Electrostatic Discharge Caution



This integrated circuit can be damaged by ESD. Texas Instruments recommends that all integrated circuits be handled with appropriate precautions. Failure to observe proper handling and installation procedures can cause damage.

ESD damage can range from subtle performance degradation to complete device failure. Precision integrated circuits may be more susceptible to damage because very small parametric changes could cause the device not to meet its published specifications.

13.7 Glossary

[SLYZ022](#) — *TI Glossary*.

This glossary lists and explains terms, acronyms, and definitions.

14 Mechanical, Packaging, and Orderable Information

The following pages include mechanical, packaging, and orderable information. This information is the most current data available for the designated devices. This data is subject to change without notice and revision of this document. For browser-based versions of this data sheet, refer to the left-hand navigation.

PACKAGING INFORMATION

Orderable Device	Status (1)	Package Type	Package Drawing	Pins	Package Qty	Eco Plan (2)	Lead/Ball Finish (6)	MSL Peak Temp (3)	Op Temp (°C)	Device Marking (4/5)	Samples
ADS1113IDGSR	ACTIVE	VSSOP	DGS	10	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR	-40 to 125	BROI	Samples
ADS1113IDGST	ACTIVE	VSSOP	DGS	10	250	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR	-40 to 125	BROI	Samples
ADS1113IRUGR	ACTIVE	X2QFN	RUG	10	3000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	N6J	Samples
ADS1113IRUGT	ACTIVE	X2QFN	RUG	10	250	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	N6J	Samples
ADS1114IDGSR	ACTIVE	VSSOP	DGS	10	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR	-40 to 125	BRNI	Samples
ADS1114IDGST	ACTIVE	VSSOP	DGS	10	250	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR	-40 to 125	BRNI	Samples
ADS1114IRUGR	ACTIVE	X2QFN	RUG	10	3000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	N5J	Samples
ADS1114IRUGT	ACTIVE	X2QFN	RUG	10	250	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	N5J	Samples
ADS1115IDGSR	ACTIVE	VSSOP	DGS	10	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR	-40 to 125	BOGI	Samples
ADS1115IDGST	ACTIVE	VSSOP	DGS	10	250	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-2-260C-1 YEAR	-40 to 125	BOGI	Samples
ADS1115IRUGR	ACTIVE	X2QFN	RUG	10	3000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	N4J	Samples
ADS1115IRUGT	ACTIVE	X2QFN	RUG	10	250	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM	-40 to 125	N4J	Samples

(1) The marketing status values are defined as follows:

ACTIVE: Product device recommended for new designs.

LIFEBUY: TI has announced that the device will be discontinued, and a lifetime-buy period is in effect.

NRND: Not recommended for new designs. Device is in production to support existing customers, but TI does not recommend using this part in a new design.

PREVIEW: Device has been announced but is not in production. Samples may or may not be available.

OBSOLETE: TI has discontinued the production of the device.

(2) **RoHS:** TI defines "RoHS" to mean semiconductor products that are compliant with the current EU RoHS requirements for all 10 RoHS substances, including the requirement that RoHS substance do not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered at high temperatures, "RoHS" products are suitable for use in specified lead-free processes. TI may reference these types of products as "Pb-Free".

RoHS Exempt: TI defines "RoHS Exempt" to mean products that contain lead but are compliant with EU RoHS pursuant to a specific EU RoHS exemption.



www.ti.com

PACKAGE OPTION ADDENDUM

22-Dec-2017

Green: TI defines "Green" to mean the content of Chlorine (Cl) and Bromine (Br) based flame retardants meet JS709B low halogen requirements of ≤ 1000 ppm threshold. Antimony trioxide based flame retardants must also meet the ≤ 1000 ppm threshold requirement.

⁽³⁾ MSL, Peak Temp. - The Moisture Sensitivity Level rating according to the JEDEC industry standard classifications, and peak solder temperature.

⁽⁴⁾ There may be additional marking, which relates to the logo, the lot trace code information, or the environmental category on the device.

⁽⁵⁾ Multiple Device Markings will be inside parentheses. Only one Device Marking contained in parentheses and separated by a "-" will appear on a device. If a line is indented then it is a continuation of the previous line and the two combined represent the entire Device Marking for that device.

⁽⁶⁾ Lead/Ball Finish - Orderable Devices may have multiple material finish options. Finish options are separated by a vertical ruled line. Lead/Ball Finish values may wrap to two lines if the finish value exceeds the maximum column width.

Important Information and Disclaimer: The information provided on this page represents TI's knowledge and belief as of the date that it is provided. TI bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. TI has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. TI and TI suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

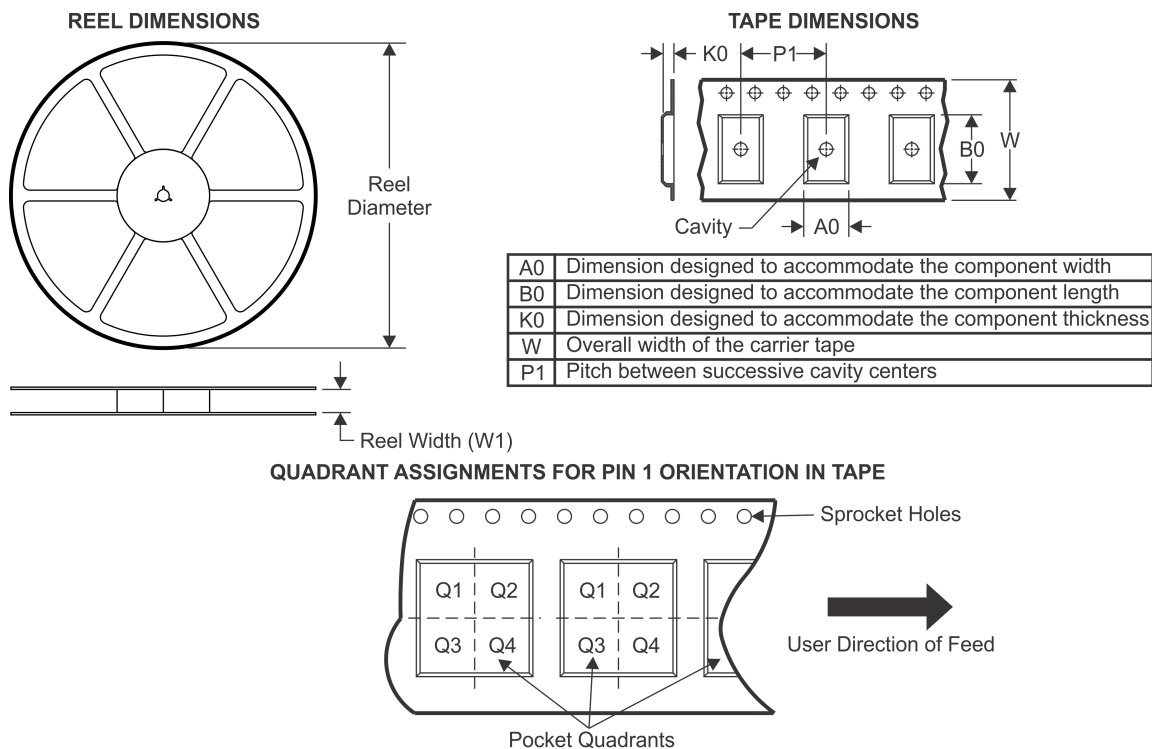
In no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI to Customer on an annual basis.

OTHER QUALIFIED VERSIONS OF ADS1113, ADS1114, ADS1115 :

- Automotive: [ADS1113-Q1](#), [ADS1114-Q1](#), [ADS1115-Q1](#)

NOTE: Qualified Version Definitions:

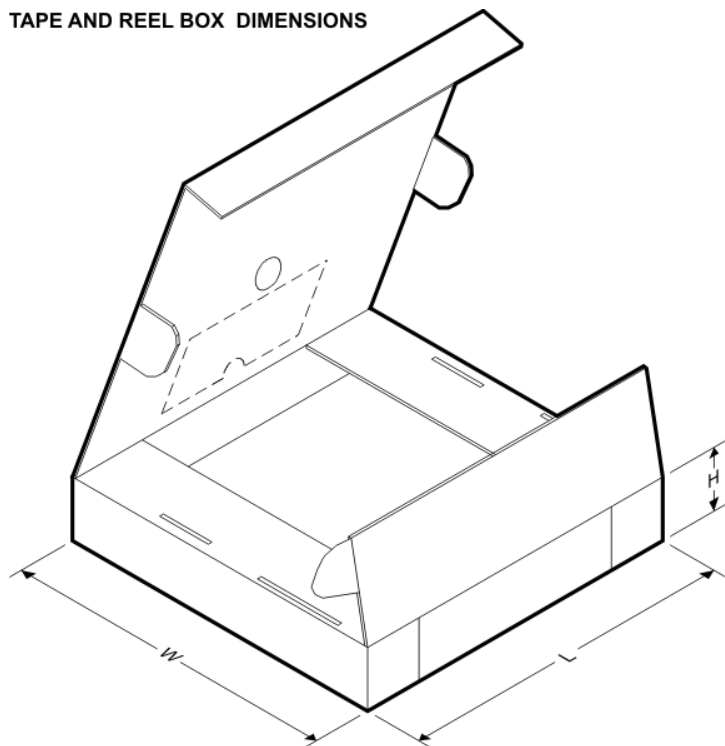
- Automotive - Q100 devices qualified for high-reliability automotive applications targeting zero defects

TAPE AND REEL INFORMATION


*All dimensions are nominal

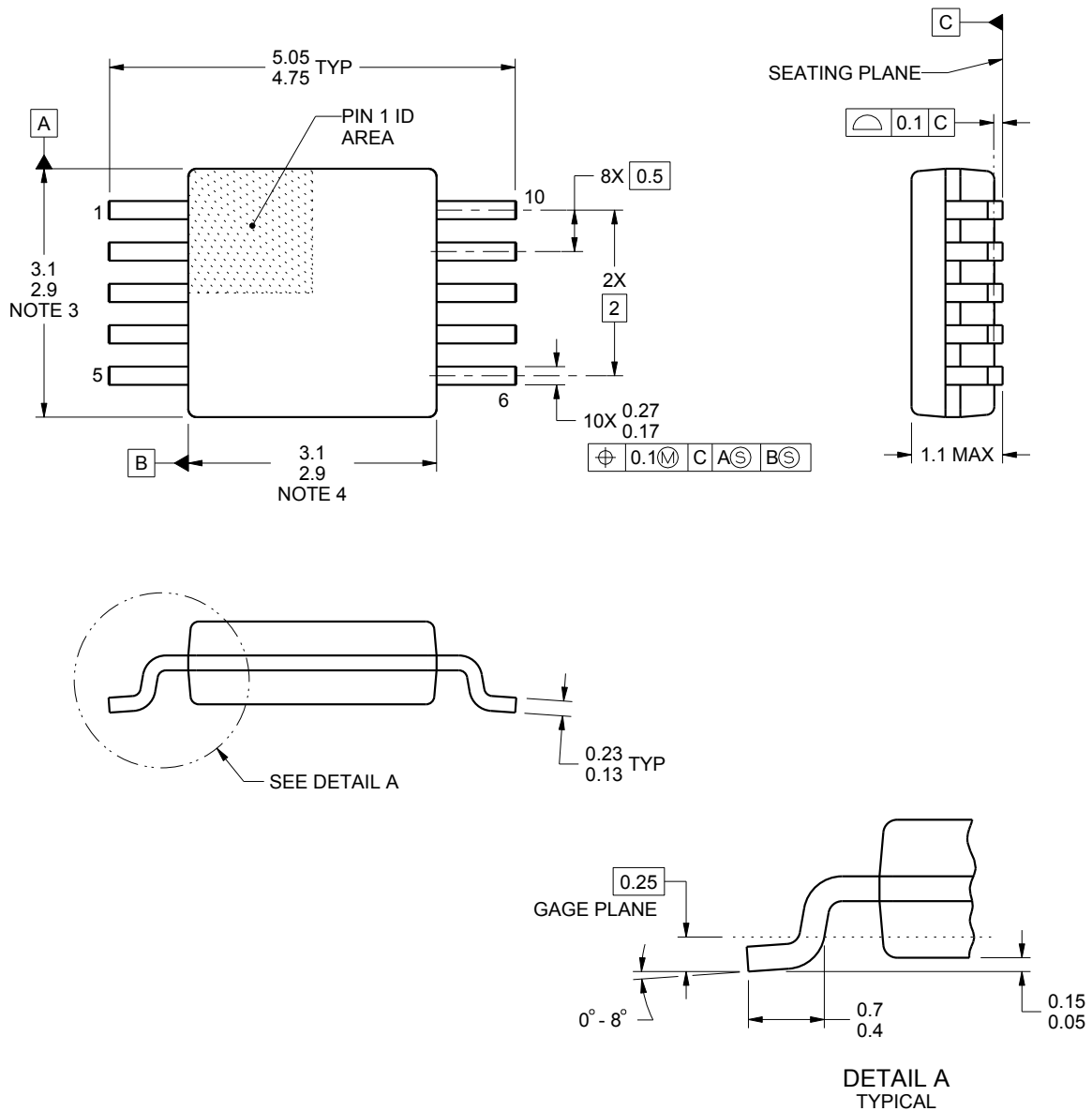
Device	Package Type	Package Drawing	Pins	SPQ	Reel Diameter (mm)	Reel Width W1 (mm)	A0 (mm)	B0 (mm)	K0 (mm)	P1 (mm)	W (mm)	Pin1 Quadrant
ADS1113IDGSR	VSSOP	DGS	10	2500	330.0	12.4	5.3	3.3	1.3	8.0	12.0	Q1
ADS1113IDGST	VSSOP	DGS	10	250	180.0	12.4	5.3	3.3	1.3	8.0	12.0	Q1
ADS1113IRUGR	X2QFN	RUG	10	3000	179.0	8.4	1.75	2.25	0.65	4.0	8.0	Q1
ADS1113IRUGT	X2QFN	RUG	10	250	179.0	8.4	1.75	2.25	0.65	4.0	8.0	Q1
ADS1114IDGSR	VSSOP	DGS	10	2500	330.0	12.4	5.3	3.3	1.3	8.0	12.0	Q1
ADS1114IDGST	VSSOP	DGS	10	250	180.0	12.4	5.3	3.3	1.3	8.0	12.0	Q1
ADS1114IRUGR	X2QFN	RUG	10	3000	179.0	8.4	1.75	2.25	0.65	4.0	8.0	Q1
ADS1114IRUGT	X2QFN	RUG	10	250	179.0	8.4	1.75	2.25	0.65	4.0	8.0	Q1
ADS1115IDGSR	VSSOP	DGS	10	2500	330.0	12.4	5.3	3.3	1.3	8.0	12.0	Q1
ADS1115IDGST	VSSOP	DGS	10	250	180.0	12.4	5.3	3.3	1.3	8.0	12.0	Q1
ADS1115IRUGR	X2QFN	RUG	10	3000	179.0	8.4	1.75	2.25	0.65	4.0	8.0	Q1
ADS1115IRUGT	X2QFN	RUG	10	250	179.0	8.4	1.75	2.25	0.65	4.0	8.0	Q1

TAPE AND REEL BOX DIMENSIONS



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Length (mm)	Width (mm)	Height (mm)
ADS1113IDGSR	VSSOP	DGS	10	2500	370.0	355.0	55.0
ADS1113IDGST	VSSOP	DGS	10	250	195.0	200.0	45.0
ADS1113IRUGR	X2QFN	RUG	10	3000	203.0	203.0	35.0
ADS1113IRUGT	X2QFN	RUG	10	250	203.0	203.0	35.0
ADS1114IDGSR	VSSOP	DGS	10	2500	370.0	355.0	55.0
ADS1114IDGST	VSSOP	DGS	10	250	195.0	200.0	45.0
ADS1114IRUGR	X2QFN	RUG	10	3000	203.0	203.0	35.0
ADS1114IRUGT	X2QFN	RUG	10	250	203.0	203.0	35.0
ADS1115IDGSR	VSSOP	DGS	10	2500	370.0	355.0	55.0
ADS1115IDGST	VSSOP	DGS	10	250	195.0	200.0	45.0
ADS1115IRUGR	X2QFN	RUG	10	3000	203.0	203.0	35.0
ADS1115IRUGT	X2QFN	RUG	10	250	203.0	203.0	35.0



4221984/A 05/2015

NOTES:

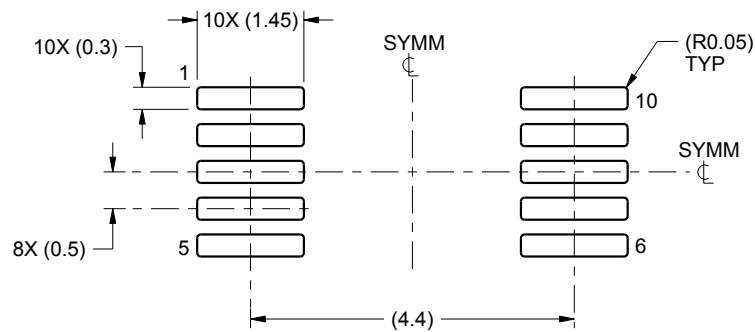
1. All linear dimensions are in millimeters. Any dimensions in parenthesis are for reference only. Dimensioning and tolerancing per ASME Y14.5M.
2. This drawing is subject to change without notice.
3. This dimension does not include mold flash, protrusions, or gate burrs. Mold flash, protrusions, or gate burrs shall not exceed 0.15 mm per side.
4. This dimension does not include interlead flash. Interlead flash shall not exceed 0.25 mm per side.
5. Reference JEDEC registration MO-187, variation BA.

EXAMPLE BOARD LAYOUT

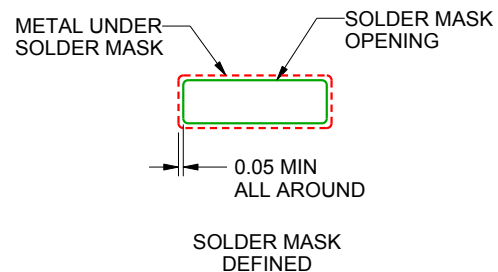
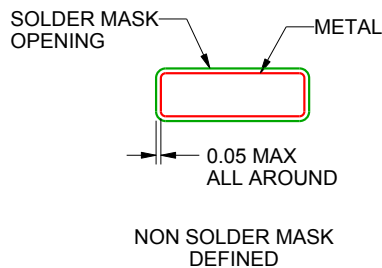
DGS0010A

VSSOP - 1.1 mm max height

SMALL OUTLINE PACKAGE



LAND PATTERN EXAMPLE
SCALE:10X



SOLDER MASK DETAILS
NOT TO SCALE

4221984/A 05/2015

NOTES: (continued)

6. Publication IPC-7351 may have alternate designs.

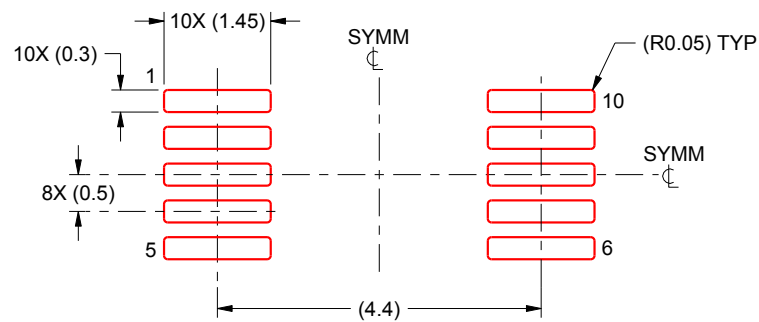
7. Solder mask tolerances between and around signal pads can vary based on board fabrication site.

EXAMPLE STENCIL DESIGN

DGS0010A

VSSOP - 1.1 mm max height

SMALL OUTLINE PACKAGE



SOLDER PASTE EXAMPLE
BASED ON 0.125 mm THICK STENCIL
SCALE:10X

4221984/A 05/2015

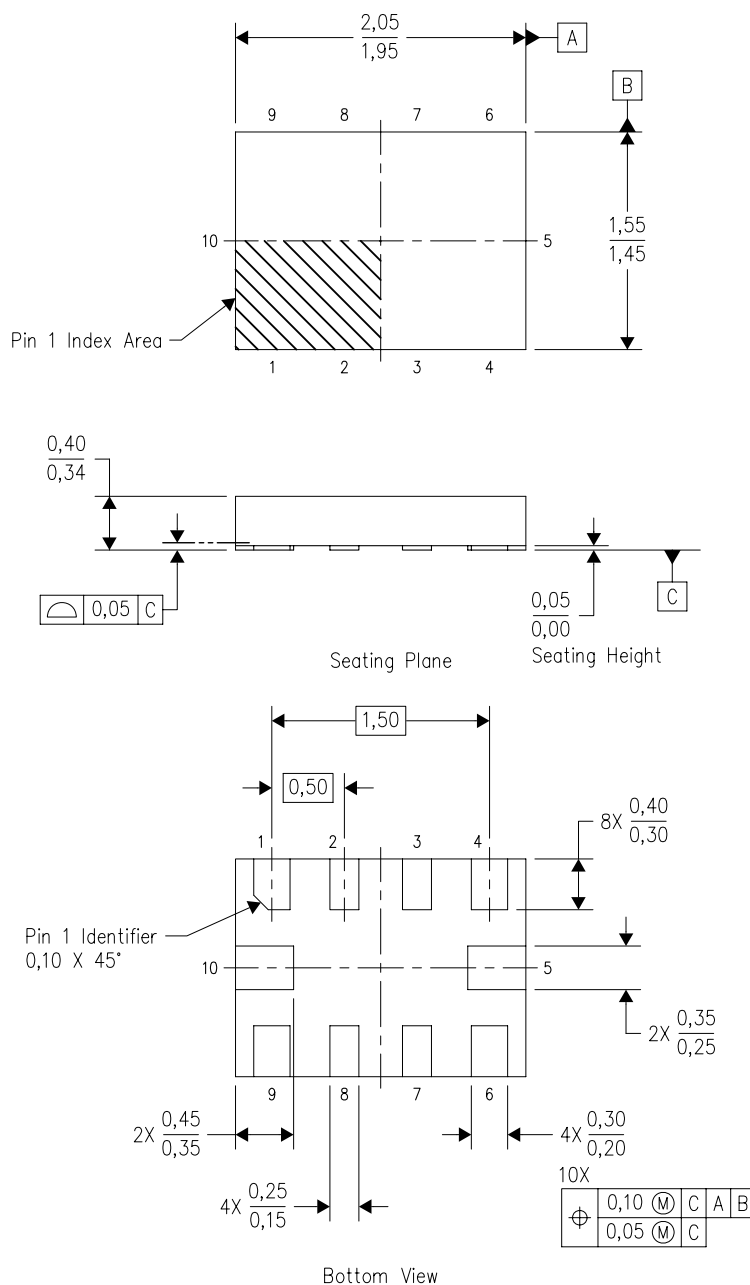
NOTES: (continued)

8. Laser cutting apertures with trapezoidal walls and rounded corners may offer better paste release. IPC-7525 may have alternate design recommendations.
9. Board assembly site may have different recommendations for stencil design.

MECHANICAL DATA

RUG (R-PQFP-N10)

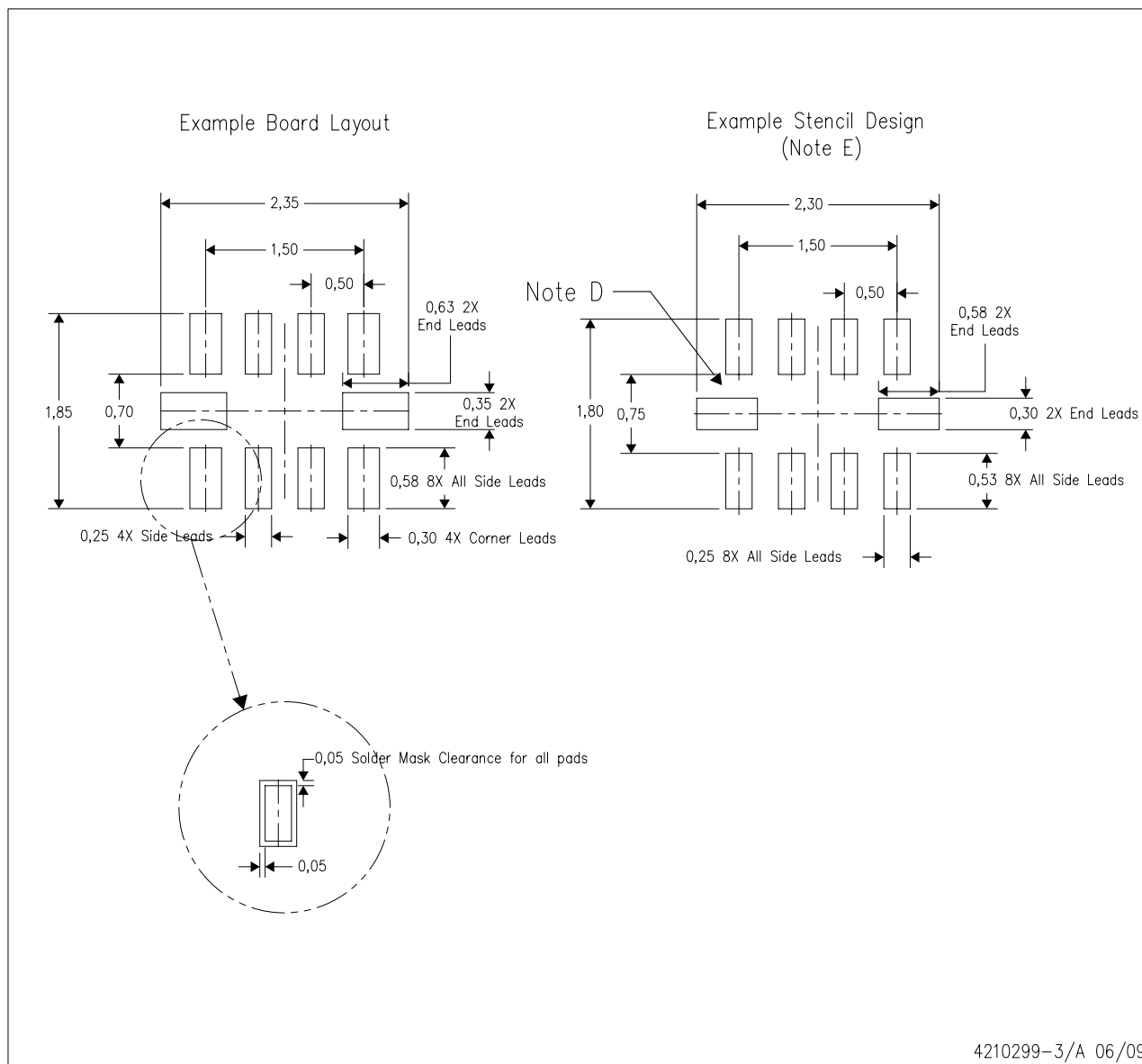
PLASTIC QUAD FLATPACK



4208528-3/B 04/2008

- NOTES:
- A. All linear dimensions are in millimeters. Dimensioning and tolerancing per ASME Y14.5M-1994.
 - B. This drawing is subject to change without notice.
 - C. QFN (Quad Flatpack No-Lead) package configuration.
 - D. This package complies to JEDEC MO-288 variation X2EFD.

RUG (R-PQFP-N10)



4210299-3/A 06/09

- NOTES:
- A. All linear dimensions are in millimeters.
 - B. This drawing is subject to change without notice.
 - C. Publication IPC-7351 is recommended for alternate designs.
 - D. Customers should contact their board fabrication site for minimum solder mask web tolerances between signal pads.
 - E. Maximum stencil thickness 0,127 mm (5 mils). All linear dimensions are in millimeters.
 - F. Laser cutting apertures with trapezoidal walls and also rounding corners will offer better paste release. Customers should contact their board assembly site for stencil design recommendations. Refer to IPC 7525 for stencil design considerations.
 - G. Side aperture dimensions over-print land for acceptable area ratio > 0.66. Customer may reduce side aperture dimensions if stencil manufacturing process allows for sufficient release at smaller opening.

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES "AS IS" AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to TI's Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products.

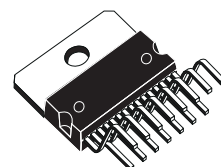
Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2018, Texas Instruments Incorporated

DUAL FULL-BRIDGE DRIVER

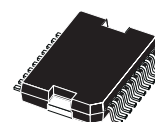
- OPERATING SUPPLY VOLTAGE UP TO 46 V
- TOTAL DC CURRENT UP TO 4 A
- LOW SATURATION VOLTAGE
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)

DESCRIPTION

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the con-



Multiwatt15

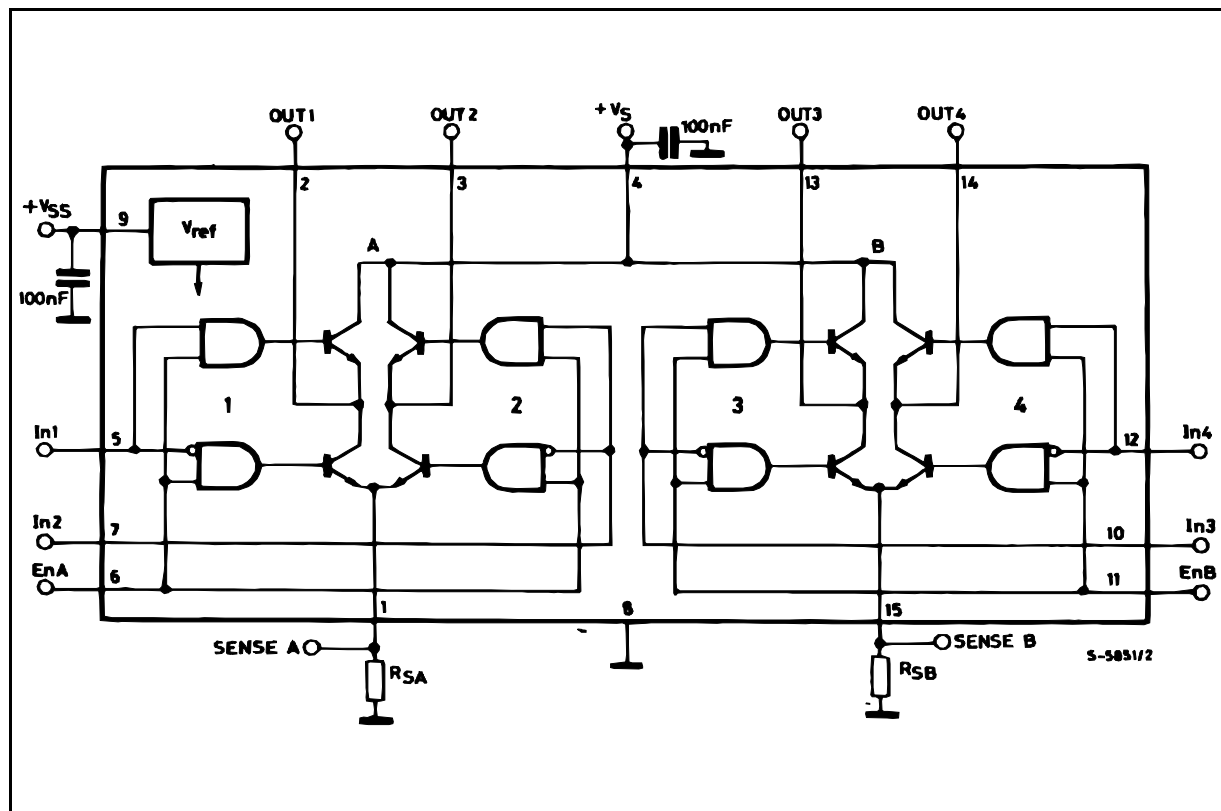


PowerSO20

ORDERING NUMBERS : L298N (Multiwatt Vert.)
L298HN (Multiwatt Horiz.)
L298P (PowerSO20)

nection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

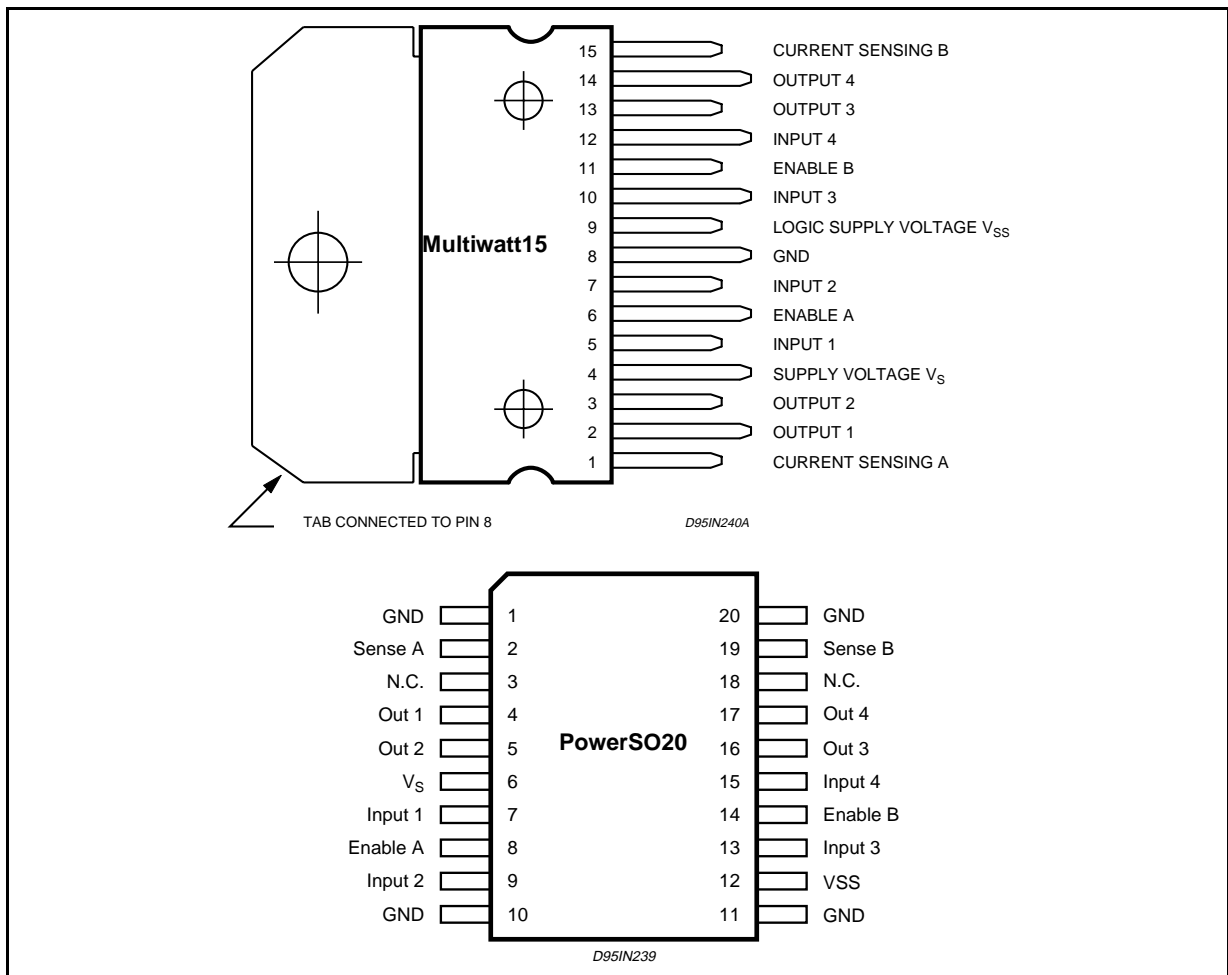
BLOCK DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Power Supply	50	V
V_{SS}	Logic Supply Voltage	7	V
V_I, V_{en}	Input and Enable Voltage	-0.3 to 7	V
I_O	Peak Output Current (each Channel)		
	– Non Repetitive ($t = 100\mu s$)	3	A
	– Repetitive (80% on –20% off; $t_{on} = 10ms$)	2.5	A
	– DC Operation	2	A
V_{sens}	Sensing Voltage	-1 to 2.3	V
P_{tot}	Total Power Dissipation ($T_{case} = 75^\circ C$)	25	W
T_{op}	Junction Operating Temperature	-25 to 130	$^\circ C$
T_{stg}, T_j	Storage and Junction Temperature	-40 to 150	$^\circ C$

PIN CONNECTIONS (top view)



THERMAL DATA

Symbol	Parameter		PowerSO20	Multiwatt15	Unit
$R_{th\ j-case}$	Thermal Resistance Junction-case	Max.	–	3	$^\circ C/W$
$R_{th\ j-amb}$	Thermal Resistance Junction-ambient	Max.	13 (*)	35	$^\circ C/W$

(*) Mounted on aluminum substrate

PIN FUNCTIONS (refer to the block diagram)

MW.15	PowerSO	Name	Function
1;15	2;19	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
2;3	4;5	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
4	6	V _S	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
5;7	7;9	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
6;11	8;14	Enable A; Enable B	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
8	1,10,11,20	GND	Ground.
9	12	V _{SS}	Supply Voltage for the Logic Blocks. A100nF capacitor must be connected between this pin and ground.
10; 12	13;15	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
13; 14	16;17	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.
–	3;18	N.C.	Not Connected

ELECTRICAL CHARACTERISTICS (V_S = 42V; V_{SS} = 5V, T_j = 25°C; unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V _S	Supply Voltage (pin 4)	Operative Condition	V _{IH} +2.5		46	V
V _{SS}	Logic Supply Voltage (pin 9)		4.5	5	7	V
I _S	Quiescent Supply Current (pin 4)	V _{en} = H; I _L = 0 V _i = L		13	22	mA
		V _i = H		50	70	mA
I _{SS}	Quiescent Current from V _{SS} (pin 9)	V _{en} = L V _i = X			4	mA
		V _{en} = H; I _L = 0 V _i = L		24	36	mA
		V _i = H		7	12	mA
		V _{en} = L V _i = X			6	mA
V _{iL}	Input Low Voltage (pins 5, 7, 10, 12)		–0.3		1.5	V
V _{iH}	Input High Voltage (pins 5, 7, 10, 12)		2.3		V _{SS}	V
I _{iL}	Low Voltage Input Current (pins 5, 7, 10, 12)	V _i = L			–10	μA
I _{iH}	High Voltage Input Current (pins 5, 7, 10, 12)	V _i = H ≤ V _{SS} –0.6V		30	100	μA
V _{en} = L	Enable Low Voltage (pins 6, 11)		–0.3		1.5	V
V _{en} = H	Enable High Voltage (pins 6, 11)		2.3		V _{SS}	V
I _{en} = L	Low Voltage Enable Current (pins 6, 11)	V _{en} = L			–10	μA
I _{en} = H	High Voltage Enable Current (pins 6, 11)	V _{en} = H ≤ V _{SS} –0.6V		30	100	μA
V _{CEsat} (H)	Source Saturation Voltage	I _L = 1A	0.95	1.35	1.7	V
		I _L = 2A		2	2.7	V
V _{CEsat} (L)	Sink Saturation Voltage	I _L = 1A (5)	0.85	1.2	1.6	V
		I _L = 2A (5)		1.7	2.3	V
V _{CEsat}	Total Drop	I _L = 1A (5)	1.80		3.2	V
		I _L = 2A (5)			4.9	V
V _{sens}	Sensing Voltage (pins 1, 15)		–1 (1)		2	V

ELECTRICAL CHARACTERISTICS (continued)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
$T_1 (V_i)$	Source Current Turn-off Delay	$0.5 V_i$ to $0.9 I_L$ (2); (4)		1.5		μs
$T_2 (V_i)$	Source Current Fall Time	$0.9 I_L$ to $0.1 I_L$ (2); (4)		0.2		μs
$T_3 (V_i)$	Source Current Turn-on Delay	$0.5 V_i$ to $0.1 I_L$ (2); (4)		2		μs
$T_4 (V_i)$	Source Current Rise Time	$0.1 I_L$ to $0.9 I_L$ (2); (4)		0.7		μs
$T_5 (V_i)$	Sink Current Turn-off Delay	$0.5 V_i$ to $0.9 I_L$ (3); (4)		0.7		μs
$T_6 (V_i)$	Sink Current Fall Time	$0.9 I_L$ to $0.1 I_L$ (3); (4)		0.25		μs
$T_7 (V_i)$	Sink Current Turn-on Delay	$0.5 V_i$ to $0.9 I_L$ (3); (4)		1.6		μs
$T_8 (V_i)$	Sink Current Rise Time	$0.1 I_L$ to $0.9 I_L$ (3); (4)		0.2		μs
$f_c (V_i)$	Commutation Frequency	$I_L = 2A$		25	40	KHz
$T_1 (V_{en})$	Source Current Turn-off Delay	$0.5 V_{en}$ to $0.9 I_L$ (2); (4)		3		μs
$T_2 (V_{en})$	Source Current Fall Time	$0.9 I_L$ to $0.1 I_L$ (2); (4)		1		μs
$T_3 (V_{en})$	Source Current Turn-on Delay	$0.5 V_{en}$ to $0.1 I_L$ (2); (4)		0.3		μs
$T_4 (V_{en})$	Source Current Rise Time	$0.1 I_L$ to $0.9 I_L$ (2); (4)		0.4		μs
$T_5 (V_{en})$	Sink Current Turn-off Delay	$0.5 V_{en}$ to $0.9 I_L$ (3); (4)		2.2		μs
$T_6 (V_{en})$	Sink Current Fall Time	$0.9 I_L$ to $0.1 I_L$ (3); (4)		0.35		μs
$T_7 (V_{en})$	Sink Current Turn-on Delay	$0.5 V_{en}$ to $0.9 I_L$ (3); (4)		0.25		μs
$T_8 (V_{en})$	Sink Current Rise Time	$0.1 I_L$ to $0.9 I_L$ (3); (4)		0.1		μs

1) Sensing voltage can be $-1 V$ for $t \leq 50 \mu s$; in steady state $V_{sens} \min \geq -0.5 V$.

2) See fig. 2.

3) See fig. 4.

4) The load must be a pure resistor.

Figure 1 : Typical Saturation Voltage vs. Output Current.

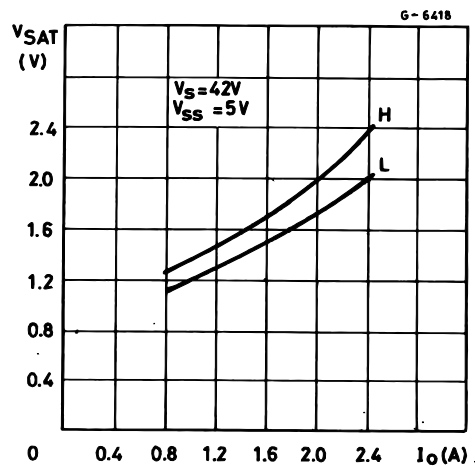
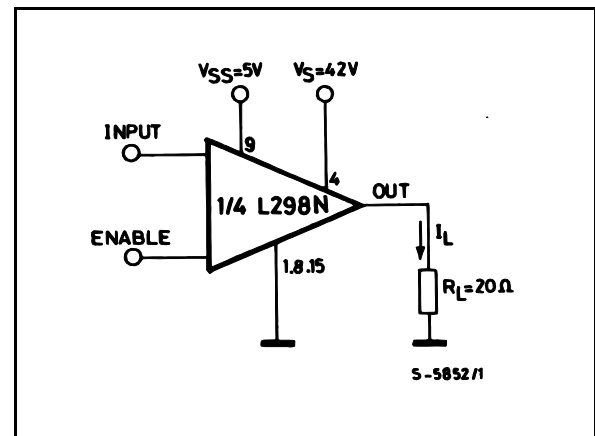
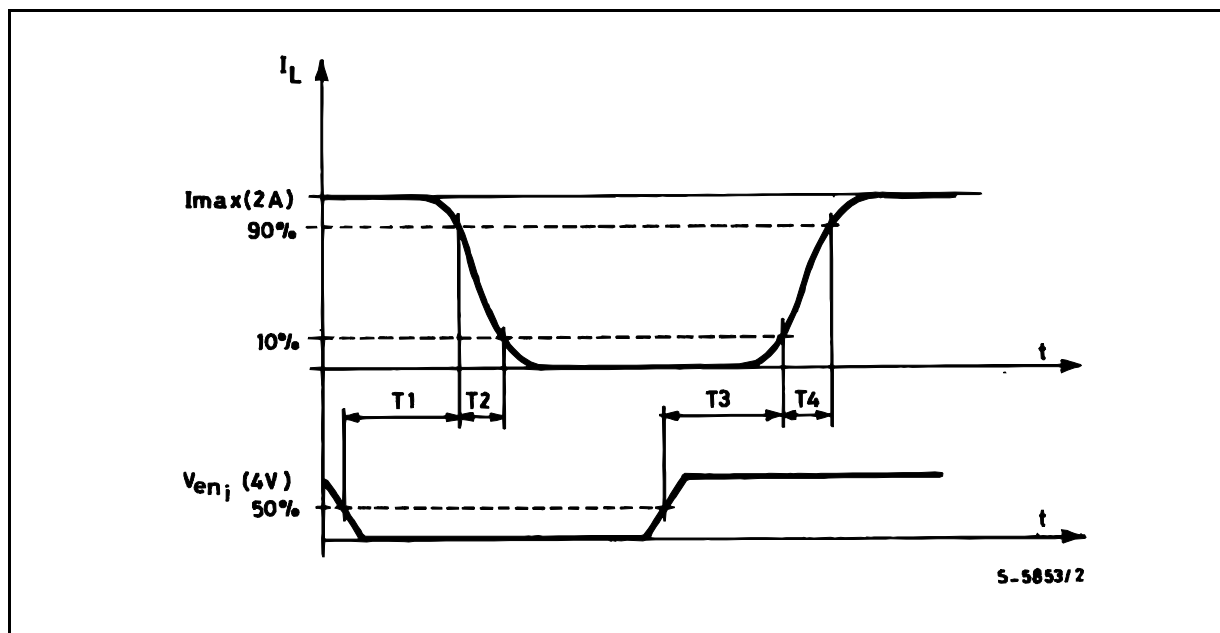
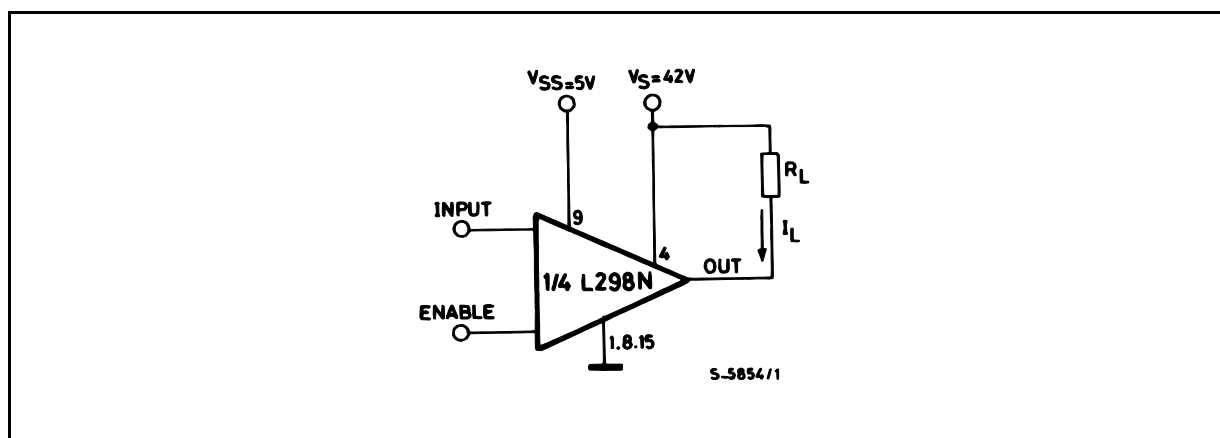


Figure 2 : Switching Times Test Circuits.



Note : For INPUT Switching, set EN = H
For ENABLE Switching, set IN = H

Figure 3 : Source Current Delay Times vs. Input or Enable Switching.**Figure 4 :** Switching Times Test Circuits.

Note : For INPUT Switching, set EN = H
 For ENABLE Switching, set IN = L

Figure 5 : Sink Current Delay Times vs. Input 0 V Enable Switching.

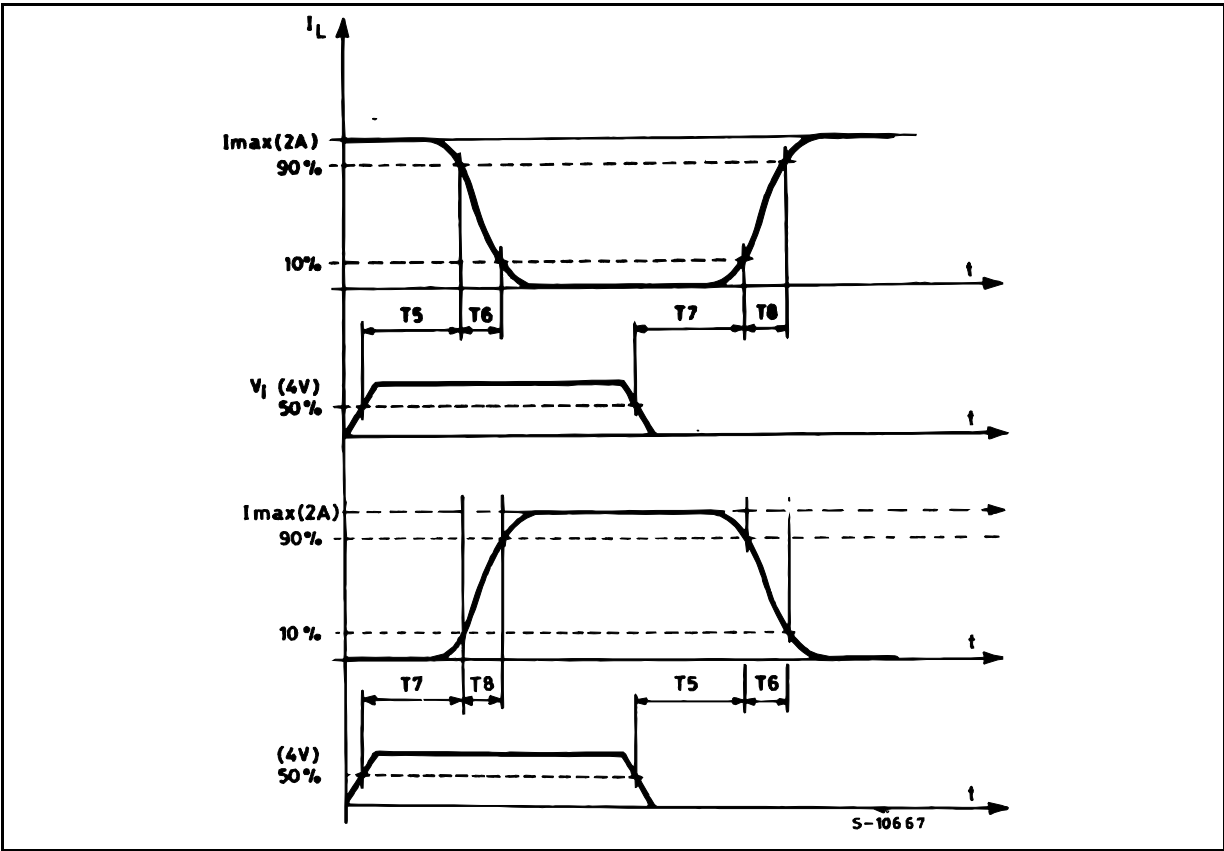


Figure 6 : Bidirectional DC Motor Control.

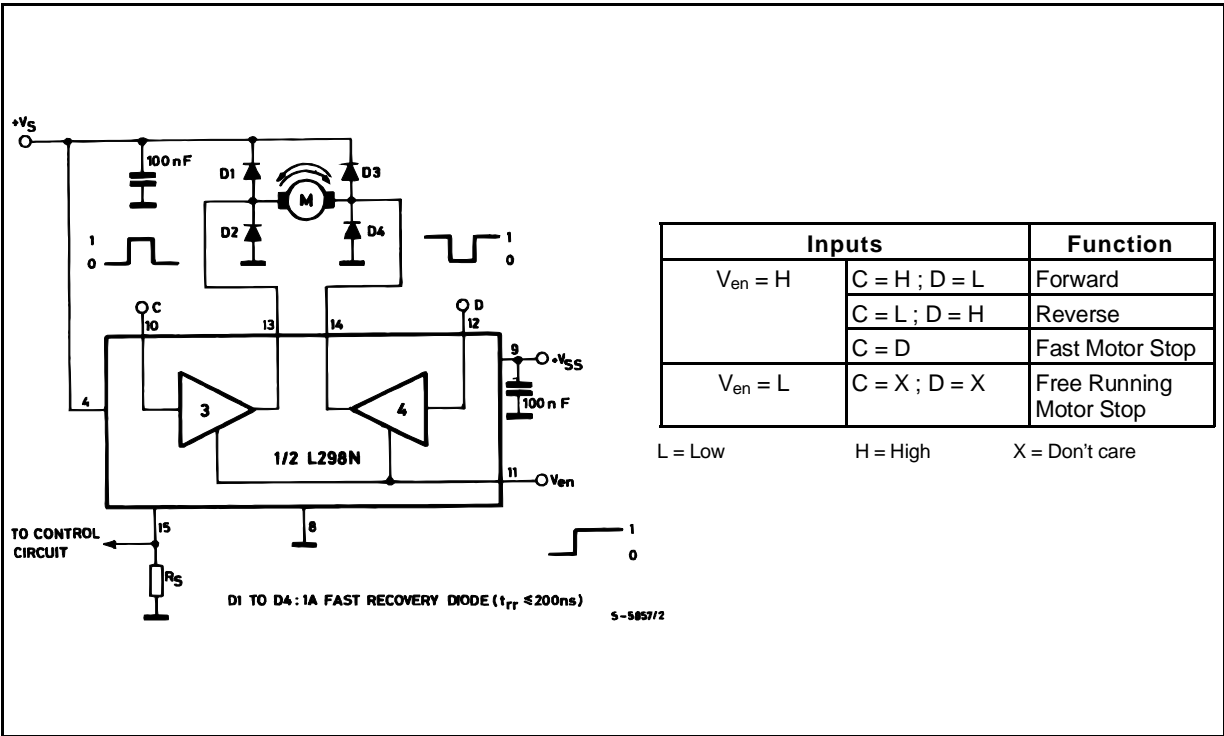
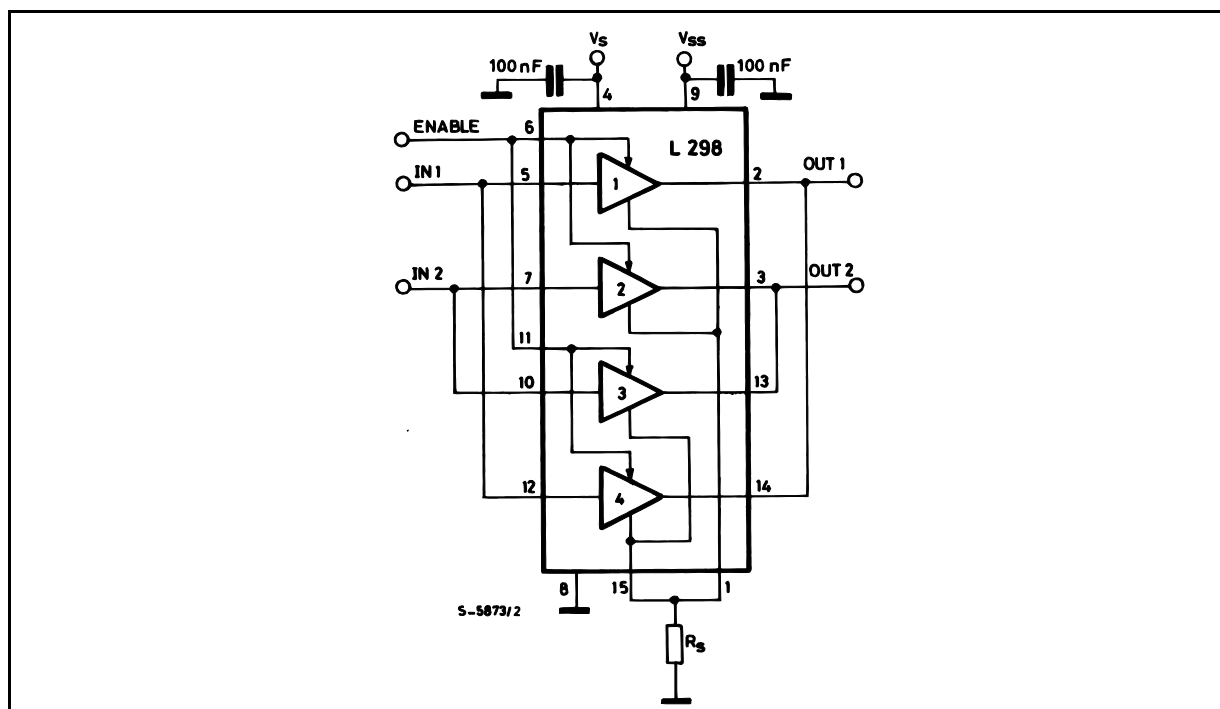


Figure 7 : For higher currents, outputs can be paralleled. Take care to parallel channel 1 with channel 4 and channel 2 with channel 3.



APPLICATION INFORMATION (Refer to the block diagram)

1.1. POWER OUTPUT STAGE

The L298 integrates two power output stages (A ; B). The power output stage is a bridge configuration and its outputs can drive an inductive load in common or differenzial mode, depending on the state of the inputs. The current that flows through the load comes out from the bridge at the sense output : an external resistor (R_{SA} ; R_{SB} .) allows to detect the intensity of this current.

1.2. INPUT STAGE

Each bridge is driven by means of four gates the input of which are $In1$; $In2$; EnA and $In3$; $In4$; EnB . The In inputs set the bridge state when The En input is high ; a low state of the En input inhibits the bridge. All the inputs are TTL compatible.

2. SUGGESTIONS

A non inductive capacitor, usually of 100 nF, must be foreseen between both V_S and V_{SS} , to ground, as near as possible to GND pin. When the large capacitor of the power supply is too far from the IC, a second smaller one must be foreseen near the L298.

The sense resistor, not of a wire wound type, must be grounded near the negative pole of V_S that must be near the GND pin of the I.C.

Each input must be connected to the source of the driving signals by means of a very short path.

Turn-On and Turn-Off : Before to Turn-ON the Supply Voltage and before to Turn it OFF, the Enable input must be driven to the Low state.

3. APPLICATIONS

Fig 6 shows a bidirectional DC motor control Schematic Diagram for which only one bridge is needed. The external bridge of diodes $D1$ to $D4$ is made by four fast recovery elements ($t_{rr} \leq 200$ nsec) that must be chosen of a V_F as low as possible at the worst case of the load current.

The sense output voltage can be used to control the current amplitude by chopping the inputs, or to provide overcurrent protection by switching low the enable input.

The brake function (Fast motor stop) requires that the Absolute Maximum Rating of 2 Amps must never be overcome.

When the repetitive peak current needed from the load is higher than 2 Amps, a paralleled configuration can be chosen (See Fig.7).

An external bridge of diodes are required when inductive loads are driven and when the inputs of the IC are chopped ; Schottky diodes would be preferred.

This solution can drive until 3 Amps In DC operation and until 3.5 Amps of a repetitive peak current.

On Fig 8 it is shown the driving of a two phase bipolar stepper motor ; the needed signals to drive the inputs of the L298 are generated, in this example, from the IC L297.

Fig 9 shows an example of P.C.B. designed for the application of Fig 8.

Figure 8 : Two Phase Bipolar Stepper Motor Circuit.

This circuit drives bipolar stepper motors with winding currents up to 2 A. The diodes are fast 2 A types.

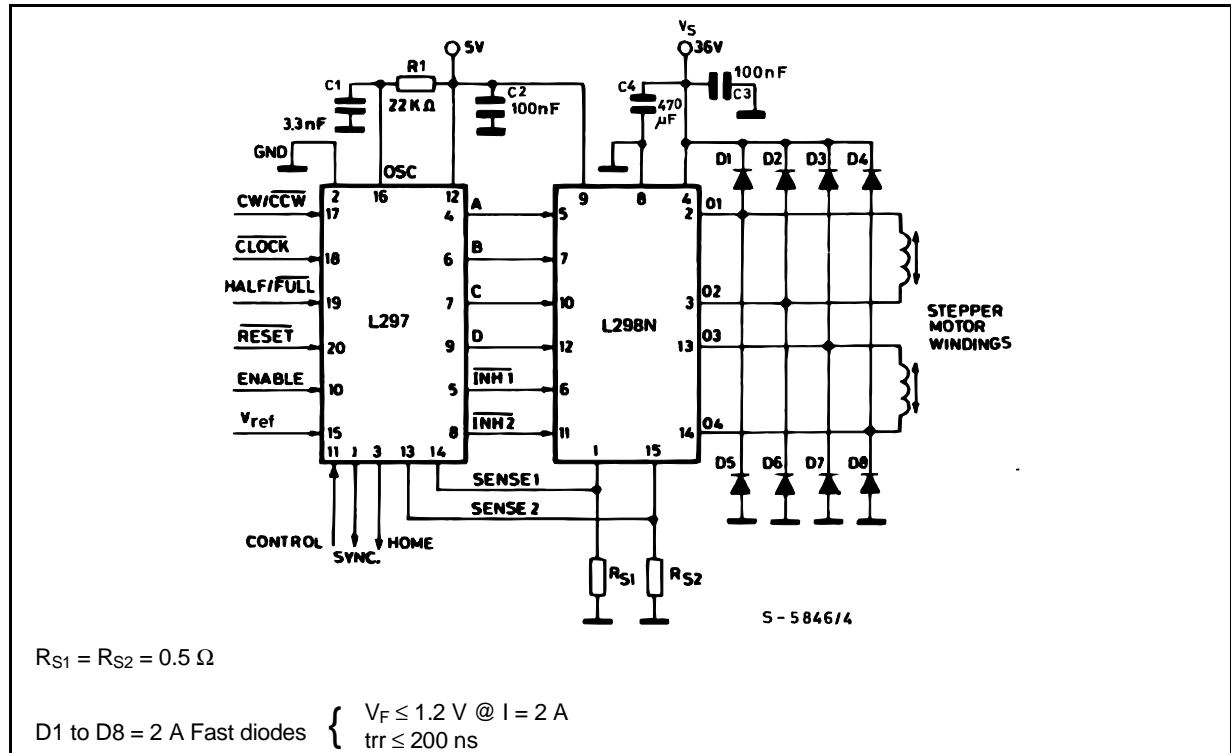


Fig 10 shows a second two phase bipolar stepper motor control circuit where the current is controlled by the I.C. L6506.

Figure 9 : Suggested Printed Circuit Board Layout for the Circuit of fig. 8 (1:1 scale).

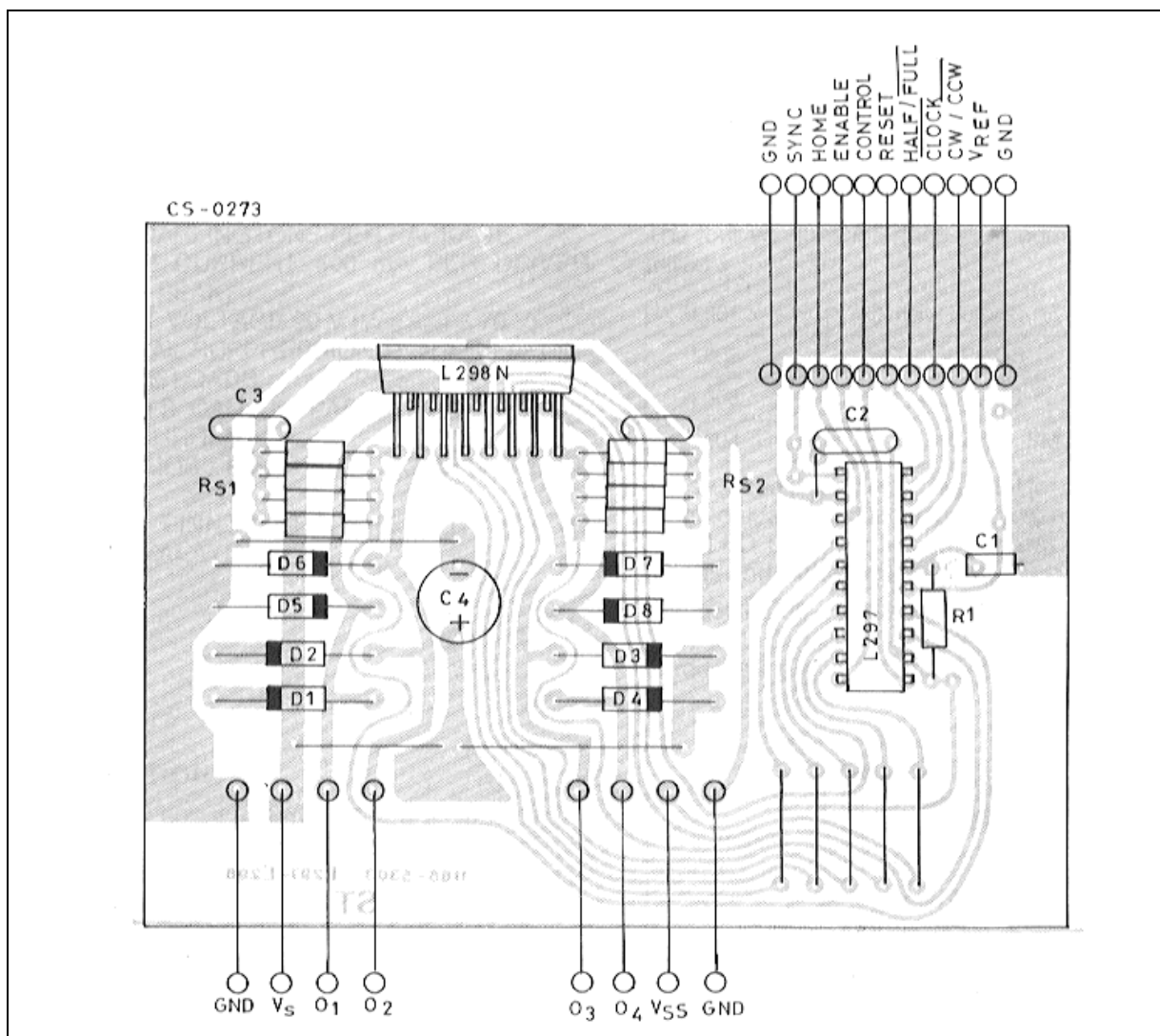
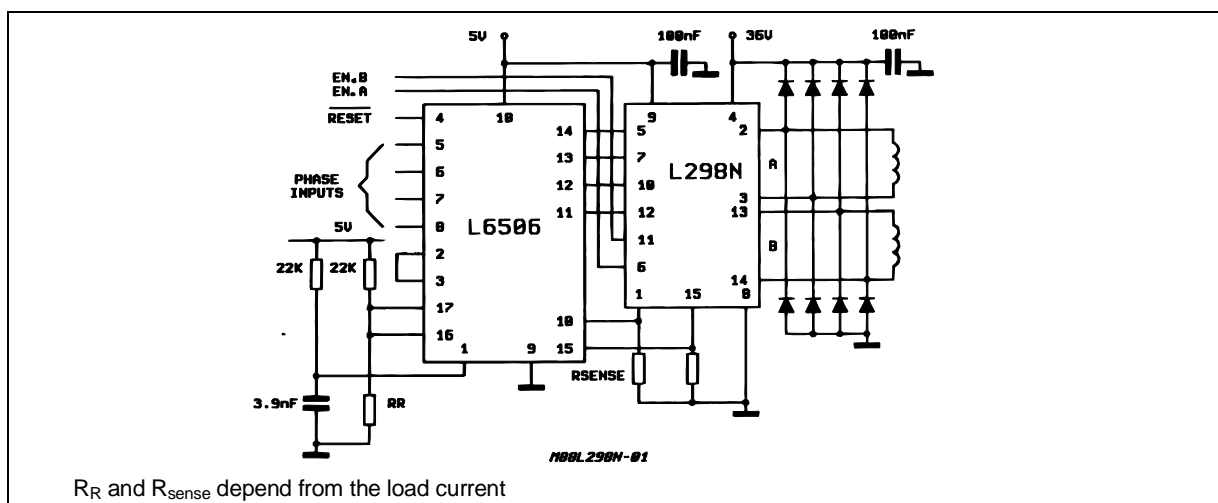


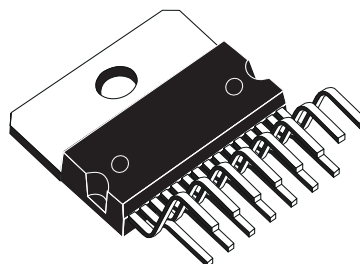
Figure 10 : Two Phase Bipolar Stepper Motor Control Circuit by Using the Current Controller L6506.



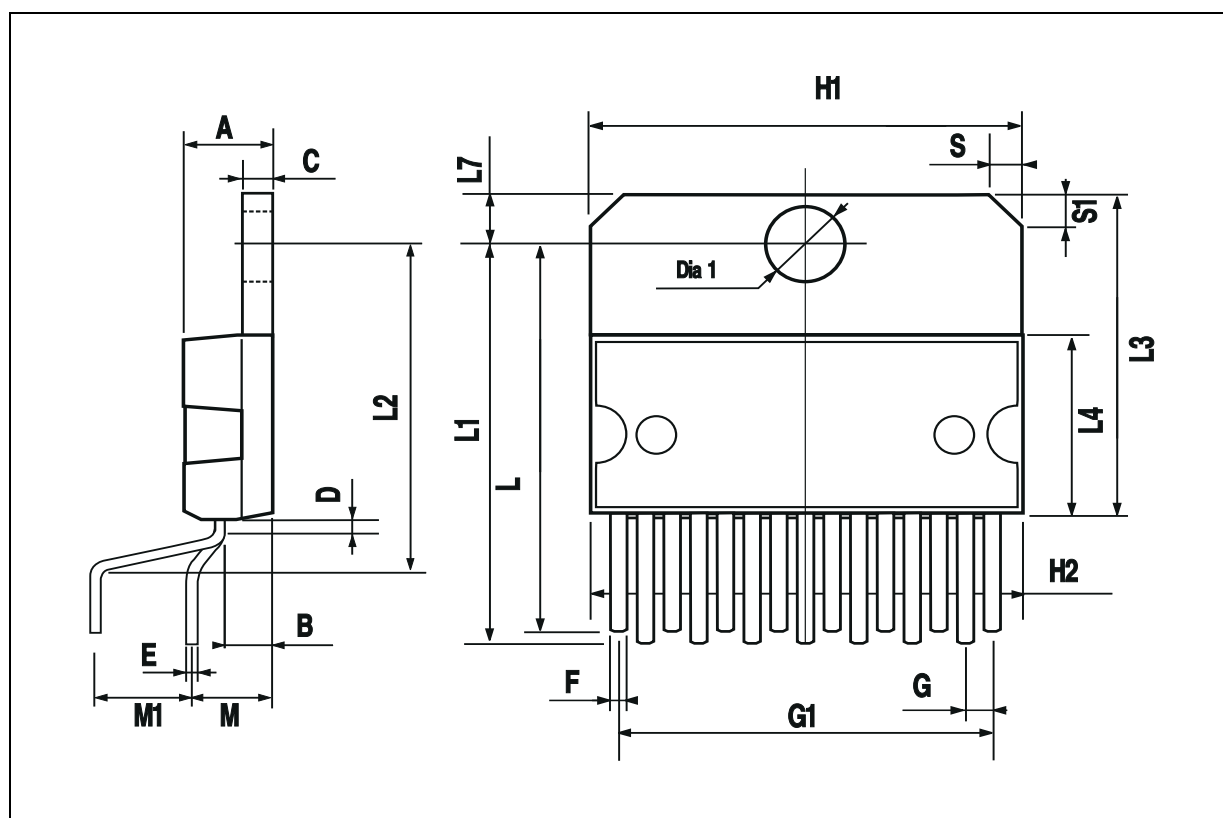
L298

DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			5			0.197
B			2.65			0.104
C			1.6			0.063
D		1			0.039	
E	0.49		0.55	0.019		0.022
F	0.66		0.75	0.026		0.030
G	1.02	1.27	1.52	0.040	0.050	0.060
G1	17.53	17.78	18.03	0.690	0.700	0.710
H1	19.6			0.772		
H2			20.2			0.795
L	21.9	22.2	22.5	0.862	0.874	0.886
L1	21.7	22.1	22.5	0.854	0.870	0.886
L2	17.65		18.1	0.695		0.713
L3	17.25	17.5	17.75	0.679	0.689	0.699
L4	10.3	10.7	10.9	0.406	0.421	0.429
L7	2.65		2.9	0.104		0.114
M	4.25	4.55	4.85	0.167	0.179	0.191
M1	4.63	5.08	5.53	0.182	0.200	0.218
S	1.9		2.6	0.075		0.102
S1	1.9		2.6	0.075		0.102
Dia1	3.65		3.85	0.144		0.152

OUTLINE AND MECHANICAL DATA

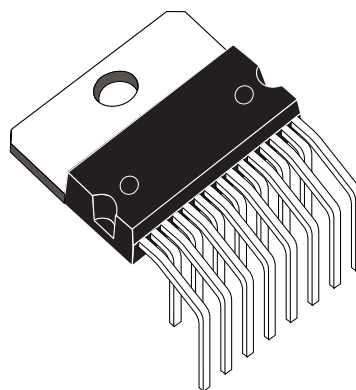


Multiwatt15 V

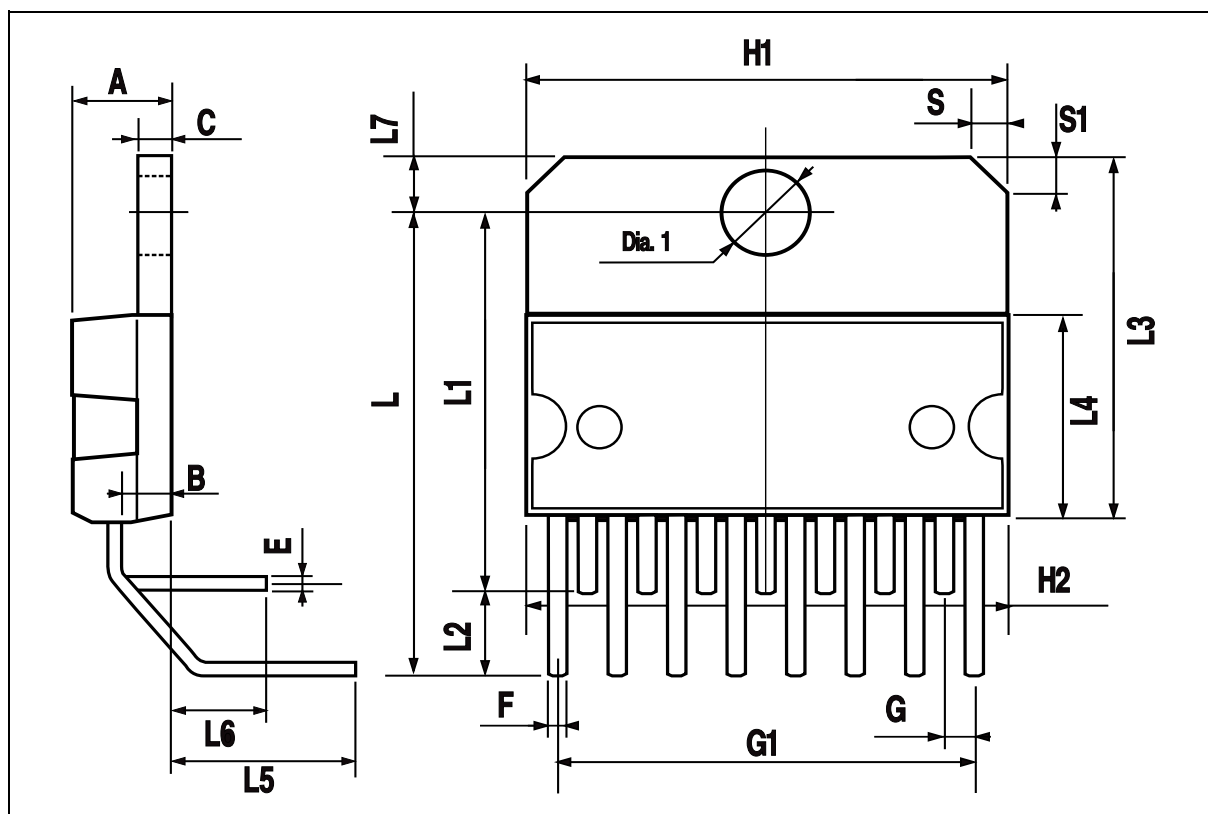


DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			5			0.197
B			2.65			0.104
C			1.6			0.063
E	0.49		0.55	0.019		0.022
F	0.66		0.75	0.026		0.030
G	1.14	1.27	1.4	0.045	0.050	0.055
G1	17.57	17.78	17.91	0.692	0.700	0.705
H1	19.6			0.772		
H2			20.2			0.795
L		20.57			0.810	
L1		18.03			0.710	
L2		2.54			0.100	
L3	17.25	17.5	17.75	0.679	0.689	0.699
L4	10.3	10.7	10.9	0.406	0.421	0.429
L5		5.28			0.208	
L6		2.38			0.094	
L7	2.65		2.9	0.104		0.114
S	1.9		2.6	0.075		0.102
S1	1.9		2.6	0.075		0.102
Dia1	3.65		3.85	0.144		0.152

OUTLINE AND MECHANICAL DATA



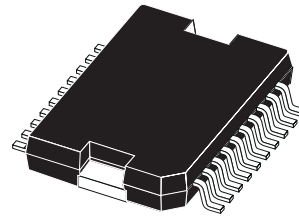
Multiwatt15 H



DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			3.6			0.142
a1	0.1		0.3	0.004		0.012
a2			3.3			0.130
a3	0		0.1	0.000		0.004
b	0.4		0.53	0.016		0.021
c	0.23		0.32	0.009		0.013
D (1)	15.8		16	0.622		0.630
D1	9.4		9.8	0.370		0.386
E	13.9		14.5	0.547		0.570
e		1.27			0.050	
e3		11.43			0.450	
E1 (1)	10.9		11.1	0.429		0.437
E2			2.9			0.114
E3	5.8		6.2	0.228		0.244
G	0		0.1	0.000		0.004
H	15.5		15.9	0.610		0.626
h			1.1			0.043
L	0.8		1.1	0.031		0.043
N	10° (max.)					
S	8° (max.)					
T		10			0.394	

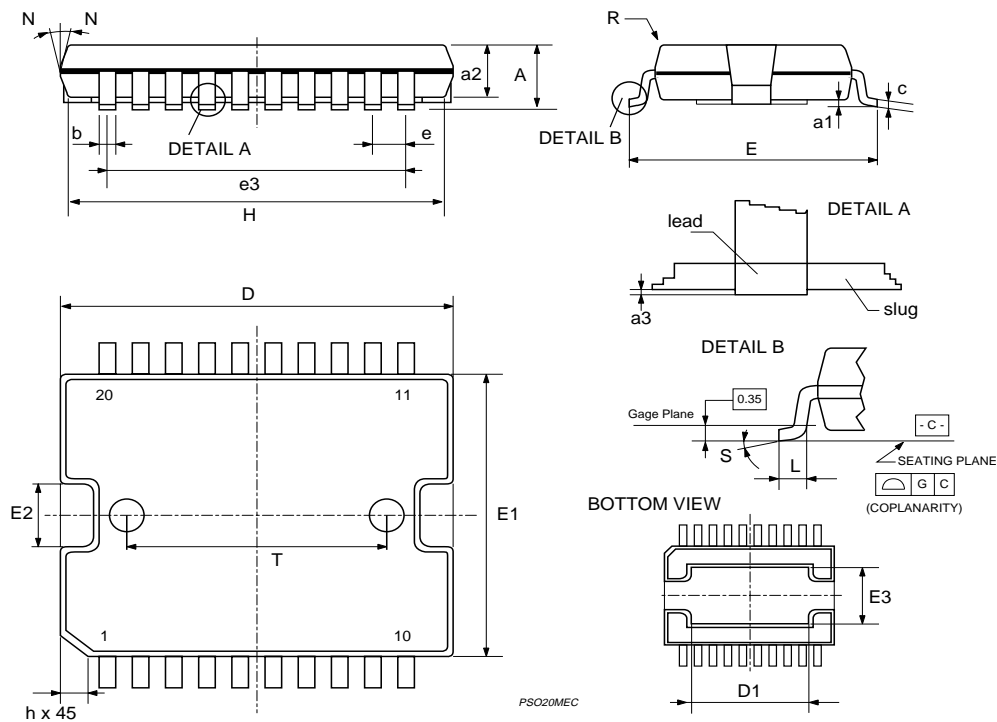
(1) "D and F" do not include mold flash or protrusions.
 - Mold flash or protrusions shall not exceed 0.15 mm (0.006").
 - Critical dimensions: "E", "G" and "a3"

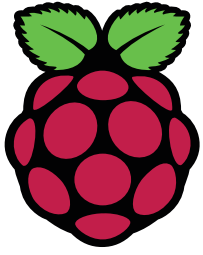
OUTLINE AND MECHANICAL DATA



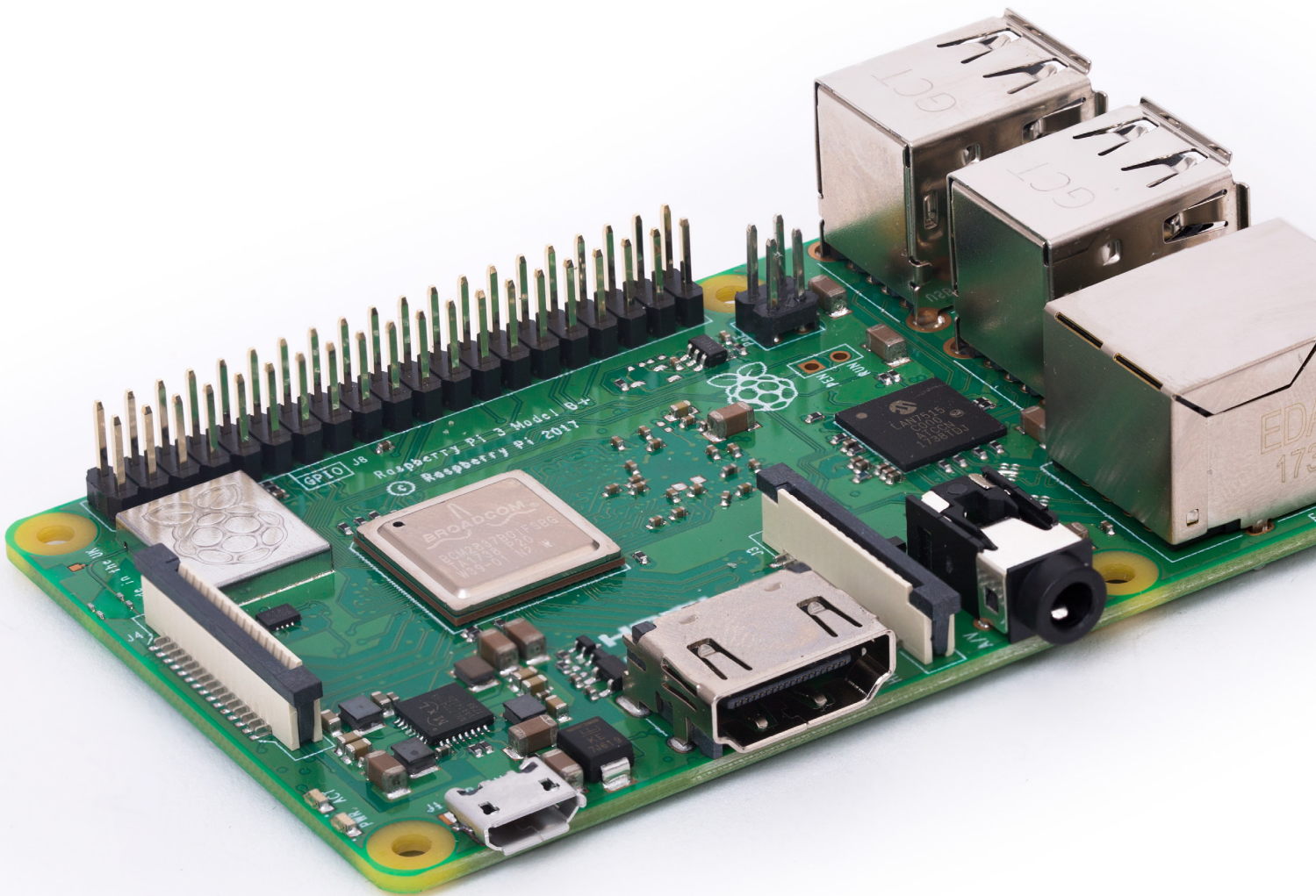
JEDEC MO-166

PowerSO20

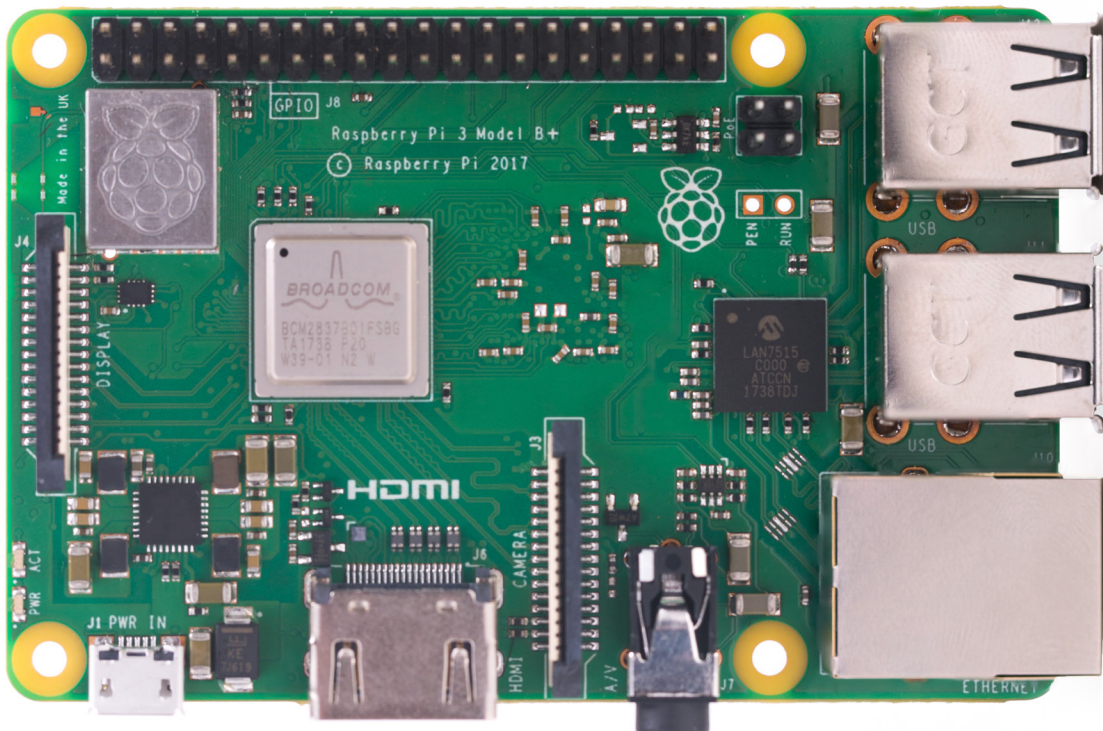




Raspberry Pi 3 Model B+



Overview



The Raspberry Pi 3 Model B+ is the latest product in the Raspberry Pi 3 range, boasting a 64-bit quad core processor running at 1.4GHz, dual-band 2.4GHz and 5GHz wireless LAN, Bluetooth 4.2/BLE, faster Ethernet, and PoE capability via a separate PoE HAT

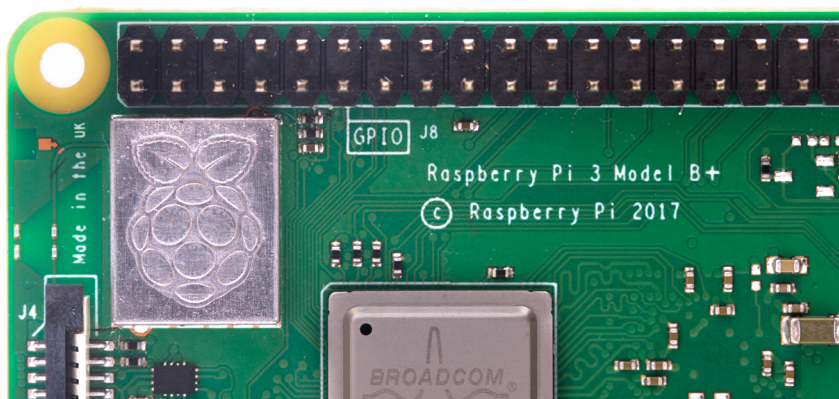
The dual-band wireless LAN comes with modular compliance certification, allowing the board to be designed into end products with significantly reduced wireless LAN compliance testing, improving both cost and time to market.

The Raspberry Pi 3 Model B+ maintains the same mechanical footprint as both the Raspberry Pi 2 Model B and the Raspberry Pi 3 Model B.

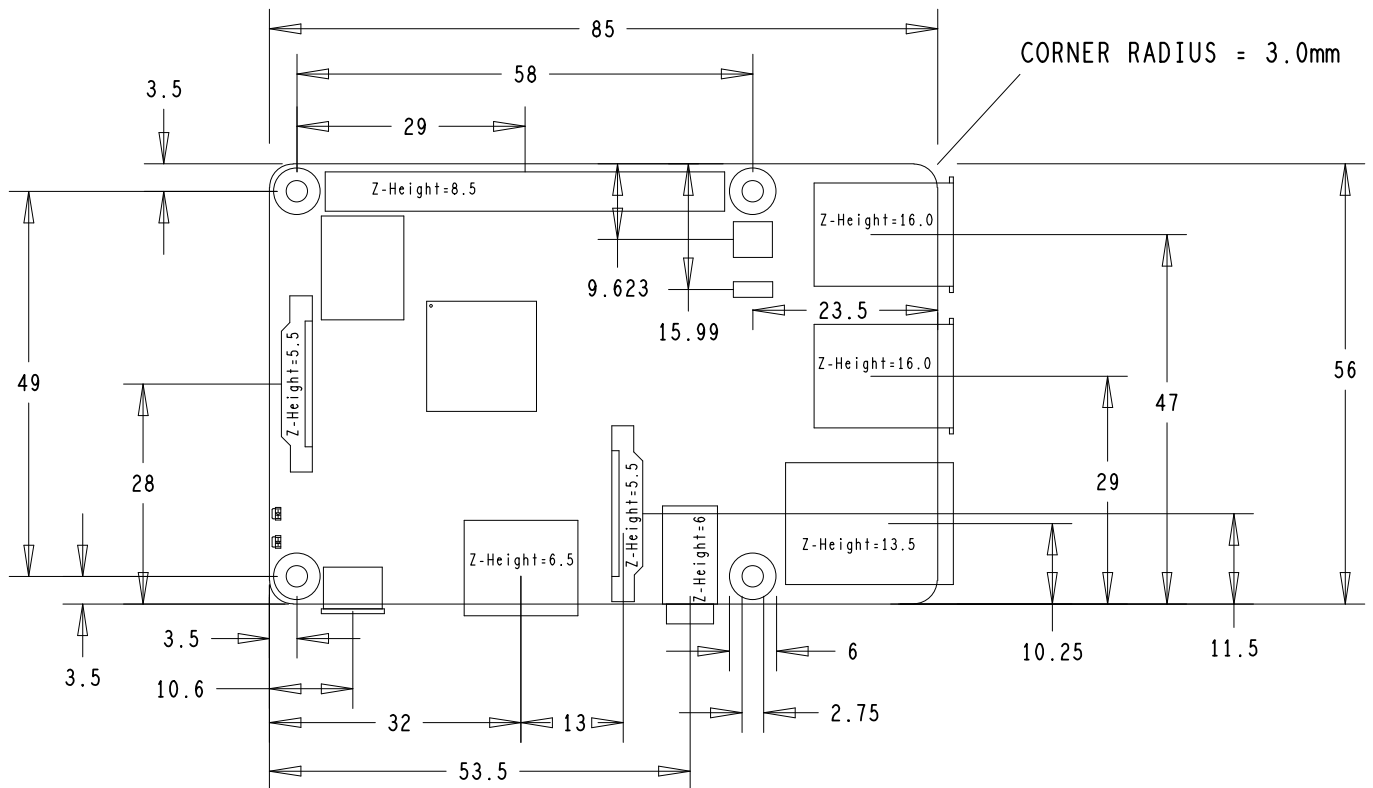


Specifications

Processor:	Broadcom BCM2837B0, Cortex-A53 64-bit SoC @ 1.4GHz
Memory:	1GB LPDDR2 SDRAM
Connectivity:	<ul style="list-style-type: none">■ 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE■ Gigabit Ethernet over USB 2.0 (maximum throughput 300Mbps)■ 4 × USB 2.0 ports
Access:	Extended 40-pin GPIO header
Video & sound:	<ul style="list-style-type: none">■ 1 × full size HDMI■ MIPI DSI display port■ MIPI CSI camera port■ 4 pole stereo output and composite video port
Multimedia:	H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0 graphics
SD card support:	Micro SD format for loading operating system and data storage
Input power:	<ul style="list-style-type: none">■ 5V/2.5A DC via micro USB connector■ 5V DC via GPIO header■ Power over Ethernet (PoE)–enabled (requires separate PoE HAT)
Environment:	Operating temperature, 0–50 °C
Compliance:	For a full list of local and regional product approvals, please visit www.raspberrypi.org/products/raspberry-pi-3-model-b+
Production lifetime:	The Raspberry Pi 3 Model B+ will remain in production until at least January 2023.



Physical specifications



Warnings

- This product should only be connected to an external power supply rated at 5V/2.5A DC. Any external power supply used with the Raspberry Pi 3 Model B+ shall comply with relevant regulations and standards applicable in the country of intended use.
- This product should be operated in a well-ventilated environment and, if used inside a case, the case should not be covered.
- Whilst in use, this product should be placed on a stable, flat, non-conductive surface and should not be contacted by conductive items.
- The connection of incompatible devices to the GPIO connection may affect compliance, result in damage to the unit, and invalidate the warranty.
- All peripherals used with this product should comply with relevant standards for the country of use and be marked accordingly to ensure that safety and performance requirements are met. These articles include but are not limited to keyboards, monitors, and mice when used in conjunction with the Raspberry Pi.
- The cables and connectors of all peripherals used with this product must have adequate insulation so that relevant safety requirements are met.

Safety instructions

To avoid malfunction of or damage to this product, please observe the following:

- Do not expose to water or moisture, or place on a conductive surface whilst in operation.
- Do not expose to heat from any source; the Raspberry Pi 3 Model B+ is designed for reliable operation at normal ambient temperatures.
- Take care whilst handling to avoid mechanical or electrical damage to the printed circuit board and connectors.
- Whilst it is powered, avoid handling the printed circuit board, or only handle it by the edges to minimise the risk of electrostatic discharge damage.



Escuela Superior de Ingeniería
Calle Alberto Aguilera 25
28015 Madrid, España
www.comillas.edu
Tel: (+34) 91 542 28 00
E-mail: ويا@comillas.edu